Building Desktop Flows

Scenario

In this lab, you will be creating two desktop flows. One to automate the funding Windows app and the other to automate the inspection website.

High-level lab objectives

- Use recording to capture steps to automate for both web and Windows desktop applications
- Use input and output variables
- Use flow control to handle conditional logic
- Use loop condition and wait control
- Use JavaScript to build JSON

Exercise 1: Automate Desktop Funding Application

Task 1: Practice using app

In preparation for recording practice using the app to help eliminate accidental steps from being recorded.

- Browse to **C:\Labs\Resources\Funding manager app** and launch the **Woodgrove Bank Funding Manager.exe** app.
- Enter **your username** for Username, `pass@word1` as Password, and click **Login**.
- Enter `MC3747` for Loan number and click **Lookup**.
- Click **Draw Funds**.
- Enter `100000` for Amount, `123` for Inspection Job #, `30` for Risk Score, check the Borrower Approved Draw checkbox, and click **Draw Funds**.
- Click **OK**.
- Repeat steps **3** to **6** until you feel comfortable with application.
- Close the application.

Task 2: Record using the app

Create a new desktop flow in the **Construction Funding** solution and record the steps using the **Woodgrove Funding Manager** application.

- Navigate to `https://make.powerapps.com/` and make sure you have the Dev environment selected.

Select **Solutions** and open the **Construction Funding** solution.

Click **+ New** and select **Automation | Desktop flow**.

Enter `Woodgrove Funding Manager Draw` for Flow name and click **Launch app**.

Power Automate Desktop should launch.

Go to the **Actions** pane, expand the **System** group, and double click on the **Run application** action.

Click on the **Select file** button of the Application path.

Select the **Woodgrove Bank Funding Manager.exe** file located in the lab folder **C:\Labs Resources\Funding manager app** and click **Open**.

Click **Save**.

Click **Run**.

The application should start, and the flow should complete.

Do not close the application.

Go to the desktop flow and click **Recorder**.

Do not start recording yet. Close or minimize all but the funding manager application. You will be recording steps and modifying steps. Make sure to follow the directions here, and only click Finish when we tell you to.

Remember, you will provide the values below once you start recording.

> Username: **your username**.

> Password: `pass@word1`

> Loan number: `MC3747`

> Amount: `100000`

> Inspection job #: `123`

> Risk score: `30`

DO NOT click the **OK** button on the Draw Confirmation screen.

Click **Record** and select **Next**. On Recorder Pane, click on **Record**.

The recording should start.

Capture steps, same as you practiced.

DO NOT click on the OK button.

On the confirmation screen of the Windows app right click on the **Amount** value and select **Get text(“”).**

Right click on the **Funds transfer number** value and select **Get text(“”).**

Right click on the **Funds sequence number** value and select **Get text(“”).**

Click **OK**.

Go to the recorder and click **Done**.

The steps you recorded should look like the image below.

Close the **Woodgrove Funding Manager** application.

If Run Application is no longer your first step, Drag the **Run application** action from the bottom to the top of the steps.

Click **Run**. Do not interact with your computer until the run completes.

Your flow variables should now look like the image below but might have different values.

Click **Save** and wait for the flow to be saved.

Do not close the desktop flow.

Task 3: Add Input and output variables

In this task, you will define the input and output variables that you'll use to replace the hard coded values recorded. The input variables will be used to pass data from the calling cloud flow. The output variables will be used to return data to that flow.

On the **Variables** pane, click the plus button **(+)** and select **Input** to add a new input variable.

Configure the variable as follows:

> Variable name: `Username`

> Data type: Text

> Default value: **Your Username**

> External name: `Username`

> Description: `Username`

Mark the variable as sensitive, and click **Save**.

Click **(+)** and select **Input** again.

Enter `Password` for Variable name, select **Text** for Data type, `pass@word1` for Default value, `Password` for External name, `Password` for Description, mark the variable as sensitive, and click **Save**.

Click **(+)** and select **Input** again.

Enter `LoanNumber` for Variable name, select **Text** for Data type, `MC3747` for Default value, `LoanNumber` for External name, `Loan number` for Description, and click **Save**.

Click **(+)** and select **Input** again.

Enter `RequestedAmount` for Variable name, select **Text** for Data type, `100000` for Default value, `RequestedAmount` for External name, `Requested amount` for Description, and click **Save**.

