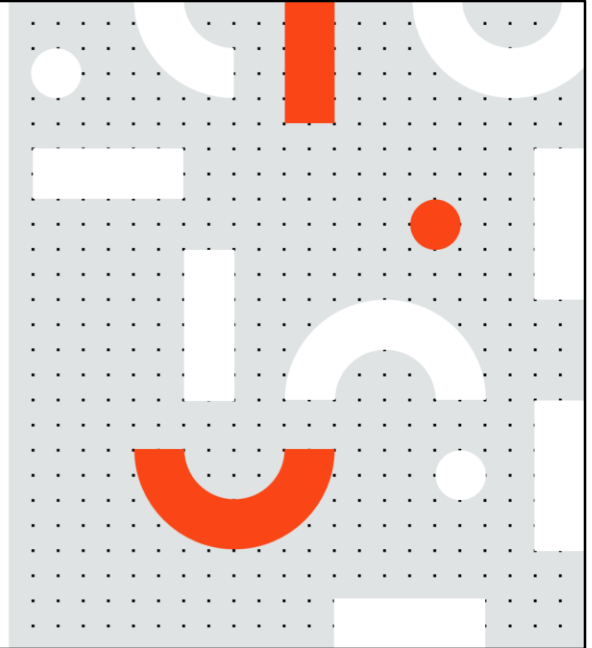


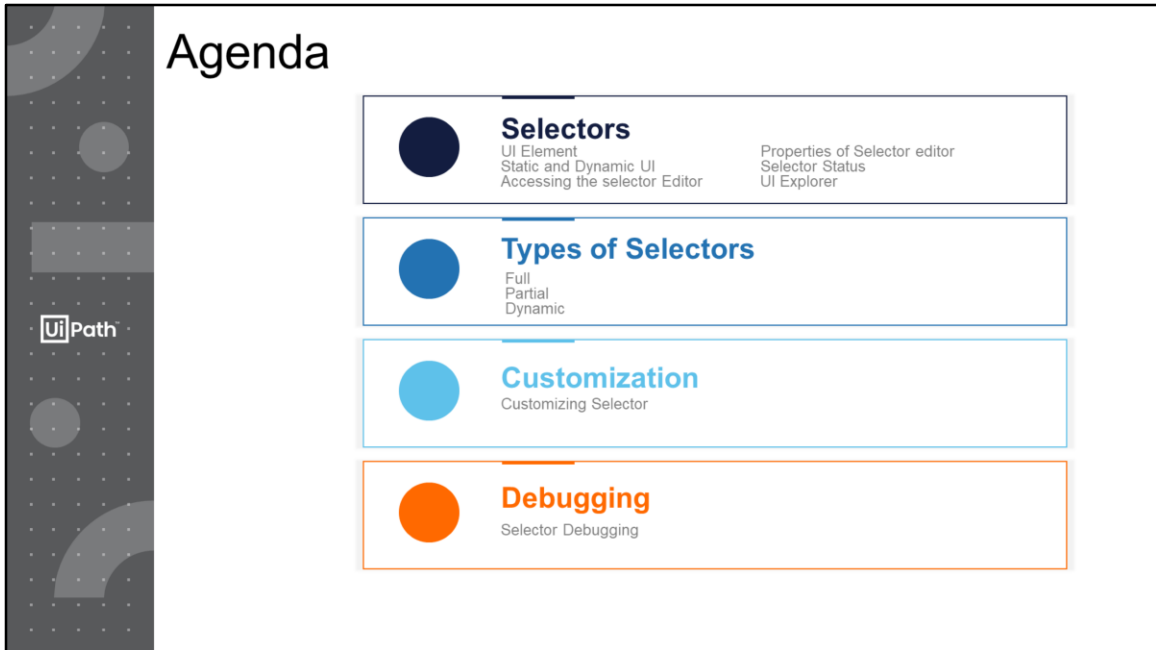
RPA Design & Development v1.1

Lesson 10 Selectors



After completing this module, students should be able to:

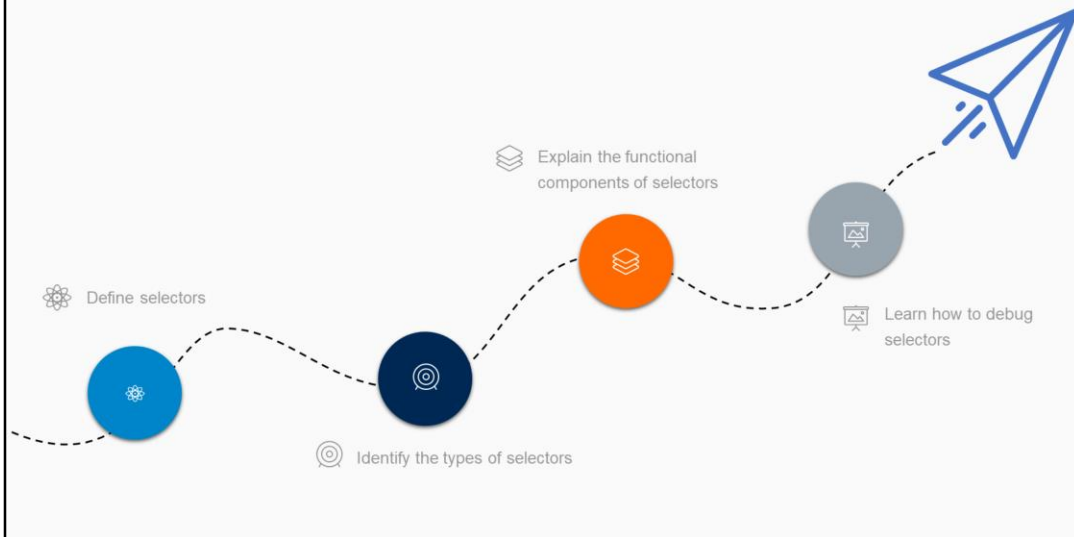
- Understand and apply various functionalities of Selectors



In this lesson, we will cover the following topics:

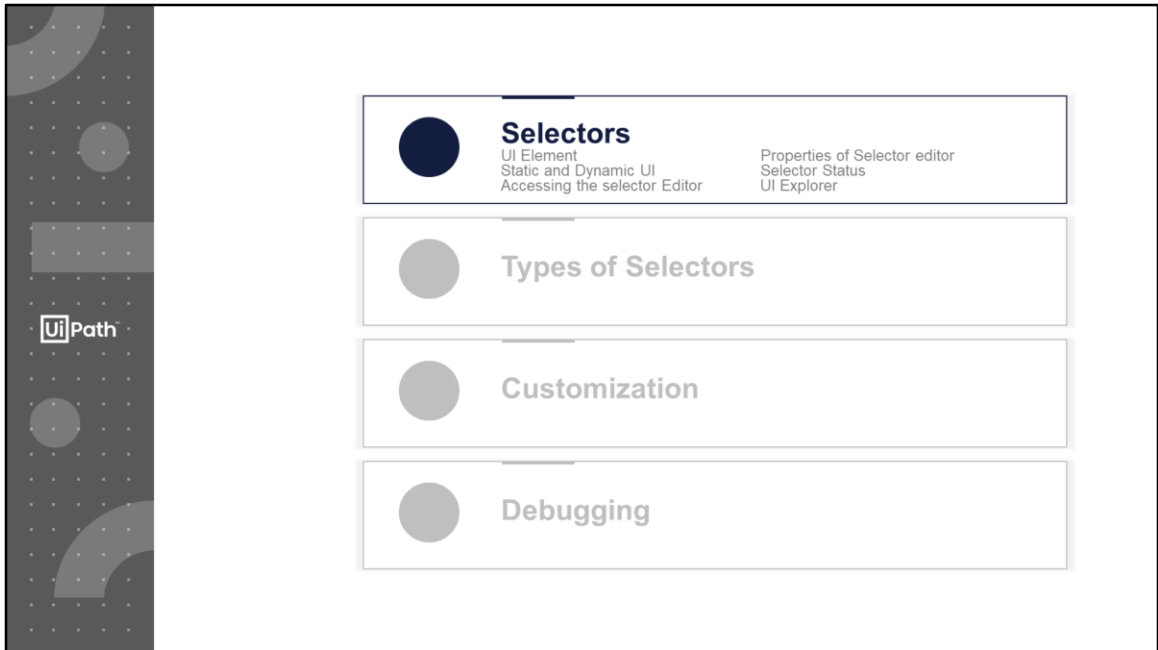
- Selectors
 - UI Element
 - Static & Dynamic UI
 - Accessing the selector Editor
 - Properties of Selector Editor
 - Selector Status
 - UI Explorer
 - Properties UI Explorer
 - Property Explorer
- Types of Selectors
 - Full Selector
 - Partial Selector
 - Dynamic Selector
- Customization
 - Customizing Selector
- Debugging
 - Selector Debugging

Learning Objectives



After completing this lesson, you will be able to:

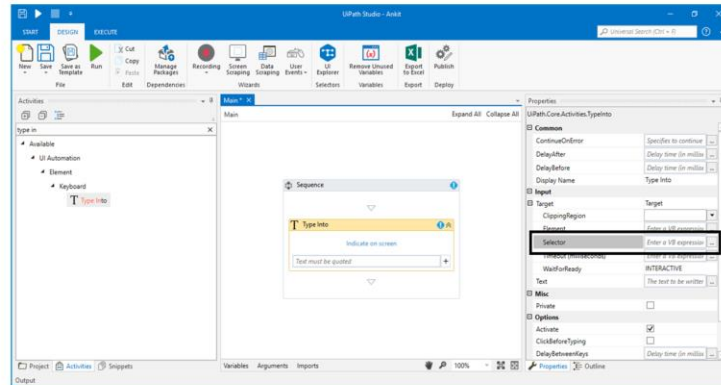
- Define selectors
- Identify the types of selectors
- Explain the functional components of selectors
- Learn how to debug selectors



In this topic, we will learn about selectors.

Selectors

Selectors are a fundamental part of UiPath Studio that are used to recognize the objects on the project screen.



In the UiPath Studio to start or execute a specific task or process in the UI, you require the buttons, windows, drop down list and advanced features which is the combination of **Selectors**. It is used to recognize the objects on the project screen. It contains the program code in XML format. In the UiPath studio, you can start selector activities by UI Explorer activity.

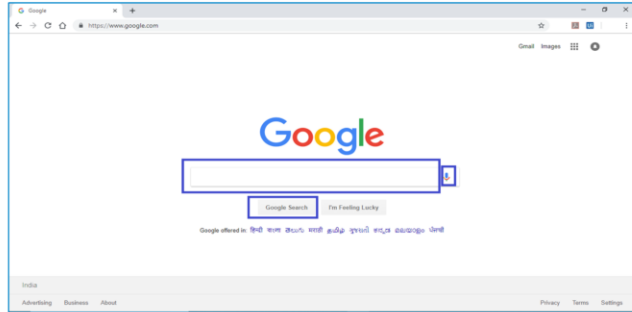
Selectors are the address of an element in the UiPath Studio. When we have to automate a specific task or activity action, we have to communicate with various UI Elements. The selector can be looked upon as a specific identifier to find a particular UI element amongst multiple applications. Lets first understand what is a UI Element.

UI Element

UI element refers to all graphical user interface pieces that construct an application.

Example of UI elements:

- Search bar
- Search button
- Audio button

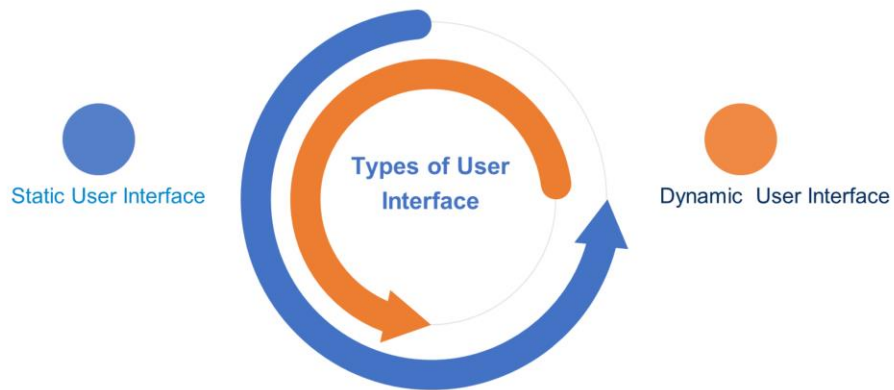


UI Element refers to all graphical user interface pieces that construct an application. Many times the selectors have generated automatically in UiPath Studios during the development time of Activity where you do not need to specify further input. For example, when you open www.google.com, you see the following items:

- A search bar(Text Field)
- Search Button
- A mike shaped image for audio search.

User Interface and its Types

The UI elements are embedded in **User Interface** that acts like a container holding all the pieces that construct an application.



To act on these UI elements, for example, by clicking, we need to find that element. Selectors are the way to find and identify these elements. In short, selectors are the address of the UI Elements. In order to automate specific actions in the UI, we are required to interact with various windows, buttons, drop-down lists and many other complex elements.

