

UNIT-IV

Advanced Automation Concepts and Techniques: Recording and Advanced UI Interaction, Recording Introduction, Basic and Desktop Recording, Web Recording, Input/output Methods, Screen Scraping, Data Scraping, Scraping advanced techniques, Selectors, Selectors, Defining and Assessing Selectors, Customization, Debugging,

Dynamic Selectors, Partial Selectors, RPA Challenge, Image, Text & Advanced Citrix Automation, Introduction to Image & Text Automation, Image-based automation, Keyboard based automation, Information Retrieval, Advanced Citrix Automation challenges, Best Practices, Using tab for Images, Starting Apps, Excel Data Tables & PDF, Data Tables in RPA, Excel and Data Table basics, Data Manipulation in excel, Extracting Data from PDF, Extracting a single piece of data, Anchors, Using anchors in PDF.

Advanced Automation Concepts and Techniques**Recording and Advanced UI Interaction****Advanced UI Interaction**

Advanced UI interactions are input and output interactions. In other words, it refers to the types of input methods and output techniques that are used while automating.

Input methods

The input that we give in the form of text can be of three types:

1. Default
2. Simulate
3. Window message

-Default is the generated method, while the other two are available in the Properties panel..

-The other two methods work in the background. Out of these three methods, the simulate type is the fastest method and is mostly preferred because in the window message input type, it types only the lowercase characters.

Output methods

These are the methods we use for getting our output, which can be in the form of text or images.

The available methods are

- a. Native
- b. Full-text
- c. OCR

Native is, by default, the generated method to extract data from the window. When you indicate any element, the scraping window appears, and here all of the options can be found.

In OCR, there are two types of *OCR engines*: One is *Google OCR* and the other is *Microsoft OCR*. We can choose whichever displays better results. Also, we can adjust the scale mentioned in the properties of the OCR.

Recording Introduction

The task recorder is the main reason for RPA's success. With the task recorder, we can create a basic framework for automation. The user's actions on the screen are recorded by the recorder and turned into a recording sequence in the current project. That's how Robots are able to mimic human actions. The recording is a collection of execution steps that have to be taken, on the applications in the scope, in order to accomplish a task.

These steps can be recorded one by one (manually) by pointing it on the screen or many steps in a go that is, automatically.

There are four types of recording in UiPath Studio:

1. Basic
2. Desktop
3. Web
4. Citrix

1. Basic recorder: A basic recorder is used to record activities on the desktop. This type of recorder is used for single activities and simple workflows. The action here is self-contained and not contained in separate windows.

2. Desktop recorder: The desktop recorder, like the basic recorder, is used to record activities on the desktop. However, it is used to record and automate multiple actions and complex workflows. Each activity here is contained in an Attach Window component

-The Attach Window component is especially important to ensure that other windows of the same application do not interfere in the workflow.

3. Web recorder: The web recorder, as the name suggests, is used to record actions on web applications and browsers.

4. Citrix recorder: Citrix is used to record virtual machines, VNC, and Citrix environments. This recording allows only keyboard, text, and image automation.

Some actions are recordable while others are not:

1. **Recordable actions:** Left-click on buttons, check boxes, drop-down lists, and other GUI elements. Text typing is also recordable.
2. **Actions that cannot be recorded:** Keyboard shortcuts, mouse hover, right-click. Modifier keys such as Ctrl and Alt cannot be recorded.

There are two types of recording:

1. **Automatic recording:** This is for recording multiple actions in one go. This is a very good feature for preparing a solid foundation for automating a task. It can be invoked with the Record icon available in basic, desktop, and web recorders.

Examples: hotkeys, right-click, double-click, and a few more.

2. **Manual recording:** This type of recording is used to record each step one at a time and hence offers more control over the recording.

Also, it can record all actions that cannot be recorded using automatic recording such as keyboard shortcuts, mouse hover, right-click, modifier keys, such as Ctrl and Alt, finding text from apps, and many other activities.

Note: Citrix recorder can only record a single action (manual recording).

Shortcut keys:

F2 key: This pauses the recording for 3 seconds. The countdown menu is also shown on the screen.

Right-click: Exits the recording.

Esc key: Exits the recording. If one presses the Esc key again, then the recording will be saved.

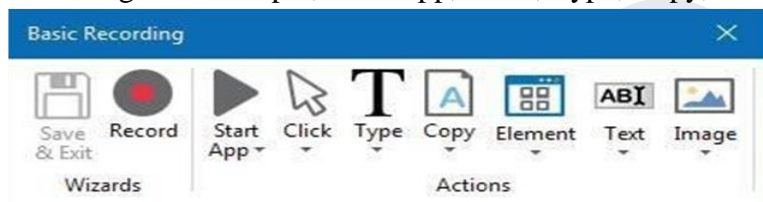
Recording

The functions of these recordings. The operations that can be completed with the help of recording are as follows:

- Click (clicking a UI element: button, image, or icon)
- Type (typing any value into the available text field)
- Copy and paste

Basic Recorder

Recording. For example; Start App, Click, Type, Copy, and so on.



1. **Start App:** This is used to start an application. When we left-click on this option, we are asked to point to an application that we want to open. When we are done, we can click on the Save & Exit option.
 2. **Click:** Another option is Click, which is used to click on a UI element. This feature is used as a mouse input. That is, it is used for clicking, checking, or selecting an item. When we click on this option, we are asked to indicate the location of the UI element we want to click. We can change the type of click to right-click or double-click in the Click Type property from the Properties panel.
 3. **Type:** Another option shown in the recording panel is Type. As the name suggests, it is used for typing something inside the indicated element. All you need to do is to indicate the area where you want to type. Then, you need to type your input in the popup that appears for typing
 4. After you are done typing, do not forget to press the Enter key. When the Enter key is pressed, the step is recorded.
 5. You can then click on Save & Exit to view the recording sequence.
 6. The recording sequence is shown in the following screenshot. You can change the text you have written (by changing the value of the Type in the block). You can write the desired text in double quotes (“ ”), or you can simply use a variable to store the data
- There are UI three more options in the recording panel:
- a. Element
 - b. Text
 - c. Image

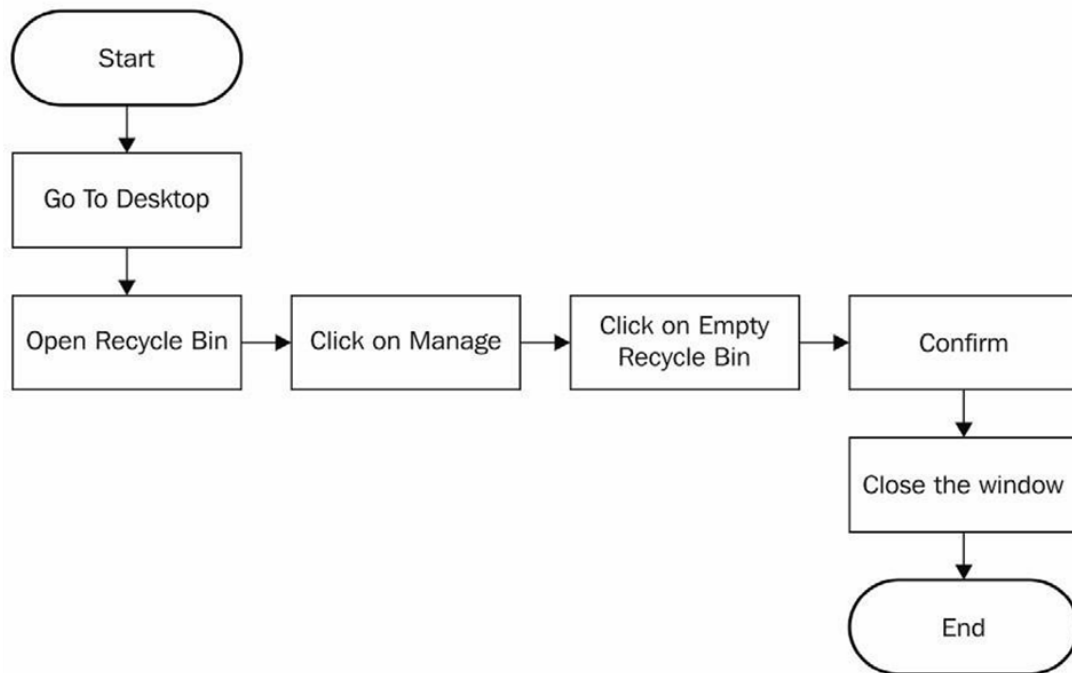
Let us illustrate two examples of using the UiPath recorder for Web-based Applications:

1. Emptying the trash folder in Gmail (web-based application)
2. Emptying Recycle Bin (Windows-based application)

The first one is to show a recording of a web-based application, and the second is Windows-based.

Desktop Recording

We are going to automate emptying the Recycle Bin. There are various steps that are involved. Let's map the process of how to empty the Recycle Bin:



This diagram is simple and more detailed than in the Emptying trash in Gmail example; we need to do exactly the same steps in order to perform this task.

1. Open UiPath Studio and choose a blank project.
2. Since we are working in the recorder, and since we are working on the desktop and not a web application, we are required to choose the desktop recorder.

Start the recorder and simply perform the following steps:

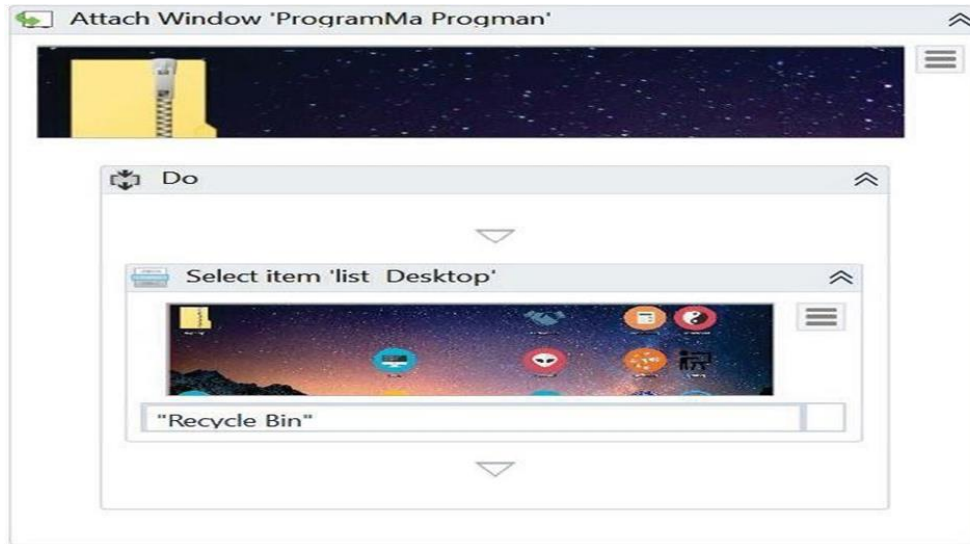
1. Go to the desktop by pressing the Windows + D keys.
2. Open Recycle Bin by clicking on Recycle Bin and then pressing Enter key.
3. Click on the Manage tab of the Recycle Bin folder.
4. Click on the Empty Recycle Bin button.
5. Confirm by clicking on the Yes button in the dialog box.
6. Close the Recycle Bin folder by pressing the cross button.
7. Press the Esc key and Save and exit the recorder.

Now your recording is ready to view, let's examine each step recorded:

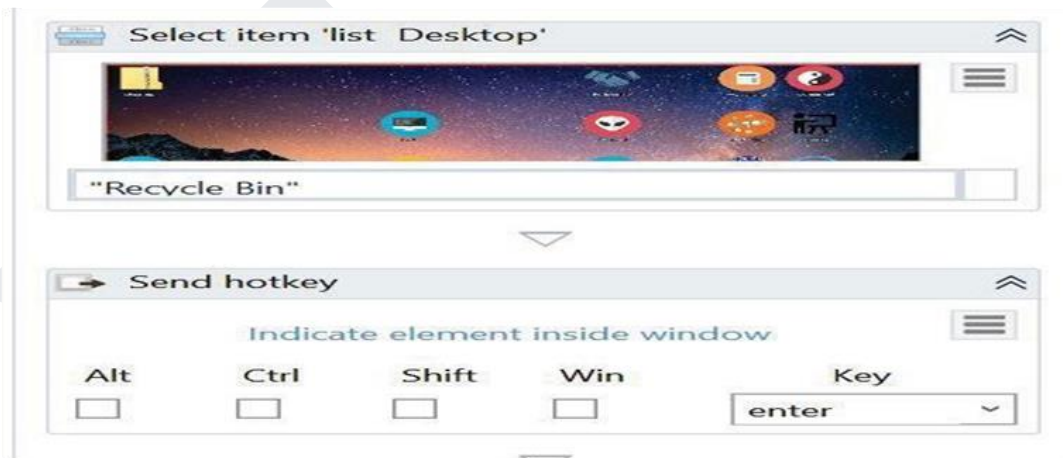
1. Go to the desktop by pressing the Windows + D keys: This step is not recorded! Never

mind, it is not needed. Please note that the recorded steps attach themselves to an application, and execute commands for that application, so the next step (Open Recycle Bin) will be executed on the desktop whether you are there or not.

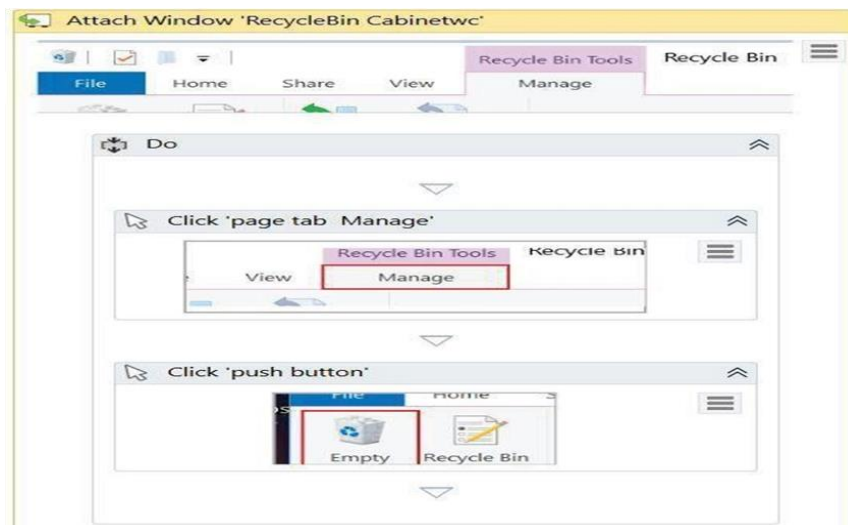
2. Open Recycle Bin by clicking on Recycle Bin and then pressing the Enter key - We can see the recorded step in the following screenshot:



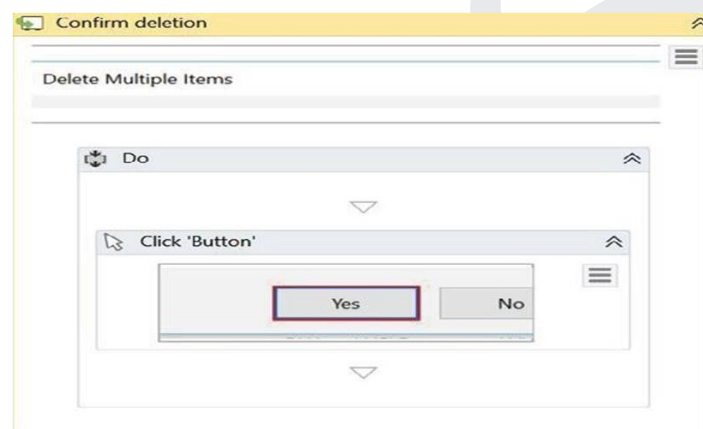
- a. selecting the Recycle Bin is recorded, not the Enter key. We should manually add that step. Search for Send hot key in the Activities window and insert it into the workflow just below the Select item 'list Desktop' step, as shown in the following screenshot:



- b. Click on the Manage tab of the Recycle Bin folder: This is recorded as it is and so is the fourth step, click on the Empty Recycle Bin button:



