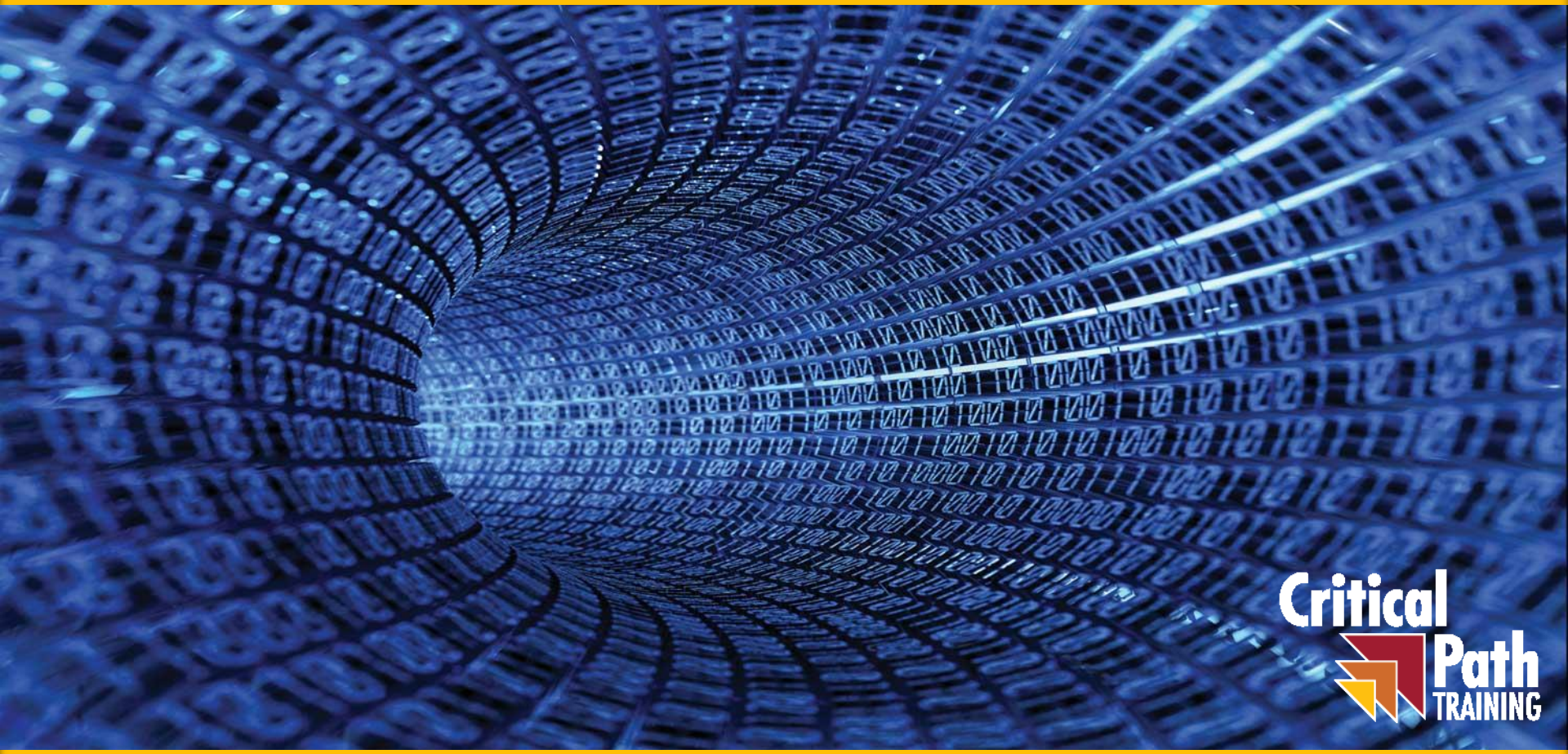
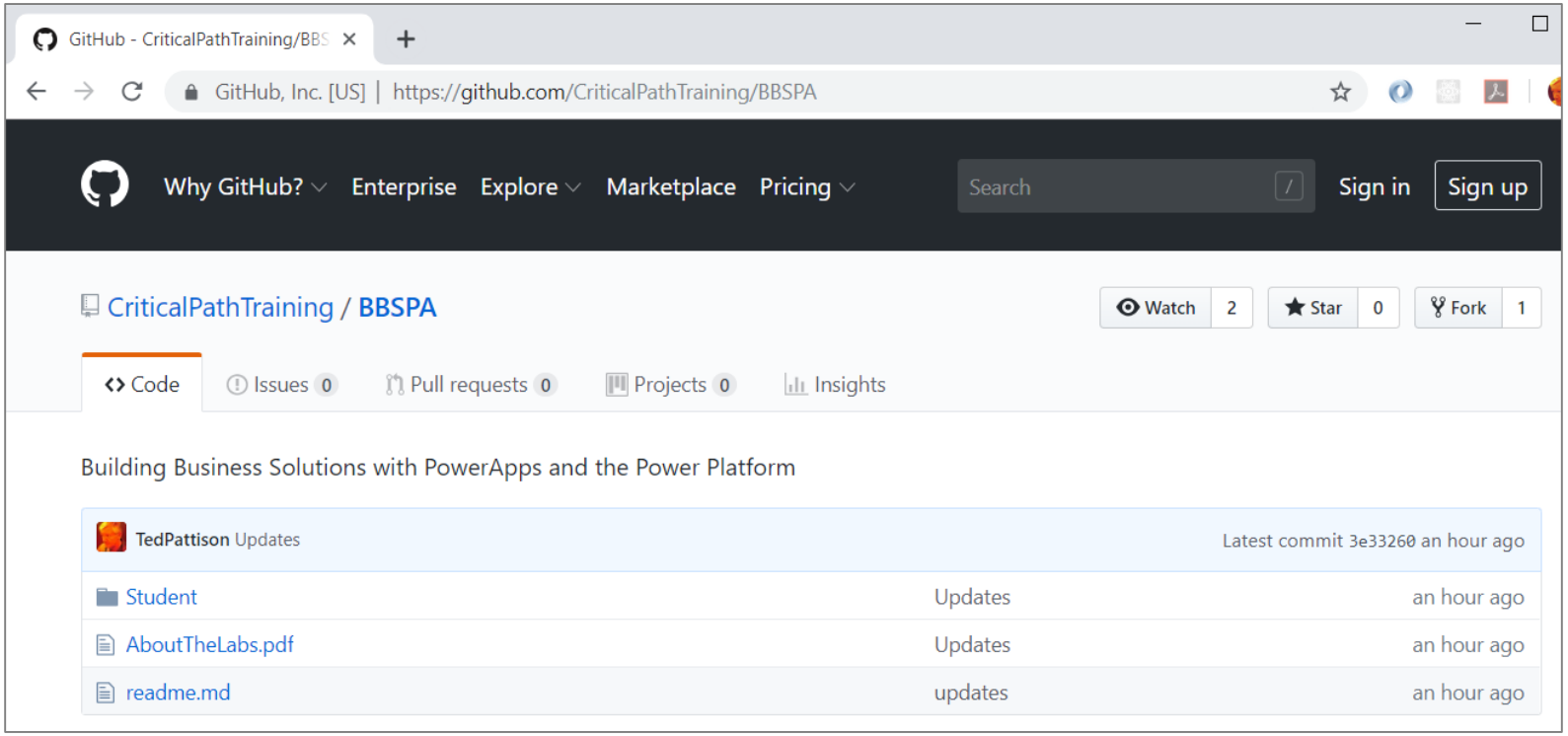
Getting Started with the Power Platform



Downloading Student Files

- Student files maintained in a GitHub repository
  - https://github.com/CriticalPathTraining/BBSPA