Click **(+)** and select **Input** again.

Enter `InspectionJobNumber` for Variable name, select **Text** for Data type, `123` for Default value, `InspectionJobNumber` for External name, `Inspection job number` for Description, and click **Save**.

Click **(+)** and select **Input** again.

Enter `BorrowerApproved` for Variable name, select **Text** for Data type, `Yes` for Default value, `BorrowerApproved` for External name, `Borrower approved` for Description, and click **Save**.

Click **(+)** and select **Input** again.

Enter `RiskScore` for Variable name, select **Text** for Data type, `30` for Default value, `RiskScore` for External name, `Risk score` for Description, and click **Save**.

You should now have 7 input arguments. They're displayed in alphabetical order.

Select **(+)** and select **Output**.

Enter `FundedAmount` for Variable name, select **Text** for Data type, `FundedAmount` for External name, `Funded amount` for Description, and click **Save**.

Click **(+)** and select **Output** again.

Enter `FundTransferNumber` for Variable name, select **Text** for Data type, `FundTransferNumber` for External name, `Fund transfer number` for Description, and click **Save**.

Click **(+)** and select **Output** again.

Enter `FundSequenceNumber` for Variable name, select **Text** for Data type, `FundSequenceNumber` for External name, `Fund sequence number` for Description, and click **Save**.

Click **(+)** and select **Output** again.

Enter `FundingStatus` for Variable name, select **Text** for Data type, `FundingStatus` for External name, `Funding status` for Description, and click **Save**.

You should now have **11** total variables 7 inputs and 4 outputs.

Click **Save** and wait for the flow to be saved.

Task 4: Change flow to use input variables

In this task, you will modify hard coded values in the steps to use the input variables you just defined.

Go to the populate username action and double click on it.

Remove the current value and click on the **{x}** select variable button.

Select **Username** for variable and click **Select**.

Click **Save**.

Go to the populate password action and double click on it.

Click on then encryption button and select **Input as text, variable or expression**.

Click on the **{x}** select variable button.

Select **Password** for variable and click **Select**.

Click **Save**.

Go to the populate loan number action and double click on it.

Remove the current value and click on the **{x}** select variable button.

Select **LoanNumber** for variable and click **Select**.

Click **Save**.

Go to the populate amount action and double click on it.

Remove the current value and click on the **{x}** select variable button.

Select **RequestedAmount** for variable and click **Select**.

Click **Save**.

Go to the populate inspection job # action and double click on it.

Remove the current value and click on the **{x}** select variable button.

Select **InspectionJobNumber** for variable and click **Select**.

Click **Save**.

Go to the populate risk score action and double click on it.

Remove the current value and click on the **{x}** select variable button.

Select **RiskScore** for variable and click **Select**.

Click **Save**.

The actions you changed should now look like the image below.

Click **Save** and wait for the flow to be saved.

Task 5: Change to populate output variables

In this task, you will modify hard coded values in the steps to use the output variables you just defined.

In your steps, locate the **Get details of a UI element in window** action for the amount and double click on it.

Expand the **Variables produced section** and click on the **{x}** select variable button.

Select **FundedAmount**.

Click **Save**.

Locate the **Get details of a UI element in window** action for the transfer number and double click on it.

Expand the **Variables produced section** and click on the **{x}** select variable button.

Select **FundTransferNumber**.

Click **Save**.

Locate the **Get details of a UI element in window** action for the sequence number and double click on it.

Expand the **Variables produced section** and click on the **{x}** select variable button.

Select **FundSequenceNumber**.

Click **Save**.

You should now have three output variables populated.

Click Save and wait for the flow to be saved.

Task 6: Handle denied draws

In this task, you will handle if the funding is denied by adding conditional logic to the flow. This will ensure the flow does not have an error when certain controls are not available, and you will return an output variable indicating that the funding was denied.

Open the Woodgrove Bank Funding Manager application and go through the steps with the following parameters to get the denied message.

> Username: **your username**.

> Password: `pass@word1`

> Loan number: `MC3747`

> Amount: `100000`

> Inspection job #: `123`

> Risk score: `90`

Leave the denied message and don’t click OK.

In Power Automate Desktop, go to the **Actions** pane and expand the **UI automation** group.

Drag the **If window contains** action and drop it above the **Get details of the UI element in window** of the amount.

