


Professional Developer's Introduction to the PowerApps and Flow



Our Newest PowerApps Training Course

- Advanced PowerApps and Flow training
 - Designed for students who have already attended Microsoft's App-in-a-day
 - Three days filled with plenty of hands-on lab exercises
 - Learn advanced builder skills & best practices
 - More info at <https://CriticalPathTraining.com>



[Training](#)[Courses](#)[Schedule](#)[Video Gallery](#)[Competition](#)[Contact Us](#)

[Home](#) > [Training Courses](#) > [Business Users](#)

Building Business Solutions with PowerApps and Flow

[Course Overview \(PDF\)](#)

Building Business Solutions with Power Apps and Flow is an intensive 3-day training class designed for people who have already attended the Microsoft App-in-a-Day (AIID) training and are ready to move their PowerApps and Flow skills to the next level. Students will learn best practices for building canvas apps and flows to update and manage content in Azure SQL, SharePoint Online, Excel workbooks and OneDrive for Business.

After providing a foundation for building canvas apps and flows, this course teaches students how to build business solutions with the Common Data Service for Apps (CDS) by creating custom entities and model-driven apps. The class also examines advanced techniques required in real-world PowerApps scenarios such as creating custom connectors and conducting application lifecycle management (ALM) using managed solutions and multiple environments for development, UAT and production.

[Student Prerequisites](#)

Upcoming Offerings

Date	Location	Instructor	Action
Apr-15	Tampa, FL	Ted Pattison	Register
Jun-24	Tampa, FL	Ted Pattison	Register

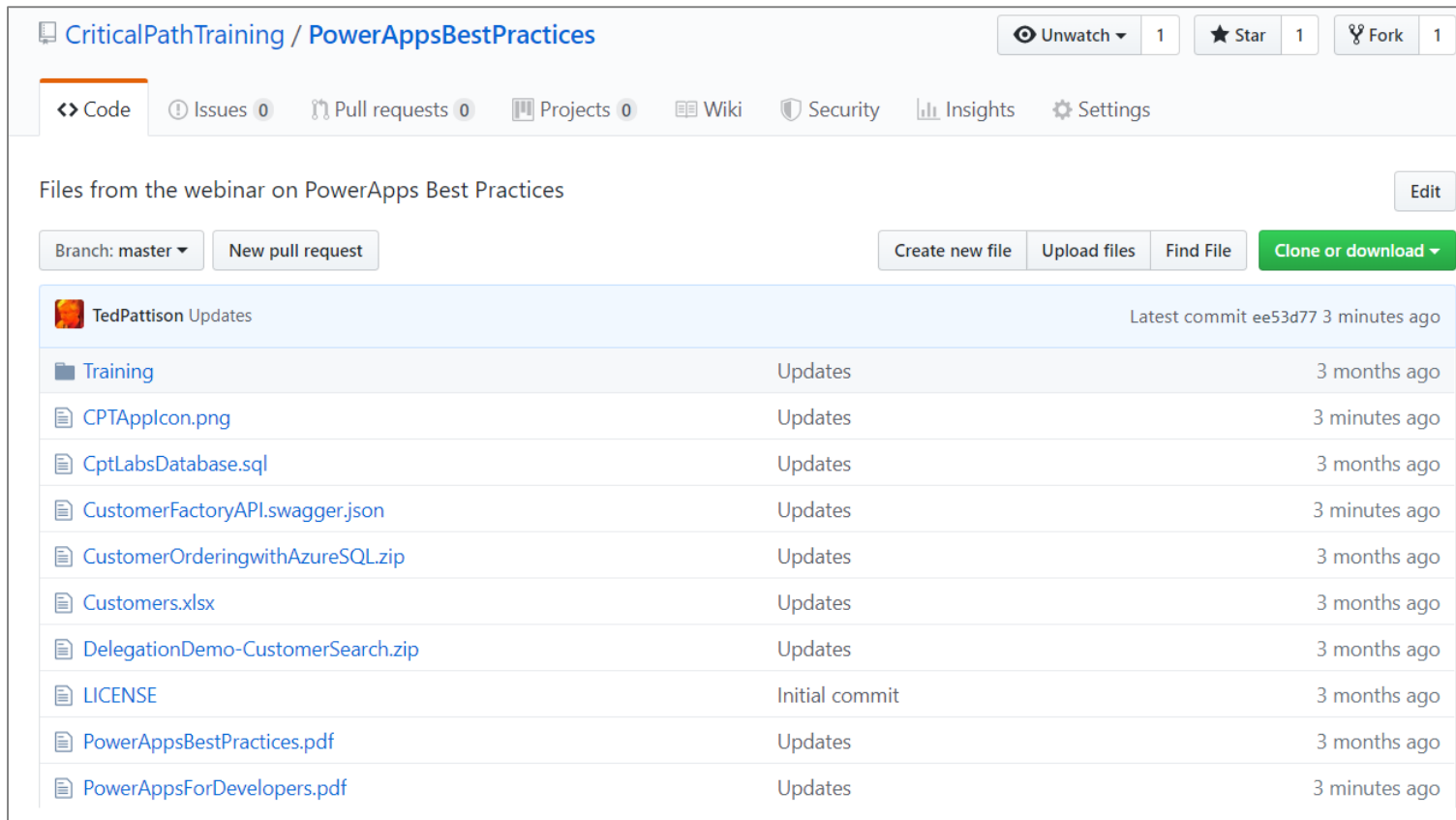
Course Details

Course Code	BBSPA
Course Version	1.0
Course Length	3 Days



Download the Slides and Demo Files

- <https://github.com/CriticalPathTraining/PowerAppsBestPractices>



CriticalPathTraining / PowerAppsBestPractices


Unwatch 1 Star 1 Fork 1

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

Files from the webinar on PowerApps Best Practices Edit

Branch: master New pull request

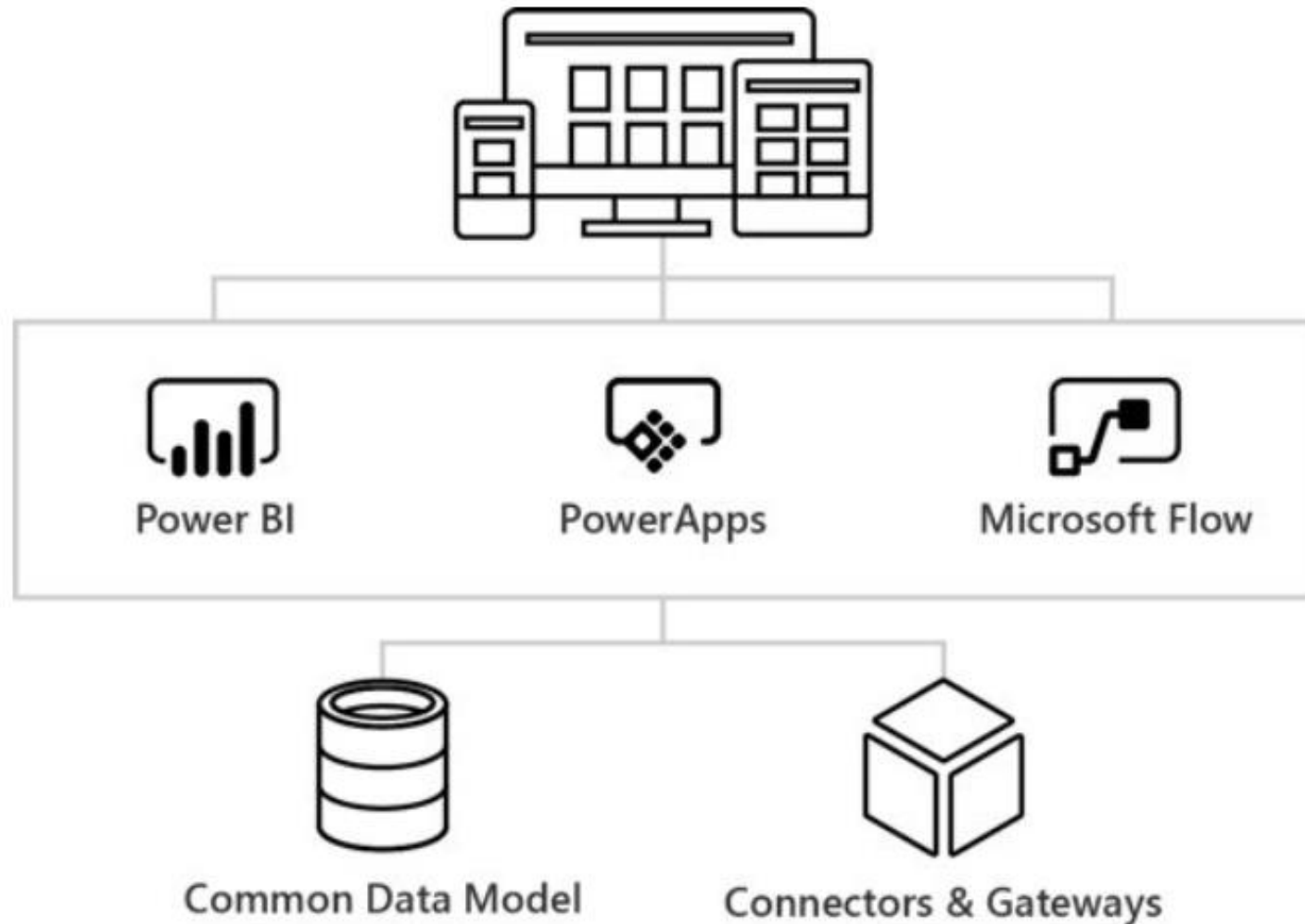
Create new file Upload files Find File Clone or download