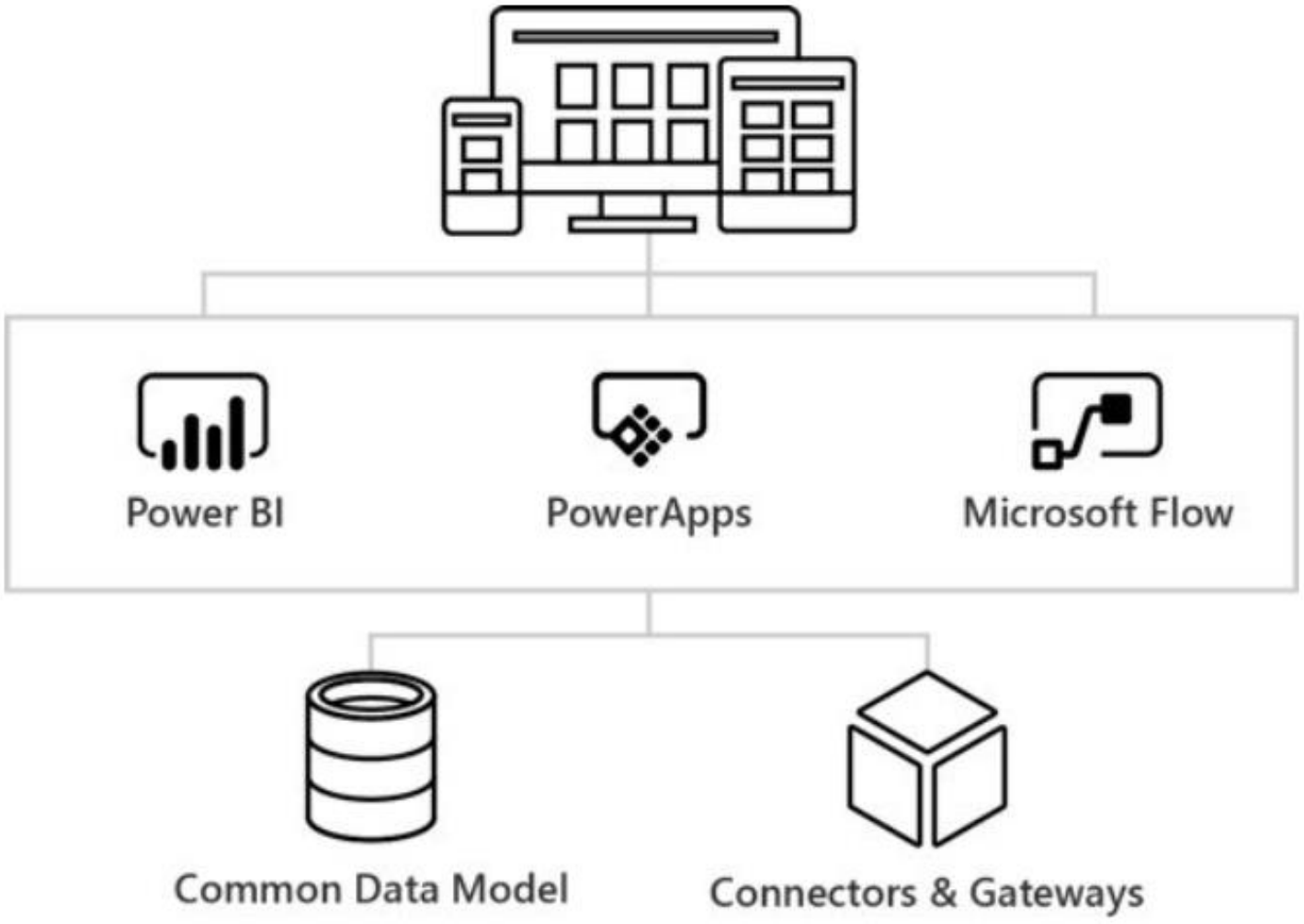
Student Background Questionnaire

- What is your name?
- What are you doing with PowerApps and Flow?
- Which products and services have you used?
  - PowerApps and Flow
  - Microsoft Excel
  - Office 365
  - SharePoint Online
  - Power BI
  - Dynamics 365
  - Others

Agenda

- Getting Started with the Power Platform
- Creating Canvas Apps
- Writing PowerApps Expressions
- Working with Connectors and Data Binding
- Understanding Delegation

What is the Power Platform?

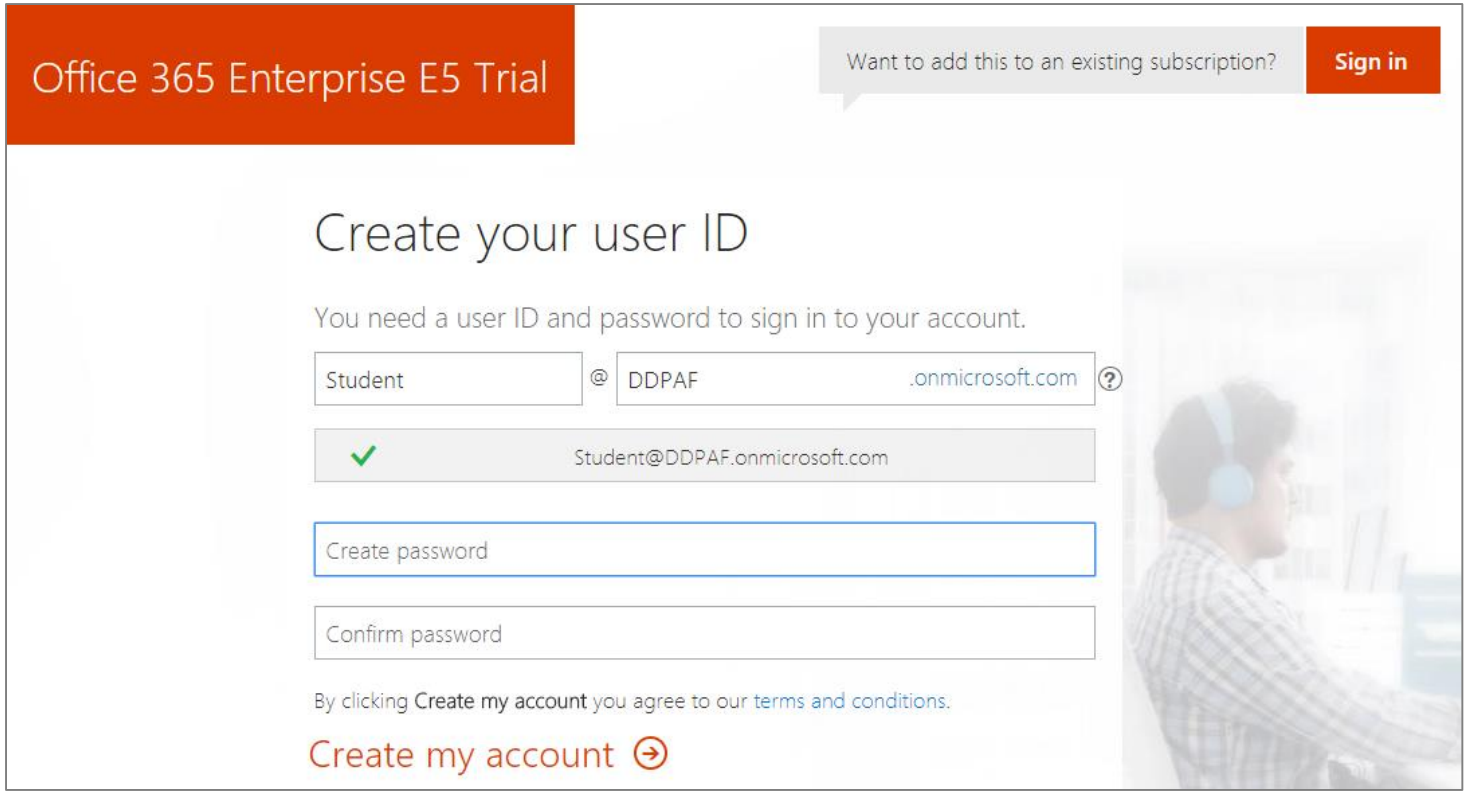


What Can You Build with PowerApps?

- Canvas Apps
  - Built using PowerApps Studio
- Connections
  - Used to connect Canvas apps to external data
- Flows
  - Used to process data and run workflows
- Common Data Service for Apps (CDS for Apps)
  - Used to create business-centric data solutions
- Model-driven Apps
  - Application platform built on top of CDS for Apps

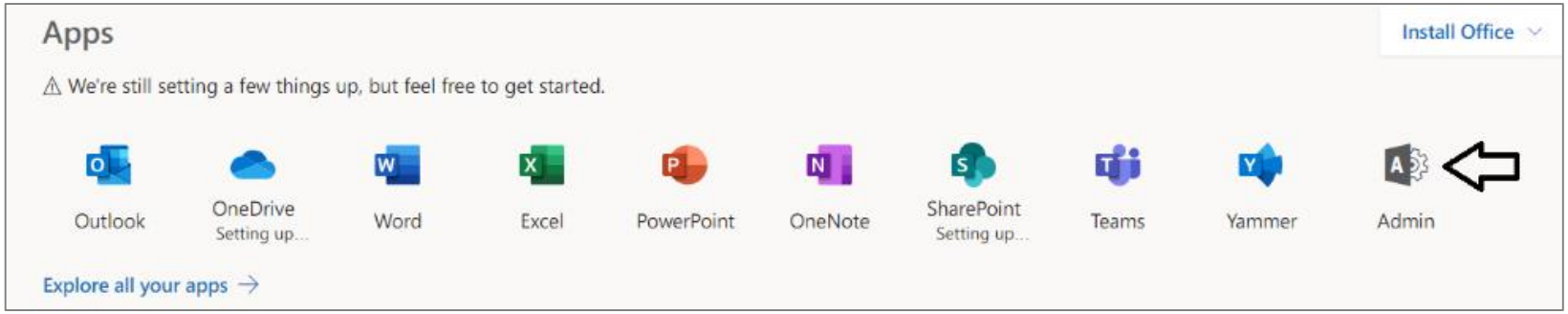
Creating an Office 365 E5 Trial Tenant

- All students will create an Office 365 trial tenant
  - Provides an isolated development environment for lab exercises
  - Trial accounts will last for 30 days

