

The background features abstract, overlapping green geometric shapes, primarily triangles and polygons, in various shades of green, creating a modern and dynamic visual effect.

ROBOTIC PROCESS AUTOMATION LABORATORY 21CSL75


Build a workflow using Web Recorder in UiPath Studio to Sign in to UiPath's website.

- Ask the user's email address and password.**
- Open the sign in page of UiPath's Website.**
- Sign in to the website using the user's credentials.**

Activities

- ▶ **Input Dialog**
- ▶ **Open Browser**
- ▶ **Web Recorder**
- ▶ **Click**
- ▶ **Type**

```
libraryOptions : {  
  "includeOriginalXaml": false,  
  "privateWorkflows": []  
},  
"processOptions": {  
  "ignoredFiles": []  
},  
"fileInfoCollection": [],  
"modernBehavior": false,  
"saveToCloud": false  
},  
"expressionLanguage": "VisualBasic",  
"entryPoints": [  
  {  
    "filePath": "Main.xaml",  
    "uniqueId": "1affd13b-864e-47b5-95b0-4987b9a07565",  
    "input": [],  
    "output": []  
  }  
]
```

 Input Dialog

Dialog Title

{ }

"EMAIL-ID"

Input Label

{ }

"ENTER UR MAIL ID"


Input Type

Text Box

Value entered

{ }

EmailAddress

 Input Dialog

Dialog Title

{ }

"PASSWORD"

Input Label

{ }

"ENTER UR PASSWORD"

