



Microsoft Power Platform CONFERENCE

POWER BI

POWER AUTOMATE

POWER APPS

POWER VIRTUAL AGENTS

POWER PAGES

Elevate Your Power Automate Workflows with Error Handling

Using Adaptive Cards in Microsoft Teams

Norm Young

 [stormin_30](#)



Norm Young

Who am I?

- Microsoft MVP
- Sr Strategic Consultant at AvePoint

When it comes to tech, ask me about:

- Dataverse and Model-driven Apps
- Microsoft Lists and SharePoint
- Power Automate
- Adaptive Cards

When it comes to life outside of tech, ask me about:

- Bikes and bike racing
- Star Wars from originals, prequels too sequels




What are we talking about?

- Everything eventually fails and our best created Flows are no exception
- Power Automate error messages aren't timely or informative
- **Error handling leveraging Adaptive Cards** can create engaging and informative error notifications that can be easily viewed and acted upon within **Microsoft Teams**



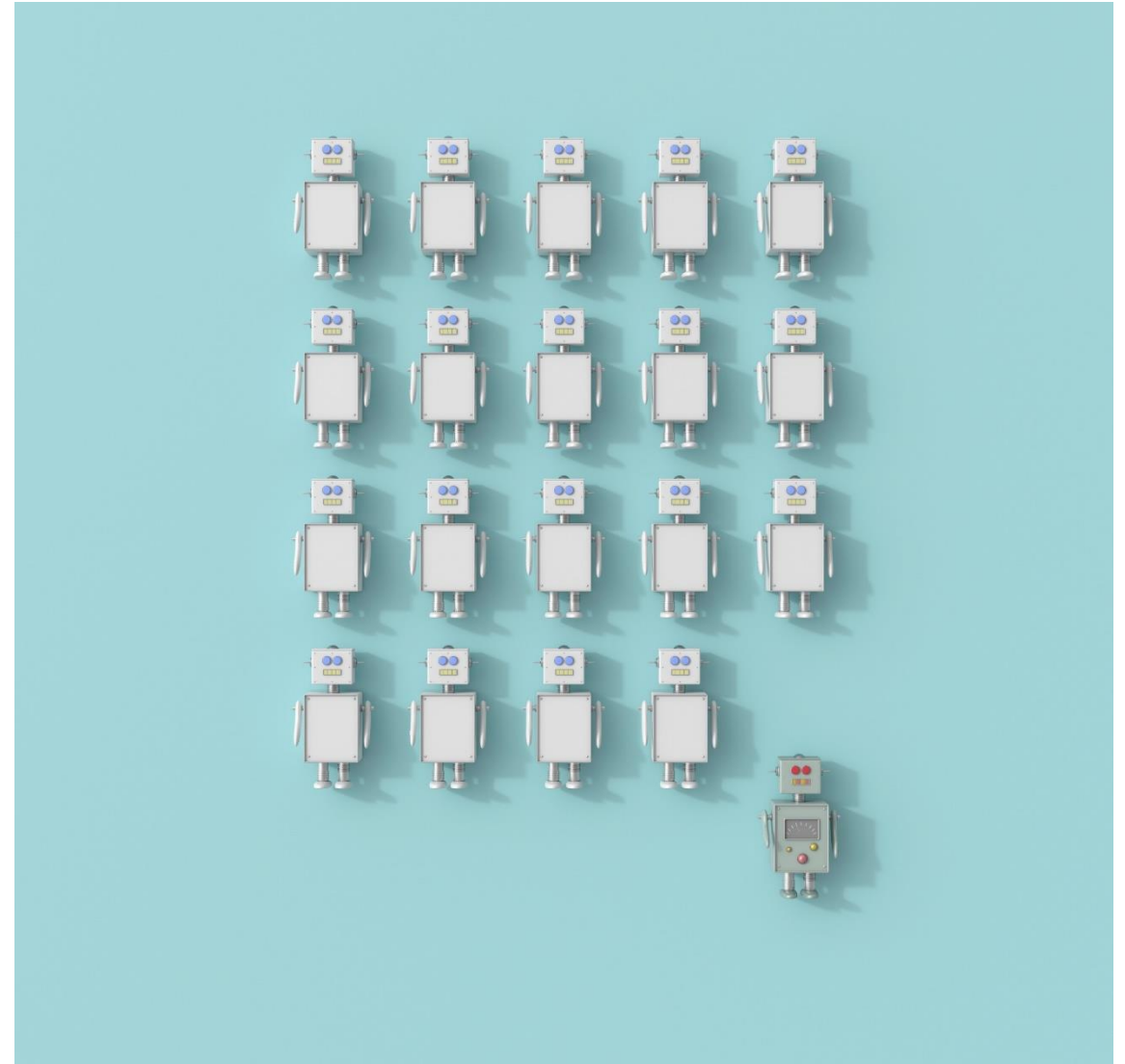
Why?