3. Confirming by clicking on the Yes button on the dialog box is also recorded. Similarly:

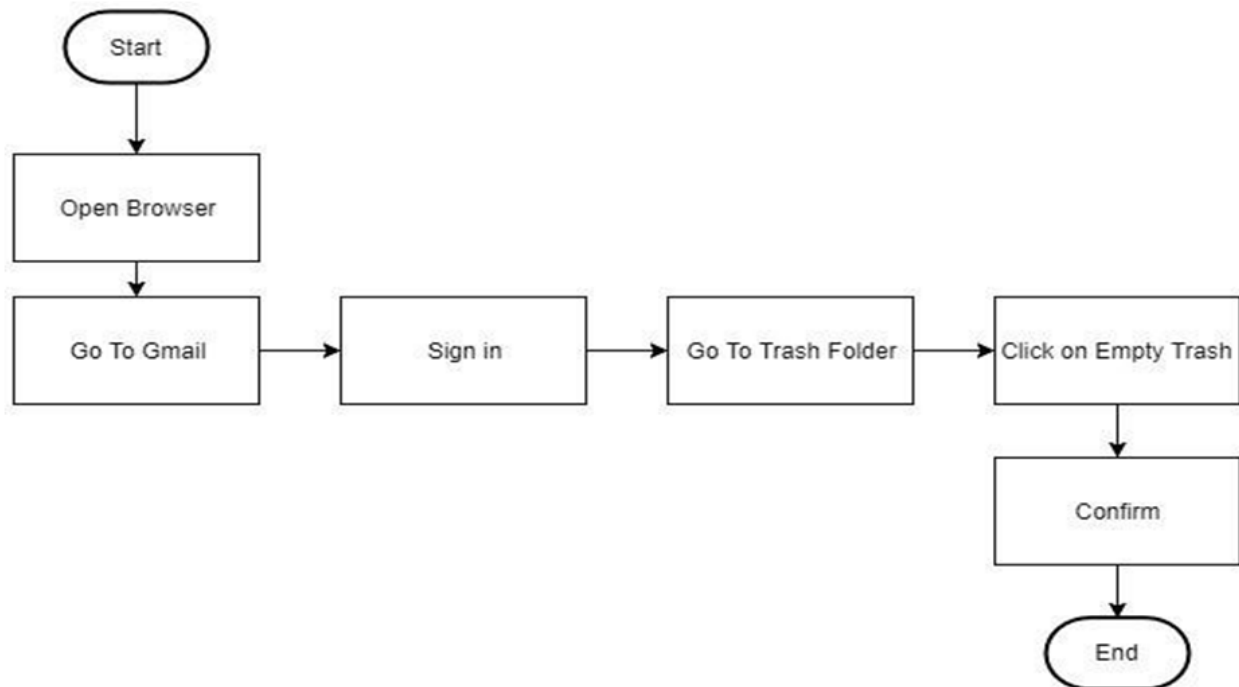


In the last step, closing the Recycle Bin folder by pressing the cross button, you may have to indicate an anchor. Save it and press F5 to run it. You see how easy it is to record steps taken on a computer and automate them.

Web Recording

Emptying Trash in Gmail

1. This is an example of how we can empty a folder in Gmail with the help of a UiPath Robot, solely on the basis of recording.
2. To do this, we are going to record all the actions that have to be performed to empty this Trash folder so that our Robot understands the sequence to be performed.
3. The process flow of this simple activity is in the following diagram:



Process flow for emptying Gmail trash

4. First and foremost, we begin with a blank project in UiPath Studio and then choose Web recorder from the Recording drop-down list:



5. To click on the Recording option and select the type of recording. We will use Web recording for this process since we are working on a website.

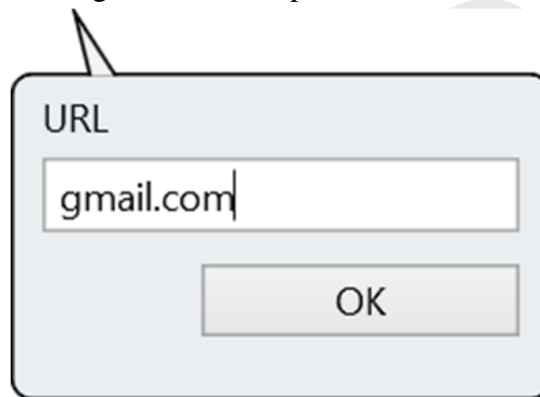
6. Just click on the Recording icon at the top of the page. From the four types of recording that appear, choose Web recording. A Web Recording panel will appear, as shown in the

above screenshot.

7. Notice Open Browser between Record and Click; this is available with web recorder to record steps in browser-based applications.
8. Preparation: Open your favourite browser, navigate to <https://gmail.com>, and keep this browser open.

The following are the six steps in our process flow:

1. **Open Browser:** Although we have already opened Gmail in the browser, we did not record that step. Here, we will note that step in the recorder using the Open Browser button in the recorder. A drop-down menu will appear. Again, choose Open Browser from the drop-down menu. It will ask to highlight the browser, highlight the already opened browser and click on the top of the browser.
2. **Go to gmail.com:** You will be prompted to enter the URL of the website to navigate to. Type <https://gmail.com> or gmail.com and press OK:



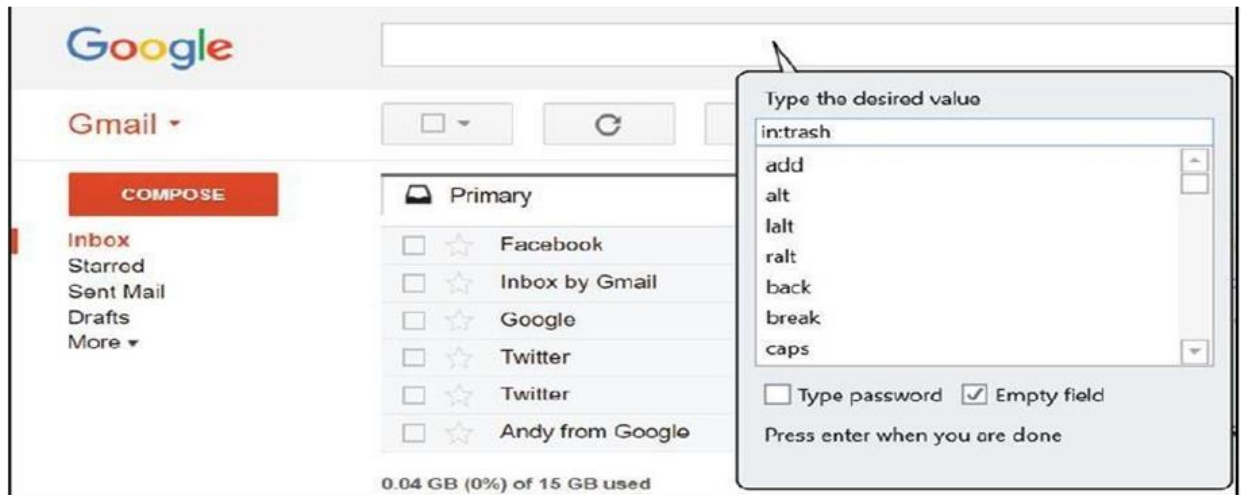
3. **Sign In:** Start recording by clicking on the Record icon of the recording panel. Go to the already open Gmail and click on the Email or Phone field. UiPath will pop up a prompt for typing the email:



- Type Email in the box provided by the UiPath recorder and press *Enter*. The Gmail textbox will automatically fill up with your typed content. Click on the **NEXT** button of the Gmail interface; it will also get recorded.
- Now, you have recorded an entry in the password field. For

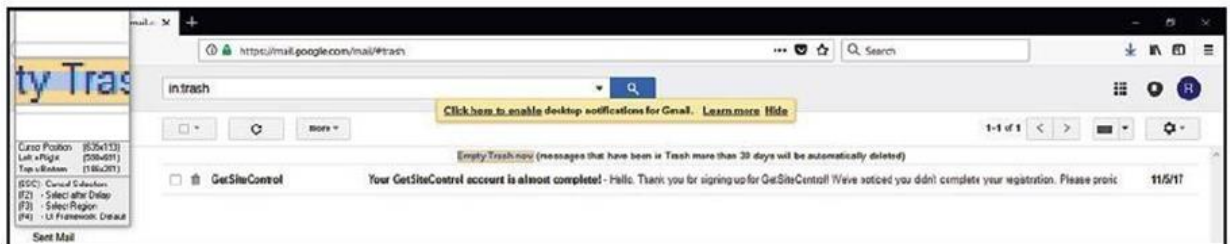
simplicity, you may the password in the prompt provided by UiPath. Type your password in the text field of the popup that appears.

- Then, click **NEXT** to log in to your account. Clicking on the **NEXT** button will also get recorded.
4. **Locate Trash Folder:** In this step, we have to click on the search box of Gmail and type in trash in the UiPath prompt, and hit Enter:



Now, click on the Search button beside the search box. It will also get recorded automatically and the Trash folder will appear.

5. Click on Empty Trash now: Once you are done with clicking on the Trash action, You can see a link showing Empty Trash now. Hover the mouse on this link and it will get highlighted, Click on it to delete all the messages in the Trash folder:



6. Confirm: When you click on Empty Trash now, a confirmation dialog will appear asking your permission for the action. Just confirm your action by clicking on the OK button.
- In the indicate anchor wizard, we have to indicate the adjacent button, that is, the Cancel button, so that the recorder will identify that the button is adjacent to Cancel.
 - Now recording is complete, press Esc to get to the recording dialog. Click on the Save & Exit button.
 - Then, in UiPath Studio, you can see a recording sequence in the Designer panel.
 - Now run it by pressing the; F5 key; it should perform the same task again. You have created your first Robot, which empties trash from your Gmail!

Screen Scraping

We can interact with the controls in the UI. Sometimes, you may need to click on a particular button or extract information from a textbox. Either we take some action on a control, or we read/write. We will go into detail on how to do this accurately.

Extraction is a primary feature of RPA, enabling UI automation. Behind the scenes, many technologies are at work on the seamless extraction of information from the UI. When typical RPA techniques are not successful, OCR technology is used to extract information. We will learn about using OCR and other techniques in the following topics:

- a. Screen Scraping
- b. When to use OCR
- c. Types of OCR available
- d. How to use OCR

Screen Scraping

Screen scraping is a data collection method used to gather information shown on a display to use for another purpose. Typically, screen scraping is used to collect data from one application to then translate it to another -- or, more controversially, to steal data.

Visual data is collected as plain text from onscreen information, such as text, images, or charts that appear on the desktop, in an application, or on a website.

By comparison, gathering data from a display through an automatic process like screen scraping is significantly faster and more efficient than collecting data manually.

Screen scraping can be performed automatically with a scraping program or manually by an individual extracting data. Screen scraping programs are designed to search through and recognize elements in a user interface (UI). Data from that display is extracted and transformed into text. When the displayed data contains images, a screen scraper utilizes optical character recognition (OCR) technology to gather information.

Data Scraping

Data scraping, or web scraping, is a process of importing data from websites into files or spreadsheets. It is used to extract data from the web, either for personal use by the scraping operator or to reuse the data on other websites. There are numerous software applications for automating data scraping.

Data scraping is commonly used to:

- Collect business intelligence to inform web content
- Determine prices for travel booking or comparison sites
- Find sales leads or conduct market research via public data sources
- Send product data from eCommerce sites to online shopping platforms like Google Shopping

Data scraping has legitimate uses, but is often abused by bad actors. For example, data scraping

is often used to harvest email addresses for the purpose of spamming or scamming. Scraping can also be used to retrieve copyrighted content from one website and automatically publish it on another website.

Some countries prohibit the use of automated email harvesting techniques for commercial gain, and it is generally considered an unethical marketing practice.

Scraping advanced techniques

Scraping dynamic and JavaScript-heavy sites: Many websites today use JavaScript to dynamically load content. This can make it difficult to scrape the data using traditional methods. One way to get around this is to use a tool like Selenium, which can simulate human browser activity. This allows you to interact with the JavaScript on the page and extract the data you need.

Dealing with CAPTCHAs and IP blocking: Some websites use CAPTCHAs to prevent automated scraping. CAPTCHAs are challenges that are designed to be easy for humans to solve but difficult for bots. There are a number of ways to get around CAPTCHAs, such as using a CAPTCHA solver service or using a proxy server.

Using proxies and rotating IP addresses: Proxies are servers that act as intermediaries between your computer and the website you are scraping. Using a proxy can help to hide your real IP address and prevent the website from blocking you. Rotating IP addresses means using a different proxy for each request. This can help to further protect your identity and prevent the website from detecting that you are scraping.

Using crawlers and spiders: Crawlers and spiders are programs that can automatically navigate through websites and extract data. These tools can be used to scrape large amounts of data from multiple websites.

Using APIs: Many websites offer APIs that can be used to programmatically access data. APIs are often easier to use than scraping, and they can provide a more reliable source of data.

Using a hybrid approach: In some cases, it may be necessary to use a hybrid approach to web scraping. This means using a combination of automated and manual techniques. For example, you could use a crawler to extract the data from a website, and then manually verify the data and correct any errors.

It is important to note that web scraping is not always legal. Some websites have terms of service that prohibit scraping. It is important to check the terms of service of the website you are scraping before you begin.

Selectors

Selectors in Robotic Process Automation (RPA) are used to identify the elements on a user interface (UI) that you want to interact with. There are a number of different types of selectors, each of which has its own strengths and weaknesses. Some of the most common selectors include:

UIPath selectors: These selectors are used in UiPath Studio to identify elements on the UI. They are based on XPath syntax, but they also include some additional features, such as the ability to select elements based on their accessibility properties.

CSS selectors: These selectors use CSS syntax to identify elements on the UI. They are less powerful than UIPath selectors, but they are often easier to use.

XPath selectors: These selectors use XPath syntax to identify elements on the UI. They are the most powerful type of selector, but they can also be the most difficult to use.

Defining and assessing selectors

Defining and assessing selectors in RPA is the process of determining which selectors to use to identify the elements on the UI that you want to interact with. This can be a challenging task, as there is no one-size-fits-all approach. The best way to define and assess selectors is to experiment and see what works best for the specific UI you are automating.

Customization

Customization in RPA is the process of modifying selectors to make them more specific or to select different elements. This can be done by adding or removing attributes, by using more complex XPath expressions, or by using other techniques.

Debugging

Debugging in RPA is the process of finding and fixing errors in your automation code. This can be a challenging task, as errors can be caused by a variety of factors, such as incorrect selectors, invalid XPath expressions, or network problems. The best way to debug your automation code is to use a tool like UiPath Studio's Debug Recorder to record your interactions with the UI and then use the recorder to step through your code and see where the errors are occurring.

Here are some examples of how to use selectors in RPA:

```
# Select the element with the ID "my_element"
UiPath.UIPathSelectors.ById("my_element")

# Select the element with the class name "my_class"
UiPath.UIPathSelectors.ByClassName("my_class")

# Select the element with the text "My Text"
UiPath.UIPathSelectors.ByText("My Text")
```

Dynamic Selectors

Dynamic Selectors in RPA are selectors that use variables or arguments to identify elements on the UI. This can be useful for automating tasks where the UI elements change dynamically, such as the text of a button or the contents of a table.

To use a dynamic selector in RPA, you can use a variable or argument in the selector string. For example, the following selector would select the button with the text that is stored in the

variable my_text:

```
UiPath.UiPathSelectors.ByText(my_text)
```

You can also use dynamic selectors to select elements based on their attributes. For example, the following selector would select the element with the attribute id that is equal to the value of the variable my_id:

```
UiPath.UiPathSelectorsById(my_id)
```

Partial Selectors

Partial Selectors in RPA are selectors that do not contain the full selector for the element that you want to interact with. This can be useful for automating tasks where the UI elements are nested or where the full selector is difficult to determine.

To use a partial selector in RPA, you can use the Attach Window or Attach Browser activity to attach to the window or browser that contains the element that you want to interact with. Once you have attached to the window or browser, you can use a partial selector to select the element.

