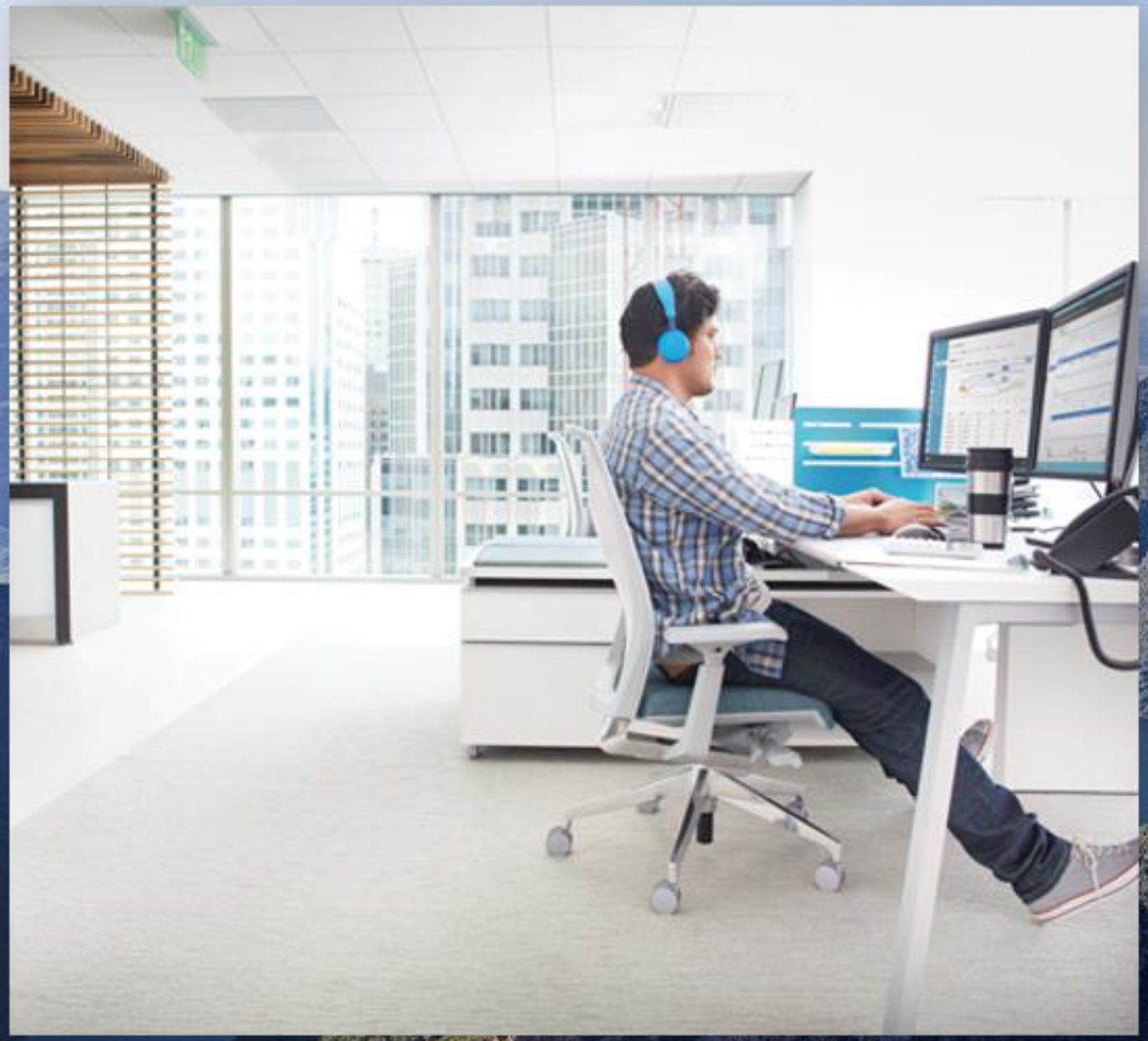
# PowerApps Control \ Components Framework介绍

Jinyu Xu | GPS CSA

## Overview

- Introducing PCF
- Developing PCF
- Testing and debugging PCF
- Deploying PCF



# Introducing PowerApps Component Framework

# PowerApps Component Framework

*跨PowerApps 三类应用程序的统一框架，可提供标准化体验和更丰富的通用控件集*

## Visualize

Field Data
Dataset Data
Unbound Controls

## Extend

Metadata Driven
Configurable
Reusable

## Portable

Solution Aware
Controls and config
Responsive

# PowerApps Component Framework

- PCF是 PowerApps/ D365 Customer Engagement 的可扩展平台，允许在应用程序的深处配置控件。
- 控件可以替换Forms上的列、View、Forms或Dashboard上的Subgrid。
- 可以根据数据类型配置控件。
- 这意味着一个控件可以应用于多个表的相同数据类型或视图的任何列。
- 在 Nodejs 之上以 React 原生构建，提供丰富的开发体验和视觉刺激的控件

# PCF控件和Web resource的区别

以前，为了改变表单或视图上元素的外观和感觉，开发人员不得不从头开始创建 Html Web 资源以彻底检查 UI 的某些部分。

Html web 资源的问题是：

- 与后面页面上的元素没有原生互操作性
- 无法在多个表单、元素、视图上重用代码
- 需要大量工作才能使用较新的编程模型，例如 Angular 或 React
- 他们总是看起来与使用的应用程序脱节

这些问题都已通过 PCF Controls 得到解决！

# PCF控件的优势

- 访问一组丰富的框架 API，这些 API 公开了组件生命周期管理、上下文数据和元数据等功能
- 通过 Web API、实用程序和数据格式化方法、摄像头、位置和麦克风等设备功能以及对话框、查找和整页渲染等易于调用的 UX 元素无缝访问服务器
- 支持现代网络实践
- 性能优化
- 可重用性
- 将所有文件zip到一个solution文件中

