

Getting Started with Microsoft Flow

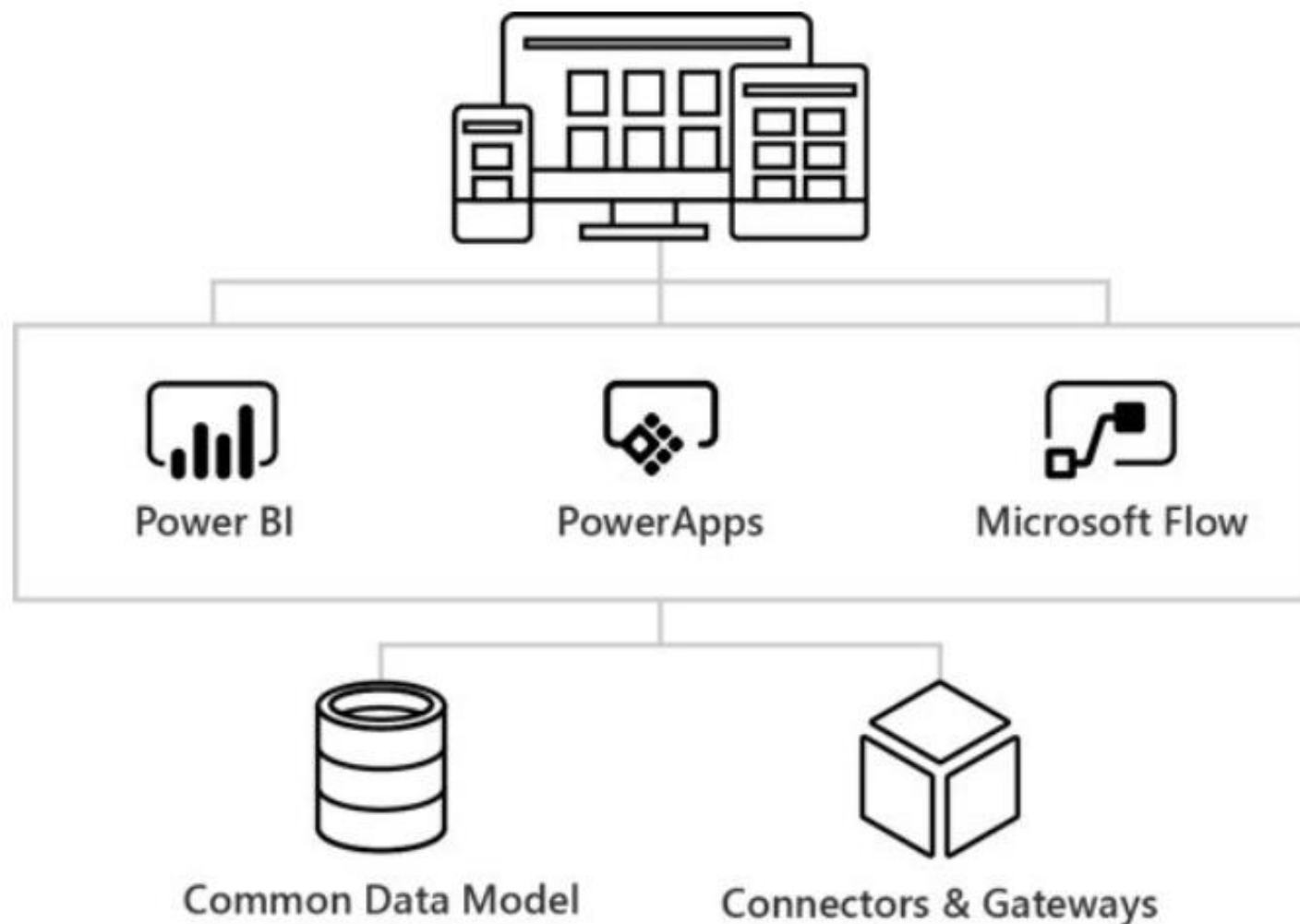


Agenda

- Getting Started with the Power Platform
- Building Flows using Triggers and Actions
- Using Control-of-Flow Actions
- Writing Advanced Flow Expressions
- Executing a Flow from a Canvas App



What is the Power Platform?



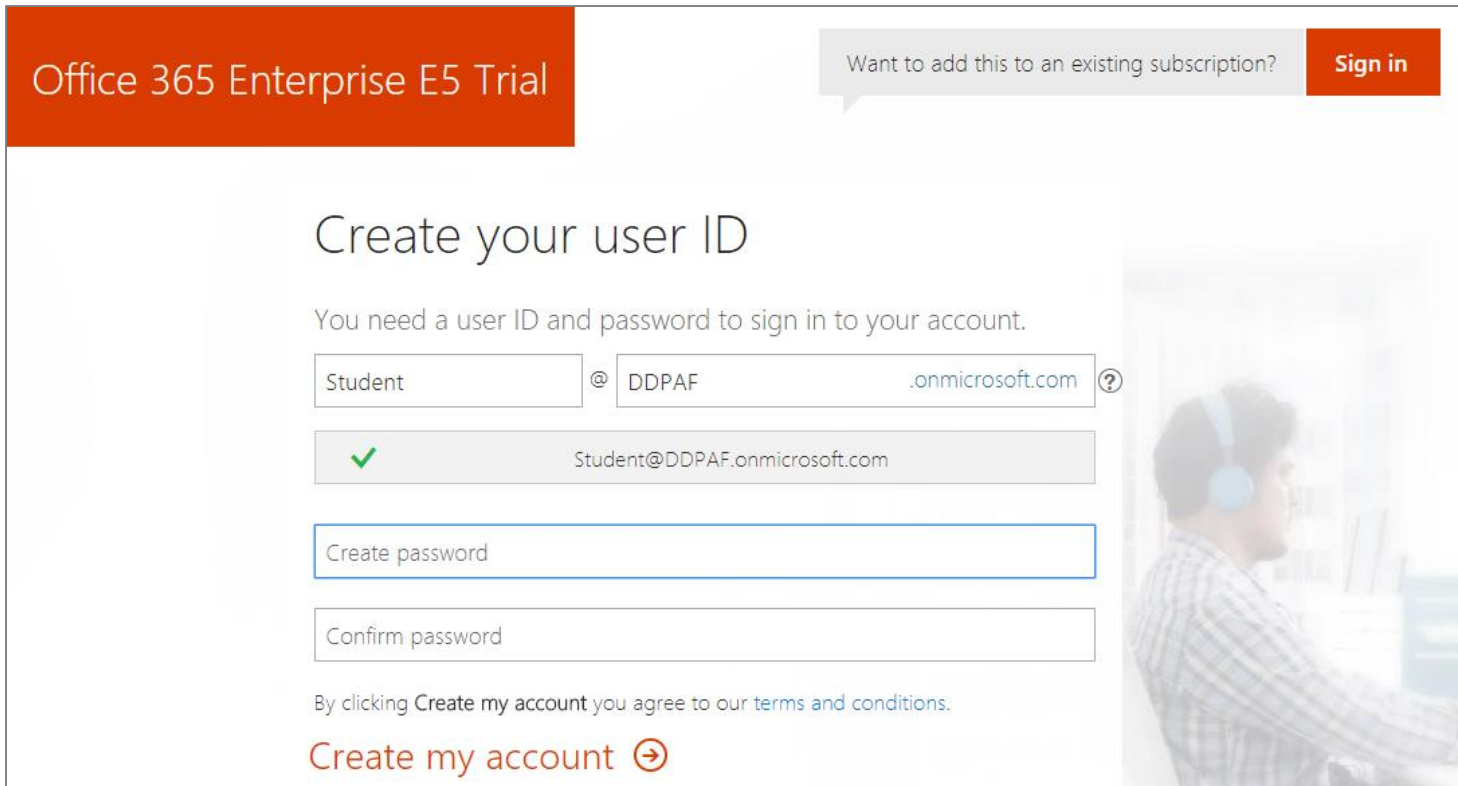
What Can You Build with PowerApps?

- 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



The screenshot shows the 'Create your user ID' page for an Office 365 Enterprise E5 Trial. At the top left, there is an orange banner with the text 'Office 365 Enterprise E5 Trial'. At the top right, there is a grey button that says 'Sign in' and a link that says 'Want to add this to an existing subscription?'. The main heading is 'Create your user ID'. Below this, it says 'You need a user ID and password to sign in to your account.' There are three input fields: the first contains 'Student', the second contains '@ DDPAF' followed by a help icon, and the third contains '.onmicrosoft.com'. Below these fields, there is a green checkmark icon and the text 'Student@DDPAF.onmicrosoft.com'. There are two more input fields: 'Create password' and 'Confirm password'. At the bottom, there is a link that says 'By clicking Create my account you agree to our terms and conditions.' and a red button that says 'Create my account' with a right arrow icon. On the right side of the page, there is a blurred image of a person wearing a headset and working at a computer.

Office 365 Enterprise E5 Trial

Want to add this to an existing subscription? [Sign in](#)

Create your user ID

You need a user ID and password to sign in to your account.

Student @ DDPAF .onmicrosoft.com ?

✓ Student@DDPAF.onmicrosoft.com

Create password

Confirm password

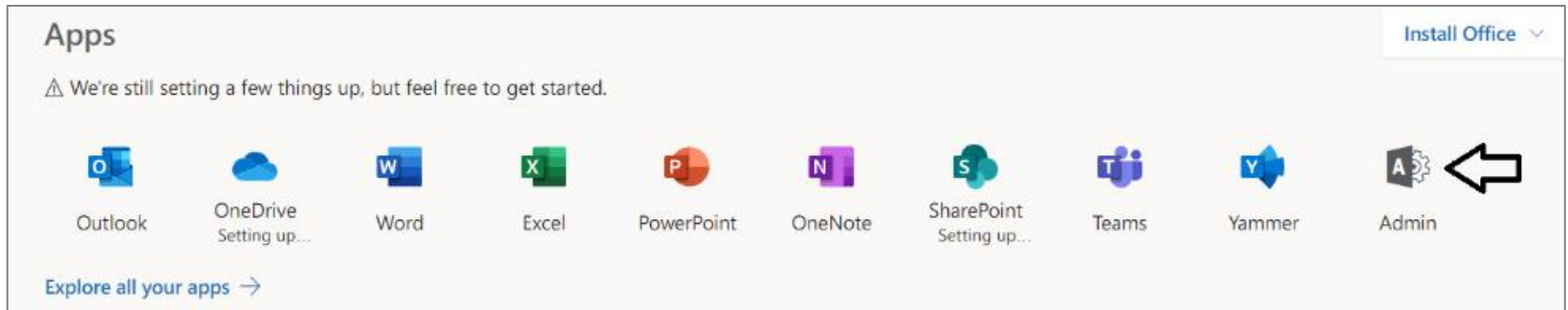
By clicking [Create my account](#) you agree to our [terms and conditions](#).