For example, the following selector would select the button with the text "My Button" in the window with the title "My Window":

```
UiPath.UiPathSelectors.ByText("MyButton", UiPath.UiPathSelectors.Window("My Window"))
```

You can also use partial selectors to select elements based on their attributes. For example, the following selector would select the element with the attribute id that is equal to the value of the variable my_id in the window with the title "My Window":

```
UiPath.UiPathSelectorsById(my_id, UiPath.UiPathSelectors.Window("My Window"))
```

Dynamic selectors and partial selectors can be very useful for automating tasks where the UI elements change dynamically or where the full selector is difficult to determine. By using these selectors, you can make your automation code more robust and reliable.

RPA Challenge

An RPA Challenge is a competition or contest to test and showcase the skills of RPA developers. Challenges can be focused on a specific aspect of RPA, such as automating a particular task or process, or they can be more general.

RPA challenges are a great way for developers to learn new skills, improve their existing skills, and network with other RPA professionals. They can also be a fun way to compete against others and test your skills.

Here are some examples of RPA challenges:

- Automating a specific task or process: For example, a challenge might be to automate the process of filling out a form, extracting data from a PDF, or sending an email.

- **Building an RPA workflow:** For example, a challenge might be to build an RPA workflow to automate a customer onboarding process or a financial reporting process.
- **Creating a chatbot:** For example, a challenge might be to create a chatbot that can answer customer questions or provide support.

RPA challenges can be hosted by individual organizations, such as RPA software vendors or consulting firms, or they can be part of larger industry events, such as conferences or trade shows.

RPA challenge Instructions:

- **Read the rules carefully:** Make sure you understand the rules of the challenge before you start working on your solution. This includes understanding the scope of the challenge, the deliverables, and the judging criteria.
- **Plan your solution:** Before you start coding, take some time to plan your solution. This will help you to avoid making mistakes and to ensure that you meet the requirements of the challenge.
- **Test your solution thoroughly:** Once you have implemented your solution, be sure to test it thoroughly to make sure that it works as expected.
- **Submit your solution on time:** Be sure to submit your solution before the deadline.

Participating in an RPA challenge is a great way to learn new skills, improve your existing skills, and network with other RPA professionals. If you are interested in RPA, I encourage you to participate in a challenge.

Image, text, and advanced Citrix automation

Image, text, and advanced Citrix automation can be used to automate a wide range of tasks in Citrix environments. Some examples include:

Image recognition: Image recognition can be used to identify and interact with elements on the Citrix screen, such as buttons, text boxes, and menus. This can be useful for automating tasks where the location or appearance of the elements may change, such as logging into a Citrix application or entering data into a form.

Text recognition: Text recognition can be used to extract text from the Citrix screen and store it in a variable. This can be useful for automating tasks where you need to process the text, such as filling out a web form or generating a report.

Advanced Citrix automation: Advanced Citrix automation techniques can be used to automate complex tasks in Citrix environments, such as interacting with multiple applications, handling errors, and executing macros.

Here are some examples of how image, text, and advanced Citrix automation can be used to automate tasks in Citrix environments:

- **Log into a Citrix application:** You can use image recognition to identify and click on the login button for the Citrix application. You can then use text recognition to extract the username and password fields from the login screen and enter your credentials.

- **Enter data into a form:** You can use image recognition to identify and click on the input fields in the form. You can then use text recognition to extract the data that you need to enter into the form from a spreadsheet or other source.
- **Generate a report:** You can use text recognition to extract data from a Citrix application and store it in a variable. You can then use the data in the variable to generate a report using a spreadsheet or other tool.
- **Interact with multiple applications:** You can use advanced Citrix automation techniques to interact with multiple Citrix applications at the same time. For example, you could use one application to enter data into a form and another application to submit the form.
- **Handle errors:** You can use advanced Citrix automation techniques to handle errors that may occur during your automation. For example, you could use a try-catch block to handle errors that occur when interacting with a Citrix application.
- **Execute macros:** You can use advanced Citrix automation techniques to execute macros in Citrix environments. This can be useful for automating complex tasks that involve multiple steps.

Image, text, and advanced Citrix automation can be a powerful tool for automating a wide range of tasks in Citrix environments. By using these techniques, you can save time and improve the efficiency of your business processes.

Introduction to Image & Text Automation

Image and text automation in UiPath is the process of using UiPath Studio to automate the processing of images and text. This can be used to automate a wide range of tasks, such as:

- Extracting data from images and text documents
- Classifying images and text documents
- Converting images and text documents to other formats
- Generating images and text documents

Image and text automation in UiPath can be implemented using a variety of different activities, including

Image Recognition Activities: These activities allow you to identify and interact with images on the screen. For example, you can use the Click Image activity to click on a specific image, or the Extract Text From Image activity to extract text from an image.

Text Automation Activities: These activities allow you to manipulate text on the screen. For example, you can use the Type Text activity to type text into a field, or the Compare Text activity to compare two pieces of text.

OCR Activities: These activities allow you to extract text from images and PDF files. For example, you can use the Get OCR Text activity to extract text from an image, or the Convert PDF To Text activity to convert a PDF file to a text file.

To implement image and text automation in UiPath, you will typically need to follow these steps:

- Identify the task that you want to automate.
- Design the UiPath Studio workflow.
- Add the necessary activities to the workflow.
- Configure the activities.
- Test and deploy the workflow.

Here are some tips for implementing image and text automation in UiPath:

- Start with a simple task. Once you have successfully automated a simple task, you can move on to more complex tasks.
- Use the UiPath Studio documentation to learn more about the different image and text automation activities.
- Record the screen while you perform the task that you want to automate. This can help you to identify the steps that you need to automate and to configure the UiPath Studio activities correctly.
- Use UiPath Studio's debugging tools to troubleshoot any problems that you encounter.
- Get help from the UiPath community if you need it. There are a number of experts in the UiPath community who can help you to automate your image and text processing tasks.

Image and text automation in UiPath can be a powerful tool for automating a wide range of tasks. By following the tips above, you can implement image and text automation in UiPath and start to reap the benefits of this technology.

Here are some examples of how image and text automation can be used in UiPath:

- Extract data from invoices and receipts to automate accounting tasks.
- Extract data from medical records to automate healthcare tasks.
- Extract data from insurance claims to automate insurance tasks.
- Extract data from legal documents to automate legal tasks.
- Extract data from product images to automate manufacturing and retail tasks.
- Extract data from customer reviews to automate customer service tasks.

These are just a few examples of how image and text automation can be used in UiPath. With a little creativity, you can use image and text automation to automate a wide range of tasks in your own organization.

Image-based automation

Image-based automation in RPA is the use of computer vision technology to automate tasks that involve interacting with images on the screen. This can be used to automate a wide range of tasks, such as:

- Extracting data from images and text documents, such as invoices, receipts, and medical records

- Classifying images and text documents, such as product images and customer reviews
- Converting images and text documents to other formats, such as PDF to XML or JPEG to PNG
- Generating images and text documents, such as reports and marketing materials

Image-based automation in RPA can be implemented using a variety of different tools and technologies. Some of the most popular tools include:

UiPath: UiPath is a low-code/no-code RPA platform that includes image-based automation capabilities.

ABBYY: ABBYY is a company that develops software for automating tasks such as optical character recognition (OCR), document processing, and data capture.

Google Cloud Vision: Google Cloud Vision is a cloud-based image and text processing API that can be used to automate tasks such as object detection, image classification, and text analysis.

To implement image-based automation in RPA, you will typically need to follow these steps:

- Identify the task that you want to automate.
- Choose an image-based automation tool or technology.
- Design the RPA automation process.
- Implement the RPA automation process.
- Test and deploy the RPA automation process.

Keyboard-based Automation

Keyboard-based automation in RPA is the use of software to automate tasks that involve interacting with the keyboard. This can be used to automate a wide range of tasks, such as:

- Entering data into forms
- Filling out web pages
- Typing commands in applications
- Navigating through menus
- Pressing keyboard shortcuts

To implement keyboard-based automation in UiPath, you can use the following steps:

- Identify the task that you want to automate. What specific actions do you need to perform on the keyboard?
- Open UiPath Studio and create a new blank process.
- Add the necessary activities to the process. You can use the Type Into activity to type text into a field, the Send Hotkey activity to send keyboard shortcuts, and the Click activity to click on elements on the screen.
- Configure the activities. For example, if you are using the Type Into activity, you will need to enter the text that you want to type into the Text field.

- Test and deploy the process. Once you have added the necessary activities and configured them, you can test the process to make sure that it works correctly. Once you are satisfied with the process, you can deploy it to production.

How to use keyboard-based automation in UiPath:

- **Automate the process of logging into and navigating applications.** You can use the *Type Into* activity to enter your username and password into the login screen, and then use the *Click* activity to click on the login button. Once you are logged in, you can use the *Send Hotkey* activity to send keyboard shortcuts to navigate through the application.
- **Automate the process of filling out forms and entering data.** You can use the *Type Into* activity to enter data into the form fields. You can also use the *Send Hotkey* activity to send keyboard shortcuts to move between fields and to submit the form.
- **Automate the process of generating reports and presentations.** You can use the *Type Into* activity to type text into the report or presentation. You can also use the *Send Hotkey* activity to send keyboard shortcuts to format the text and to insert images and charts.

Keyboard-based automation in UiPath is a powerful tool that can be used to automate a wide range of tasks.

Information Retrieval

To retrieve information in UiPath, we can use the following steps:

- **Identify the information that you want to retrieve.** What specific data do you need to extract?
- **Identify the sources where the information is located.** Is the information stored in a database, a website, or a document?
- **Choose a UiPath activity to retrieve the information.** Some popular options include the *Get Text* activity, the *Extract Data From Table* activity, and the *Get From Database* activity.
- **Configure the UiPath activity to retrieve the information from the source.** This will involve providing the activity with the necessary information, such as the database connection string or the URL of the website.
- **Store the retrieved information in a variable.** This will allow you to use the information in other parts of your UiPath workflow.
- **Test and deploy your UiPath workflow.** Once the workflow has been implemented, you need to test it to make sure that it is working correctly. Once you are satisfied with the workflow, you can deploy it to production.

Example:

Here are some examples of how to retrieve information in UiPath:

- **Extract data from invoices and receipts to automate accounting tasks.** You can use

the *Get Text* activity to extract the data from the invoices and receipts and store it in a variable. This data can then be used to automate accounting tasks such as generating reports and paying bills.

- **Extract data from medical records to automate healthcare tasks.** You can use the *Extract Data From Table* activity to extract the data from medical records and store it in a variable. This data can then be used to automate healthcare tasks such as scheduling appointments and generating reports.
- **Extract data from insurance claims to automate insurance tasks.** You can use the *Extract Data From Table* activity to extract the data from insurance claims and store it in a variable. This data can then be used to automate insurance tasks such as processing claims and generating reports.
- **Extract data from legal documents to automate legal tasks.** You can use the *Get Text* activity to extract the data from legal documents and store it in a variable. This data can then be used to automate legal tasks such as drafting documents and preparing for trials.
- **Extract data from product images to automate manufacturing and retail tasks.** You can use the *Get Text* activity to extract the data from product images, such as the product name, description, and price. This data can then be used to automate manufacturing and retail tasks such as updating inventory and generating product listings.
- **Extract data from customer reviews to automate customer service tasks.** You can use the *Get Text* activity to extract the data from customer reviews, such as the customer's name, rating, and feedback. This data can then be used to automate customer service tasks such as responding to customer inquiries and identifying customer satisfaction trends.

Advanced Citrix Automation challenges

- **Image-based automation:** Citrix uses a proprietary image-based protocol to render applications, which can make it difficult to automate tasks using traditional methods. UiPath provides a number of image-based automation activities, such as *Get Image* and *Click Image*, but these activities can be challenging to use in Citrix due to the way it renders applications.
- **Citrix keyboard shortcuts:** Citrix uses a number of keyboard shortcuts to control various functions of the Citrix environment, such as switching between windows and launching applications. These keyboard shortcuts can be difficult to automate using traditional methods, as they often require the use of special keys or combinations of keys. UiPath provides a number of keyboard automation activities, such as *Send Hotkey* and *Type Into*, but these activities can be challenging to use in Citrix due to the way it renders keyboard shortcuts.
- **Citrix security:** Citrix is a highly secure platform, which can make it difficult to automate tasks that require privileged access. For example, automating tasks that require the user to enter a password or log in to a system can be challenging in Citrix. UiPath provides a number of security features, such as *Run as Another User*, but these features can be challenging to implement in Citrix environments.
- **Citrix performance:** Citrix applications can be performance-sensitive, so it is important to automate tasks in a way that does not impact the performance of the application. This can be challenging, especially for complex tasks or tasks that require a lot of data to be

transferred. UiPath provides a number of performance optimization features, such as Caching and Delay, but these features can be challenging to implement in Citrix environments.

- **Citrix version compatibility:** Citrix releases regular updates to its platform, and it is important to ensure that your UiPath automation solutions are compatible with the latest version of Citrix. This can be challenging, as it requires keeping your UiPath automation solutions up to date.
- **Citrix environment complexity:** Citrix environments can be complex, with multiple applications and servers. This can make it difficult to design and implement UiPath automation solutions that are scalable and maintainable.
- **Citrix troubleshooting:** Troubleshooting UiPath automation solutions in Citrix environments can be challenging, as there are a number of potential factors that can impact the performance and reliability of automation.

Best practices for overcoming advanced Citrix automation challenges:

- **Use the UiPath Citrix Extension:** The UiPath Citrix Extension is a free extension that provides a number of features to help you automate Citrix applications in UiPath. These features include image-based automation support, keyboard automation support, and Citrix security support.
- **Use UiPath Studio:** UiPath Studio is a powerful automation platform that provides a number of features to help you design, implement, and test UiPath automation solutions. These features include a drag-and-drop interface, a library of pre-built activities, and a debugging environment.
- **Use the UiPath community:** The UiPath community is a large and active community of UiPath users who can provide support and advice. If you are having trouble automating a Citrix task in UiPath, you can post a question in the UiPath forum or search the UiPath knowledge base for help.

Using Tab for Images

To use the Tab key for images in UiPath, we can use the following steps:

- Open UiPath Studio and create a new blank process.
- Add an *Image* activity to the process.
- In the *Image* activity properties, set the *Image* property to the image that you want to automate.
- Add a *Send Hotkey* activity to the process.
- In the *Send Hotkey* activity properties, set the *Hotkey* property to *Tab*.
- Connect the *Image* activity to the *Send Hotkey* activity.
- Run the process.

When the process runs, the UiPath Robot will navigate to the image and press the Tab key. This will cause the focus to move to the next element on the screen. If this doesn't work then check for the following:

- Make sure that the image that you want to automate is visible on the screen.

- If the image is not visible on the screen, you can use the Click Image activity to click on the image to make it visible.
- You can use the Delay activity to add a delay between the Image activity and the Send Hotkey activity. This can be useful if the image takes some time to load or if the next element on the screen takes some time to appear.
- You can use the Try/Catch activity to handle any errors that may occur when using the Tab key. For example, if the next element on the screen is not focusable, the Robot will throw an error. You can use the Try/Catch activity to handle this error and continue with the process.
-

we can use this approach to automate tasks such as

- **Filling out forms:** You can use the Tab key to navigate to the next field on a form and enter data.
- **Navigating through menus:** You can use the Tab key to navigate to the next menu item and select it.
- **Switching between windows:** You can use the Tab key to switch to the next window in focus

Starting Apps

Start Process

UiPath.Core.Activities.StartProcess

Launches a specified application, and can optionally pass a list of arguments to it.

Properties

1. **ContinueOnError** - Specifies if the automation should continue even when the activity throws an error. This field only supports Boolean values (True, False). The default value is False. As a result, if the field is blank and an error is thrown, the execution of the project stops. If the value is set to True, the execution of the project continues regardless of any error.

NOTE: If this activity is included in Try Catch and the value of the ContinueOnError property is True, no error is caught when the project is executed.

2. **DisplayName** - The display name of the activity.
3. **Input:**
 - Arguments - The parameters that can be passed to the application at startup.
 - FileName - The full file path where you can find the executable file of the application to be opened.
 - WorkingDirectory - Path of the current working directory. This field accepts only string variables.

NOTE: Files with all extension types are supported in this field. If you add a file with a non-executable extension type, it is opened with its corresponding application, if available.