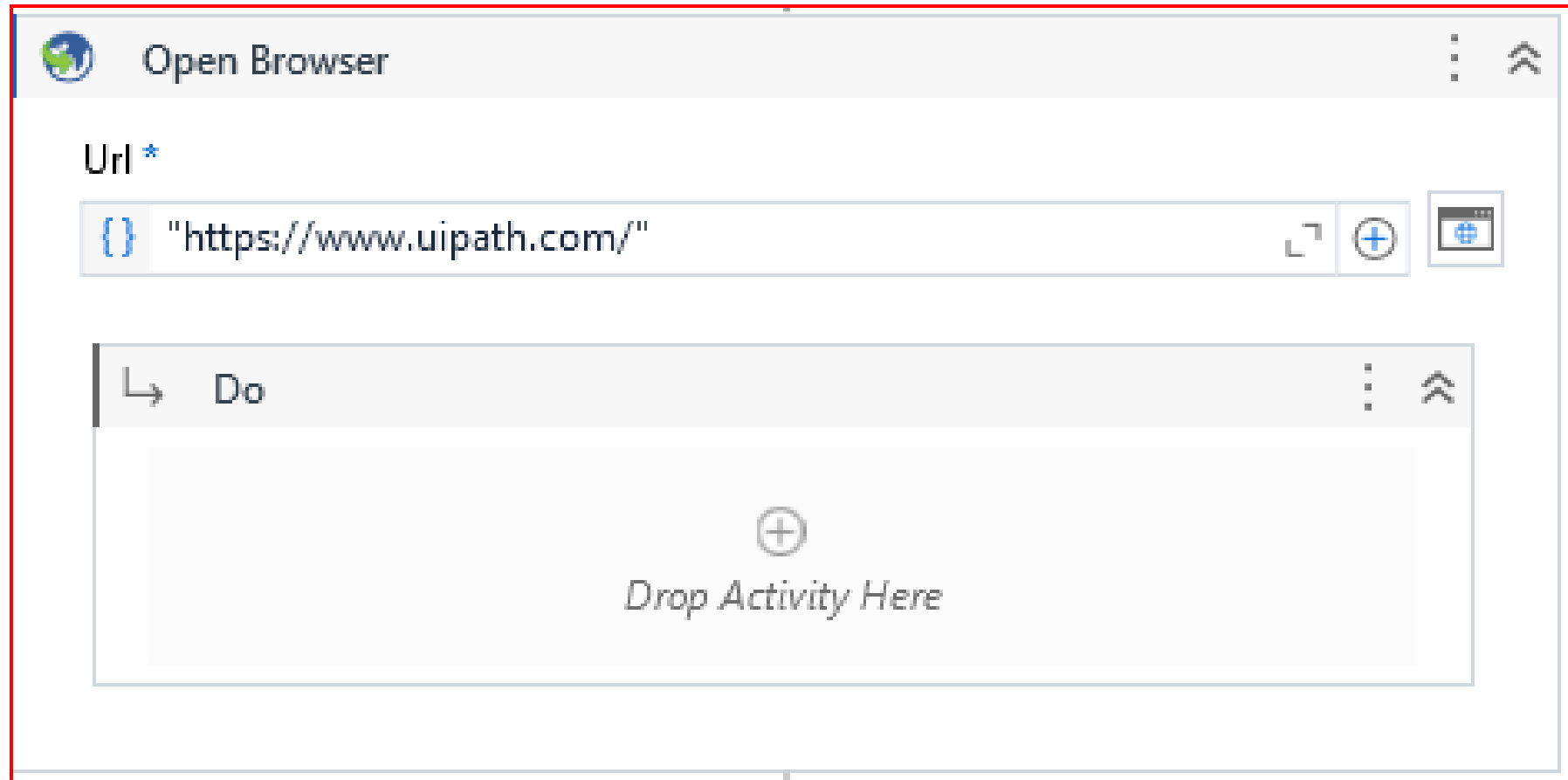
Input Type

Text Box

Value entered

{ }

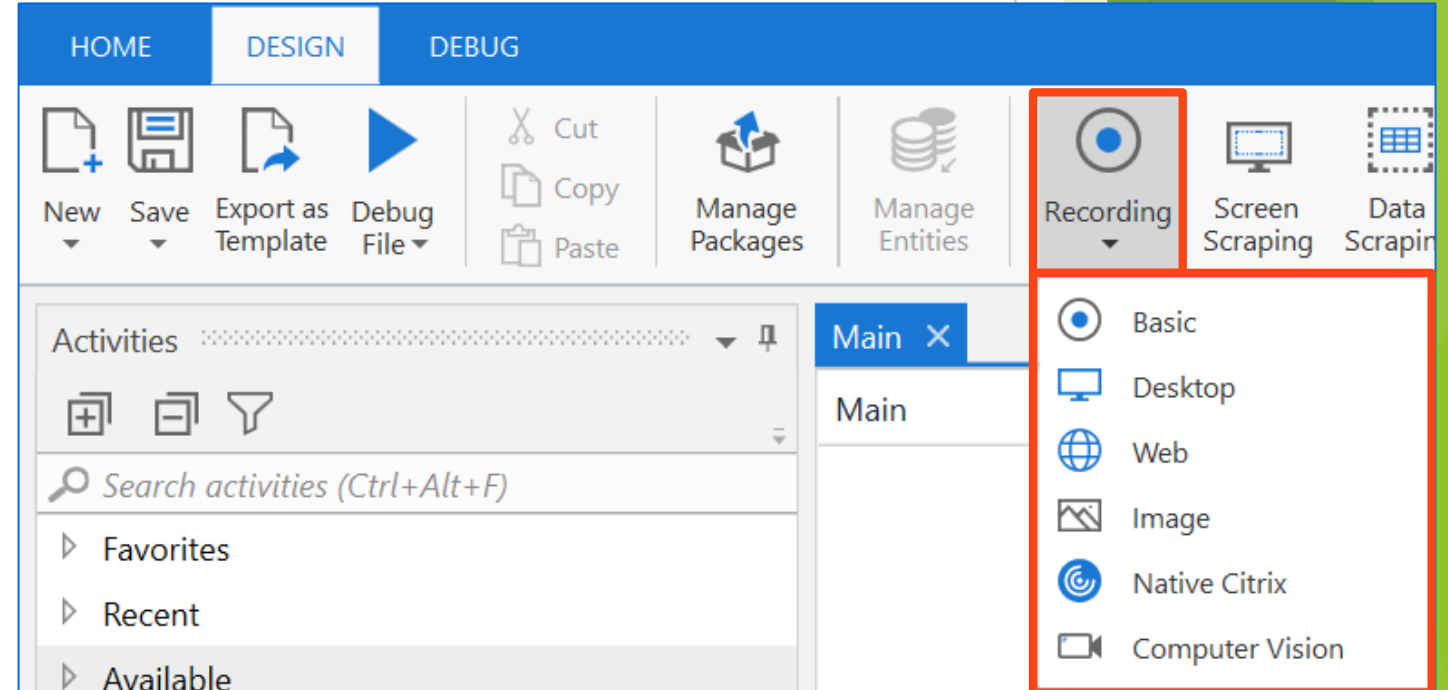
Password



TASK Recoder

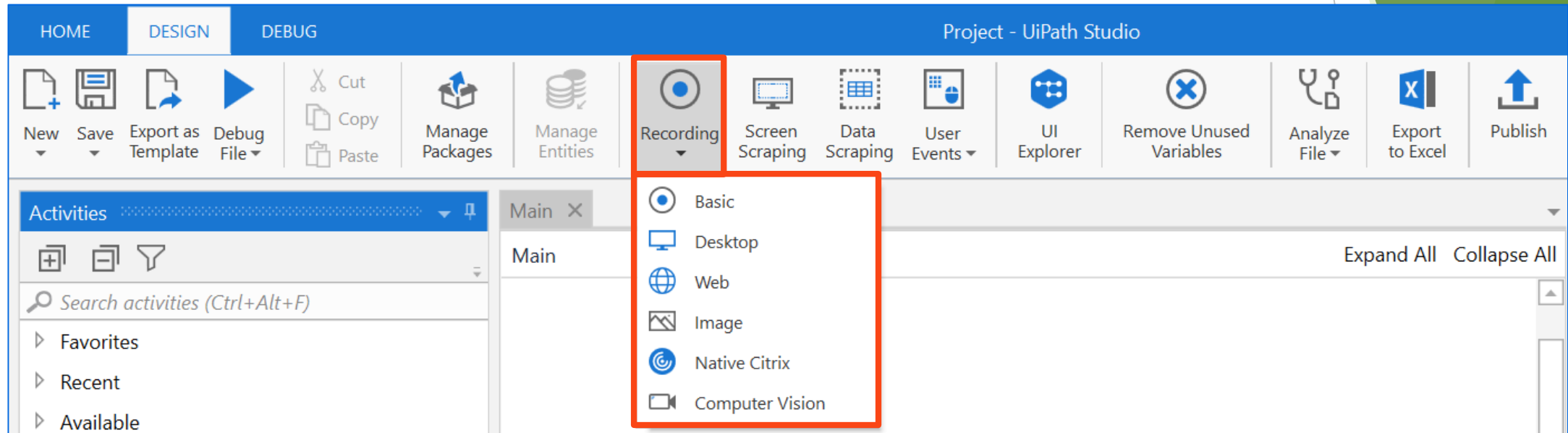
The recording functionality of UiPath Studio helps in capturing user's actions on the screen and translating them into sequences.

- The recording tool can be accessed from the 'Design' tab in UiPath Studio.
- While recording, all user interface elements are highlighted, allowing easy identification of buttons, fields, menus, or elements with which the user interacts.
- Once the recording ends, a sequence is created, containing the activities performed by the user.



Types of Recording

UiPath Studio consists of six recorders that come with their own controllers to perform recording.

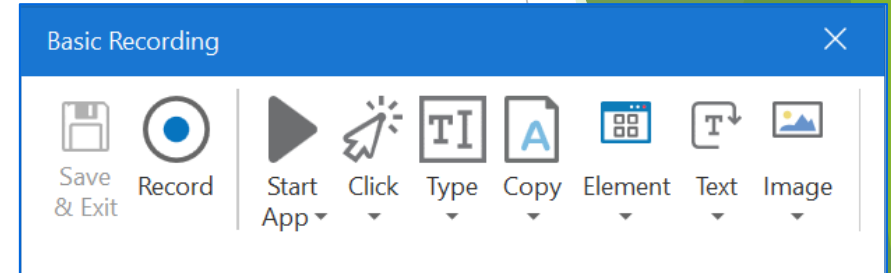


- Basic Recording
- Desktop Recording
- Web Recording
- Image Recording
- Native Citrix Recording
- Computer Vision Recording

Basic, Desktop & Web Recording

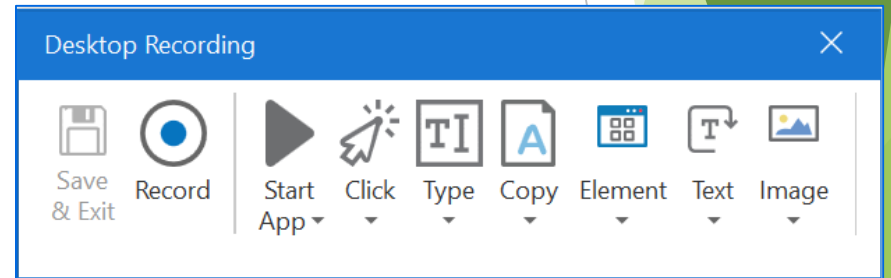
Basic Recording

Generates a full selector for each activity and no container.



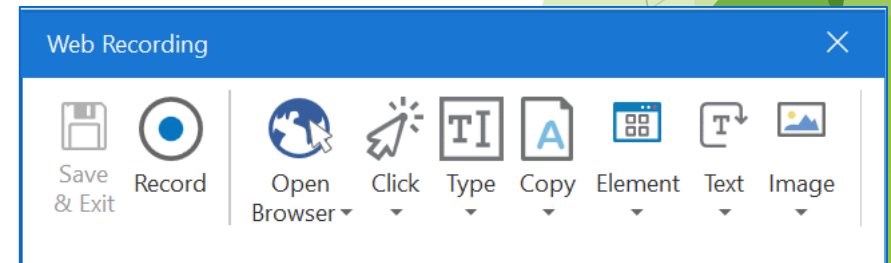
Desktop Recording

Suitable for all types of desktop apps and multiple actions.



Web Recording


Used for recording in web apps and browsers



- START
- Use two **Input Dialog** activities and take the email address and password from the user.
- Store received input in two variables.
- Use an **Open Browser** activity and open URL – “www.uipath.com”. s
- Use the Web Recorder in UiPath Studio to:

- Click the *Sign in* link on the next page.
- Type in the user's email address and password.
- Click on the *Sign in* button.

- STOP

The background features abstract, overlapping green geometric shapes, primarily triangles and polygons, in various shades of green, creating a modern, layered effect.

Mail : rpalab2024@gmail.com
Pwd : Rpa@bitm2024

Build a workflow using Format, Join, IndexOf, Split, and Substring methods that extract key information from a text and prints in a different format.