Common reasons
for error handling

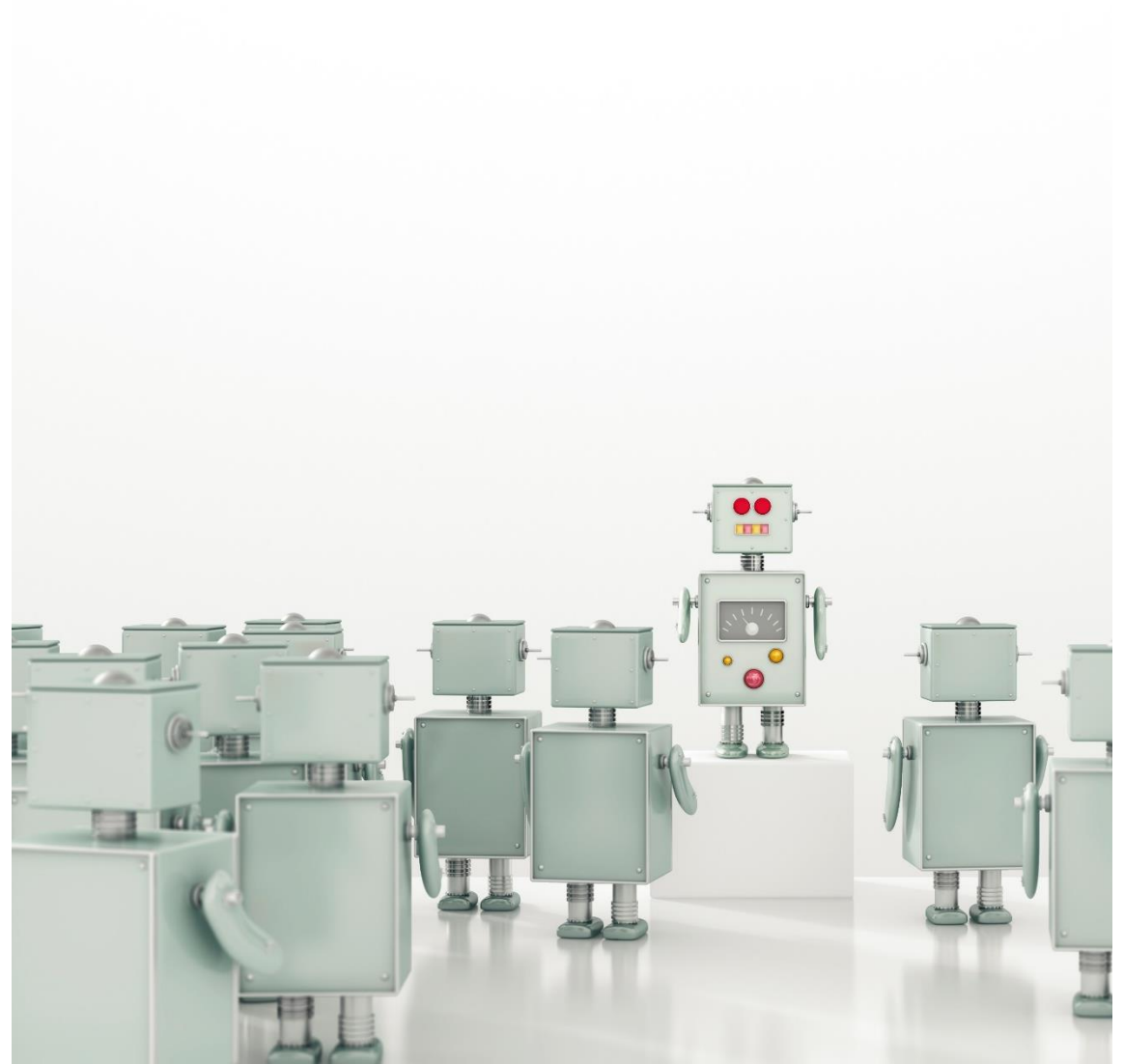
Responsiveness

Flow is part of a larger business process. Not responding to failures in a timely manner can lead to further process breakdown.