[Create my account](#) ➔

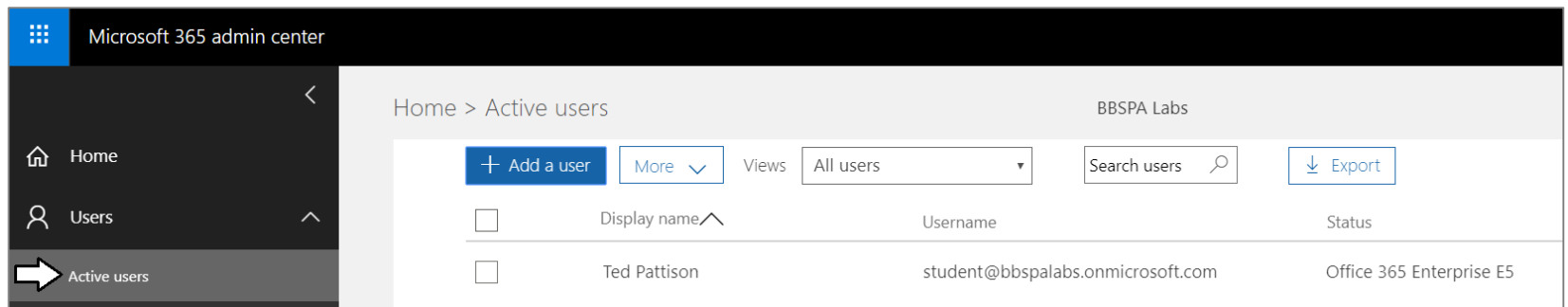


Microsoft 365 Admin Center

- Navigate to the Microsoft 365 Admin center

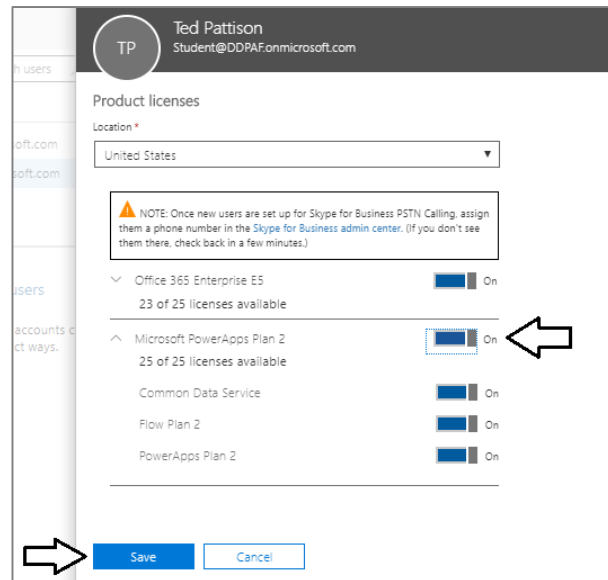
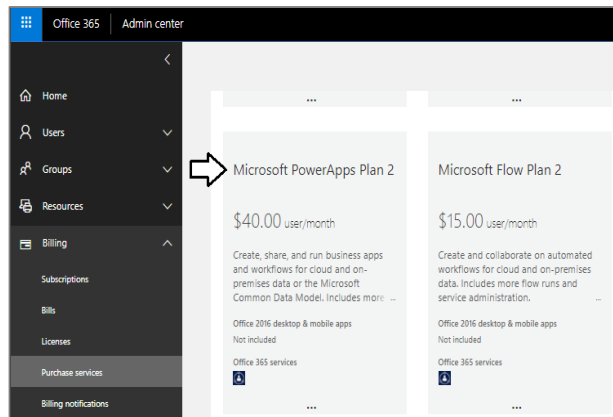


- Allows for management of users accounts and licensing



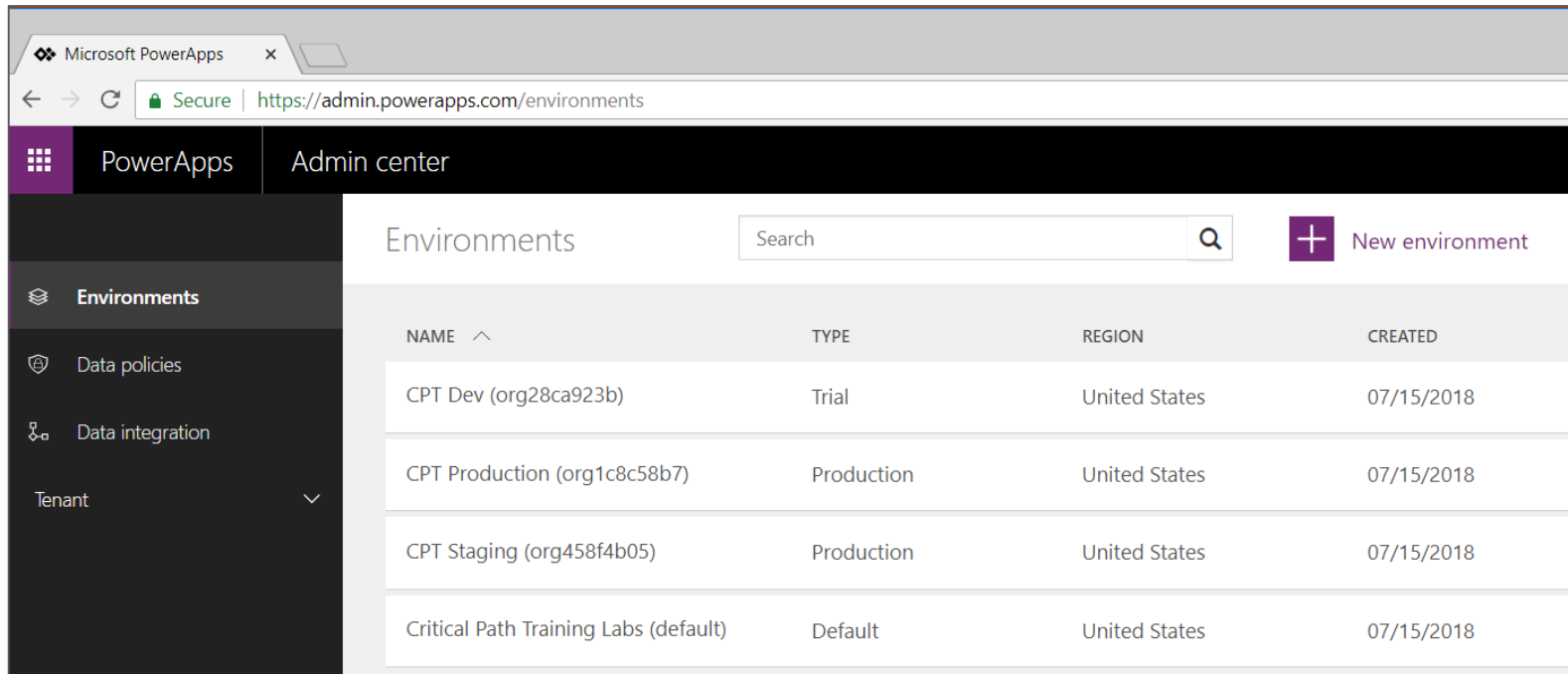
Configuring a PowerApps Plan 2 License

- 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



The screenshot displays the Microsoft PowerApps Admin Center interface. The browser address bar shows the URL <https://admin.powerapps.com/environments>. The left sidebar contains navigation options: Environments, Data policies, Data integration, and Tenant. The main content area is titled 'Environments' and includes a search bar and a '+ New environment' button. A table lists the existing environments for the tenant.

NAME ^	TYPE	REGION	CREATED
CPT Dev (org28ca923b)	Trial	United States	07/15/2018
CPT Production (org1c8c58b7)	Production	United States	07/15/2018
CPT Staging (org458f4b05)	Production	United States	07/15/2018
Critical Path Training Labs (default)	Default	United States	07/15/2018





DEMO

Configuring PowerApps Plan 2 Licenses

Agenda

- ✓ Getting Started with the Power Platform
- Building Flows using Triggers and Actions
 - Using Control-of-Flow Actions
 - Writing Advanced Flow Expressions
 - Executing a Flow from a Canvas App



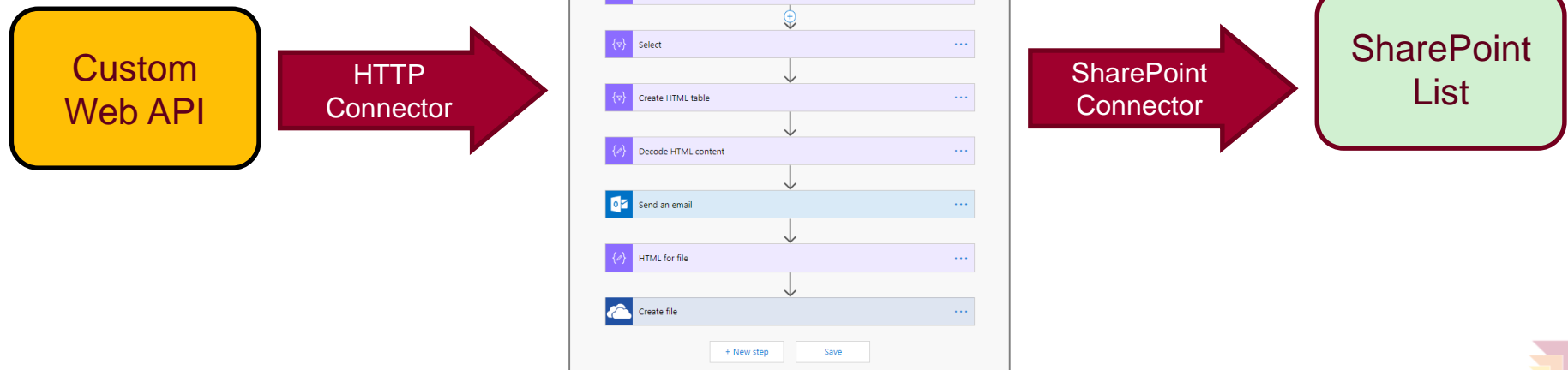
What is Flow?

- Service for automating workflows across other services
 - Designed by Microsoft for business users more than developers
- What can you do with Flow?
 - Get notifications
 - Copy files
 - Collect data
 - Automate approvals



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



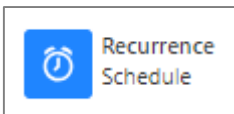
Examples Services to Use with Flow

- SharePoint Online
 - Trigger a flow when a new item is added
 - Trigger a flow when a new document is uploaded
 - Add an action to create or update a list item
- Twitter
 - Trigger a flow when a tweet contains a specific hashtag
 - Track tweet by sending email or write to SharePoint list
 - Add an action to reply to a tweet

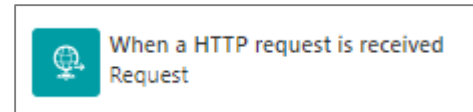
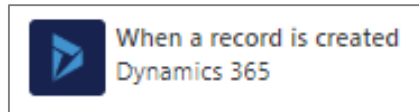
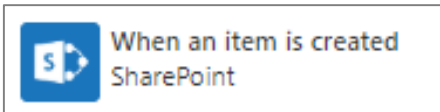


Flow Trigger Types

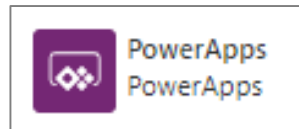
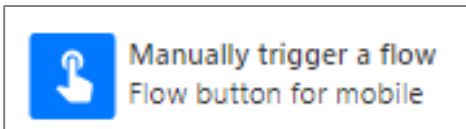
- Scheduled Flow Triggers
 - Runs periodically based on an interval



- Automated Flow Triggers
 - Runs when something happens

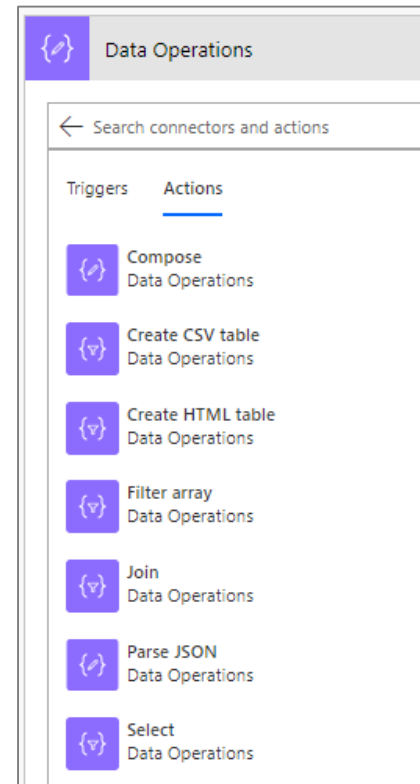
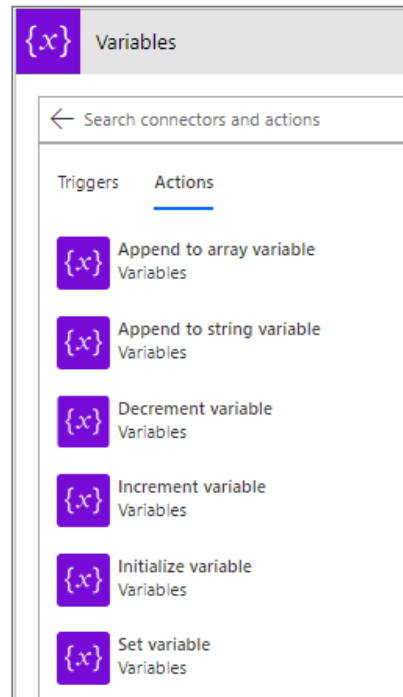
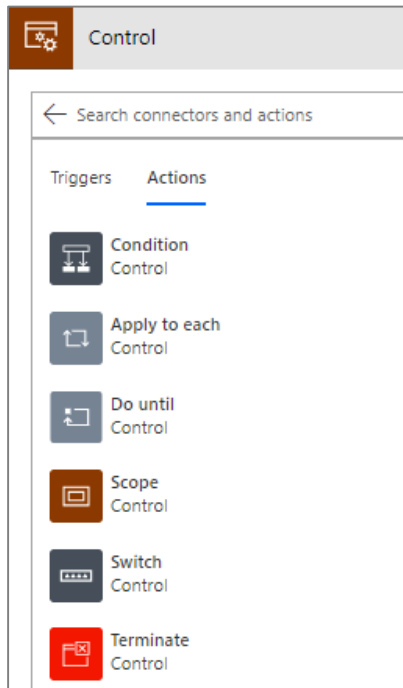