 TedPattison Updates Latest commit ee53d77 3 minutes ago

Training	Updates	3 months ago
CPTApplcon.png	Updates	3 minutes ago
CptLabsDatabase.sql	Updates	3 months ago
CustomerFactoryAPI.swagger.json	Updates	3 minutes ago
CustomerOrderingwithAzureSQL.zip	Updates	3 months ago
Customers.xlsx	Updates	3 months ago
DelegationDemo-CustomerSearch.zip	Updates	3 months ago
LICENSE	Initial commit	3 months ago
PowerAppsBestPractices.pdf	Updates	3 months ago
PowerAppsForDevelopers.pdf	Updates	3 minutes ago



Power Platform Overview



Agenda

- Canvas Apps
- PowerApps Components
- PowerApps Control Framework
- Flows
- Custom Connectors
- Not enough time to include...
 - Common Data Service for Apps (CDS)
 - Model-driven Apps



Challenges with Building Canvas Apps

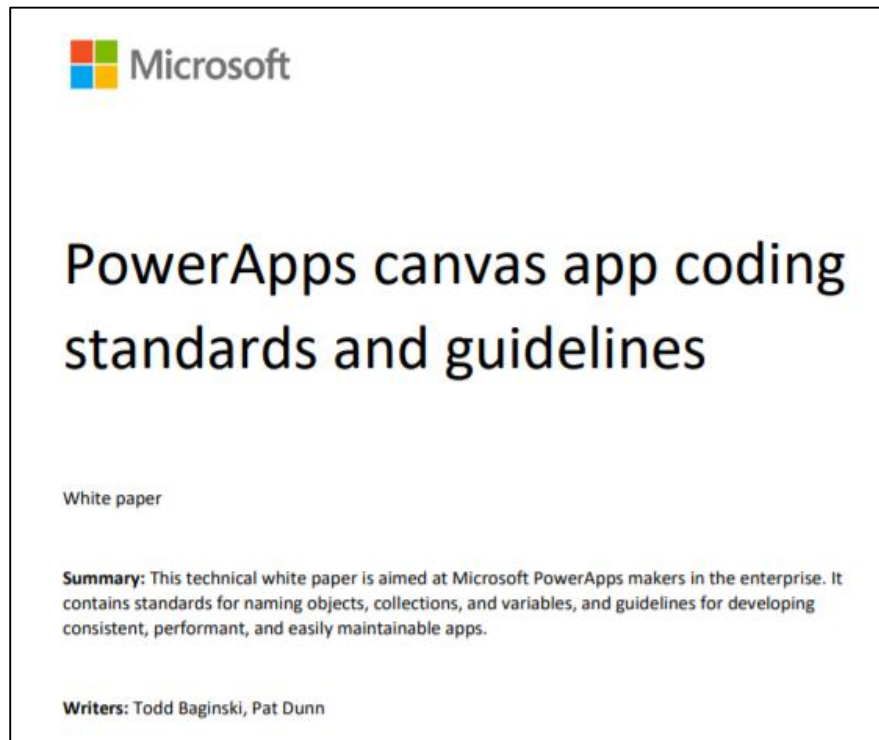
- Who is the typical PowerApps application maker?
 - A developer without a software development background
- What is the maker is responsible for?
 - designing, building, testing, deploying
- What are the challenges in deploying canvas apps?
 - Building projects that are easy to maintain and extend
 - Building consistency across team members
 - Building canvas apps with better performance



Read the Canvas Apps Whitepaper

- PowerApps canvas app coding standards and guidelines

<https://powerapps.microsoft.com/en-us/blog/powerapps-canvas-app-coding-standards-and-guidelines/>



Todd Baginski

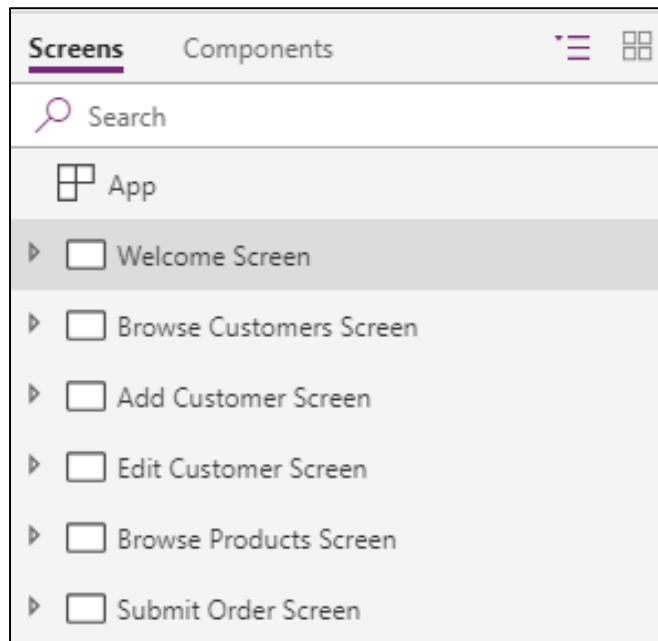


Pat Dunn



Creating Screen Names

- It's import to create screen names correctly
 - Screen names read aloud by screen readers
 - Names should include spaces and avoid abbreviations
 - Screen name should end with the word "Screen"
 - Screen name should reflect purpose of screen



Control Naming Convention

- Define and document a naming convention
 - Use this list as a starting point

button	btn
camera control	cam
canvas	can
card	crd
collection	col
combo box	cmb
date picker	dte
drop down	drp
radio button	rad
form	frm
gallery	gal

group	grp
header page shape	fdr
html text	html
icon	ico
image	img
label	lbl
page section shape	sec
shapes (rectangle, etc.)	shp
table data	tbl
text input	txt
timer	tim

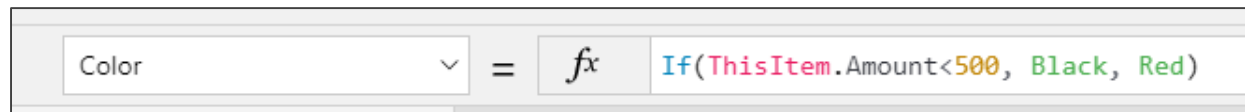


PowerApps Formula Language

- PowerApps provides its own Formula Language
 - Designed to be as similar as possible to Excel Formula language
 - PowerApps Formula Language includes built-in set of functions
- You write formulas for specific properties
 - Set the Text property for a label



- Set the Color property of the label text



- Write an formula to filter the items shown in a gallery



Events and State Changes

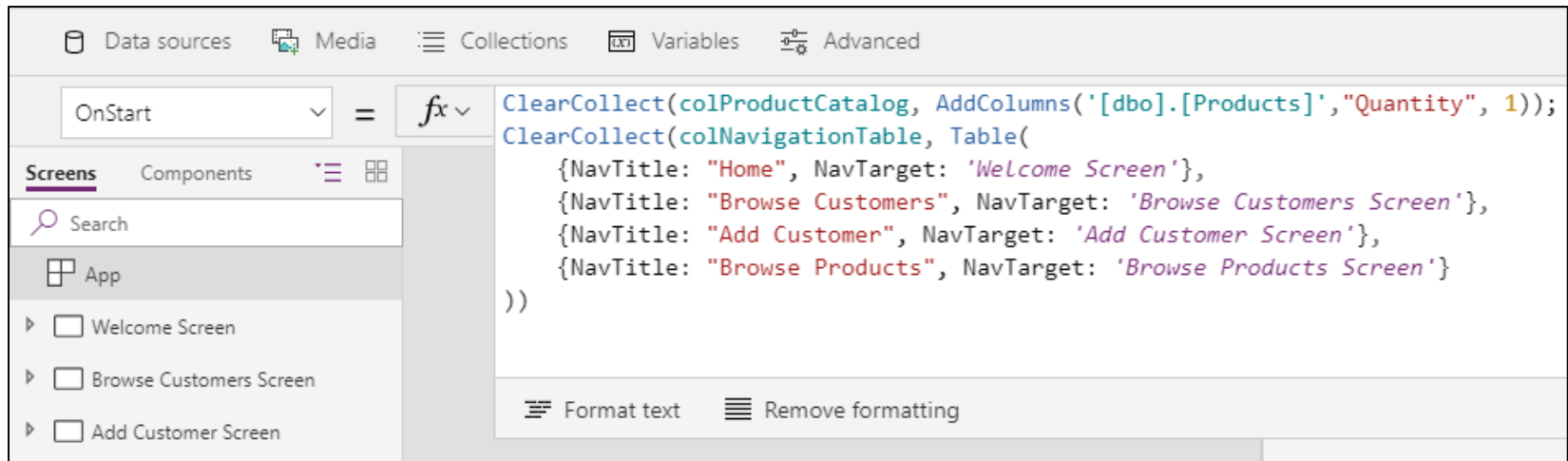
- Formulas for event properties can contain imperative logic
 - `onSelect`, `onVisible`, `onStart`, etc.
- Imperative logic is used to take action
 - Set value of global variable or context variable
 - Add item to a collection
 - Navigate between screens
 - Submit data to server
- You can chain actions together with chaining operator (;)

```
fx Collect(Customers, {ID: NextCustomerId, FirstName: FirstNameTextInput.Text, LastName: LastNameTextInput.Text});  
Reset(FirstNameTextInput);  
Reset(LastNameTextInput);  
Set(NextCustomerId, NextCustomerId+1);
```