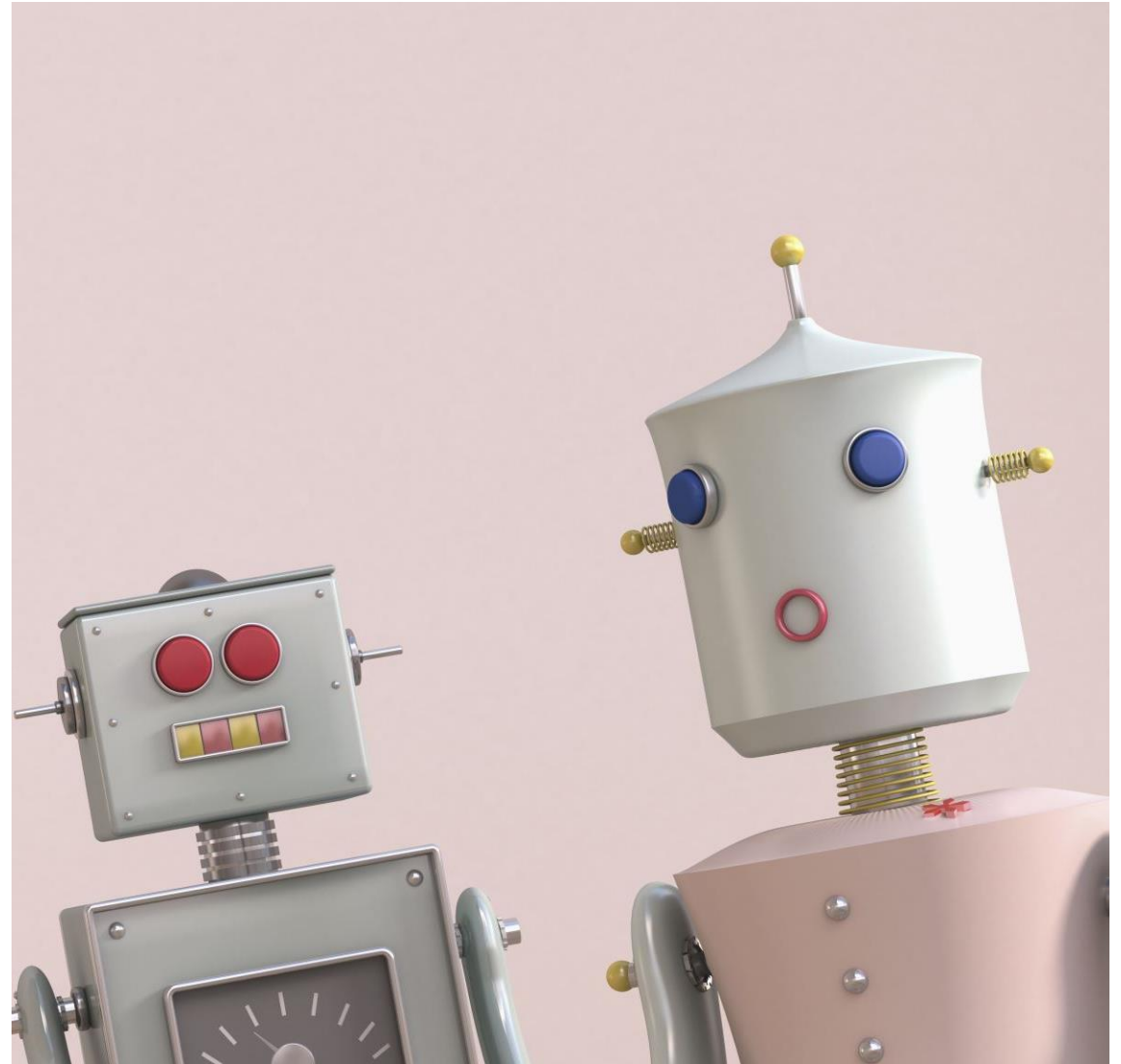
Importance

Important business processes are dependent on Flow outcomes. Continuity is paramount to business operations.



Support

Flow owner(s) are not responsible for providing day to day support and failure notifications must be delivered to others besides Flow owner(s).



Flow Overview

Our Flow error handling is known as **Try-Catch-Finally**. At a high-level our Flow pattern will:

- **Try** to run the primary actions that make up the workflow
- **Catch** any errors and take follow up actions for known issues
- **Finally**, if there are still issues it can run a final set of fail-safe actions

Flow Pattern

Scope actions (Flow)

- Encapsulate a block of actions and inherit the last terminal status (Succeeded, Failed, Cancelled) of actions inside

Debug information (Flow)

- What actions failed

Configuration information (Flow & Dataverse)

- Who are the Admin's

Notifications (User & Admin)

- Actionable Microsoft Teams Adaptive Cards

Logging (Flow & Dataverse)

- Central logging of all Flow executions

Power Apps | MDA Demo

Home Customers Accounts Contacts Sales Products Opportunities Quotes Sales

Opportunity 3 - Saved
Opportunity

General Related

Opportunity Details

Opportunity Name * Opportunity 3

Opportunity Stage Closed Won

Opportunity Amount \$5,000.00

Close Date 7/10/2023

Probability of Close 95

Lead Information

Lead Source Email campaign

Next Steps/Action Plan

Next Steps/Action Plan Send welcome kit and start onboarding

Competitor Information

Competitors ---

Outcome

Win/Loss Reason Won

Forecast

Flow

Manage

Create a flow

See your flows

Run

Move Opportunity to Sales

Alpine Ski House (sample)
Contact / Account

Norm Young
Owner

Quotes

New Quote

Quote ID	Grand Total	Customer Name
<input type="checkbox"/> Q-1002	1,597.00	Alpine Ski House (sample)

Rows: 1

Timeline

Search timeline

Enter a note...

Demo Scenario

Triggered from Model-driven App

- Creates a new Sales record from information in the Opportunity table

Business logic requirements

- Flow should only execute if the difference between the Opportunity amounts and Quotes are not +/- 15%

