RPA Design and Development V3.0

STUDENT MANUAL – Lesson 4

Slide 1

RPA Design and Development

V3.0

Welcome to ‘RPA Design and Development Course’.

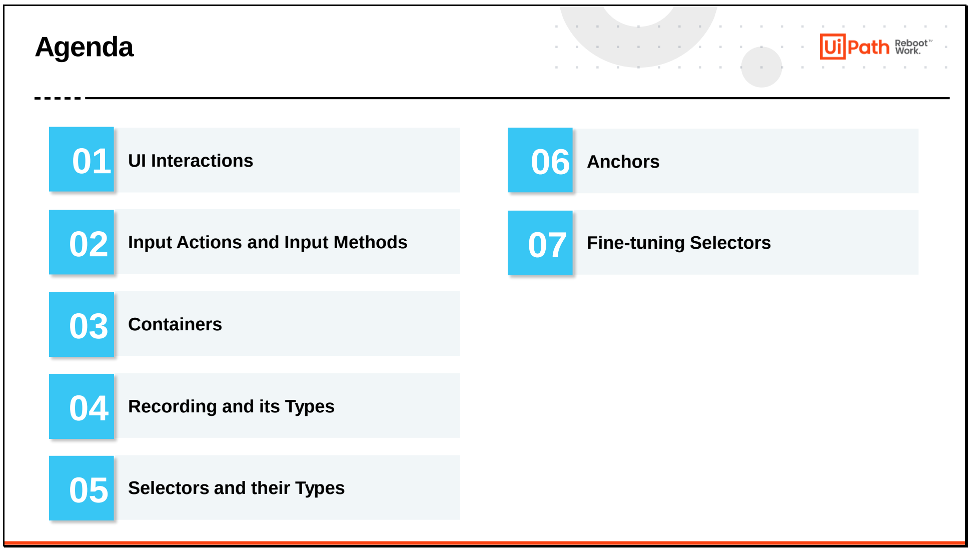
Slide 2

Lesson 4

Selectors

The fourth lesson of this course is Selectors.

Slide 3



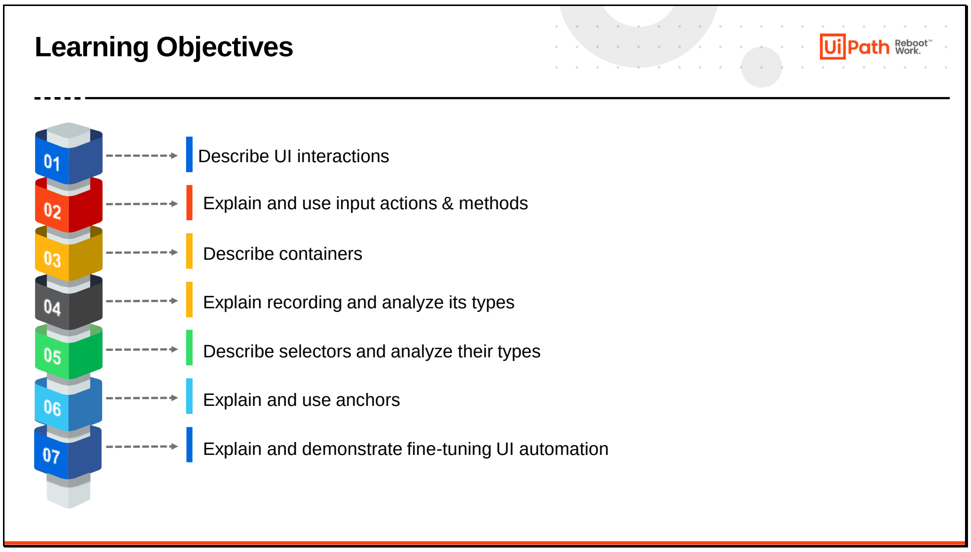
Agenda

- 01 UI Interactions
- 02 Input Actions and Input Methods
- 03 Containers
- 04 Recording and its Types
- 05 Selectors and their Types
- 06 Anchors
- 07 Fine-tuning Selectors

The agenda of this lesson is:

- UI interactions
- Input Actions and Input Methods
- Containers
- Recording and its types
- Selectors and their types
- Anchors
- Fine-tuning Selectors

Slide 4



Learning Objectives

01
02
03
04
05
06
07

- Describe UI interactions
- Explain and use input actions & methods
- Describe containers
- Explain recording and analyze its types
- Describe selectors and analyze their types
- Explain and use anchors
- Explain and demonstrate fine-tuning UI automation

By the end of this lesson, you will be able to:

- Describe UI interactions
- Explain and use input actions and methods
- Describe containers
- Explain recording and analyze its types
- Describe selectors and analyze their types
- Explain and use anchors
- Explain and demonstrate fine-tuning selectors

Slide 5

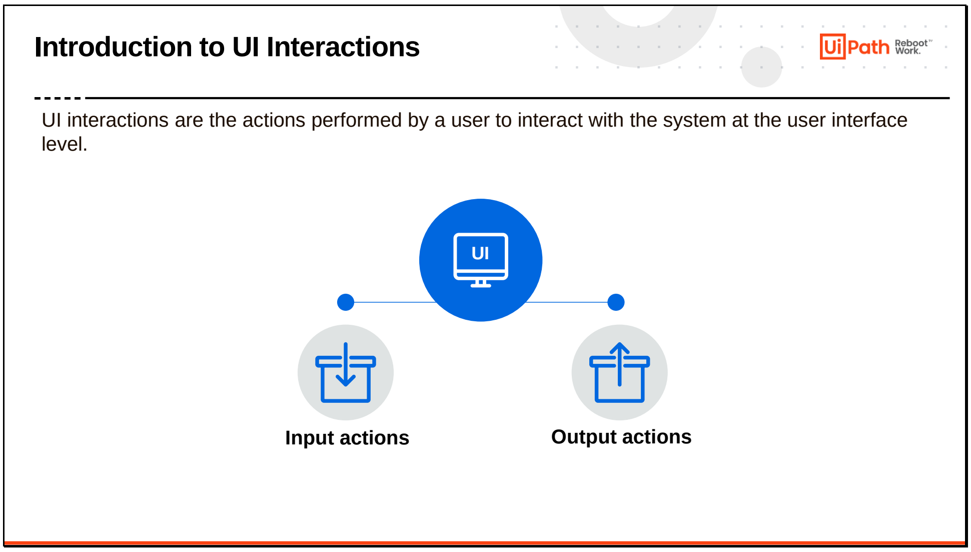
UI Interactions

- Introduction to UI Interactions
- Types of UI Interactions

UiPath Reboot™ Work.

This section introduces UI interactions and their types.

Slide 6



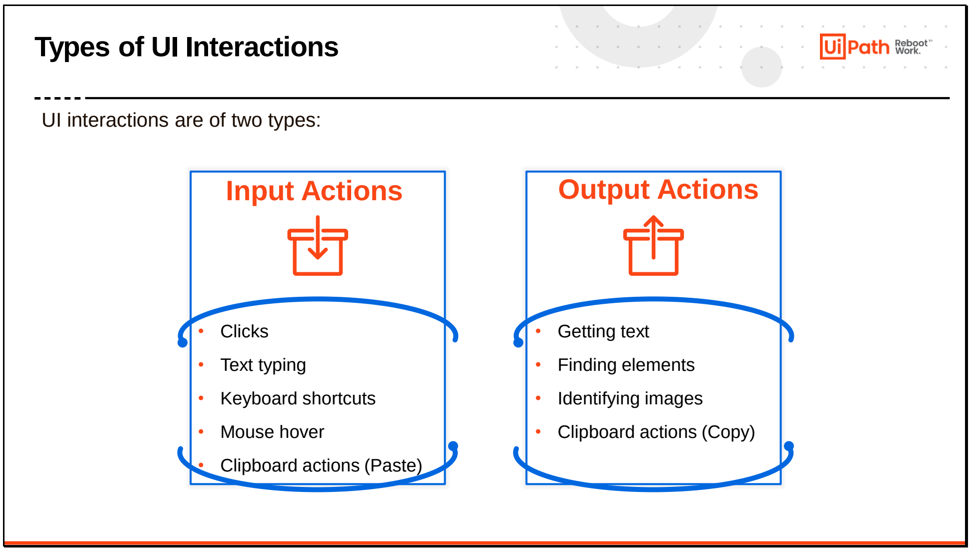
Introduction to UI Interactions

UI interactions are the actions performed by a user to interact with the system at the user interface level.

UI elements refer to all graphical user interface pieces that constitute an application (such as windows, check boxes, text fields or drop-down lists). UI automation can be easily implemented by using UI interactions which are the actions performed by a user to interact with the system at the user interface level.

There are two types of UI interactions that can appear in automation: **input actions and output actions**.

Slide 7



Types of UI Interactions

UI interactions are of two types:

Input Actions

- Clicks
- Text typing
- Keyboard shortcuts
- Mouse hover
- Clipboard actions (Paste)

Output Actions

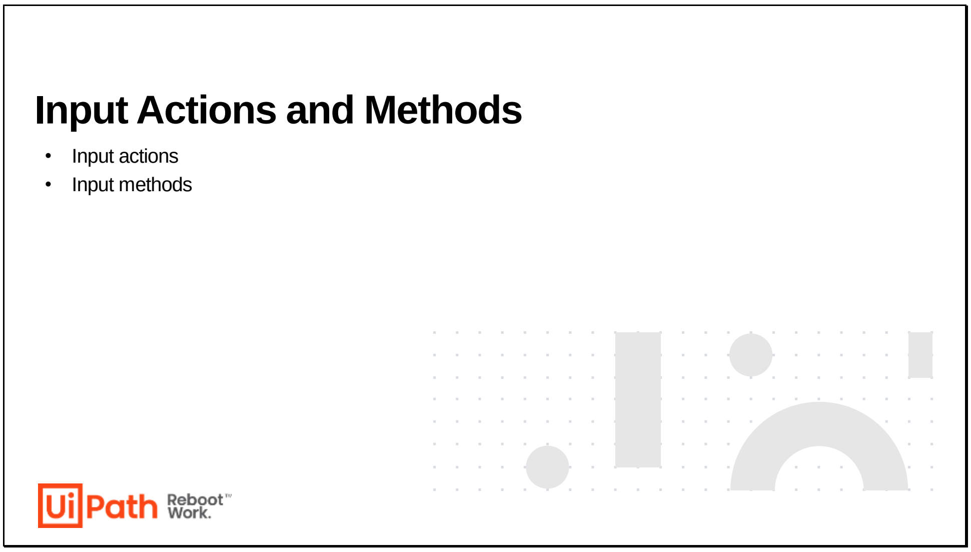
- Getting text
- Finding elements
- Identifying images
- Clipboard actions (Copy)

- **Input actions** are those actions through which the user sends data to an application (by clicking or typing) to produce an outcome. It means inserting data into an application. Every time the user clicks on a UI element, information is sent to the application and there is an outcome.
  These actions include:
  - Clicks
  - Text typing
  - Keyboard shortcuts
  - Mouse hover
  - Clipboard actions (Paste)

- **Output actions** are those actions through which the user takes out data from an application. It means reading data from an application.
  These actions include:
  - Getting text
  - Finding elements
  - Identifying images
  - Clipboard actions (Copy)

(Output actions and methods are explained in detail in the next lesson as this lesson focuses on Input methods).