4. Misc

Private - If selected, the values of variables and arguments are no longer logged at Verbose level.

5. Options:

- ExecutionType - Specifies the process execution type. The process can be executed asynchronously, which indicates that the activity starts the process, but does not wait for it to finish, or synchronously, which indicates that the activity starts the process and waits for it to finish within the allotted timeout. By default, the process is executed asynchronously.
- Timeout (milliseconds) - Specifies the amount of time (in milliseconds) to wait for the process to finish before an error is thrown. This property is valid only for synchronous execution. The default value is 30000 milliseconds (30 seconds). A value of -1 indicates that the activity should wait indefinitely for the process to exit.

Example of using the Start Process activity

The example below explains how to use the Start Process activity for opening the Notepad application.

1. Open Studio and create a new Process named by default Main.

NOTE: Create a new .txt file, save it with the name StartProcess.txt and place it in the project folder.

2. Drag a Sequence container in the Workflow Designer.

Create the following variables:

Variable Name	Variable Type	Default Value
NotepadPath	GenericValue	
NotepadFilePath	GenericValue	

3. Drag an Assign activity inside the Sequence
 - In the Properties panel, add the variable NotepadPath in the To field and the executable application, in this case "*notepad.exe*", in the Value field.
4. Add another Assign activity and place it below the first one.
 - In the Properties panel, add the variable *NotepadFilePath* in the To field and the actual path of the file, in this example "*StartProcess.txt*", in the Value field.
5. Drag a Start Process activity below the Assign activities.
 - In the Properties panel, add the variable *NotepadFilePath* in the Arguments field.
 - Add the variable NotepadPath in the FileName field.
6. Run the process. The automation process opens the StartProcess.txt file.

Excel Data Tables & PDF

UiPath can be used to automate a variety of tasks related to Excel data tables and PDFs. Here are some examples:

Excel data tables

- **Read data from an Excel table:** You can use the Read Range activity to read data from an Excel table into a UiPath variable.
- **Write data to an Excel table:** You can use the Write Range activity to write data from a UiPath variable to an Excel table.
- **Filter an Excel table:** You can use the Filter Data Table activity to filter an Excel table based on specified criteria.
- **Sort an Excel table:** You can use the Sort Data Table activity to sort an Excel table based on specified criteria.
- **Calculate statistics on an Excel table:** You can use the Calculate Data Table Statistics activity to calculate statistics on an Excel table, such as the average, sum, and standard deviation.

PDFs

- **Extract data from a PDF:** You can use the Extract Data From PDF activity to extract data from a PDF table or from the text of a PDF.
- **Generate a PDF from an Excel table:** You can use the Generate PDF activity to generate a PDF from an Excel table.
- **Merge multiple PDFs:** You can use the Merge PDF activity to merge multiple PDFs into a single PDF.
- **Split a PDF into multiple pages:** You can use the Split PDF activity to split a PDF into multiple pages.

Save Excel File As PDF

UiPath.Excel.Activities.Business.SaveAsPdfX

– Saves an Excel workbook as a PDF file. The activity can be used with an Excel file selected for a parent Use Excel File activity or with the Project Notebook.

Some Options for saving as PDF file:

1. **Workbook** - Click Plus on the right side of the field and then, from the menu, select the Excel workbook to save as PDF. Alternatively, you can select Open in Advanced Editor and enter a VB expression.
2. **Save as file** - Click Browse next to the field, and then browse to the folder where to create the file and enter the file name. Alternatively, you can indicate the full path of the file to create by using one of the options in the Plus menu on the right side of the field:
 - a. **Data from the Project Notebook**, a parent Excel file or Outlook account. For example, select an Excel file, and then select a cell that contains a file path.
 - b. **Use Saved Value** - Select a value in the form of a file path that you previously

saved for later use in the project.

- c. Text - Enter a file path in the Text Builder.
- d. Ask when run - Prompt for a file path when the project is executed.
- e. Open in Advanced Editor - Enter a VB expression.

If a file with the same name already exists in the specified location, the file is overwritten. If only a file name is specified, the PDF file is created in the project folder.

Data Tables in RPA

Data is a very integral part of any process and it isn't any different for UiPath automation. In most data driven processes, we are either reading data or modifying it. Data Table is the type of variable that can store data as a simple spreadsheet with rows and columns. You can identify each piece of data based on its unique column and row coordinates.

In Data Tables, the regular convention of identifying the columns and the rows are applied. Columns are identified through capital letters and rows through numbers.

A Data Table is an in-memory representation of a single database table which has a collection of rows and columns.

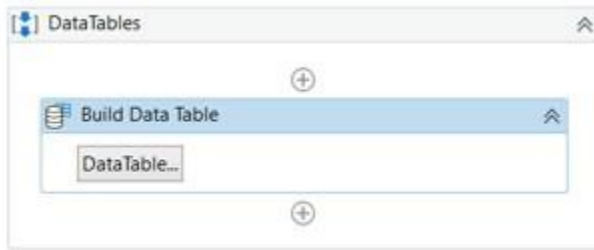
How are Data Tables Created?

The most common activities and method to create Data Tables are:

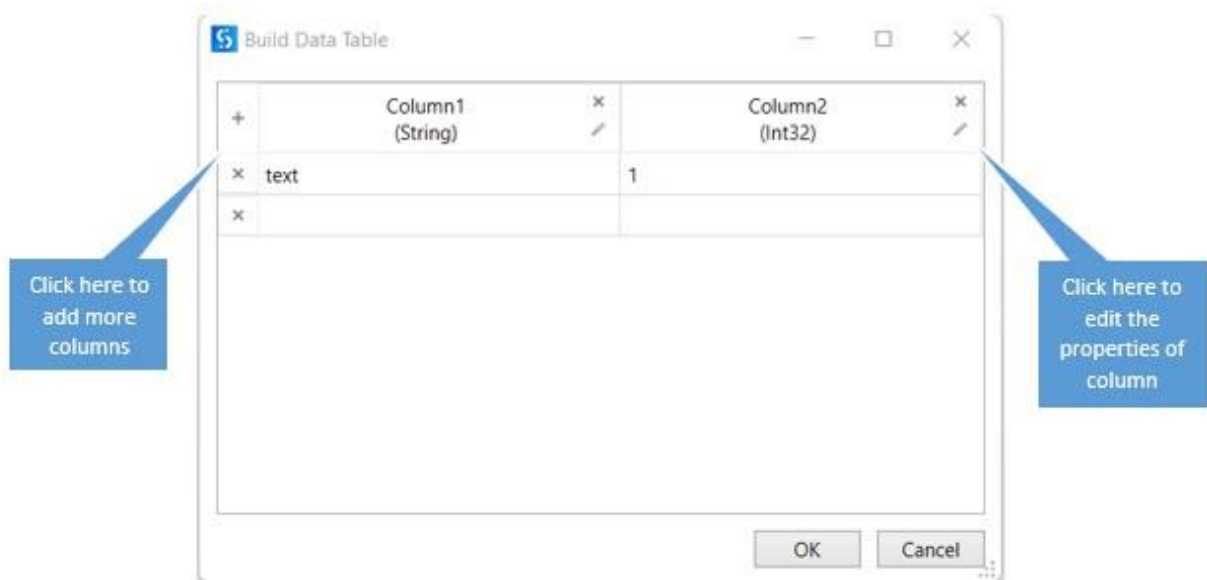
1. The Build Data Table Activity
2. The Read Range Activity
3. The Read CSV Activity
4. The Data Scraping Action
5. The Generate Data Table from Text Activity

1. **Build Data Table Activity:** – By using this activity, you choose the number of columns and the data type of each of them. Moreover, you can configure each column with specific options like allow null values, unique values, auto-increment (for numbers), default value and length (for Strings).

- a. Drag and Drop Build Data Table Activity from the activity panel.

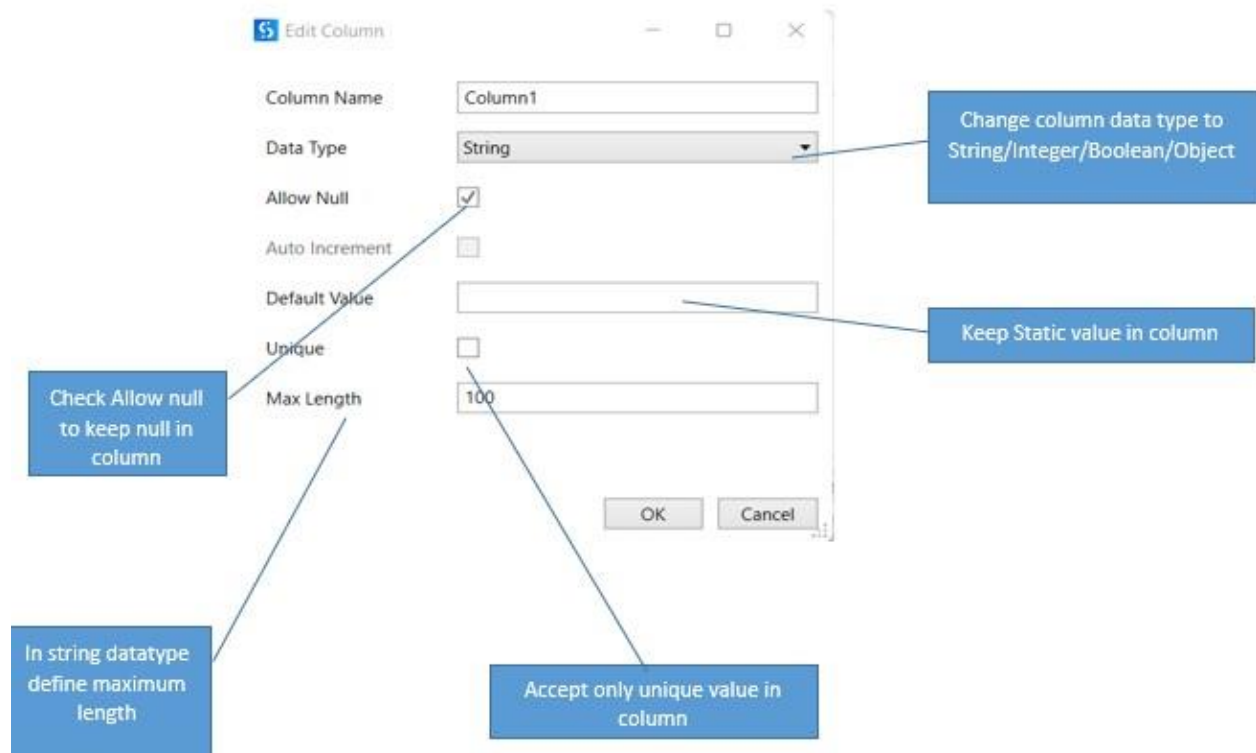


b. Click on Data Table and Configure column as per requirements.



c. Column properties configuration

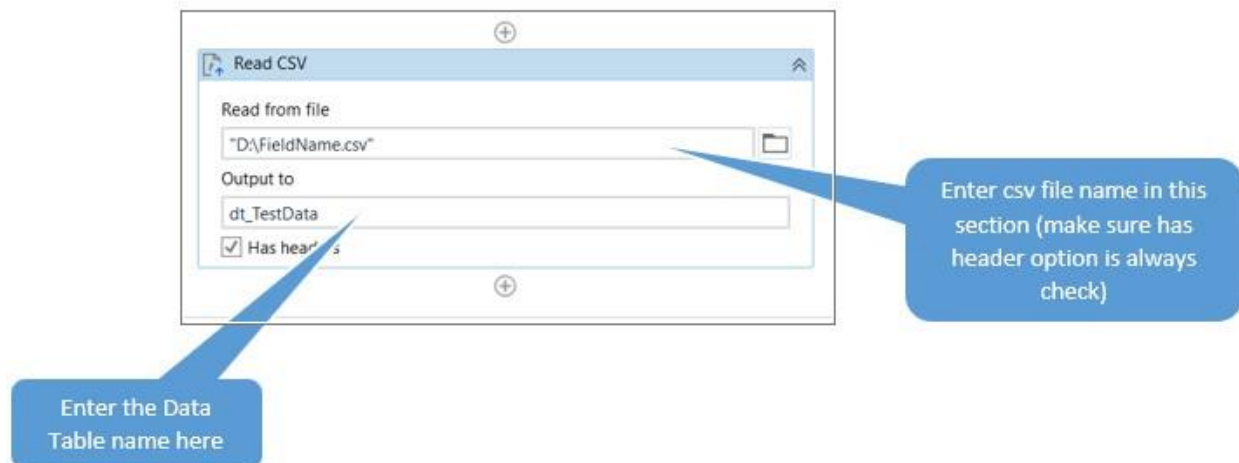




2. Read CSV Activity: – This activity captures the content of a csv file and stores it in a Data Table variable. Although not commonly used anymore, there are still legacy or internal-built applications with this kind of documents. This activity resides under the app integration section under CSV option.



You can directly drag and drop the activity into the workflow and start using it.

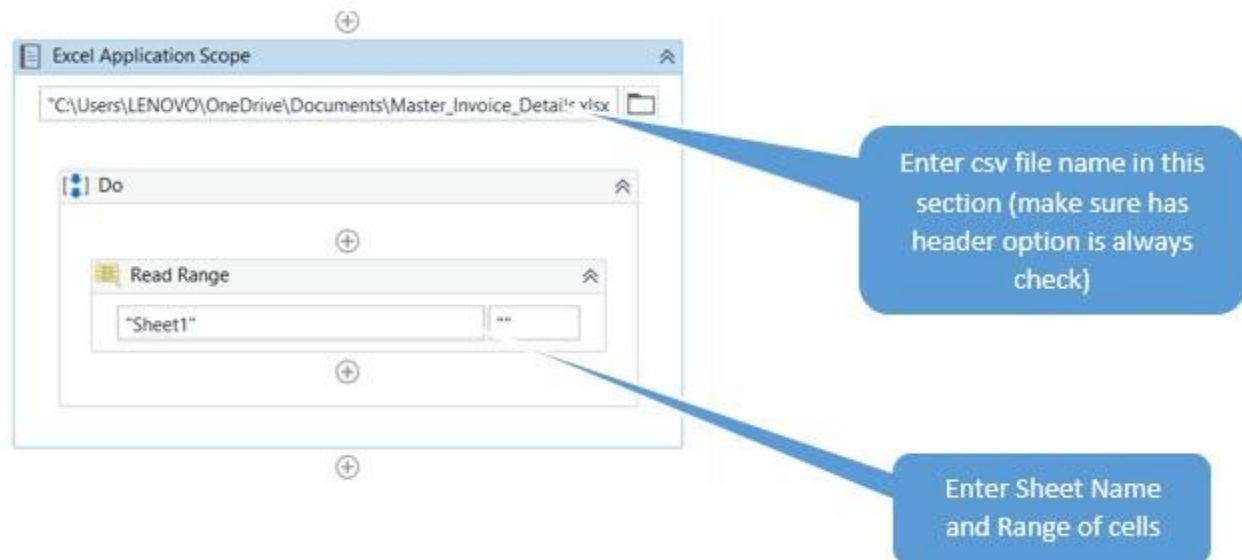


3. Read Range Activity: – This is the most common activity to create data table which is being used when you are working on the excel files. This activity gets the content of worksheet (or a selection from that worksheet) and stores it in a data table variable, which can be created from the properties panel using Ctrl + K.

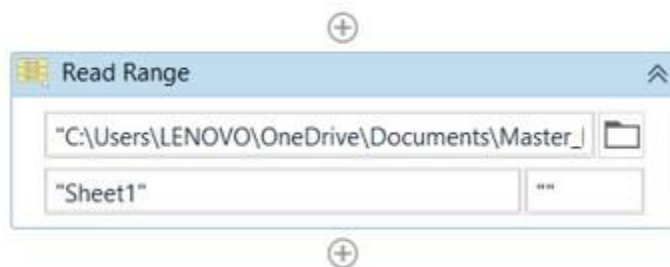


There are two types of Read Range activities available in Activity panel. One under the Excel Activity and the other is under the Workbook Activity. There is no hard difference between both the activities. Both populate the data table as an output. For Excel activity Read Range you need to use Excel Application scope whereas for workbook activity you can directly use the Read Range.