App OnStart

- App **OnStart** property used to initialize state in app
 - Event provides support for initializing state in app at startup
 - Commonly used to initialize global variables and collections
 - Right-click **App** in left navigation to run **OnStart** while in editor



Declarative vs Imperative Functions

- Non-highlighted functions used to return values
 - These are declarative functions
- Highlighted functions used to perform actions
 - These are imperative functions

Abs	Collect	Day	HashTags	Max	Rand	Shuffle	TrimEnds
Acceleration	Color	Defaults	Hour	Mid	Refresh	Sin	Ungroup
Acos	ColorFade	Degrees	If	Min	Remove	Sort	Update
Acot	ColorValue	Disable	IfError	Minute	Removelf	SortByColumns	UpdateContext
AddColumns	Compass	Distinct	IsBlank	Mod	RenameColumns	Split	UpdateIf
And	Concat	Download	IsEmpty	Month	Replace	Sqrt	Upper
App	Concatenate	DropColumns	IsMatch	Navigate	Reset	StartsWith	User
Asin	Connection	EditForm	IsNumeric	NewForm	ResetForm	StdevP	Validate
Atan	Count	Enable	IsToday	Not	Revert	Substitute	Value
Atan2	Cos	EndsWith	Language	Notify	RGBA	SubmitForm	VarP
Average	Cot	Errors	Last	Now	Right	Sum	ViewForm
Back	CountA	EncodeUrl	LastN	Operators	Round	Switch	Weekday
Blank	CountIf	Exit	Launch	Or	RoundDown	Table	Year
Calendar	CountRows	Exp	Left	Param	RoundUp	Tan	
Char	DataSourceInfo	Filter	Len	Patch	SaveData	Text	
Choices	Date	Find	Ln	Pi	Search	Time	
Clear	DateAdd	First	LoadData	PlainText	Second	TimeValue	
ClearCollect	DateDiff	FirstN	Location	Power	Select	TimeZoneOffset	
Clock	DateTimeValue	ForAll	LookUp	Proper	Set	Today	
Coalesce	DateValue	GroupBy	Lower	Radians	ShowColumns	Trim	



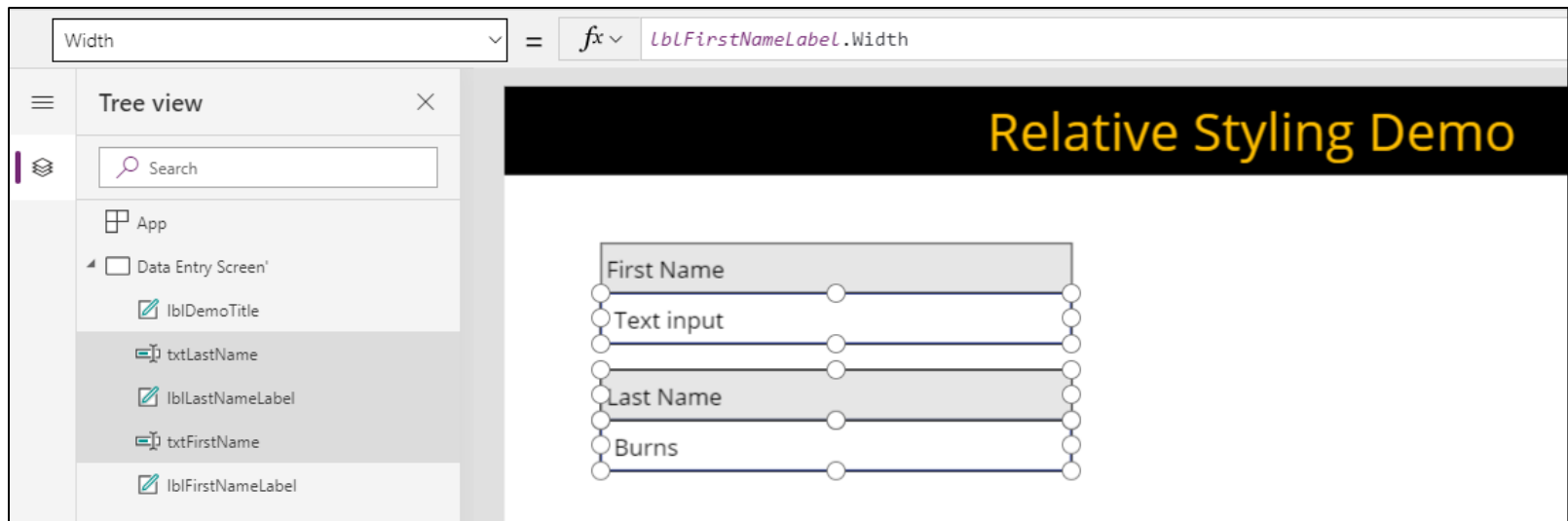
DEMO



Examining the Customer Ordering Canvas App

Smart Designing Screen

- Use relative styling
 - Calculate control property values from other controls
 - Common to use properties like X, Y, Width, Height, Size, Fill, etc.



Working with Edit Forms and Data Cards

- Form acts as a container for data cards
 - Each form binds to a single record
 - Within a form, each data card binds to an underlying field
 - Each data card contains an encapsulated set of child controls

The screenshot displays a Visual Studio IDE window titled 'frmAddCustomer'. On the left, a Solution Explorer shows a list of data cards and other screens:

- First Name_DataCard1
- Last Name_DataCard1
- Company_DataCard1
- Email Address_DataCard1
- Work Phone_DataCard1
- Home Phone_DataCard1
- Address_DataCard1
- City_DataCard1
- State_DataCard1
- Zipcode_DataCard1
- ☐ Edit Customer Screen
- ☐ Browse Products Screen
- ☐ Submit Order Screen
- ☐ Order Confirmation Screen

The main design area on the right shows the layout of the 'frmAddCustomer' form. It features two columns of data cards, each with a text box and a label:

- Left Column:**
 - * First Name (text box)
 - Company (text box)
 - Work Phone (text box)
 - Address (text box)
 - State (text box)
- Right Column:**
 - * Last Name (text box)
 - * Email Address (text box)
 - Home Phone (text box)
 - City (text box)
 - Zipcode (text box)

The form is enclosed in a rectangular frame with corner handles for resizing.



Using Patch Instead of an Edit Form

- Sometimes edit forms are not the best option
 - **Patch** function used to create and update records in data source

```
Patch(  
    '[dbo].[Orders]',  
    Defaults('[dbo].[Orders]'),  
    {  
        CustomerId: galCustomers.Selected.CustomerId,  
        OrderAmount: Sum(  
            colShoppingCart,  
            Total  
        ),  
        OrderDate: Today()  
    }  
)
```

- **Patch** function can be used with **ForAll** to insert multiple records

```
ForAll(  
    colShoppingCart,  
    Patch(  
        '[dbo].[OrderDetails]',  
        Defaults('[dbo].[OrderDetails]'),  
        {  
            OrderId: First(colLastOrder).OrderId,  
            ProductId: ProductId,  
            Quantity: Quantity,  
            Total: Total  
        }  
    )  
)
```



State Variables



- Collections
 - Created as tables at app scope
 - Managed using `Collect`, `Clear` and `ClearCollect`
 - Can be stored to local device using `SaveData` & `LoadData`
- Context variables
 - Created as primitive, record or table at screen scope
 - Managed using `UpdateContext` and `Navigate`
- Global variables
 - Created as primitive, record or table at app scope
 - Created and managed using `Set` function