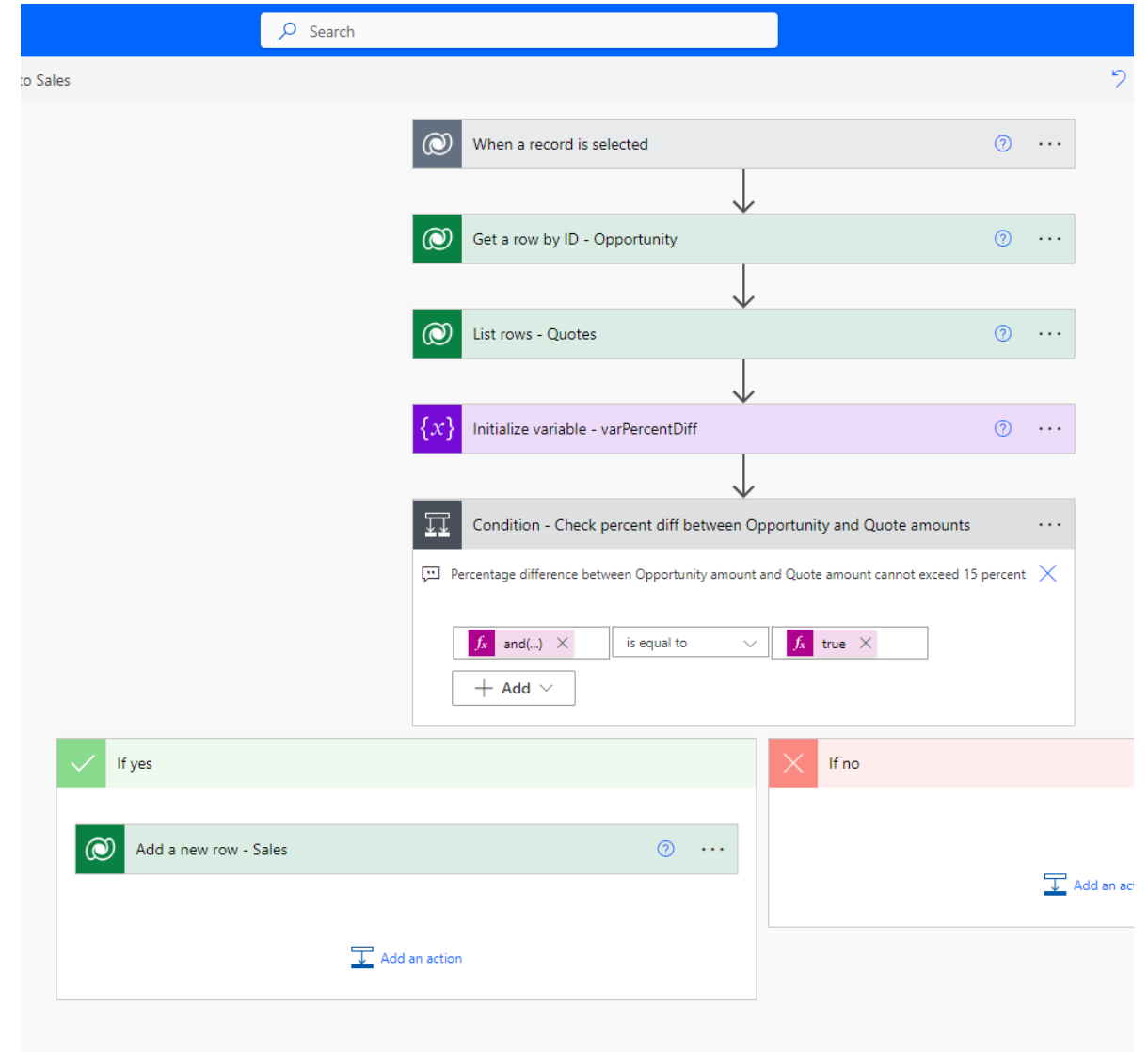
Current Flow Build

Cloud Flow

- Get **Opportunity** details for selected record
- Get associated **Quotes** for Opportunity
- Calculate **percent difference** between Opportunity and Quote amounts
- If +/- 15% create **Sales** record

Missing

- No User notification if Sales record not created
- Limited Admin notifications if Flow fails



Part 1

Initial Flow Setup

- **Configuration** table
- **Log** table
- **Debug** string variable
- **Parameters** object variable

Power Apps | MDA Demo

Home
Administration
Configuration
Logs

Save Save & Close New Deactivate Delete Refresh Check Access Assign Flow Word Templates Run Report Share

CONFIG-1000 - Saved
Configuration

CONFIG-1000
Configuration Name Norm Young Owner

General Related

Flow Name Move Opportunity to Sales

Stage Admin

User Norm Young (Available) x

Primary Email * ny@normyoung.ca

Configuration table

For each “production” Flow track

- Flow Name, Stage and User info
- One row per Flow and User

Purpose

- Data driven notifications to Admin’s

Power Apps | MDA Demo

Home
Administration
Configuration
Logs

LOG-1000 - Saved
Logs

Norm Young
Owner

LOG-1000
Name

General Related

Flow Name	Move Opportunity to Sales		
Date and time	9/13/2023	8:38 PM	
Flow Status	Failure		
Flow Link	https://make.powerautomate.com/manage/environments/3c586091-d3e1-e78d-830f-b5e0bfabddf3/flows/4d3b8347-fe47-f732-2792-0ccb53aebcdd/runs/08585069693576224191346043232CU14		

Log table

For each “production” Flow track

- Flow Name, Date-time, Status, Link to failed Flow
- One row per Flow execution

Purpose

- Provide centralized long-term logging for all “production” Flow executions

varDebug

For each production Flow

- Declare String variable
- Append **Scope** terminal status to *varDebug*

Purpose

- Use debug info for Admin notification to better understand where Flow failure occurred

The screenshot shows a Salesforce Flow Builder interface. At the top, a grey header bar contains the text 'When a record is selected' and a question mark icon. Below this, a purple header bar for the 'Initialize variable - varDebug' step is highlighted. An arrow points from the first step to this second step. The main area of the step contains a description: 'Used to store debug information for admin notifications and logging'. Below this, there are three fields: '* Name' with the value 'varDebug', '* Type' with a dropdown menu showing 'String', and 'Value' with a text area containing the text 'List of action names and statuses:'.