Power Apps component framework API reference

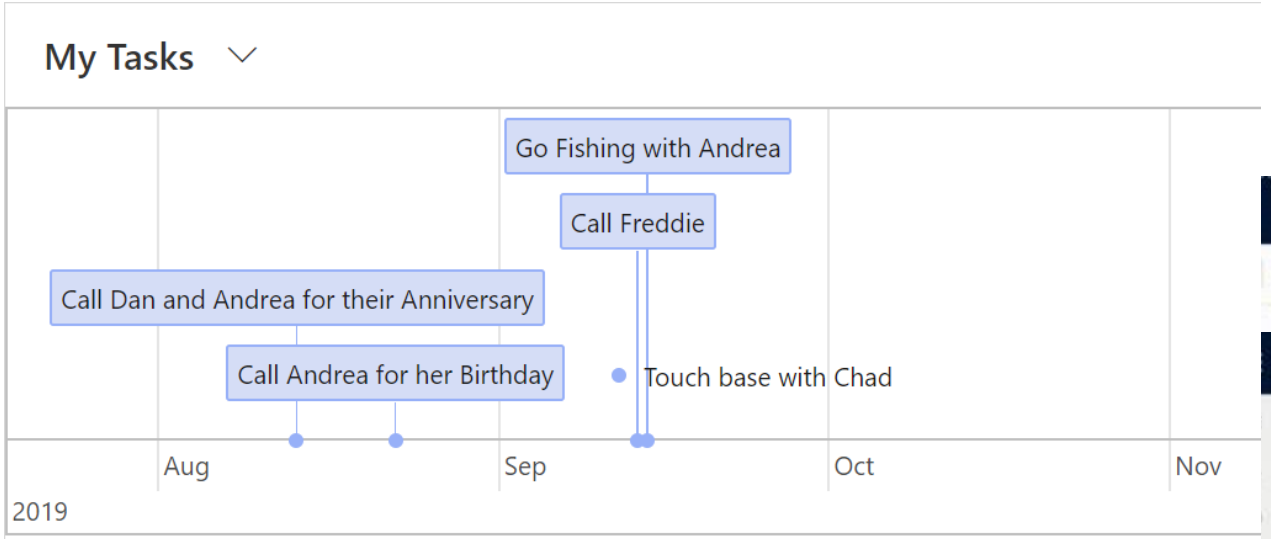
# PCF控件的能力

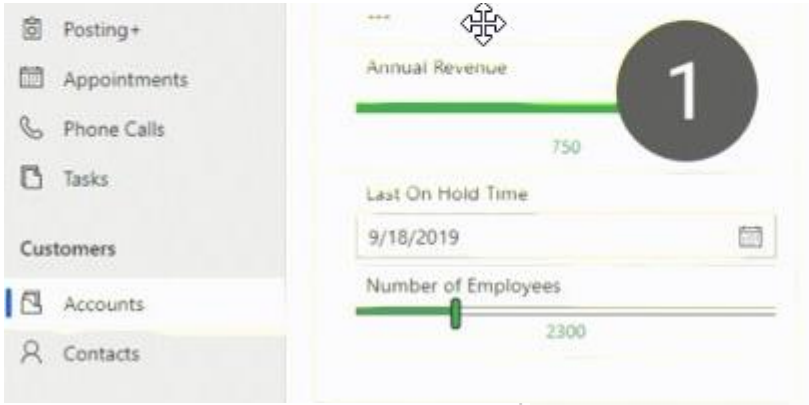
- PCF 控件的一般功能包括能够为表单或视图上的列提供不同的 UI。
- 创建后，可以根据需要为尽可能多的列和视图配置相同的控件。
- 适用于 PowerApps 三种应用模型

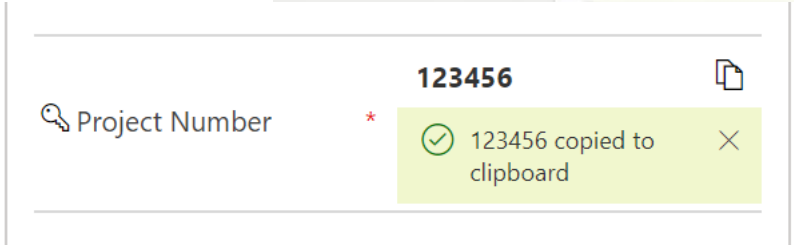
**举例:**

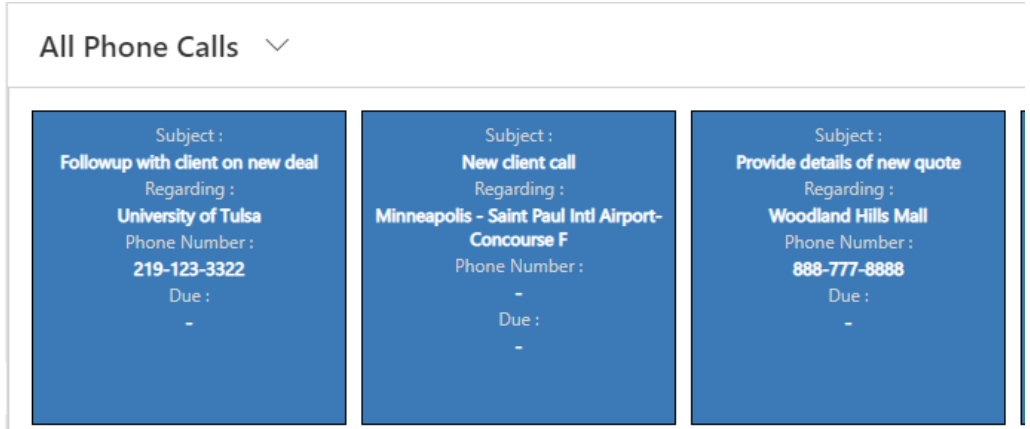
- Form
  - Component replaces a column on a form
    - Custom Date Picker
    - Linear Slider for numeric columns
    - Custom Phone Number Formatting
- Data-set
  - Component replaces a view or subgrid
    - Addresses On a Map
    - Custom Editable grid
    - Show data differently