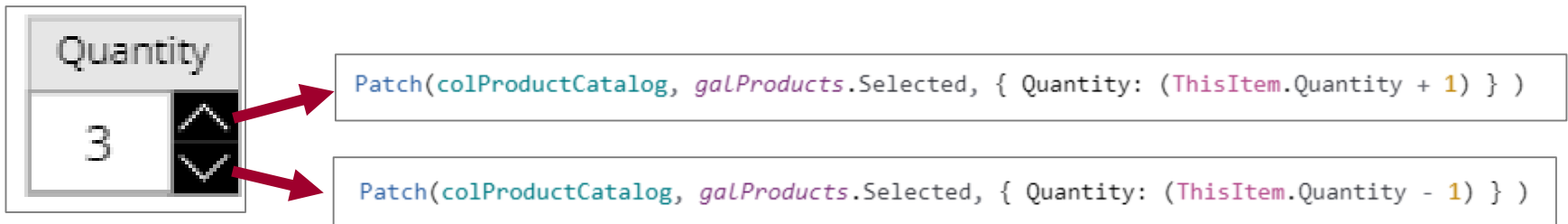
Updatable Collection Columns

- Often helpful to add updatable column to collections

```
ClearCollect(colProductCatalog, AddColumns('[dbo].[Products]', "Quantity", 1));
```

	Batman Action Figure A super hero who sometimes plays the role of a dark knight.	Quantity 1	ADD TO CART
	Captain America Action Figure A super action figure that protects freedom and the American way of life.	Quantity 3	ADD TO CART

- Columns updated using calls to Patch



Capturing Return Value from Patch

```
ClearCollect(
    collastOrder,
    Patch(
        '[dbo].[Orders]',
        Defaults('[dbo].[Orders]'),
        {
            CustomerId: galCustomers.Selected.CustomerId,
            OrderAmount: Sum(
                colShoppingCart,
                Total
            ),
            OrderDate: Today()
        }
    )
);
ClearCollect(
    collastOrderDetails,
    ForAll(
        colShoppingCart,
        Patch(
            '[dbo].[OrderDetails]',
            Defaults('[dbo].[OrderDetails]'),
            {
                OrderId: First(collastOrder).OrderId,
                ProductId: ProductId,
                Quantity: Quantity,
                Total: Total
            }
        )
    )
);
```

CustomerId	OrderAmount	OrderDate	OrderId
10	44.85	4/3/2019	14

collastOrder

Id	OrderId	ProductId	Quantity	Total
36	14	1	1	14.95
37	14	4	2	19.9
38	14	6	10	10

collastOrderDetails




Populating UI from Cached State

CustomerId	OrderAmount	OrderDate	OrderId
10	44.85	4/3/2019	14

collLastOrder

Id	OrderId	ProductId	Quantity	Total
36	14	1	1	14.95
37	14	4	2	19.9
38	14	6	10	10

collLastOrderDetails



Order Confirmation

[Home](#)[Browse Customers](#)[Add Customer](#)[Browse Products](#)

The Order Has Been Submitted Successfully

Order Information




Order ID:

Order Date:

Order Amount:

Customer Shipping Address:

Items purchased in Order #0014

No	Product	List Price	Total	
1	Batman Action Figure	\$14.95	\$14.95	
2	Green Hulk Action Figure	\$9.95	\$19.90	
10	Twitter Follower Action Figure	\$1.00	\$10.00	

Shopping Cart Total: \$44.85

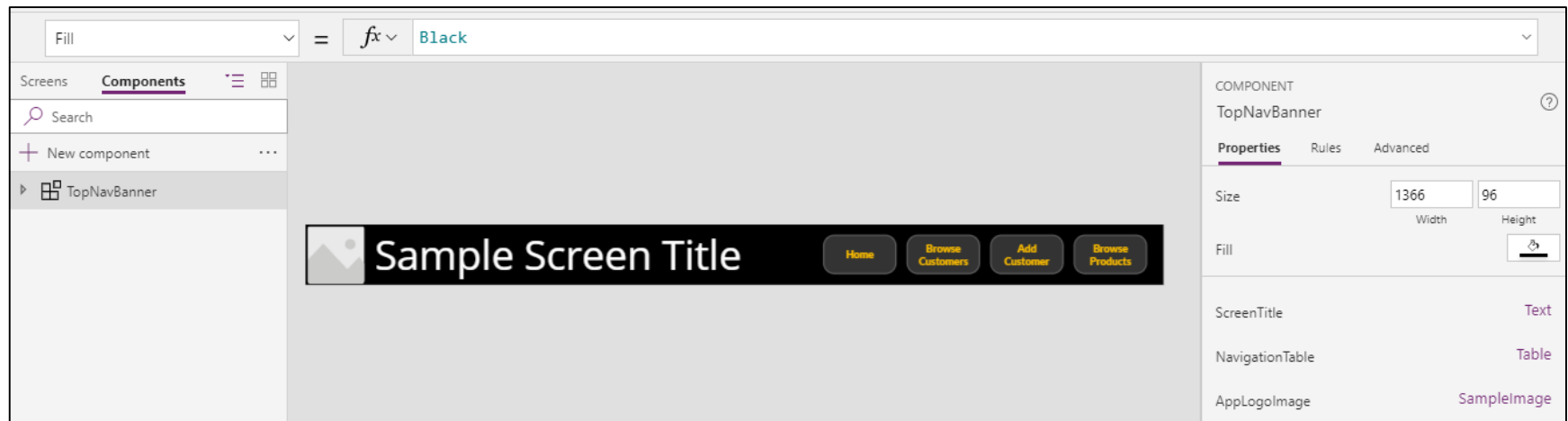
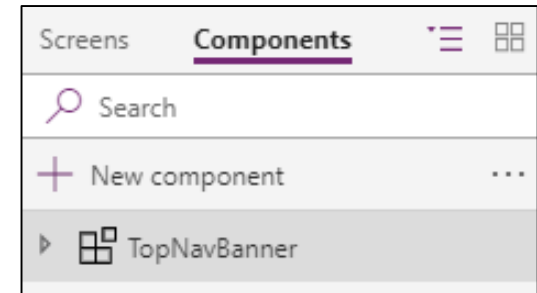
Agenda

- ✓ Canvas Apps
- PowerApps Components
 - PowerApps Control Framework
 - Flows
 - Custom Connectors

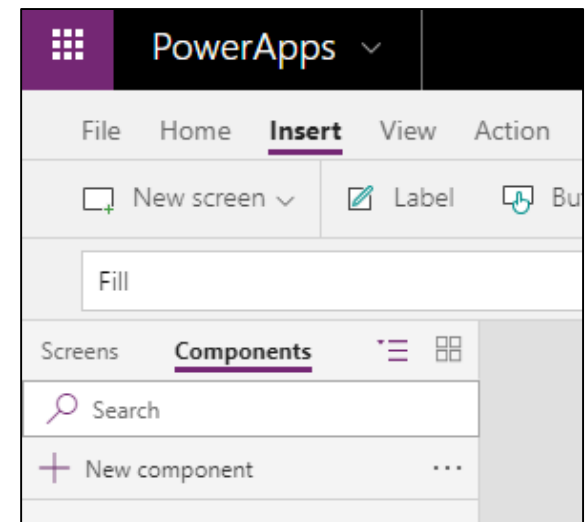
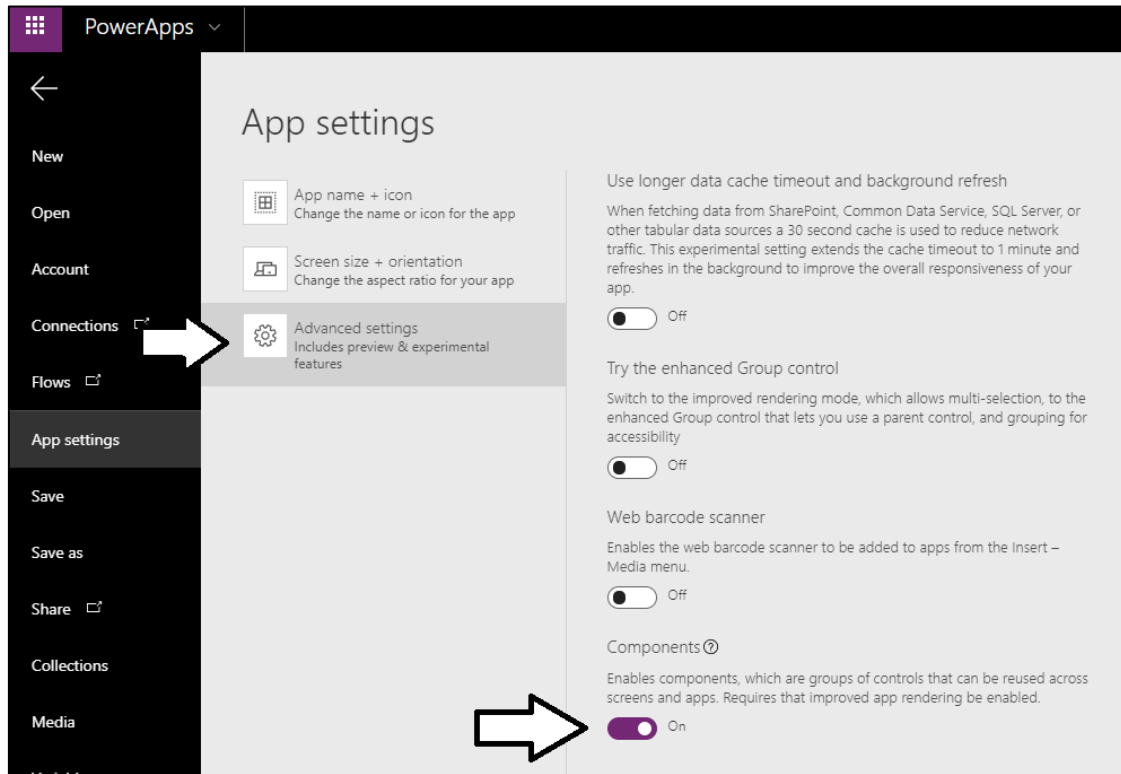