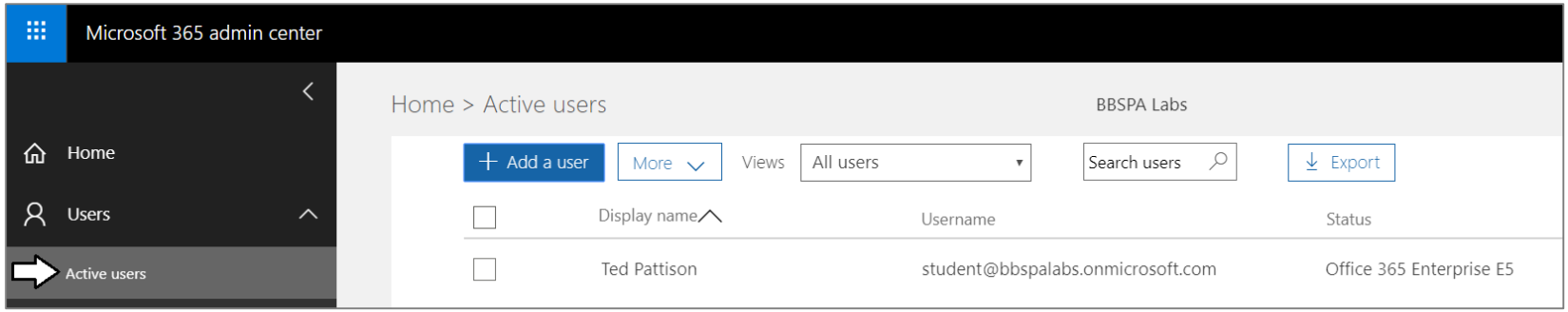


Microsoft 365 Admin Center

- Navigate to the Microsoft 365 Admin center

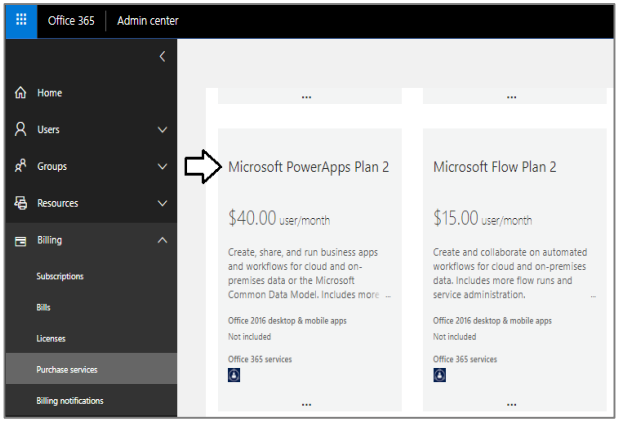


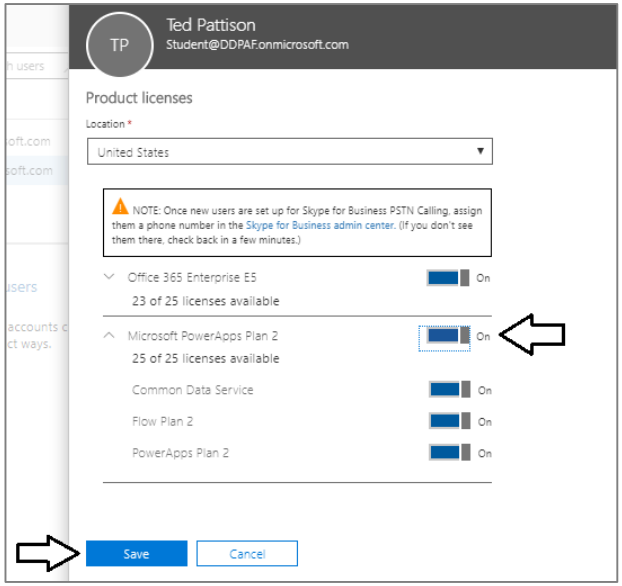
- Allows for management of users accounts and licensing



Configuring a PowerApps Plan 2 License

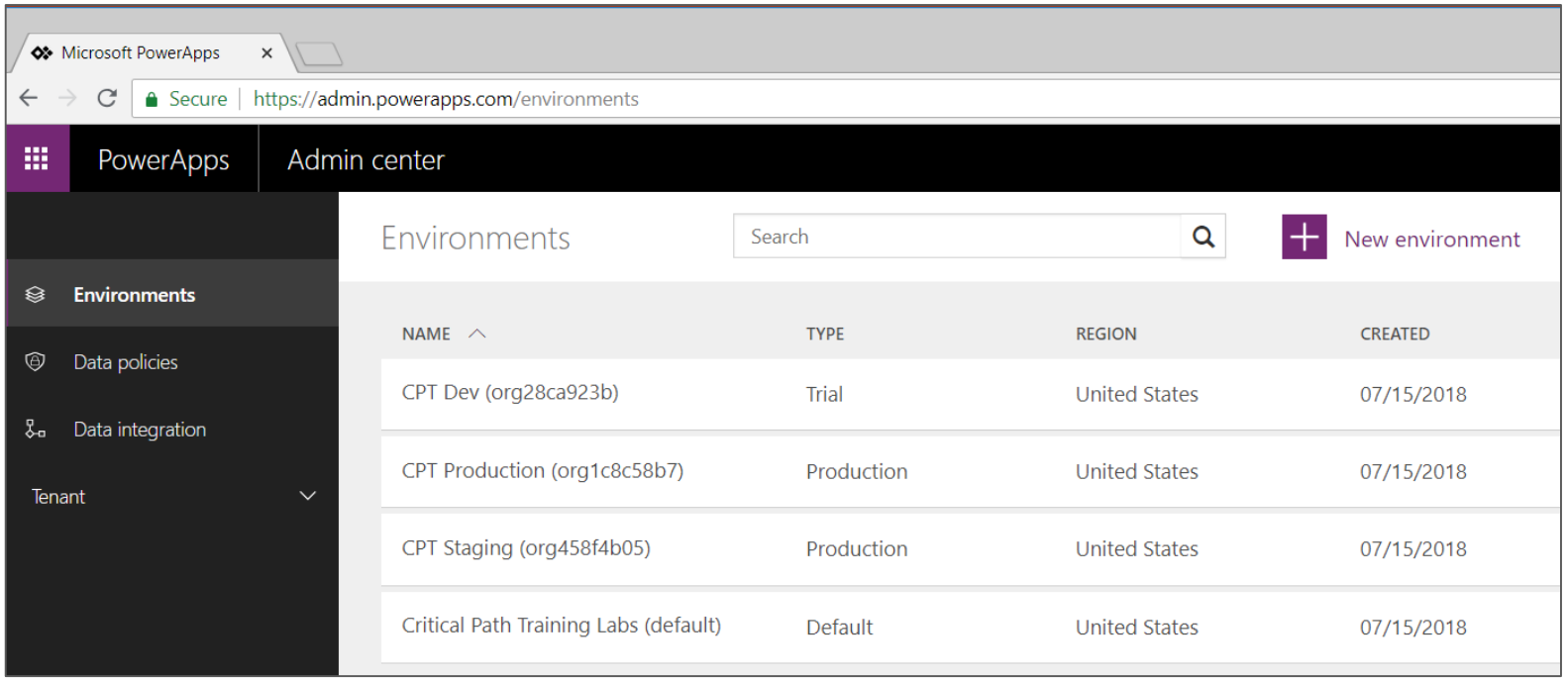
- Certain design tasks require PowerApps Plan 2
  - You can start a 30-day trial for PowerApps Plan 2
  - License must be assigned to individual user accounts





PowerApps Admin Center & Environments

- PowerApps architecture based on environments
  - Environment provides context for creating apps and flows
  - Every tenant is automatically created with default environment
  - Organization can create multiple environments for dev & staging
  - PowerApps Plan 2 license required to manage environments



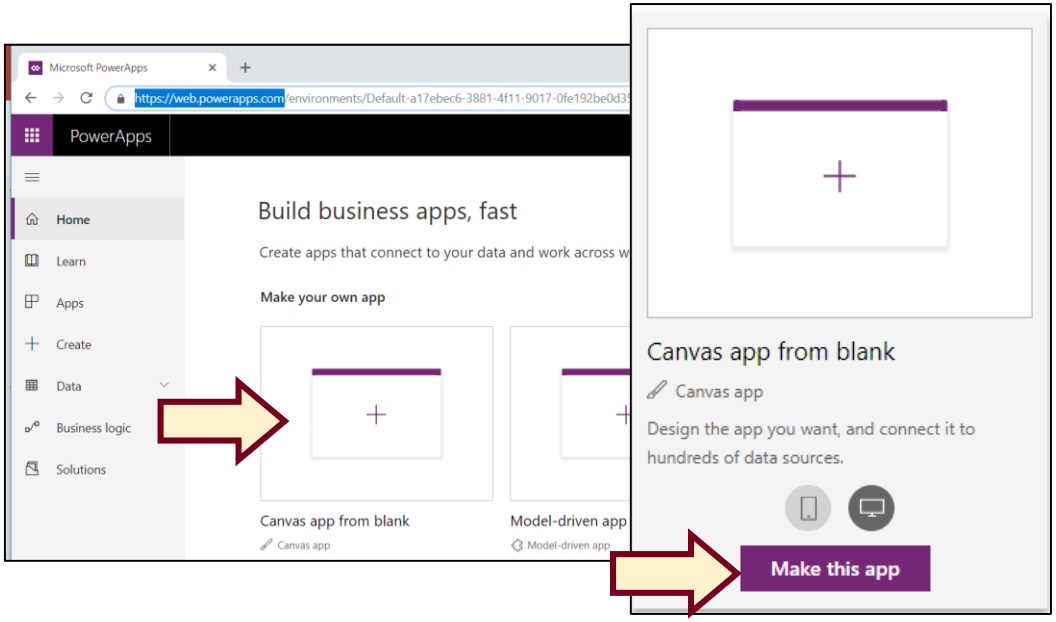

DEMO
Configuring PowerApps Plan 2 Licenses

Agenda

- ✓ Getting Started with the Power Platform
- ➢ Creating Canvas Apps
- Writing PowerApps Expressions
- Working with Connectors and Data Binding
- Understanding Delegation

