

2024 Power Automate Coding Standards



Table Of Contents

Introduction	4
Naming Conventions	5
Why Use Power Automate Naming Conventions?	
Action Names	
Trigger Names	
Variable Names	
Connection Reference Names	7
Variables	9
How Should I Store Values In A Flow?	9
Variables	9
Compose Action	
Environment Variables	
Configuration Data Table	
Get A Single Key-Value Pair From Configuration Data	
Retrieve All Key-Value Pairs For A Configuration Data Group	15
Commenting Code	
Why Write Code Comments?	
Tips For Writing Good Comments	
Display Power Automate Expressions In The Flow Comments	
Do Not Use The Collaborative Comments Feature For Coding Comments	
Connection References	
Use Connection References Instead Of Connections	
Remove Duplicate Connection References From Solutions	
Assign Connection Ownership To Service Accounts	
Error-Handling	
Try, Catch, Finally Pattern	
Group Multiple Actions Inside Of A Scope Action	
Terminate The Flow With A Status Of Success Or Failure	
Notify The Process Owner Of The Flow Failure	

	Get The Flow Error Message For The Failed Action	30
	Include The Flow Run History URL	33
Pei	formance Optimization	35
	Enable Concurrency Control In Apply To Each Loops	35
	Execute Flow Actions In Parallel Branches	36
	Use Filter Queries When Retrieving Large Datasets	37
	Reduce Table Sizes By Selecting Columns	38
	Stay Below The Data Connector API Limits	40
	Identify Bottlenecks With Process Advisor	41
	Favor Compose Over Declaring Variables When Creating Arrays	42
	Exploit Batch Update APIs	43
	Avoid Using On-Premise Actions	44
Flo	w Architecture & Design Tips	45
	Writing Reusable Code	45
	Security Architecture	48
	Platform Considerations	50
	Designing Flow Triggers	53
	General Coding Patterns	56

Introduction

Welcome to the Power Automate Coding Standards for cloud flows.

In this guide you will find 60+ pages of coding rules, guidelines and best practices I use everyday to create Power Automate cloud flows. I have spent the last 3 years building Power Automate flows every day. Now I want to share the knowledge I've gained in this set of easy-to-understand, actionable examples.

Power Automate already has an <u>official set of cloud flow coding standards</u> released back in 2019. So why did I make my own? A few reasons:

- I wanted an updated set of standards and guidelines for 2024 that includes all of the latest features
- These coding standards can be continuously improved as new Power Automate features hit "general availability" in 2024, 2025, 2026 and beyond
- Readers can leave a comments on <u>my website</u> describing their own best practices which I can incorporate into future versions

I hope you enjoy my Power Automate Coding Standards For Cloud Flows.



Support The Site 💙

I don't have ads on my website because I believe in delivering the best learning experience possible. This website is paid for out of my own pocket. If you've found value in the free resources I've created, I'd love to have your support.

Click on the "Buy Me A Cat Treat" button below to support the site.



Naming Conventions

Why Use Power Automate Naming Conventions?

A consistent naming pattern makes it easier for other developers to understand what a flow does without having to review the details of every individual action. Cleanliness of the Power Automate code speeds up work and reduces bugs.

Flow Names

A flow name should begin with a <u>verb</u> (action word), describe what outcome the flow will achieve and include the trigger type. Use proper case. Be as concise as possible.

Good Examples	Bad Examples	Bad Reason
Send Daily Status Report (Scheduled)	Send Daily Status Report	No trigger type at end of name
Get Currency Rates From Web (Automated)	Currency Rates From Web	No verb at beginning of name
Change App Ownership (Instant)	Change app ownership (Instant)	Not proper case

Prefix the flow with a group name at the start when it is part of a set of related flows. Do this to communicate the relationship between flows. Not every flow built requires a group name. If a flow does not belong to a set then do not add a group name.

Examples With Group Name

Generate Contract: Create & Send PDF Contract (Automated)

Generate Contract: Calculate Line Item Prices (Instant)

Generate Contract: Assemble Terms & Conditions (Instant)

Action Names

A flow action name must always start with the action's original description. It is important to include this so developers can understand which action was selected without having to expand it. Then add more details about what the action is doing within the context of the current flow.

Use proper case. Separate the action name and its context using a colon.

Good Examples	Bad Examples	Bad Reason
Get Items: Inspections Table	Download Items From The Inspections Table	Original SharePoint connector action name missing
Compose: Email message body	Compose	Missing description
Send Email (V2): Daily Report To Operations Leadership	Send Email (V2) – Daily Report To Operations Leadership	Used a dash instead of a colon

Trigger Names

Automated flow trigger names should include the table or event name in the trigger name.

Example	
When An Item Is Created: Safety Incidents	

Scheduled flow trigger names must display the recurrence schedule.

Example

Recurrence: On The 1st & 15th Of The Month At 5PM

Instant flows trigger names should also describe the event that triggers them.

Example

Manually Trigger A Flow: When An Admin Wants To Create A New SharePoint Team Site

Power Apps flow trigger names should describe the event the app which triggers the flow to start.

Example
Power Apps (V2): When The User Submits A New File To Upload

Variable Names

A flow variable name should begin with a prefix of "v" and describe its subject/purpose in concise manner. Use camel case. with no spaces between each word. Be as concise as possible.

Good Examples	Bad Examples	Bad Reason
vQuantityInBox	varQuantityInBox	Do not prefix with var
vEmployeeFullName	strEmployeeFullName	Prefix str to denote a string data type is not necessary
vlsPastDue	vispastdue	Use camel case

Connection Reference Names

A connection reference name starts with a noun to describe the connection account. Then it is followed by the connection type, the solution name and a unique identifier. These are automatically added when building inside of a solution.

Good Examples	Bad Examples	Bad Reason
SvcAcct-SharePoint CurrencyExchangeRates-ac3d5	SharePoint CurrencyExchangeRates-ac3d5	Missing noun at beginning
SvcAcct-Dataverse CurrencyExchangeRates-be005	SvcAcct-Dataverse	No solution name or unique identifier found
SystemAlerts-Office 365 Outlook CurrencyExchangeRates-32a0c	DavidJohnson-Office 365 Outlook CurrencyExchangeRates-32a0c	Noun is not generic.

Variables

How Should I Store Values In A Flow?

