

Integrating External Data Using Power Automate



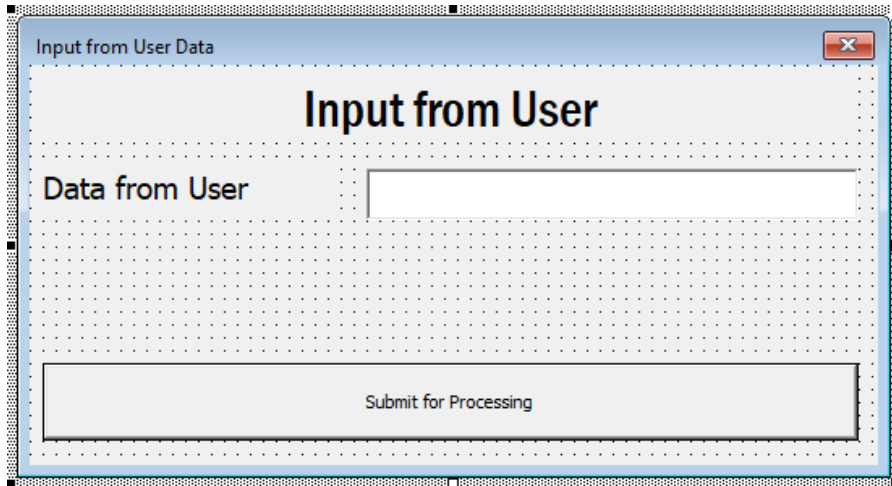
Brent Allen

FREELANCE CONSULTANT

@Macrordinary_CA <http://macrordinary.ca>



VBA Can Get Data Outside Of Ranges



Input from User Data

Input from User

Data from User

Submit for Processing

VBA Input Forms

```
Sub getExternalSpreadsheet()  
    Dim wbExternalWorkbook As Workbook  
    Dim sExternalFilename As String  
  
    sExternalFilename = Application.GetOpenFilename("Microsoft Excel Workbook (*.xlsx), *.xlsx")  
    Set wbExternalWorkbook = Workbooks.Open(sExternalFilename)  
  
End Sub
```

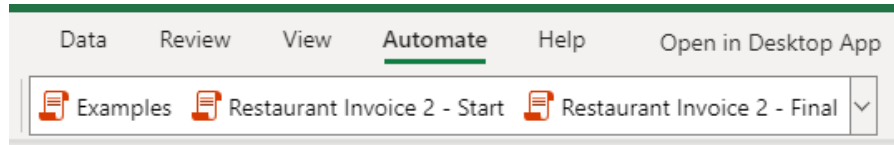
Open Files Through VBA



Office Script has security considerations that make it harder to work with external files.



Office Script Hasn't Operated Outside Excel



Only Triggered from Automate Menu

```
function main(workbook: ExcelScript.Workbook) {  
  // Set names for worksheets.  
  let worksheet = workbook.getWorksheet("Invoice");  
  let worksheet = workbook.getWorksheet("Orders");  
}
```

Workbook
defined in function

Can Only Use Active Workbook



Power Automate can
trigger an Office Script.



What is Power Automate?

Microsoft Power Platform

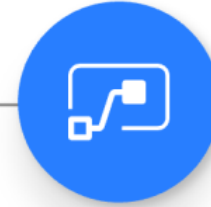
The low-code platform that spans Office 365, Azure, Dynamics 365, and standalone applications



Power BI
Business analytics



Power Apps
Application development



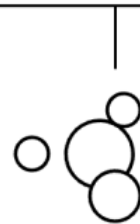
Power Automate
Process automation



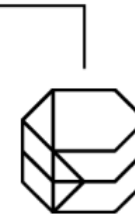
Power Virtual Agents
Intelligent virtual agents



**Data
connectors**



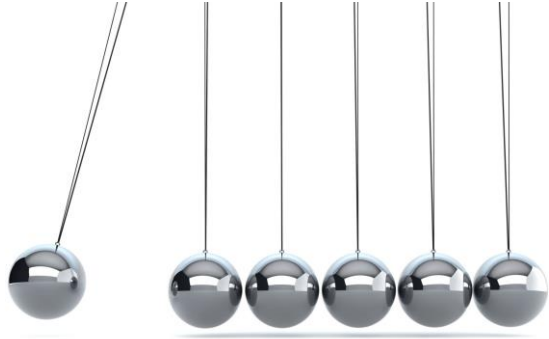
AI Builder



**Common
Data Service**



How Does Power Automate Work?