All applications are built by using a series of containers, which are embedded one inside the other, for example, the Notepad Application which has the caption and controls section, the menu section, the text input section, etc. These containers are known as the user interface. It builds an HTML web page or any other application in a very similar way.

Static and Dynamic User Interface

Static Interface Scenario

Address

Phone Number

Last Name

Role in Company

First Name

Email

Company Name

Submit

The UI element entitled "Address" will always be found at this exact pixel coordinate in the part of the web page. If the layout does not change, the selector will remain valid through its operations.

Dynamic Interface Scenario

Company Name

First Name

Address

Email

Phone Number

Role in Company

Last Name

Submit

In this example, the layout has changed although it contains the same UI elements. The selector from the first example would become invalid as the pixel positioning of the "Address" element has changed.

When our application or webpage has elements at a fixed address, it is called static interface.

When our application or webpage does not have elements at a fixed address, it is called dynamic interface. During the automated processes, some RPA products rely on the screen positioning of individual UI elements and pixel coordinates, but these methods are not always as dependable as they can change.

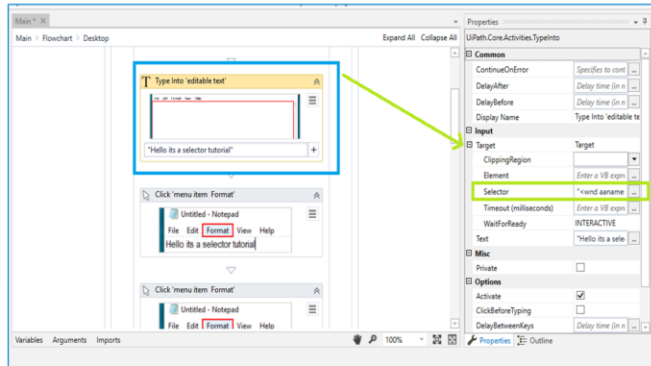
To overcome issues like these, UiPath Studio uses the tool that we know as selectors. These can store the attributes and characteristics of a GUI element along with all its parents in the shape of an XML fragment. Most of the time, selectors are automatically generated by UiPath Studio and you do not require further input from the user. This is especially so if the app that you are trying to automate is a static UI.

Selector Editor

The **selector editor** window enables us to see the automatically generated selectors and edit their attributes.

To access this window:

- Go to the Workflow Designer panel
- Click on the activity whose selector you want to edit
- In the properties panel, click on the option "TARGET"



To access this window:

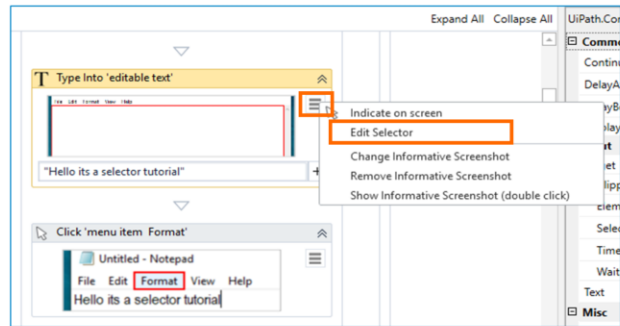
1. Go to the Workflow Designer panel.
 2. Click on the activity whose selector you want to edit/access.
 3. In the properties panel find the option "TARGET" and expand it.
 4. Click on the (...) button (on the right-hand side of the selector editing field).
- It would display the Expression Editor

Selector Editor (Contd.)

The [selector editor](#) window can also be accessed by:

To access this window:

- Click the hamburger button next to the selector field in the properties panel
- Click on "Edit Selector "

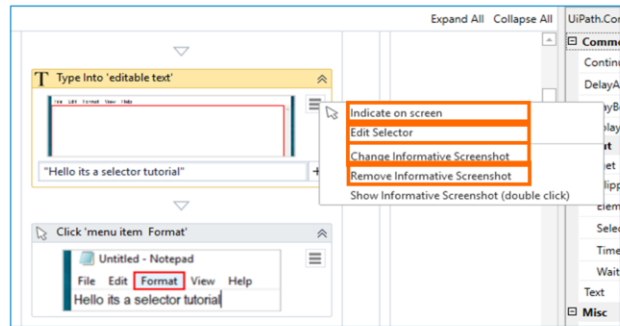


You can use this by using the selector field button. Once the drop-down appears you can click on "edit selector " for editing the selector.

Properties of the Selector Editor

On clicking the hamburger button, the following options are displayed

- Indicate on Screen
- Edit Selector
- Change Informative Screenshot
- Remove Informative Screenshot
- Show Informative Screenshot (Double Click)



Properties/ Actions we get by clicking the hamburger button:

When we click on the hamburger button, the following options appear in a drop-down list:

Indicate on Screen: Indicate on screen is used to ease and simplify the automation where the target element is changed, and the selectors are automatically created.

Edit Selector: It is used to edit the selectors already created. If the selectors do not work due to any reason, they can be edited by using the “edit selector” option.





Change Informative Screenshot: When we select any element, a sample screenshot is created, in case we want to change the screenshot, this option can be used.

Remove Informative Screenshot: An auto-generated screenshot can be deleted by using this option.

Show Informative Screenshot (Double Click): You can double click on the process to show the informative screenshot or image.

Selector status

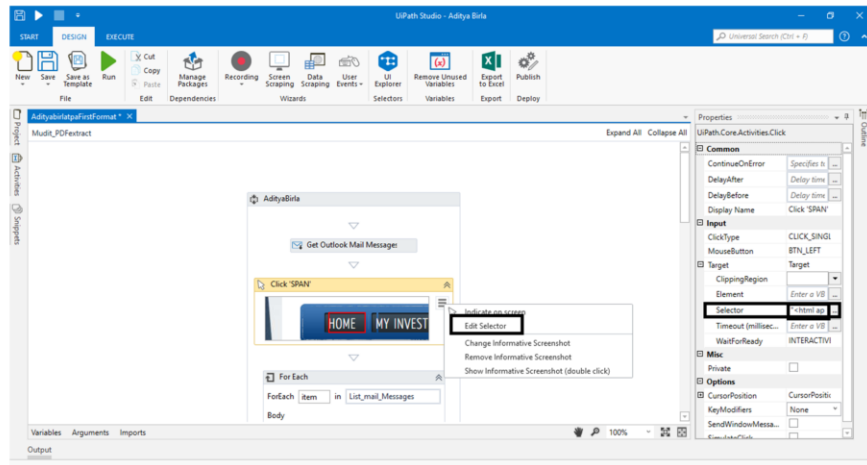
The **selector status** can be viewed in the selector editor window.