Click the UI element dropdown and then click **Add UI element**.

Hold the Ctrl key and Click on the text **Draw denied contact bank**.

Click **Save**.

Click on the **…** More actions button of the Get details of a UI element in window action for the amount.

Select **Copy**.

Right click on the Get details of a UI element in window action for the amount and select **Paste**.

You should now have two the Get details of a UI element in window action for amount.

Drag the first Get details of a UI element in window action for amount and drop it inside the **If window** contains.

Double click on the Get details of a UI element in window action for amount inside the If.

Click on the **UI element** dropdown and select **Draw denied contact bank** and click **Select**.

Expand the **Variables produced** and click on the **{x}** select variable button.

Select **FundingStatus**.

Click **Save**.

Drag another If window contains action and drop it below the **End** if.

Click on the **Check if window** dropdown and select **Doesn’t contain UI element**.

Click on the **UI element** dropdown select **Draw denied contact bank** and click **Select**.

Click **Save**.

Move all three Get details of a UI element in window actions outside of the If and drag them to the inside of the second if condition.

Expand the **Variables** action group and drag **Set variable** action to the second If window contains.

Click **{x}** select variable select **FundingStatus**.

Type `Approved` for Value and click **Save**.

Task 7: Add Close of app and Test Run

In the **Actions** pane, search for **close**. Drag **Close window** and drop it after the last action.

Select **Windows 'Request Funds Draw'** for Window and then click **Select**.

Click **Save** on the Close window step.

Click **Save** and wait for the flow to be saved.

Close the funding manager app if it is still running.

Click **Run**. Do not interact with the VM until the run completes.

The flow should run successfully. Review the output variables and make sure the **FundingStatus** is set to **Approved**.

Locate the **RiskScore** variable and double click on it.

Change the **Default value** to **85** and click **Save**.

Click **Run** again. Do not interact with the VM until the run completes.

The flow should run successfully. Review the output variables and make sure the **FundingStatus** is set to **Draw denied contact bank**.

Locate the **RiskScore** variable and double click on it.

Change the **Default value** to **30** and click **Save**.

Click **Save** and wait for the flow to be saved.

You may close the desktop flow.

Exercise 2: Automate Inspection web site

Task 1: Practice using site

In preparation for recording practice using the app to help eliminate accidental steps from being recorded.

Note: Before running this desktop flow from Power Automate Desktop, please ensure to enable **Power Automate Extension** in your Web browser.

Navigate to `https://fabrikaminspectionstest.azurewebsites.net/`

Navigate to the **Request Inspection** page.

Enter your name for Inspection Account#, enter `123 Main Street` for Property Address, enter `Test work item` for Work to Inspect, and then click **Request Inspection**.

Copy the **Job Number** and keep it in your clipboard.

Go to the **Inspection Results** page.

Enter your name for Inspection Account #, paste the Job number you copied, and click **Check Inspection**.

The Job Status should show **In progress**.

Go to the **Request Inspection** page and repeat the steps until you are comfortable with the inspection request process.

Task 2: Record using app

Create a new desktop flow in the **Construction Funding** solution and record the steps using the **Inspection** web app.

Navigate to `https://make.powerapps.com/` and make sure you have the Dev environment selected.

Select **Solutions** and open the **Construction Funding** solution.

Click + New and select Automation - Desktop flow.

Enter `Perform Site Inspection` for Flow name and click **Launch app**.

Power Automate Desktop flow designer should launch. You may need to click **Open** on the browser to permit the action.

Expand the **Browser automation** group and double click on the **Launch new Microsoft Edge** action.

Provide https://fabrikaminspectionstest.azurewebsites.net for Initial URL and click **Save**.

DO NOT start recording yet.

Minimize or close all but desktop flow application.

Click **Run**.

Microsoft Edge should load and navigate to the URL you provided.

DO NOT close this Browser session.

Go back to the desktop flow and click **Recorder**.

The recorder pane should come to view.