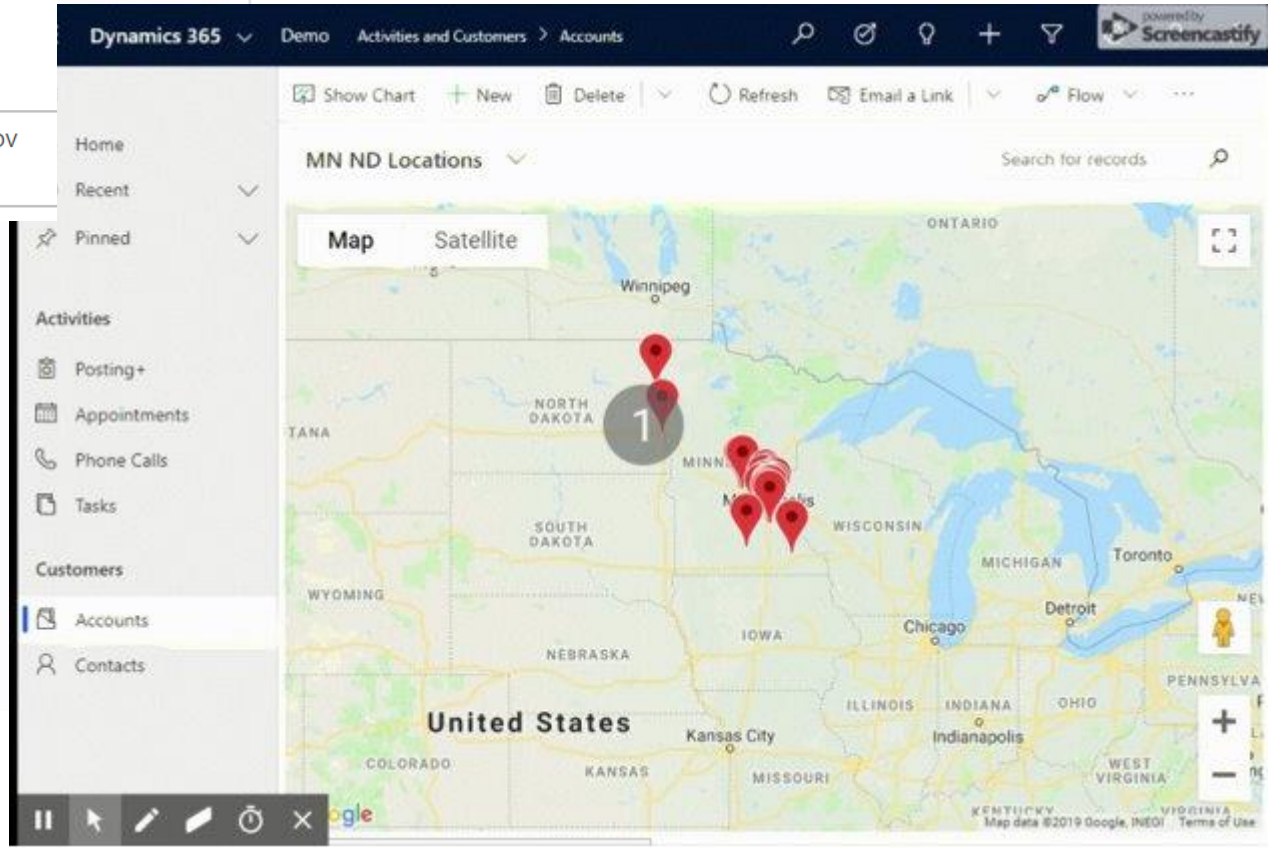
# PCF实例

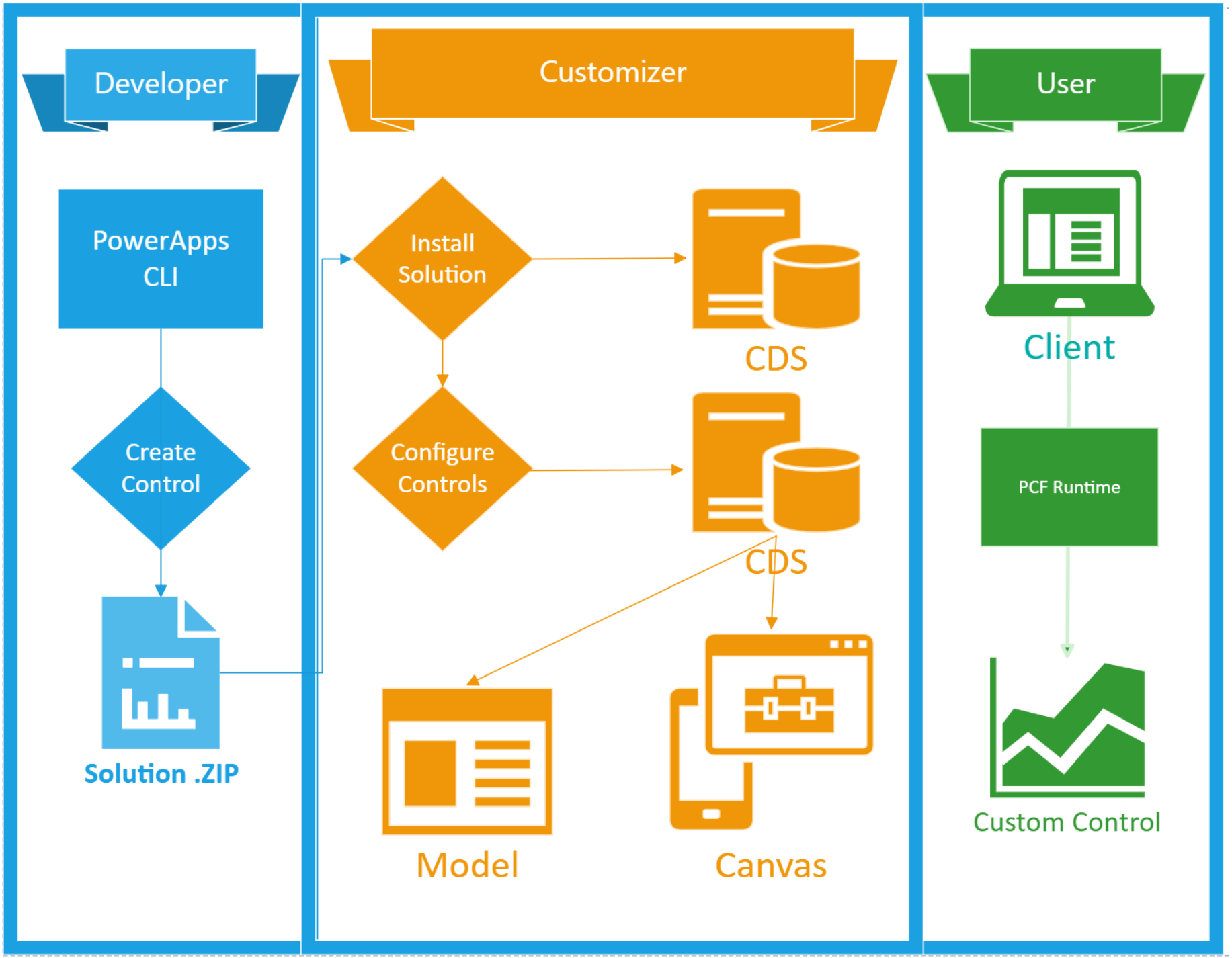

My Tasks
Go Fishing with Andrea
Call Freddie
Call Dan and Andrea for their Anniversary
Call Andrea for her Birthday
Touch base with Chad
Aug
Sep
Oct
Nov
2019


Posting+
Appointments
Phone Calls
Tasks
Customers
Accounts
Contacts
Annual Revenue
750
Last On Hold Time
9/18/2019
Number of Employees
2300


Project Number
*
123456
123456 copied to clipboard


All Phone Calls
Subject :
Followup with client on new deal
Regarding :
University of Tulsa
Phone Number :
219-123-3322
Due :
-
Subject :
New client call
Regarding :
Minneapolis - Saint Paul Intl Airport-Concourse F
Phone Number :
-
Due :
-
Subject :
Provide details of new quote
Regarding :
Woodland Hills Mall
Phone Number :
888-777-8888
Due :
-


Add to
Dynamics 365
Demo
Activities and Customers
Accounts
Screencastify
Show Chart
New
Delete
Refresh
Email a Link
Flow
Home
Recent
Pinned
Activities
Posting+
Appointments
Phone Calls
Tasks
Customers