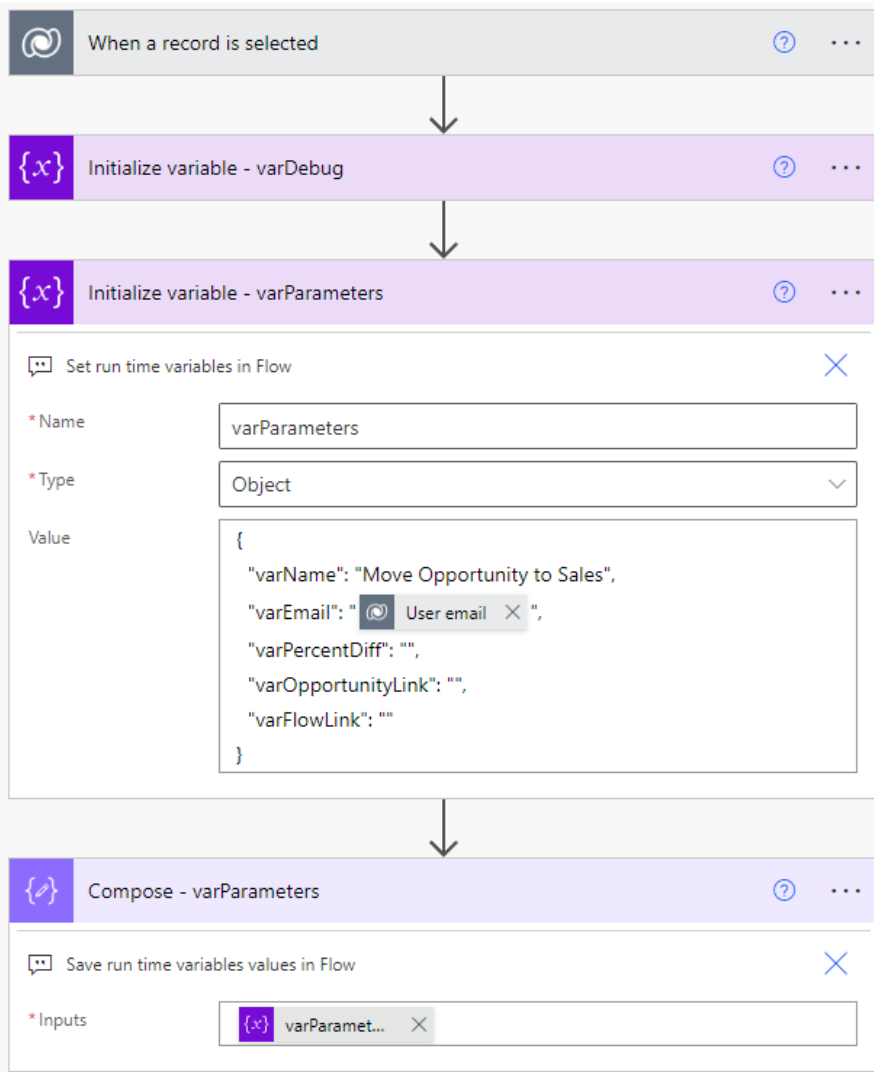
varParameters

For each production Flow

- Initialize Object variable
- Compose Object variable

Purpose

- Reducing multiple variable declarations
- Simplify Flow management and maintenance