Slide 8



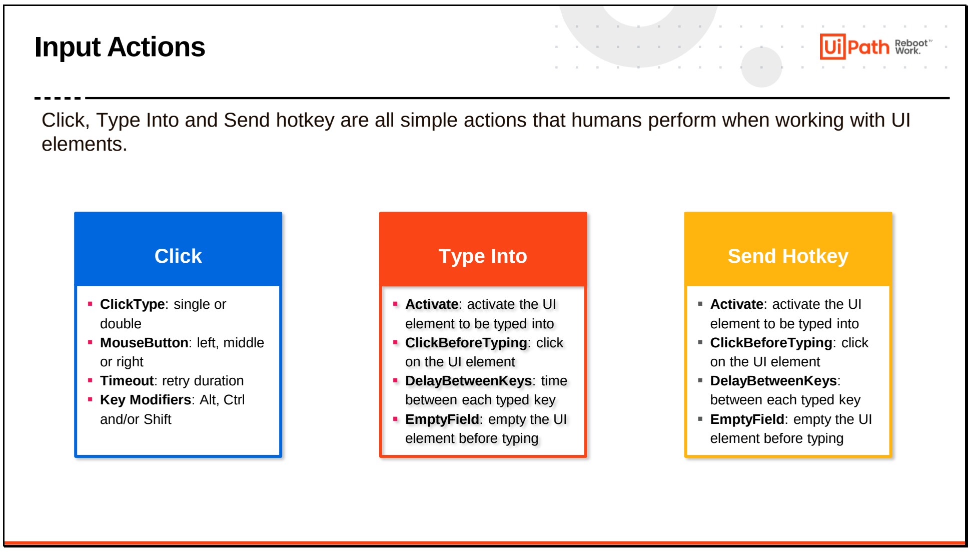
Input Actions and Methods

- Input actions
- Input methods

UiPath Reboot Work.

This section introduces input actions and input methods in UiPath Studio.

Slide 9



Input Actions

Click, Type Into and Send hotkey are all simple actions that humans perform when working with UI elements.

Click

- **ClickType**: single or double
- **MouseButton**: left, middle or right
- **Timeout**: retry duration
- **Key Modifiers**: Alt, Ctrl and/or Shift

Type Into

- **Activate**: activate the UI element to be typed into
- **ClickBeforeTyping**: click on the UI element
- **DelayBetweenKeys**: time between each typed key
- **EmptyField**: empty the UI element before typing

Send Hotkey

- **Activate**: activate the UI element to be typed into
- **ClickBeforeTyping**: click on the UI element
- **DelayBetweenKeys**: between each typed key
- **EmptyField**: empty the UI element before typing

In UiPath Studio, the main input actions are Click, Type Into and Send Hotkey which are also the main actions that a human user performs to input data in an application.

1. Click: Clicks a specified UI element.

- **Click Type**: Specifies the type of mouse click (single, double, up, down) used when simulating the click event. By default, a single click is selected.
- **Mouse Button**: The mouse button (left, right, middle) used for the click action. By default, the left mouse button is selected.
- **Timeout**: Specifies the amount of time (in milliseconds) to wait for the activity to run before the SelectorNotFoundException error is thrown. (Default value= 30000 milliseconds).
- **Key Modifiers**: Enable the users to add a key modifier. The available options are: Alt, Ctrl, Shift, Win.

2. Type Into: Sends keystrokes to a UI element.

- **Activate**: On selecting this check box, the specified UI element is brought to the foreground and activated before the text is written.
- **Click Before Typing**: On selecting this check box, the specified UI element is clicked before the text is written.
- **Delay Between Keys**: Delay time (in milliseconds) between two keystrokes. (Default value= 10 milliseconds; Maximum value= 1000 milliseconds.)
- **Empty Field**: On selecting this check box, all previously existing content in the UI element is erased before writing the text.

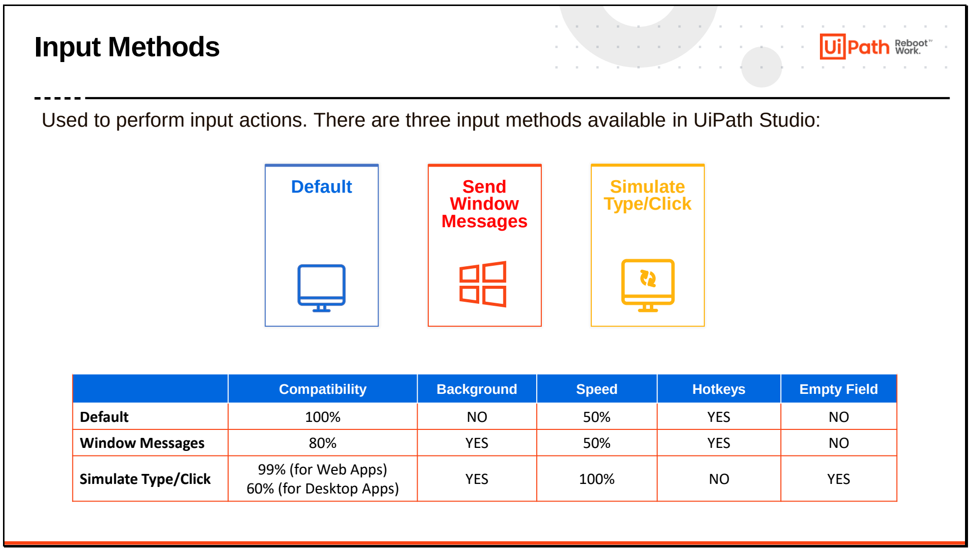
Sends keyboard shortcuts to a UI element.

- **Activate**: On selecting this check box, the specified UI element is brought to the foreground and activated before the text is written.
- **Click Before Typing**: On selecting this check box, the specified UI element is clicked before the text is written.
- **Delay Between Keys**: Delay time (in milliseconds) between two keystrokes. (Default value= 10 milliseconds; Maximum value= 1000 milliseconds.)
- **Empty Field**: On selecting this check box, all previously existing content in the UI element is erased before writing the text.

All the input actions share some (common) properties:

- **Delay**: Set a delay before or after the click.
- **WaitForReady**: Wait for the target to become ready by verifying certain application tags.

Slide 10



Input Methods

Used to perform input actions. There are three input methods available in UiPath Studio:

Default

Send Window Messages

Simulate Type/Click

	Compatibility	Background	Speed	Hotkeys	Empty Field
Default	100%	NO	50%	YES	NO
Window Messages	80%	YES	50%	YES	NO
Simulate Type/Click	99% (for Web Apps) 60% (for Desktop Apps)	YES	100%	NO	YES

All the input actions are presented using clicks or types. A human uses the mouse and the keyboard for input. To replicate these actions, UiPath Studio supports three input methods for performing input actions:

- **Default method:** This method captures the input given while interacting with the UI element of the screen. For example, selecting a UI element on the screen with the help of a mouse or keyboard.

- **Send Window Messages:** This method performs similar actions as the default method. As you can see on your screen it is less compatible with the application being used. This function captures actions performed using keyboard and mouse.

- **Simulate Type/Click:** This method is also similar to default and send window messages. The points which differentiate it from the other two are speed and input source. This action is much faster than default and send window messages and it does not capture input given by the keyboard.

These methods are similar in nature however, there are differences in their performance. The slide displays a comparison chart for these input methods.
Comparison based on different parameters for these methods:

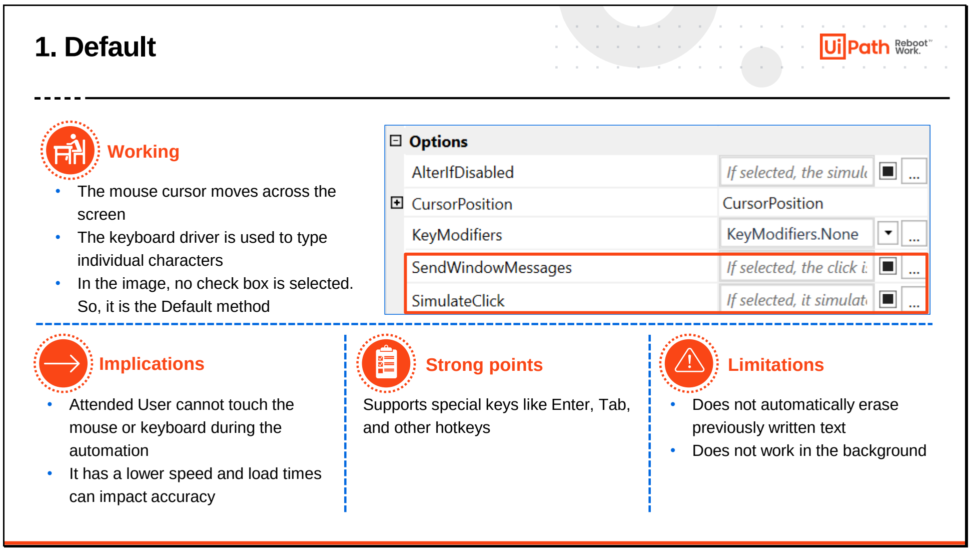
- **Compatibility**: The default method is 100% compatible, given that it does exactly what a human would do. Simulate Type/Click does a great job for web automation but has significant limitations when it comes to compatibility with desktop applications.
- **Background**: Both WindowMessages and Simulate work in the background, while the Default doesn’t.

- **Speed**: Simulate method correctly identifies the controls used by the target apps, its speed is significantly higher than the speed of the other 2 methods.
- **Hotkeys**: Only Default and WindowMessages can send hotkeys.
- **Empty Field**: Empty field is a property of UI interacting activities. It is used to erase the data already present in the input area of the UI element. For example, a textbox containing previously filled data which is accessed repetitively in the code. When the Default or Send Window Message method is used, the user has to ensure that the ‘empty field’ property is checked for erasing the already present data. On the other hand, Simulate method empties the field by default.

The user must understand the differences between all three types of input methods and when to use them.

To know more, visit: https://docs.uipath.com/studio/v2020.10/docs/input-methods

Slide 11



1. Default

Working

- The mouse cursor moves across the screen
- The keyboard driver is used to type individual characters
- In the image, no check box is selected. So, it is the Default method

⊟ **Options**

AlterIfDisabled	If selected, the simul	▣ ...
⊞ CursorPosition	CursorPosition	
KeyModifiers	KeyModifiers.None	▾ ...
SendWindowMessages	If selected, the click i	▣ ...
SimulateClick	If selected, it simulat	▣ ...

Implications

- Attended User cannot touch the mouse or keyboard during the automation
- It has a lower speed and load times can impact accuracy

Strong points

Supports special keys like Enter, Tab, and other hotkeys

Limitations

- Does not automatically erase previously written text
- Does not work in the background

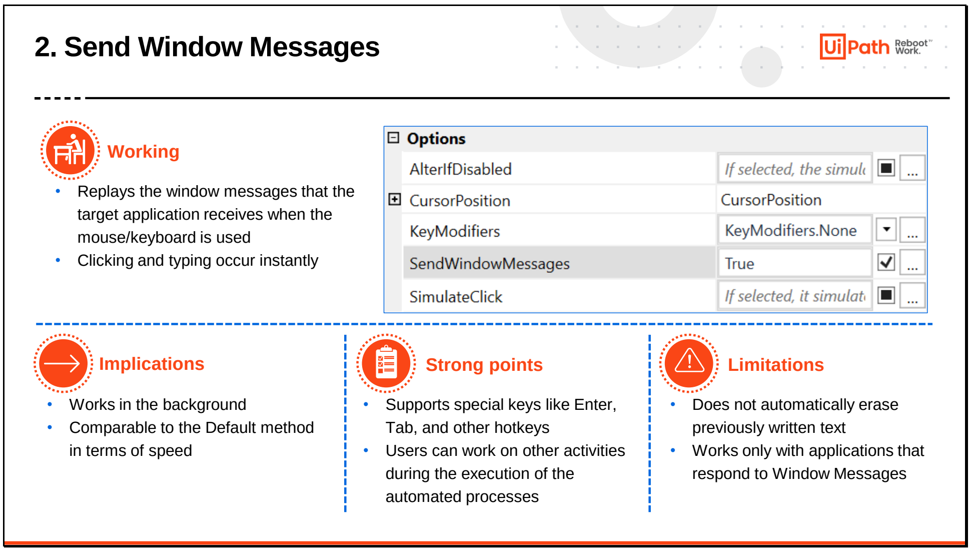
Each input action lets the user switch between the three methods. ‘Simulate Type/Click/Hover’ and ‘SendWindowMessages’ can be chosen by checking the corresponding box from the Properties panel; if none of the two boxes is checked, the Default input method is used.