Accounts
Contacts
MN ND Locations
Search for records
Map
Satellite
Winnipeg
ONTARIO
NORTH DAKOTA
SOUTH DAKOTA
WISCONSIN
MICHIGAN
Toronto
Detroit
Chicago
WYOMING
IOWA
NEBRASKA
ILLINOIS
INDIANA
OHIO
Indianapolis
United States
Kansas City
COLORADO
KANSAS
MISSOURI
WEST VIRGINIA


Developer
Customizer
User
PowerApps CLI
Create Control
Solution .ZIP
Install Solution
Configure Controls
CDS
CDS
Model
Canvas
Client
PCF Runtime
Custom Control

# Developing PowerApps Component Framework

# PCF控件开发的前提条件

- 选择你喜欢的 IDE
  - Visual Studio, Visual Studio Code, Notepad, Eclipse
    - Visual Studio Code is Free! https://code.visualstudio.com/download
- 提前安装
  - Install these before starting your first control
    - .NET Framework Developer pack 4.6.2
      - https://dotnet.microsoft.com/download/dotnet-framework/net462
    - NodeJs/NPM
      - https://www.npmjs.com/get-npm
    - PowerApps CLI (command line interface)
      - https://aka.ms/PowerAppsCLI

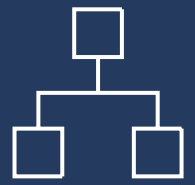
# PCF 由什么组成?