Variables, the Compose action, and Environment Variables, and Configuration Data all have a useful role to play in Power Automate flows. This section discusses where and when to use each of them.

Variables

<u>Variables store values</u> that can be referenced in other flow actions. Use a variable when the value will need to be updated at multiple points during the flow run.

Avoid creating variables to store constants. A constant is a value supplied when the variable is initialized that does not change during the flow's execution.

Do use a variable to store the result of a calculation. Do not use variables when a direct reference can be made in the flow action.

$\{x\}$ Set varia	ble: vInvoiceTotal	?
*Name	vInvoiceTotal	\vee
*Value	f_x add() \times	

Compose Action

Variables cannot be used inside of Apply To Each loops when Concurrency is enabled. Substitute the <u>Compose action</u> for variables within loops to maintain the performance benefit of parallelism.

When a value is stored inside of a Compose action it cannot be updated. Only use it to store values that will not change.

The Compose action is often cited as an alternative to Power Automate variables. Don't overuse this technique. Compose actions can make flows less readable at a glance because they always show the word Output when referencing a prior action.

{ / Compos	e: Invoice Line	···
* Inputs	<pre>{ 'ItemID': ProductCode x , 'Description': Title x , 'Quantity': Quantity x , 'UnitPrice': UnitPrice x , 'ExtPrice': fx mul() x }</pre>	

Environment Variables

A developer following ALM best practices will create flows in a development environment, then move the flow to a testing environment for testing and into a production environment for go-live. The flow will often need to behave differently in each environment. Use <u>environment variables</u> to store configuration values and avoid hardcoding them into the flow. Editing a flow inside a managed solution will cause an unwanted unmanaged layer to appear.

For example, a flow sending an email targets an email distribution list while in production but only sends messages to the developer while in development and test environments.

Leverage a single environment variable to replace several of the same hardcoded values across multiple flows.

			Muvanceu i ···································	Kemove ~		Edit Email Notifications Ta ×
						Display name *
urrency Exc	change ALM Sam	ole > En	vironment vari	ables		Email Notifications Target
o Ie	Display name ↑ 🖂		Name \vee	Туре	Manage	Name * 🛈
o 🗉	Email Notifications Tar	i	md_EmailNotific	Environment Vari	No	md_EmailNotificationsTarget
						Description
						Data Type *
						Me Text
						Default Value
						md@matthewdevaney.com
						Current Value
						Override the default value by setting the current valu for your environment.
						sgreen@matthewdevaney.com; djoh
						Save Cancel
						Save Cancel
						Save Cancel
				\downarrow		Save Cancel
Se	end an email (V	(2)		+		Save Cancel
• To	end an email (V	(2) @ Er	nail Notificati	• ×		Save Cancel
€ Subject	end an email (V	'2) [@] Er Daily Exc	nail Notificati hange Rate - L	· × JSD to CAD		Save Cancel
To * Subject * Body	end an email (V	(2) [@] Er Daily Exc Cont	nail Notificati hange Rate - U ▼ 12	• × JSD to CAD	<u>J</u>	Save Cancel ⑦ ··· □ □ □ □ □ □ □ ● ♡ ×
To * Subject * Body	end an email (V	(2) [@] Er Daily Exc Font {♥} O	nail Notificati hange Rate - U ▼ 12 utput ×	↓ . × JSD to CAD 2 ▼ B <i>I</i> I	J /	Save Cancel ⑦ … □ □ □ □ □ □ ○ ②
* To * Subject * Body	end an email (v	2) ⓐ Er Daily Exc Ont ⟨♥} Ou	nail Notificati hange Rate - L	• × JSD to CAD		

Configuration Data Table

An environment variable should not be created when it will only be used in a single flow. Instead, create a table called *Configuration Data* and use it to hold key-value pairs containing the flow settings.

The Configuration Data table includes the following single-line text columns:

Key – describes the key-value pair, must be unique to the group.

Value – the value to be used inside the flow

Group – identifies a set of key-value pairs, making it possible to get all needed key-value pairs at once

ID	Кеу	Value	Group
1	DistributionList	alerts@matthewdevaney.com	DailyNotificationsFlow
2	EnableSendingEmail	true	DailyNotificationsFlow
3	DelaySeconds	60	DailyNotificationsFlow
4	OutputFolder	C:/Documents/Sales Reports	GenerateReportFlow
5	DelaySeconds	15	GenerateReportFlow

Replace as many hardcoded values in a flow as possible. This will eliminate the need to redeploy a flow to make changes to its behavior.

Get A Single Key-Value Pair From Configuration Data

A cingle value kov value	nair can be obtained i	using the Cat Itam	(SharaDaint) ar Cat Dr	www.(Datawarca).action
A Siligle value Kev-value	Dall call be obtailled i			JW (Dalaverse) action.

* Site Address	Matthew Devaney Blog -	
	https://matthewdevaney.sharepoint.com/sites/MatthewDevaneyBlog	\checkmark
*List Name	Configuration Data	\checkmark
*Id	4	
Limit Columns by View	Avoid column threshold issues by only using columns defined in a view	\checkmark

To use the configuration data value select it from the dynamic content menu.

ld <	Dynamic content Expression
_	Search dynamic content
	Get item: Configuration Data - Output Folder
	ID List item id. Use this value for specifying the item to act o.
	S Title
	бу Кеу
	S Value
	Group
	S Color Tag
	Compliance Asset Id

Retrieve All Key-Value Pairs For A Configuration Data Group

All key-value pairs for a group (e.g. DailyNotificationsFlow) can be retrieved using this flow pattern.