It captures human actions performed with a mouse and keyboard.

It has an advantage over the other two methods that it uses the keyboard driver through which it can replicate the touch of every key, including special keys. Also, this method is 100% compatible with all applications.

In terms of limitations, it doesn’t work in the background. The window must be active and on top of other applications or webpages. Moreover, if an editable field already contains text, it doesn’t erase it, instead, it starts writing after it. Also, since typing each character and moving the mouse across the screen are not the fastest ways to solve a task, this method is the slowest of all three.

Slide 12



2. Send Window Messages

Working

- Replays the window messages that the target application receives when the mouse/keyboard is used
- Clicking and typing occur instantly

⊟ **Options**

AlterIfDisabled	If selected, the simul… ■ …
⊞ CursorPosition	CursorPosition
KeyModifiers	KeyModifiers.None ▾ …
SendWindowMessages	True ☑ …
SimulateClick	If selected, it simulat… ■ …

Implications

- Works in the background
- Comparable to the Default method in terms of speed

Strong points

- Supports special keys like Enter, Tab, and other hotkeys
- Users can work on other activities during the execution of the automated processes

Limitations

- Does not automatically erase previously written text
- Works only with applications that respond to Window Messages

The Send Window Messages input method can be activated from the Properties tab, by checking the ‘SendWindowMessages’ box.

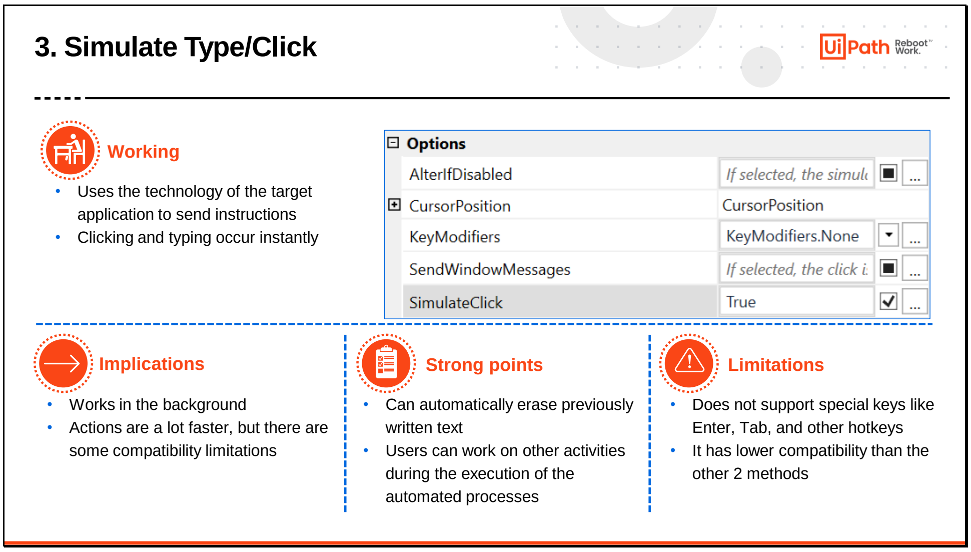
This method also captures human actions performed with the help of a mouse or keyboard.

The advantage of this method is that it works in the background. As a result, the mouse doesn’t move on the screen, and typing isn’t done by keyboard simulation. Instead, the clicks and the typing appear instantly.

As the window messages support hotkeys and shortcuts, this method can be used to send special characters in applications.

Just like the Default method, it doesn’t erase the text that is already typed in the target UI element by default.

Slide 13



3. Simulate Type/Click

Working

- Uses the technology of the target application to send instructions
- Clicking and typing occur instantly

⊟ **Options**

AlterIfDisabled	*If selected, the simul*	■ ...
⊞ CursorPosition	CursorPosition	
KeyModifiers	KeyModifiers.None	▾ ...
SendWindowMessages	*If selected, the click i*	■ ...
SimulateClick	True	☑ ...

Implications

- Works in the background
- Actions are a lot faster, but there are some compatibility limitations

Strong points

- Can automatically erase previously written text
- Users can work on other activities during the execution of the automated processes

Limitations

- Does not support special keys like Enter, Tab, and other hotkeys
- It has lower compatibility than the other 2 methods

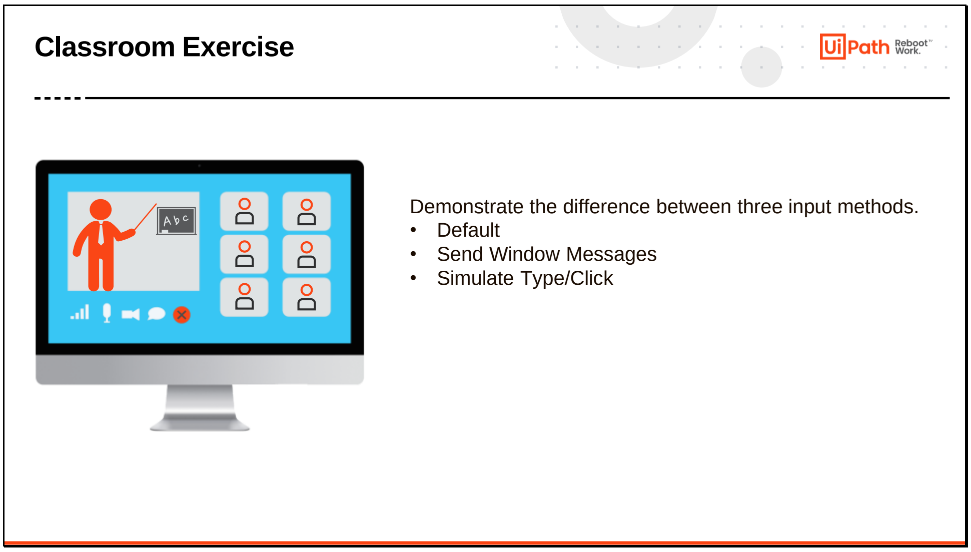
The Simulate input method can be activated from the Properties tab by checking the Simulate Type/Simulate Click box.

This method captures human actions performed with the help of a mouse.

This method uses the technology of the target application for sending instructions. It is the fastest of the three input methods but only works with applications whose controls react to the commands sent. This method can function in the background.

By default, this method erases the existing text in a UI Element but doesn’t support hotkeys and special keys. In general, it is used for web automation, where its compatibility is close to 100% (vs desktop applications, it is around 60%).

Slide 14



Classroom Exercise

Demonstrate the difference between three input methods.

- Default
- Send Window Messages
- Simulate Type/Click

Demonstrate the difference between three input methods – Default, Send Window Messages, and Simulate Type/Click.

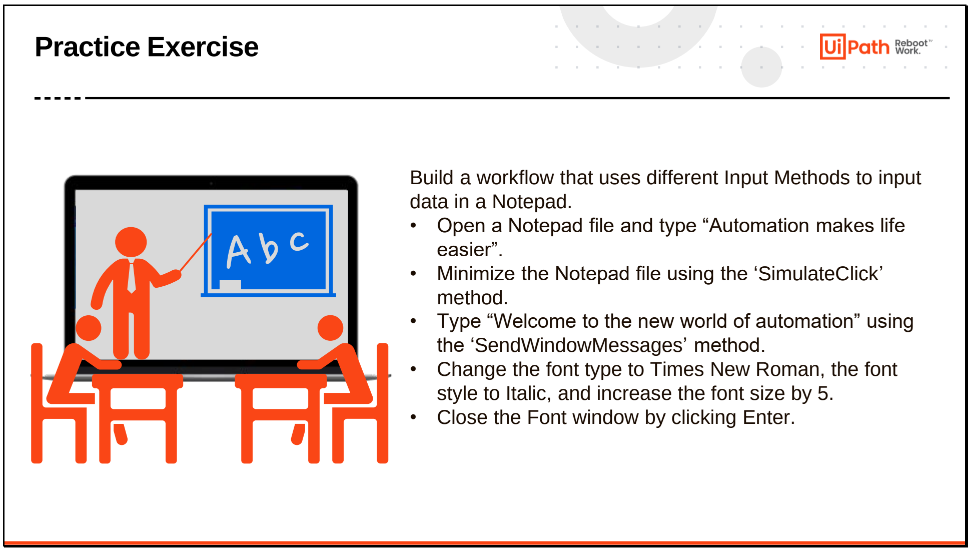
1. Open a Notepad application on your desktop.
2. Go to UiPath Studio, search for an Open Application activity from the Activities panel and drag it in the Designer panel.
3. Click the "Indicate window on screen" link and indicate the already open Notepad window.
    3.1 Navigate to its Properties panel and click the ellipsis icon of the Selector property.
    3.2 From the Selector Editor window, uncheck Title and click OK.
4. Search for a Type Into activity and drag it in the Do container.
5. Click the "Indicate element inside window" link and indicate the notepad text editor area.
6. Enter the text, "Automation is changing the world", in the Type Into activity.
7. Drag and drop a Click activity below the Type Into activity.
8. Click the "Indicate element inside window" and indicate the minimize button of the Notepad window.
9. Take another Type Into activity and place it below the Click activity.
10. Click the "Indicate element inside window" and indicate the notepad text editor area.
11. Enter the text, "Automation makes life easier".
12. Click "+" icon and select enter.
13. Cut the special key string for enter and move it before the text.
14. Take another Click activity, paste it below the Type Into activity.
15. Click the "Indicate element inside window" link and click on the notepad icon in the taskbar to restore it.
16. Run the process.

By default, input process does not erase pre-existing texts and supports special key. But writing in the background does not work.

Next, set the SendWindowMessages property to True for the Type Into activities from the properties panel and re-run the process. The second text gets written on a new line while the notepad was minimized. Here, input method works in the background, supports special keys, and does not erase pre-existing texts.

Next, set select the SimulateType property to True and deselect the SendWindowMessages and rerun the process. The second text gets written after replacing the first text while the notepad was minimized. The text shows the special key for enter – [k(enter)]. Here, input method works in the background. But it does not support special keys and erases pre-existing texts.

Slide 15



Practice Exercise

Build a workflow that uses different Input Methods to input data in a Notepad.