Excel Activity

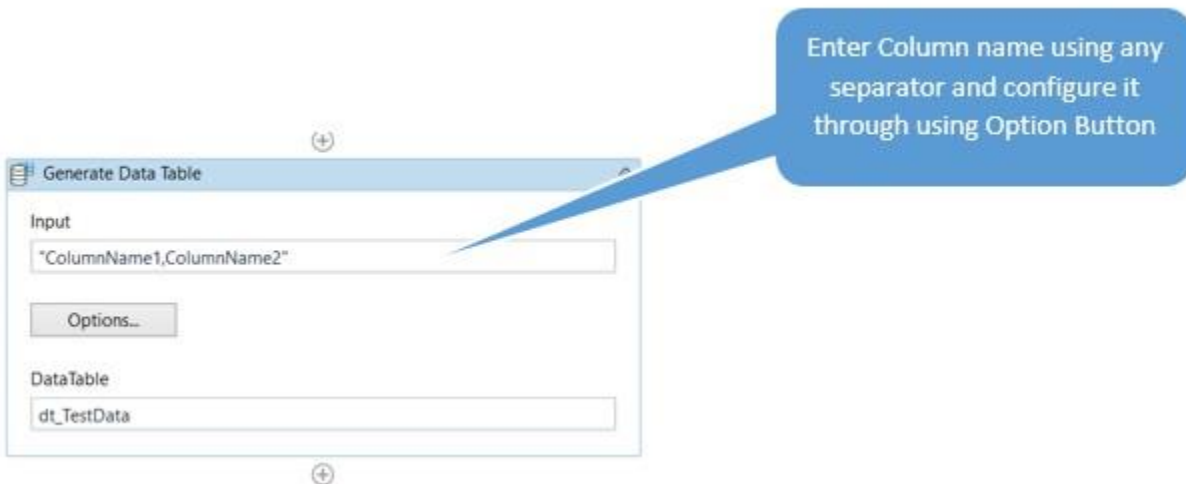


Workbook Activity



4. **Data Scraping Action:** – This functionality of UiPath studio enables you to extract structured data from your browser, application, or document to a Data Table.

5. **Generate Data Table from Text Activity:** – This can be used to create a Data Table from structured text, by letting the user indicate the row and column separators. You can use this activity by drag and drop from activity panel.



UiPath offers a broad range of activities that can be used to work with Data Table Variable: –

1. **Add Data Column:** – Adds a column to an existing Data Table variable. The input data can be of Data Column type or the column can be added empty by specifying the data type and configuring the options (allowing null values, requesting unique values, auto-incrementing, default value and maximum length).
2. **Add Data Row:** – Adds a new row to an existing Data Table variable. The input data can be of Data Row type or can be entered as an Array Row, by matching each object with the data type of each column.
3. **Build Data Table:** – Is used to create a Data Table using a dedicated window. This activity allows the customization of number of columns and type of data for each column.
4. **Clear Data Table:** – Clears all the data in an existing Data Table variable.
5. **Filter Data Table:** – Allows filtering a Data Table through a Filter Wizard using various conditions. This activity can be configured to create a new Data Table for the output of the activity or to keep the existing one and filter (delete) the entries that do not match the filtering conditions.
6. **For Each Row in Data Table:** – Is used to perform a certain activity for each row of Data Table (like For Each Loop).
7. **Generate Data Table from Text:** – Can be used to create a Data Table from structured text by letting the user indicate the row and column separators.
8. **Join Data Tables:** – Combines rows from row tables by using values common to each other using the Join Wizard, according to a join rule that answers the question “What to do with the data that doesn’t match?”. It is one of the most useful activities in business scenarios, where working with more than one Data Table is very common.
9. **Lookup Data Table:** – It is like VLOOKUP in Excel. You can search for a provided value in the specified Data Table and the Row Index returns its value. It can also be configured to return the value from a cell with the given coordinates (Row Index and Target Column).
10. **Merge Data Table:** – Is used to append a specified Data Table to the current Data Table. The operation is simpler than Join Data Type activity, as it has 4 predefined

actions to perform over the missing schema.

11. **Output Data Table:** – Writes a Data Table to a string using the CSV format.
12. **Remove Data Column:** – Removes a certain column from a specified Data Table. The input may consist of the column index, column name or Data Column variable.
13. **Remove Data Row:** – Removes a row from a specified Data Table. The input may consist of the row index or a Data Row variable.
14. **Remove Duplicate Row:** – Removes the duplicate rows from a specified Data Table variable, keeping only the first occurrence.
15. **Sort Data Table:** – Can sort a Data Table in an ascending or descending order of the values in a specific column.
16. **Get Row Item:** – Retrieves a value from a row in a Data Table according to a specified column.
17. **Update Row Item:** – Assigns a specified value to the indicated column of a Data Table row.

Excel and data tables are two of the most important features of UiPath. Excel is a powerful spreadsheet application that can be used to store and manipulate data in a variety of ways. Data tables are a type of variable in UiPath that can be used to store data in a structured format.

Excel and Data Table basics

- Reading data from Excel: You can use the Read Range activity to read data from an Excel table into a UiPath variable.
- Writing data to Excel: You can use the Write Range activity to write data from a UiPath variable to an Excel table.
- Filtering a data table: You can use the Filter Data Table activity to filter a data table based on specified criteria.
- Sorting a data table: You can use the Sort Data Table activity to sort a data table based on specified criteria.
- Calculating statistics on a data table: You can use the Calculate Data Table Statistics activity to calculate statistics on a data table, such as the average, sum, and standard deviation.

To use Excel and data tables in UiPath, you will need to:

- Install the UiPath Excel Activities package. This package provides a number of activities that can be used to read and write data to Excel, as well as to manipulate data tables.
- Create a data table variable. This variable will store the data that you want to manipulate.
- Use the UiPath Excel Activities to read data from Excel and write data to Excel.
- Use the UiPath data table activities to manipulate the data table variable.

Here is a simple example of how to use Excel and data tables in UiPath:

```
// Create a data table variable.
```

```
DataTable dt = new DataTable();
```

```
// Read data from Excel and store it in the data table variable.
```

```
dt ReadRange(@"C:\Temp\data.xlsx");
```

```
// Filter the data table variable.
```

```
dt = dt.Filter("Country = 'United States'");
```

```
// Write the data table variable to Excel.
```

```
WriteRange(@"C:\Temp\filtered_data.xlsx", dt);
```

This example reads data from an Excel file called data.xlsx and stores it in a data table variable. The data table variable is then filtered to only include rows where the Country column is equal to United States. The filtered data table variable is then written to an Excel file called filtered_data.xlsx.

You can use UiPath to automate a wide variety of tasks related to Excel and data tables. For example, you could use UiPath to:

- Generate reports from Excel data.
- Extract data from PDFs and store it in Excel data tables.
- Merge multiple Excel data tables into a single data table.
- Update Excel data tables based on data from other sources.

Data Manipulation in Excel

The example below explains how to copy ranges from a workbook, paste, and append them to another workbook. It presents activities such as Read Range Workbook, Write Range Workbook, and Append Range Workbook. You can find these activities in the UiPath.Excel.Activities package.

This is how the automation process can be built:

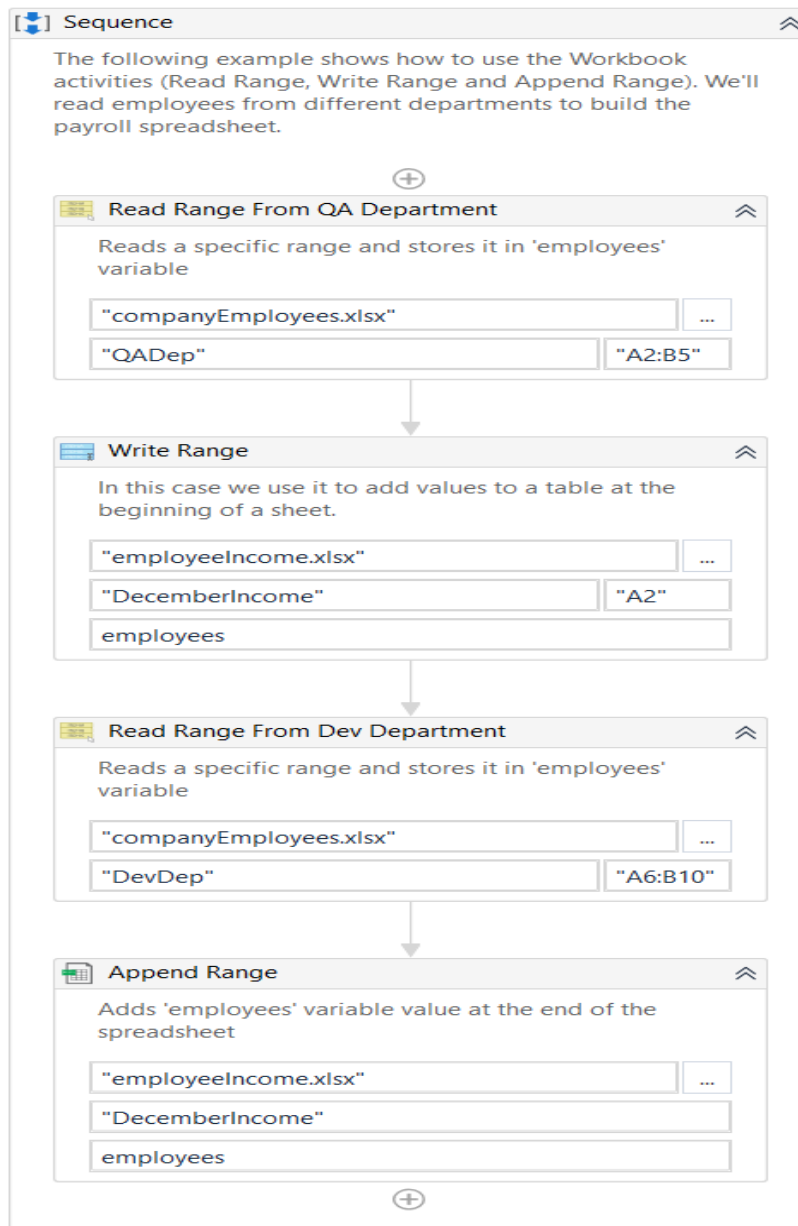
1. Open Studio and create a new Process.
2. Drag a Sequence container in the Workflow Designer.
 - Create the following variable:

Variable Name	Variable Type	Default Value

employees	DataTable	N/A
-----------	-----------	-----

3. Add a *Read Range* activity inside the Sequence container.
 - Add the expression "companyEmployees.xlsx" in the Workbook path field.
 - Add the value QADep in the Sheet field and set the cells range at "A1:B5".
4. Add a *Write Range* activity below the Read Range activity.
 - Add the expression "employeeIncome.xlsx" in the Workbook path field.
 - Add the value "DecemberIncome" in the Sheet field and set the cells range at "A2".
 - Add the variable employees in the Data Table field.
5. Add a *Read Range* activity below the Write Range activity.
 - Add the expression "companyEmployees.xlsx" in the Workbook path field.
 - Add the value DevDep in the Sheet field and set the cells range at "A6:B10".
6. Add an *Append Range* activity below the Read Range activity.
 - Add the expression "employeeIncome.xlsx" in the Workbook path field.
 - Add the value "DecemberIncome" in the Sheet field.
 - Add the variable employees in the Data Table field.

This is how the workflow should look:



7. Run the process. The activities copy data from one workbook and add it into a second workbook.

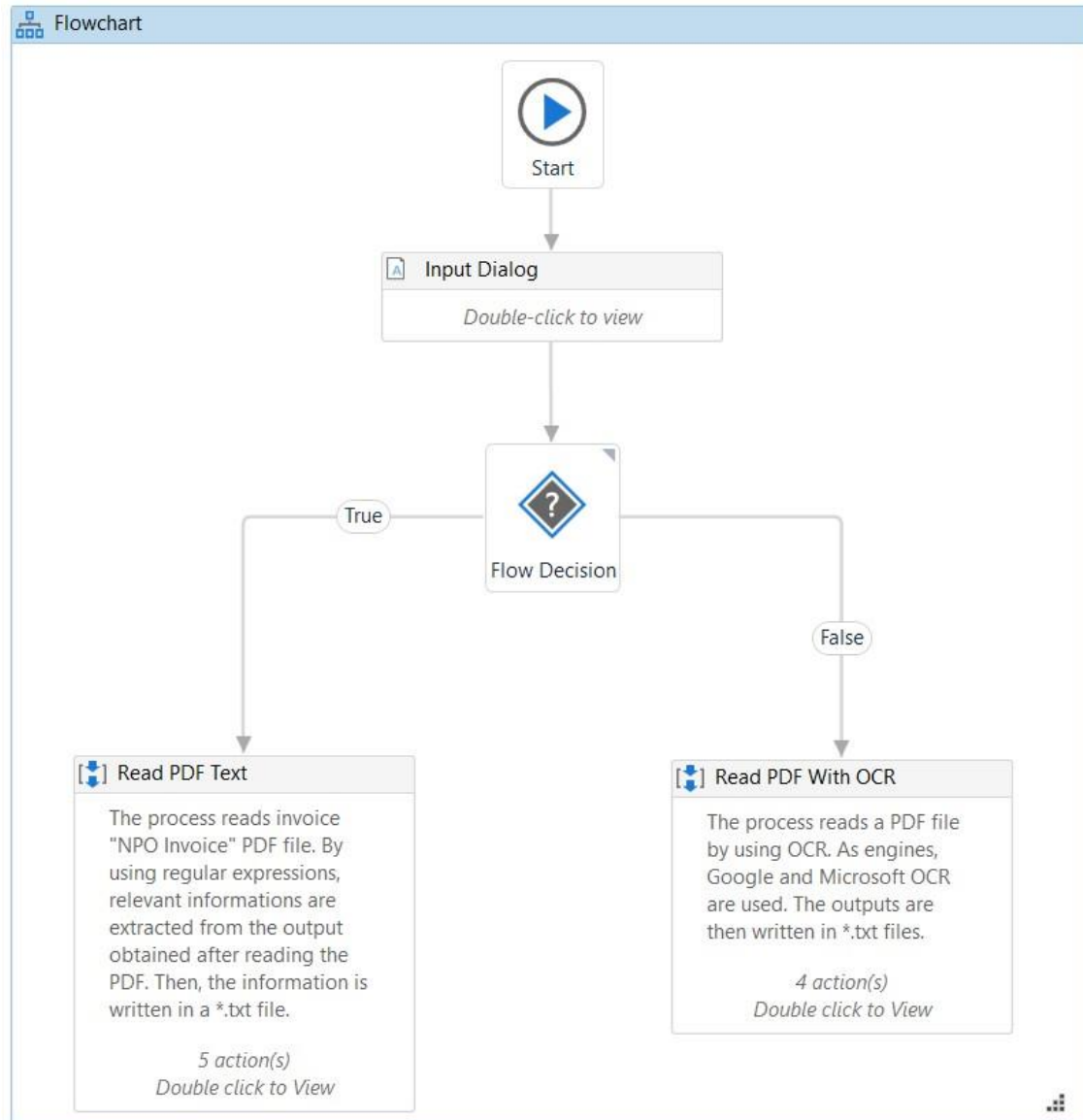
Extracting Data from PDF

Depending on your needs, you can use a simple activity that can recognize the characters, or use one with an OCR engine. The benefits of using an OCR engine are that the document reading can be applied even on scanned, signed, or handwritten documents.

The example below presents two situations of reading a .pdf file:

1. The first one explains how to read the .pdf file while using the *Read PDF Text* activity.
2. The second one explains how to read the .pdf file while using the *Read PDF With OCR* activity.

The main difference between the two scenarios is that the second one is also using OCR engines, meaning that the details of extracted information are more accurate than in the first case if the analyzed file is an image, scanned, or includes signed or handwritten fields. You can find both activities in the UiPath.PDF.Activities package.



Read PDF Text activity.

1. Install the UiPath PDF Activities package. This package provides the activities that you need to extract data from PDFs.