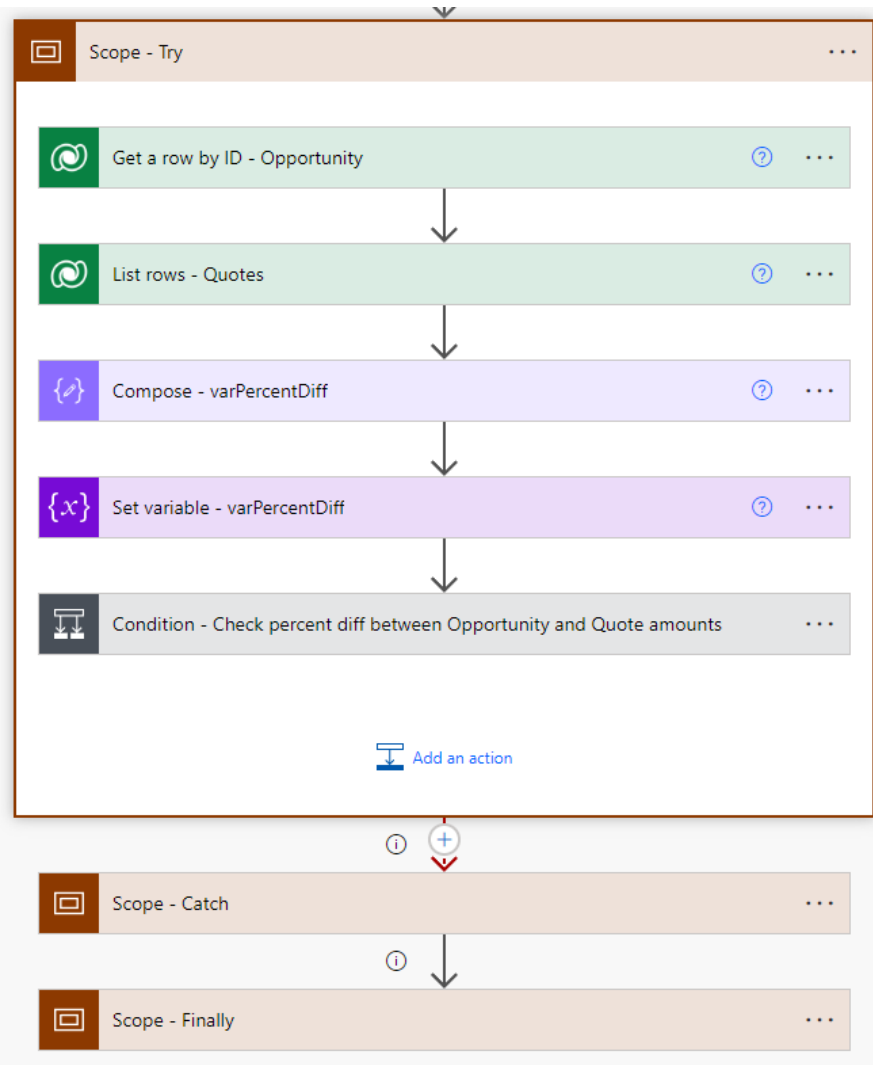
Part 2

Try, Catch, Finally

- **Scope** actions
- **Configure after run** settings

Scope actions

- 1 Scope container for each **Try**, **Catch** and **Finally** block of actions
- Variable declarations must happen outside of the Scope



Configure after run

- Checks if action should run based on the previous action status
- Statuses:
 - *Is successful*
 - *Has failed*
 - *Is skipped*
 - *Has timed out*
- For Try-Catch-Finally:
 - If **Try** has failed, then run **Catch**
 - Regardless of **Try** and **Catch** status run **Finally**

The image displays three sequential configuration panels for a workflow, connected by arrows. Each panel has a title bar, a list of actions with their status, and a set of checkboxes to configure when the next action should run.

'Scope - Try' should run after:

- Compose - varParameters (Succeeded)
 - ☒ is successful
 - ☐ has failed
 - ☐ is skipped
 - ☐ has timed out

'Scope - Catch' should run after:

- Scope - Try (Failed)
 - ☐ is successful
 - ☒ has failed
 - ☐ is skipped
 - ☐ has timed out

'Scope - Finally' should run after:

- Scope - Catch (Succeeded, Failed, Skipped, Timed out)
 - ☒ is successful
 - ☒ has failed
 - ☒ is skipped
 - ☒ has timed out



Flow Build Demo

Scope, Object variables, Configure after run

- Add **Scope** containers
 - Move existing actions into **Try**
 - Update Flow to use **Object** variables
 - Set **Catch** *Configure after run*
 - Set **Finally** *Configure after run*
-

Demo 1: Scope, Object variables, Configure after run

Power Automate

Search

Environments
Norm Young's Environ...

?

?

Home

Create

Templates

Learn

My flows

Approvals

Solutions

Process mining

AI models

Desktop flow activity

More

Power Platform

Move Opportunity to Sales

Undo

Redo

Comments

Save

Flow checker

Test

When a record is selected

Initialize variable - varDebug

Initialize variable - varParameters

Compose - varParameters

Get a row by ID - Opportunity

List rows - Quotes

Initialize variable - varPercentDiff

Condition - Check percent diff between Opportunity and Quote amounts

+ New step

Save

Ask a chatbot

Part 3

Business logic &
End-user experience

- **Adaptive Card** for end user notification
- **Dynamic data** for business context and improved user experience

Expressions

Build dynamic link to **Opportunity** record:

- Uses output from **Model-driven App** and **Opportunity** tables
- Complicated but makes the User experience actionable

Dynamic Value Expression [Format data by examples](#)

```
setProperty(variables('varParameters'), 'varOpportunityLink', concat(
  first(split(first(outputs('List_rows_-_Model-driven_Apps'))['body/value'])?['@odata.id'], '/api/')), '/main.aspx?appid=', first(outputs('List_rows_-_Model-driven_Apps'))['body/value']?['appmoduleid'], '&pagetype=entityrecord&etn=ny_opportunity&id=', outputs('Get_a_row_by_ID_-_Opportunity')?['body/ny_opportunityid']))
```

Save Cancel

Formulas Dynamic values

String functions

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

[See more](#)