- On-demand Flow Triggers
 - Runs when a user clicks a button



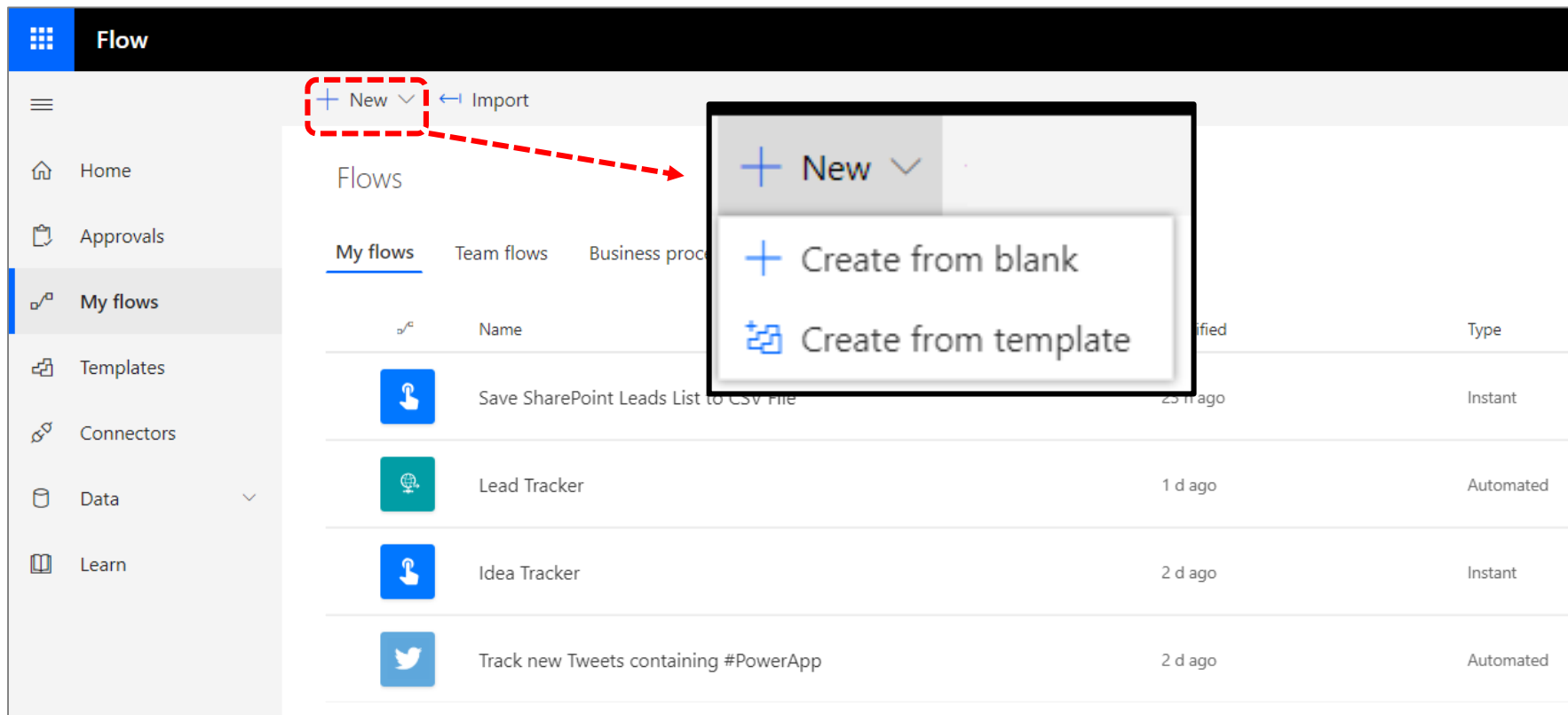
Core Action Categories

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



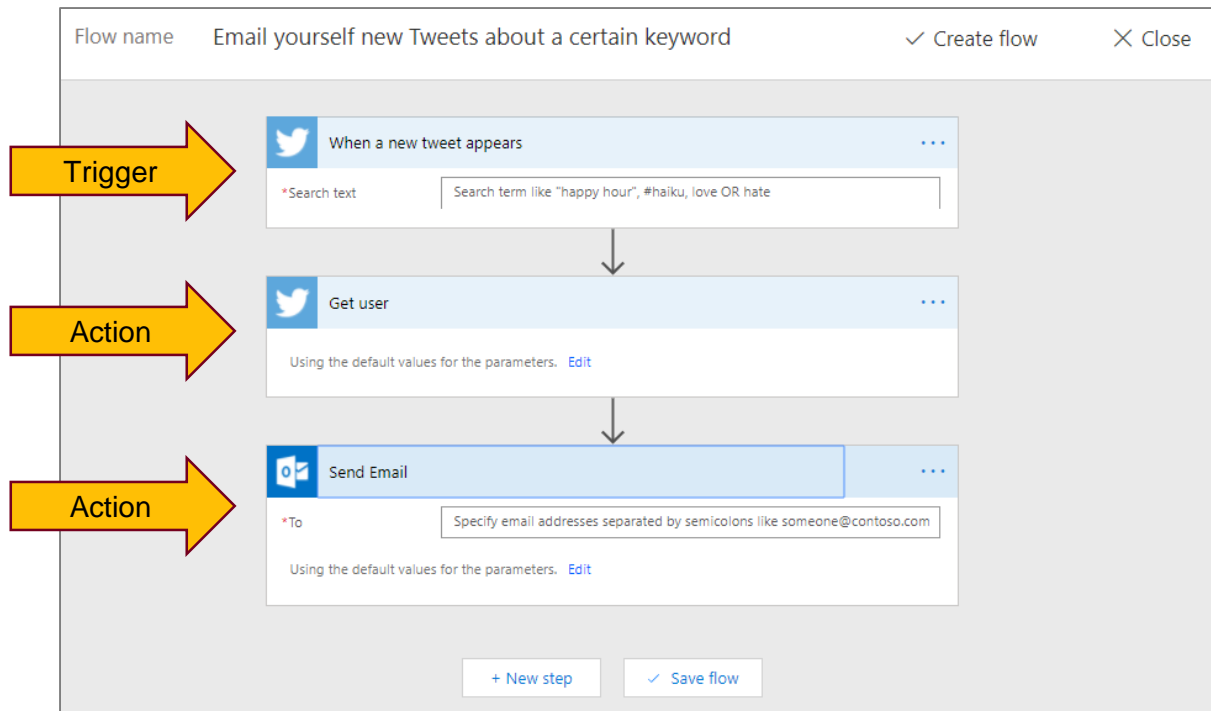
Creating and Managing Flows

- Flow user portal allows users to manage and edit flows
 - Accessible through <https://flow.microsoft.com>
 - Flow can be created from blank or from template



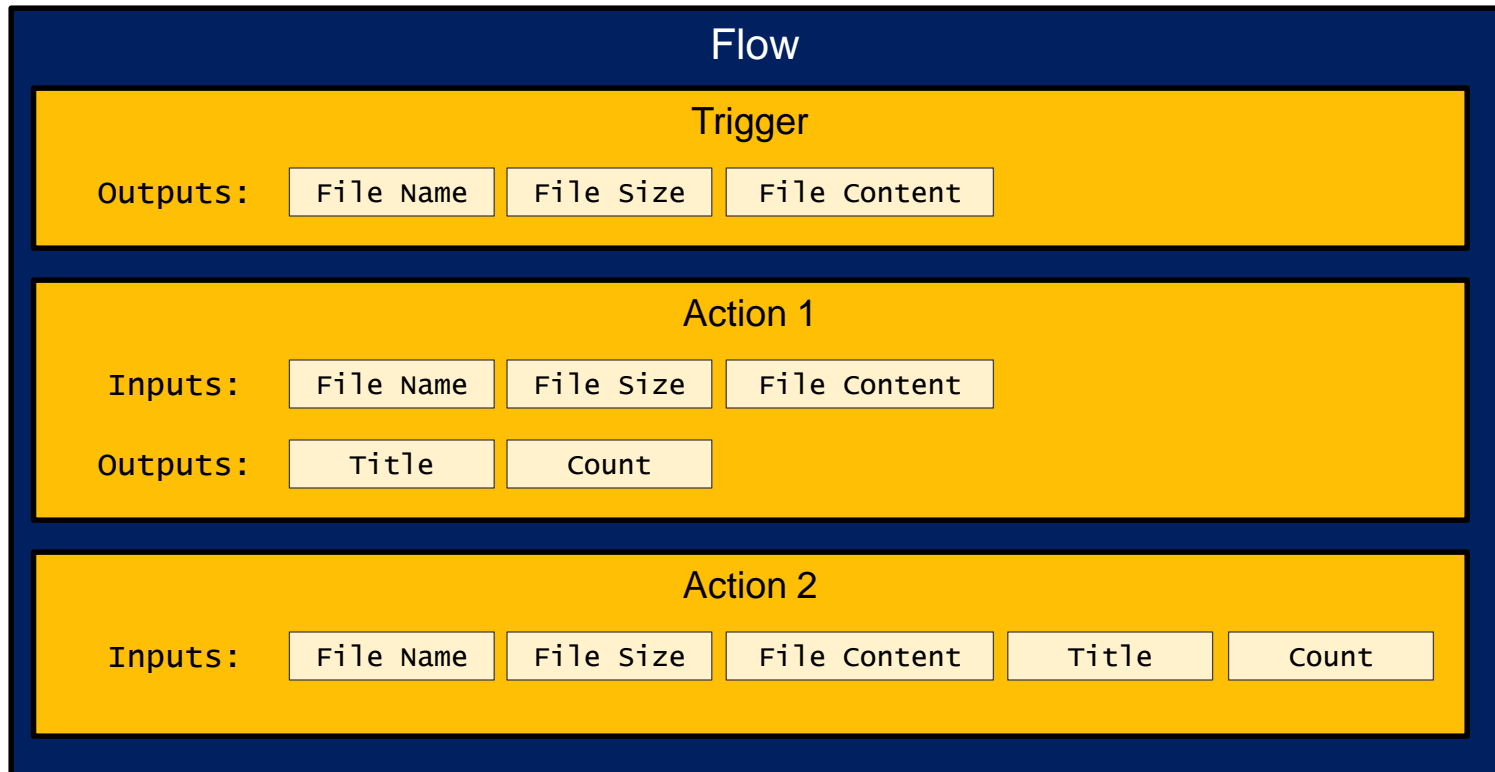
Working with the Flow Designer

- Flow Designer provides UI for building flows
 - Somewhat similar to SharePoint Designer workflow designer
 - You build flows by adding and configuring steps
 - There are 3 types of steps: triggers, actions or conditions



Data Automatically Flows from Step to Step

- Data in flows added by step outputs
 - Data added in step output is available in later steps
 - It's easy to configure step input data using output data in previous steps
 - Certain outputs displayed/hidden based on types of input and output



Flow Checker

- Automatically checks flows for errors and omissions

The screenshot displays the Microsoft Flow Builder interface. The main workspace shows a flow titled "Send an email when a new item is created in SharePoint." with two steps:

- When a new item is created** (SharePoint connector):
 - * Site Address**: Example: `https://contoso.sharepoint.com/sites/sitename`. Below the input field is the error message: "Include a Site Address."
 - * List Name**: SharePoint list name. Below the input field is the error message: "Include a List Name."
- Send an email** (Outlook connector):
 - * To**: Specify email addresses separated by semicolons like `someone@contoso.com`. Below the input field is the error message: "Using the default values for the parameters. [Edit](#)"