Option	Description
Validate	<p>The button shows the status of the selector by checking the validity of the selector definition and the visibility of the target element on the screen.</p> <p>The Validate button has three states:</p> <ul style="list-style-type: none"> Validate Selector is being validated Validate Valid selector Validate Invalid selector Validate Modified selector, revalidate <p>The button is correlated with UI Explorer validation states.</p>
Indicate Element	<p>Indicate a new UI element to replace the previous one.</p>
Repair	<p>Enables you to re-indicate the same target UI element and repair the selector. This operation does not completely replace the previous selector. The button is available only when the selector is invalid.</p>
Highlight	<p>Brings the target element in the foreground. The highlight stays on until the option is disabled with a click. The button is enabled only if the selector is valid.</p>
Edit Attributes	<p>Contains all the application components needed to identify the target application (a window, a button etc.). This section is editable.</p>
Edit Selector	<p>Holds the actual selector. This section is editable.</p>
Open in UI Explorer	<p>Launches the UI Explorer. The option is enabled only for valid selectors.</p>

In the selector editor window, you can see the selector's status where the validity status can be either green, red, yellow or grey. If the selector is valid then it will appear as green. If the selector is validated then it shows in grey color but in the invalid selector case, it will show in red color. In case the change is made in selector after validations, the color appears as yellow.

Ui Explorer

The **Ui Explorer** is used to customize the selector.

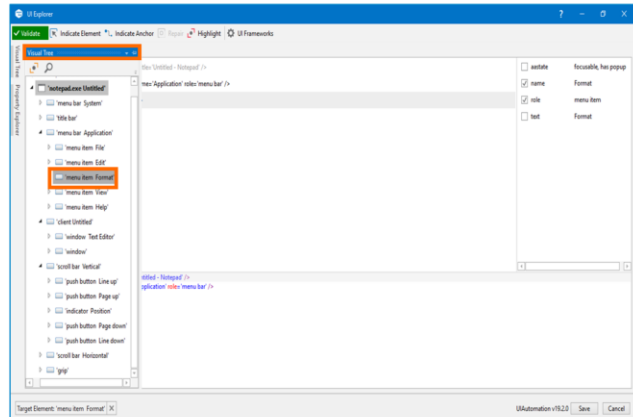


UI Explorer is used to customizing the selector. It is an advanced version of the selector which provides flexibility to the selector. UIExplorer is a more advanced version of the selector. Also, It is a tool that gives us the flexibility to customize the selector. We can Access UIExplorer from the Design panel, or from the edit selector option by:

1. Click “Edit Selector”
2. The Selector editor window will appear.
3. Find the “open in UI explorer” option and click on it.

Properties of Ui Explorer

- The Visual Tree is located on the left-hand side of UI Explorer
- To change the format of text written in a notepad, click on "menu item Format" button



We have the Visual Tree located on the left-hand side of UI Explorer.

Visual Tree: is defined as a “list of containers from the parent container to the Target UI element.”

For Example:

We can change or modify the text written format with the help of notepad.

After this, we can associate with our process in the UI element by clicking the format button. So here if we define this interaction in a tree it will be like:

Container 1: notepad

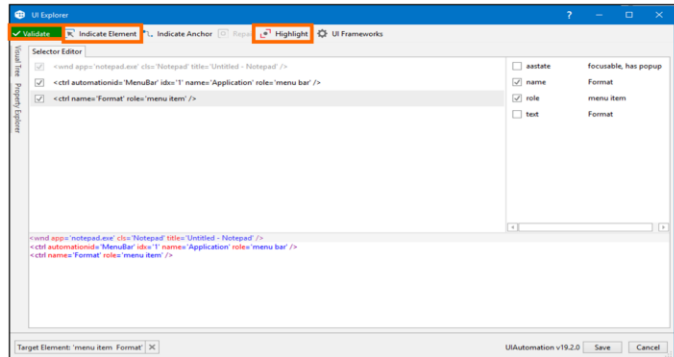
Container 2: menu bar

Container 3: font

Property Explorer

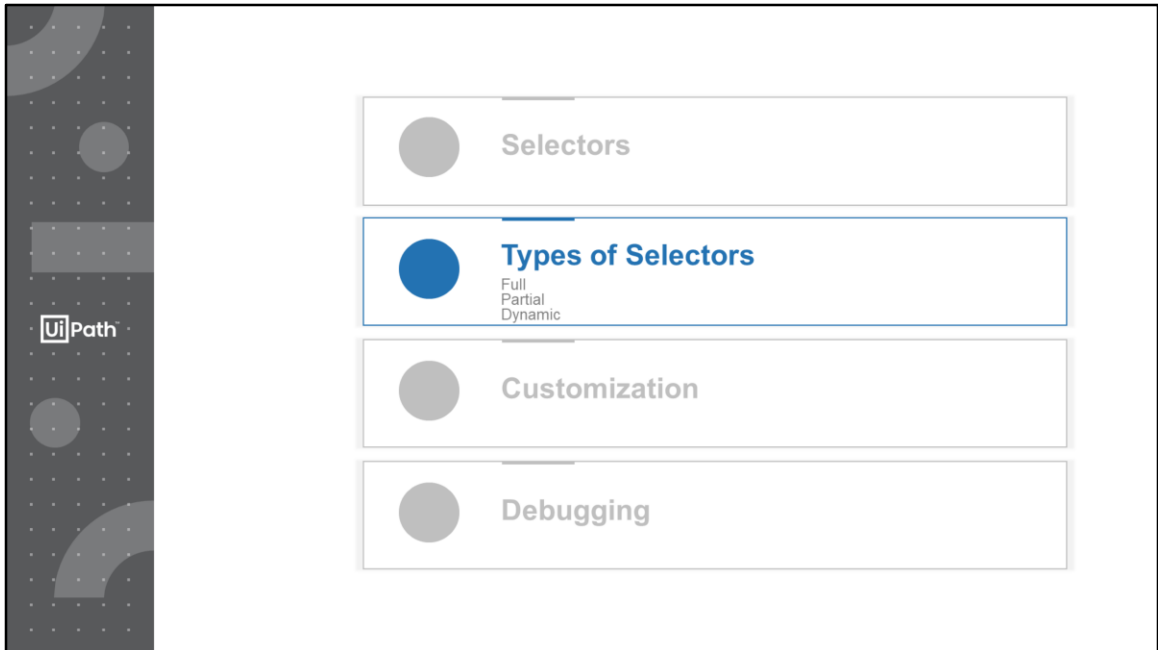
The **Property Explorer** contains features which make UI Explorer functionality exclusive.

- Validate
- Indicate Element
- Highlight



Property Explorer: contains features which make UI Explorer functionality exclusive.