* Site Address	Matthew Devaney Blog -	\checkmark
*List Name	Configuration Data	~
Limit Entries to Folder	Select a folder, or leave blank for the whole list	6
Include Nested Items	Return entries contained in sub-folders (default = true)	\sim
Filter Query	Group eq 'DailyNotificationsFlow'	
Order By	An ODATA orderBy query for specifying the order of entries.	
Top Count	Total number of entries to retrieve (default = all).	
Limit Columns by View	Avoid column threshold issues by only using columns defined in a view	\checkmark
Hide advanced options	^	
{ ▽ } Select: Key-Va	lue Pairs	
{▽} Select: Key-Va *From	lue Pairs	
{▽} Select: Key-Va *From *Map	lue Pairs iso Value x iso Key x iso Value x	… × 宿

{	e: Key-Value Pairs Object	•••
first(json(rep	lace(replace(string(body('Select:_Key-Value_Pairs')), '},{', ','),'null,','""',')))	\times
* Inputs	f_x first() ×	
⟨↗⟩ Parse JS0	ON: Configuration Data	
*Content	Outputs ★	
* Schema	<pre>{ "type": "object", "properties": { "DistributionList": { "type": "string" }, "EnableSendingEmail": { "type": "string" }, "DelaySeconds": { } } }</pre>	
	Generate from sample	

The following expression is used in the *Compose: Key-Value Pairs Object* flow action.



Configuration data values for the group can be accessed in the dynamic content menu.

Dynamic content Expression	
Search dynamic content	
Parse JSON: Configuration Data	
Ø Body	
 DistributionList 	
EnableSendingEmail	
DelaySeconds	
Compose: Key-Value Pairs Object	
⊘ Outputs	

Commenting Code

Why Write Code Comments?

Write comments to describe the <u>intended goal</u> of a section of Power Automate code. Knowing the intended goal helps identify mismatches between it and the actual code outcome. Code that has comments takes significantly less time for other developers to understand.

Tips For Writing Good Comments

Use these suggestions to write high-quality comments:

- Do write comments that describe the intent of a code section. Intent means the goal.
- Do not write comments that simply restate what is already made clear in the flow action's name
- Do keep comments updated when the goal of a code section changes.
- Do not write so many comments they are impossible to maintain. Comments create their own technical debt.
- Do write comments in full sentences and use plain language
- Do not use abbreviations, acronyms or slang

Get items: Car	Inventory C	0			
Use the Power Apps Inventory list's Title	search bar value to perform a search for matching substrings in the Car and CarModel columns.	×			
Site Address	Matthew Devaney Blog -				
	https://matthewdevaney.sharepoint.com/sites/MatthewDevaneyBlog	\sim			
List Name	Car Inventory	\sim			
imit Entries to Folder	Select a folder, or leave blank for the whole list	đ			
nclude Nested Items	Return entries contained in sub-folders (default = true)				
filter Query	substringof(" searchTerm × ', Title) or substringof(" searchTerm × ', CarModel)				
Drder By	An ODATA orderBy query for specifying the order of entries.				
l <mark>op Count</mark>	100				
Limit Columns by View	Avoid column threshold issues by only using columns defined in a view	V			

Display Power Automate Expressions In The Flow Comments

Copy and paste Power Automate expressions into the comments section so they can be quickly browsed. The flow does not show the full expression without hovering on it or taking additional steps to open the code editor. Make sure to update the comments each time the expression is changed.

{\$	Compose: Updated Invoice Amount	···· (2)
بت ا	dd(items('Apply_to_each:_Company_Invoice')?['InvoiceAmount'], 10)	×
* Inpu	f_x add() ×	

Do Not Use The Collaborative Comments Feature For Coding Comments

Collaborative comments are stored in a <u>Dataverse</u> table named *Comment*. They are not stored in the flow definition. When a flow is exported to another environment, its comments do not come with it. This is why collaborative comments should be used to conduct code reviews and not to comment your flows.

		0		Comment	s	
(PowerApps	(V2)	U				
Create file:	Agreement To Purchase	0		Matthe Update t	w Devaney the site address to a	n
* Site Address	Matthew Devaney Blog - https://matthewdevaney.sharepoint.com/sites/MatthewDevaney	Blog 🗸		environn different	nent variable. This w across DEV, TEST ar	ill be nd PROD
* Folder Path	/Agreement To Purchase	(a	@menti	on or reply	
" File Name * File Content	ile.name × image: File Content ×			Matthe Add a Ti	w Devaney tle, All files in the do	cument
				library st		
S Undate file		•		library sł	on or reply	
Update file	properties: Contracts Matthew Devaney Blog - https://matthewdevaney.sharepoint.com/sites/MatthewDevaney	Blog V	••••	library si	on or reply	
Update file *Site Address *Library Name	properties: Contracts Matthew Devaney Blog - https://matthewdevaney.sharepoint.com/sites/MatthewDevaney Contracts	Blog V	••••	library si	on or reply	
Update file * Site Address * Library Name * Id	properties: Contracts Matthew Devaney Blog - https://matthewdevaney.sharepoint.com/sites/MatthewDevaney Contracts I temId x	Blog V		library si	on or reply	
* Site Address * Library Name * Id Title	properties: Contracts Matthew Devaney Blog - https://matthewdevaney.sharepoint.com/sites/MatthewDevaney Contracts Intermid x	Blog V		library si	on or reply	
* Site Address * Library Name * Id Title CustomerName	properties: Contracts Matthew Devaney Blog - https://matthewdevaney.sharepoint.com/sites/MatthewDevaney Contracts Image: ItemId x Contracts Customer Name x	Blog V		library si	on or reply	
Solution ContractDate	properties: Contracts Matthew Devaney Blog - https://matthewdevaney.sharepoint.com/sites/MatthewDevaney Contracts ItemId x Customer Name x Contract Date x Contract Date x	Blog V		library si	on or reply	
Solution Update file Site Address Library Name Id Title CustomerName ContractDate Industry	properties: Contracts Matthew Devaney Blog - https://matthewdevaney.sharepoint.com/sites/MatthewDevaney Contracts Contracts ItemId × Contract Date × Contract Date × Contract Date × Contra	Blog V		ibrary si	on or reply	

Connection References

Use Connection References Instead Of Connections

Flow actions requiring authentication must use either a connection or a connection reference. *Connections* allow Power Automate to interact with other online services (SharePoint, Outlook, etc). *Connections references* do the same but they hold a reference to a shared connection instead of connecting directly themselves.

Always use Connection References to reduce maintenance efforts. Swapping out connections for a flow action must be done individually. Connection references can changed in one place and will update all connected flows. Fixing broken connections is much faster too.

Connection references in a managed solution can be updated without creating an unmanaged layer. Never directly edit flows in production and test environments to avoid introducing bugs.



Remove Duplicate Connection References From Solutions

There should only be one connection reference within a solution for each online service being accessed. Remove any duplicate connection references from the solution. The exception should be made when an online service must be accessed with multiple sets of credentials. For example, connecting to Outlook with two different Shared Mailboxes.