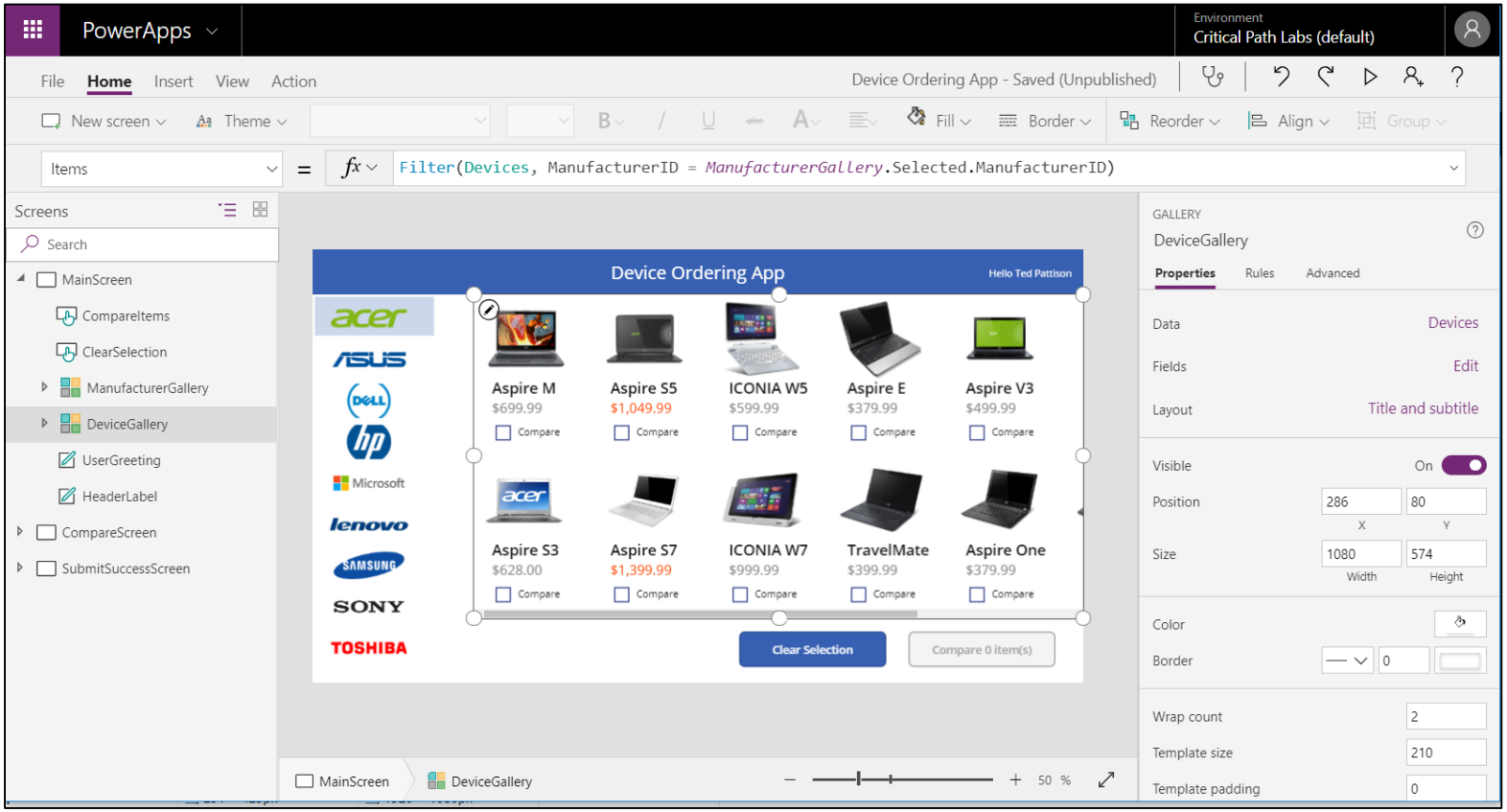
Creating a New Canvas App

- Create Canvas apps from the PowerApps Home page
  - Navigate to https://web.powerapps.com
  - Chose **Canvas app from blank** or **Start from data**
  - Choose between **Phone** form factor and **Desktop/Tablet** form factor
  - Clicking **Make this app** redirects browser to **https://create.powerapps.com**



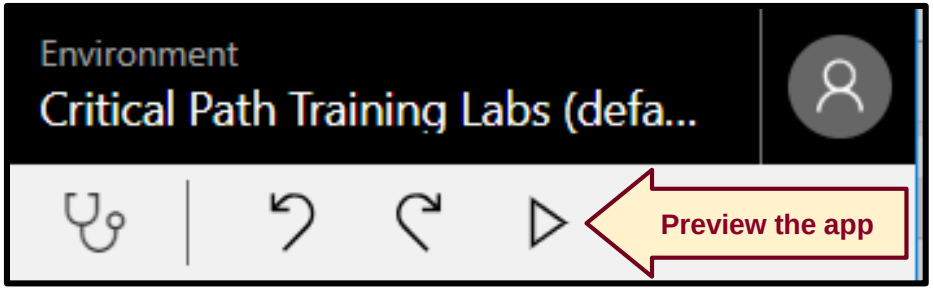
Getting Started with PowerApps Studio

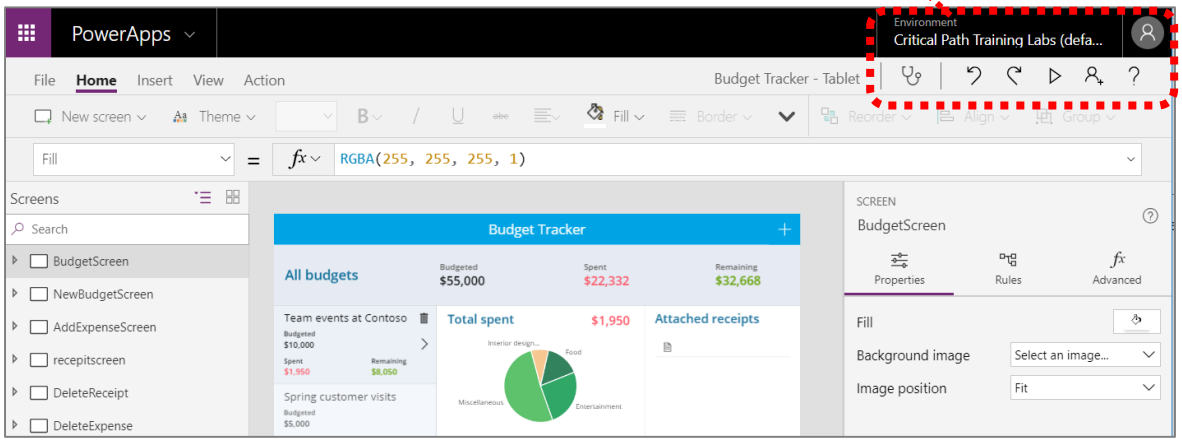
- PowerApps Studio for the Web is used to build apps
  - Environment supported across platforms (Windows & Mac)
  - Supports all popular, modern browsers



Running an App from PowerApps Studio

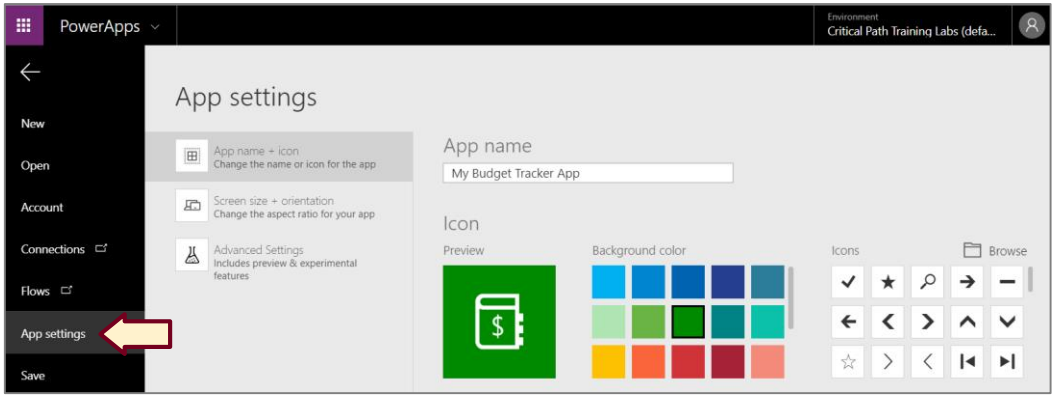
- You can run the app using the PowerApps Studio toolbar
  - Run the app by clicking the **Preview the App** button
  - Stop a running app to return to design mode



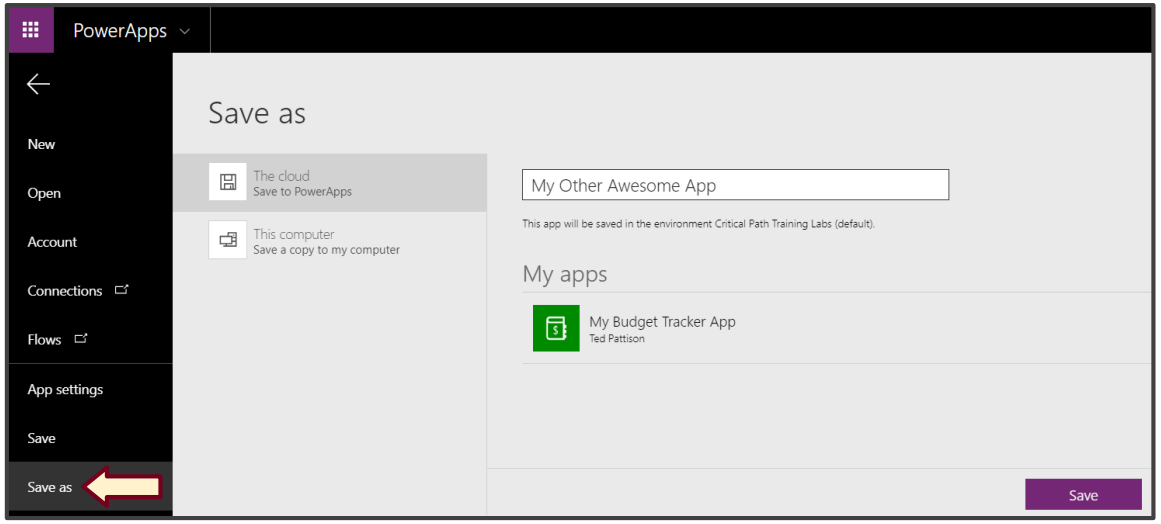


Saving an App to the Cloud

- Before saving, first you should configure App settings



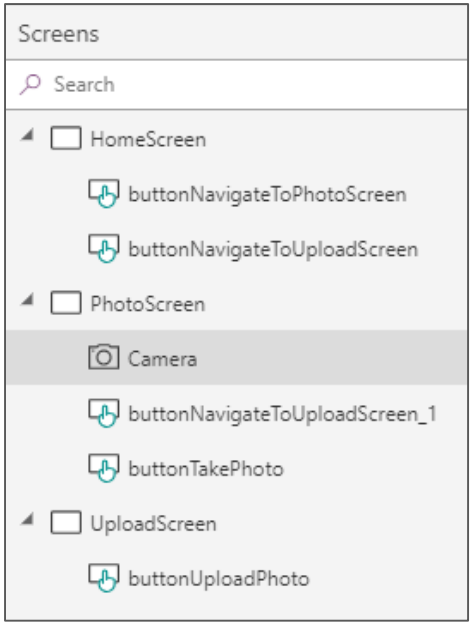
- Save app to cloud using Save or Save As command