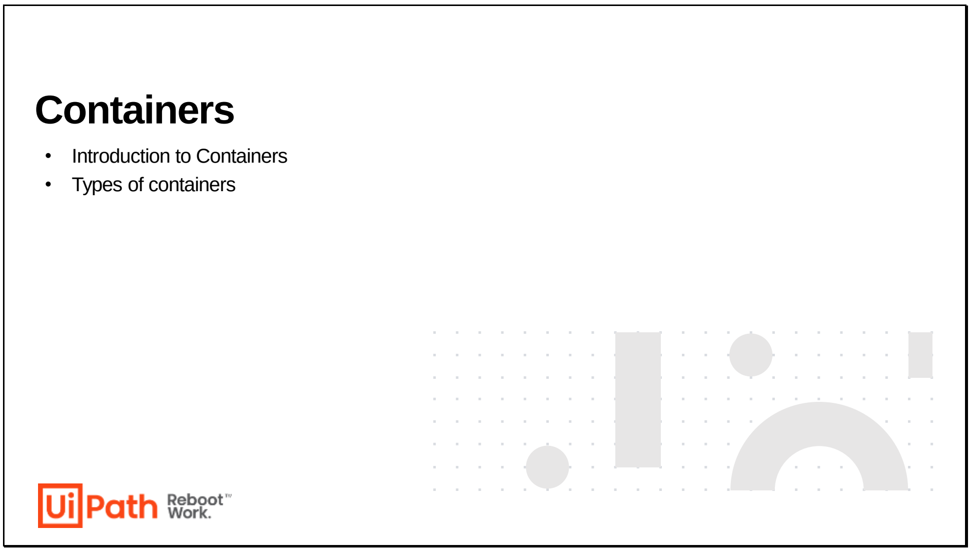
- Open a Notepad file and type “Automation makes life easier”.
- Minimize the Notepad file using the ‘SimulateClick’ method.
- Type “Welcome to the new world of automation” using the ‘SendWindowMessages’ method.
- Change the font type to Times New Roman, the font style to Italic, and increase the font size by 5.
- Close the Font window by clicking Enter.

Build a workflow that uses different Input Methods to input data in a Notepad.

Process Overview

- START
- Use Open Application activity to indicate a Notepad file.
- Use Type Into activity to enter “Automation makes life easier”.
- Minimize the Notepad window using the ‘Simulate Click’ method in the Click activity
- Use Type Into activity to enter “Welcome to the new world of automation” using the ‘Send Window Messages’ method.
- Use Send Hotkey activity to send “Ctrl + A”.
- Use Click, Attach Window, and Type Into activities to change the font type to Times New Roman, font style to Italic and increase the font size by 5.
- Use Send Hotkey activity to close the Font window of the Notepad window.
- STOP

Slide 16



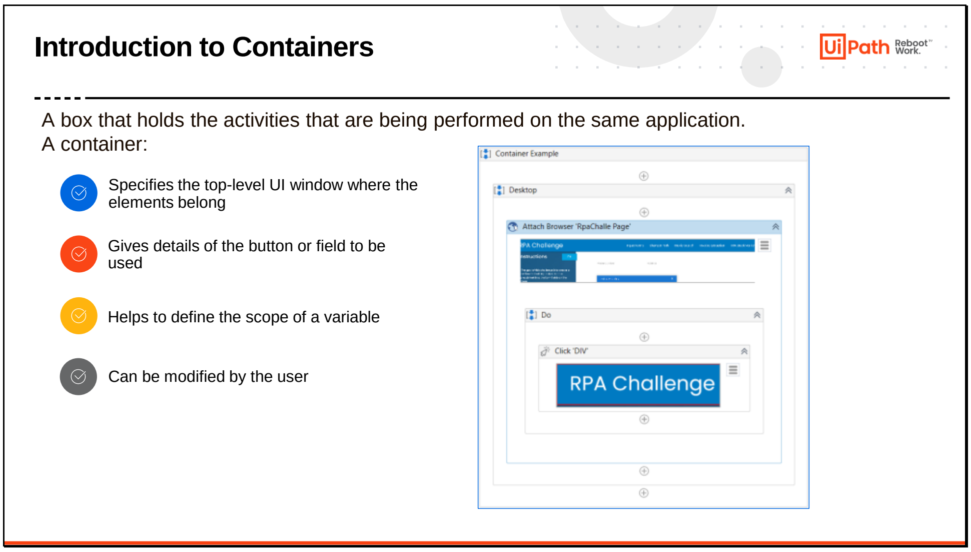
Containers

- Introduction to Containers
- Types of containers

UiPath Reboot™ Work.

This section explains containers and its types.

Slide 17



Introduction to Containers

A box that holds the activities that are being performed on the same application.
A container:

- Specifies the top-level UI window where the elements belong
- Gives details of the button or field to be used
- Helps to define the scope of a variable
- Can be modified by the user

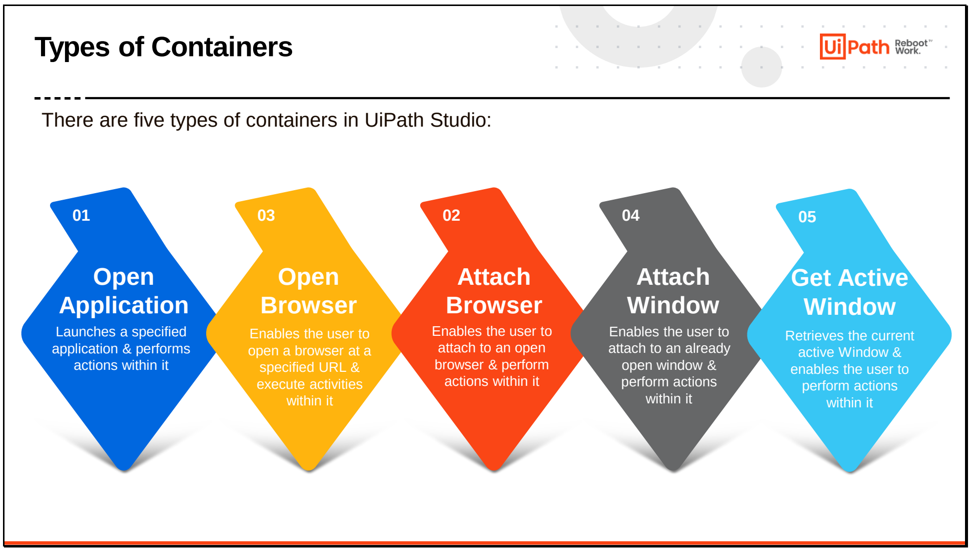
The Target UI Automation activity identifies the UI element the activity works with. Target is composed of multiple pieces: **the container**, **selector**, and **clipping region**, to ensure that a UI element is correctly identified.

A container is a box that holds the activities that are being performed on the same application. It specifies the top-level window where the elements belong. It gives more context of the intended button or field, so that different windows or different areas of the same app can be identified. A container also helps to define the scope of a variable. These are the UI activities.

When recording activities are used, containers are automatically generated, but users can make changes to them in the Properties panel. Alternatively, the user can drag and drop the required container from the Activities panel in UiPath Studio.

Whenever a mouse is hovered over a UI element, a square box appears around the element highlighting the area. This is the clipping region that is picked when clicked. This highlighting and appearance of a square box helps the user in identifying the area whose selector will be considered.

Slide 18



Types of Containers

There are five types of containers in UiPath Studio:

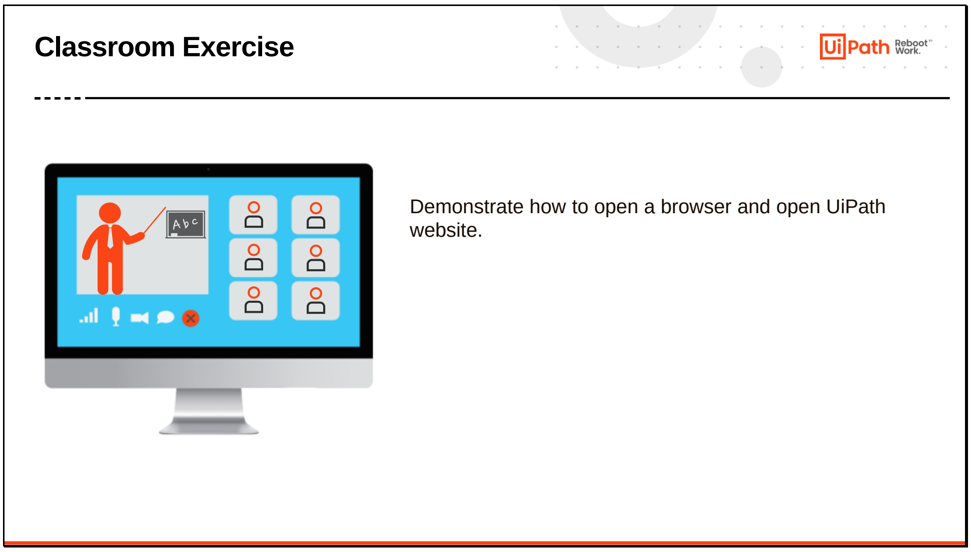
01
Open Application
Launches a specified application & performs actions within it
03
Open Browser
Enables the user to open a browser at a specified URL & execute activities within it
02
Attach Browser
Enables the user to attach to an open browser & perform actions within it
04
Attach Window
Enables the user to attach to an already open window & perform actions within it
05
Get Active Window
Retrieves the current active Window & enables the user to perform actions within it

There are five types of containers in UiPathStudio. These are:

- Open Application: Launches a specified application and performs multiple actions within it. It can also pass a list of arguments to the application.
- Attach Browser: Enables the user to attach to an already open browser and perform multiple actions within it. This activity is automatically generated when using the Web recorder.
- Attach Window: Enables the user to attach to an already open window and perform multiple actions within it. This activity is automatically generated when using the Desktop recorder.
- Open Browser: Enables the user to open a browser at a specified URL and execute multiple activities within it.
- Get Active Window: Retrieves the currently active Window and enables the user to perform multiple actions within it.

Scopes such as Open Browser or Open Application must be used when performing multiple actions on the same UI element, as this method is more reliable instead of targeting each activity on the same selector.

Slide 19



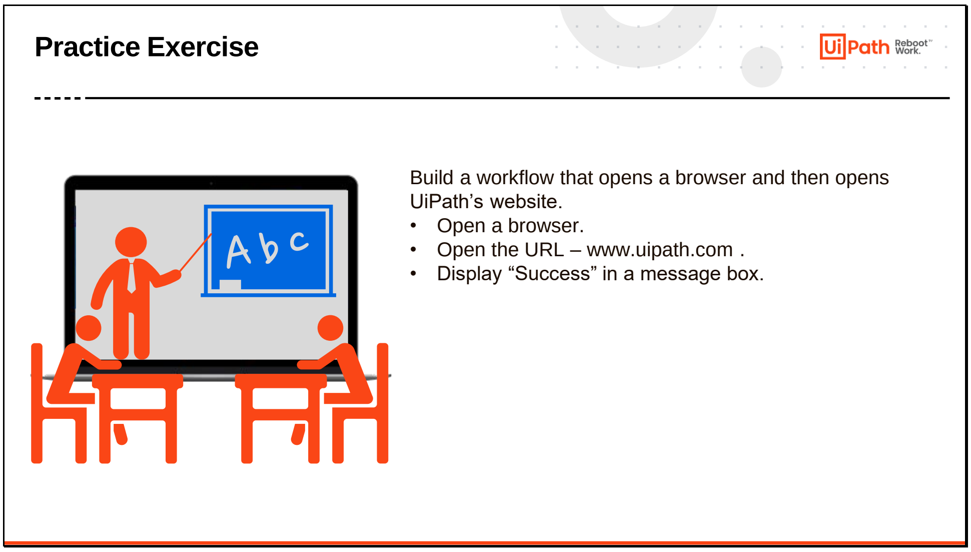
Classroom Exercise

Demonstrate how to open a browser and open UiPath website.

Demonstrate how to open a browser and a website by using the Open Browser activity.

- Open UiPath Studio.
- Drag and Drop an Open Browser activity in the main workflow.
- Enter the website URL "http://www.uipath.com" in the text area.
- Run the program. It will open UiPath's website.

Slide 20



Practice Exercise

Build a workflow that opens a browser and then opens UiPath's website.

- Open a browser.
- Open the URL – www.uipath.com .
- Display "Success" in a message box.