Ē	Display name \uparrow \checkmark		Name \vee
ę	Dataverse Common	:	md_sharedcommondataserviceforapps_da752
ę	Dataverse CurrencyExchangeRates-b00e5	:	md_sharedcommondataser X Remove Duplicate
Ÿ	Desktop flows	:	md_shareduiflow_a23c4
Ÿ	Excel Online (Business) Common	:	md_ExcelOnlineBusinessCommon
Ÿ	File System	;	md_sharedfilesystem_371cd

Power Automate Configuration Information > Connection references

De-duplicate Connection References In Environments

In addition to removing duplicate Connection references inside of a solution, they must also be de-duplicated within an environment.

When multiple solutions are imported into an environment it is common for them to each include their own connection references.

Assign Connection Ownership To Service Accounts

A Service Account with the System Administrator security role should own the connections being used by connection references. This enables centralized management of connections under an account with elevated permissions.

Do not use a developer's personal account as the connection owner. To change connection details another developer would have to login as the original developer.

Having a Service Principal own the connections would be ideal. However, Power Platform does not support the use of Service Principals under the Per User license. A more costly, yet higher capacity, Per Flow licensing plan is required.



Error-Handling

Try, Catch, Finally Pattern

Check for errors and tell Power Automate how to handle them at possible failure points in the flow. The Try, Catch, Finally pattern for error-handling pattern is found in most programming languages:

Try – Attempt to execute one or more flow actions

Catch – When an error occurs, do these steps next

Finally - always run these actions whether the previous actions succeeded or failed

Here's how to create the Try, Catch, Finally pattern with Power Automate.

⟨𝒫⟩ Compos	e: Test Value	Try	
* Inputs	f_x div() ×		
1	0 🗸		
$\{x\}$ Initialize	variable: vFlowRunFailed	Catch	
* Name	vFlowRunFailed		
* Туре	Boolean		\sim
Value	f_x true \times		
	0		
🔁 Send an	email (V2): Flow Notification	Finally	
*To	Matthew Devaney 🛛 🗙		
*Subject	Your Flow Name - Completed		
* Body	Font ▼ 12 ▼ B	/ ⊻ / ☷ ⋿ ☲ ≡ &	? 🗞 >
	Flow Name: Your Flow Name Status: Completed		

Configure the "run after settings" as shown for the flow actions below.

	\checkmark		
Compose	Test Value		••
* Inputs	<i>f</i> x div() ×		
	î 🗸		
'Initialize variable	: vFlowRunFailed' should run af	ter:	
{∅} Compo Failed, S	ose: Test Value kipped, Timed out	is successful has failed is skipped has timed out	
	Done	Cancel	
		1	
'Send an email (V	2): Flow Notification' should ru	n after:	
{x} Initializ Succeed	e variable: vFlowRunFailed ed, Failed, Skipped, Timed out	 is successful has failed is skipped has timed out 	
	Done	Cancel	

Group Multiple Actions Inside Of A Scope Action

When multiple actions have the same error-handling requirements place them inside a scope action. A scope action holds a set of other actions.

e: Test Value
\checkmark
e: Test Value 2
\downarrow
e: Test Value 3
Add an action
⊙ 🕂
variable: vFlowRunFailed
vFlowRunFailed
Boolean
fx true ×
0 🕀
email (V2): Flow Notification
Matthew Devaney ×
Your Flow Name - Completed
Font ▼ 12 ▼ B <i>I</i> U / 🗄 🗄 🗐 🖗 🔗
Flow Name: Your Flow Name

Then configure the run-after requirements for the scope. Now any action inside the scope will trigger the error-handling behavior.

Itialize variable: vFlowRunFailed' should r	un after:
Scope: Try	is successful
Failed, Skipped, Timed out	🖌 has failed
**	is skipped
	🖌 has timed out
Done	Cancel

Terminate The Flow With A Status Of Success Or Failure

A flow using error-handling will continue to run after it encounters an error. It will end with a status of succeeded even though there was a failure. To overcome this issue, include terminate actions as the final flow steps to ensure to correct status is displayed in the flow run history.

	Condition Condition Is equal to + Add ~	VFlowRu	*
If yes		X If no	
Send an	email (V2): Flow Notification Failed	Send an	email (V2): Flow Notification Succeeded
* To	Many Baker ×	* To	Matthew Devaney ×
* Subject	Your Flow Name - Failed	* Subject	Your Flow Name - Completed
*Body	Font	* Body	Font ▼ 12 ▼ B I U I ⊟ ⊟ ⊡ Ø ⊗ Flow Name: Your Flow Name Status: Completed
Show advanced op	otions ~	Show advanced or	ations 🗸
		_	\downarrow
Terminat	te: Flow Failed ····	Terminat	te: Flow Succeeded
* Status	Failed ~	* Status	Succeeded V
Code	DivideByZero	3	
Message	Error: Cannot Divide By Zero		Add an action
	Add an action		
	And diracion		

Notify The Process Owner Of The Flow Failure

Alert the business process owner when the flow fails to let them know there was an issue. If no one is notified when the flow fails then it will not get fixed and could continue to happen.

* To	Mary Baker X
*Subject	Your Flow Name - Failed
* Body	Font ▼ 12 ■ I U I ≡ ≡ ≡ ≡ ∅ ∅ Flow Name: Your Flow Name Status: Failed Failure Reason: Divide By Zero Image: Status and Status Image: S
Show advanced or	otions 🗸

Get The Flow Error Message For The Failed Action

The reason for flows failures are not often not known in advance. This makes writing an error message before-hand difficult. By using the **actions** expression we can extract a useful error message from the flow action that failed.

То	Mary Baker ×
Subject	Your Flow Name - Failed
Body	Font ▼ 12 ▼ B I U I E E E I B I <
	actions('Compose:_Test_Value')?['error']?['messag

Add a failure reason section to the flow failure notification and use this expression.

actions('Compose:_Test_Value')?['error]'?['message']

When the flow fails an error message will appear in the flow run history.