DEMO
Creating an App with the Budget Tracker App Template

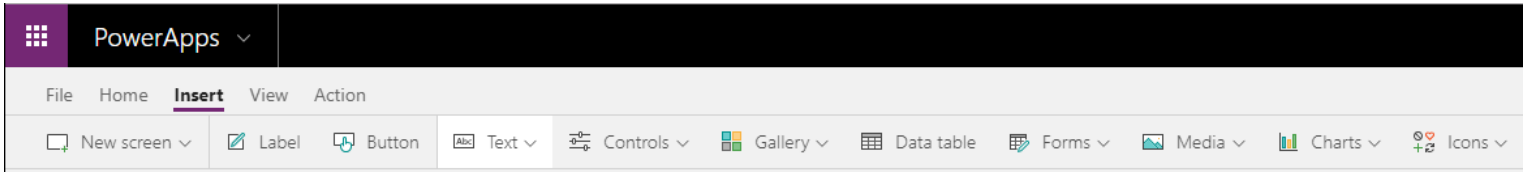
Building Apps using Screens and Controls

- Screens provide the top-level objects in PowerApps UI
  - Your app must have one screen but can have multiple screens
  - You design screens by adding and configuring controls
  - Left tree view shows hierarchical view of screens and controls
  - You can rename screens and controls using left tree view menu

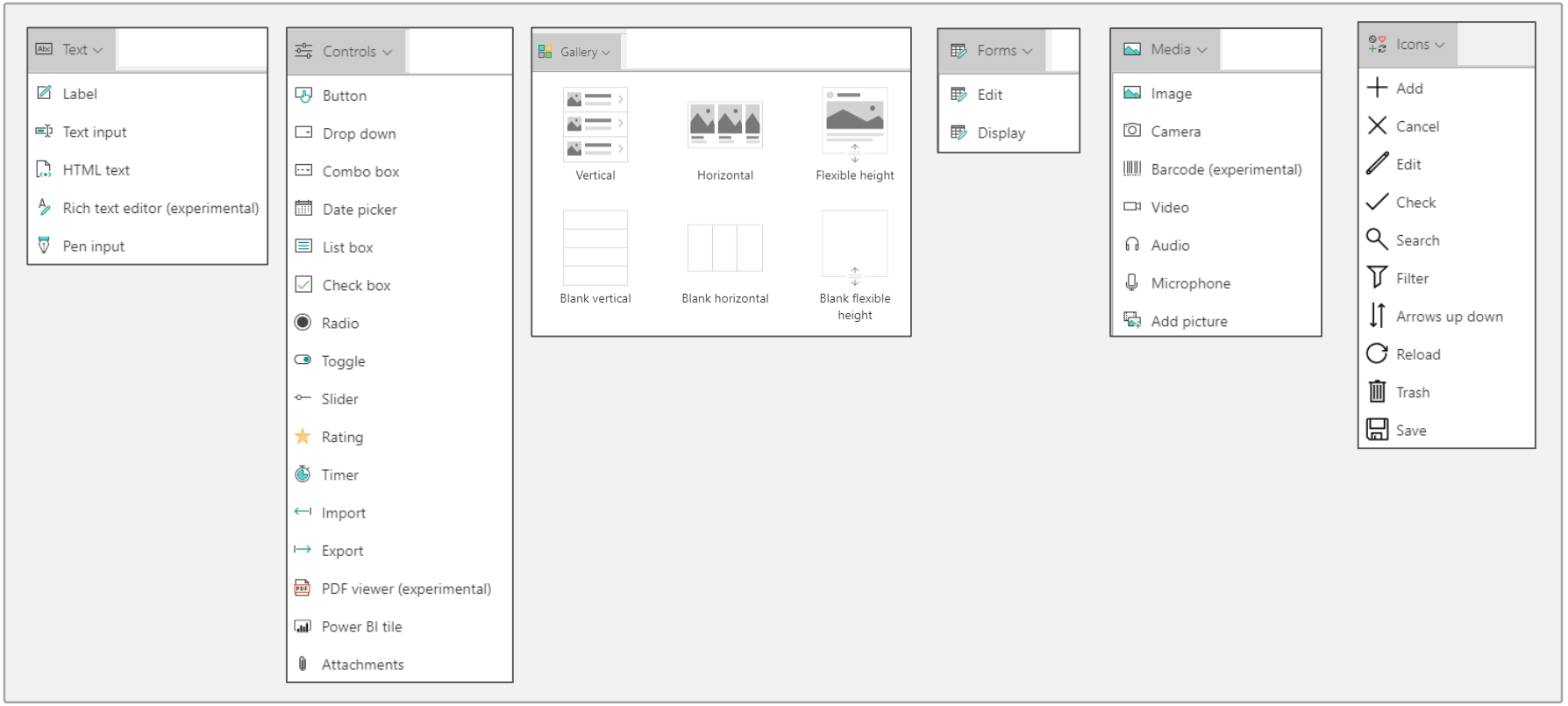


Adding Controls to a Screen

- You add controls to a screen using the **Insert** ribbon tab



- PowerApps provides extensive set of controls for web and mobile apps

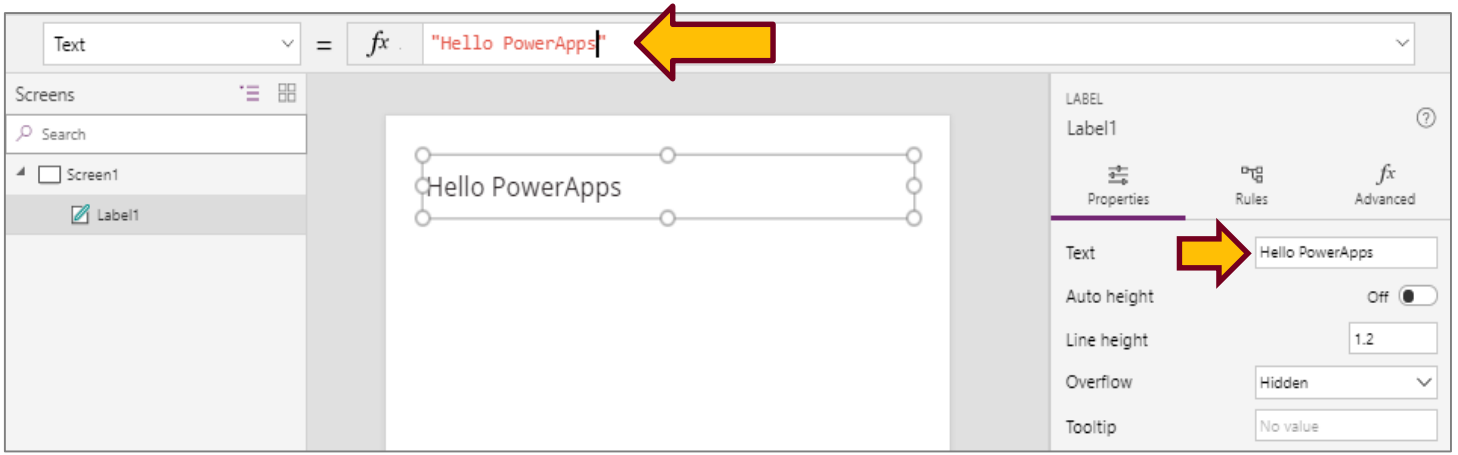


Agenda

- ✓ Getting Started with the Power Platform
- ✓ Creating Canvas Apps
- ➢ Writing PowerApps Expressions
- Working with Connectors and Data Binding
- Understanding Delegation

Configuring Control Properties

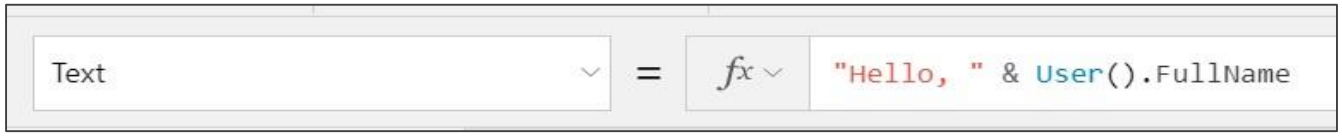
- Control properties can be set two different ways
  - Property values can be set using Properties pane
  - Property values can be set using Formula bar



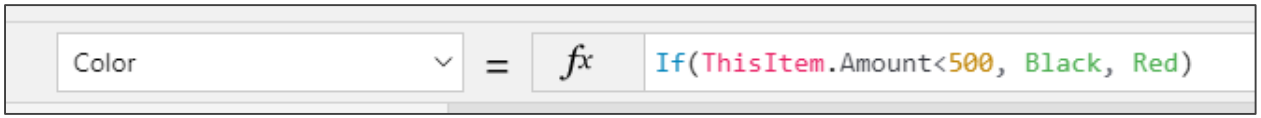
- Building apps with PowerApps requires shift in thinking
  - You don’t write code to set property values like in VBA
  - Control properties configured using formulas
  - You develop using **declarative style** instead of **procedural style**