DO NOT start recording. Once you’ve reviewed the steps below, start recording. When you start recording you will repeat the steps you went through during the practice as outlined below.

i. Go to the **Request Inspection** page.

ii. Enter your name for **Inspection Account**

iii. Enter `123 Main Street` for Property Address.

iv. Enter `Test work item` for Work to Inspect.

v. Click on the **Request Inspection** button.

vi. Right click on the **Job Number** and select **Extract element value | Text: ('your job number here')**. Remember the Job number.

vii. Navigate to the Inspection Results page.

viii. Provide your name again.

ix. Enter the **Job number**.

x. Click on the **Check Inspection** button.

xi. Right click on the **Job status** and select **Extract element value | Text: ('In progress')**.

Note: Depending on your speed, the Job Status may already be **Completed**. In this case select corresponding option **Extract element value | Text: ('In progress')**.

xii. Go back to the recorder and click **Done**.

Click **Record** and perform the steps.

After you finish recording, your recorder actions should look like the image below.

Click **Save** and wait for the flow to be saved.

Task 3: Add input and output variables

In this task, you will define the input and output variables that you'll use to replace the hard coded values recorded. The input variables will be used to pass data from the calling cloud flow. The output variables will be used to return data to that flow.

Open the **Variables** pane, click the plus button **(+)** and select **Input**.

Enter `InspectionAccountNumber` for Variable name, select **Text** for Data type, `Jane Doe` for Default value, `InspectionAccountNumber` for External name, `Inspection account number` for Description, and click **Save**.

Click **(+)** and select **Input** again.

Enter `PropertyAddress` for Variable name, select **Text** for Data type, `123 Main Street` for Default value, `PropertyAddress` for External name, `Property address` for Description, and click **Save**.

Click **(+)** and select **Input** one more time.

Enter `WorkToInspect` for Variable name, select **Text** for Data type, `Test work item` for Default value, `WorkToInspect` for External name, `Work to inspect` for Description, and click **Save**.

Click **(+)** and select **Output**.

Enter `InspectionStatus` for Variable name, select **Text** for Data type, `InspectionStatus` for External name, `Inspection status` for Description, and click **Save**.

Click **(+)** and select **Output** again.

Enter `SitePhotos` for Variable name, select **Text** for Data type, `SitePhotos` for External name, `Site photos` for Description, and click **Save**.

Click **(+)** and select **Output** one more time.

Enter `JobNumber` for Variable name, select **Text** for Data type, `JobNumber` for External name, `Job number` for Description, and click **Save**.

You should now have three input and three output variables.

Task 4: Change to use variables

In this task, you will modify hard coded values in the steps to use the variables you just defined.

Go to the Populate text field on web page for the account number and double click on it.

Clear the Text value and click on the **{x}** select variable button.

Select **InspectionAccountNumber** and click **Select**.

Click **Save**.

Go to the Populate text field on web page for the property address and double click on it.

Clear the Text value and click on the **{x}** select variable button.

Select **PropertyAddress** and click **Select**.

Click **Save**.

Go to the Populate text field on web page for the work to inspect and double click on it.

Clear the Text value and click on the **{x}** select variable button.

Select **WorkToInspect** and click **Select**.

Click **Save**.

Go to the first Get details of element on web page and double click on it.

Click on the variable name, change it to **JobNumber**, and click **Save**. You are typing in this field.

Go to the Populate text field on web page for the second account number and double click on it.

Clear the Text value and click on the **{x}** select variable button.

Select **InspectionAccountNumber** and click **Select**.

Click **Save**.

Go to the last Populate text field on web page and double click on it.

Clear the Text value and click on the **{x}** select variable button.

Select **JobNumber** and click **Select**.

Click **Save**.

Go to the last Get details of element on web page and double click on it.

Click on variable and then click on the **{x}** select variable button.

Select **InspectionStatus**.

Click **Save** and close the Fabrikam Inspection website.

If your Launch new Microsoft Edge is not the first action in the flow, then drag the Launch new Microsoft Edge action and drop it before all the recorded actions.

Click **Run** and wait for the run to complete.

You should get an error similar to the one below. You see this error because the job number will be different for each run, but the flow is trying to match it with job number generated during the recording.

Go the **UI elements** tab and double click on the **Table data cell 'xxxxx'** UI.

Select the last element form the **Elements** list **Table data cell 'xxxxx'**, uncheck the **Text** attribute, check the **Ordinal** attribute, change the value or the Ordinal from 0 to **1**, and click **Save**. We are telling the flow to use the value in the second cell of the table, the table cells are zero based where 0 is the first cell and 1 is the second cell.