NPUTS	
nputs	
N "name": "Compose: Test Value".	•
"startTime": "2023-07-24T20:54:39.15897537".	
"endTime": "2023-07-24T20:54:39.1597008Z",	
"trackingId": "eb10a6ae-c5fd-490f-8635-1f1176bffdcb",	-
"clientTrackingId": "08585113748065274146679115067CU05",	
"clientKeywords": [
"testFlow"	•
UTPUTS	
Outputs	
j) "aada", "Dadoaaaat"	×.
code : Badkequest ,	
"error": {	
"code": "InvalidTemplate".	
"message": "Unable to process template language express	sions in
3	
3	

Then the message gets included in the flow failure notification to the business process owner who will take action on it.

INPUT	5	
То		
mbal	ken@matthewdevaney.com	
Subje	t	
Your	- Flow Name - Failed	
Body		
i Stat	low Name: Your Flow Name tus: Failed	1
Fai	lure Reason: Unable to process template language expressions	in
Show	more 🗸	
OUTPL	ΠS	
Click to	download	

Include The Flow Run History URL

A clickable link to the flow run details should be added to the failure notification. The message can be forwarded to a developer who can quickly identify the failed flow run in the history. Then they can review the issue and diagnose the problem. To do this, use this following coding pattern:

Compos	se: Get Workflow Details		····
Inputs	f_x workflow() ×		
	\downarrow		
Composition	se: Flow Run URL		····
Inputs	https://us.flow.microsoft.com/manage	/environments/	
	() tags.environm × /flows/ (Name × /runs/	run.name ×
	1	2	3
	-		
	\vee		~
Send ar	n email (V2): Flow Failure		····
Send ar	n email (V2): Flow Failure		··· ©
Send ar	n email (V2): Flow Failure MB Mary Baker		···· ⑦
Send ar To Subject	n email (V2): Flow Failure Mary Baker × Your Flow Name - Failed		···· ⑦ ····
Send an To Subject Body	n email (V2): Flow Failure Mary Baker Your Flow Name - Failed		?
Send an To Subject Body /	n email (V2): Flow Failure Mary Baker × Your Flow Name - Failed		···· ⑦ ····
Send an To Subject Body Flow Name: Te	n email (V2): Flow Failure MB Mary Baker × Your Flow Name - Failed		?
Send an To Subject Body > Flow Name: Te Status: Failed<	email (V2): Flow Failure Mary Baker Your Flow Name - Failed		?
Send an To Subject Body > Flow Name: Te Status: Failed<	est Flow est Flow h email (V2): Flow Failure		

Flow expression #1 gets the environment's unique identifier.

outputs('Compose:_Get_Workflow_Details')?['tags']?['environmentName']

Flow expression #2 extracts the **flow's** unique identifier.

outputs('Compose:_Get_Workflow_Details')?['name']

Flow expression #3 finds the flow run's unique identifier.

```
outputs('Compose:_Get_Workflow_Details')?['run']?['name']
```

The failure notification email looks like this:

M Your Flow Name - Failure - mbdevaney@gmail.com - Gmail - Google Chrome			×
a mail.google.com/mail/u/0/?ui=2&view=btop&ver=1je30dsc0178v&search=inbox&th=%23thread-f:17	747620	219173	3466
Your Flow Name - Failure > Intox ×			ð
Matthew Devaney to me ▼ Flow Name: Test Flow Status: Failure Click here to check the flow run details (↑ Reply) (↑ Forward)	☆	Ł	1

Performance Optimization

Enable Concurrency Control In Apply To Each Loops

Apply To Each loops run sequentially by default. If there are 20 items to loop over, the flow will run them in order: 1, 2, 3... until it reaches item 20. Enable concurrency control in the Apply to Each action settings to run up to 50 actions at the same time.

Concurrency Co For each loops exec	ontrol ute sequentially by defau	It. Override the default setting to custo	mize the degree of
parallelism.	3		
Concurrency Contro	01		
On On			
Degree of Parallelisr	n	O	20
Tracked Proper	ties		
Key		Value	
	Done	Cancel	

Execute Flow Actions In Parallel Branches

Power Automate executes flow actions in sequence. But actions can also be <u>run in parallel</u> when they are not dependent upon each another. For example, a flow that gets multiple lists of items in sequence may be reorganized to get all lists of items at the same time.

	\$	Manually trigger a flow	۰۰۰ ا	
Get items: Timesheets Approved	Ţ	· ···	Get items: Timesheets Rejected	····
	(2)	Compose: Timesheets Summary	۰۰۰ (۱	
	a	Send an email (V2): Timesheets Summary	Ø ····	
		+ New step Save		

Use Filter Queries When Retrieving Large Datasets

It takes more time to retrieve a large set of records from a datasource than a small set of records. Apply filters when fetching data to reduce the number of rows being downloaded.

If there is no option to filter the records before downloading them use the Filter Data Operation immediately afterwards. Smaller datasets require less iterations in loops.

Example: Dataverse – List Rows action

Dist rows: A	ctive Accounts	⑦ ····
Table name	Accounts	~
Select columns	Enter a comma-separated list of column unique names t	o limit which columns a
ilter rows	statecode eq 0	
Sort By	Columns to sort by in OData orderBy style (excluding loo	okups)
xpand Query	Enter an Odata style expand query to list related rows	
Fetch Xml Query	Enter a Fetch XML query for advanced customization	
Row count	Enter the number of rows to be listed (default = 5000)	
Skip token	Enter the skip token obtained from a previous run to list	rows from the next pag
Partition ID	An option to specify the partitionId while retrieving data	for NoSQL tables

Reduce Table Sizes By Selecting Columns

Similarly, a table with more columns is slower to download than a table with less columns. Always define which columns should be selected to minimize the size of the dataset being downloaded.