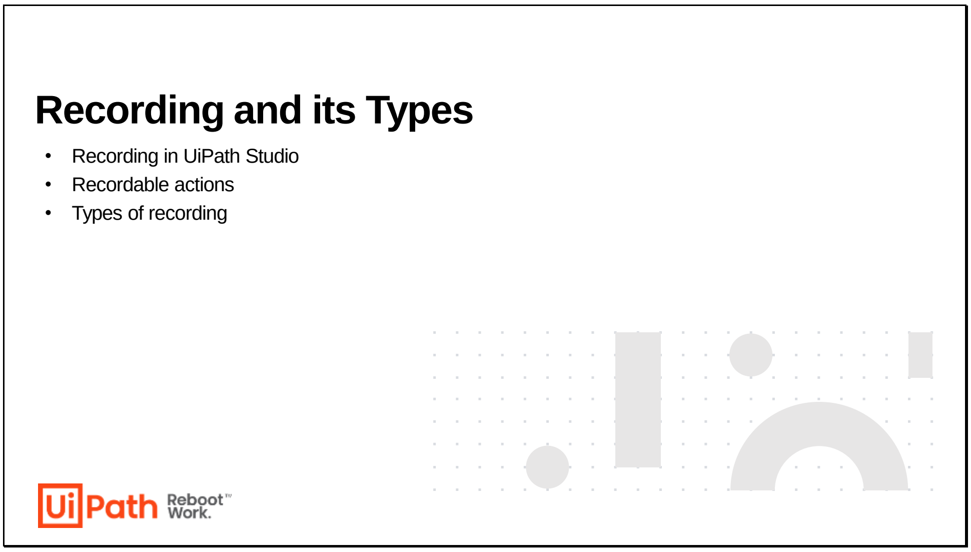
Build a workflow that opens a browser and then opens UiPath's website.

- Open a browser.
- Open the URL – www.uipath.com .
- Display "Success" in a message box.

Process Overview

- START
- Use an Open Browser activity and enter the website URL – "www.uipath.com".
- Use a Message Box activity and enter text "Success".
- STOP

Slide 21



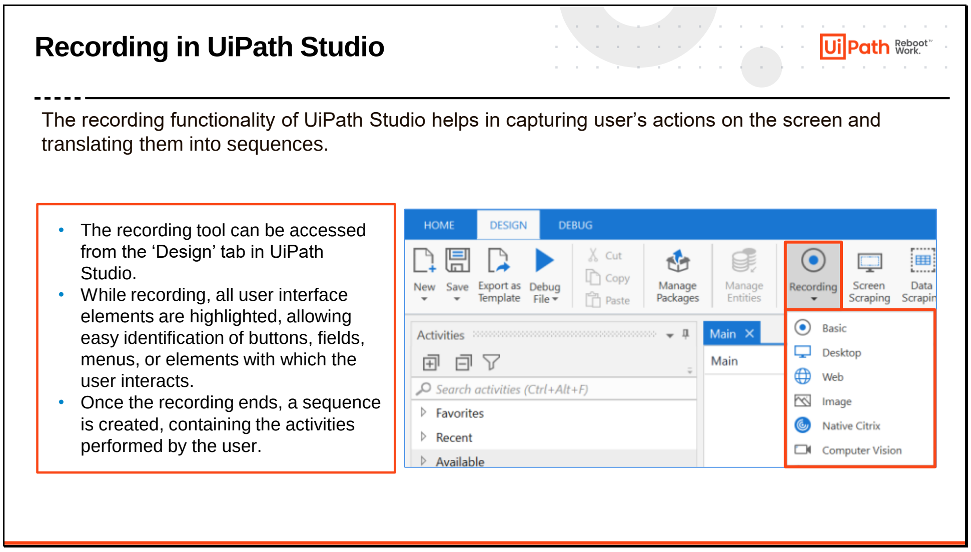
Recording and its Types

- Recording in UiPath Studio
- Recordable actions
- Types of recording

UiPath Reboot™ Work.

This section explains recording in UiPath Studio and its types in detail.

Slide 22



Recording in UiPath Studio

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

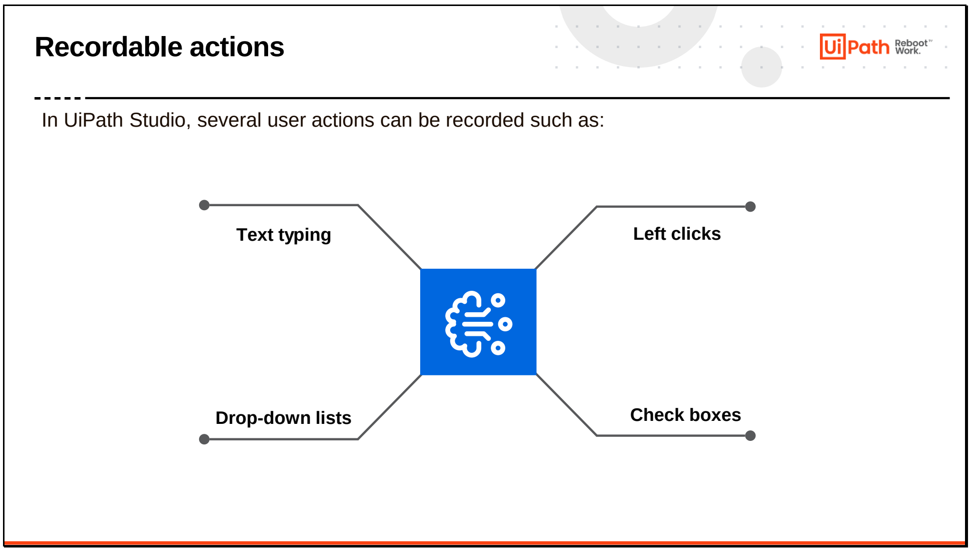
- The recording tool can be accessed from the 'Design' tab in UiPath Studio.
- While recording, all user interface elements are highlighted, allowing easy identification of buttons, fields, menus, or elements with which the user interacts.
- Once the recording ends, a sequence is created, containing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

UiPath Studio recording is a technique that helps in capturing the manual actions of the user performed on the screen and translating them into sequences.

Recording is a crucial tool in UiPath Studio as it captures the step-by-step activity of the user. It automatically generates the sequence of all the workflows which are needed for automation. Recording helps in identifying the elements on the screen, reading data from display, and so on. Recording helps users to save a lot of time when automating the business processes.

All user interface elements are highlighted while recording to ensure that correct buttons, fields or menus are selected. The recorder can work with different types of applications, and different types of environments.

Slide 23



Recordable actions

In UiPath Studio, several user actions can be recorded such as:

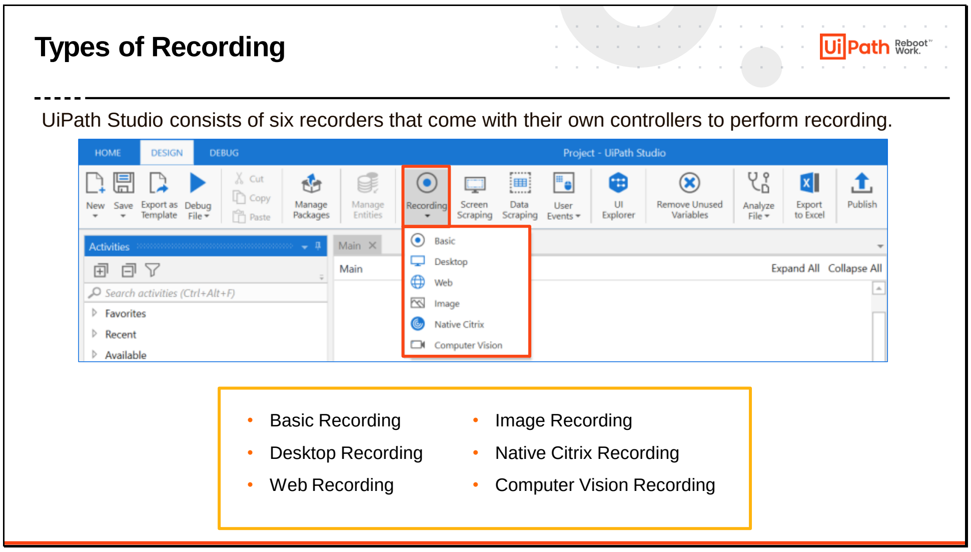
In UiPath Studio, there are several user actions that can be recorded. These include:

- Text typing
- Left clicks
- Drop-down lists
- Check boxes

The following **Keyboard shortcuts** are available for use while recording:

- F2 – pauses the recording for 3 seconds. A countdown timer is displayed in the bottom left corner of the screen. Can be useful with menus that automatically hide.
- Esc – exits the automatic or manual recording. If the Escape key is pressed again, the recording is saved as a sequence, and you return to the main view.
- Right-click – exit the recording.

Slide 24



Types of Recording

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

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

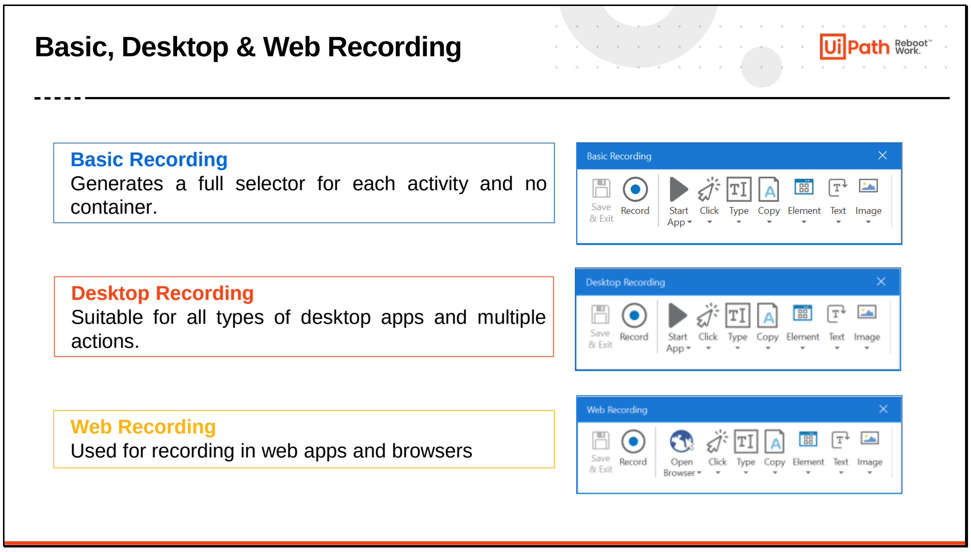
Recording:

In UiPath Studio, there are six types of recorders, and each comes with its own controllers to perform specific recording actions. These are:

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

These are discussed in detail in the subsequent slides.

Slide 25



Basic, Desktop & Web Recording

Basic Recording
Generates a full selector for each activity and no container.

Desktop Recording
Suitable for all types of desktop apps and multiple actions.

Web Recording
Used for recording in web apps and browsers

A. Basic Recording:
This recording generates a full selector for each activity with no container.

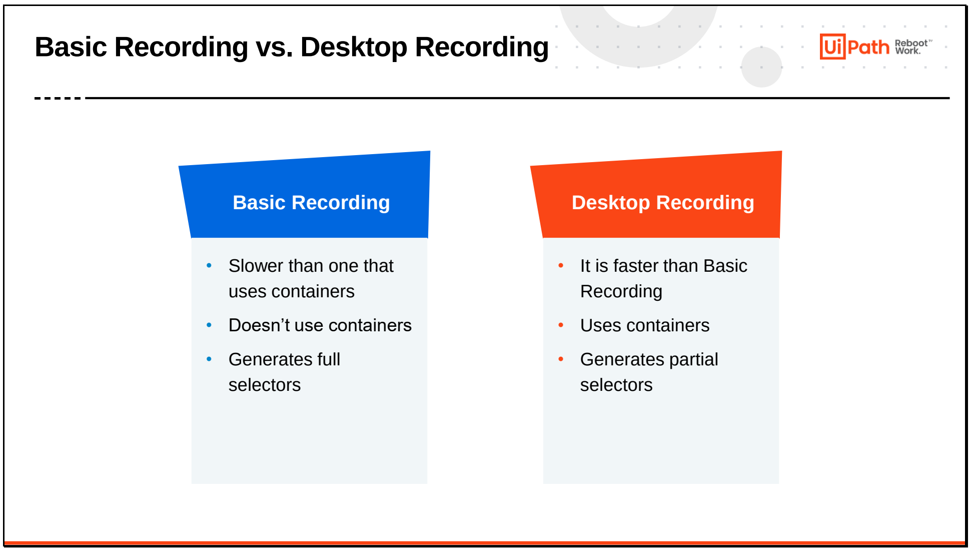
B. Desktop Recording:
This recording is suitable for all types of desktop apps.