Designing Components

- Steps to using components
 - Create new component
 - Add component properties
 - Implement component UI and behavior
 - Add component to screens in your canvas apps



Enabling Components in a Canvas App



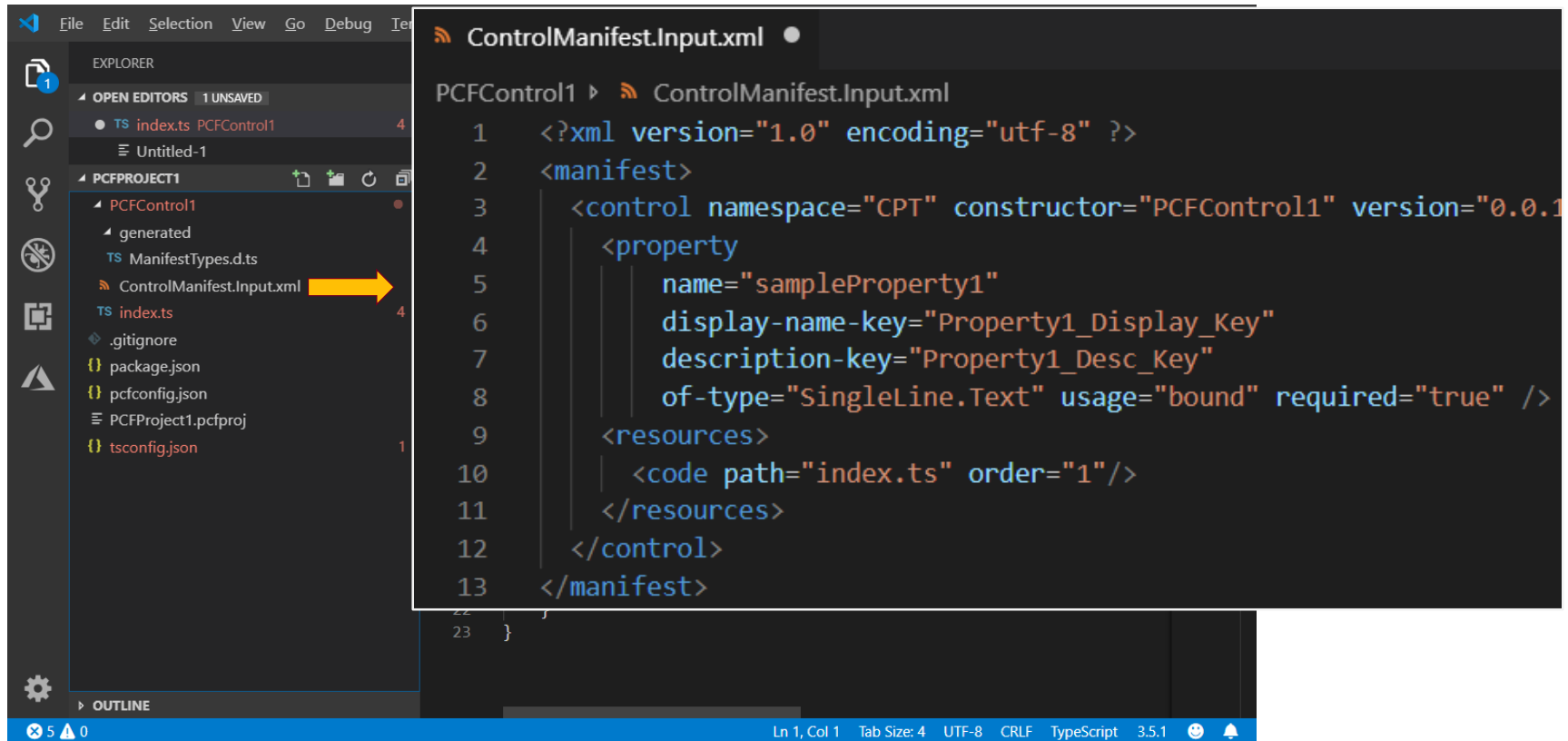
Agenda

- ✓ Canvas Apps
- ✓ PowerApps Components
- PowerApps Control Framework
 - Flows
 - Custom Connectors



PowerApps Control Framework

- Controls developed using Node.js environment
 - You implement behavior using TypeScript



The screenshot displays the Visual Studio Code interface. On the left, the Explorer pane shows the project structure for 'PCFPROJECT1'. The file 'ControlManifest.Input.xml' is highlighted with a yellow arrow. The main editor area shows the content of 'ControlManifest.Input.xml', which is an XML manifest for a PowerApps control. The manifest defines a control named 'sampleProperty1' with a 'SingleLine.Text' type, a 'bound' usage, and a required property. It also specifies the control's resources, including a TypeScript file 'index.ts'.

```
ControlManifest.Input.xml

PCFControl1 > ControlManifest.Input.xml

1  <?xml version="1.0" encoding="utf-8" ?>
2  <manifest>
3    <control namespace="CPT" constructor="PCFControl1" version="0.0.1"
4      <property
5        name="sampleProperty1"
6        display-name-key="Property1_Display_Key"
7        description-key="Property1_Desc_Key"
8        of-type="SingleLine.Text" usage="bound" required="true" />
9      <resources>
10       <code path="index.ts" order="1"/>
11     </resources>
12   </control>
13 </manifest>
```



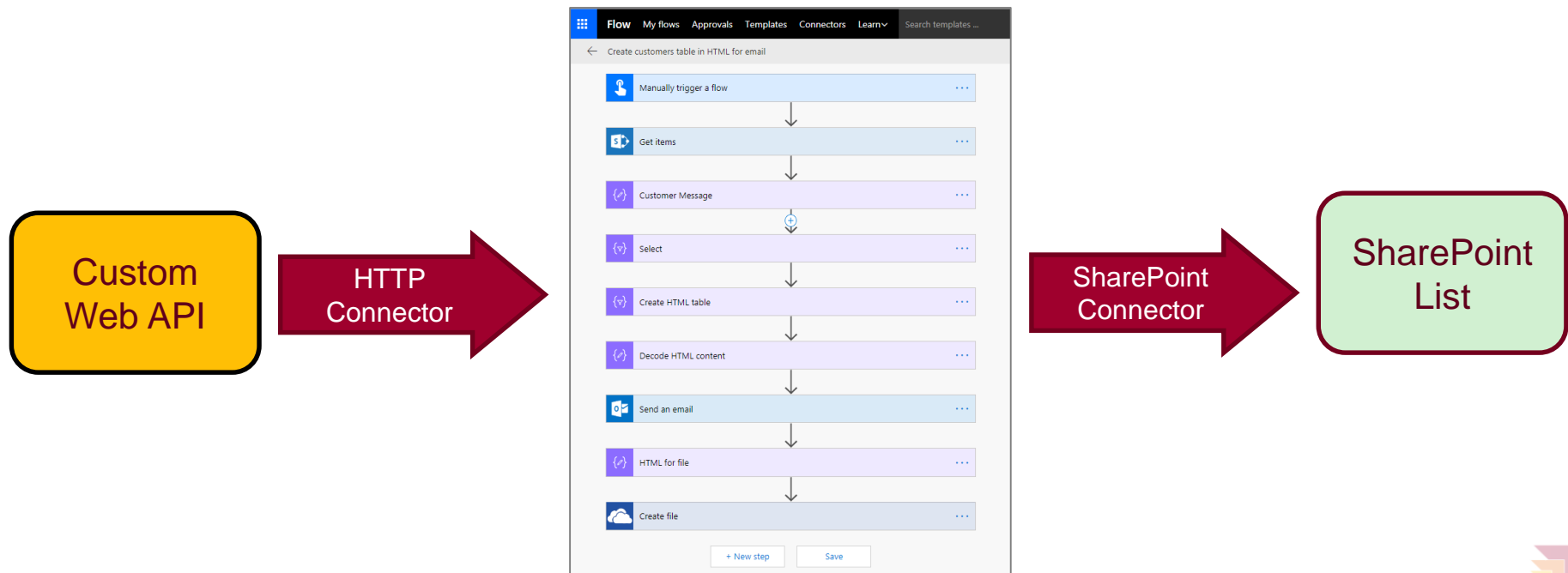
Agenda

- ✓ Canvas Apps
- ✓ PowerApps Components
- ✓ PowerApps Control Framework
- Flows
 - Custom Connectors