Example: SharePoint Get Items action

* Site Address	Matthew Devaney Blog -	
	https://matthewdevaney.sharepoint.com/sites/MatthewDevaneyBlog	\sim
* List Name	Projects Backlog	\sim
Limit Entries to Folder	Select a folder, or leave blank for the whole list	đ
Include Nested Items	Return entries contained in sub-folders (default = true)	\sim
Filter Query	An ODATA filter query to restrict the entries returned (e.g. stringColum	n eq 'stri
Order By	An ODATA orderBy query for specifying the order of entries.	
Top Count	Total number of entries to retrieve (default = all).	
Limit Columns by View	Project Notifications Columns Only	\sim

Example: Dataverse – List Rows action

Table name	Contacts	×
Select columns	contactid, firstname, lastname, address1_name	
Filter rows	Enter an OData style filter expression to limit which rows are li	sted
Sort By	Columns to sort by in OData orderBy style (excluding lookups)	
Expand Query	Enter an Odata style expand query to list related rows	
Fetch Xml Query	Enter a Fetch XML query for advanced customization	
Row count	Enter the number of rows to be listed (default = 5000)	
Skip token	Enter the skip token obtained from a previous run to list rows	from the next pag
Partition ID	An option to specify the partitionId while retrieving data for N	oSQL tables

Stay Below The Data Connector API Limits

Every flow data connector has API limits for throughput. After a set number of API calls per minute the data connector will become throttled to protect the service. Be aware of the API limits for each connector used in a flow. API limits can be found in the Power Automate documentation for the data connector.

Example: Dataverse connector limits

Username	securestring	Username credential	True
Password	securestring	Password credential	True

Throttling Limits

Name	Calls	Renewal Period
API calls per connection	600	60 seconds

Actions

Add attachment	Adds a new attachment to the specified list item.	
Annrove hub site join request	Approve bub site join request. This will return an approval token	

Identify Bottlenecks With Process Advisor

Process advisor tracks the average duration of individual flow actions as well as total duration. <u>Inspect flows using</u> <u>process advisor</u> to see where bottlenecks are occurring. Focus attention on the slowest actions when improving the flow.



Processes > Update Invoices 20X Faster Process > Analytics

Favor Compose Over Declaring Variables When Creating Arrays

The <u>Compose action</u> executes twice as fast as declaring variables. For large arrays this is a huge time saver. Use Compose instead of a variable to build arrays when the array does not have to be modified.

••
×
_

Exploit Batch Update APIs

Look for batching capabilities in APIs. Batching allows us to do a high-volume of transactions at high-speed. For instance, SharePoint lists have <u>batch create</u>, <u>update and delete abilities</u> only accessible through the SharePoint HTTP action. Read the API documentation for the desired service to check if this is available.

Site Address	😢 siteAddress 🗙		×
Method	POST		\sim
* Uri	/_apī/\$batch		
Headers	X-RequestDigest	digest	×
	Content-Type	multipart/mixed;boundary=bat chxactions() ×	×
	Enter key	Enter value	
Body	fr actions(.)	×	

Avoid Using On-Premise Actions

Flow actions that connect to cloud resources are faster than on-premise records. For example, it would be faster to access <u>files stored in OneDrive</u> than <u>files stored on the local File System</u> via a gateway.

Create file	in OneDrive FASTER	(?)
* Folder Path	Ĩ	6
* File Name	companyassets.xlsx	
* File Content	Sile content ×	
Update file	in Local File System SLOWER	? · · ·
Update file	in Local File System SLOWER	· · ©

Flow Architecture & Design Tips

Writing Reusable Code

Create Child Flows To Store Repetitive Logic

A flow that is called by another flow and <u>passes back the result is known as child flow</u>. Child flows should be created when there is repetitive logic used multiple times in the same flow. Or when there is repetitive logic used across multiple flows. Write the child flow once and use it in several locations.

Child flows are also recommended to abstract complex flow logic. Moving sections of code from a lengthy flow into child flows can make it more manageable/easy to understand.

Configuration and display		🏷 Undo 🦿 Redo 📮 Comments	Save 😵
L Manually t	rigger a flow	····	
	\downarrow		
Get Configura	ation Values		
Get Config	uration Dictionary	····	
* Child Flow	Retrieve Configuration Dictionary (Child)	~	
* Automation_Name	SOXAuditRequests		
	\downarrow		
<i>⟨𝒫</i> ⟩ Parse Conf	iguration return string to JSON	····	
Parse JSON		×	
* Content	dictionary x		
* Schema	<pre>{ "type": "object", "properties": { "cPathScreenshotsFolder": "type": "string" }, "cPathLogFile": { "type": "string" }, "cPathArchiveFolder": { "cPathArchiveFold</pre>	: {	

Favor Custom Connectors Over The HTTP Action For Reusability

When no suitable connector exists, we have the option to use an HTTP action or a custom connector to extend Power Automate's capabilities. <u>Custom connectors are preferable</u> because they can be created once and deployed to the rest of the organization for everyone to use. For example, a custom connector to find the US Dollar exchange rate has many use-cases.

		🏷 Undo 🦿 Redo 🏳 Com
Ø Rec	urrence: Every Day At 6AM	
	++	
O List	rows: Currencies	····
Apply	to each: Currency	
*Select an outp	out from previous steps	
© value	×	
Curr	rency Scoop: Convert USD to Another Curre	ency ⑦
* from	USD	
* to	Abbreviation x	
* amount	100	
		n
	Add an action	

Always Build Flows Inside Of A Solution

Build flows inside of a solution so they can be <u>easily transported across environments</u>. As new flow actions are added, and connection references are created, those connection references will automatically be placed in the solution. Environment variables used by the flow should also be added to the solution as well.



Security Architecture

Secure Inputs/Outputs For Passwords, API Keys and Secrets

Credentials coming from Azure Key Vault, CyberArk, etc. should never be displayed in the flow run history. Change the <u>flow action settings to secure inputs/outputs</u> for every step where sensitive information is showing.