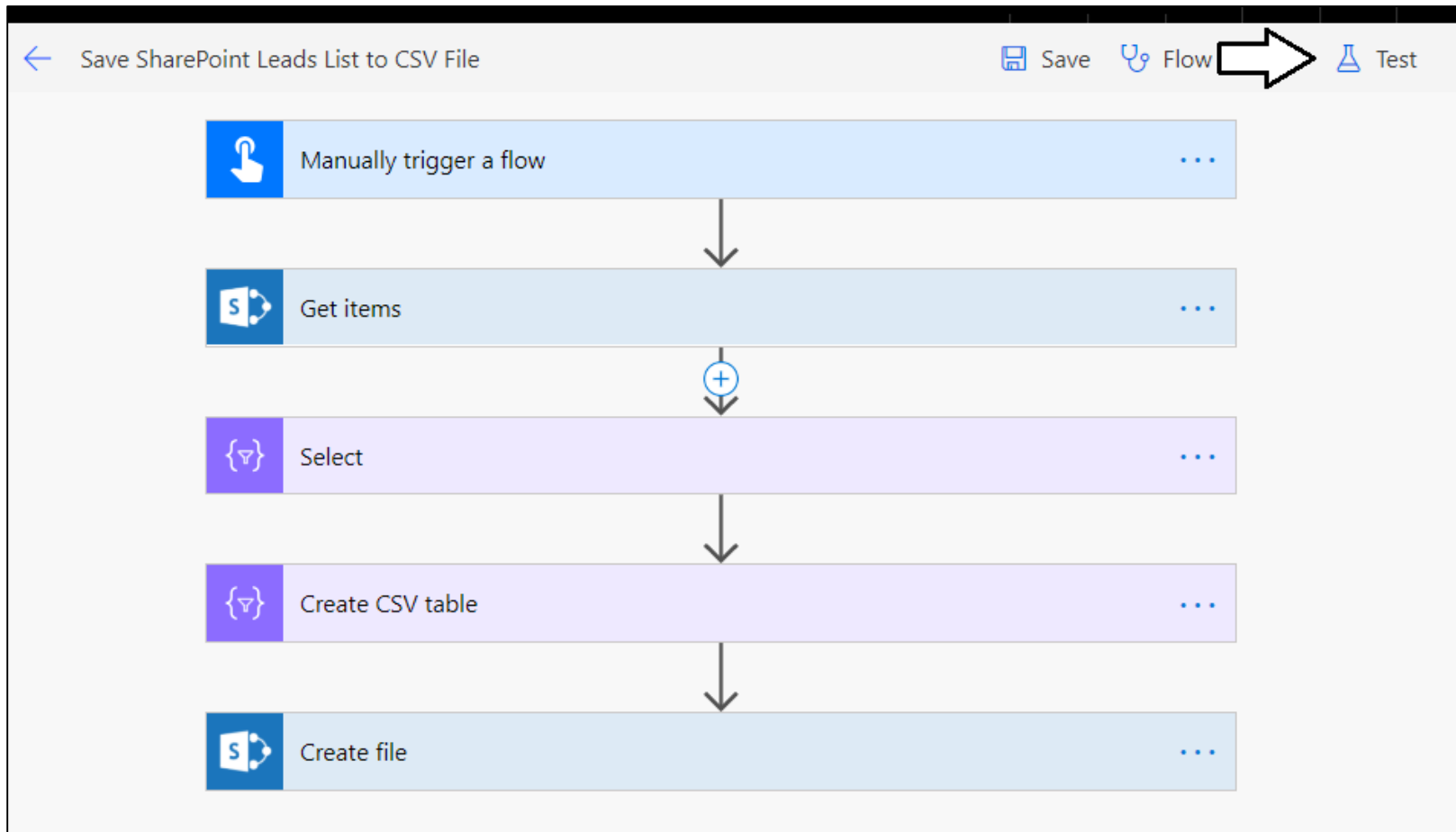
A downward arrow indicates the flow sequence from the first step to the second.

The **Flow Checker** panel on the right side of the screen shows the detected errors:

- Errors (2)**
 - When a new item is created (2)**
 - Include a Site Address.
 - Include a List Name.

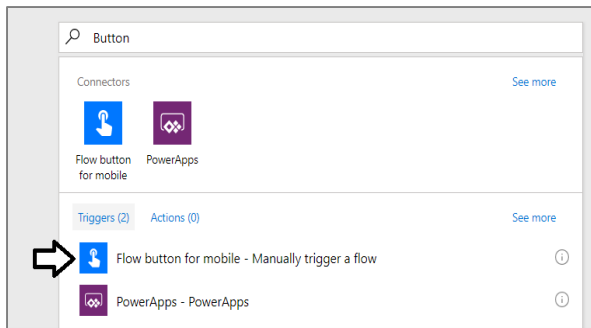
Testing a Flow

- Flow can be run/tested from edit mode



Creating a Flow Trigger by Manual Button

- Use Run button for Mobile as flow trigger



- Trigger can be extended with one or more input fields

A screenshot of the 'Run flow' dialog box for the 'Idea Tracker' flow. The dialog box has a title bar that says 'Run flow'. Below the title bar, there is a section for the flow details, which includes the flow name 'Idea Tracker' and the owner 'Owners: Ted Pattison'. Below this, there is a 'See details' link. The main part of the dialog box contains two input fields: 'Quality *' with a dropdown menu showing 'Good', and 'Idea *' with a text box containing 'Learn PowerApps for career advancement'. At the bottom of the dialog box, there is a blue 'Run flow' button and a grey 'Cancel' button. An arrow points to the 'Run flow' button.




Building Out The Idea Tracker Flow


Flow name

Idea Tracker

✓ Update flow

✕ Close

 Manually trigger a flow

 Quality

Please enter your input

...

List Options


Lame


Good


Great


Brilliant


Enter another option










 Idea

Please enter your input

...

+


 Add an input

 Send an email

* To


Student@DDPAF.onmicrosoft.com;

* Subject

 Quality ✕

Idea for consideration

* Body

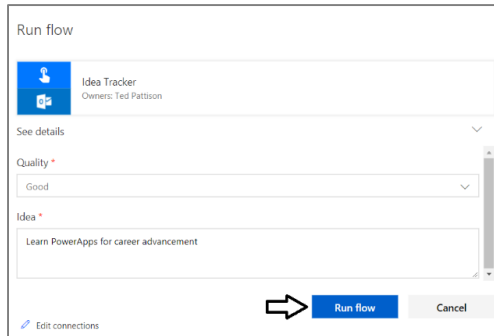
 Idea ✕

Show advanced options ▾

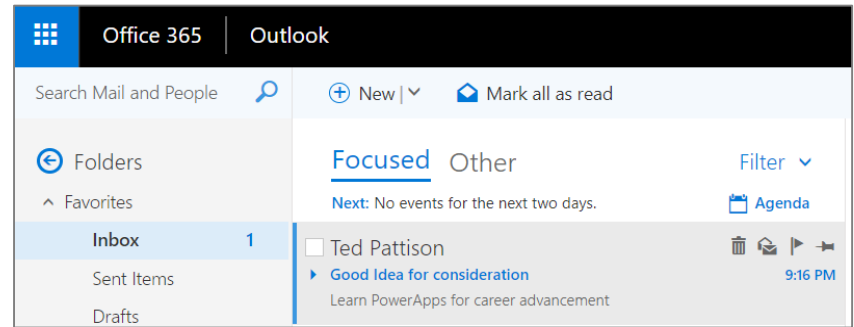
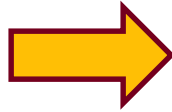


Where Should You Write the Idea?

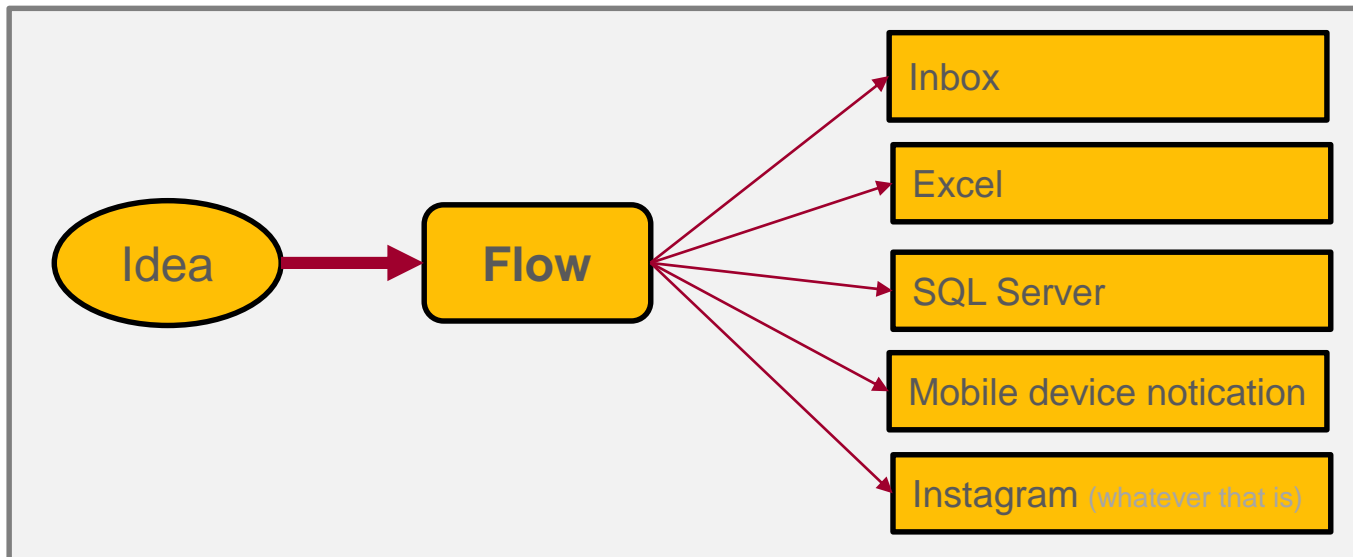
- You can track the idea by sending an email



The 'Run flow' dialog box for the 'Idea Tracker' flow (owned by Ted Pattison) is shown. It includes a 'See details' section with a 'Quality' dropdown set to 'Good' and an 'Idea' text box containing 'Learn PowerApps for career advancement'. At the bottom, there is a 'Run flow' button and a 'Cancel' button.



- Or get even more extravagant

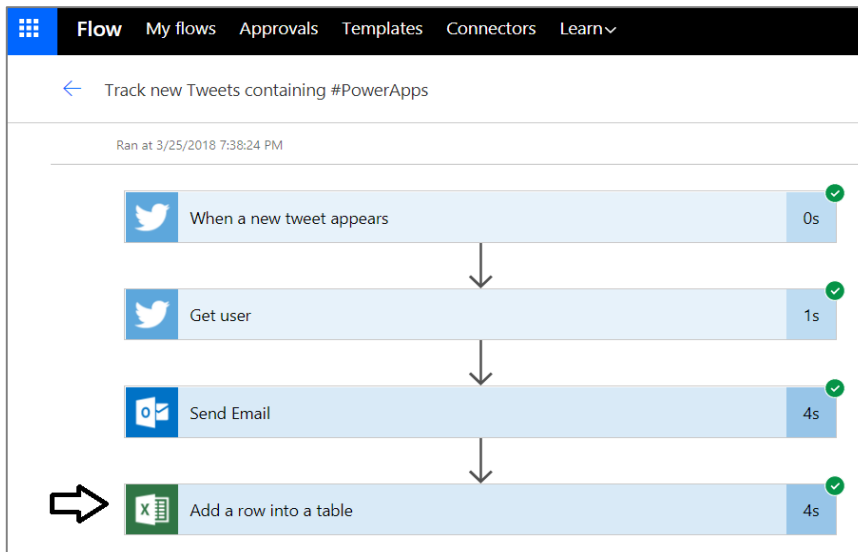


Run History

- Flow provides history flows that have run

RUN HISTORY			
➡	✓ Succeeded	9 minutes ago	8 seconds
	✓ Succeeded	2 hours ago	0 seconds
	✓ Succeeded	3 hours ago	0 seconds

- Provides read-only view of data for auditing & monitoring





DEMO

Demo 1

Creating and Testing a Simple Flow

Creating a Flow for a SharePoint List

CP

Critical Path Labs Team Site

+ New

Quick edit

Export to Excel

Flow

PowerApps

...

Expenses

Expense

Category

Date

Amount



Verizon - Telephone and Internet	Operations	1/1/2018	\$923.00
Electricity Bill	Operations	2/5/2018	\$338.00





Create a flow
See your flows

Create a flow



Start with a template and create automated tasks between your SharePoint data and other apps. Choosing a template will open the Microsoft Flow site where you'll finish creating your flow.





Send a customized email when a new SharePoint list item is added





Start approval when a new item is added



When a message is posted on a group, create a SharePoint list item



Request manager approval for a selected item



When an object is created in Dynamics 365, create a list item

Show more






DEMO

Demo 2: Creating a Flow for a Selected SharePoint Item


Building a Flow To Listen for Tweets

Flow name Track new Tweets containing #PowerApps ✓ Update flow ✕ Close


 When a new tweet appears ...

* Search text


↓

 Get user ...

* User name

 Tweeted by ✕


↓

 Send Email ...