Triggers



Actions

What Sort of Activities Could Act As Triggers?



A specific time of day



Uploading a file to
SharePoint



Button click in
PowerApps



Configuring the Office Script Action



Getting the Script Ready

There are no “active”
elements of the
workbook

Cannot refer to the
Active Worksheet

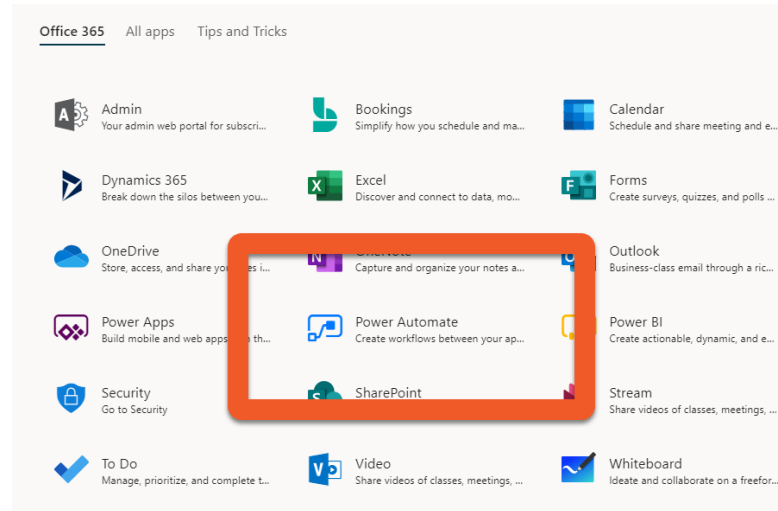
Cannot refer to
Selected Range

Workbook with
specified worksheet
must exist

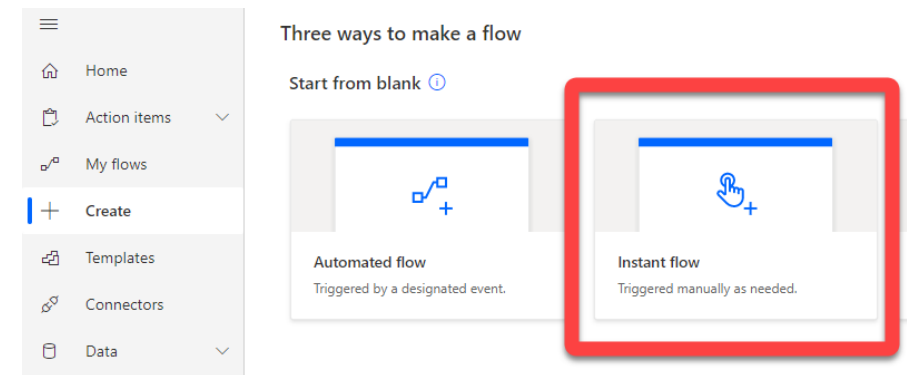
```
1  function main(workbook: ExcelScript.Workbook) {  
2  
3      // THIS WILL FAIL IN POWER AUTOMATE.  
4      // let selectedSheet = workbook.getActiveWorksheet();  
5  
6      // Set worksheet.  
7      let testRunSheet = workbook.getWorksheet("Test Run Sheet");  
8  
9      // THIS WILL ALSO FAIL IN POWER AUTOMATE.  
10     // workbook.getSelectedRange().setValue("This will fail in Power  
        Automate!")  
11  
12     // Set some test text.  
13     testRunSheet.getRange("A1").setValue("This has been entered through Power  
        Automate!");  
14  
15 }
```



Getting Started



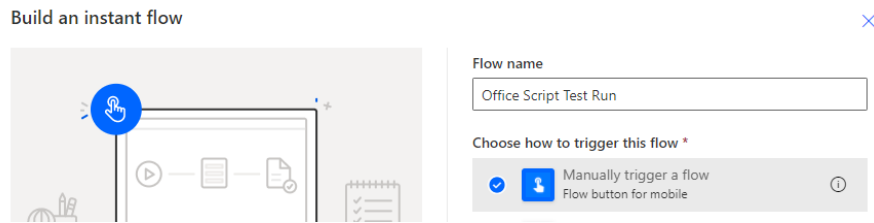
**Log in to Power Automate through
Office 365**