**Manifest File**

Control definition
- Name
- Version
- Properties
- Resource files

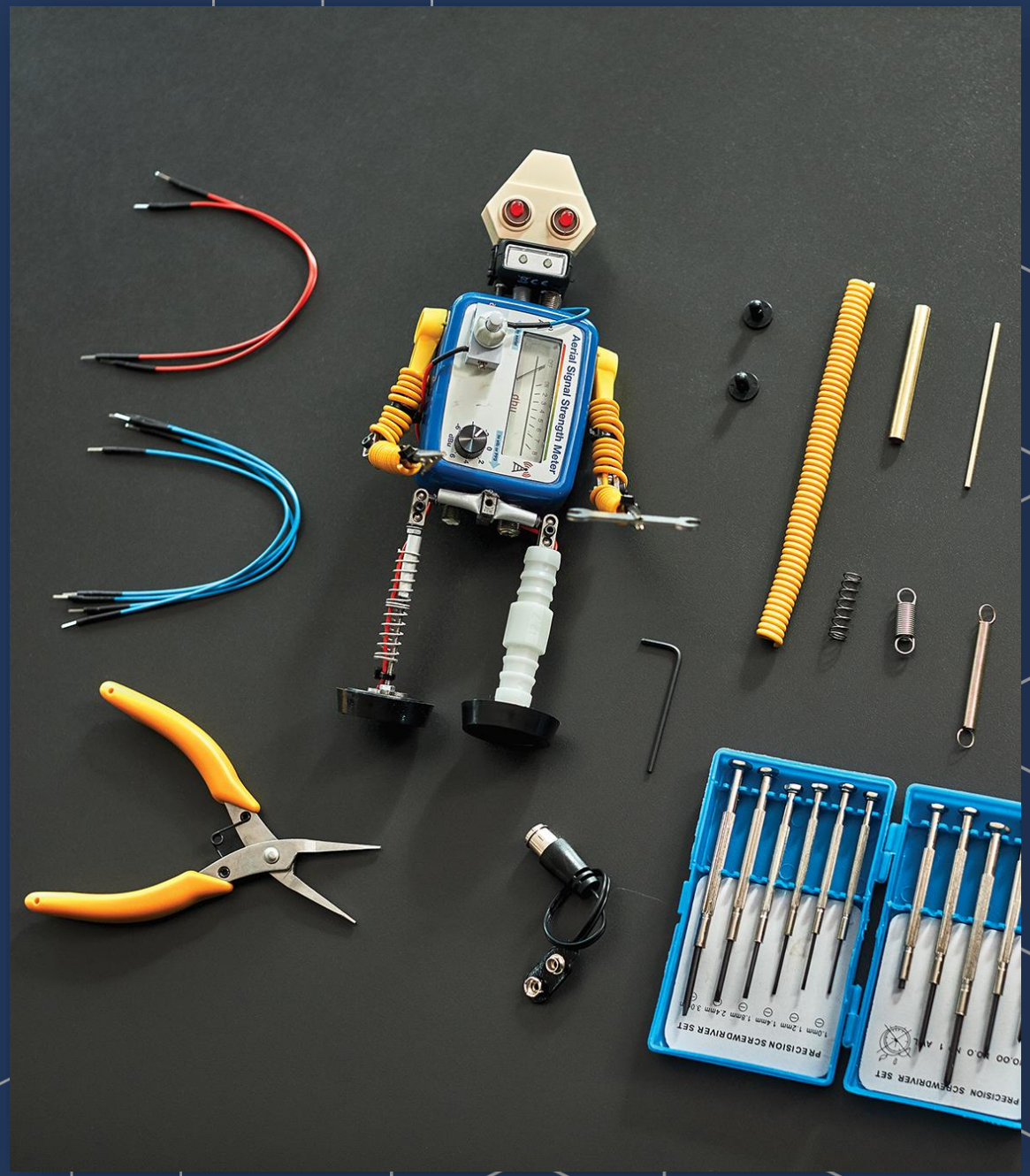
**Component Implementation**

Typescript or JavaScript
- User interface
- Functionality



**Resource Files**

Control artifacts
- JS Libraries
- CSS
- Localization
- Images etc



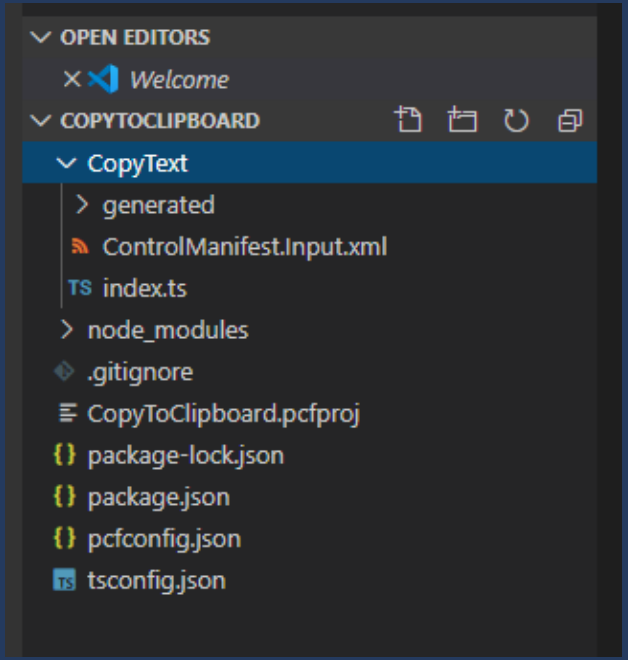

Sample PCF Component (code)

# PCF 控件开发

- 通过使用 VS 的开发人员命令提示符传递基本参数来创建一个新的组件项目

```
c:\CopyText>pac pcf init --namespace SampleNamespace --name CopyText --template field
The PowerApps component framework project was successfully created in 'c:\CopyText'.
Be sure to run 'npm install' in this directory to install project dependencies.
```

- **使用命令 npm install安装项目Build工具**
- 最常使用的几个文件
  - Index.ts
    - Main logic implemented in this file
    - Typescript
  - ControlManifest.Input.xml
    - Configuration file
    - Reference supporting code files here, define input/output params



# PCF – Index.ts

- Out of box, required methods provided by default
  - constructor
  - Init (required)
  - updateView (required)
  - getOutputs (optional)
  - destroy (required)

CopyText > TS index.ts > ...

```typescript
import {IInputs, IOutputs} from "./generated/ManifestTypes";

export class CopyText implements ComponentFramework.StandardControl<IInputs, IOutputs> {
    constructor(){ }
    public init(context: ComponentFramework.Context<IInputs>,
        notifyOutputChanged: () => void, state: ComponentFramework.Dictionary,
         container:HTMLDivElement)
    {
        // Add control initialization code
    }

    public updateView(context: ComponentFramework.Context<IInputs>): void
    {
        // Add code to update control view
    }

    public getOutputs(): IOutputs
    {
        return {};
    }

    public destroy(): void
    {
        // Add code to cleanup control if necessary
    }
}
```

# PCF – ControlManifest.Input.xml

- Configuration file
- Define input/output params
- Elements
  - control
    - Defines the namespace and name of control
  - property
    - Input/output param, can be bound to a field or dataset
  - resources
    - Additional code or string files referenced with paths
  - feature-usage
    - Whether any device specific features need to be enabled