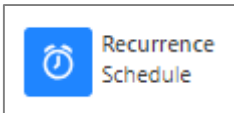
Building Blocks of Flow

- **Triggers** - events that start a flow
- **Actions** - tasks and operation executed by flow
- **Services** - sources and destinations for data
- **Connectors** - wrappers to communicate with service APIs

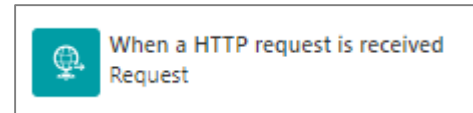
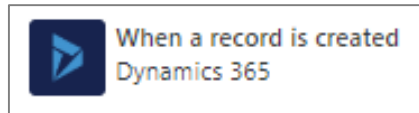
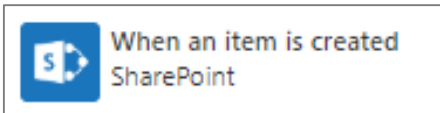


Flow Trigger Types

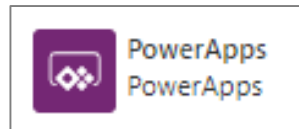
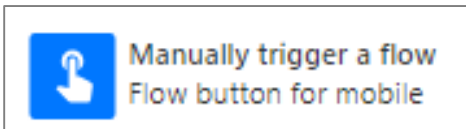
- Scheduled Flow Triggers
 - Runs periodically based on an interval



- Automated Flow Triggers
 - Runs when something happens

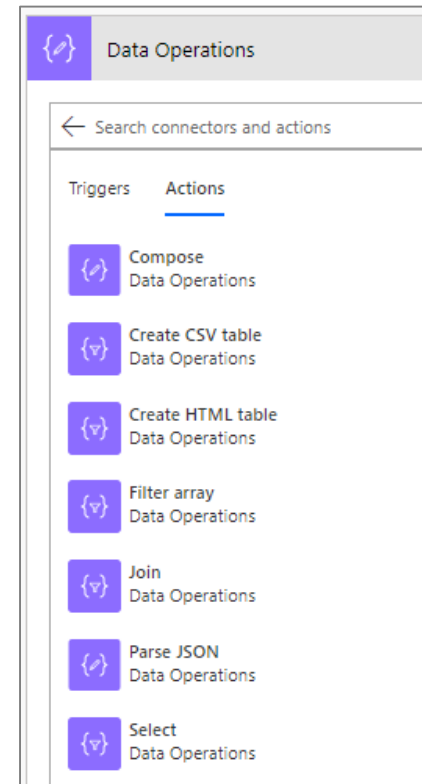
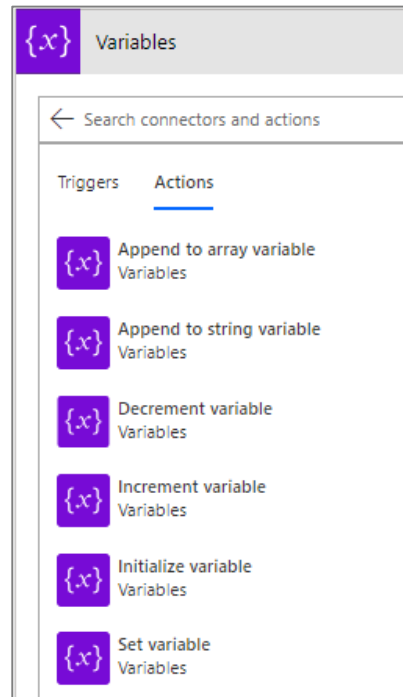
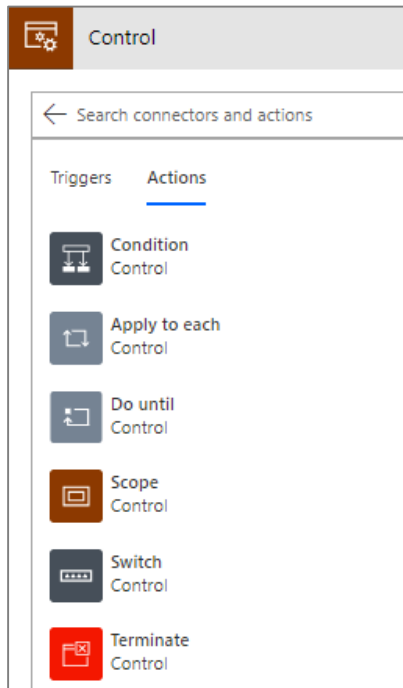


- On-demand Flow Triggers
 - Runs when a user clicks a button

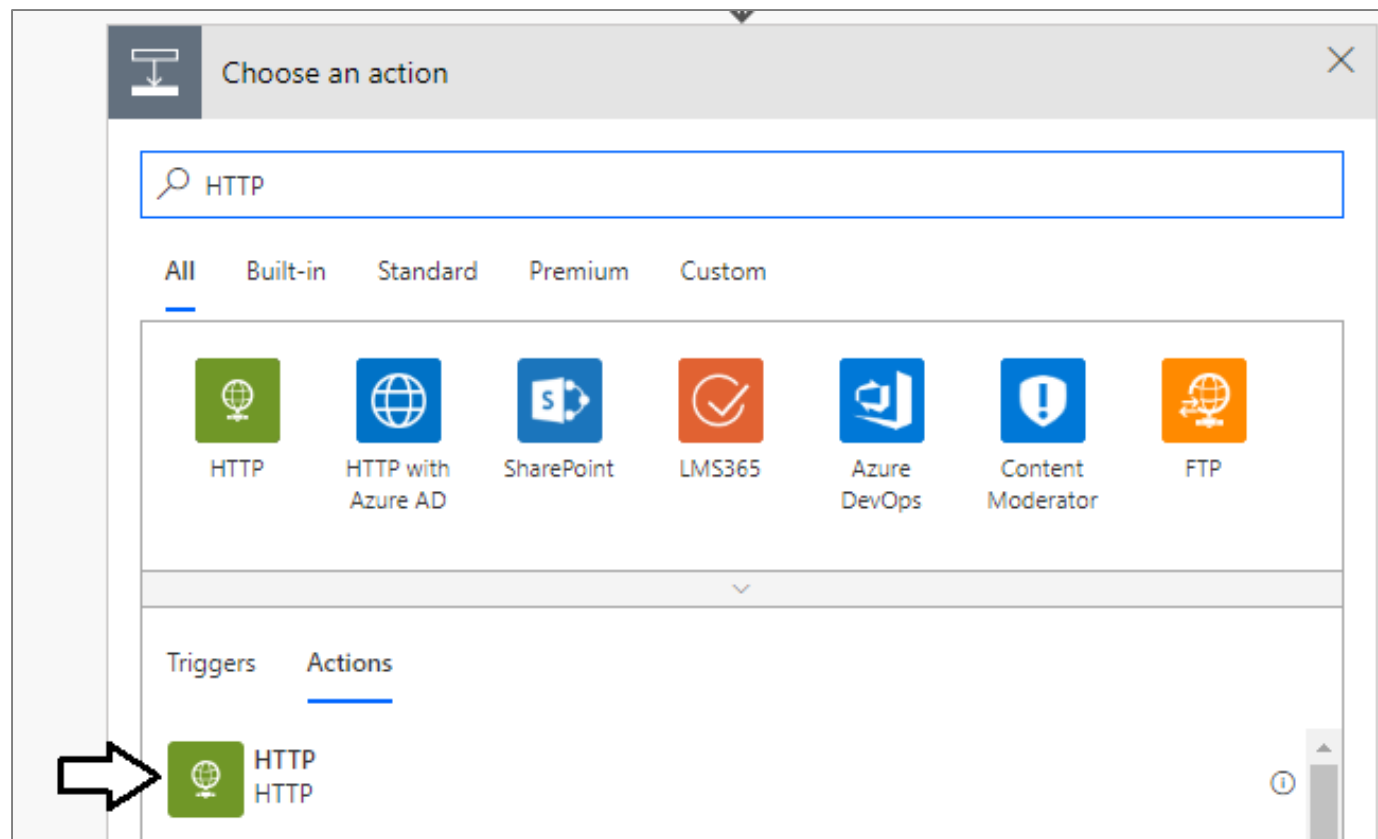


Core Action Categories

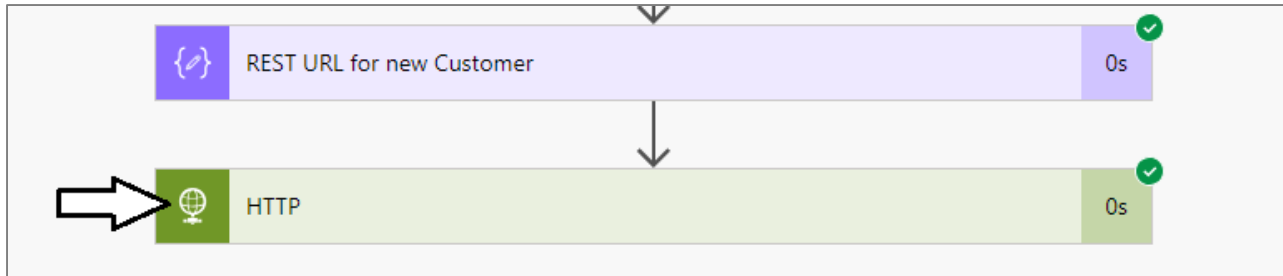
- **Control:** actions to provide control-of-flow
- **Variables:** actions to manage state within flow lifetime
- **Data operations:** action to process data & prepare content