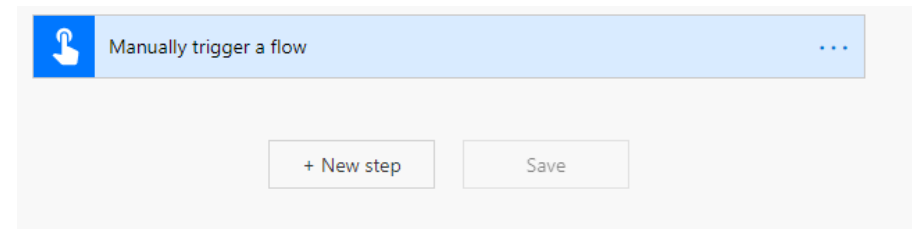
Create an Instant Flow



Creating Your Flow



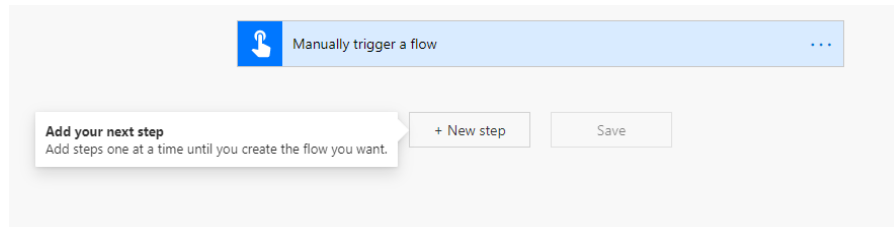
Select Manually trigger a flow and provide a name.



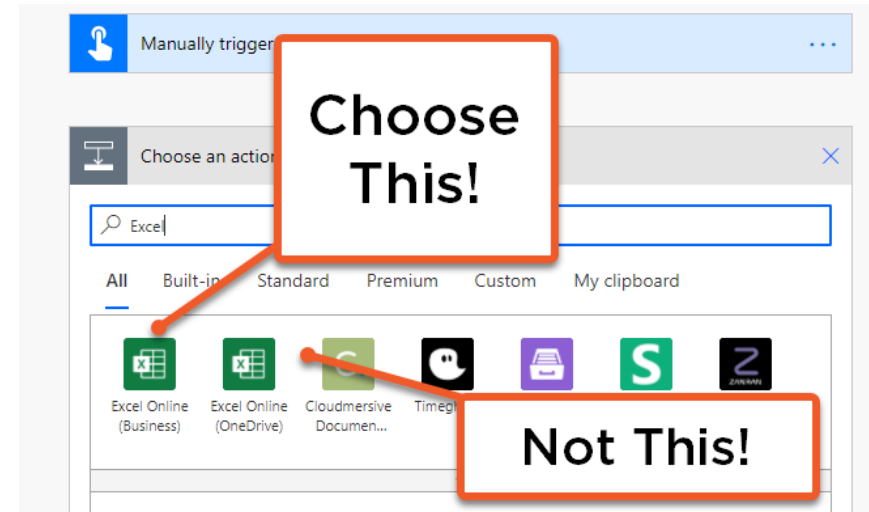
Flow is ready to go!



Add a New Step



Click on the New Step button



Choose action from Excel Online (Business)



Configure the Run Script Action

Select the Run Script action

Choose Excel file location (OneDrive or SharePoint)

Choose library and file

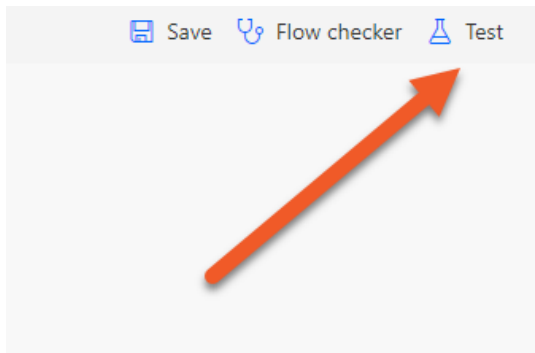
Choose script

The screenshot shows the configuration interface for the 'Run script (Preview)' action in Microsoft Flow. The action is connected to a 'Manually trigger a flow' trigger. The configuration fields are as follows:

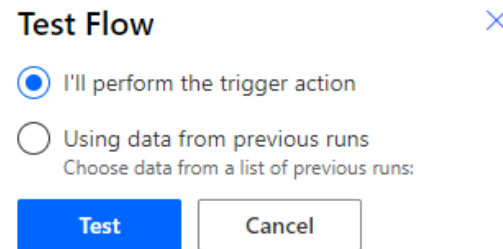
- Location***: A dropdown menu showing 'Group - Pluralsight Courses'.
- Document Library***: A dropdown menu showing 'Documents'.
- * File**: A text input field showing '/Demo Files/Office Script Test Run.xlsx' with a folder icon on the right.
- Script***: A dropdown menu showing 'Office Script Test Run'.