- Use the text "You always wanted to study Automation Training. The materials are available in the following : UiPath Blog, UiPath Academy." for extraction.
- Extract "Automation Training" from the first sentence. - Extract "UiPath Blog" and "UiPath Academy" from the second sentence.
- Display "study Automation Training from: UiPath Blog; UiPath Academy" in a message box.

Activities used

- ▶ Sequence
- ▶ Assign
- ▶ String Methods

Name	Variable type	Scope	Default
Message	String	Sequence - String Manipulation	
Study	String	Sequence - String Manipulation	
places	List<String>	Sequence - String Manipulation	

Places :
`System.Collections.Generic.List<System.String>`

To	Value
Message	"You always wanted to study Automation Training. The materials are available in the following places: UiPath Blog, UiPath Academy."

To	Value
Study	<code>message.Split("."c).First.ToString.Substring(message.LastIndexOf("study"))</code>

- `Split("."c).First.ToString` extracts the first sentence of the String and converts it to a String |

You always wanted to study Automation Training.

To	Value
Study	<code>message.Split("."c).First.ToString.Substring(message.LastIndexOf("study"))</code>

2. `Substring(message.LastIndexOf("study"))` extracts the Substring from "study"

study Automation Training.

To	Value
places	<code>message.Split("."c)(1).ToString.Split(": "c).Last.ToString.Split(", "c).ToList</code>

“1. `message.Split("."c)(1).ToString` extracts the second sentence of the String and converts it to a String.

The materials are available in the following places:
UiPath Blog, UiPath Academy.

To	Value
places	<code>message.Split("."c)(1).ToString.Split(": "c).Last.ToString.Split(", "c).ToList</code>

2. `Split(": "c).Last.ToString` splits the remaining String and keeps only the last part of it.

UiPath Blog, UiPath Academy.

To	Value
places	<code>message.Split("."c)(1).ToString.Split(": "c).Last.ToString.Split(", "c).ToList</code>

3. `Split(", "c).ToList` takes each string separated by a comma and adds it as an element in the List variable”.

Places : UiPath Blog, UiPath Academy.

```
String.Format("{0} from: {1}", study, String.Join(";", places))
```

String.Format: This method formats a string using placeholders like {0}, {1}, etc. Each placeholder is replaced by the corresponding value passed as arguments.

{0} will be replaced by the value of study.

{1} will be replaced by the result of String.Join(";", places).

String.Join(";", places): This method takes an array or collection of strings, places, and joins them into a single string with each element separated by ;.

Build a workflow using Split and Contains methods that extract sentences containing “RPA” from a paragraph.

- Store a paragraph in a string variable using an Assign activity.
- Store all sentences from the text in an array using a Split method.
- Loop through each sentence and identify sentences containing “RPA” using Contains method.
- Store all identified sentences in an MS Word file.

ACTIVITIES USED

- ▶ Sequence
- ▶ Assign
- ▶ For each
- ▶ If
- ▶ Attach Window
- ▶ Type Into

- START
- Use an **Assign** activity to store a paragraph in a string variable called **newText**.
- Use **newText.Split(".")** and store all sentences in an array called **newSentence**
- Use a **For Each** activity and iterate through each item in the array **newSentence**
- Use an **If** activity within the For Each activity to identify sentence that contains the string "UiPath" in it. Use **item.Contains("RPA")** as condition.
- Use a **Type Into** activity to store result in an MS Word file.
- STOP

To	Value
newText	"Robotic process automation (RPA) is a software technology that makes it easy to build, deploy, and manage software robots that emulate humans actions interacting with digital systems and software. RPA streamlines workflows, which makes organizations more profitable, flexible, and responsive. RPA is noninvasive and can be rapidly implemented to accelerate digital transformation. It also increases employee satisfaction, engagement, and productivity by removing mundane tasks from their workdays. "

To	Value
newSentence	newText.Split(".",c)

For Each currentText

In *

newSentence

Body

If

Condition *

currentText.Contains("RPA")

Then

Attach Window 'notepad.exe Untitled'

Do

Type Into 'editable text'

Text *

currentText.ToString + "[k(enter)]"

The image shows a workflow editor interface. At the top is a loop block labeled 'For Each currentText'. Inside the loop, there is an 'In' block with a variable 'newSentence'. Below this is a 'Body' block containing an 'If' block. The 'If' block has a 'Condition' 'currentText.Contains("RPA")'. If the condition is true, the 'Then' block executes two actions: 'Attach Window 'notepad.exe Untitled'' and 'Do'. The 'Do' block contains a 'Type Into 'editable text'' action. The 'Text' property of this action is set to 'currentText.ToString + "[k(enter)]"'. The 'Type Into' action block is highlighted with a blue border. In the background, a Notepad window titled 'notepad.exe Untitled' is visible, showing a red header bar and a menu bar with options like File, Edit, Format, and View.

Build a workflow using Concat and Join method that merges two lists containing the UK and Spain city names, sorts it, capitalizes the first letter of each item, and displays it in a message box.

- Create a list containing three UK cities in all capital letters.
- Create another list containing three Spain cities in small letters.
- Merge both the lists together.
- Sort the final list in alphabetical order from A to Z.
- Capitalize only the first letter of all the items in the final list.
- Display the final list in a message box in string format.

Name	Variable type	Scope	Default
SpainCities	List<String>	Sequence List Demo	- new List (of String) from {"madrid", "valencia", "barcelona"}
UKCities	List<String>	Sequence List Demo	- new List (of String) from {"MANCHESTER", "BRISTOL", "GLASGOW"}
AllCities	List<String>	Sequence List Demo	-
AllCitiesProperCase	List<String>	Sequence List Demo	- new List(of String)

(x) Assign

Save to

{ } AllCities



=

Value to save

{ } SpainCities.Concat(US

L¹



Fix

Use Variables



```
1 SpainCities.Concat(USCities).ToList()
```

Invoke Method

TargetType (null) ▼

TargetObject {} AllCities L¹ ⊕

MethodName Sort

For Each currentText

In *

{}

AllCities

↳

+

Body

✕

Assign

Save to

{}

currentText

+

=

Value to save

{}

System.Char.ToUpper(

↳

+

↓

+

+

Append Items to Collection

Collection *

{}

AllCitiesProperCase

↳

+

Items *

1 item in Collection

↳

Assign > Set value (InArgument)

Fix Use Variables ▾

```
1 System.Char.ToUpper(currentText(0)) & currentText.Substring(1).ToLower()
```

Message Box

Text *

{ }

String.Join(", ", AllCitiesProperCase)

L⁷

+

Build a workflow using data table activities to join two library databases using matching student ID and display the output in a message box.

- Create a data table variable and populate it with student ID and name of students.
- Create another data table variable, and populate it with student ID and book names
- Join both the data tables based on matching student ID.
- Remove the student ID column and sort the final data table as per student names in alphabetical order from A to Z.
- Display the final data table containing the student and book names in a message box as a string.