* To

* Subject
 Name ✕"/>

* Body
Tweet:

 Tweet text ✕

Tweeted by:

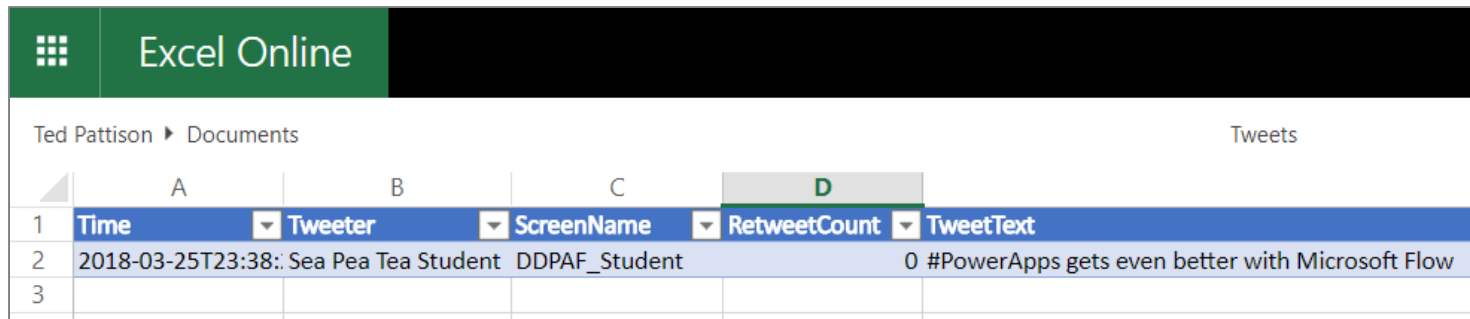
 Tweeted by ✕

Show advanced options ▾



Updating a Table in an Excel Workbook

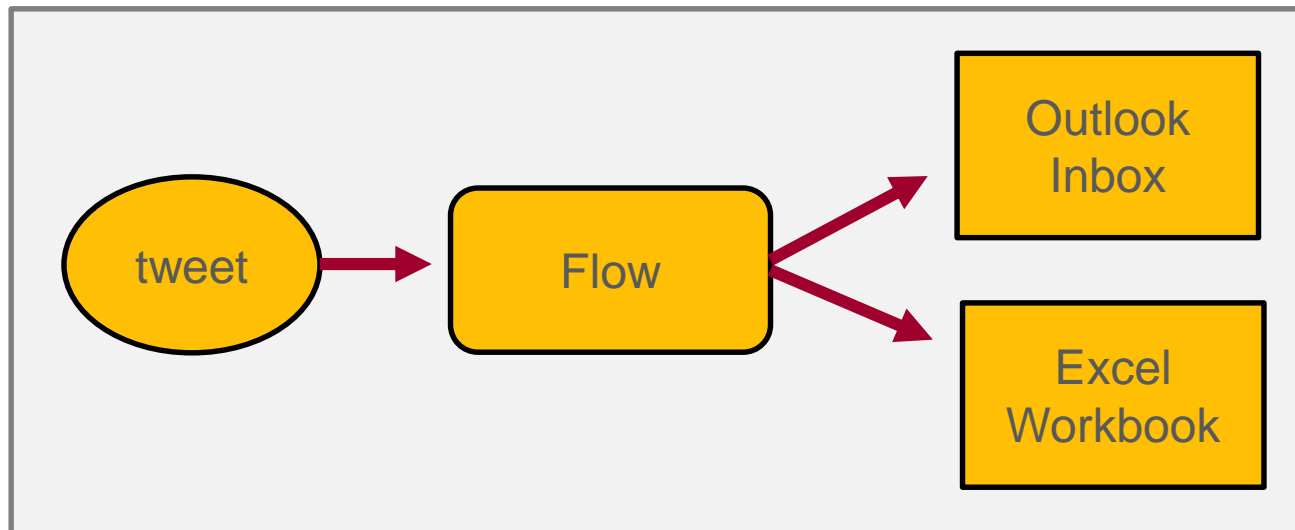
- Flow now writes tweets to Excel workbook as well as sending email





The screenshot shows the Excel Online interface. At the top, there's a green header with the Excel logo and 'Excel Online'. Below it, a breadcrumb trail shows 'Ted Pattison > Documents'. The main area displays a table with the following columns: Time, Tweeter, ScreenName, RetweetCount, and TweetText. The first row of data shows a tweet from 'Sea Pea Tea Student' with a retweet count of 0. The second row is empty.

	A	B	C	D	
1	Time	Tweeter	ScreenName	RetweetCount	TweetText
2	2018-03-25T23:38:	Sea Pea Tea Student	DDPAF_Student	0	#PowerApps gets even better with Microsoft Flow
3					

- Observation: It's easy to design a flow to write data to multiple services



Run History Details

 Add a row into a table 4s 

INPUTS

Source

OneDrive for Business

Drive

OneDrive

File

/Tweets.xlsx

Table

Tweets

item

```
{  
  "Time": "2018-03-25T23:38:28.5422853Z",  
  "Tweeter": "Sea Pea Tea Student",  
  "ScreenName": "DDPAF_Student",  
  "RetweetCount": "0",  
  "TweetText": "#PowerApps gets even better with Microsoft Flow"  
}
```

Time

2018-03-25T23:38:28.5422853Z



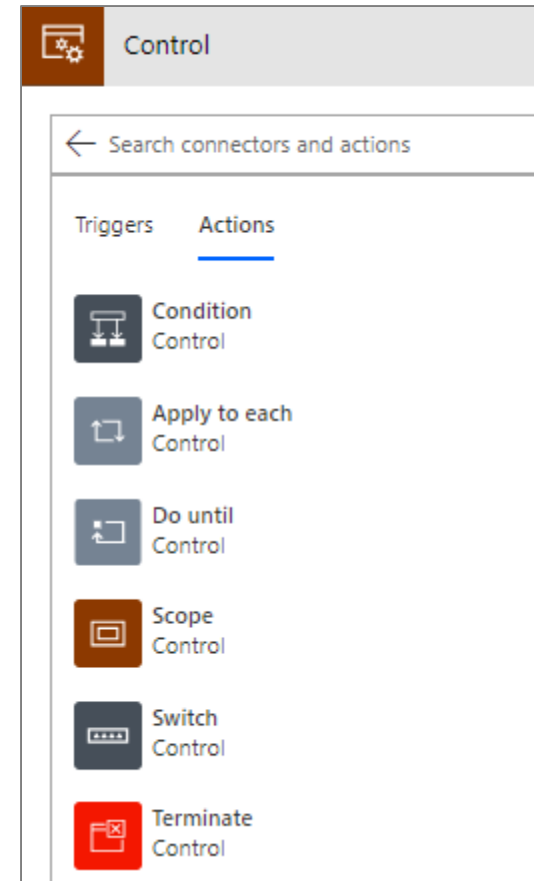
Agenda

- ✓ Getting Started with the Power Platform
- ✓ Building Flows using Triggers and Actions
- Using Control-of-Flow Actions
 - Writing Advanced Flow Expressions
 - Executing a Flow from a Canvas App
 - Handling Flow Execution Errors

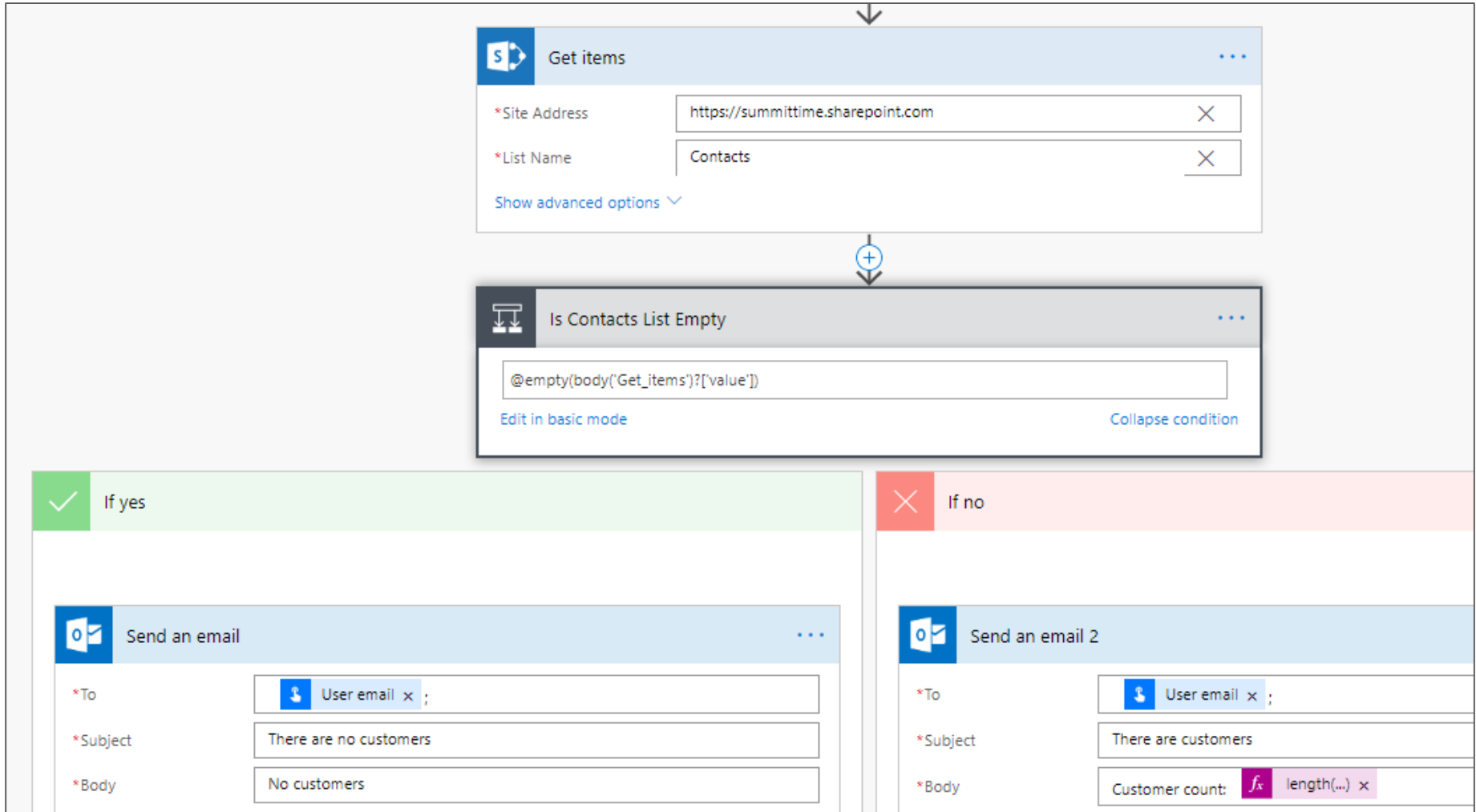


Control of Flow

- Condition
 - Provides logical structure for If Then Else
- Apply to each
 - Enumerate through collection (e.g. list items)
- Do until
 - Repeat until condition changes
- Scope
 - Create an action container with a private scope
- Switch
 - Select Case flow
- Terminate
 - Completes a flow