2. Create a new UiPath workflow.
3. Add the *Read PDF Text* activity to the workflow.
4. In the *Read PDF Text* activity properties, specify the path to the PDF file that you want to extract data from.
5. Add the activity that you want to use to extract the data from the PDF file. For example, if you want to extract data from a table in the PDF file, you would add the *Extract Data From Table* activity.
6. In the properties of the *data extraction* activity, specify the template or delimiters that you want to use to extract the data.
7. Run the workflow.

Read PDF With OCR activity:

1. To extract data from a PDF with OCR in UiPath, you can use the following steps:
2. Install the UiPath PDF Activities package.
3. Create a new UiPath workflow.
4. Add the Read PDF With OCR activity to the workflow.
5. In the Read PDF With OCR activity properties, specify the following:
 - **Path:** The path to the PDF file that you want to extract data from.
 - **OCR Engine:** The OCR engine that you want to use. UiPath supports a number of different OCR engines, such as Google OCR and Tesseract OCR.
6. Add the activity that you want to use to extract the data from the PDF file. For example, if you want to extract data from a table in the PDF file, you would add the Extract Data From Table activity.
7. In the properties of the data extraction activity, specify the template or delimiters that you want to use to extract the data.
8. Run the workflow.

Extracting a single piece of data

To extract a single piece of data from a PDF using anchors in UiPath, you can use the following steps:

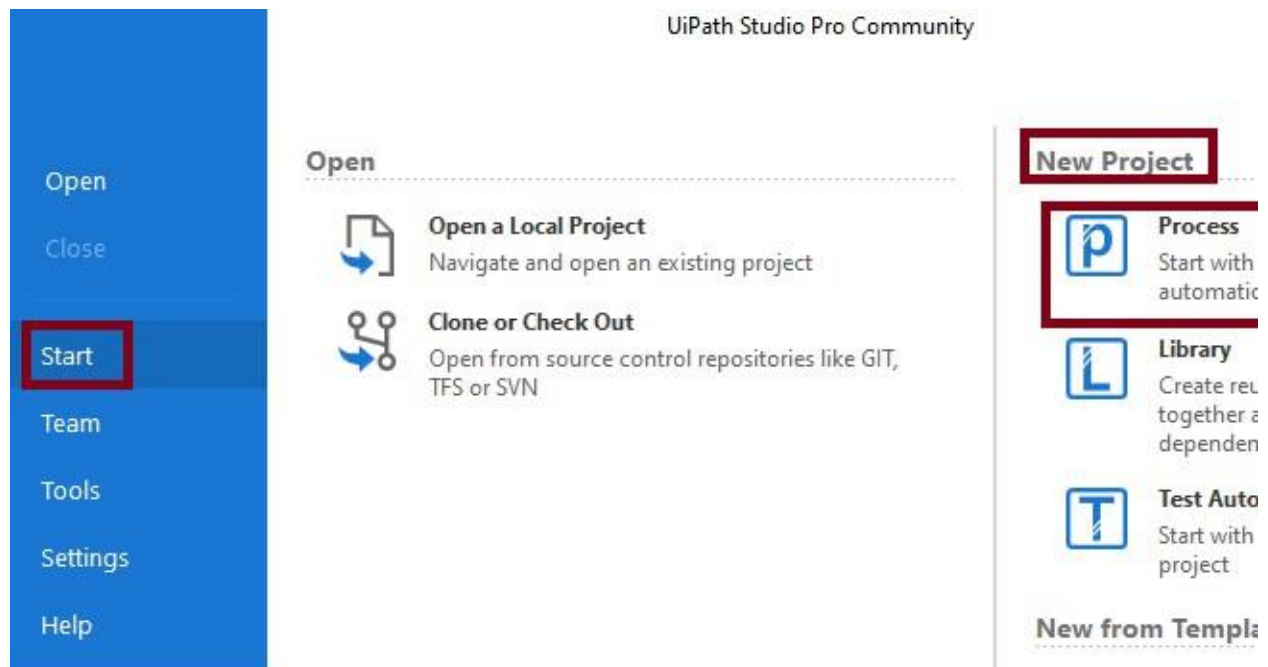
1. Install the UiPath PDF Activities package.
2. Create a new UiPath workflow.
3. Add the *Find Element* activity to the workflow.
4. In the *Find Element* activity properties, specify the following:
 - **Selector:** The anchor selector. This is a selector that uniquely identifies the element that you want to extract the data from.
 - **Relative:** True. This will indicate that the anchor selector is relative to the PDF file.
5. Add the *Get Text* activity to the workflow.
6. In the *Get Text* activity properties, specify the following:
 - **Element:** The output of the *Find Element* activity.
7. Connect the *Find Element* activity to the *Get Text* activity.
8. Run the workflow.

Once the workflow has finished running, the extracted data will be stored in the output variable of the Get Text activity.

Example:

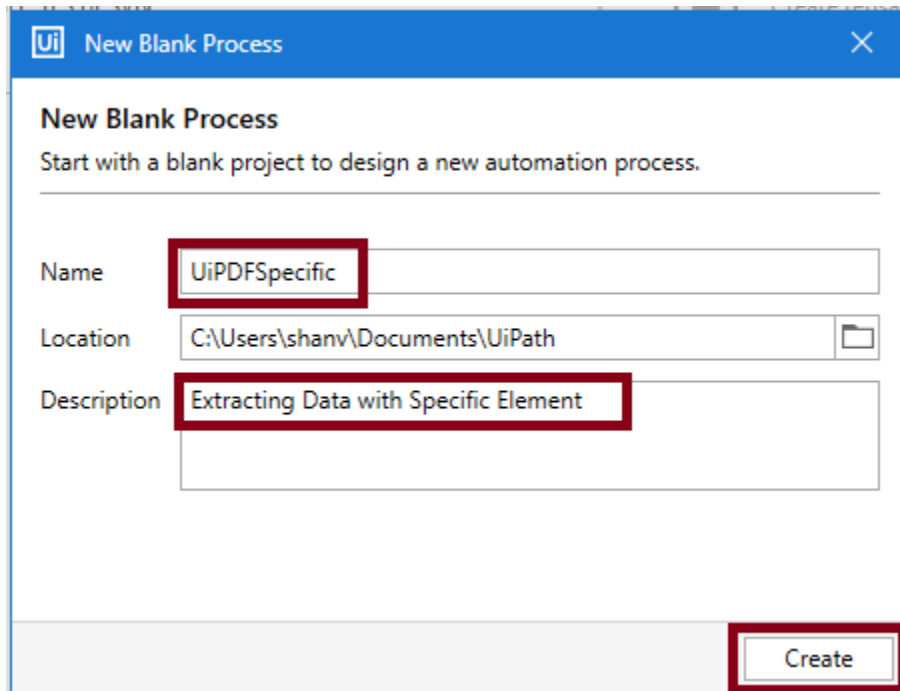
Step 1

Open UiPath Studio -> Start -> New Project-> Click Process



Step 2

Now, create new Blank Process, and name it UiPdfSpecific and give your description.



New Blank Process

Start with a blank project to design a new automation process.

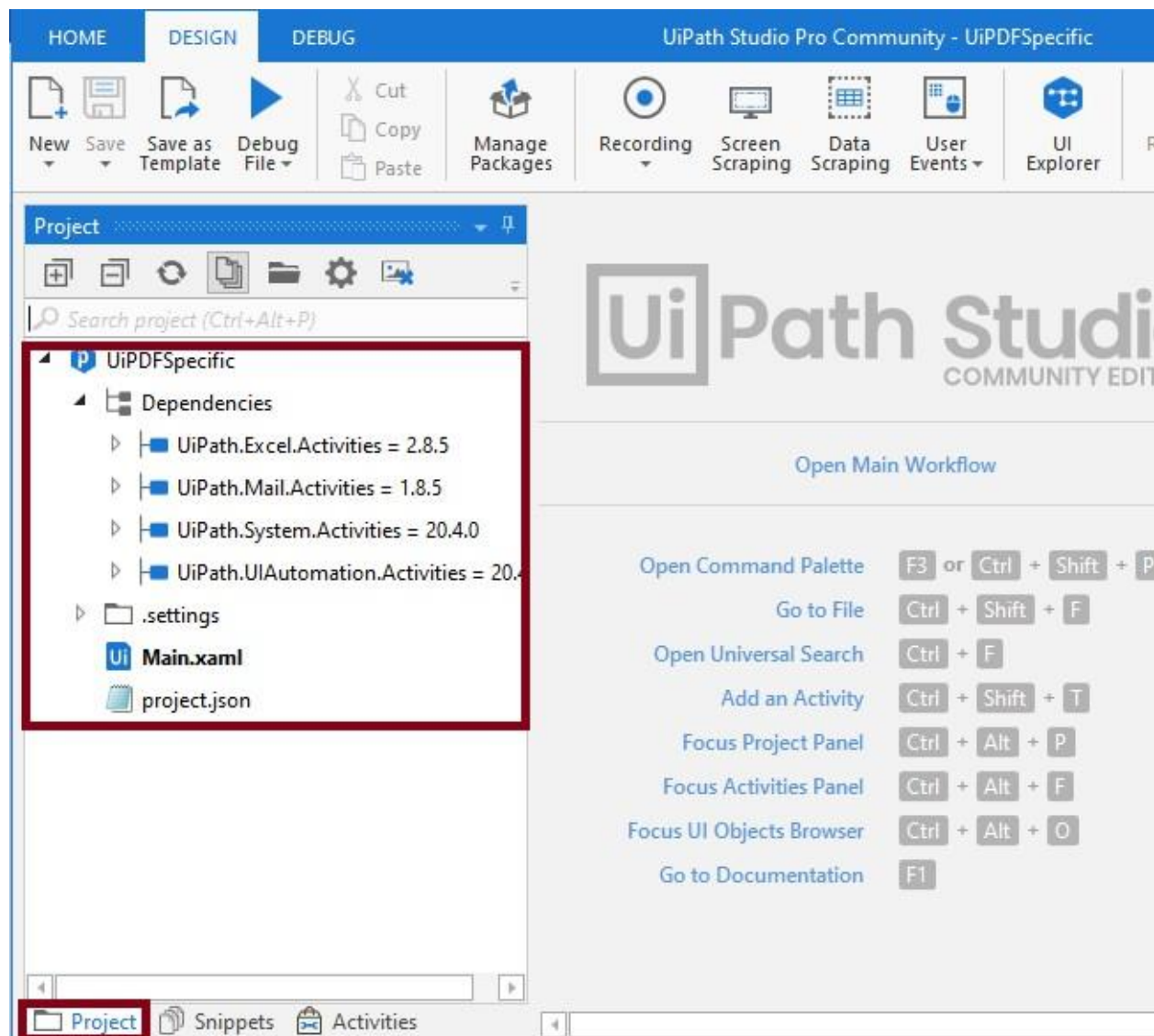
Name:

Location:

Description:

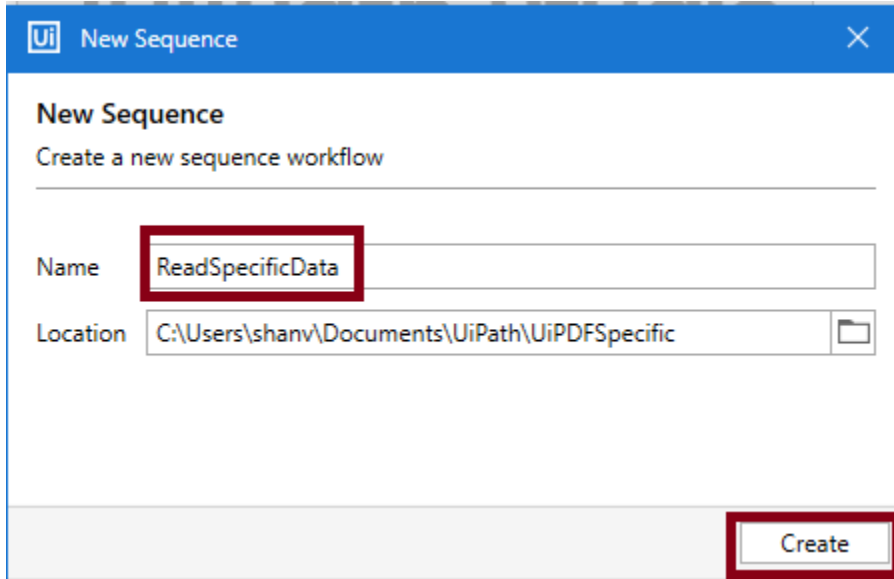
Step 3

After that UiPath studio creates the project UiPdfSpecific with supporting files.



Step 4

Next, for extracting the specific text in PDF document, create a new sequence workflow named ReadSpecificData:



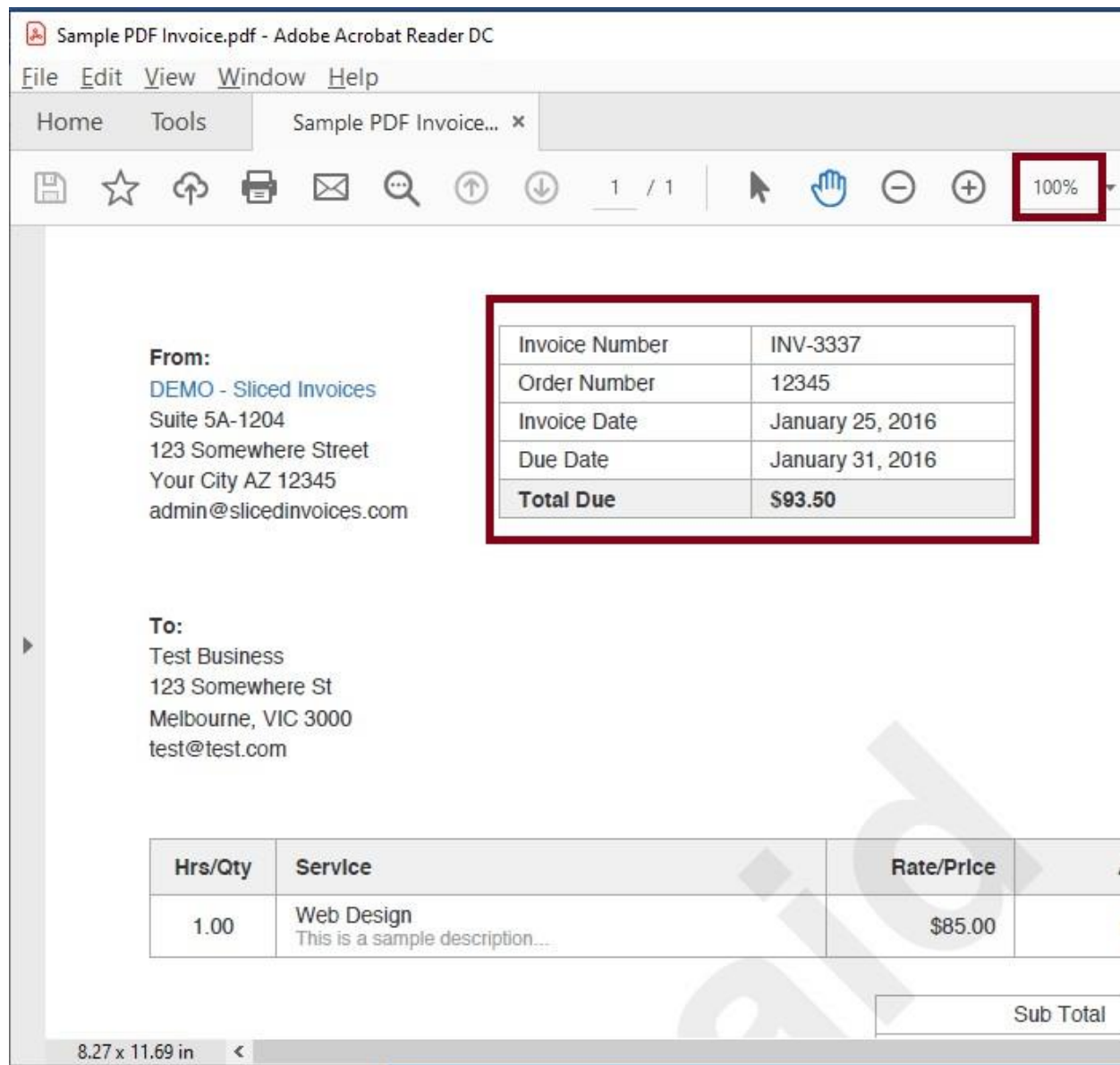
New Sequence

Create a new sequence workflow

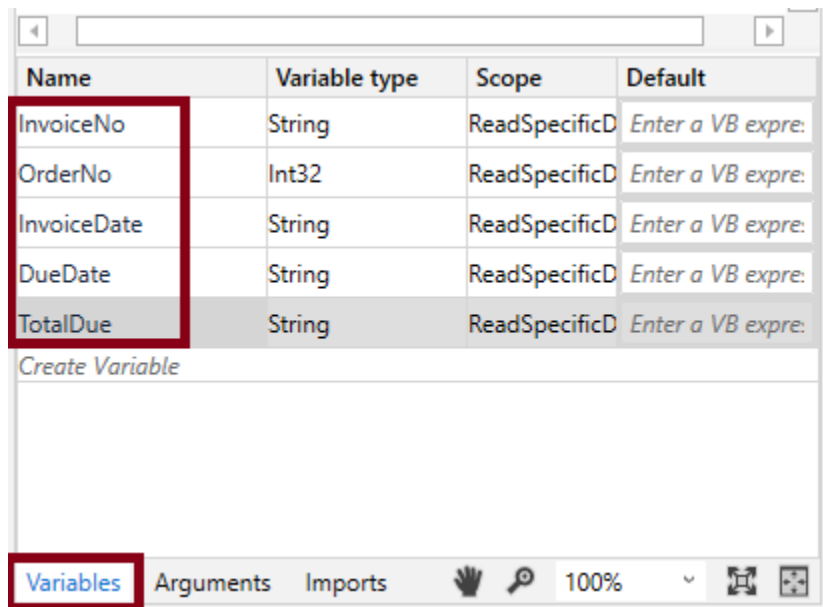
Name

Location

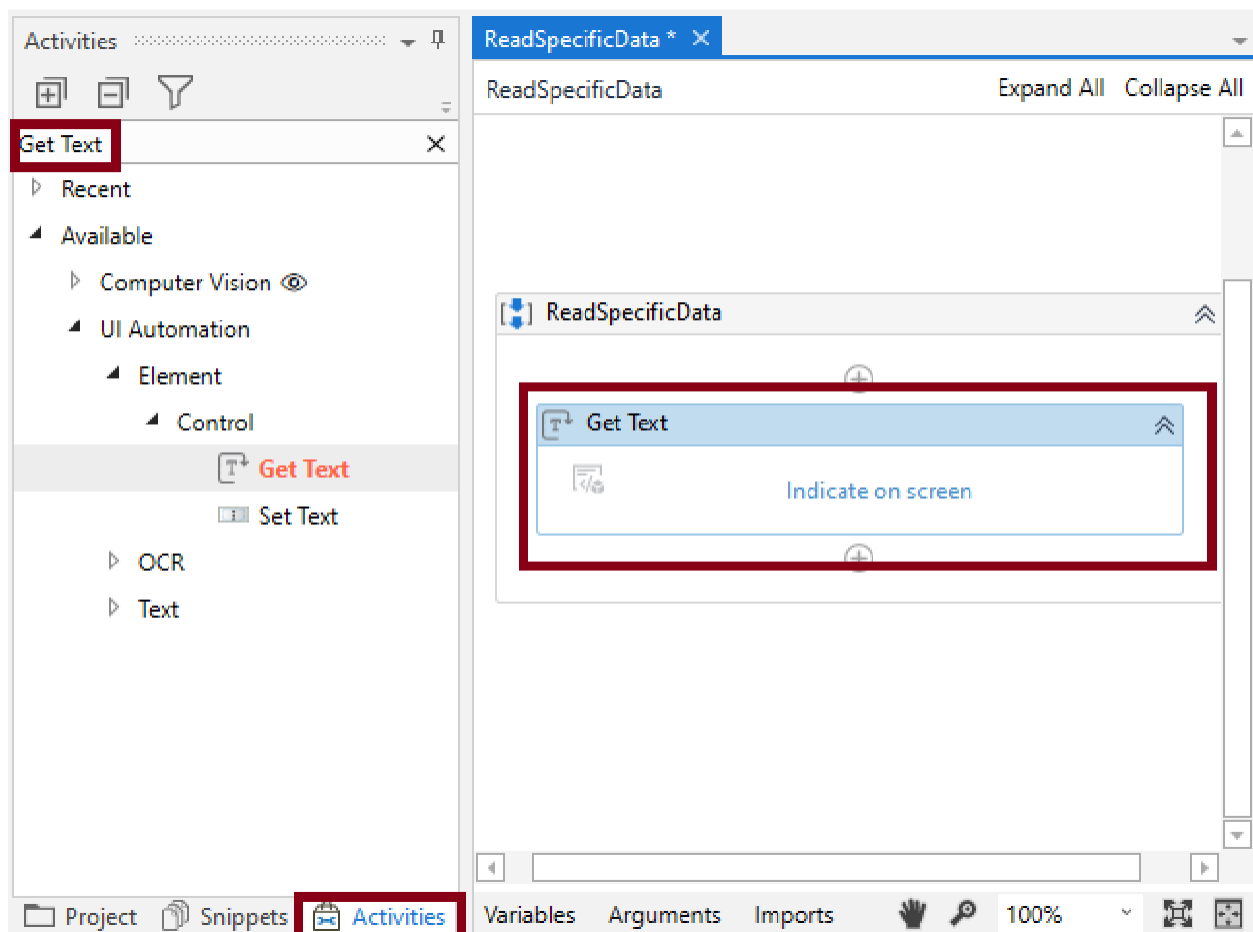
Sample PDF with Data,



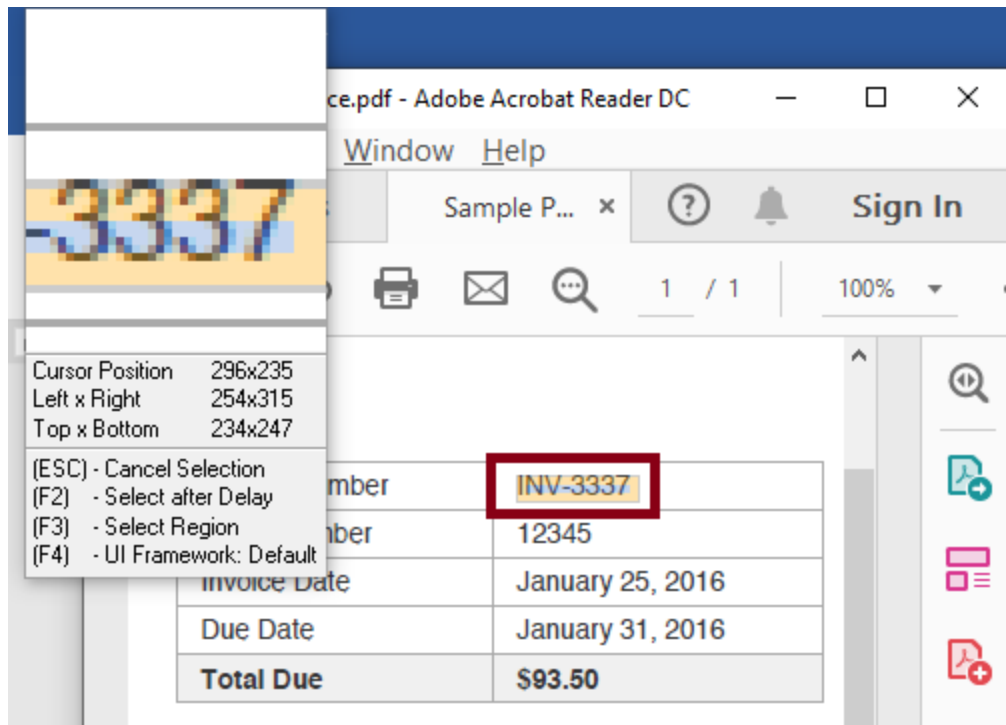
Create 5 variables, InvoiceNo as a String Data Type, OrderNo as a Int32 Data Type, InvoiceDate as a String Data Type, DueDate as a String Data Type, TotalDue as a String Data Type.



For getting the invoice number, Click Activities -> search GetText activity->Drag and drop into sequence,



Click the Indicate in screen, select the Invoice number:



Set the element value property of InvoiceNo:

The screenshot displays the UiPath IDE interface. The main workspace shows a 'ReadSpecificData' activity with a table containing two rows: 'umber' with value 'INV-3337' and 'mber' with value '12345'. The 'Properties' pane on the right shows the 'UiPath.Core.Activities.GetValue' activity. The 'Output' section is expanded, and the 'Value' property is set to 'InvoiceNo'. The 'Variables' pane at the bottom left shows a list of variables: 'InvoiceNo' (String), 'OrderNo' (Int32), 'InvoiceDate' (String), 'DueDate' (String), and 'TotalDue' (String). The 'Variables' tab is selected, and the 'InvoiceNo' variable is highlighted. The 'Properties' tab is also visible in the bottom right.

Name	Variable type	Scope	Default
InvoiceNo	String	ReadSpecificD	Enter a V
OrderNo	Int32	ReadSpecificD	Enter a V
InvoiceDate	String	ReadSpecificD	Enter a V
DueDate	String	ReadSpecificD	Enter a V
TotalDue	String	ReadSpecificD	Enter a V

Similarly, for getting the ordernumber, click Activities -> search GetText activity->Drag and drop in to sequence, click the Indicate in screen, select the Order Number, Set the element value property as OrderNo,

The screenshot displays the UiPath Studio interface. On the left, the 'Variables' pane shows a table with the following data:

Name	Variable type	Scope	Default
InvoiceNo	String	ReadSpecificD	Enter a V
OrderNo	Int32	ReadSpecificD	Enter a V
InvoiceDate	String	ReadSpecificD	Enter a V
DueDate	String	ReadSpecificD	Enter a V
TotalDue	String	ReadSpecificD	Enter a V

The 'OrderNo' variable is highlighted with a red box. On the right, the 'Properties' pane shows the 'Value' property of the 'Get Text' activity set to 'OrderNo', also highlighted with a red box. The 'Get Text' activity is highlighted in the workflow, and its target is set to 'text 12345'.

Similarly, for getting the invoice date, click Activities -> search GetText activity->Drag and drop in to sequence, click the Indicate in screen, select the Invoice date, set the element value property as InvoiceDate,

The screenshot displays the UiPath Studio interface. On the left, the 'Variables' pane shows a table of variables:

Name	Variable type	Scope	Default
InvoiceNo	String	ReadSpecificD	Enter a V
OrderNo	Int32	ReadSpecificD	Enter a V
InvoiceDate	String	ReadSpecificD	Enter a V
DueDate	String	ReadSpecificD	Enter a V
TotalDue	String	ReadSpecificD	Enter a V

The 'Get Text' activity in the workflow has its 'Value' output property set to 'InvoiceDate'. The 'Properties' pane on the right shows the 'Value' property set to 'InvoiceDate'.

Similarly, for getting the due date, click Activities -> search GetText activity->Drag and drop in to sequence, click the Indicator on screen, select the Due Date, set the element value property as DueDate,

The screenshot displays the UiPath Studio interface. The main workspace shows a sequence of activities. A 'Get Text' activity is selected, with its 'Target' property set to a date field in a table. The table contains the following data:

Name	Variable type	Scope	Default
OrderNo	Int32	ReadSpecificD	Enter a V
InvoiceDate	String	ReadSpecificD	Enter a V
DueDate	String	ReadSpecificD	Enter a V
TotalDue	String	ReadSpecificD	Enter a V

The 'DueDate' variable is highlighted in red. The 'Properties' pane on the right shows the 'UiPath.Core.Activities.GetValue' activity. The 'Output' section has a 'Value' property set to 'DueDate', which is also highlighted in red. The 'Variables' tab at the bottom is selected, and the 'DueDate' variable is highlighted in red.

Similarly, for getting the Total Due, click Activities -> search GetText activity->Drag and drop in to sequence, click the Indicate in screen, select the Total Due, set the element value property as Total Due,

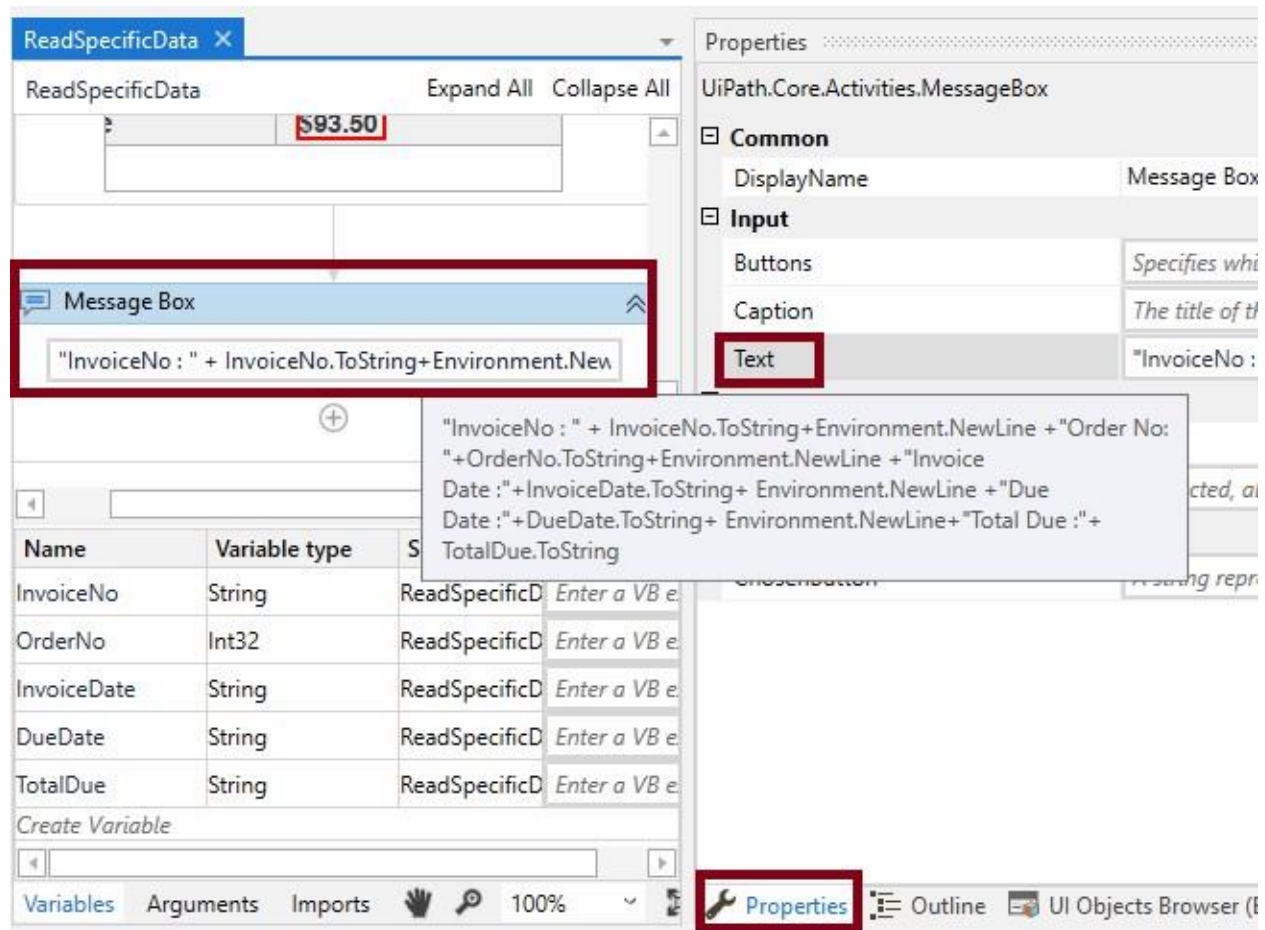
The screenshot shows the UiPath Studio interface. The main workspace displays the 'ReadSpecificData' activity, which is expanded to show a table of data. The table has two columns: 'Name' and 'Value'. The 'Name' column contains 'January 31, 2016' and '\$93.50'. The 'Value' column contains 'January 31, 2016' and '\$93.50'. The 'TotalDue' variable is highlighted in red in the table. Below the table, the 'Variables' tab is selected, showing a list of variables: InvoiceNo, OrderNo, InvoiceDate, DueDate, and TotalDue. The 'TotalDue' variable is highlighted in red. The 'Properties' pane on the right shows the 'UiPath.Core.Activities.GetValue' activity. The 'Common' section shows 'ContinueOnError' set to 'Specifies to' and 'Display Name' set to 'Get Text 'te''. The 'Input' section shows 'Target' set to 'Target'. The 'Misc' section shows 'Private' set to 'False'. The 'Output' section shows 'Value' set to 'TotalDue'.