Activities

- ▶ Build Data Table
- ▶ Join Data Table
- ▶ Remove Data Column
- ▶ Sort Data Table
- ▶ Output Data Table
- ▶ Message Box

4.2 Process Overview

- START
- Use two **Build Data Table** activities to create two tables. Store them in two DataTable variables called **dt_users** and **dt_overdueBooks**.
 - **dt_users** variable contain ID of students and name of user as string.
 - **dt_overdueBooks** variable contain ID of students and name of books as string.
- Use a **Join Data Table** activity. Choose the Inner type for the Join activity. Write the two column names to be used as Join criterion and create a new data table variable to store the output called **dt_borrowedBooks**
- Use a **Remove Data Column** activity to delete duplicate column – student ID – by specifying its index.
- Use a **Sort Data Table** activity to sort the data table based on the name of students in alphabetical order from A to Z.
- Use an **Output Data Table** activity to print the content of the data table to a String variable.
- Use the **Message Box** activity to display the output.
- STOP

+	Student ID (String)	×	Student Name (String)	×
×	101		John	
×	102		June	
×	103		Jerry	
×	104		Jason	
×	105		Jasmine	
×	106		Dhanya	
×				

 Build Data Table

+

Student ID
(String)

✕

✕

102

✕

104

✕

106

✕

✕

Book Name
(String)

Life is What You Make it

Believe in Yourself

The Alchemist

OK

Cancel

OKCancel

Join Wizard

Input DataTable 1
dt_users

Input DataTable 2
dt_overd

Join Type
Inner

Output DataTable
dt_borrowed

Column Table 1	Operation	Column Table 2
0	=	0

OK

Cancel

Student Id	Student Name	Student Id	Book Name
102	June	102	Life is what u make it
104	Jason	104	Believe in Yourself
106	Dhanya	106	The Alchemist

Remove Data Column

Column Name *

{ }

The name of the column to be removed fro

L⁷

+

Column Number *

{ }

2

L⁷

+

From Data Table *

{ }

dt_borrowed

L⁷

+

Sort Data Table

Data Table *

{ }

dt_borrowed

⌵

+

Common

DisplayNameSort Data Table

Input

Data Table

dt_borrow⌵

+

Misc

Private☐

Output

Data Table

dt_output

+

Sorting Column

Column

The DataC⌵

+

Index

1⌵

+

Name

The name⌵

+

Order

Ascending

Output Data Table as Text

Data Table *

{}

dt_output

JiPath.Core.Activities.OutputDataTable

Common

DisplayNameOutput Data Table as

Input

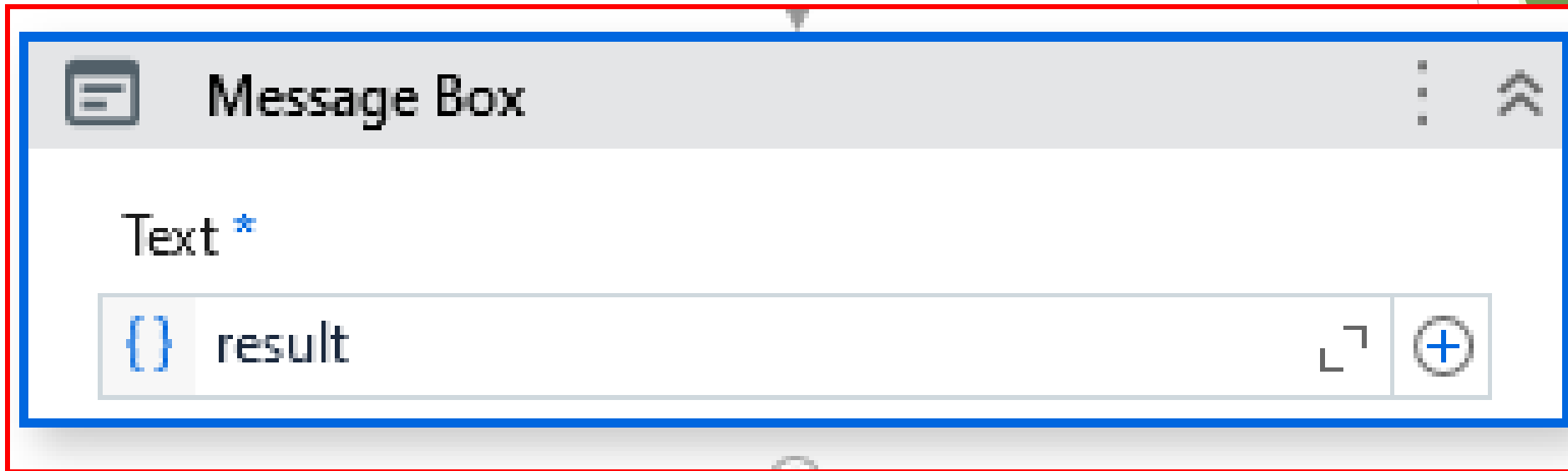
Data TableThe DataTa

Misc

Private☐

Output

Textresult



7. Build a workflow using the Data Scraping wizard that scrapes blog post titles from the UiPath Blog from multiple pages.

- Open the UiPath Blog (<https://www.uipath.com/blog>).
- Extract all blog titles and URL by navigating through all pages.
- Store scraped data in an Excel file

Activities used for

- ▶ Open Browser
- ▶ Data Scrapping
- ▶ Write CSV file

- ▶ START
- ▶ Use an Open Browser activity and enter the URL - www.uipath.com/blog.
- ▶ Use the Data Scraping button in the Design ribbon to indicate the first two blog post titles
- ▶ Rename “Column1” to “Blog Titles”.
- ▶ Use the Indicate Next Link window of the Data Scraping tool to indicate the Next link in the search results page.
- ▶ Use Write CSV activity and save the data in a CSV file.
- ▶ STOP

Open Browser

Url *

{}

"https://www.uipath.com/blog"

Do

+

Drop activity here or [generate with Autopilot](#)

Properties

UiPath.Core.Activities.OpenBrowser

Common

ContinueOnError

Specifies to continue e

+

■

...

DisplayName

Open Browser

Input

BrowserType

BrowserType.Chrome

▼

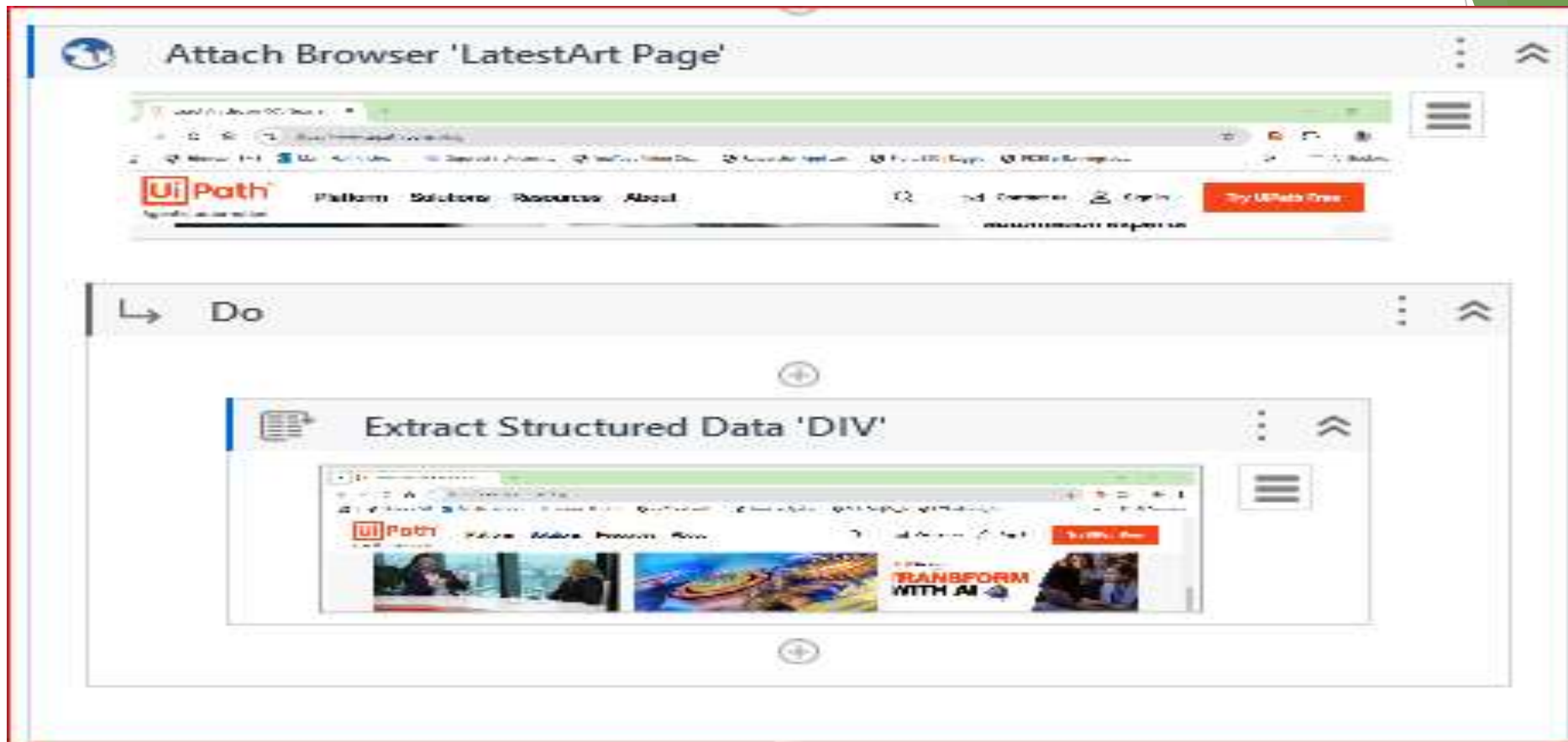
...