C. Web Recording:
It is designed for recording in web apps and browsers, generates containers and uses the **Simulate Type/Click** input method by default.

All these recordings enable the user to:

- Automatically record multiple actions performed on the screen such as Click, Type Into, Select Item and Check.
- Manually record single actions, such as:
  - Starting or closing an application (Basic & Desktop recording) / Starting or closing web browser (Web recording)
  - Clicking an interface element
  - Selecting an option from a drop-down list
  - Selecting a check box
  - Simulating keystrokes or keyboard shortcuts
  - Copying text from a UI element or performing screen scraping
  - Looking for elements or waiting for them to vanish
  - Finding an image
  - Activating a window

Slide 26



Basic Recording vs. Desktop Recording

Basic Recording

- Slower than one that uses containers
- Doesn’t use containers
- Generates full selectors

Desktop Recording

- It is faster than Basic Recording
- Uses containers
- Generates partial selectors

Difference between the Basic recording and Desktop recording:

Basic Recording:

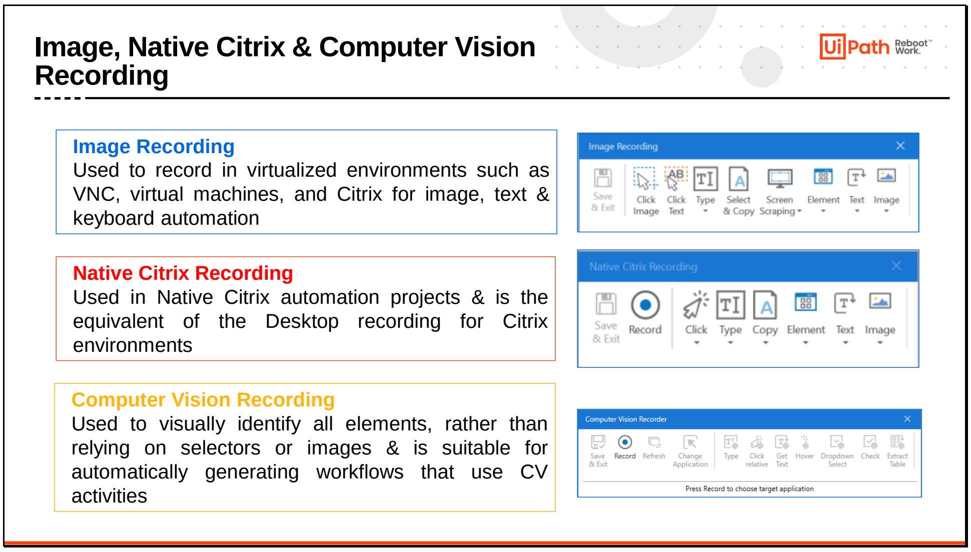
- It is slower than one that uses containers.
- It does not use containers which means other windows of the same application may interfere with the workflow.
- It generates full selectors.

In some cases, multiple windows are identical. Suppose the user opens two untitled notepad windows at the same time, then UiPath Studio will perform all actions inside the first opened window because the title of both notepad windows is the same (i.e., “untitled”). This may not be a good practice when working on files having identical names.

Desktop Recording:

- It is faster than Basic Recording.
- It uses containers to ensure that other windows of the same application don’t interfere with the workflow.
- It generates a partial selector for activities inside the Attach Window container.

Slide 27



Image, Native Citrix & Computer Vision Recording

Image Recording
Used to record in virtualized environments such as VNC, virtual machines, and Citrix for image, text & keyboard automation

Native Citrix Recording
Used in Native Citrix automation projects & is the equivalent of the Desktop recording for Citrix environments

Computer Vision Recording
Used to visually identify all elements, rather than relying on selectors or images & is suitable for automatically generating workflows that use CV activities

Press Record to choose target application

D. Image Recording:

It is used to record in virtualized environments (such as VNC, virtual machines, Citrix, and more) or SAP. It permits only image, text and keyboard automation, and requires explicit positioning. Image recording enables the user to:

- Click an image or text
- Simulate keystrokes or hotkeys
- Select and copy text from a window
- Scrape UI elements
- Look for elements
- Find an image or wait for it to vanish

E. Native Citrix Recording

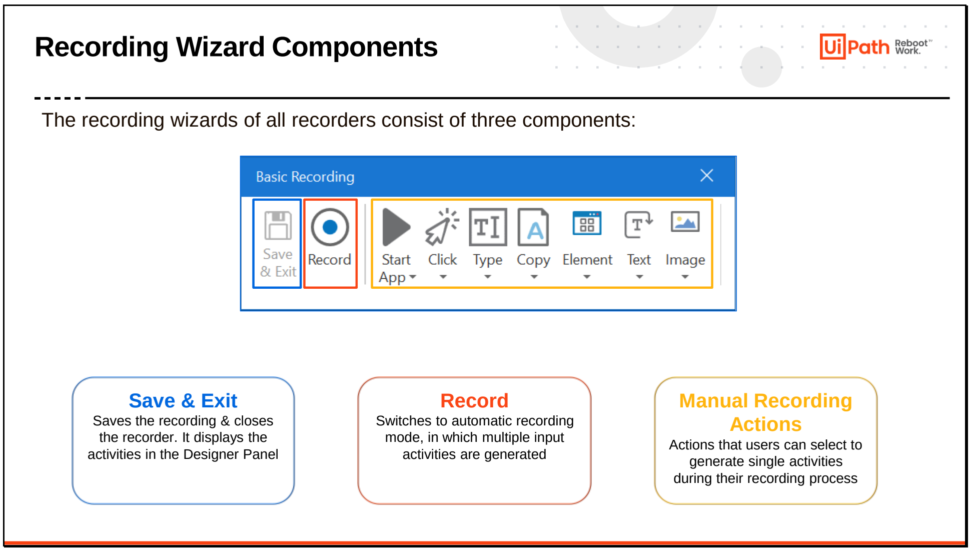
It is the equivalent of the Desktop recording, but for Citrix environments. This is used in Native Citrix automation projects to identify UI elements in Citrix Apps and automate them. This recording enables the user to automatically record multiple actions performed on the screen and manually record single actions, such as clicking an interface element, simulating keystrokes or keyboard shortcuts, copying text from a UI element or performing screen scraping, etc.

F. Computer Vision Recording

Computer Vision (CV) is a feature that allows Robots to “see” the screen and visually identify all the elements, rather than relying on selectors or images. It enables human-like recognition of user interfaces using a mix of Artificial Intelligence, OCR, fuzzy text-matching, a new image-matching algorithm for icons, and an anchoring system to tie it all together. Computer Vision solves the challenge of reliably automating Virtual Desktop Environments.
Once the user installs the Computer Vision activities pack, the Computer Vision Recorder wizard becomes available in the Ribbon. This recording is suitable for

automatically generating workflows that use the CV activities and offers the full spectrum of capabilities. It enables the user to indicate the application you want to automate, which is used as a target for all subsequent CV activities. Once an application is indicated, the selected window is sent to the CV server, where it is processed, and the wizard enters recording mode.

Slide 28



Recording Wizard Components

The recording wizards of all recorders consist of three components:

Save & Exit
Saves the recording & closes the recorder. It displays the activities in the Designer Panel

Record
Switches to automatic recording mode, in which multiple input activities are generated

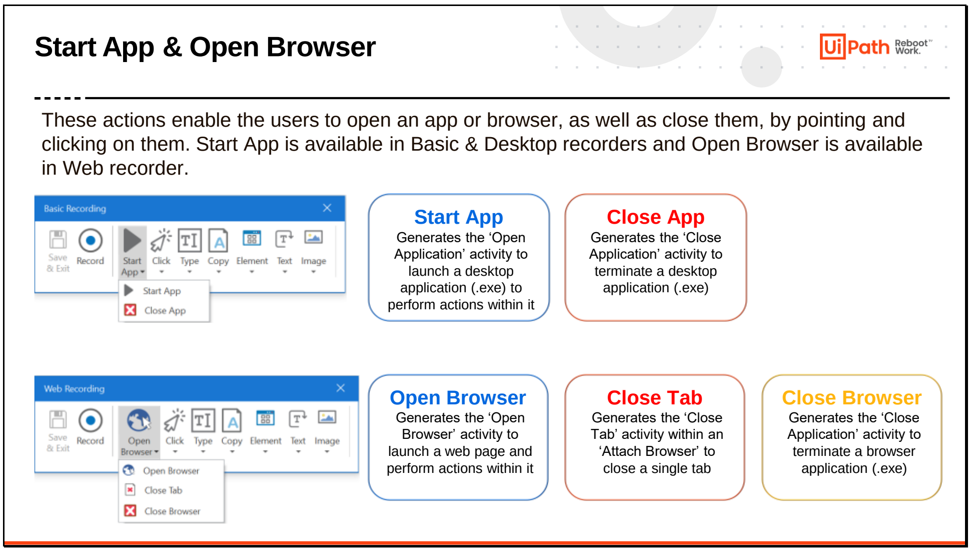
Manual Recording Actions
Actions that users can select to generate single activities during their recording process

Components of recording wizard in UiPath Studio:

The recording wizards of all recorders consist of three components:

- **Save & exit**: This option is enabled only when the recorder is on. It closes the recorder and reverts to the 'designer' window, displaying the automatically generated activities.
- **Record**: It starts the automatic recording mode and captures user actions. On completion, multiple input activities are generated. The user will see a blue screen over the area where the action is captured and yellow outlines for selectors.
- **Manual recording actions**: The remaining actions in the wizard are manual recording actions. These may be common or different for different recorders. These are discussed in the coming slides.

Slide 29



Start App & Open Browser

These actions enable the users to open an app or browser, as well as close them, by pointing and clicking on them. Start App is available in Basic & Desktop recorders and Open Browser is available in Web recorder.

Start App
Generates the 'Open Application' activity to launch a desktop application (.exe) to perform actions within it

Close App
Generates the 'Close Application' activity to terminate a desktop application (.exe)

Open Browser
Generates the 'Open Browser' activity to launch a web page and perform actions within it

Close Tab
Generates the 'Close Tab' activity within an 'Attach Browser' to close a single tab

Close Browser
Generates the 'Close Application' activity to terminate a browser application (.exe)

Start App is available for Basic & Desktop Recording. It has two types of activities:

- **Start app** is used to open the application. It is similar to the 'Open Application' activity of UiPath Studio. To use this, select **"Start App"** and then click on the desired application to open it.

- **Close app** is used to close the already open application. It is similar to the 'Close Application' activity of UiPath Studio. To use this, select **"Close App"** and then click on the desired application to close it.