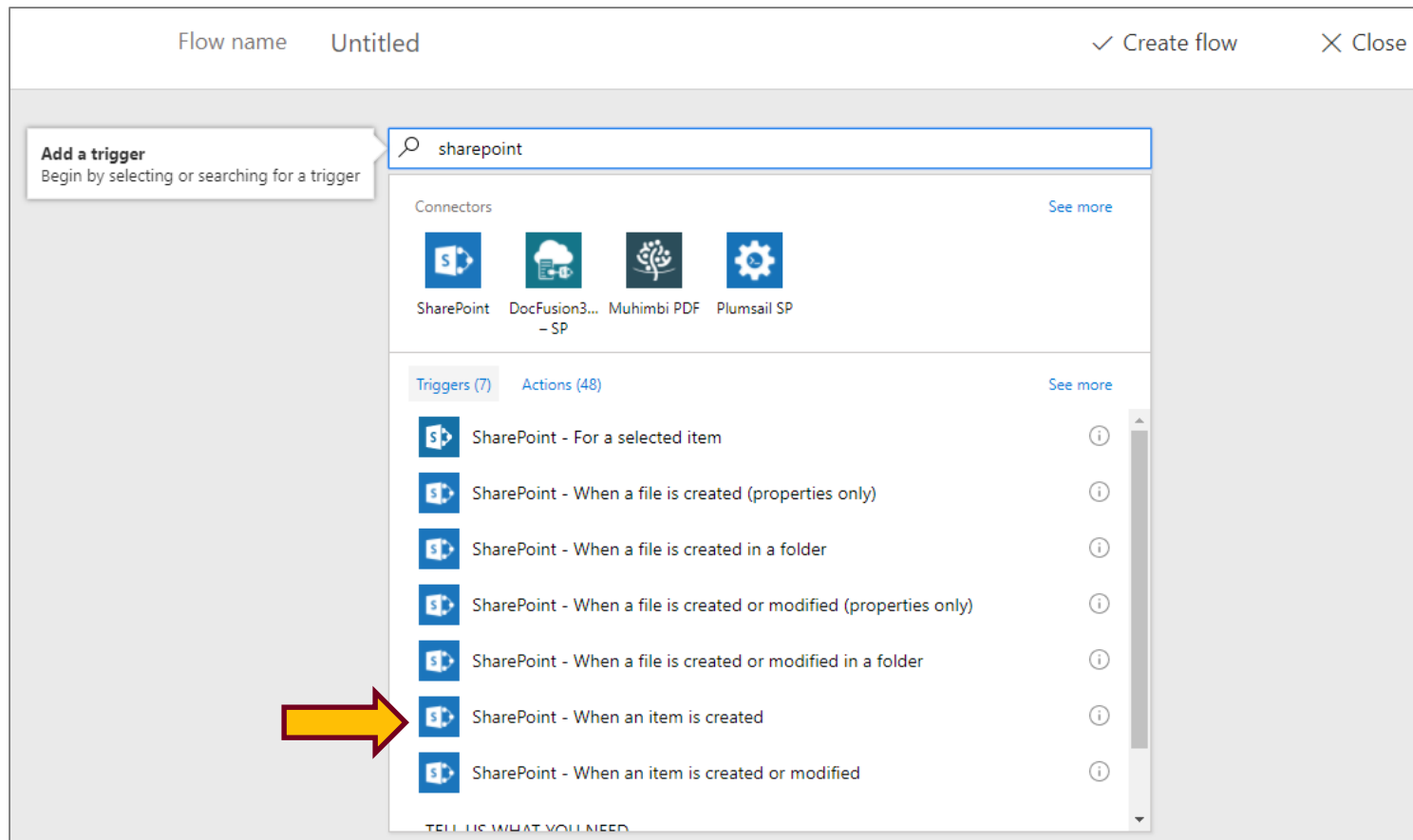


Condition Action



Another Way to Create Flows for SharePoint

- Create the flow from blank using the Flow Designer
 - Then add trigger based on SharePoint item event



Using a Condition

- Send alert email if expense amount greater than \$500
 - Condition runs test which returns true or false
 - Condition provide **If yes** branch and **If no** branch

Flow name Large Expense Alert ✓ Create flow ✕ Close

When an item is created

* Site Address

* List Name

Condition

is greater than or equal to

[Add dynamic content](#) [Edit in advanced mode](#) [Collapse condition](#)

✓ If yes ✕ If no



Build Out Branches using Actions

Flow name Large Expense Alert ✓ Create flow ✕ Close

When an item is created

Condition

Amount x is greater than or equal to 500

[Add dynamic content](#) [Edit in advanced mode](#) [Collapse condition](#)

If yes

Send an email

*To: Ted Pattison x ;

*Subject: Large Expense Alert!

*Body:

Expense: Expense x

Category: Category Value x

Amount: Amount x

Link to item x

[Add dynamic content](#)

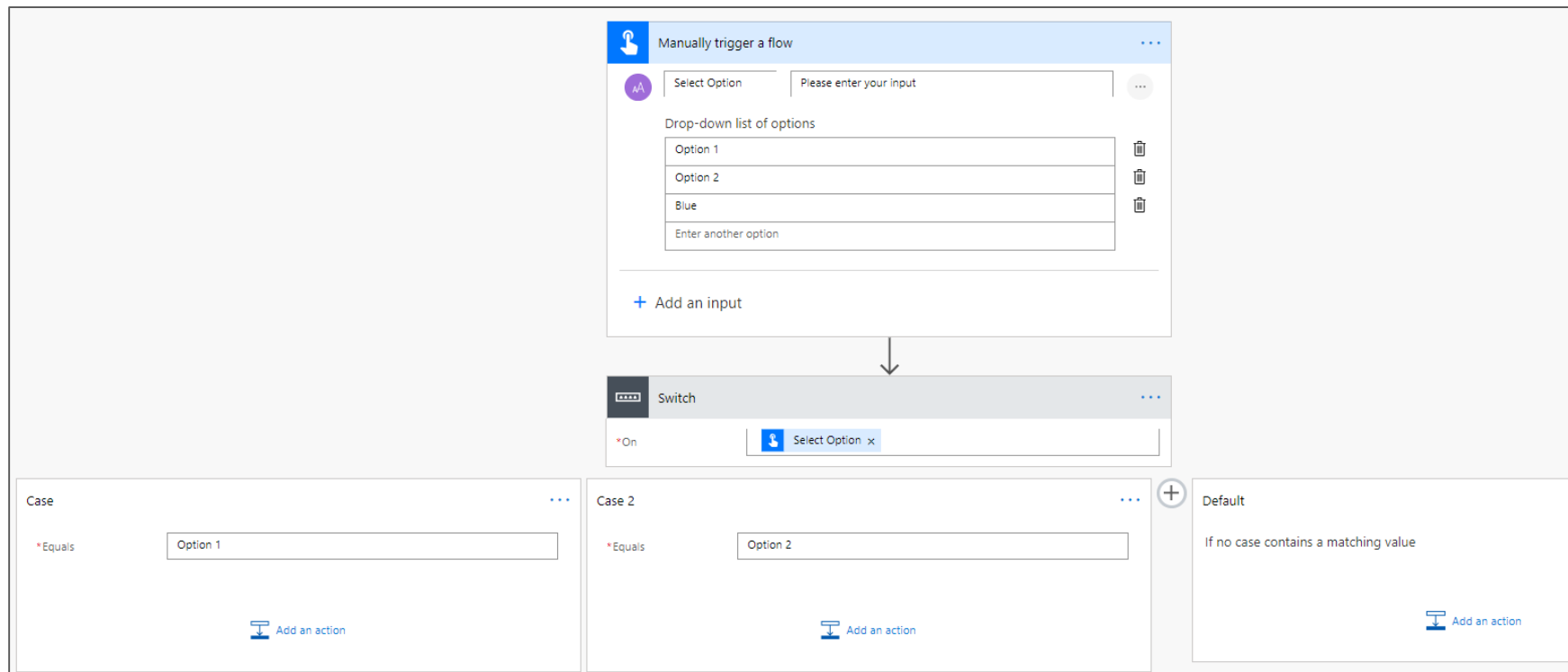
If no

[Add an action](#) [More](#)



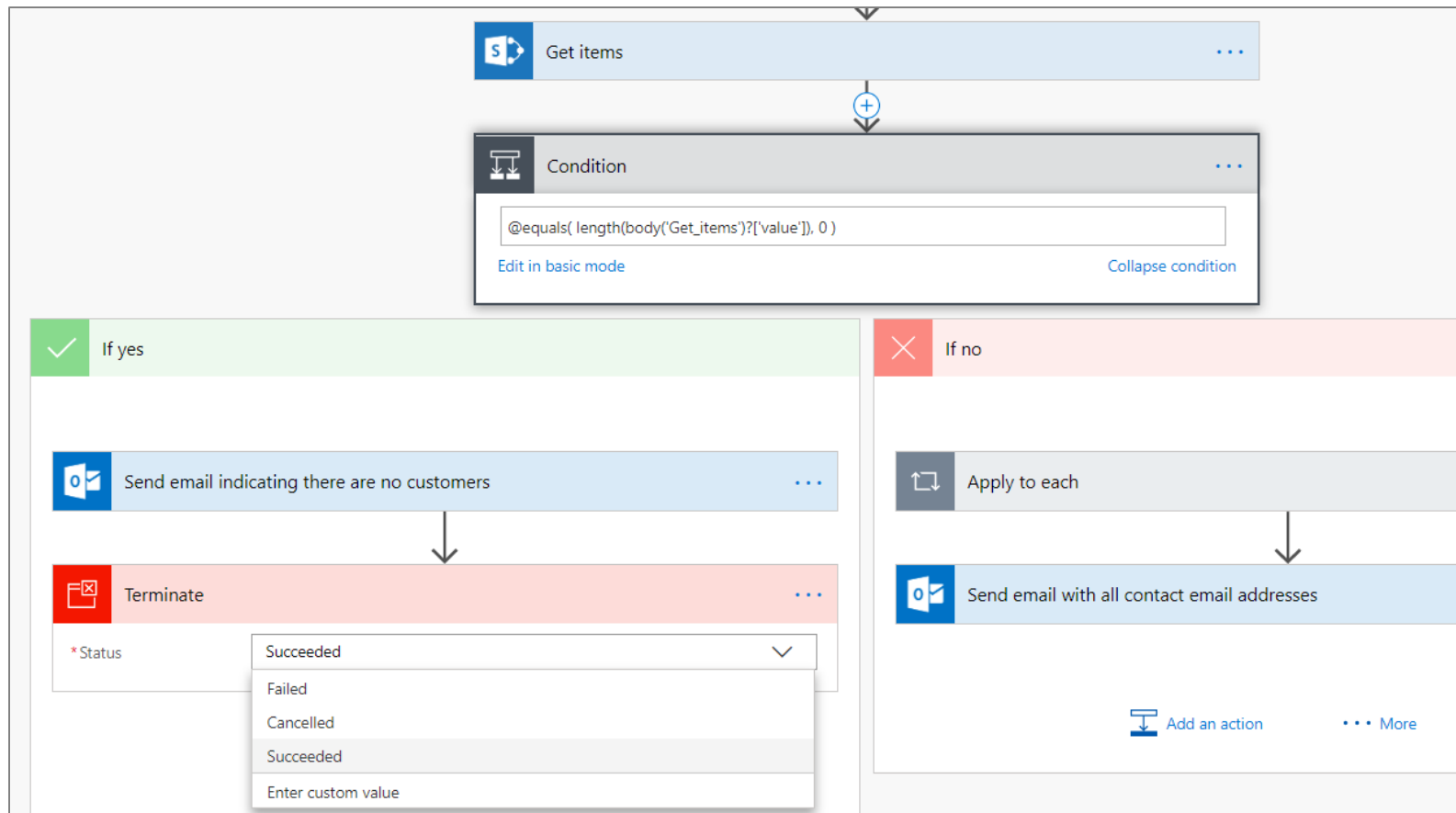
Switch Action

- Switch actions provides cases
 - Each case represents separate execution path
 - Only one execution path will execute
 - Default case executes when no other case is matched



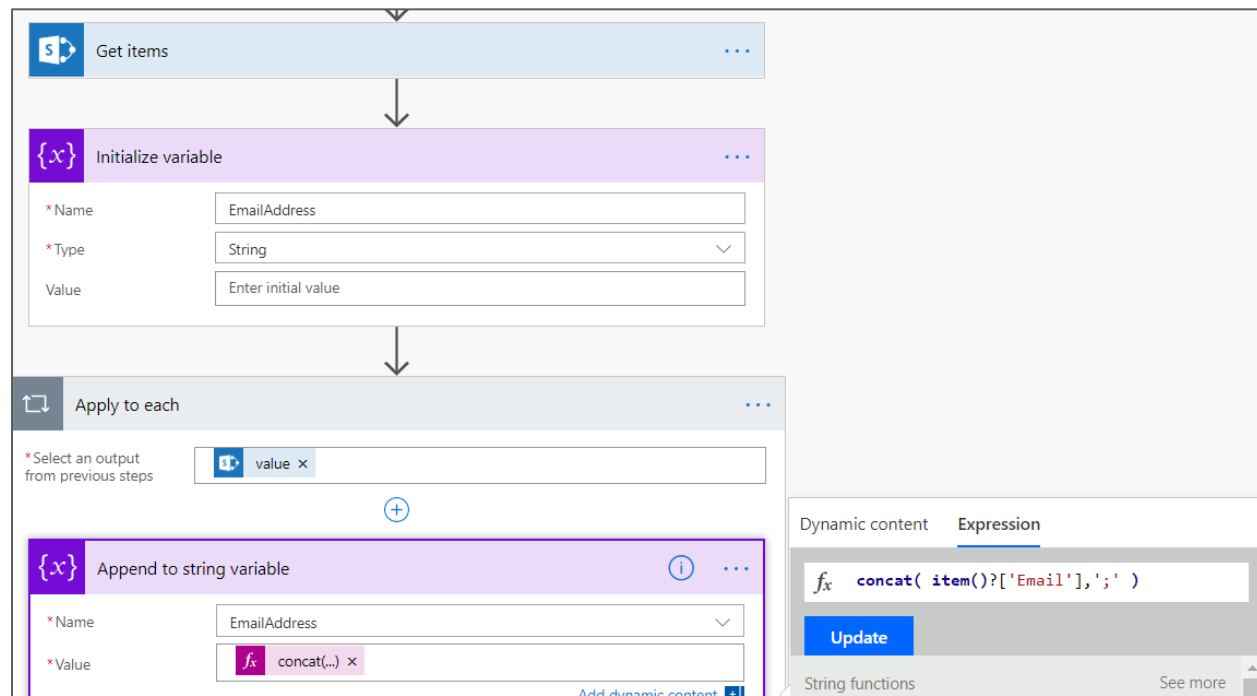
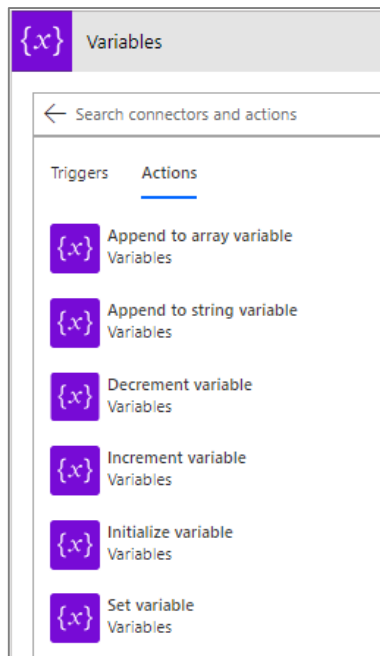
Terminate action