We will do the same thing for the inspection status. Double click on the **Table data cell 'In progress'** UI.

Note: Depending on the recording previously, this might be **Table data cell 'Completed'**.

Select the **Table row 'Job Status: In progress** element, check the **Ordinal** attribute checkbox, enter 1 for the Ordinal value, and then select **Table data cell 'In progress'**. You are selectecting the row first and then moving to edit the cell.

Uncheck the **Text** attribute, check the **Ordinal** attribute, change the value or the Ordinal from 0 to **1**, and click **Save**.

Run the flow again.

The flow run should now succeed, and the variables should look like the image below.

Click **Save** and wait for the flow to be saved.

Task 5: Add loop condition and wait control

In this task, you will be adding a loop to recheck if the inspection is done. You will add a delay to give the inspection time to be completed before checking again.

Expand the **Loops** group, drag **Loop condition** and drop it before Populate text field on a web page for the second InspectionAccountNumber action.

Click on the **{x}** select variable button of the First operand.

Select **InspectionStatus** and click **Select**.

Select **Not equal to (<>)** for Operation, enter **Completed** for the Second operand, and click **Save**.

Select the four actions after the loop and move them inside the loop by drag and drop.

Expand the **Flow control** group, drag **Wait** action and drop it before the **End** loop.

Enter **10** seconds and click **Save**.

Click **Run** and wait for the flow to complete. The flow should go through the loop until the inspection status changes to **Completed**.

Do not close the browser.

Click **Save** and wait for the flow to be saved.

Task 6: Use JavaScript to build JSON

In this task, you will use JavaScript to extract the work site inspection photos and format the data into a JSON array that can be used to populate the output variable.

Go to the **Actions** pane and expand the **Browser automation** group.

Expand the **Web data extraction** subgroup and double click on the **Extract data from web page** action.

Go back to the web page, right click on the image, and select **Extract entire HTML table**.

Notice you only get Value #1, currently, extract is not able to extract both columns of the table because column 2 is an img tag.

Click **Cancel**.

Click **Cancel** again.

Go to the web page and open the DevTools via the **F12** key. We are going to use Dev Tools to test our JavaScript.

Select the **Console** tab and click **Clear console**.

Paste the script below and press enter.

```js var table = document.getElementById("sitephotostable");

var sitephotolist = { images:[] }

for (var i = 0; i < table.rows.length; i++) {

row = table.rows[i];

namecol = row.cells[0];

imgcol = row.cells[1];
```

```js
var imgtags = imgcol.getElementsByTagName('img');

var imgsource = imgtags[0]['src'];

console.log(imgsource)

sitephotolist.images.push({"name":namecol.innerText, "url":imgsource})

}

console.log(JSON.stringify(sitephotolist))
```

Review the information you are trying to extract.

Close the DevTools.

Go back to the desktop flow.

Expand the **Browser automation** group, drag **Run JavaScript function on a web page and get the returned result** action and drop it after the **End** loop.

Select **%Browser%** for Web browser instance, paste the script below in the JavaScript function field, expand the **Variables produced** section, click on the **{x}** select variable button.

```js
function ExecuteScript()

{

var table = document.getElementById("sitephotostable");

var sitephotolist = { images:[] }

for (var i = 0; i < table.rows.length; i++) {

row = table.rows[i];

namecol = row.cells[0];

imgcol = row.cells[1];

var imgtags = imgcol.getElementsByTagName('img');

var imgsource = imgtags[0]['src'];

console.log(imgsource)

sitephotolist.images.push({"name":namecol.innerText, "url":imgsource})

}

console.log(JSON.stringify(sitephotolist))

return JSON.stringify(sitephotolist);
```

}```

Select **SitePhotos**.

Click **Save**.

Expand the **Browser automation** group, drag **Close web browser** action, and drop it to the bottom of the recorded actions.

```
![drag the item as described](media/ab371f1d6af8cee5cfedac5e556511e5.png)
```

Select **%Browser%** for web browser instance and click **Save**.

Click **Save** and wait for the flow to be saved.

Task 7: Close browser and test

Close the Fabrikam Inspection website.

Click **Run** and wait for the run to complete. The flow will loop through the loop condition multiple times, until the status changes to completed.

Go to the Variables pane and make sure the **SitePhotos** variable has the expected value.

Click **Save** and wait for the flow to be saved.