Url

"https://www.uipath.com"

⌵

+



Name	Variable type	Scope	Default
ExtractDataTable	DataTable	Main Sequence	New System.Data.DataTable



Write CSV



Write from *



ExtractDataTable



Write to what file *



"C:\Users\DELL\Desktop\RPAEX.csv"



How to write

Write



Include headers

6. Build a workflow using a Screen Scraper Wizard that scrapes text using the Tesseract OCR scraping method from an image and stores it in a Notepad.
- Search for “text images” in Google Images.
 - Pick one image containing text from the search results.
 - Scrape the text from the image using Tesseract OCR.
 - Store text in a Notepad file

- START
- Use an **Open Browser** activity and enter the URL www.google.com/images.
- Use a **Type Into** activity in the Open Browser activity and search “text images” in the URL.
- Use Screen Scraping from the Design ribbon to select an image containing text.
- Choose ‘Tesseract OCR’ as the Screen Scraping method. Change Scale to 5.
- Use **Write Text File** activity to store content in a Notepad file.
- STOP

Activities used

- ▶ Sequence
- ▶ Open Browser
- ▶ Type into
- ▶ Send HotKey
- ▶ Screen scrapping
- ▶ Write text to file

Url *

{ } "www.google.com/images"

Do

Type Into 'TEXTAREA APjFqb'

Text *

{ } "text images"

Send Hotkey

Indicate element inside browser

Alt Ctrl Shift Win Key
☐ ☐ ☐ ☐ enter

Properties

JiPath.Core.Activities.OpenBrowser

Common

ContinueOnError Specifies to continue € + [] ...

DisplayName Open Browser

Input

BrowserType BrowserType.Chrome [] ...

Url "www.google.com/image" [] +

Misc

Private ☐

Options

AutomaticallyDownloadWebDriver Automatically downlo + [] ...

CommunicationMethod Choose the communicat [] ...

Hidden Open a hidden brows + [] ...

NewSession Starts a new session o + [] ...

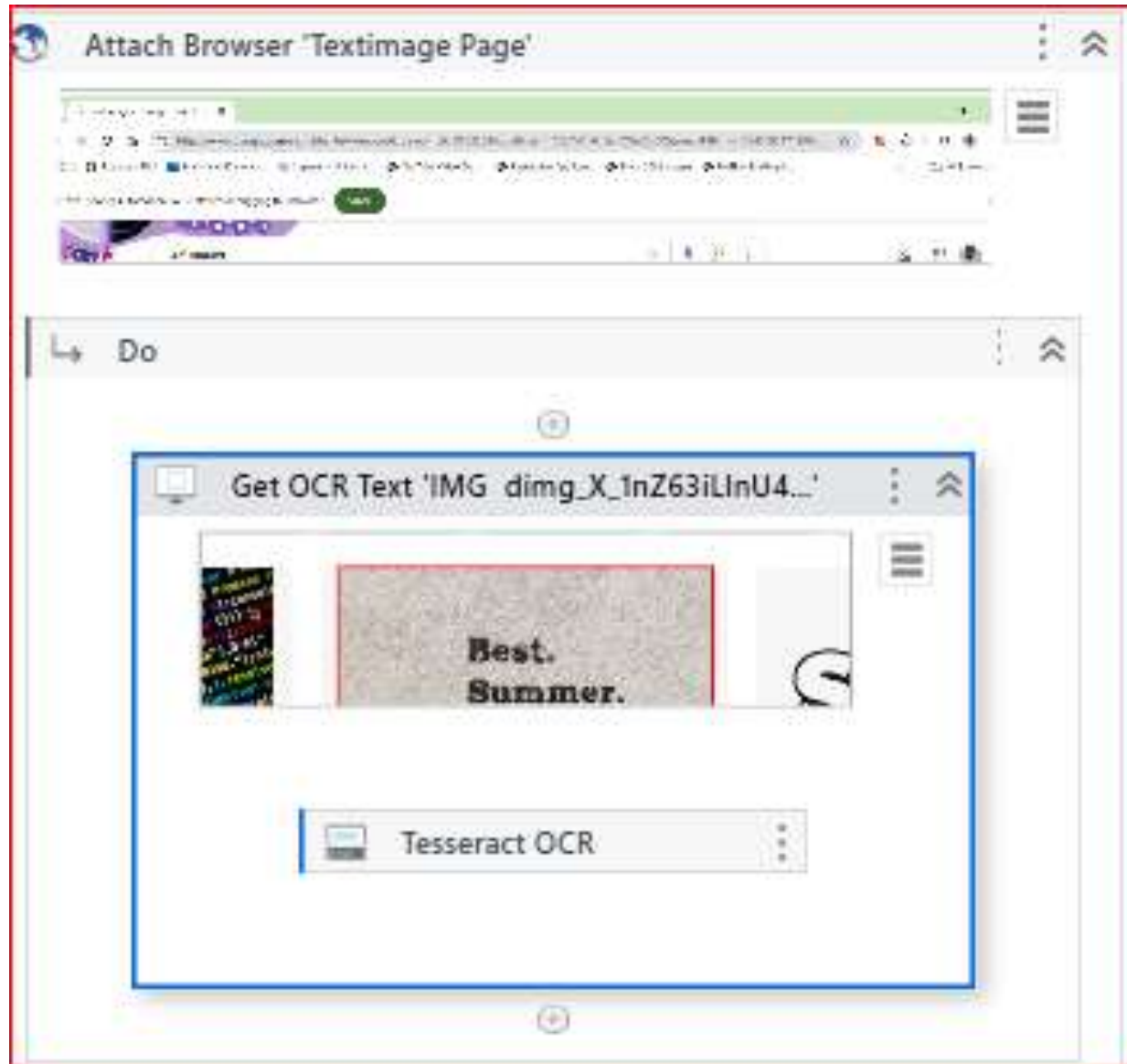
Private Open a private/incogi + [] ...