- Used to stop a flow at any point
 - Terminate status can be set to Succeeded, Cancelled, Failed

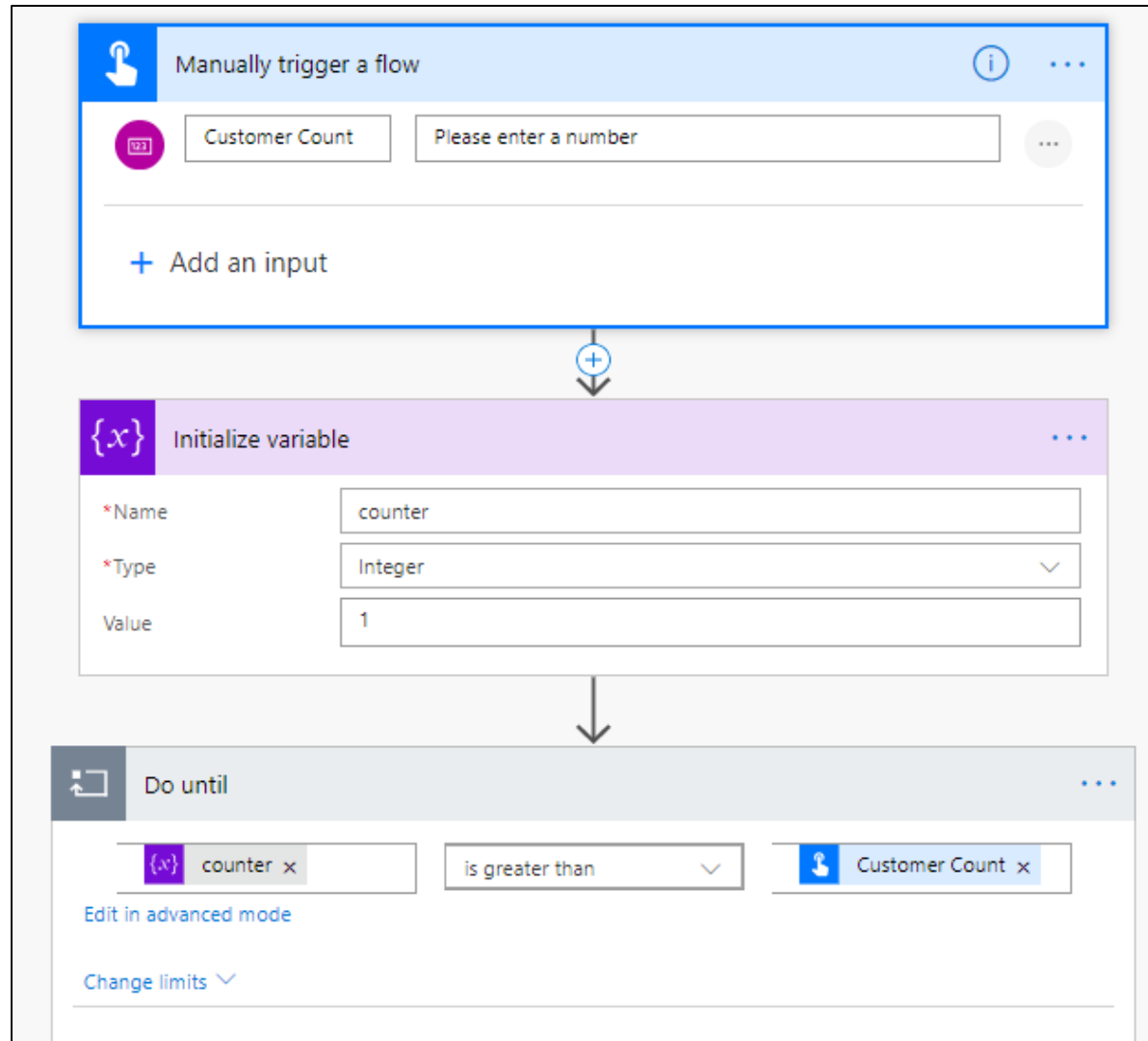


Tracking State using Variables

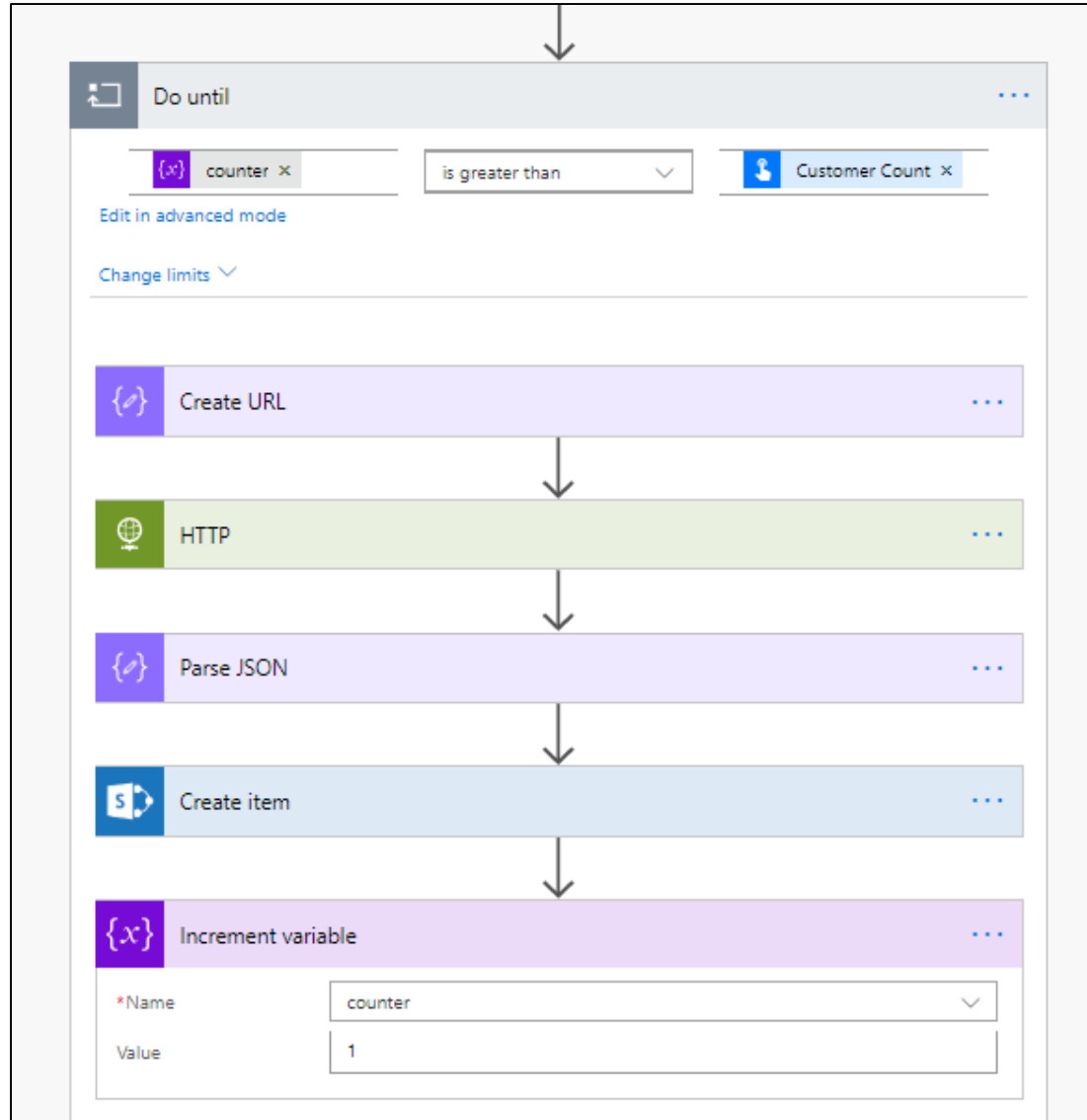
- Variables used to track state during flow lifetime
 - Initialize Variable used to create variable with Type and Value
 - Other variable actions uses to update variable values
 - By default, variable stored within flow until end of flow lifetime
 - Variables can be initialized inside Scope action to reduce lifetime



Do Until Action with Counter Variable



Executing Operations inside Do Until Loop



Using Apply to Each

- Automatically added when list is used from output
- Destination step enumerates over list items

The screenshot displays a workflow editor interface. At the top, a 'Get items' step is configured with 'Site Address' set to 'https://msd0910.sharepoint.com/' and 'List Name' set to '0769e0e8-aec5-4441-8c88-6283a6799718'. Below this, an 'Apply to each' step is shown, which is automatically added when a list is used in a previous step's output. The 'Apply to each' step has a dropdown menu showing 'value x'. Below the 'Apply to each' step, a 'Delete item' step is visible, configured with 'Site Address' set to 'Critical Path Training Labs Team Site - https://msd0910.sharepoint.com/', 'List Name' set to 'List1', and 'Id' set to 'ID x'. At the bottom of the editor, there are buttons for 'Add an action', 'Add a condition', and 'More'.



Agenda

- ✓ Getting Started with the Power Platform
- ✓ Building Flows using Triggers and Actions
- ✓ Using Control-of-Flow Actions
- Writing Advanced Flow Expressions
 - Executing a Flow from a Canvas App
 - Handling Flow Execution Errors



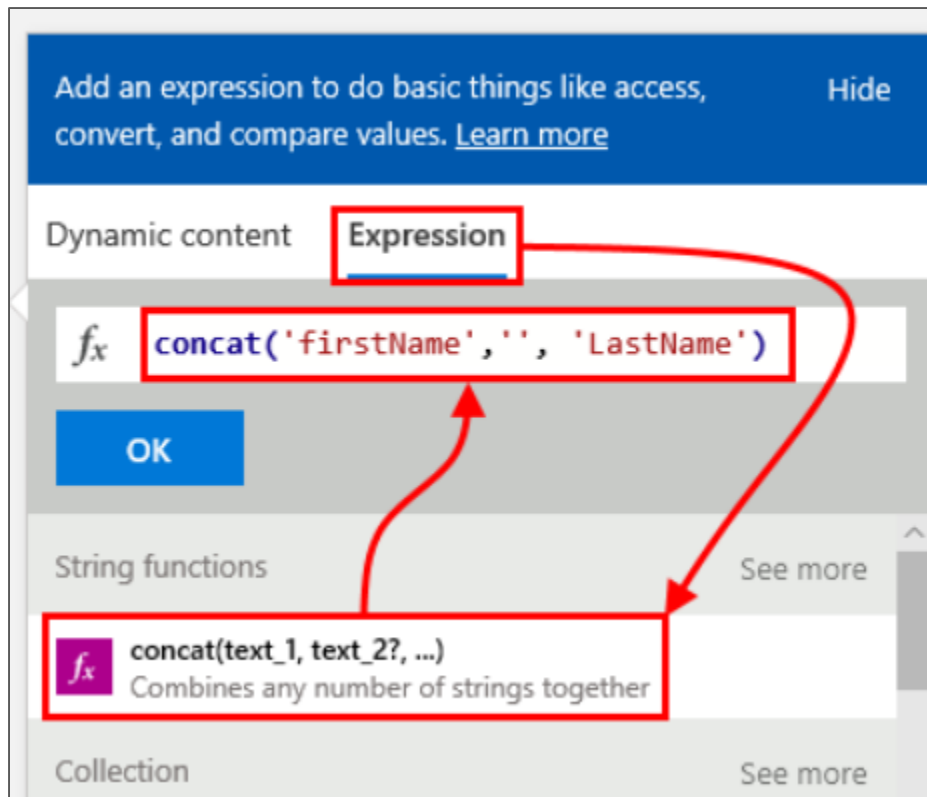
Writing Flow Expressions

- Scenarios for writing Flow expressions
 - Perform string manipulation
 - Generate a GUID or a random number
 - Convert types
 - Perform simple inline calculations
 - Handling optional values
 - Writing conditional statements using "If" statements
 - Working with arrays