# PCF – ControlManifest.Input.xml sample

CopyText > ControlManifest.Input.xml

```xml
<?xml version="1.0" encoding="utf-8" ?>
<manifest>
  <control namespace="SampleNamespace" constructor="CopyText" version="0.0.1" display-name-key="CopyText"
  description-key="CopyText description" control-type="standard">
    <!-- property node identifies a specific, configurable piece of data that the control expects from CDS -->
    <property name="sampleProperty" display-name-key="Property_Display_Key" description-key="Property_Desc_Key"
    of-type="SingleLine.Text" usage="bound" required="true" />
    <!--
      Property node's of-type attribute can be of-type-group attribute.
      Example:
      <type-group name="numbers">
        <type>Whole.None</type>
        <type>Currency</type>
        <type>FP</type>
        <type>Decimal</type>
      </type-group>
      <property name="sampleProperty" display-name-key="Property_Display_Key" description-key="Property_Desc_Key"
    -->
    <resources>
      <code path="index.ts" order="1"/>
      <!-- UNCOMMENT TO ADD MORE RESOURCES
      <css path="css/CopyText.css" order="1" />
      <resx path="strings/CopyText.1033.resx" version="1.0.0" />
      -->
    </resources>
    <!-- UNCOMMENT TO ENABLE THE SPECIFIED API
    <feature-usage>
      <uses-feature name="Device.captureAudio" required="true" />
      <uses-feature name="Device.captureImage" required="true" />
      <uses-feature name="Device.captureVideo" required="true" />
      <uses-feature name="Device.getBarcodeValue" required="true" />
      <uses-feature name="Device.getCurrentPosition" required="true" />
      <uses-feature name="Device.pickFile" required="true" />
      <uses-feature name="Utility" required="true" />
      <uses-feature name="WebAPI" required="true" />
    </feature-usage>
    -->
  </control>
</manifest>
```

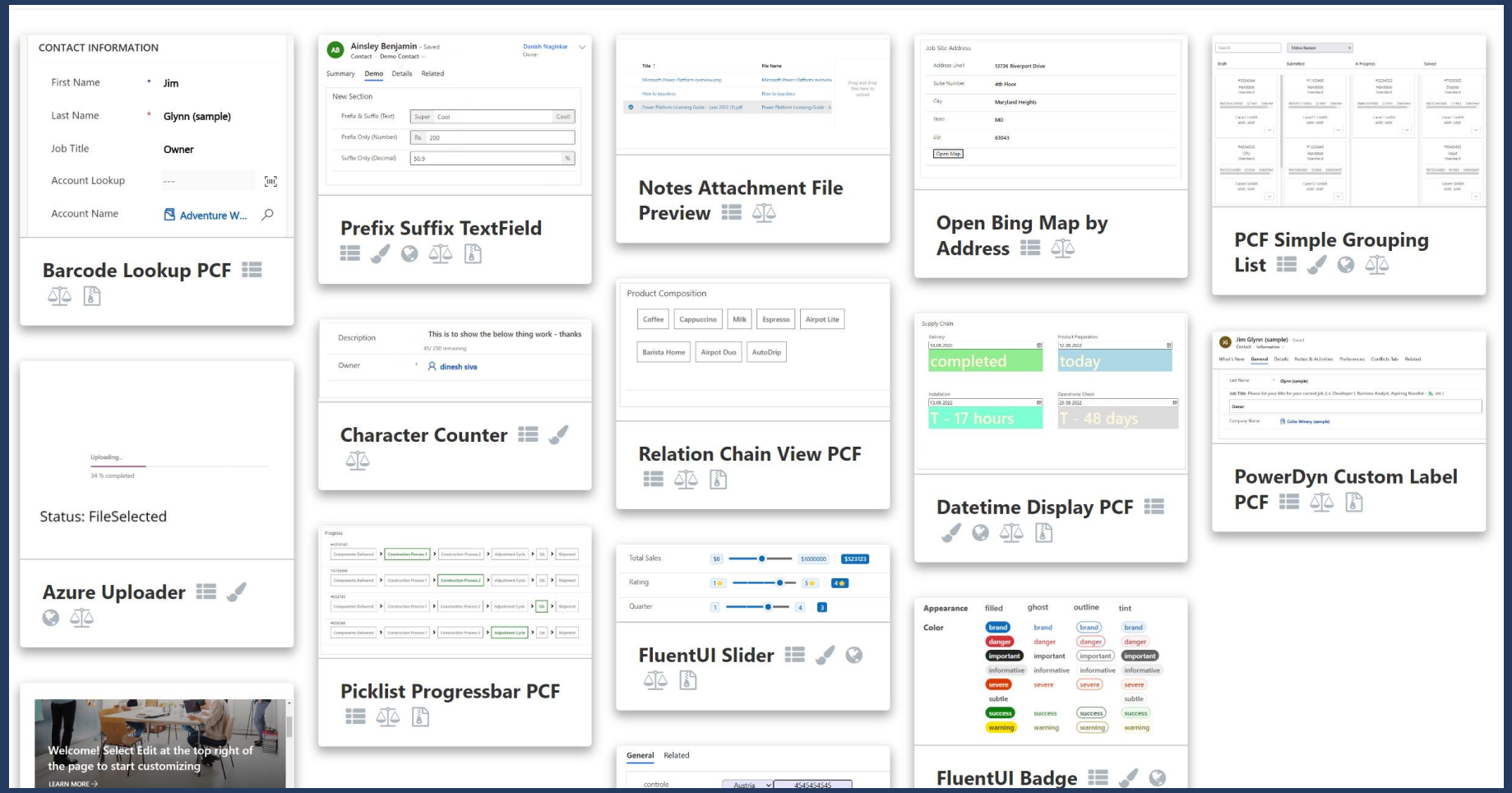
# Office UI Frabric– Additional Library

- Extensive library of controls styled to mimic office elements from common Microsoft pages
  - Office
  - Azure
  - Dynamics