UserDataFolderMode The UserDataFolderMode [] ...

UserDataFolderPath The user data folder that [] +

Output

UiBrowser The result of the activity as a +



UiPath.Core.Activities.GetOCRText

Common

ContinueOnError	Specifies to continue <input type="checkbox"/>
DisplayName	Get OCR Text 'IMG_dimg_X_1nZ63iLInU4...'

Input

Target	Target
ClippingRegion	
Element	Enter a VB expression
Selector	"<webctrl aaname='Dow..."
Timeout (milliseconds)	Enter a VB expression
WaitForReady	WaitForReady.COMPLETE

Misc

Private	<input type="checkbox"/>
---------	--------------------------

Output

Text	ImgDimgXNzilineu
WordsInfo	ImgDimgXNzilineu1

Write Text File

Text *

{ }

ImgDimgRpxnzxjMm

Write to filename *

{ }

"C:\Users\DELL\Desktop\vpas.txt"

Build a workflow using a Read PDF Text activity and extract only Email IDs and Phone Numbers from a PDF file and store it in an MS Word file.

- Read data from the PDF file using a Read PDF Text activity.
- Extract only Phone Numbers and email IDs from the PDF and store it in an MS Word file

Activities used

- ▶ Sequence
- ▶ Read PDF Text
- ▶ Find Matching Patterns- Emails
- ▶ Find Matching Patterns-Phone NO
- ▶ For Each-Emails
- ▶ For Each-Phone No
- ▶ Send HotKey

Read PDF Text

File Name *

{}

"E:\Courseware\Courseware\Practice_L"

+

Properties

UiPath.PDF.Activities.ReadPDFText

Common

DisplayName

Read PDF Text

File

FileName

"E:\Coursew"

+

Password

The passwoi

+

Input

PreserveFormatting

A flag used

+

Range

"All"

+

Misc

Private

☐

Output

Text

PdfOutput

+



Configure Regular Expression...

RegEx Builder

est Text

RegEx

Value

Quantifiers

Email



((?>[a-zA-Z\d!#\$%&'*\+-\V=?^_`{|}~]+\x20*)"((?=[\x01-\x7f])[^"\\\

Exactly

1

+

△

7

[illegible]

Full Expression

(?>[a-zA-Z\d!#\$%&'*\+-\|=/?^_`{|}~]+\x20*)(((?=([x01-\x7f])([^\n\r\x01-\x7f])*(?

Regex Option

☒ IgnoreCase

Save

Cancel

Properties

JiPath.Core.Activities.Matches

Common

DisplayName

Find Matching Patter

Input

Pattern

"((?>[a-zA-Z_]+)

Pattern options

IgnoreCase, Com

Text to search in

PdfOutput

 L⁷


Timeout (ms)

The maximality \perp^+ 

Misc

Private

Result

emails

Output

First Match

The first match is \oplus

Configure Regular Expression...

RegEx Builder

Test Text

RegEx Value Quantifiers

Advanced ⓘ (407)([0-9])* Exactly 1 + Δ ▽ ×

Full Expression

(407)([0-9])*

Regex Option

☒ IgnoreCase

Save Cancel

Properties

JiPath.Core.Activities.Matches

Common

DisplayName

Find Matching Patterns

Input

Pattern

"(407)([0-9])*" L¹ +

Pattern options

IgnoreCase, Comp L¹ ▾

Text to search in

PdfOutput L¹ +

Timeout (ms)

The maximum L¹ +

Misc

Private

☐

Result

phoneno L¹ +

Output

First Match

The first match th L¹ +

For Each currentMatch

In *
{ } emails

Body

TI Type Into 'edit Page 1 content'

Text *
{ } currentMatch.ToString

Send Hotkey

Indicate on screen

Alt

Ctrl

Shift

Win

Key

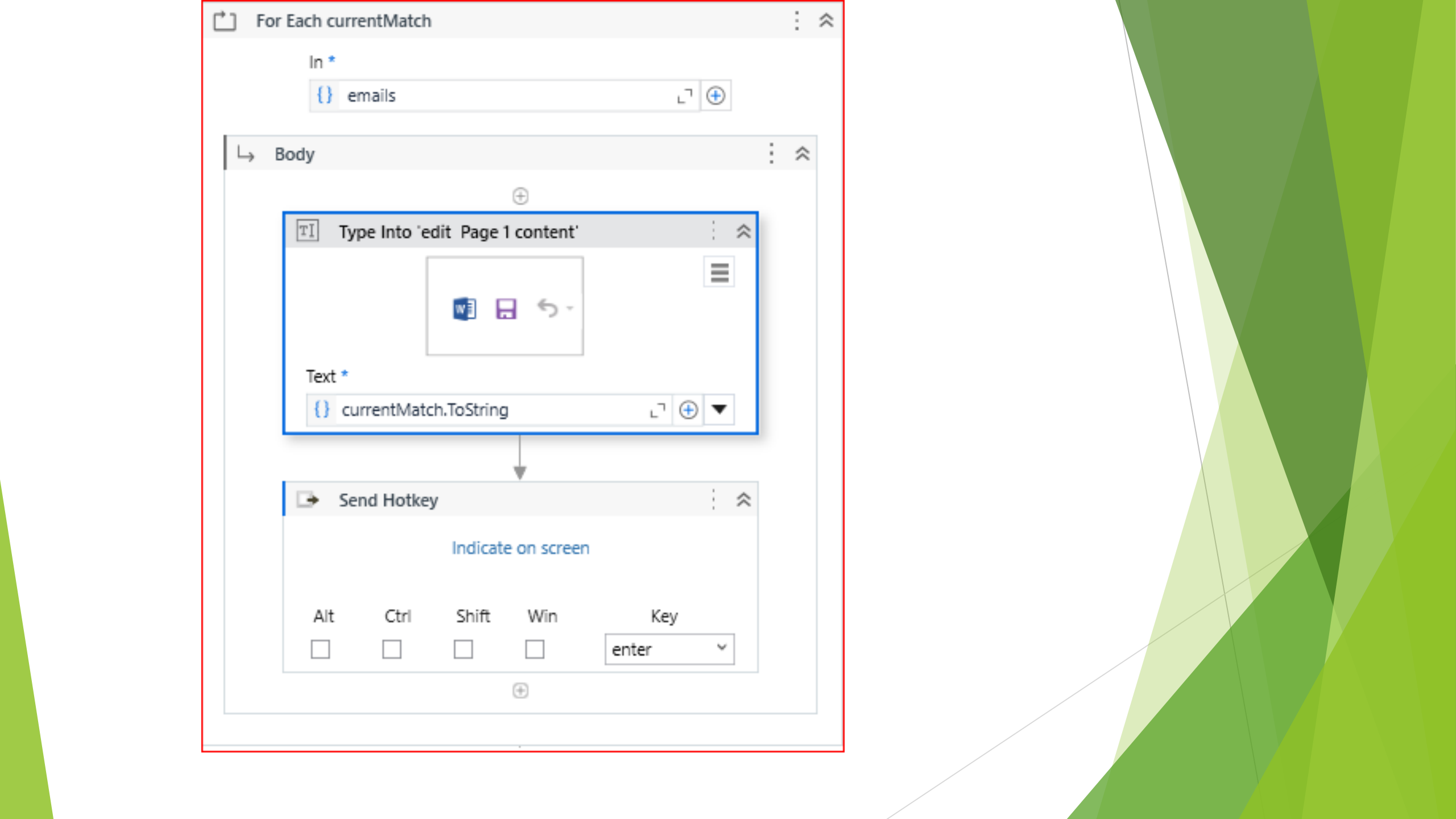
☐

☐

☐

☐

enter



For Each currentMatch

In *

{}

phoneno

Body

TI

Type Into 'edit Page 1 content'

W

D

↶

Text *

{}

currentMatch.ToString

Send Hotkey

Indicate on screen

Alt

Ctrl

Shift

Win

Key

☐

☐

☐

☐

enter

Build a workflow to display file names from a folder in the Output panel and also store names in an MS Word file.