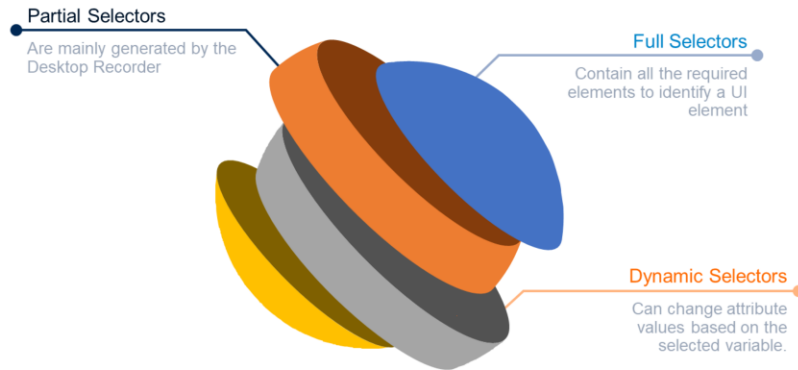
- **Validate:** It has different colors to define a selector is correct or not. (Already defined in the previous slide)
- **Indicate Element:** To indicate a particular UI element (already defined in the previous slide).
- **Highlight:** This Function is used to Highlight the UI Element which we are currently editing



In this topic, we will learn about the types of selectors.

Types Of Selectors

The **Selectors** are defined by looking at the element they target to perform their specific activity.

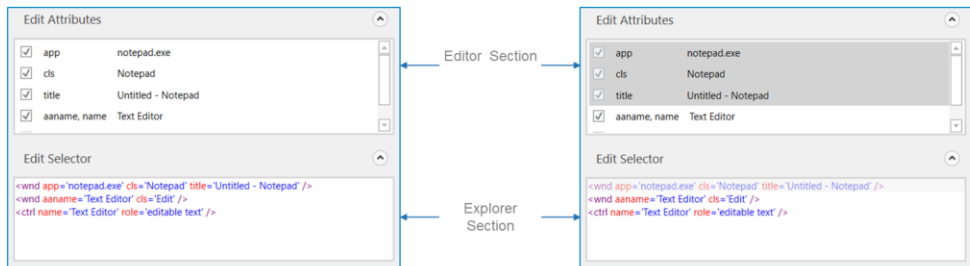


Since we now know what selectors are, let us see how we can define and assess them. It is done by looking at the element they target to perform their specific activity as well as how “deep” they go down the applications Visual Tree. There are three types of selectors:

- Full Selectors: Contain all the required elements to identify a UI element
- Partial Selectors: Are mainly generated by the Desktop Recorder
- Dynamic Selectors: Can change attribute values based on the selected variable.

Partial selectors

Best suited for performing multiple actions in the same window.



Best suited for situations in which the targeted element can constantly change **it's its** value.



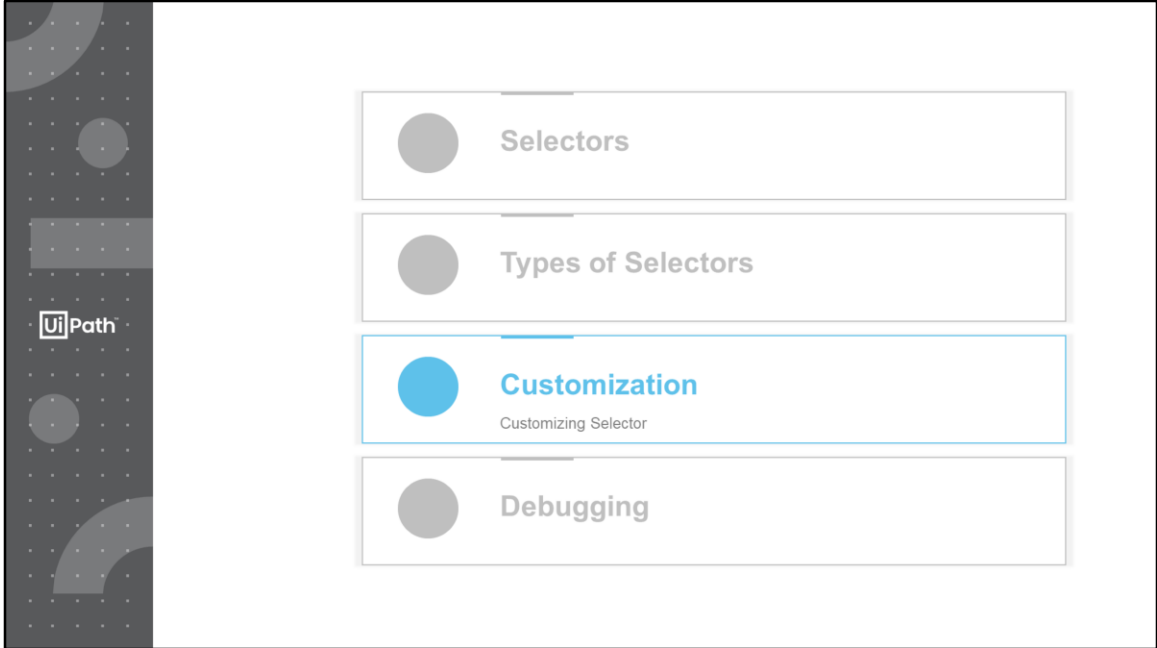
Partial Selectors: Partial selectors are the ones that are mainly generated by the Desktop Recorder. They do not contain information about the top-level window, as you can see it's grayed out in the Editor section as well as the Explorer section. They mostly recommended when performing multiple actions in the same window. It can edit only elements belonging to the partial selector. The prepended ones are grayed out and read-only.

Dynamic selectors: The dynamic selectors can change specific attribute values based on the selected variable. For example, let's presume that we have this calendar on a web page, and we want to click a specific date and receive this action as user input. To do this, we can use the dynamic selector to click the specified date by the user. We store the input from the user in a variable, and then we put it inside the selector. The robot will receive the date,

day, and month, identify the specific element from the calendar GUI and perform the required action.

Example for dynamic selector:

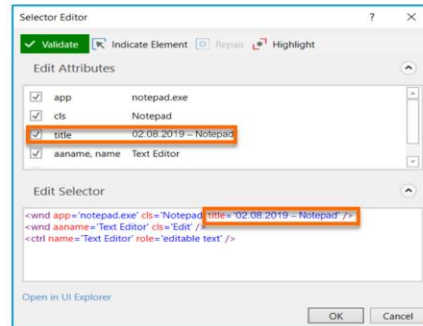
It would help you to create a robot that would add events in your calendar based on the emails that you receive, or the sticky notes that you save on your computer and other applications.