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

Input and Output Properties

- Input only
  - Value based on formula which cannot be used in other formulas
  - `Textinputbox1.Default` *(initial value)*
- Input/Output
  - Value based on formula which can be used in other formulas
  - `Textinputbox1.Fill`
- Output only
  - Value cannot be set by formula but can be used on other formulas
  - `Textinputbox1.Text` *(value always controlled by user input)*

Primitive Data Types

- Number: 3.141592
- Text: "Hello World”
- Boolean: True or False
- DateTime: 3/27/2018 12:00PM
- Date: 3/27/2018

- Any type can contain blank (i.e. null) values
  - Test for null value using IsBlank() function
  - Set null value using Blank() function

Compound Data Types

- Record

```
{ FirstName: "Chuck", LastName: "Sterling" }
```

- Table

```
Table( { FirstName: "Chuck", LastName: "Sterling" },
       { FirstName: "Ted", LastName: "Pattison" } )
```

- Shorthand for Table with one column named value

```
[ "Moe", "Curly", Larry" ]
```

- Records and tables can be nested
  - Table can contain records of tables of records…
  - Record can contain tables of records of tables …

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 (;)

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

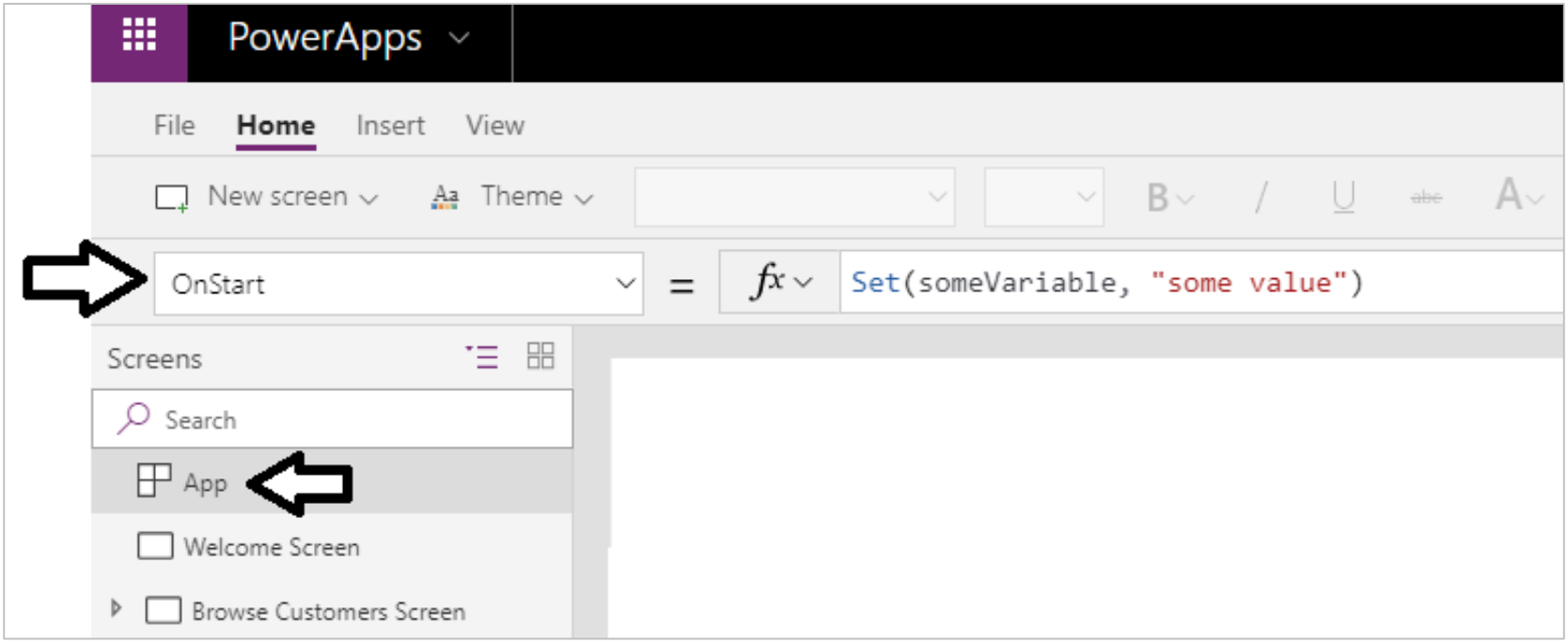
Declarative vs Imperative Functions

- 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	RemoveIf	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	