- Locate and select a folder containing multiple files.
- List the directory path of all the files in the Output panel.
- Also, store the updated names in an MS Word file and save and close it.

Activties

- ▶ Sequence
- ▶ Browse for Folder
- ▶ Assign
- ▶ Attach Window---(Select Ms word)
- ▶ For Each
- ▶ WriteLine
- ▶ TypeInto
- ▶ Send HotKey
- ▶ Click

- START
- Use a **Select Folder** activity to select a folder containing a few files.
- Use an **Assign** activity to store file names in an array.
- Use an **Attach Window** activity below the **Assign** activity and select MS Word window.
- Use a **For Each** activity to iterate through each file name in the array.
- Use a **Write Line** activity within the **For Each** activity to display file names in the Output panel.
- Use a **Type Into** activity below the Write Line activity to store file names in an MS Word file.
- Use **Click** and **Send Hotkey** activities to save and close the file.
- STOP

Name	Variable type	Scope	Default
FileList	String[]	Main Sequence	<i>Enter a VB expression</i>
FolderName	String	Main Sequence	<i>Enter a VB expression</i>



Browse For Folder



Selected folder path



FolderName



(x) Assign

Save to

Value to save

() FileList

+

=

() Directory.GetFiles(Folc

↵

+

↶

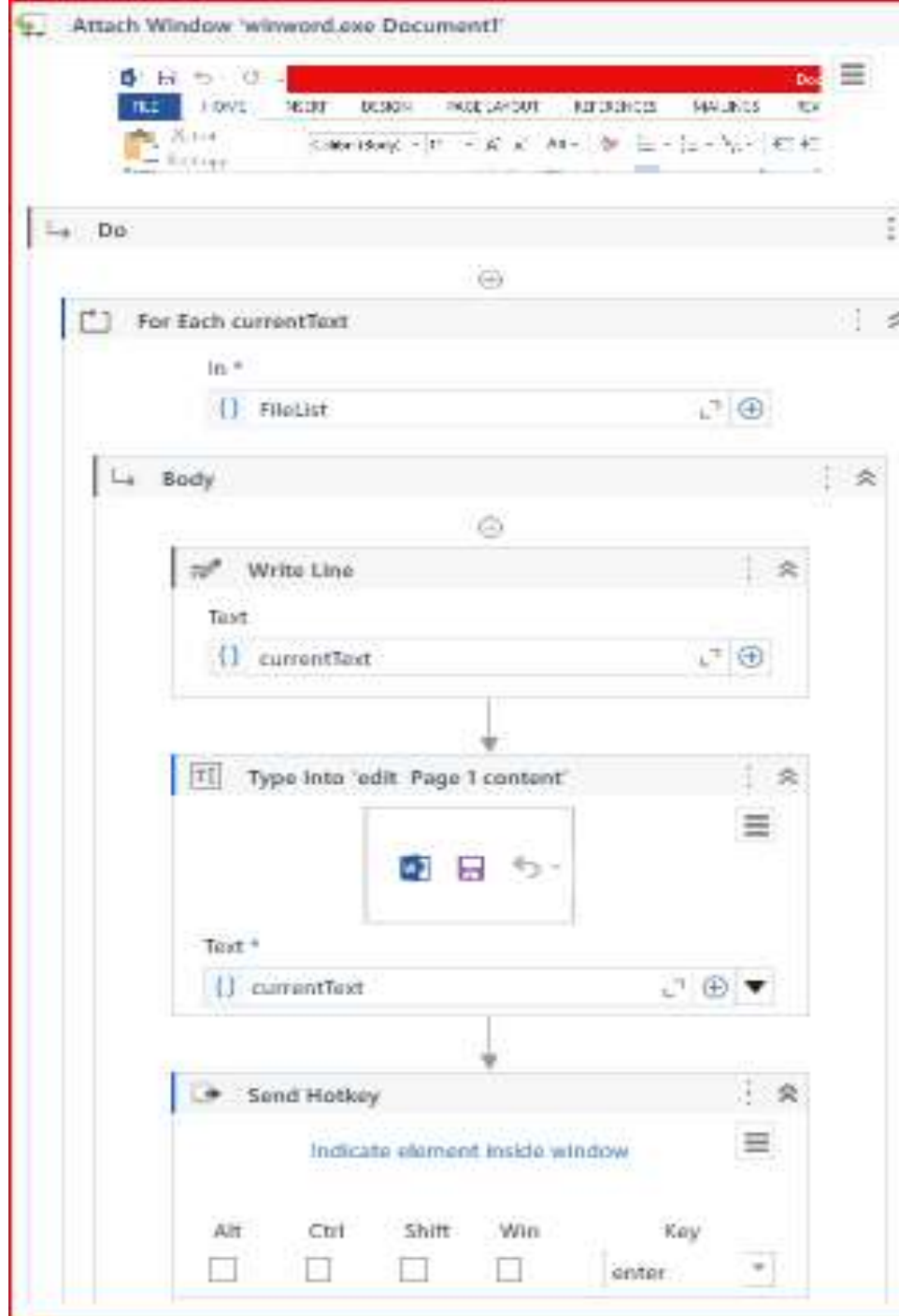
↷

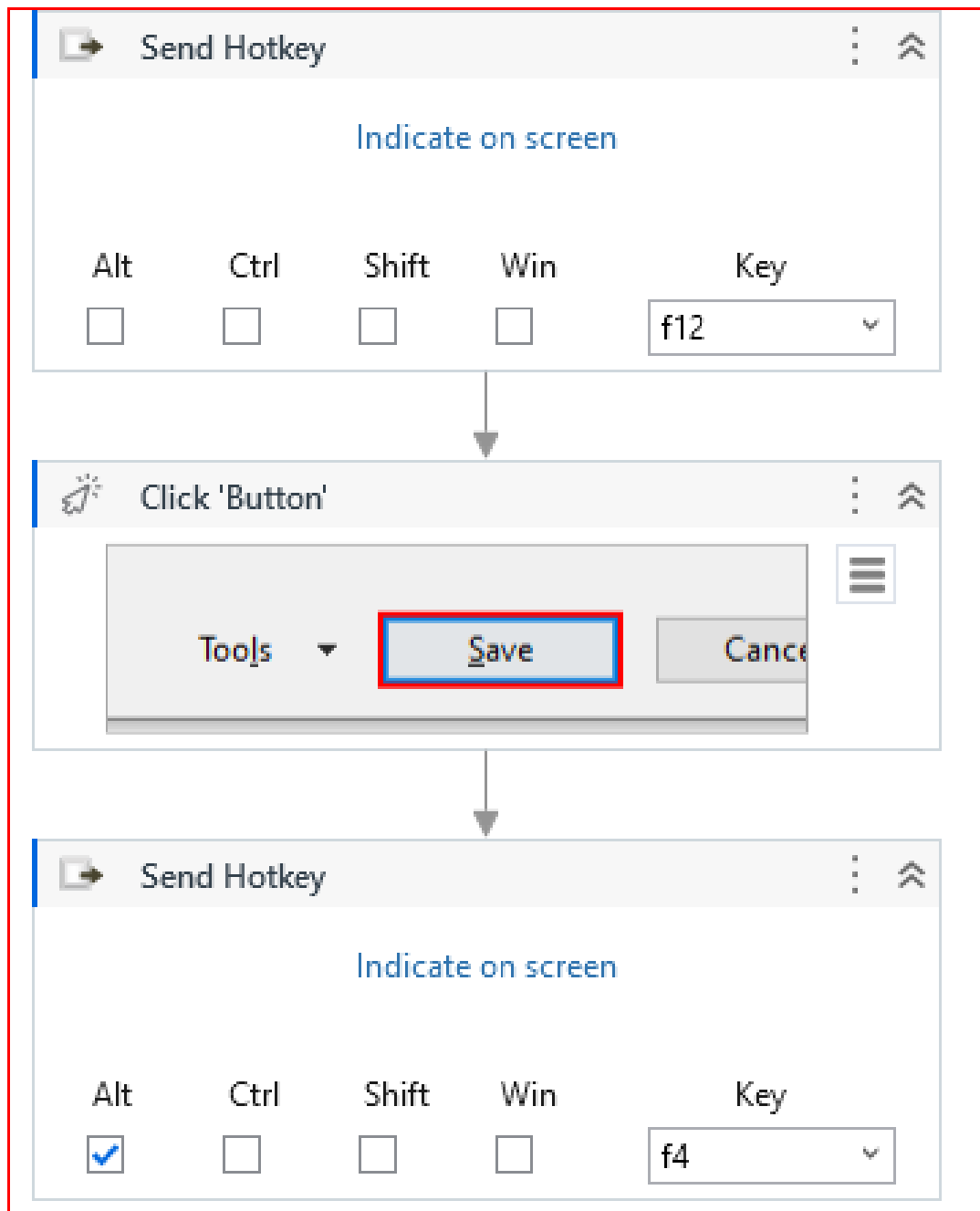
✎ Fix

Use Variables ▾

1

Directory.GetFiles(FolderName)






Build a workflow using a **Try Catch** activity to do the following:

- Take Name, Gender, and Age as the user input.
- Subtract current year with Age value to get the Year of Birth.
- Handle an error that occurs due to a reckless user input of an incorrect age containing the 11-digit number.
- Continue the process to display the Name, Gender, and Year of Birth of the user in a message box.

- START
- Use three **Input Dialog** activities within the **Try Catch** activity to ask for the Name, Gender, and Age of the user.