https://developer.microsoft.com/en-us/fabric#/controls

- npm i office-ui-fabric-react

# PCF Gallery

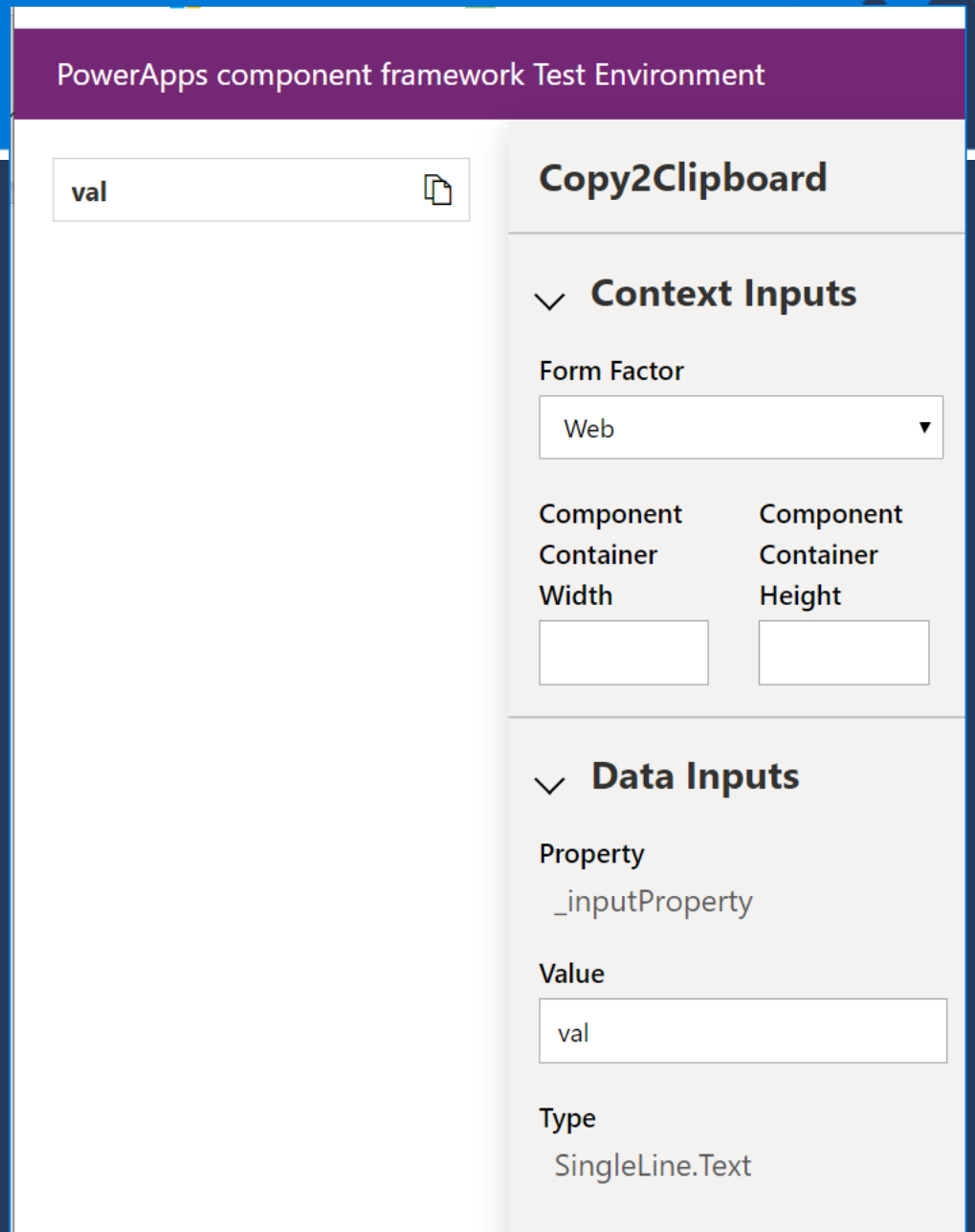



[Home | PCF Gallery]

# Testing and Debugging PowerApps Component Framework

# Testing a PCF

- When ready to test out a control, run:
  - npm run build
  - npm start
- Browser with Test Harness is opened
- To end the Test session, type
  - Ctrl+C into the terminal or cmd window



# Debugging a PCF

- Debugging is easily performed through the browser devtools, either with the PCF test harness or with the control implemented in Dynamics
- To find the index.ts file, search source for a unique namespace or method name in your code