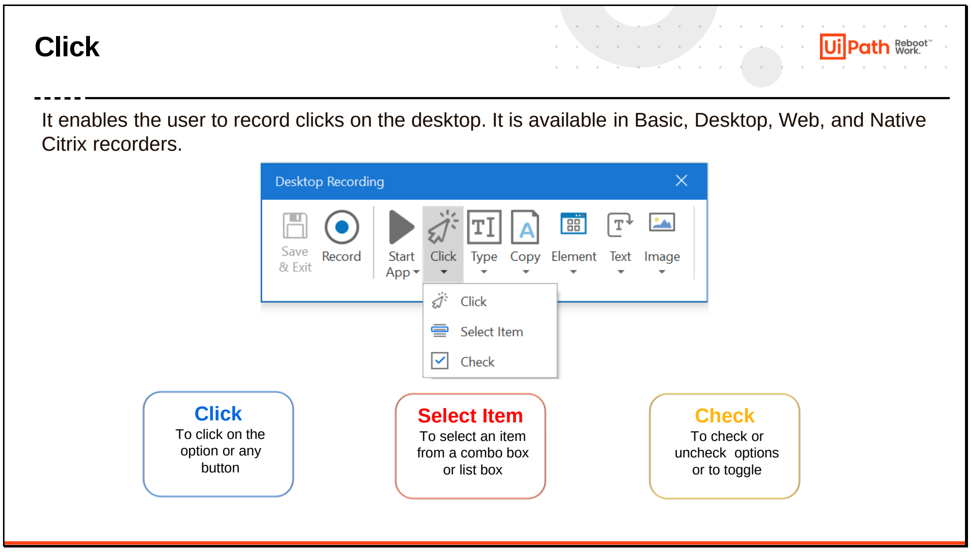
Open Browser is available for Web Recording. It has three activities:

- **Open Browser:** This is used to open a browser. The browser can be chosen (for example Internet Explorer, Firefox or Chrome) or the URL can be specified inside the quotation marks "****", or the user can store the URL inside the string variable and pass that inside it.

- **Close Tab:** Closes the Browser tab specified by the user.

- **Close Browser:** Closes the already open browser window.

Slide 30



Click

It enables the user to record clicks on the desktop. It is available in Basic, Desktop, Web, and Native Citrix recorders.

Click
To click on the option or any button

Select Item
To select an item from a combo box or list box

Check
To check or uncheck options or to toggle

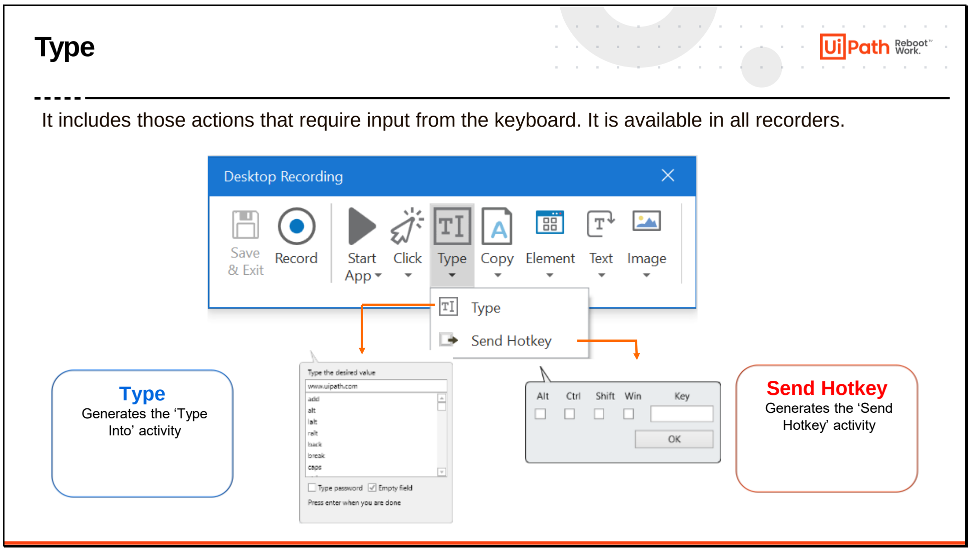
Click: Enables the user to record clicks on the desktop or a running application, select an option from a drop-down list or combo box, and select a check box or a radio button.

It is available in Basic, Desktop, Web, and Native Citrix recorders. The activity is supported by the automatic recorder and it doesn't recognize right clicks.

There are 3 options in the 'click' activity.

- **Click -** This activity is used to click on the option or any button.
- **Select Item -** This activity is used to select an item from a combo box or list box.
- **Check -** It is used to check or uncheck options or to toggle.

Slide 31



Type

It includes those actions that require input from the keyboard. It is available in all recorders.

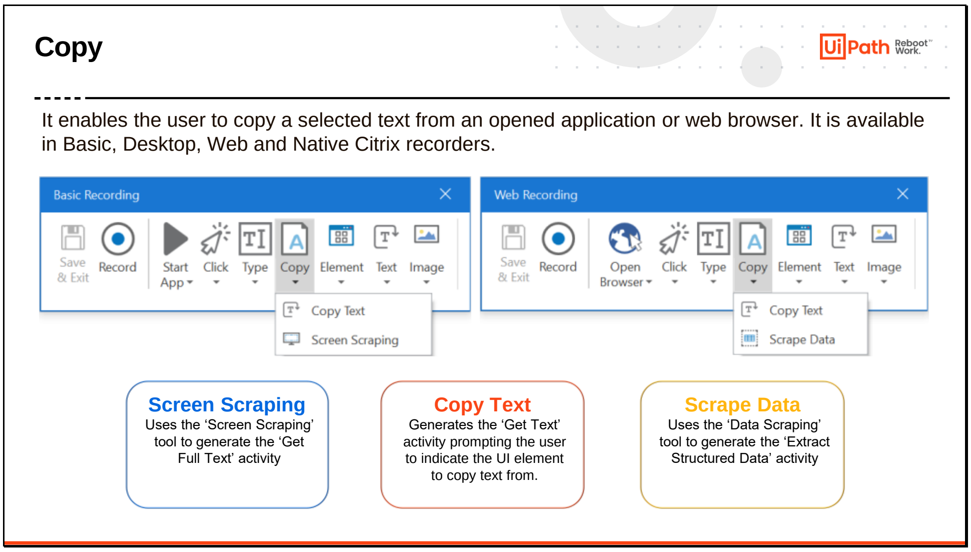
Type
Generates the 'Type Into' activity

Send Hotkey
Generates the 'Send Hotkey' activity

Type: This includes those actions that require input from the keyboard, such as keyboard shortcuts and key presses. To achieve this, two pop-up windows are used to retrieve the keyboard input.

- **Type:** generates the 'Type Into' activity prompting the user for their desired value whether it is text or special keys.
- **Send Hotkey:** generates the 'Send Hotkey' activity prompting to define an Alt-, Ctrl-, Shift-, or Win + Key to send it to an application.

Slide 32



Copy

It enables the user to copy a selected text from an opened application or web browser. It is available in Basic, Desktop, Web and Native Citrix recorders.

Screen Scraping
Uses the 'Screen Scraping' tool to generate the 'Get Full Text' activity

Copy Text
Generates the 'Get Text' activity prompting the user to indicate the UI element to copy text from.

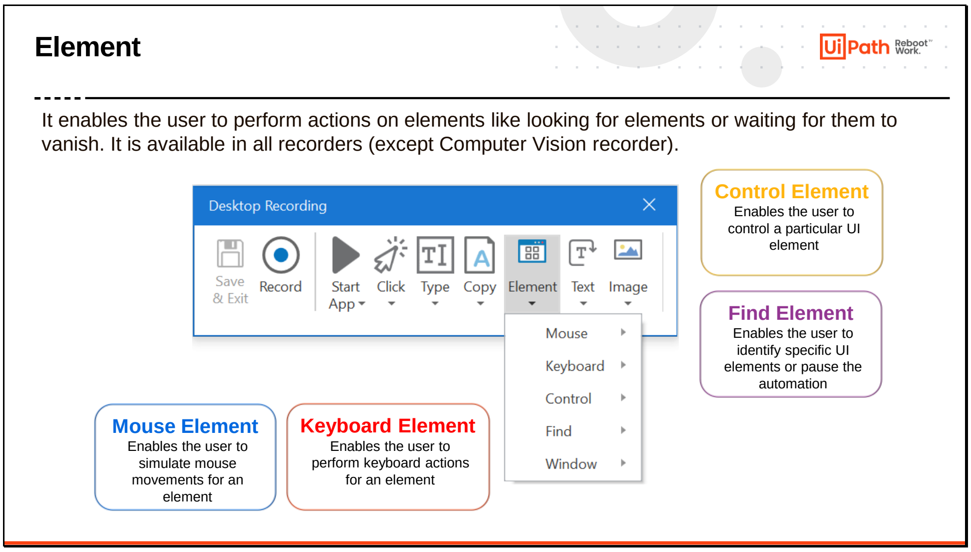
Scrape Data
Uses the 'Data Scraping' tool to generate the 'Extract Structured Data' activity

Copy: Enables the user to copy a selected text from an opened application or web browser, so that it can be used later in the project. Screen scraping is also available under the **Copy** menu, as it enables the user to extract images and text from an app or browser.

The Copy activity contains:

- Copy text (same in all three recorders): Generate Get Text activity & recognize the text and copy it for further use as it is stored in a variable.
- Screen Scraping (in Basic, Desktop and Native Citrix Recorder). It is used to read the data from the screen. It is capable of scraping both visible and non-visible UI elements from the screen.
- Scrape Data (in Web Recorder): It allows us to extract any structured data from the web, any application, document, spreadsheet, etc.

Slide 33



Element

It enables the user to perform actions on elements like looking for elements or waiting for them to vanish. It is available in all recorders (except Computer Vision recorder).

Control Element
Enables the user to control a particular UI element

Find Element
Enables the user to identify specific UI elements or pause the automation

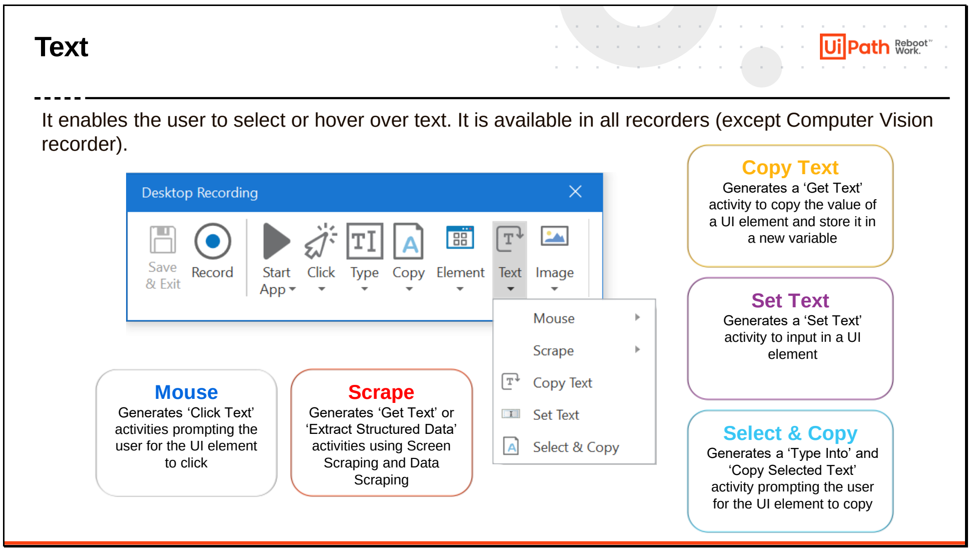
Mouse Element
Enables the user to simulate mouse movements for an element

Keyboard Element
Enables the user to perform keyboard actions for an element

Element:

- **Mouse Element**: Enables the user to simulate mouse movements for an element that cannot be recorded but give access to more functionalities, such as right-clicking, hovering or double-clicking.
- **Keyboard Element**: Enables the user to perform keyboard actions for an element. It consists of Type (generates the ‘Type Into’ activity) and Send Hotkey (generates the ‘Send Hotkey’ activity) options.
- **Control Element**: Enables the user to control a particular UI element. It consists of Select Item and Check options.
- **Find Element**: Enables the user to identify specific UI elements or pause the automation until a particular window closes or an UI element is no longer displayed.
- **Window Element**: Enables the user to close windows. Studio does this by hooking in the operating system to make sure the application is closed.