Collection

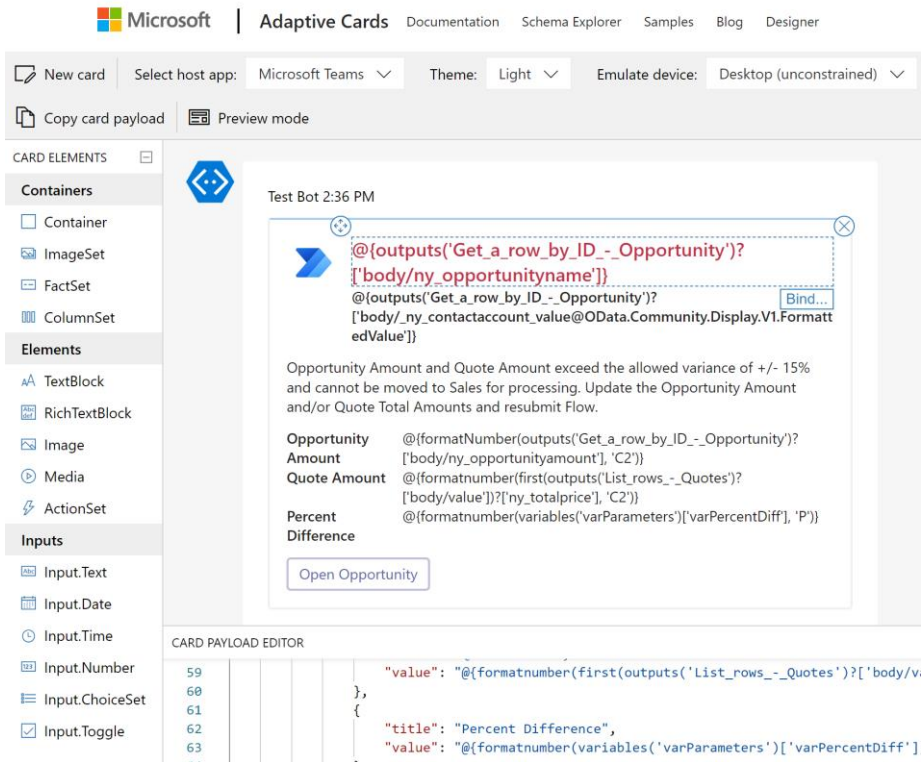
contains(collection, value)
Returns true if a dictionary contains a key, if an array contains a value, or if a string contains...

length(collection)
Returns the number of elements in an array or string

sort(collection)
Returns an array sorted in ascending order

reverse(collection)
Returns the collection in reverse order

Adaptive Cards



- Build with **Adaptive Cards Designer**
<https://adaptivecards.io/designer>
- Drag and drop canvas with configurable card elements
- Start with a sample card and customize to suite your needs
- Use **placeholders** for dynamic data to make it easier to update in Power Automate

Dynamic Data

```
Undo Redo Comments Save Flow checker Test

"items": [
  {
    "type": "TextBlock",
    "text": "🔗 Opportunity... ✕",
    "wrap": true,
    "size": "Large",
    "weight": "Bolder",
    "color": "Attention"
  },
  {
    "type": "TextBlock",
    "text": "🔗 body/_ny_c... ✕",
    "size": "Medium",
    "weight": "Bolder",
    "spacing": "None"
  }
],
"verticalContentAlignment": "Center"
}
},
{
  "type": "TextBlock",
  "text": "Opportunity Amount and Quote Amount exceed the
allowed variance of +/- 15% and cannot be moved to Sales for
processing. Update the Opportunity Amount and/or Quote Total
```

- Use outputs from previous actions and variables to provide **context** for the user
- This is an **iterative process** to get the outputs correct
- Use the **new Power Automate Expression editor** for a better user experience – see Experimental Features
- **@Data.Community.Display.Display.V1.FormattedValue** and **first()** are your friends when working Dataverse data!

Send User message



Opportunity 1

A. Datum Corporation (sample)

Opportunity Amount and Quote Amount exceed the allowed variance of +/- 15% and cannot be moved to Sales for processing. Update the Opportunity Amount and/or Quote Total Amounts and resubmit Flow.

Opportunity Amount \$35,000.00

Quote Amount \$25,000.00

Percent Difference -28.57 %

[Open Opportunity](#)

- Include corrective action message
- Card has context for user
- Include link back to record



Flow Build Demo

Expressions, Adaptive Cards Designer, dynamic data

- Generate **link** to **Opportunity** record
 - Build **Adaptive Card**
 - Add dynamic data to Card
 - Send Card to user if business requirements not met
-

Power Automate

Search

Environments
Norm Young's Environ...

?

?

Home

Create

Templates

Learn

My flows

Approvals

Solutions

Process mining

AI models

Desktop flow activity

More

Power Platform

Move Opportunity to Sales

Undo

Redo

Comments

Save

Flow checker

Test

✓ Your flow is ready to go. We recommend you test it

Scope - Try

Get a row by ID - Opportunity

List rows - Quotes

Compose - varPercentDiff

Set variable - varPercentDiff

Condition - Check percent diff between Opportunity and Quote amounts

Percentage difference between Opportunity amount and Quote amount cannot exceed 15 percent