index.ts

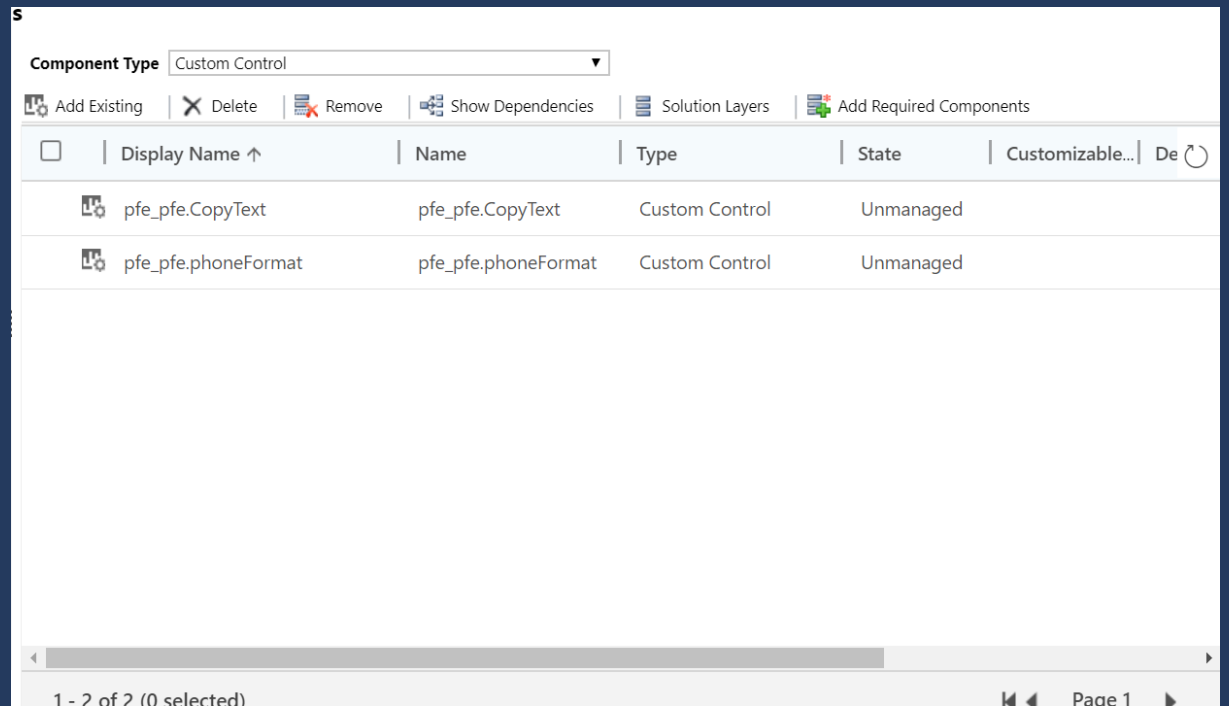
```
30  };
31  /**
32   * Used to initialize the control instance. Controls can kick off remote server calls and other initialization actic
33   * Data-set values are not initialized here, use updateView.
34   * @param context The entire property bag available to control via Context Object; It contains values as set up by t
35   * @param notifyOutputChanged A callback method to alert the framework that the control has new outputs ready to be
36   * @param state A piece of data that persists in one session for a single user. Can be set at any point in a control
37   * @param container If a control is marked control-type='standard', it will receive an empty div element within whic
38   */
39
40
41  CopyText.prototype.init = function (context, notifyOutputChanged, state, container) {
42    icons_1.initializeIcons();
43    this._container = container;
44
45    if (!util_1.isNullOrUndefined(context.parameters._inputProperty) && !util_1.isNullOrUndefined(context.parameters._
46      this._input = context.parameters._inputProperty.raw || "";
47      this.props.inputValue = this._input;
48      this.notifyOutputChanged = notifyOutputChanged;
49    }
50  };
51  /**
52   * Called when any value in the property bag has changed. This includes field values, data-sets, global values such
53   * @param context The entire property bag available to control via Context Object; It contains values as set up by t
54   */
55
56
57  CopyText.prototype.updateView = function (context) {  co  ext = {formatting: e, factory: e, navigation: e, reporting
58    if (this._input != context.parameters._inputProperty.raw || util_1.isNullOrUndefined(this._input) && !util_1.
59      this._input = context.parameters._inputProperty.raw || "";
60    }
61
62
```

"358611200"

# Deploying and Configuring PowerApps Component Framework

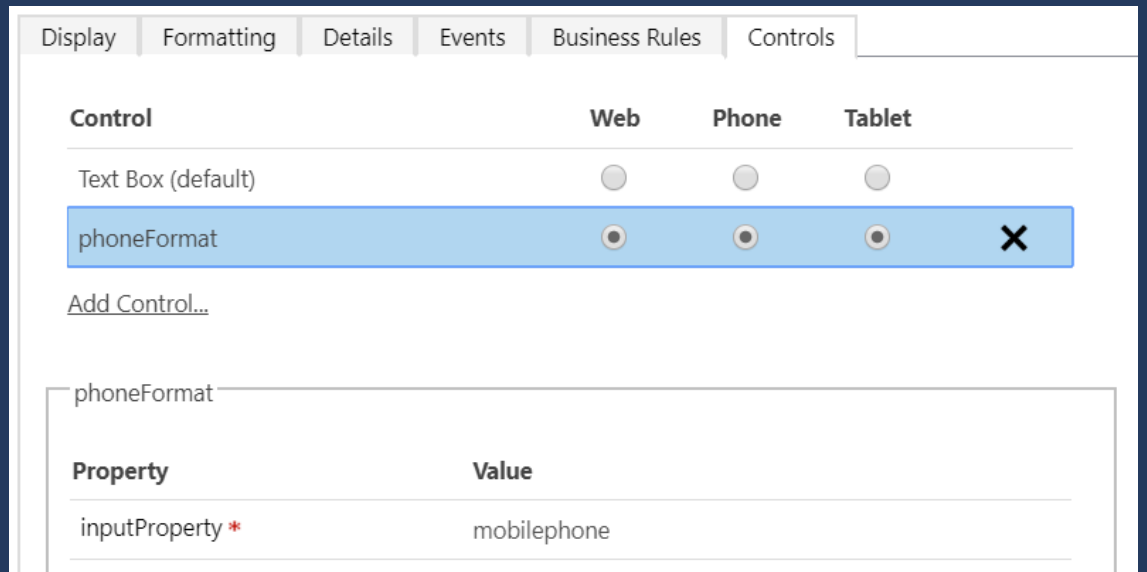
# Deploying a PCF

- Controls can be deployed to PowerApps by:
  - Packaging a solution using the cmd line
  - Pushing the component directly from the cmd line (recommended)
- After control is pushed to a dev environment, use solutions to transfer them from environment to environment.



# Configuring a PCF to be used

- Currently, only the legacy customization experience shows all controls.
- Edit a form, double click a column, click controls
- Click Add Control and select your control from the list
  - If it is not in the list, it might be a data-type mismatch.
- Ensure that you mark the radio buttons to select the new control you've configured.



Microsoft

# Thank you