In this topic, we will learn about selector customization.

Customizing Selectors

When a wildcard is used or a variable is added in between selectors, it is called **customizing selectors**. The default selector contains preset attributes.

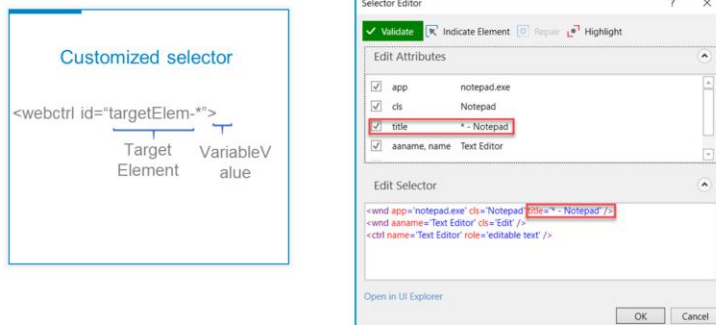


After looking at the partial, full and dynamic selectors, we should look at how to customize selectors. Their default setting contains some preset attributes that can easily change to make the selectors more reliable or tailoring them to your needs. This level of customization usually changes during the debugging phase. Let's see an example:

[<webctrl id="targetElem-212345">](#)

Customizing Selectors (Contd.)

The customized selector contains the "*" wildcard that replaces certain characters.



Wildcard types

Asterisk (*) – replaces zero or more characters
Question mark (?) – replaces a single character


adding a variable in between selectors can be called as making selectors dynamic or customizing a selector.

Let's see an example:

<webctrl id="targetElem-212345">

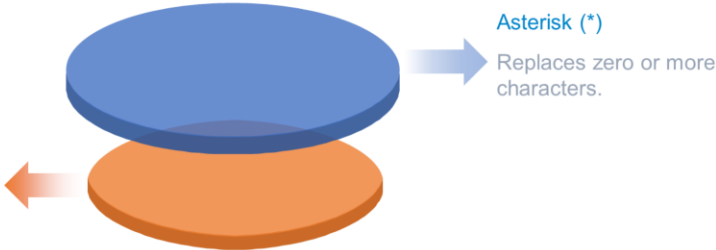
The variable value contains more characters

<webctrl id="targetElem-*">



Customizing Selectors (Contd.)

A wildcard is a special character that can replace the dynamic part of a selector. There are two types of wildcard:



Question mark (?)
Replaces a single character.

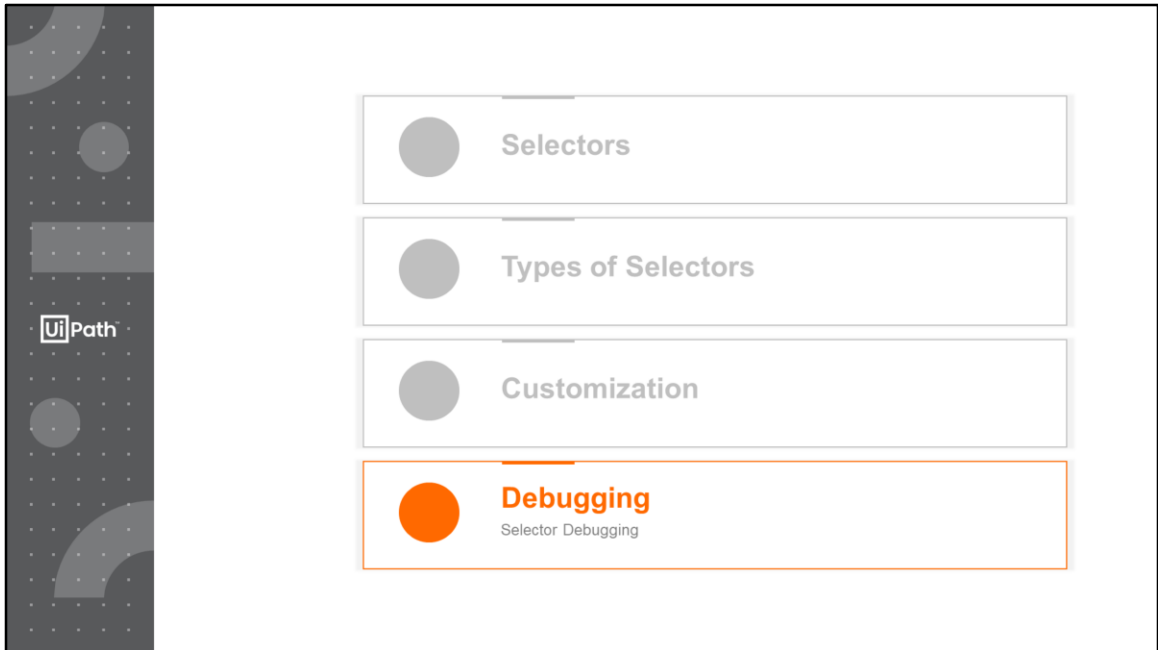
Asterisk (*)
Replaces zero or more characters.

Wildcards

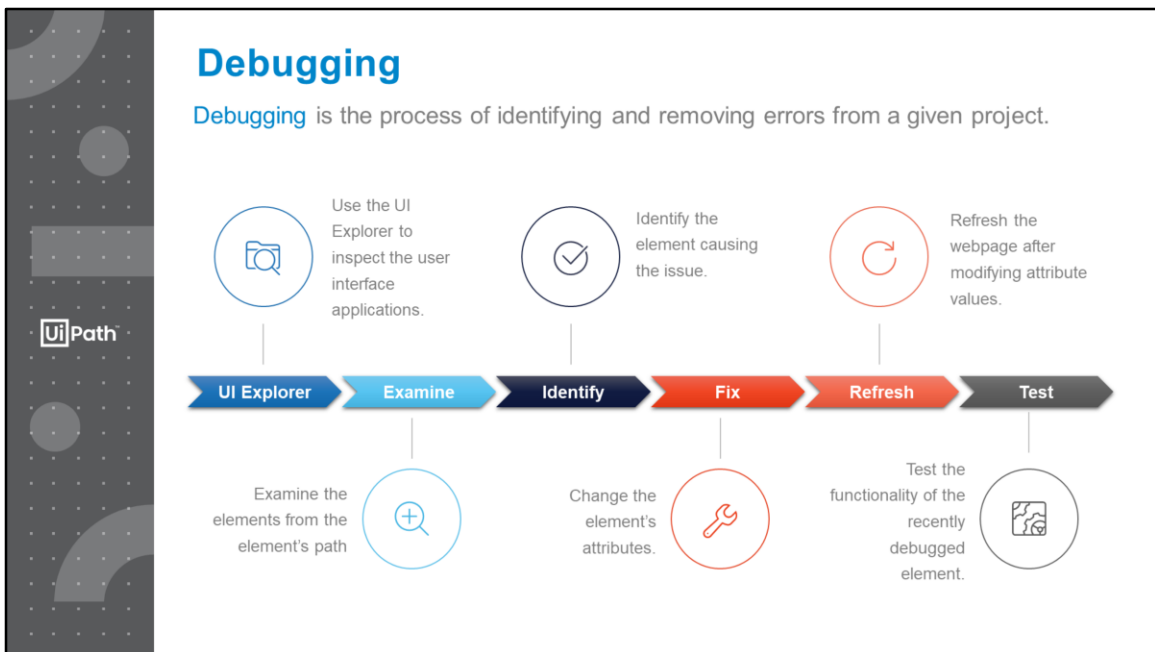
When we select a UI element; the selector captures its properties. However, these properties differ when we select the UI element of a different instance of an application with the selector. After this, The selector does not recognize the same UI element of another instance of the application. In this case, we can fix this problem by using wildcard characters or by attaching it to a live Element. Two wildcard characters are available in UiPath:

- The question mark symbol, ?, which replaces one character
- The asterisk symbol, that is, *, which replaces several characters

You might already be familiar with wildcards from their use in Windows file search, Google, or other similar engines. It's essential to understand how wildcards work because they are used extensively with selectors. There is a way to use wildcards, where we can use a file whose file name changes every day to display the current date. In this case, a static selector does not work after the limited time period such as one day. The solution replaces the dynamic part of the selector with an asterisk. In the wildcard process it would replace the name of the file from the selector with a wildcard.



In this topic, we will learn about selector debugging.



Debugging is the process of identifying and removing errors from a given project. Coupled with logging, it becomes a powerful functionality that offers you information about your project and step-by-step highlighting, so that you can be sure it is error-free.

UI Explorer is a great tool for checking and debugging selectors. It enables us to inspect all the attributes that could be used in identifying the element causing the issue. Once the element has been identified, the debugging process will start. Debugging a selector involves changing element's attributes, either adding or removing them and using wildcards where specific attributes have variable values inside them.

After each change, the webpage must be refreshed, and the selector verified for accuracy. This process isn't always done in the same way for each selector and the amount and type of debugging varies for every selector. These functionalities are helpful in the debugging process.

- ☐ Find Element
- ☐ Element Exists

- ☐ Find Children
- ☐ Get Attributes

Hence, to conclude “Debugging is a hit and trial method to identify the error and help in finding the correct selectors until the desired action is achieved.

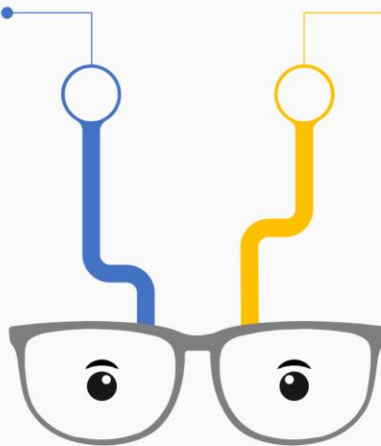
Takeaways

Selectors, Types of Selectors



Selectors rely on the GUI of the application to find elements

There are 3 types of selectors: Partial, Full and Dynamic, each having specific attributes.



The point of the Recap & Summary section is to go through the most important points covered in the lesson, after the students had the chance to see them in practice and obtain a consolidated view. The teacher should use facilitation questions to help the students map the key points and offer a safe space to get questions and comments from them.

Some examples of facilitation questions

1. What is a UI element?
2. What is a user interface?
3. What are the different types of user interfaces?

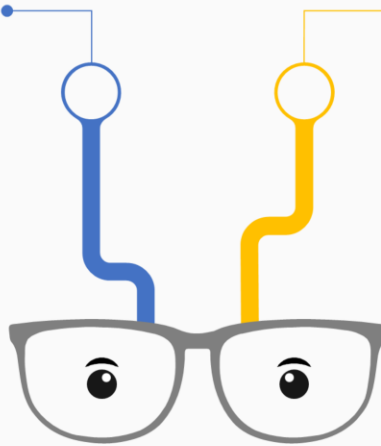
Takeaways



Customization, Debugging

The wildcard is a symbol that enables you to replace the characters during the runtime process.

You can use the UI Explorer tool for debugging.



The point of the Recap & Summary section is to go through the most important points covered in the lesson, after the students had the chance to see them in practice and obtain a consolidated view. The teacher should use facilitation questions to help the students map the key points and offer a safe space to get questions and comments from them.

Some examples of facilitation questions

1. What is a wildcard?
2. Describe the debugging process.

Questions & Answers



Q&A

Now it's your turn. What's on your mind at the end of this?

The image shows a vertical decorative bar on the left side of a slide. It features a dark grey background with a pattern of small white dots. Overlaid on this are several light grey geometric shapes: a large quarter-circle in the top-left, a smaller circle in the middle, a horizontal rectangle, another circle below that, and a larger quarter-circle in the bottom-left. The UiPath logo, consisting of a square icon with 'Ui' and the word 'Path' to its right, is positioned in the middle of this bar.

What is the format in which the selectors store GUI attributes and characteristics?

- a) XML.
- b) JavaScript.
- c) CommaSeparatedValues.

Correct answer: a) The selectors store GUI attributes and characteristics in XML format.

The slide features a dark vertical sidebar on the left with the UiPath logo and decorative geometric shapes. The main content area has a light blue header with the question.

Which of the following best defines a partial selector?

- a) Contains all the required elements to identify an UI element, including the top-level window.
- b) It's generated by the Desktop Recorder and does not contain information about the top-level window.
- c) Can change certain attribute values based on the selected variable.

Correct answer: b) It's generated by the Desktop Recorder and doesn't contain information about the top-level window.

The left side of the slide features a dark grey vertical bar with a light grey dotted pattern. Overlaid on this pattern are several geometric shapes: a large quarter-circle in the top-left, a smaller circle in the top-right, a horizontal rectangle in the middle, another circle in the bottom-left, and a larger quarter-circle in the bottom-right. The UiPath logo, consisting of a square icon with the letters 'Ui' and the word 'Path' to its right, is positioned in the center of this bar.

Which types of selector are best suited for wildcard

- a) Partial selectors.
- b) Full selectors.
- c) Dynamic selectors.