fx and(...) X

is equal to

fx true X

+ Add

✓ If yes

Add a new row - Sales

Add an action

✗ If no

Add an action

Part 4

Config, Debug &
Admin experience

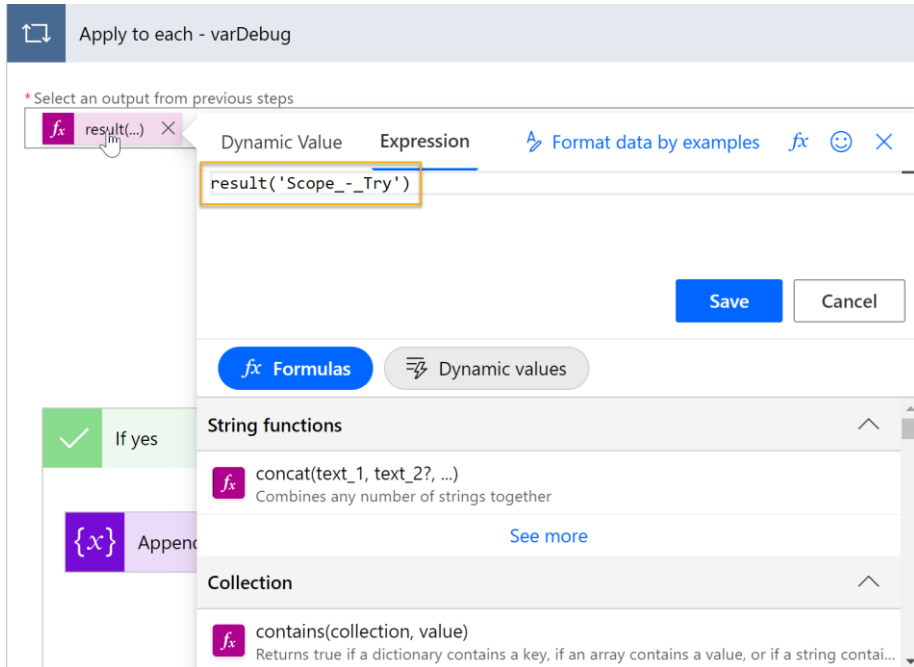
Debug Variable

- Capture all action outcomes

Catch Scope

- **Send Admin message** with link to job

Debug info



- **Append statuses** from **Try** actions into *varDebug*
- Shows each action and the outcome

Expressions

Dynamic Value Expression [Format data by examples](#)

```
setProperty(variables('varParameters'), 'varFlowLink', concat('https://  
make.powerautomate.com/manage/environments/', outputs  
( 'Compose_-_Get_run_time_workflow_details' )?[ 'tags' ]?[ 'environmentName' ]  
 . '/flows/'. outputs( 'Compose - Get run time workflow details' )?[ 'name' ]
```

[Save](#) [Cancel](#)

Formulas Dynamic values

String functions

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

[See more](#)

Collection

contains(collection, value)
Returns true if a dictionary contains a key, if an array contains a value, or if a string contain...

Build dynamic link to **Flow run** screen:

- Uses output from **workflow()** function; provide execution details on current Flow run
- Game changer for Admins!

Send Admin message



Move Opportunity to Sales

Flow error please investigate

List of action names and statuses:

- Name: List_rows_-_Quotes
- Status: Skipped
- Name: Compose_-_varPercentDiff
- Status: Skipped
- Name: Set_variable_-_varPercentDiff
- Status: Skipped
- Name: Condition_-_Check_percent_diff_between_Opportunity_and_Quote_amounts
- Status: Skipped
- Name: List_rows_-_Model-driven_Apps
- Status: Skipped
- **Name: Get_a_row_by_ID_-_Opportunity**
- **Status: Failed**

Open Flow

- Build with **Adaptive Cards Designer**
<https://adaptivecards.io/designer>
- Include debug information
- Include link to failed job

Power Automate

Home

Create

Templates

Learn

My flows

Approvals

Solutions

Process mining

AI models

Desktop flow activity

More

Power Platform

Search

Environments

Norm Young's Environ...

?

Undo

Redo

Comments

Save

Flow checker

Test

←

Move Opportunity to Sales

When a record is selected

{x}

Initialize variable - varDebug

{x}

Initialize variable - varParameters

{}

Compose - varParameters

Scope - Try

Scope - Catch

Scope - Finally

+ New step

Save

Ask a chatbot

Part 5

Logging & Putting it
all together

Finally Scope

- **Insert new row** with Flow job status

User experience

- **Failed business logic requirements** User experience
- **Failed Flow** Admin experience

Accounts My Active Accounts - x

Edit your flow | Power Automate x

Teams and Channels | General | x

+

make.powerautomate.com/environments/3c586091-d3e1-e78d-830f-b5e0bfabddfd3/solutions/d8dfa31d-5829-ee11-bdf4-000d3ae935ac/flows/5d96c840-e2cf-40b2-b242-85cccd2d18c5?backUrl=%2Fenvironments%2F3c586091-d3e1-e78d-830f-b...

Apps Norm

Power Automate

Search

Environments Norm Young's Environ...

Undo Redo Comments Save Flow checker Test

Home

Create

Templates

Learn

My flows

Approvals

Solutions

Process mining

AI models

Desktop flow activity

More

Power Platform

Move Opportunity to Sales

When a record is selected

Initialize variable - varDebug

Initialize variable - varParameters

Compose - varParameters

Scope - Try

Scope - Catch

Scope - Finally

+ New step

Save



Read now!!!

Elevate Your Power
Automate Workflows
with Error Handling:
Using Adaptive Cards
in Microsoft Teams



Thanks!!!

Let's keep the conversation going:



normyoung.ca



[norm-young](https://www.linkedin.com/in/norm-young)



[stormin_30](https://twitter.com/stormin_30)