Using the HTTP Action



Inspecting the body of the HTTP Response

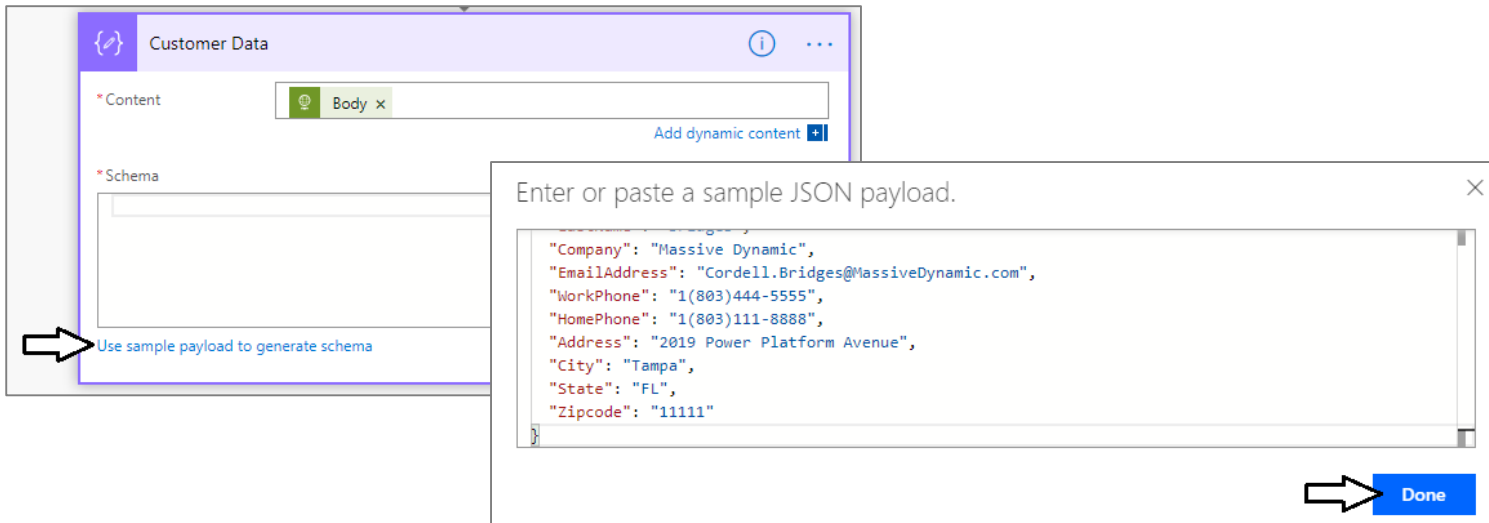
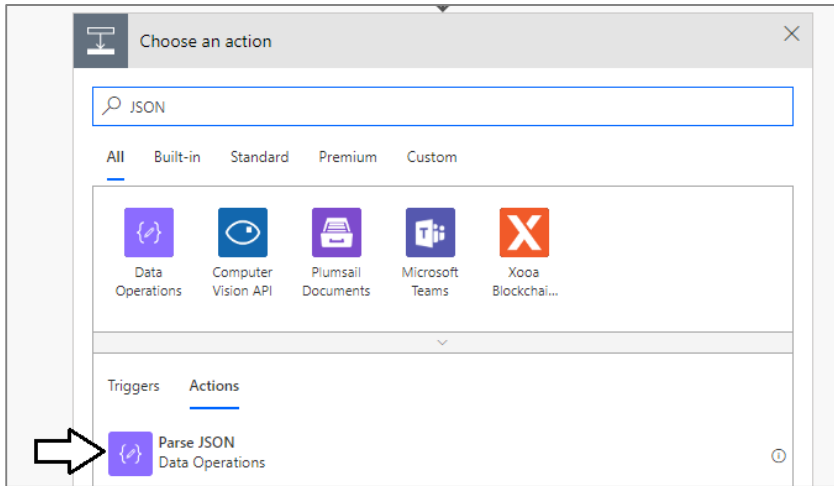


A screenshot of an HTTP response body. A large white arrow points to the "Body" label on the left. The response is a JSON object displayed in a code editor with syntax highlighting. The JSON contains metadata and customer information.

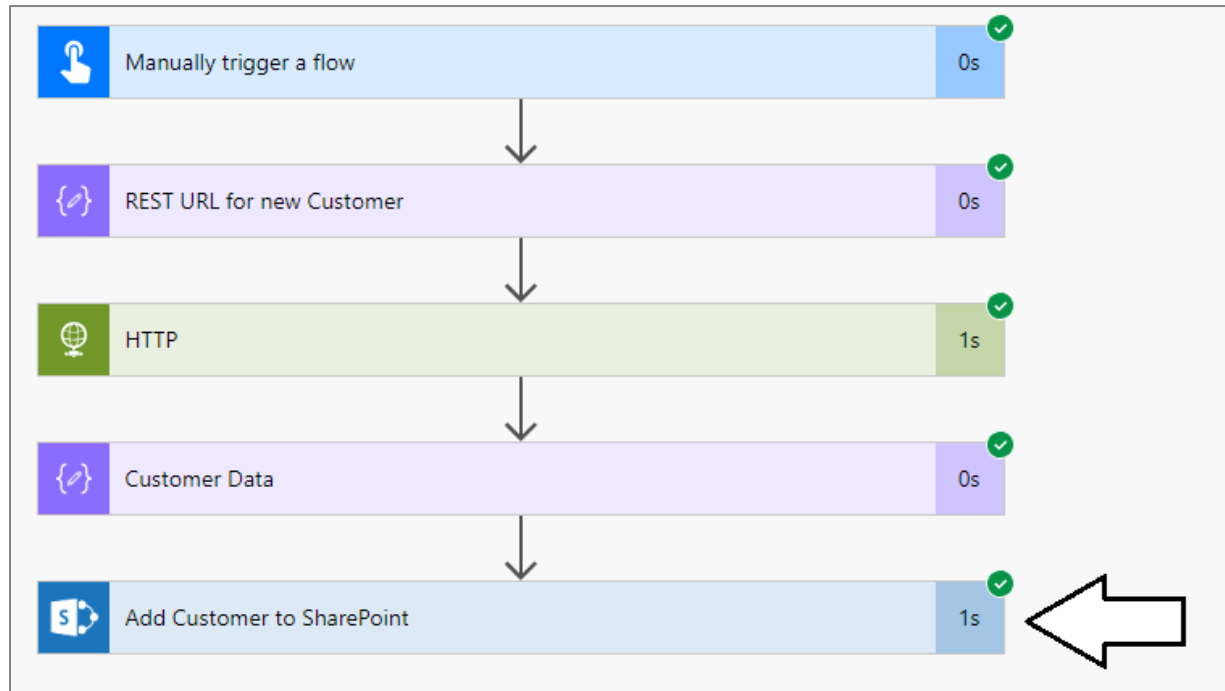
```
{  
  "odata.metadata": "http://subliminalsystems.com/api/\$metadata#",  
  "CustomerId": 47,  
  "FirstName": "Cordell",  
  "LastName": "Bridges",  
  "Company": "Massive Dynamic",  
  "EmailAddress": "Cordell.Bridges@MassiveDynamic.com",  
  "WorkPhone": "1(800)444-5555"  
}
```



Parsing JSON



Adding a New Item to a SharePoint List



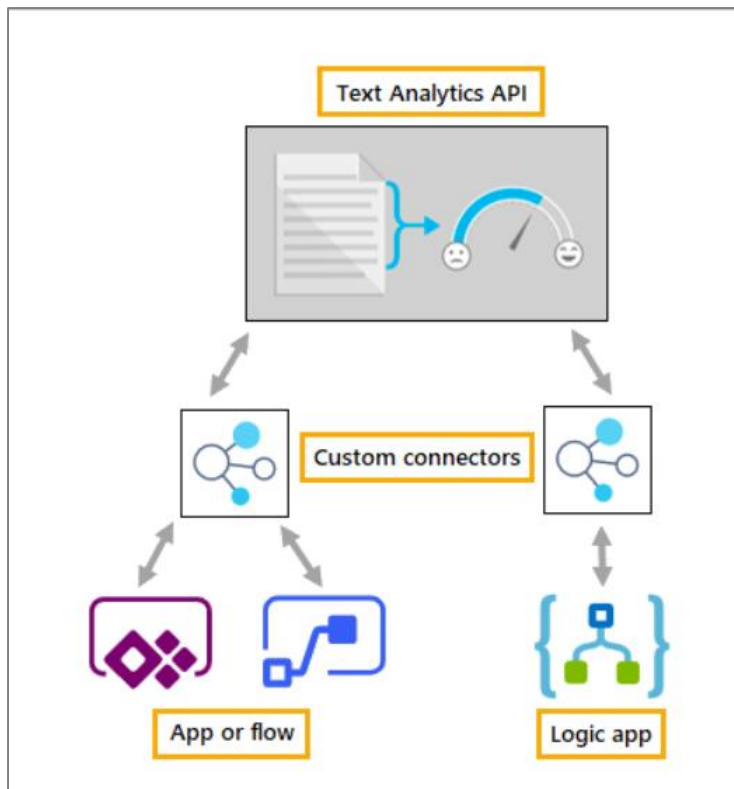
Agenda

- ✓ Canvas Apps
- ✓ PowerApps Components
- ✓ PowerApps Control Framework
- ✓ Flows
- Custom Connectors