Writing Expressions

- Expressions written in fx textbox
- Click OK to enter expressions



Workflow Definition Language (WDL)

- Flow expressions written in Workflow Definition Language
 - Same language used in Azure Logic Apps
 - WDL is more powerful yet more complicated than PowerApps
 - WDL does not overload operators like PowerApps does
 - WDL requires single quotes instead of double quotes

<code>fx "this is a test"</code>	Invalid: no double quotes
<code>fx 'this is a test'</code>	Valid
<code>fx 'this is a ' + 'test'</code>	Invalid : + operator not supported
<code>fx 'this is a ' & 'test'</code>	Invalid : & operator not supported
<code>fx concat('this is a ', 'test')</code>	Valid




Working with Strings

- Parse text together using **concat()**
- Parse out text using **substring()**
- Convert casing using **toLowerCase()** and **toUpperCase()**
- Search string using **indexOf** and **startsWith()**
- Create new GUID identifier using **guid()**

 **concat(text_1, text_2?, ...)**
Combines any number of strings together

 **substring(text, startIndex, length)**
Returns a subset of characters from a string

 **replace(text, oldText, newText)**
Replaces a string with a given string


 **guid()**
Generates a globally unique string (GUID)


 **toLowerCase(text)**
Converts a string to lowercase using the casing rules of the i...

 **toUpperCase(text)**
Converts a string to uppercase using the casing rules of the i...

 **indexOf(text, searchText)**
Returns the first index of a value within a string (case-insensi...

 **lastIndexOf(text, searchText)**
Returns the last index of a value within a string (case-insensit...

 **startsWith(text, searchText)**
Checks if the string starts with a value (case-insensitive, invar...

 **endsWith(text, searchText)**
Checks if the string ends with a value (case-insensitive, invari...



Performing Arithmetic Operations

- You cannot use standard arithmetic operators
 - No support for familiar operators such as **+**, **-**, *****, **/**
 - This does not work: **2 + 2**
 - This works: **add(2, 2)**

fx **min(collection or item1, item2?, ...)**
Returns the minimum value in the input array of numbers

fx **max(collection or item1, item2?, ...)**
Returns the maximum value in the input array of numbers

fx **rand(minValue, maxValue)**
Generates a random integer within the specified range (inclu...

fx **add(summand_1, summand_2)**
Returns the result from adding the two numbers

fx **sub(minuend, subtrahend)**
Returns the result from subtracting two numbers

fx **mul(multiplicand_1, multiplicand_2)**
Returns the result from multiplying the two numbers

fx **div(dividend, divisor)**
Returns the result from dividing the two numbers

fx **mod(dividend, divisor)**
Returns the remainder after dividing the two numbers (mod...



Handling Type Conversion

- Some conversion is automatic
 - Sometimes conversions are performed for you
 - In other cases, you must explicitly convert between types

**string(value)**

Convert the parameter to a string

**float(value)**

Convert the parameter argument to a floating-point number

**bool(value)**

Convert the parameter to a Boolean

**base64(value)**

Returns the base 64 representation of the input string

**base64ToBinary(value)**

Returns a binary representation of a base 64 encoded string

**base64ToString(value)**

Returns a string representation of a base 64 encoded string

**binary(value)**

Returns a binary representation of a value

**dataUriToBinary(value)**

Returns a binary representation of a data URI

**dataUriToString(value)**

Returns a string representation of a data URI

**dataUri(value)**

Returns a data URI of a value

**uriComponent(value)**

Returns a URI encoded representation of a value

**uriComponentToBinary(value)**

Returns a binary representation of a URI encoded string

**uriComponentToString(value)**

Returns a string representation of a URI encoded string



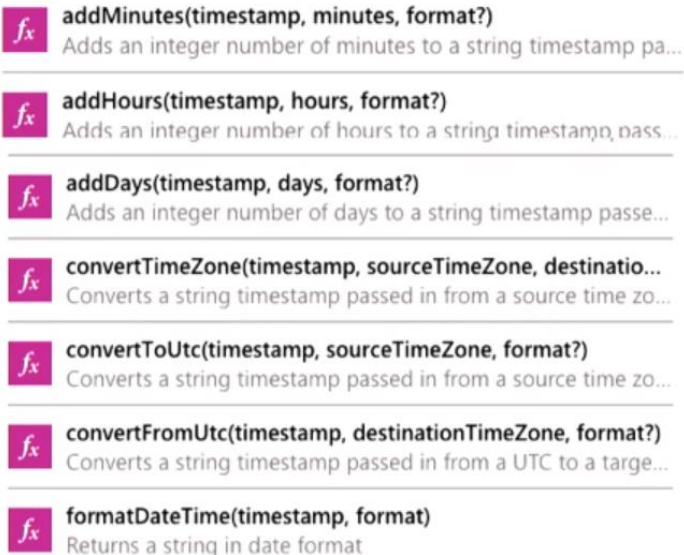
Flow Type Conversion Matrix

To From	UI adds automatically					Floating-point	Integer	Bool.	Array	JSON Object	XML content
	String	Base 64	Binary content	Data URI	URI comp.						
String	Yes	base64()	binary()	dataUri()	uriComponent()	float()	int()	bool()	split() json()	json()	xml()
Base 64	base64ToString()	Yes	base64ToBinary()	*	*	*	*	*	*	*	*
Binary content	string()	base64()	Yes	dataUri()	uriComponent()	*	*	*	*	*	*
Data URI	dataUriToString()	*	dataUriToBinary()	Yes	*	*	*	*	*	*	*
URI comp.	uriComponentToString()	*	uriComponentToBinary()	*	Yes	*	*	*	*	*	*
Floating-point	Yes	base64()	binary()	dataUri()	uriComponent()	Yes	No	No	No	No	No
Integer	Yes	base64()	binary()	dataUri()	uriComponent()	Yes	Yes	No	No	No	No
Bool.	Yes	base64()	binary()	dataUri()	uriComponent()	No	No	Yes	No	No	No
Array	join() string()	*	*	*	*	No	No	No	Select Action	Select or Compose	xml()
JSON object	string()	*	*	*	*	No	No	No	Select or Compose	Compose Action	xml()
XML content	string()	*	*	*	*	No	No	No	xpath()	xpath()	Logic apps only



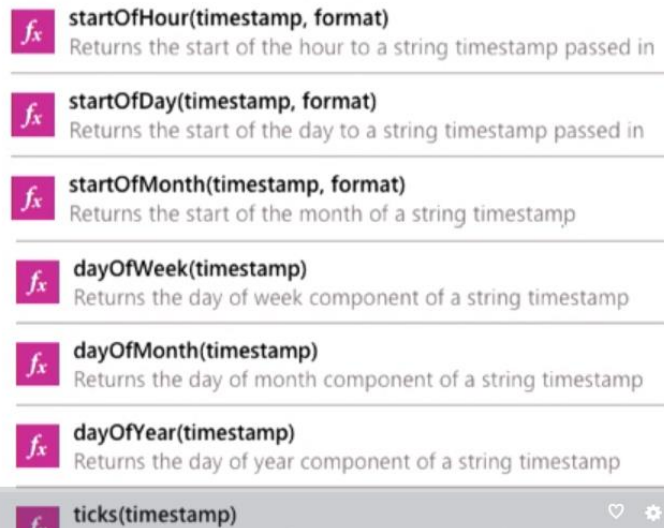
Working with Dates and Time

- Get Greenwich Meantime using **utcnow()**
- Use **add*()** functions to move time back/forward
- **convertTimeZone()** used to handle local times
- **formatDateTime()** used to format



A screenshot of a code editor window displaying a list of date and time functions. Each function is preceded by a small icon with the letter 'f' and a subscript 'x'. The functions listed are:

- addMinutes(timestamp, minutes, format?)**
Adds an integer number of minutes to a string timestamp passed in
- addHours(timestamp, hours, format?)**
Adds an integer number of hours to a string timestamp passed in
- addDays(timestamp, days, format?)**
Adds an integer number of days to a string timestamp passed in
- convertTimeZone(timestamp, sourceTimeZone, destinationTimeZone, format?)**
Converts a string timestamp passed in from a source time zone to a destination time zone
- convertToUtc(timestamp, sourceTimeZone, format?)**
Converts a string timestamp passed in from a source time zone to UTC
- convertFromUtc(timestamp, destinationTimeZone, format?)**
Converts a string timestamp passed in from a UTC to a target time zone
- formatDateTime(timestamp, format)**
Returns a string in date format



A screenshot of a code editor window displaying a list of date and time functions. Each function is preceded by a small icon with the letter 'f' and a subscript 'x'. The functions listed are:

- startOfHour(timestamp, format)**
Returns the start of the hour to a string timestamp passed in
- startOfDay(timestamp, format)**
Returns the start of the day to a string timestamp passed in
- startOfMonth(timestamp, format)**
Returns the start of the month of a string timestamp
- dayOfWeek(timestamp)**
Returns the day of week component of a string timestamp
- dayOfMonth(timestamp)**
Returns the day of month component of a string timestamp
- dayOfYear(timestamp)**
Returns the day of year component of a string timestamp
- ticks(timestamp)**



Agenda

- ✓ Building Flows using Triggers and Actions
- ✓ Using Control-of-Flow Actions
- ✓ Writing Advanced Flow Expressions
- Executing a Flow from a Canvas App
 - Handling Flow Execution Errors





DEMO

Calling a Flow from a Canvas App

Agenda

- ✓ Building Flows using Triggers and Actions
- ✓ Using Control-of-Flow Actions
- ✓ Writing Advanced Flow Expressions
- ✓ Executing a Flow from a Canvas App
- Handling Flow Execution Errors



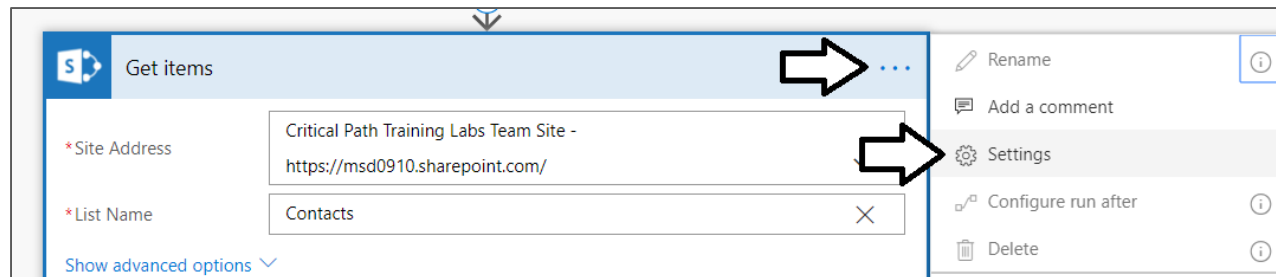
Normal action execution

- Standard behavior of a flow
 - Action steps execute in sequential order
 - Flow terminates if error occurs (failure or timeout)
- After flow runs, every action left in 1 of 4 possible states



Action Settings


- Settings let you configure
 - Async Actions
 - Timeouts
 - Retry Policy
 - Sequential Behavior
 - And more!




Error Handling

- Select the **Run after** option from action menu
 - Choose which error conditions, the arrow will turn dotted red
 - Use parallels for errors that are not at end of flow
 - Retry policy by default handles transient failures
 - Recommended to select exponential as they last a long time

'Handle duplicate file names' should run after:

 This step will only run if the flow couldn't create the file because there's already another one with the same name. It will add some numbers to the end of the file name to make it unique.

 Create file
Failed

☐ is successful
☒ has failed
☐ is skipped
☐ has timed out

Done **Cancel**

Retry Policy

A retry policy applies to intermittent failures, characterized as HTTP status codes 408, 429, and 5xx, in addition to any connectivity exceptions. The default retry policy is to retry 4 times with a 20 second delay between each attempt.

Retry policy type

Interval ⓘ

Count

Done **Cancel**



Summary

- ✓ Building Flows using Triggers and Actions
- ✓ Using Control-of-Flow Actions
- ✓ Writing Advanced Flow Expressions
- ✓ Executing a Flow from a Canvas App
- ✓ Handling Flow Execution Errors