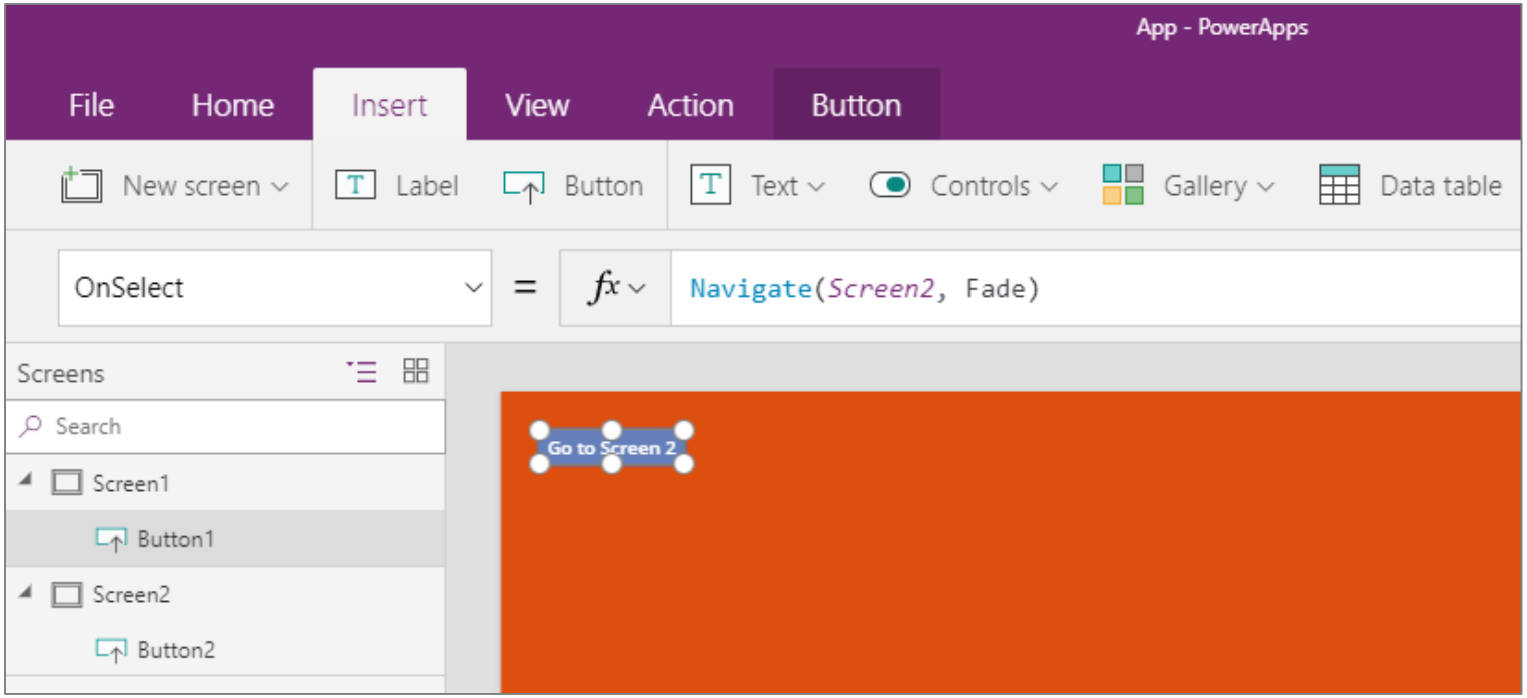
App OnStart

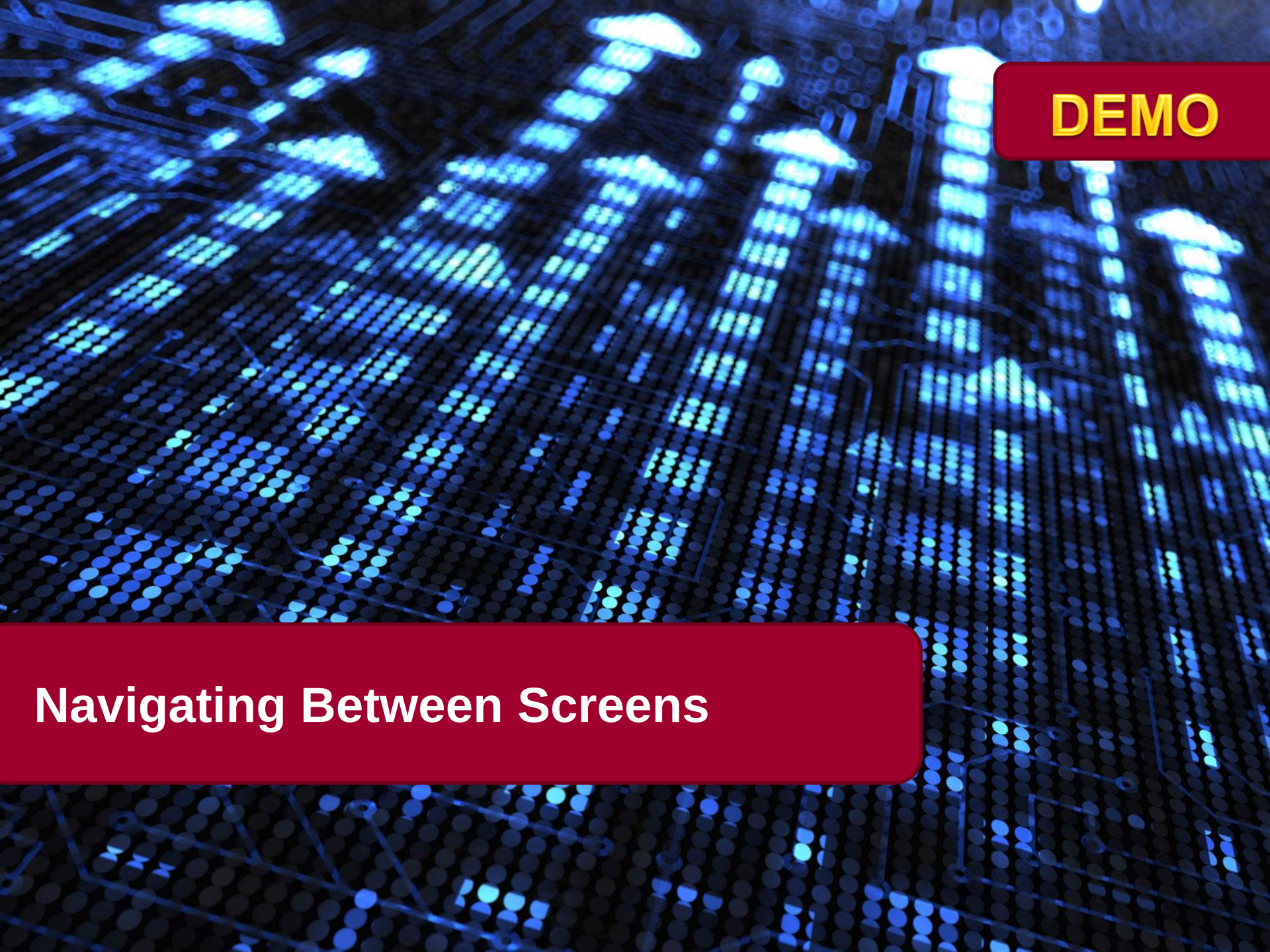
- App **OnStart** property used to initialize state in app
  - Event provides support for initializing state in app
  - Commonly used to initialize app-level variables and collections
  - Right-click **App** in left navigation to run **OnStart** while in editor



Navigating Between Screens

- Navigate to a screen using Navigate function
  - Call **Navigate** function from **OnSelect** property of **Button** control
  - **Navigate** function performs action instead of returning a value



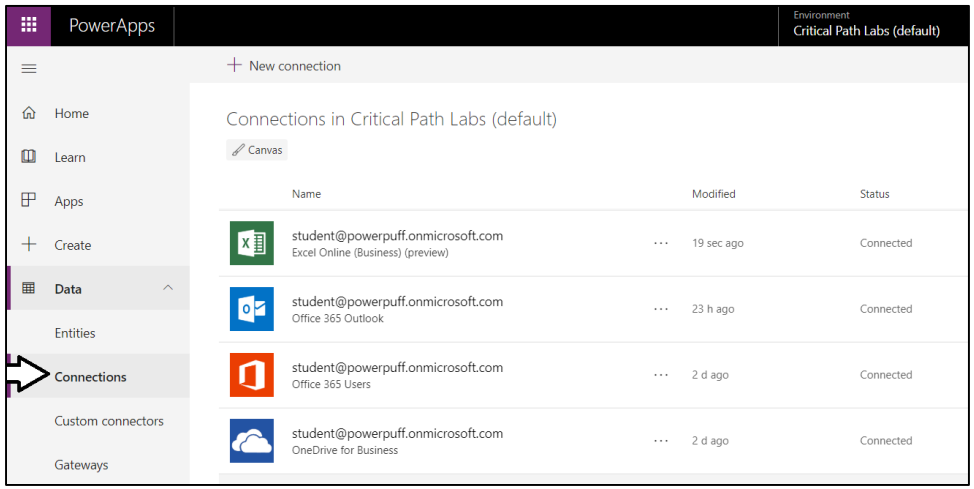

DEMO
Navigating Between Screens

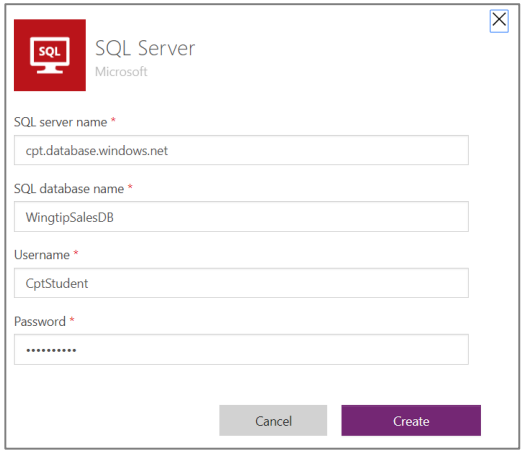
Agenda

- ✓ Getting Started with the Power Platform
- ✓ Creating Canvas Apps
- ✓ Writing PowerApps Expressions
- ➢ Working with Connectors and Data Binding
- Understanding Delegation

Understanding Connectors & Connections

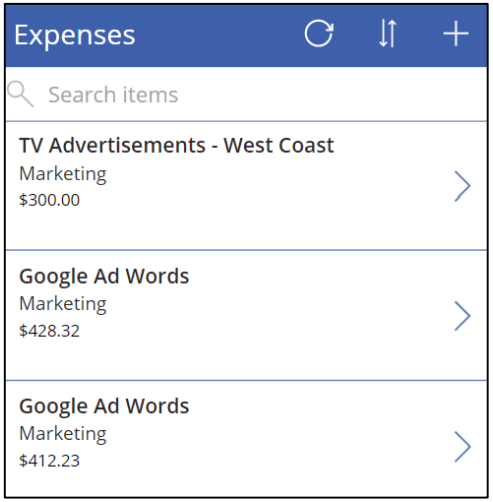
- What is a Connector?
  - API wrapper that PowerApps uses to interact with datasource
- What is a Connection?
  - Configuration created to connect to a specific datasource
  - Each connection is created using a specific connector
  - Connection also caches login credentials and granted permissions
  - Connections can be shared across users



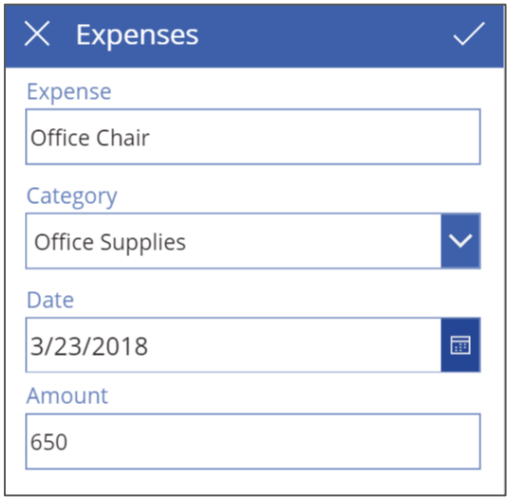


Data Binding with Galleries and Forms

- Table binding
  - Gallery control
  - DataTable control

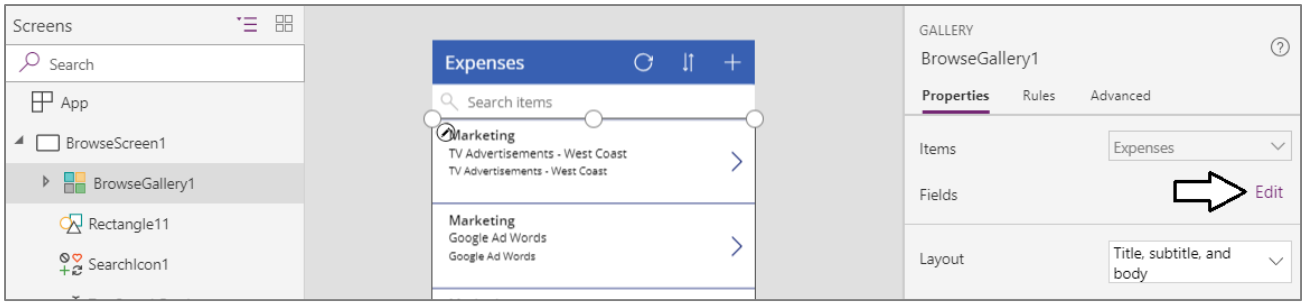


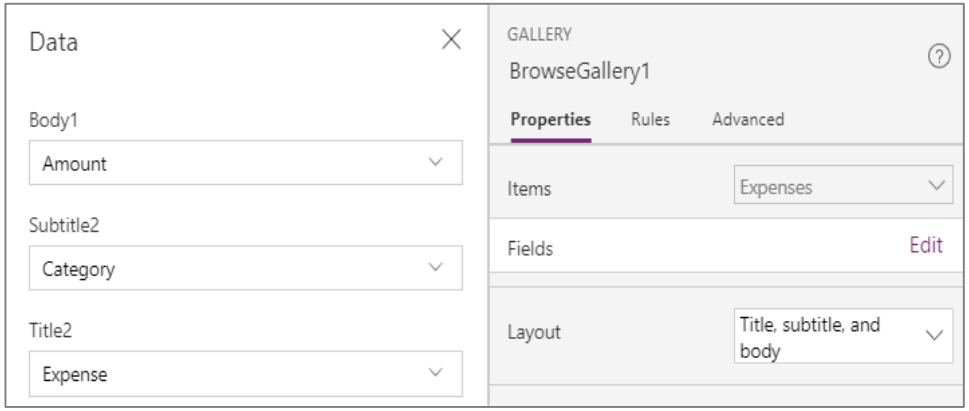
- Single-record binding
  - Display form control
  - Edit form control