- Use an **Assign** activity to subtract the age from the current year to get the year of birth of the user
- Use Exception Type: System.Exception in the Catches section of the **Try Catch** activity to handle reckless input from the user. Store error in a string variable.
- Use a **Message Box** activity to display the Name, Gender, and Year of Birth of the user along with the Error, if any.
- STOP

Name	Variable type	Scope	Default
userName	String	Sequence - 'A workflow using Try Catch activity to catch certain errors'	
userGender	String	Sequence - 'A workflow using Try Catch activity to catch certain errors'	
intUserAge	Int32	Sequence - 'A workflow using Try Catch activity to catch certain errors'	
intYearOfBirth	Int32	Sequence - 'A workflow using Try Catch activity to catch certain errors'	
inputError	String	Sequence - 'A workflow using Try Catch activity to catch certain errors'	

 Input Dialog

Dialog Title

{ }

"NAME"

L⁷

+

Input Label

{ }

"ENTER UR NAME"

L⁷

+

Input Type

Text Box

∨

Value entered

{ }

userName

+

Input Dialog

Dialog Title

{}

"AGE"

↵

+

Input Label

{}

"ENTER UR AGE"

↵

+

Input Type

Text Box


▼

Value entered

{}

intUserAge

+

 Input Dialog ⋮ ⌵

Dialog Title

{ } "GENDER" ↵ +

Input Label

{ } "ENTER UR GENDER" ↵ +

Input Type

Text Box ⌵

Value entered

{ } userGender +

(x) Assign

Save to

Value to save

{}

intYearOfBirth

+

=

{}

date.Today.Year()-intU

⌵

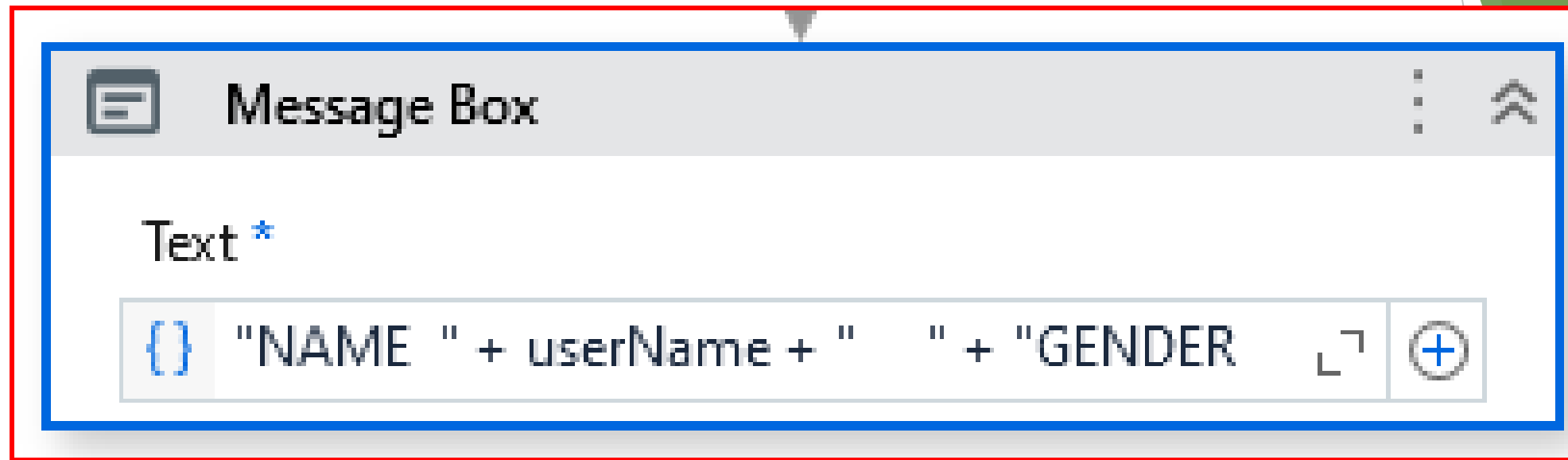
+

↶ ↷

✎ Fix

Use Variables ▾

1 date.Today.Year()-intUserAge



Message Box > Content (InArgument<Object>)

The text to be displayed in the message box.

  |  Fix | Use Variables 

```
1 | "NAME " + userName + " " + "GENDER " + userGender + " " + "YEAR OF BIRTH " + intYearOfBirth.ToString()
```


(x) Assign

Save to

{ } inputError



=

Value to save

{ } exception.Message



Message Box

Text *

{ } inputError