Your flow ran successfully.			
	Manually trigger a flow	Os Os	
		↓ ↓	
	Get secret	0s	
	INPUTS		
	Content not shown due to security	configuration	
	content not shown due to secarily	comgutation.	
	OUTPUTS		
	Content not shown due to security	r configuration.	
	Connection: md@matthewdevaney.c	om	
		V 0	
	₩	15	
	INFOIS		
	Content not shown due to security	configuration.	
	Content not shown due to security	configuration.	
	Content not shown due to security OUTPUTS	configuration.	
	Content not shown due to security OUTPUTS	configuration.	
	Content not shown due to security OUTPUTS Status code 200	Show rew outputs	
	Content not shown due to security OUTPUTS Status code 200	Show rew outputs	
	Content not shown due to security OUTPUTS Status code 200 Headers	configuration.	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev	Configuration.	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding	Configuration. Show rew outputs	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding Connection	Configuration. Show rew outputs	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding Connection Vary	Configuration.	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding. Connection Vary Body	Cookie	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding Connection Vary Body	Configuration.	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding Connection Vary Body Realtime Currency Exchan "1. From Currency Exchan "1. From Currency Cade"	configuration.	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding Connection Vary Body Realtime Currency Exchan "1. From Currency Code" "2. From Currency Code"	show rew outputs > Value Churked keep-alive Cookie ge Rate": { "USD", "United States Dollar",	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding Connection Vary Body Realtime Currency Exchan "1. From_Currency Code" "2. From_Currency Code" "3. To_Currency Code" "3. To_Currency Code" "3. To_Currency Code"	configuration. Show raw outputs	
	Content not shown due to security OUTPUTS Status code 200 Headers Kev Transfer-Encoding Connection Vary Body Realtime Currency Exchan "1. From_Currency Code" "2. From_Currency Code" "3. To_Currency Code" "3. To_Currency Code" "3. To_Currency Code" "5. Exchange Rate": "10	<pre>configuration. Show raw outputs > Value Churked keep-alive Cookie ge Rate": { "USD", "USD", "USD", "USD", "USD", "Japanese Ven", 9.24100000", </pre>	

Enable Elevated Permissions By Configuring Run As User

Power Automate flows are run in the context of a user account. The user account is what determines the permissions a flow has. Flows with instant or Power Apps triggers are run in the context of the user who performed the action to start them.

<u>Configure the Run As Users setting</u> to impersonate another user account during the flow run. Use this ability to grant elevated permissions inside the flow that the user would not normally have.



Platform Considerations

Be Aware Of Power Platform Request Limits

There is a maximum amount of <u>Power Platform requests</u> a user can make in a 24 hour period. Most flow triggers and actions count as 1 request. To get a rough of estimate the number of requests a flow will make multiply the anticipated number of flow runs by the total number of actions plus the trigger.

Purchase a Power Automate premium or a Power Automate Process license to increase the request limits if needed. Or reduce the number of flow runs/actions to minimize the number of requests made.

	Power Platfor	m a	dmin	center		Request a report	\times
=			î.	+ New 🖒 Refresh		We'll notify you when your report is ready to download. It could take a few	
ଇ	Home			Downloadable Reports		minutes. Reports expire 30 days after they have been requested.	
A	Environments			Download and view reports. Reports expire 30 days after t	hey have been re	Choose a report	
Ŀ	Analytics	\sim		Depart Tune	Report Date:	Microsoft Power Platform requests	
20	Billing (Preview)	Y		Report type	Report Date:	Scope	
(j)	Settings			Power Platform Request Licensed User	July oth - Augu	Licensed User	
20	Resources	Y		Power Platform Request Non-Licensed User	July 6th - Augu	O Non-licensed User	
C	Help + support			Power Platform Request Per Flow Licensed Flows	July 6th - Augu	O Per Flow Licensed Flows	
G.	Data integration						
(jp)	Data (preview)						
Da	Policies	\vee					
0	Admin centers	\sim	*				
	Power Platform Conference 2023 <u>Register now</u>					Submit Cancel	

Decide Whether To Use Premium Flow Connectors

Any flows containing a <u>premium connector</u> will require additional licensing to run. Consider moving premium functionality into flows run by the service account as opposed to an end-user to save costs. For example, a flow using premium actions to generate and collect a contract signature could be triggered indirectly by the end user and run by the service account.

Always follow the licensing rules set forth by Microsoft. Do not engage in multiplexing activity.



Keep The 30 Days Flow Duration Limit In Mind

After 30 days a flow run will terminate even if it is not done running. For example, a Wait For Approval action that receives no response from the application approver.

To <u>restart a long running flow</u> set a timeout value of 29d in the advanced settings. Then create a parallel branch with actions to re-trigger the flow when only when the previous step times out. Terminate the flow as the final action in this branch.

	Settings for 'Start and wait for an approval'		
	Secure Inputs Secure inputs of the operation.		
	Sectore inputs		
	Secure Outputs Secure outputs of the operation and references of output pr Secure Outputs	operties.	
	(Off		
	Timeout Limit the maximum duration an asynchronous pattern may t timeout of a single request.	ake. Note: this does not alter the request	
	Duration () P29D		
	Retry Policy A retry policy applies to intermittent failures, characterized a addition to any connectivity exceptions. The default is an exp	is HTTP status codes 408, 429, and 5xx, in pomential interval policy set to retry 4 times.	
	Type Default	~	
	Tracked Properties		
	Key Value		
	Done	Cancel	
		-	\downarrow
Condition	•••	'Update item: Timeout For Appro	wal Action' should run after:
		Start and wait for an ap Timed out	oproval is successful has failed is skipped whas timed out
		Do	ne Cancel

Designing Flow Triggers

Replace The Power Apps V1 Trigger With V2

Always use the Power Apps V2 trigger instead of V1. The V2 trigger allows us to define parameters inside the trigger action and set them as required/not required. It also has a new data type called File which allows us to pass in images and attachments from Power Apps.

The V1 trigger is not recommended. Parameters are created by using an "Ask In Power Apps" option in the dynamic content window. This tends to make a mess since parameters cannot be deleted when they are no longer needed.

dFileToDocumentLibrary	🏷 Undo 🤇 Redo 🖵 Comments 🔚 Save
PowerApps (V2) ⑦ …
File Content	t Please select file or image
Customer N	Name Please enter your input
Contract Da	ate Please enter or select a date (YYYY-MM-DD)
Customer T	ype Please enter your input
+ Add an inpu	ut

Apply Select And Filter Criteria To Automated Triggers

Never use the When A Record Is Added, Modified or Deleted trigger for Dataverse without <u>defining Select & Filter</u> <u>criteria</u>. Selecting columns only triggers a flow when the targeted column is modified. Filtering ensures the flow only triggers when a specific criteria are met. A trigger without filter conditions will start after every record added, modified or deleted and its actions will count towards daily Power Platform request limits.