Name	Variable type	Scope	Default
InvoiceNo	String	ReadSpecificD	Enter a V
OrderNo	Int32	ReadSpecificD	Enter a V
InvoiceDate	String	ReadSpecificD	Enter a V
DueDate	String	ReadSpecificD	Enter a V
TotalDue	String	ReadSpecificD	Enter a V

Click Activities -> search Message Box activity->Drag and drop into sequence and set the text as:

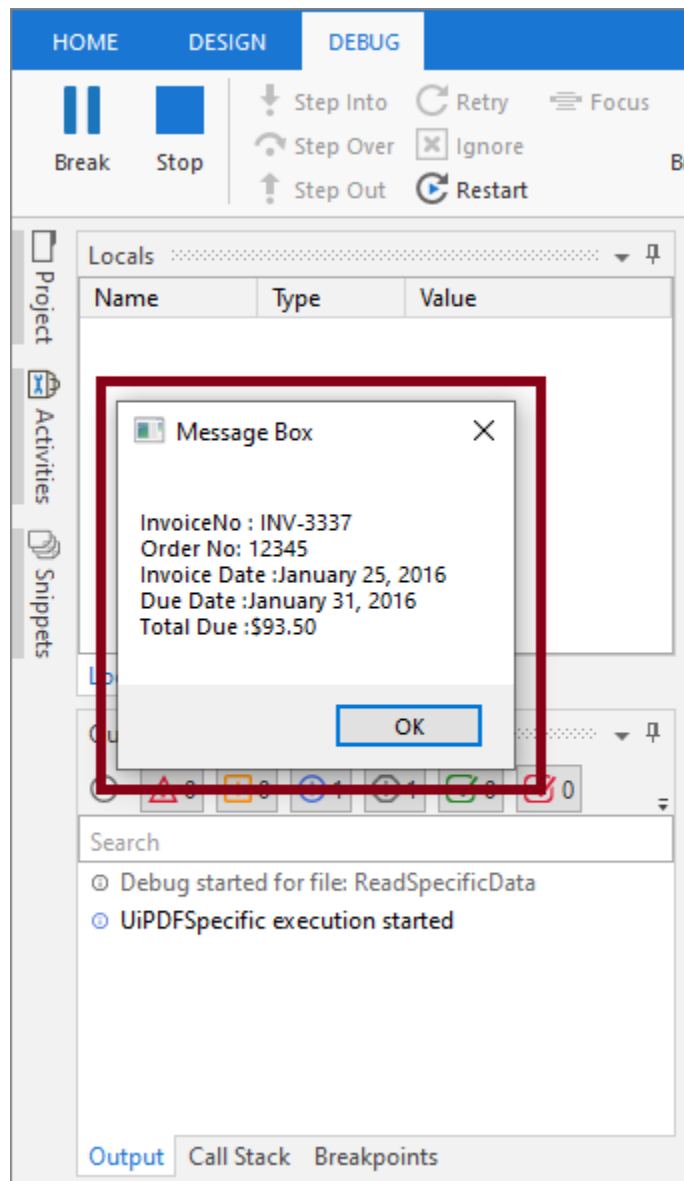
```
"InvoiceNo : " + InvoiceNo.ToString+Environment.NewLine + "Order No:
"+OrderNo.ToString+Environment.NewLine + "Invoice Date :"+InvoiceDate.ToString+
Environment.NewLine + "Due Date :"+DueDate.ToString+ Environment.NewLine+"Total Due
:"+ TotalDue.ToString
```





Step 5

For running your project, Select debug file -> Run. The output of the UiPdfSpecific project is,



Anchors

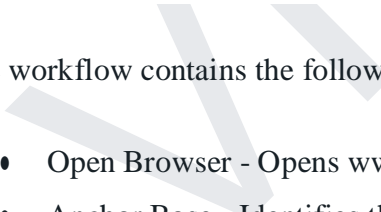
UiPath.Core.Activities.AnchorBase

A container that searches for a UI element by using other UI elements as anchors. This should be used when a reliable selector is not available.

An anchor in UiPath is a UI element that is used to uniquely identify another UI element. This can be useful for automating tasks that involve interacting with UI elements that are not unique

or that may change their position over time.

Anchors are used when you want to interact with an element that has an unstable selector. In this example, the workflow writes a particular text string in a field which changes its position every time the www.rpachallenge.com web page loads.



The workflow contains the following activities:

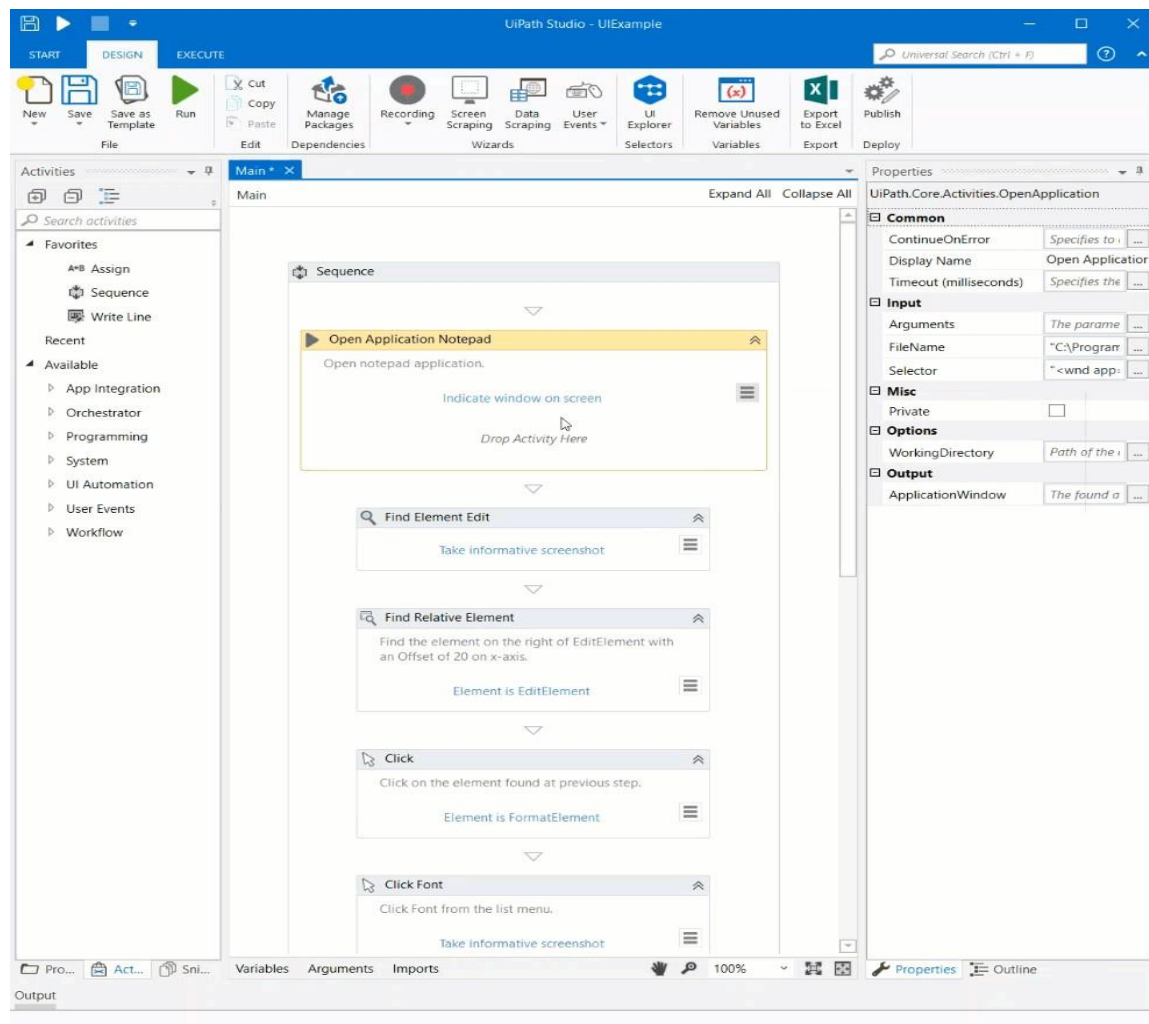
- Open Browser - Opens www.rpachallenge.com in Internet Explorer.
- Anchor Base - Identifies the target field and writes the sample text:
 - Left side - The Find Element activity identifies the First Name field.
 - Right side - The Type Into activity writes "Example" in the First Name field.

For example, you could use the title bar of a window or the header of a table as an anchor.

Once you have identified the anchor, you can use it to identify the other UI element that you

want to interact with. To do this, you can use the Find Element activity and specify the anchor selector in the Relative property of the activity.

Here is an example of how to use an anchor to automate a task in UiPath:



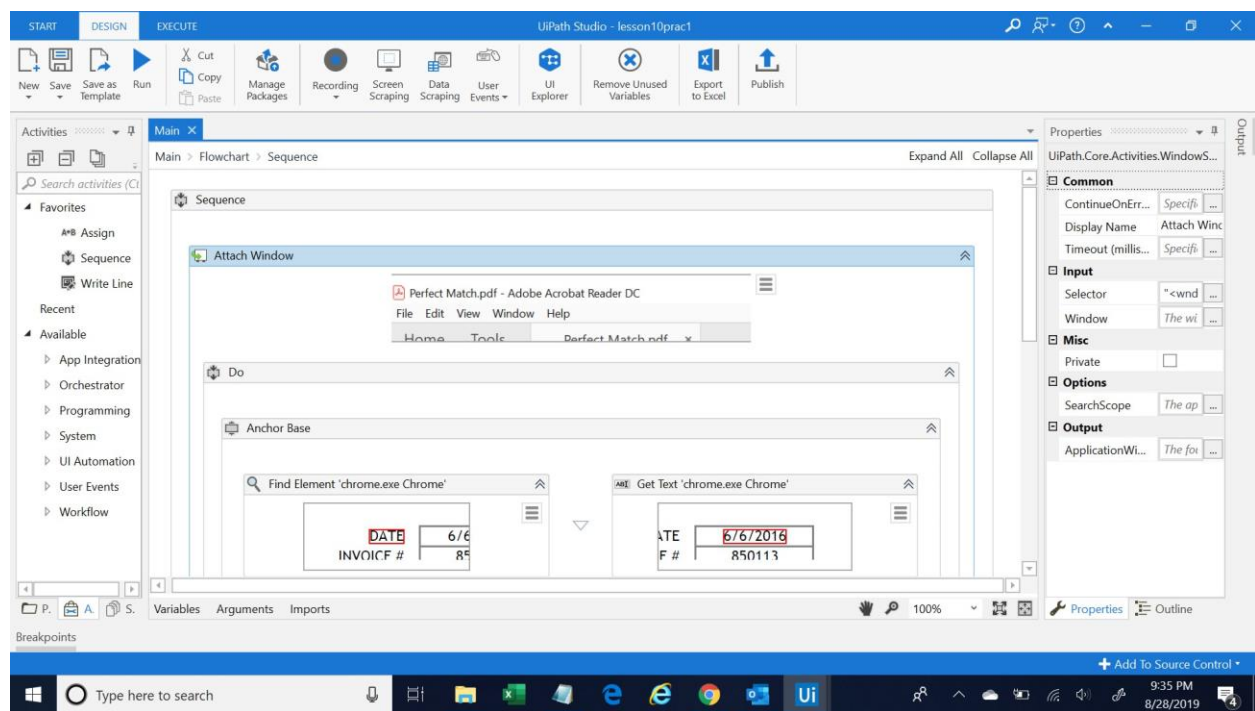
1. Open the window that contains the UI elements that you want to interact with.
2. Add a *Find Element* activity to the workflow and specify the anchor selector in the *Relative property* of the activity.
3. Add the activity that you want to use to interact with the other UI element. For example, if you want to click on the other UI element, you would add a *Click* activity.
4. Connect the *Find Element* activity to the activity that you want to use to interact with the other UI element.

5. Run the workflow.

Using anchors in PDF

Anchors are UI elements that are used to uniquely identify another UI element. They can be used in RPA to automate tasks that involve interacting with UI elements in PDFs, such as extracting data or filling out forms.

how to use anchors in PDF in UiPath:



(or)

- Extract the customer name from a PDF invoice:

Anchor selector: `//*[text()='Customer Name:']/following-sibling::*`

- Extract the total amount from a PDF invoice:

Anchor selector: `//*[text()='Total Amount:']/following-sibling::*`

- Extract the product name from a PDF product catalog:

Anchor selector: `//*[text()='Product Name:']/following-sibling::*`

You can also use anchors to extract data from tables in PDFs. To do this, you can use the Extract Data From Table activity and specify the anchor selector for the table in the Table property of the activity.

- Install the UiPath PDF Activities package.
- Create a new UiPath workflow.
- Add the Find Element activity to the workflow.
- In the Find Element activity properties, specify the following:
- Selector: The anchor selector. This is a selector that uniquely identifies the element that you want to interact with.
- Relative: True. This will indicate that the anchor selector is relative to the PDF file.
- Add the Extract Text activity to the workflow.
- Connect the Find Element activity to the Extract Text activity.
- Repeat steps 3-6 for each piece of data that you want to scrape from the PDF.
- Add an activity to save the scraped data to a database, spreadsheet, or other application.
- Run the workflow.

Here is an example of a UiPath workflow that scrapes the customer name, product name, and price from a PDF invoice:

```
// Import the necessary UiPath activities.
```

```
using UiPath.Core.Activities;
```

```
using UiPath.PDF.Activities;
```

```
// Create a new workflow.
```

```
var workflow = new Workflow();
```

```
// Add the Find Element activity to scrape the customer name.
```

```
workflow.Activities.Add(new FindElement()
```

```
{
```

```
    Selector = @"//*[@text()='Customer Name:']/following-sibling::*",
```

```
    Relative = true
```

```
});
```

```
// Add the Extract Text activity to extract the customer name.
```

```
workflow.Activities.Add(new ExtractText()
{
    Element = new FindElementOutput()
    {
        Selector = @"//*[@text()='Customer Name:']/following-sibling::*"
    }
});
```

```
// Add the Find Element activity to scrape the product name.
```

```
workflow.Activities.Add(new FindElement()
{
    Selector = @"//*[@text()='Product Name:']/following-sibling::*",
    Relative = true
});
```

```
// Add the Extract Text activity to extract the product name.
```

```
workflow.Activities.Add(new ExtractText()
{
    Element = new FindElementOutput()
    {
        Selector = @"//*[@text()='Product Name:']/following-sibling::*"
    }
});
```

```
    }  
});  
  
// Add the Find Element activity to scrape the price.  
workflow.Activities.Add(new FindElement()  
{  
    Selector = @"//*[@text()='Price:']/following-sibling::*",  
    Relative = true  
});  
  
// Add the Extract Text activity to extract the price.  
workflow.Activities.Add(new ExtractText()  
{  
    Element = new FindElementOutput()  
    {  
        Selector = @"//*[@text()='Price:']/following-sibling::*"  
    }  
});  
  
// Save the scraped data to a database, spreadsheet, or other application.  
  
// Run the workflow.  
workflow.Start();
```

This workflow will scrape the customer name, product name, and price from the PDF invoice and store the scraped data in the output variables of the Extract Text activities. You can then use the scraped data in other activities in your workflow, or you can save the scraped data to a database, spreadsheet, or other application.

Using anchors to scrape data from PDFs in UiPath is a powerful way to extract data accurately and efficiently.