Working with the Data Pane

- You use the Data pane to configure data binding
  - Select a data-bound control and then display Data pane
  - Data pane allows you to change layout for data binding
  - Once you select layout, you can then map fields below

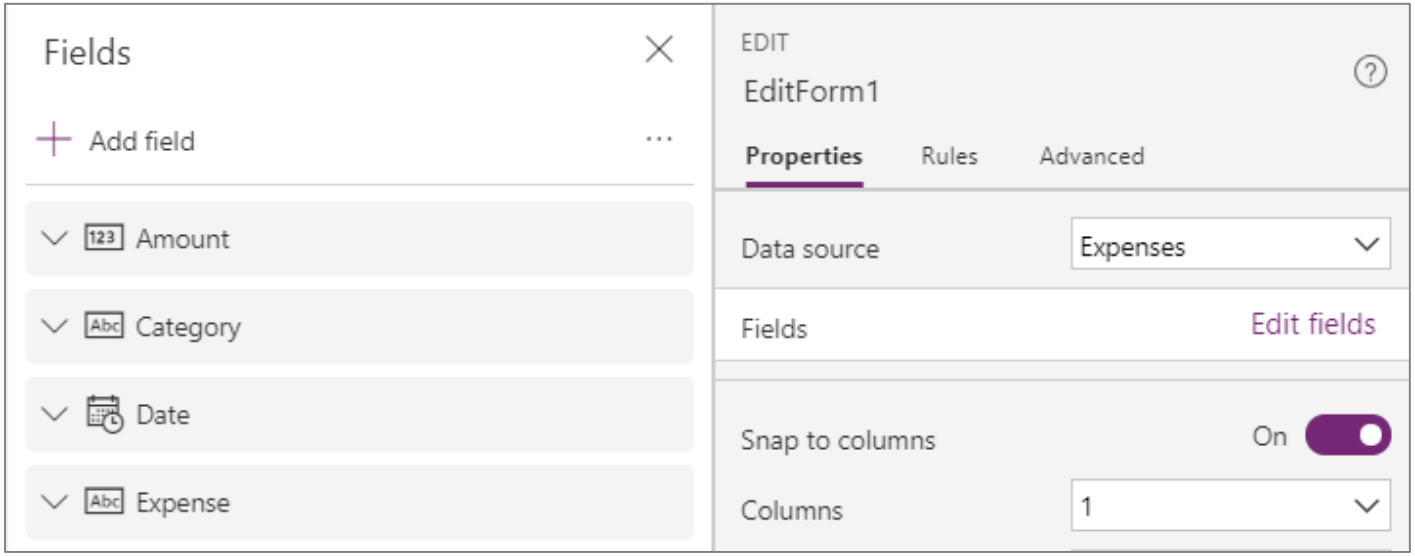





DEMO
Creating an App using the
Start from Data Template

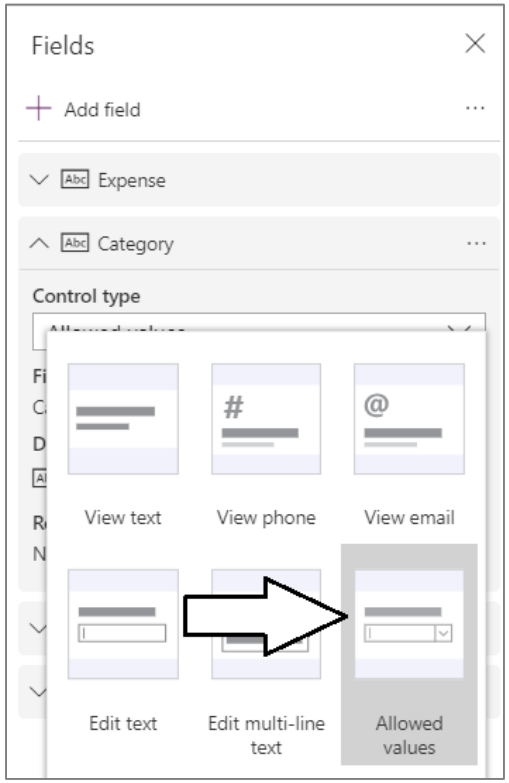
Understanding Forms and Data Cards

- 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



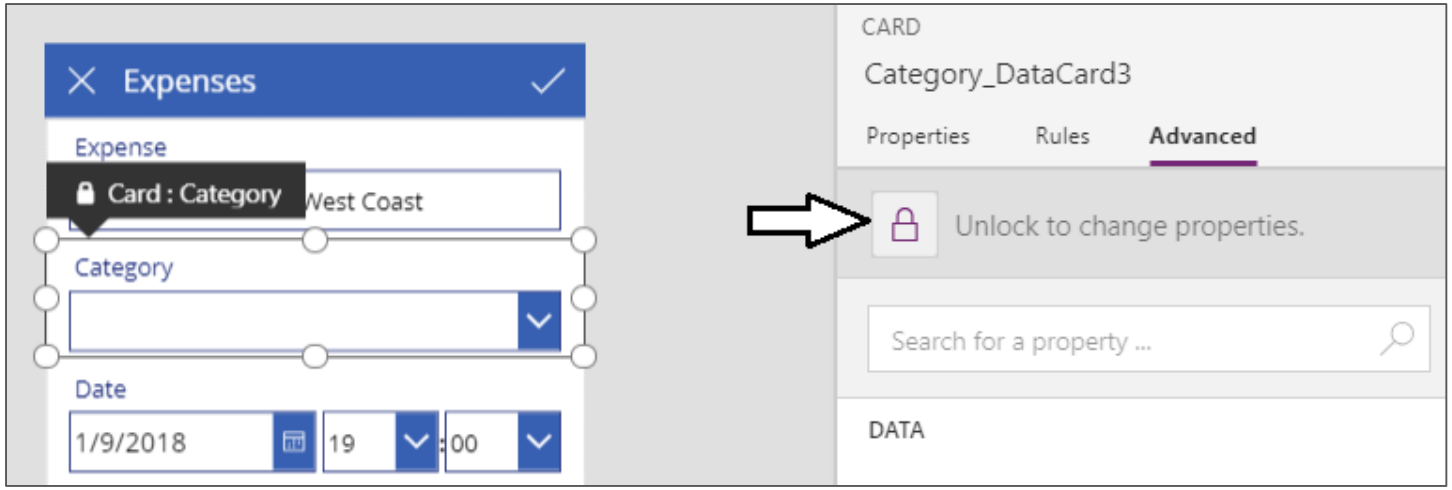
Changing a Field's Data Card Type

- Fields in Form control get default data card
  - Use Data pane to change data card used by any field
  - Different data cards offer different editing experiences



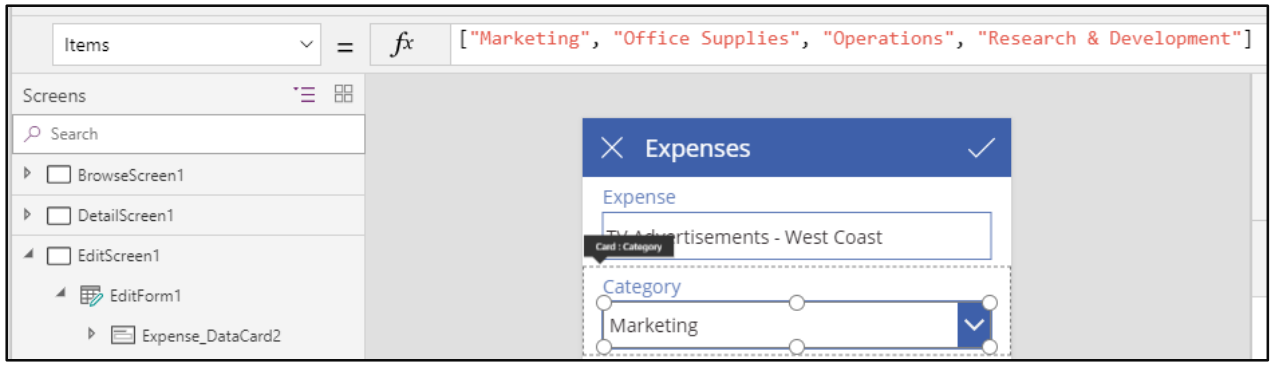
Customizing a Data Card

- By default, data cards are locked and cannot be edited
  - In many scenarios, you should leave data cards locked
  - Some scenarios call for unlocking data cards to customize them



Populating a Dropdown Combobox

- Once a data card is unlocked you can customize it
  - Add formula for **Items** property to populate combo box



  - Dropdown list provides better user experience than textbox


DEMO
Customizing Forms and Data Cards

Agenda

- ✓ Getting Started with the Power Platform
- ✓ Creating Canvas Apps
- ✓ Writing PowerApps Expressions
- ✓ Working with Connectors and Data Binding
- ➢ Understanding Delegation

Understanding Delegation

- **Delegation** is act of pushing work to data source
  - Work usually involves filtering and sorting
  - Work can also involve aggregation
  - Delegation minimizes amount of data retrieved by app
- Not all connectors support delegation
  - Non-delegate-able connectors only return 500 items
  - Searching through data becomes unpredictable

Delegate-able Functions

- **Filter functions**
  - *Filter*, *Search*, and *LookUp* can be delegated
- **Sorting functions**
  - *Sort* and *SortByColumns* can be delegated
- **Aggregate functions**
  - *Sum*, *Average*, *Min*, and *Max* can be delegated
  - Not all data sources support this delegation

https://powerapps.microsoft.com/en-us/tutorials/delegation-list/

Summary

- ✓ Getting Started with the Power Platform
- ✓ Creating Canvas Apps
- ✓ Writing PowerApps Expressions
- ✓ Working with Connectors and Data Binding
- ✓ Understanding Delegation