* Change type	Added or Modified	\checkmark
* Table name	Document Automation Processings	\checkmark
* Scope	Organization	\checkmark
Select columns	aib_processingstatus	
Filter rows	aib_processingstatus eq 914700001	
Delay until	Enter a time to delay the trigger evaluation, eg. 2020-01-01T10:10:00Z	
Run as	Choose the running user for steps where invoker connections are used	\checkmark

Create Advanced Trigger Conditions With Flow Expressions

Use Power Automate expressions to create flow trigger conditions that <u>are not available out-of-the-box</u>. For example, the recurrence trigger does not have any option to trigger only on the last Friday of the month. But by opening the recurrence trigger settings and adding trigger conditions it can be achieved.

Custom Tracking Id Set the tracking id for the	run. For split-on this trac	king id is for the initiating r	equest.
Tracking Id			
Loncurrency Contro Limit number of concurre Concurrency control chan Limit Off Trigger Conditions Specify one or more expri	DI nt runs of the flow, or lea ges the way new runs are essions which m <mark>u</mark> st be tru	ve it off to run as many as p e queued. It cannot be undo ue fo <mark>r th</mark> e trigger to fire.	oossible at the same time. one once enabled.
@or(equals(int(utcN	ow('dd')),01),equals(in	t(utcNow('dd')),15))	×

General Coding Patterns

Flatten Nested IF Condition Actions

Multiple IF condition action nested inside of one another make it difficult to understand what will happen next in the flow. To make matters worse, when we open a flow the condition actions are collapsed and we must expand them to see what's going on inside.

Make conditional logic only one layer deep wherever possible. Instead of nesting IFs try to break up logic into several sequential IF statements.



(+)	
Condition High Priority and utstar Type on HR	
And ~	
Varightinia a repuir to	V 16 98 8 110
and the second se	
versuet. a sec.a to	
fyes	fro
Send an email (V2) To Hit Vice President	
	T Add ar action
T And an advan	
÷	
Condition: High Priority and VissueType ne HR	
and v	
	V 1 100 100
_ [] sansuet	a - √ H6
- + Add ~	
100 Aug	No.
1.70 s.20	
Send an email (V2): To Company Executives DL U ····	The last sector
The Acad an action	
÷	
Condition: Low Priority and vissueType og Hi	a (
And V	
	V MAX
Interfigit Rule and a series and a series again	19 V 16
+ Add ~	
V fym	× đro
Send an email (V2): To HR General incuines	
	T Add an actor
Add an action	
TT Condition: You Highly and History he re-	
And ~	
utersweit, x is equal to	V A faise H ····
- D MARGINE * Shot qua	10 V HR
- + ada -	
then.	if no
10 Lond to small ACD Companying States in the	
send an aman (x-p; -companywose cameral incluines 🛛 🗇 ***	T Age an action
The are an action	—
	Ves Ves

Eliminate Unnecessary Apply To Each Loops

The SharePoint – Get Items action can be used to obtain a single record with a matching unique value. For example, get the list item where email address equals matthewdevaney@outlook.com from the Employees SharePoint list. Any subsequent action using the Employee data will be created inside of a loop. The Get Items action always returns a table even if only one item is returned

Write a flow expression that includes the first function to eliminate the Apply To Each loop. It takes the first record from an array and allows access to its values. Loops should not be added to a flow when there will only be one iteration.

* Site Address	Matthew Devaney Blog -	
	https://mattnewdevaney.sharepoint.com/sites/MattnewDevaneyBlog	~
*List Name	Projects Backlog	~
Limit Entries to Folder	Select a folder, or leave blank for the whole list	C
Include Nested Items	ted Items Return entries contained in sub-folders (default = true)	
Filter Query	Title eq 'Expense Report App'	
Order By	An ODATA orderBy query for specifying the order of entries.	
Top Count	1	
Limit Columns by View	Avoid column threshold issues by only using columns defined in a view \checkmark	
	V	
Apply to each Pro	piect Backlog Item	
Apply to each Pro	oject Backlog Item	
Apply to each Pro	oject Backlog Item	
Apply to each Pro	oject Backlog Item vious steps	
Apply to each Pro	oject Backlog Item vious steps mated Hours) .
Apply to each Pro	oject Backlog Item vious steps mated Hours) •



Generate Fetch XML For The Dataverse List Rows Action

When using the Dataverse List Rows action filter the rows with Fetch XML queries instead of OData queries. Fetch XML can be generated by performing an advanced filter in a model driven app and selecting the Fetch XML button. Fetch XML has more query options than OData and does not need to be manually written by the developer.

wing live data Change to retained	i data 🔍		
ND V	∽ Equals	∽ Active ×	×
— 🗌 Modified On	∼ Today	×	
$+$ Add \vee			

V \bigcirc (?) ... List rows: Exchange Rates * Table name Exchange Rates Select columns Enter a comma-separated list of column unique names to limit which columns a Filter rows Enter an OData style filter expression to limit which rows are listed Sort By Columns to sort by in OData orderBy style (excluding lookups) Expand Query Enter an Odata style expand guery to list related rows Fetch Xml Query <fetch version="1.0" mapping="logical" no-lock="false" distinct="true"> <entity name="md_currency"> <attribute name="md_currencyid"/> <attribute name="md_name"/> <order attribute="md_name" descending="false"/> <attribute name="ownerid"/> <attribute name="modifiedon"/> <attribute name="modifiedby"/> <attribute name="md_abbreviation"/> <attribute name="md_symbol"/> <filter type="and"> <condition attribute="statecode" operator="eq" value="0"/> <condition attribute="modifiedon" operator="today"/> </filter> </entity> </fetch> Row count Enter the number of rows to be listed (default = 5000) Skip token Enter the skip token obtained from a previous run to list rows from the next pa-Partition ID An option to specify the partitionId while retrieving data for NoSQL tables Hide advanced options

Use A State Machine Pattern

Suppose there is an approvals workflow with many stages: 1st level, 2nd level, 3rd level. The flow progresses through each stage until it successfully terminates. Now let's add another requirement. When an approval is rejected the workflow must be returned to the previous level. How can we accomplish this without writing a large amount of redundant code?

We can use a <u>State Machine pattern</u> to detect the current state of the workflow and direct it to the next action. To do this, we create a Switch statement inside of a Do-Until loop. At the start of each loop we determine the current state (Level 1, Level 2, Level 3). Then we run the appropriate actions determined by the Switch.

Once all actions for the state are completed the approval is updated to a new state. The Do-Until loop repeats until it meets the looping exit conditions.