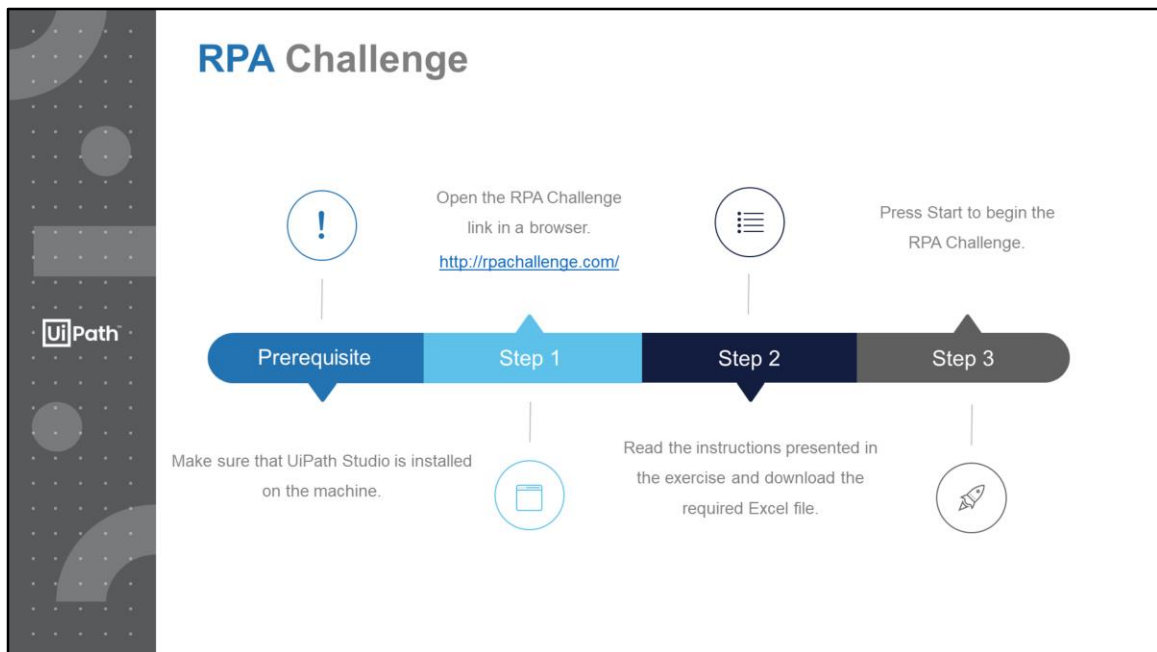
Correct answer: c) Dynamic selectors.

The image shows a vertical decorative bar on the left side of a slide. It features a dark grey background with a pattern of small white dots. Overlaid on this are several light grey geometric shapes: a large quarter-circle in the top-left, a smaller circle in the middle, and another large quarter-circle in the bottom-left. The UiPath logo, consisting of a square icon with 'Ui' and the word 'Path' to its right, is positioned in the middle of this bar.

Which set of activities can help you when debugging selectors?

- a) "Identify Element", "List Plugins" and "Expand Selectors".
- b) "Find Element", "Element Exists", "Find Children" and "Get Attributes".
- c) "Find Bug", "Show Element Path" and "List Extensions".

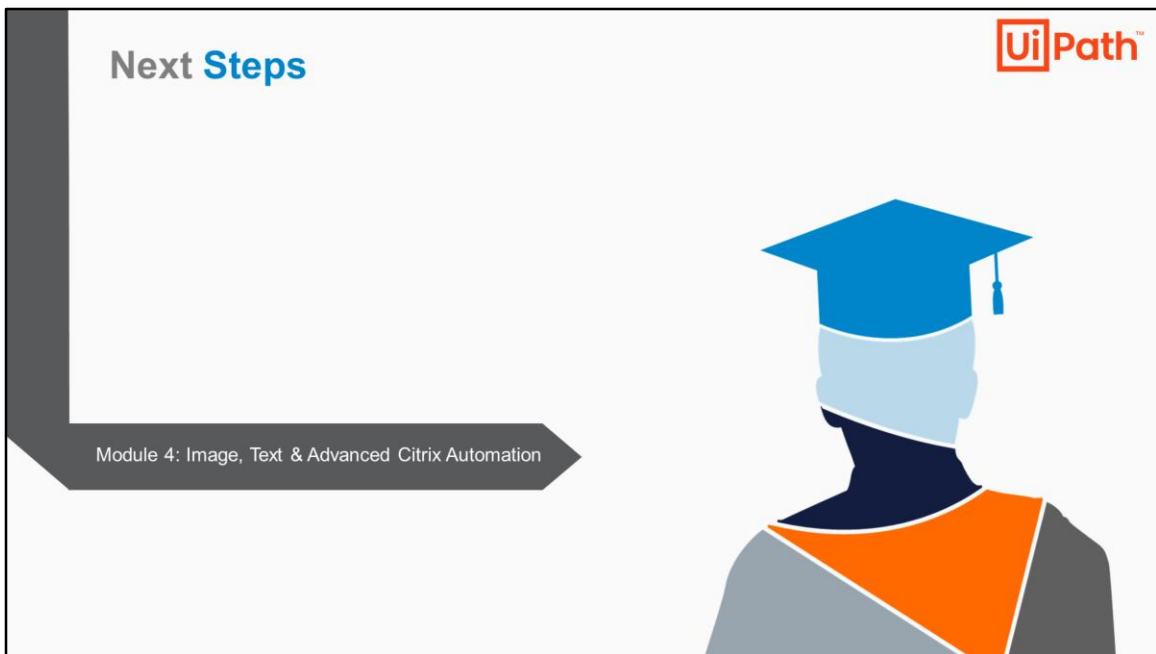
Correct answer: b) "Find Element", "Element Exists", "Find Children" and "Get Attributes"



Exercise Solution:

1. Read the .xlsx file with the read range activity. The Add Headers Option must be checked. Use the CTRL+K shortcut inside the Output-DataTable field of this activity and enter the name of the variable in which you want to store the data from the excel.
2. Use the Open browser activity and use the RPA challenge link - "http://rpachallenge.com/"
3. Use the attach browser activity to attach to the RPA Challenge page.
4. Use the Click activity on the Start button. This click activity should have the "WaitForReady" property on "NONE" and the SimulateClick property checked.
5. Use the For Each Row activity with the previously created variable for the data stored inside the excel.
6. Inside the loop, use the Type Into activity and select the first field in which you want to type. The text that needs to be entered is the data from the excel. Depending on what field you have picked you must enter the correct value.
Example: First name field = row("First Name").ToString.
7. Repeat step 6 for all the other fields.
8. Use the Click activity to click submit after all the fields have been completed

and we can get the next row with sets of data. This activity should be at the bottom of all Type Into activities and inside the loop.



In the next lesson, we will be covering Image, Text & Advanced Citrix Automation.