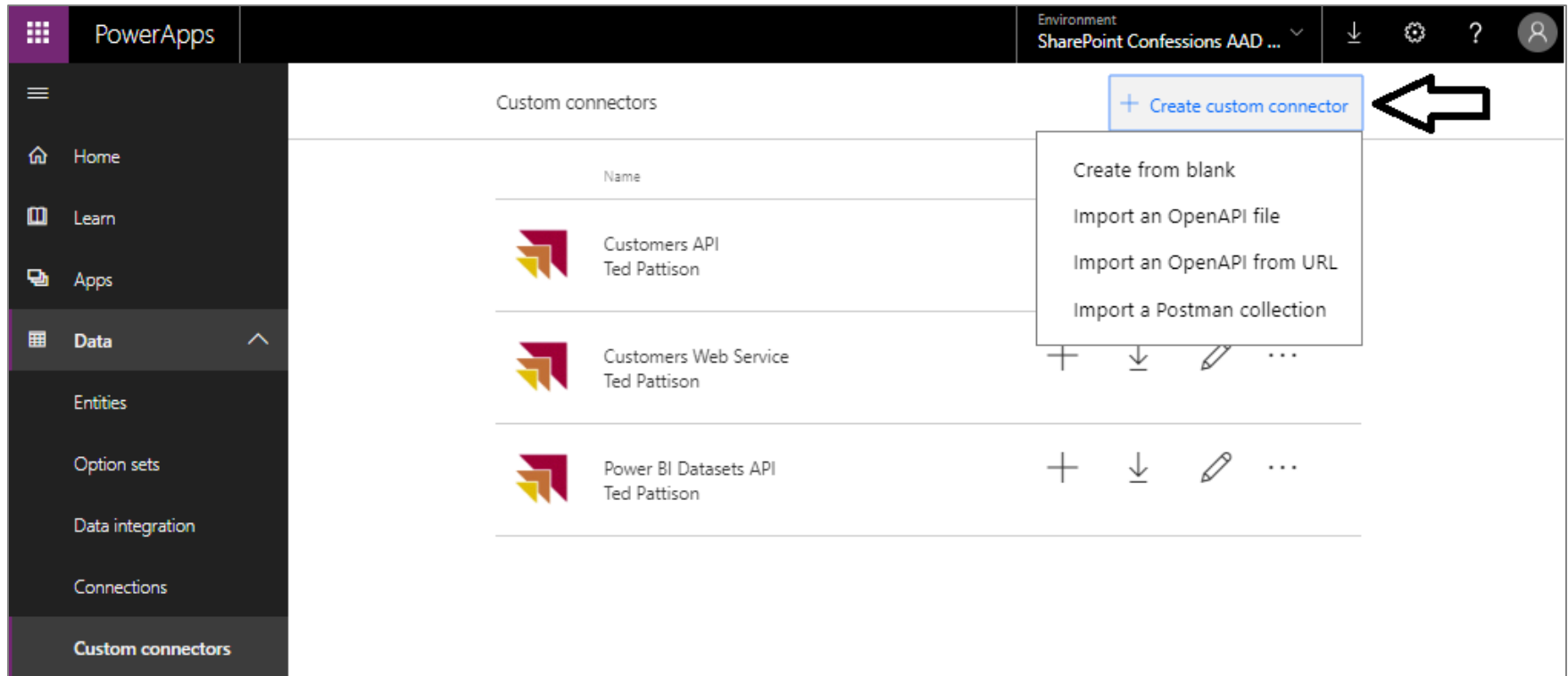


Standard Connectors vs Custom Connectors












- PowerApps Supports two types of connectors
 - Standard connectors supplied out-of-box and vetted by Microsoft
 - Custom connectors created by organizations for their own use



Creating a New Custom Connector



The screenshot shows the PowerApps interface. The top bar includes the 'PowerApps' logo, the environment 'SharePoint Confessions AAD ...', and icons for download, settings, help, and user. The left sidebar contains navigation links: Home, Learn, Apps, Data (selected), Entities, Option sets, Data integration, Connections, and Custom connectors. The main area is titled 'Custom connectors' and contains a table of existing connectors. A '+ Create custom connector' button is highlighted with a blue box and a large black arrow pointing to it. A dropdown menu is open from this button, showing four options: 'Create from blank', 'Import an OpenAPI file', 'Import an OpenAPI from URL', and 'Import a Postman collection'.

Name	
 Customers API Ted Pattison	
 Customers Web Service Ted Pattison	   
 Power BI Datasets API Ted Pattison	   



Defining Requests

Request + Import from sample

* Verb

The verb describes the operations available on a single path.

GET

* URL

This is the request URL.

http://subliminalsystems.com/api/Customers/

Path

Path is used together with Path Templating, where the parameter value is actually part of the operation's URL.

Query

Query parameters are appended to the URL. For example, in /items?id=####, the query parameter is id.

Headers

These are custom headers that are part of the request.

Body

The body is the payload that's appended to the HTTP request. There can only be one body parameter.

Request + Import from sample

* Verb

The verb describes the operations available on a single path.

GET

* URL

This is the request URL.

http://subliminalsystems.com/api/Customers({CustomerId})

Path

Path is used together with Path Templating, where the parameter value is actually part of the operation's URL.

* CustomerId ...

Query

Query parameters are appended to the URL. For example, in /items?id=####, the query parameter is id.

Headers

These are custom headers that are part of the request.

Body

The body is the payload that's appended to the HTTP request. There can only be one body parameter.



Defining the Response

← Connector Name Customers Web Service

1. General > 2. Security > 3. Definition > 4. Test

✓ Update connector ✕ Close

▼ Actions (2)

Actions determine the operations that users can perform. Actions can be used to read, create, update or delete resources in the underlying connector.

1 GetCustomers ...

2 **GetCustomer** ...

+ New action

▼ References (0)

References are reusable parameters used by both actions and triggers.

← Back

Response

+ Import from sample

* Name

default

Headers

These are custom headers that are part of the response.

References Used

The following are the references used by this entity

Body

The payload that is available on the response. These are the tokens that will show up as the outputs in designer.

Address ... BirthDate ... City ... Company ... CustomerId ...

EmailAddress ... FirstName ... FirstPurchaseDate ... Gender ...

HomePhone ... LastName ... LastPurchaseDate ... State ... WorkPhone ...

Zipcode ... odata.metadata ...

Validation

This helps you identify potential issues with this response.

Validation

✓

Validation succeeded.

← Security

Test →




Summary

- ✓ Canvas Apps
- ✓ PowerApps Components
- ✓ PowerApps Control Framework
- ✓ Flows
- ✓ Custom Connectors



Our Newest PowerApps Training Course

- Advanced PowerApps and Flow training
 - Designed for students who have already attended Microsoft's App-in-a-day
 - Three days filled with plenty of hands-on lab exercises
 - Learn advanced builder skills & best practices
 - More info at <https://CriticalPathTraining.com>



[Training](#)[Courses](#)[Schedule](#)[Video Gallery](#)[Competition](#)[Contact Us](#)

[Home](#) > [Training Courses](#) > [Business Users](#)

Building Business Solutions with PowerApps and Flow

[Course Overview \(PDF\)](#)

Building Business Solutions with Power Apps and Flow is an intensive 3-day training class designed for people who have already attended the Microsoft App-in-a-Day (AIID) training and are ready to move their PowerApps and Flow skills to the next level. Students will learn best practices for building canvas apps and flows to update and manage content in Azure SQL, SharePoint Online, Excel workbooks and OneDrive for Business.

After providing a foundation for building canvas apps and flows, this course teaches students how to build business solutions with the Common Data Service for Apps (CDSA) by creating custom entities and model-driven apps. The class also examines advanced techniques required in real-world PowerApps scenarios such as creating custom connectors and conducting application lifecycle management (ALM) using managed solutions and multiple environments for development, UAT and production.

[Student Prerequisites](#)

Upcoming Offerings

Date	Location	Instructor	Action
Apr-15	Tampa, FL	Ted Pattison	Register
Jun-24	Tampa, FL	Ted Pattison	Register

Course Details

Course Code	BBSPA
Course Version	1.0
Course Length	3 Days