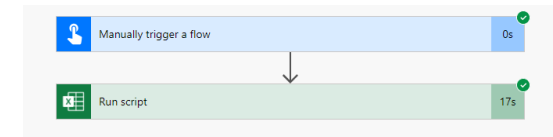
Test the Workflow



Click on Test



Choose I'll perform
the trigger action



Script has run
properly!



Demo



Run the Winger Place script from Power Automate



Adding Parameters to an Office Script



Why Do We Want Parameters?



Get the time when a file was uploaded



Get information from a Power Automate button click



Use fields from a Power App



Steps to Add Parameters to an Office Script



Add parameters into the Main function



Add parameters into the Power Automate action



Use parameters in the Office Script



Add Parameters to the Main Function

```
1  function main(  
2      workbook: ExcelScript.Workbook,  
3      uploadTime: string,  
4      uploadOtherDetails: string[],  
5      uploadPerson?: string  
6  )
```

Add parameters after the workbook

Variable type must be defined (string, number or Boolean)

Can be a single value or an array

Optional values can be added using a question mark after the parameter name



Add Parameters to Power Automate Action

All parameters will be made available once the script is selected

Required values will have red asterisks

New items can be added to arrays with the button

The screenshot shows the configuration interface for the 'Run script (Preview)' action. The fields are as follows:

- Location ***: A dropdown menu showing 'Group - Pluralsight Courses'.
- Document Library ***: A dropdown menu showing 'Documents'.
- * File**: A text field containing '/Demo Files/Office Script Test Run.xlsx' with a folder icon on the right.
- Script ***: A dropdown menu showing 'Office Script Test Run'.
- uploadTime ***: A field with a clock icon and the text 'Converted time' followed by a close button (X).
- *uploadOtherDetails**: A dashed box containing:
 - An 'Items' section with a list containing 'User email' followed by a close button (X).
 - An '+ Add new item' button.
- uploadPerson**: A field with a person icon, the text 'userName', and a close button (X).



Add Parameters to Office Script

Parameter names can
be referred to in main
function.

Optional values not
entered are undefined

Arrays start at zero
index

```
19 // Set some test text.
20 testRunSheet.getRange("A1").setValue("This has been entered through Power
    Automate!");
21 testRunSheet.getRange("A2").setValue("This file was uploaded at " +
    uploadTime);
22
23 if (uploadPerson != undefined) {
24     testRunSheet.getRange("A3").setValue("This file was uploaded by " +
        uploadPerson);
25 } else {
26     testRunSheet.getRange("A3").setValue("The uploader of this file is
        anonymous.")
27 }
28
29 testRunSheet.getRange("A4").setValue("Other details: "+ uploadOtherDetails
    [0]);
30
```



Results in Office Script

```
1 function main(  
2     workbook: ExcelScript.Workbook,  
3     uploadTime: string,  
4     uploadOtherDetails: string[],  
5     uploadPerson?: string  
6 )
```

Added parameters to
main

```
19 // Set some test text.  
20 testRunSheet.getRange("A1").setValue("This has been entered through Power  
21 Automate!");  
22 testRunSheet.getRange("A2").setValue("This file was uploaded at " +  
23     uploadTime);  
24 if (uploadPerson != undefined) {  
25     testRunSheet.getRange("A3").setValue("This file was uploaded by " +  
26         uploadPerson);  
27 } else {  
28     testRunSheet.getRange("A3").setValue("The uploader of this file is  
29         anonymous.")  
30 }
```

Used parameters in
script

	A	B	C	D	E
1	This has been entered through Power Automate!				
2	This file was uploaded at 8/24/2020 1:31 AM				
3	This file was uploaded by Brent Allen				
4	Other details: YnJlbnRABWFjcm9yZGluYXJ5LnNh				
5					

Results exported to
Excel!



Demo



Add parameters from a Power App into a Power Automate

Use parameters in an Office Script



Summary



Configured Power Automate to run an Office Script

Added parameters to an Office Script

Ran an Office Script with parameters in Power Automate



Course Summary



Contrasted Office Scripts to VBA and Office Add-Ins

Enabled and recorded an Office Script

Customized a recorded Office Script

Wrote an Office Script without the Script Recorder

Ran from outside Excel using Power Automate



Thank you and enjoy Office
Scripts!