Slide 34



Text

It enables the user to select or hover over text. It is available in all recorders (except Computer Vision recorder).

Copy Text Generates a 'Get Text' activity to copy the value of a UI element and store it in a new variable

Set Text Generates a 'Set Text' activity to input in a UI element

Select & Copy Generates a 'Type Into' and 'Copy Selected Text' activity prompting the user for the UI element to copy

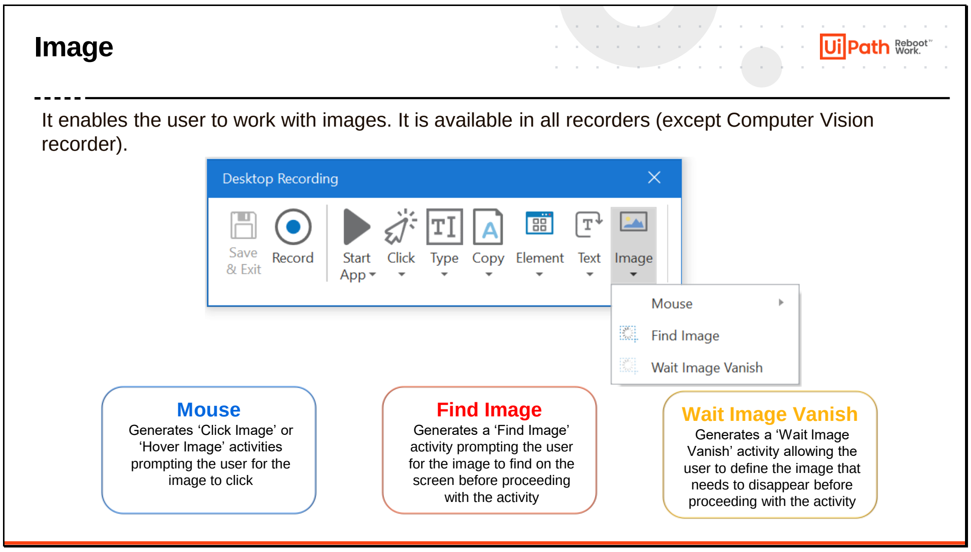
Mouse Generates 'Click Text' activities prompting the user for the UI element to click

Scrape Generates 'Get Text' or 'Extract Structured Data' activities using Screen Scraping and Data Scraping

Text: Enables the user to select or hover over text to make tooltips visible for scraping, right-click to make the context menu visible, copy and paste text and many others. It is available in all recorders and is known as Get Text in Computer Vision recording.

- **Mouse:** It includes 4 categories for different methods of using the mouse in relation to a text:
    1. Click
    2. Double-click
    3. Right-click
    4. Hover
- **Scrape**: Used for data scraping. It consists of three categories:
    1. Screen Scraping
    2. Data Scrape
    3. Scrape Relative
- **Copy Text:** This is the same activity as the one created under the Copy category.
- **Set Text:** This activity is used to identify a UI element and send a text to it. It is very similar to Type Into, the main difference between both of them is that you can assign keystrokes in Type Into, but in Set Text, you cannot do that.
- **Select & Copy:** This is used to generate two activities with a single step:
    - Select a text in UI element
    - Copy the text and store it in another variable

Slide 35



Image

It enables the user to work with images. It is available in all recorders (except Computer Vision recorder).

Mouse
Generates 'Click Image' or 'Hover Image' activities prompting the user for the image to click

Find Image
Generates a 'Find Image' activity prompting the user for the image to find on the screen before proceeding with the activity

Wait Image Vanish
Generates a 'Wait Image Vanish' activity allowing the user to define the image that needs to disappear before proceeding with the activity

Image: Enables the user to wait for a specified image to disappear, to find a specific image in an app or website, right-click or hover over an image and others (Useful with UI elements that are saved as graphics)

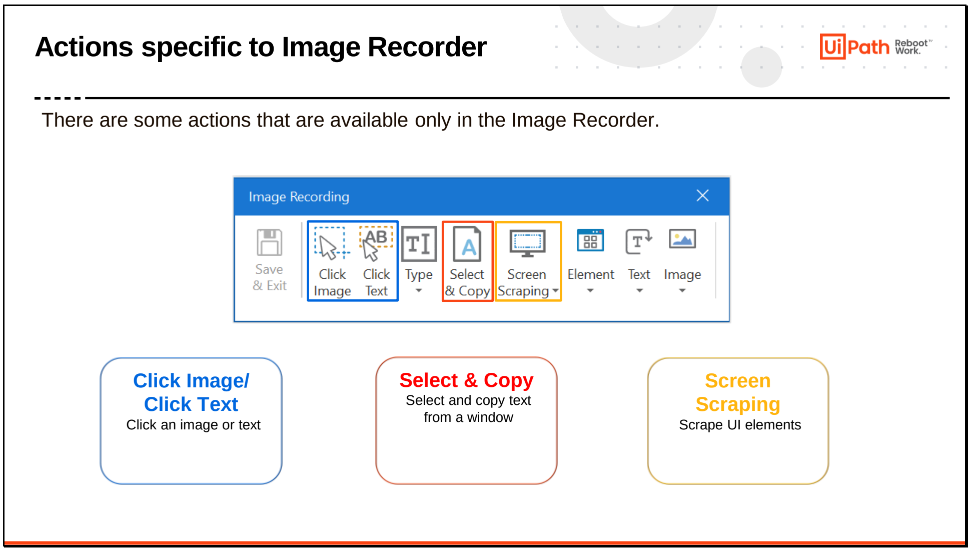
It brings together several events to recognize images and treat them as triggers:

- **Mouse:** The Mouse sub-category contains different methods of using the mouse in relation to images, click, double-click, right-click and hover. By using these properties, these activities can be customized to use modifier keys.

- **Find Image:** It is used to find the image on the screen and returns a Boolean value in the output.

- **Wait Image Vanish:** It waits for the image to disappear from the screen, another activity starts when the image is inactive or invisible from the screen. This activity also has a 30 seconds timeout.

Slide 36



Actions specific to Image Recorder

There are some actions that are available only in the Image Recorder.

Click Image/ Click Text
Click an image or text

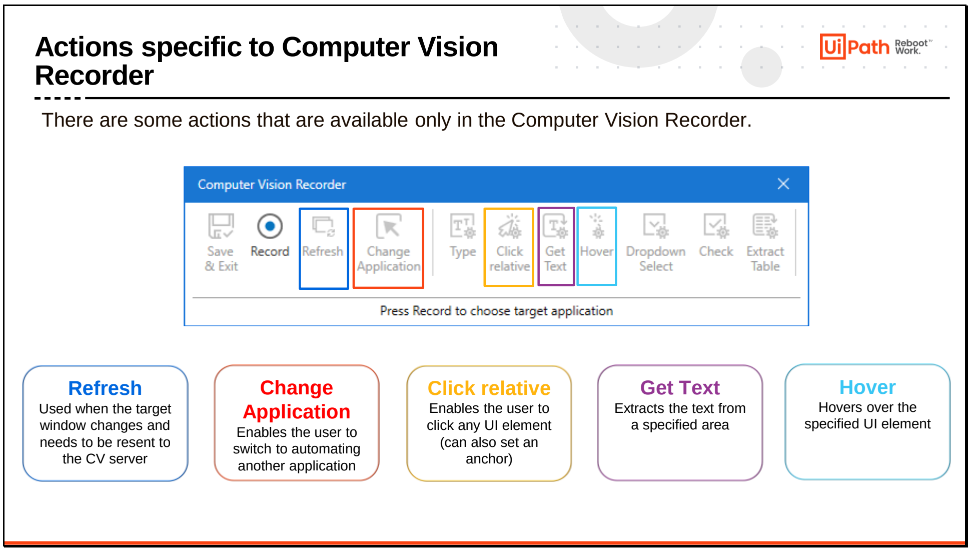
Select & Copy
Select and copy text from a window

Screen Scraping
Scrape UI elements

There are some actions that are available only in the Image Recorder. These are:

- **Click Image/Click Text**: Click an image or text
- **Select & Copy**: Select and copy text from a window
- **Screen Scraping**: Scrape UI elements

Slide 37



Actions specific to Computer Vision Recorder

There are some actions that are available only in the Computer Vision Recorder.

Refresh
Used when the target window changes and needs to be resent to the CV server

Change Application
Enables the user to switch to automating another application

Click relative
Enables the user to click any UI element (can also set an anchor)

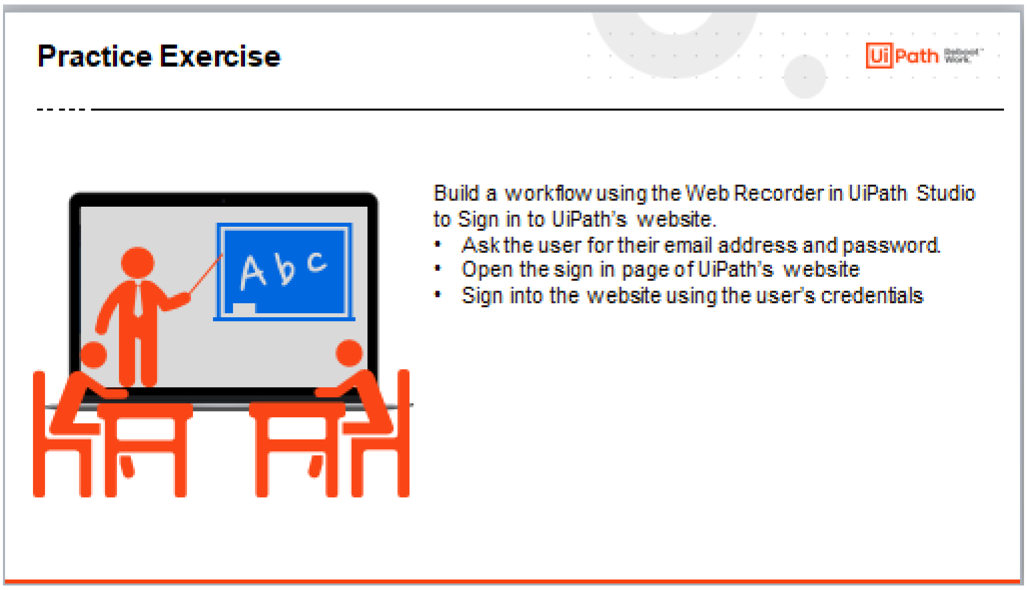
Get Text
Extracts the text from a specified area

Hover
Hovers over the specified UI element

There are some actions that are available only in the Computer Vision Recorder. These are:

- **Refresh**: Used when the target window changes and needs to be resent to the CV server. This can happen after clicking a button that enters another section of the application.
- **Change Application**: Enables the user to indicate another application window to switch to automating another application.
- **Click relative**: Enables the user to click any UI element. If necessary, the wizard can also set an anchor for the area the user wants to click.
- **Get Text**: Extracts the text from a specified area. This action can also request an anchor if necessary.
- **Hover**: Hovers over the specified UI element.

Slide 38



Practice Exercise

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

- Ask the user for their email address and password.
- Open the sign in page of UiPath's website
- Sign into the website using the user's credentials

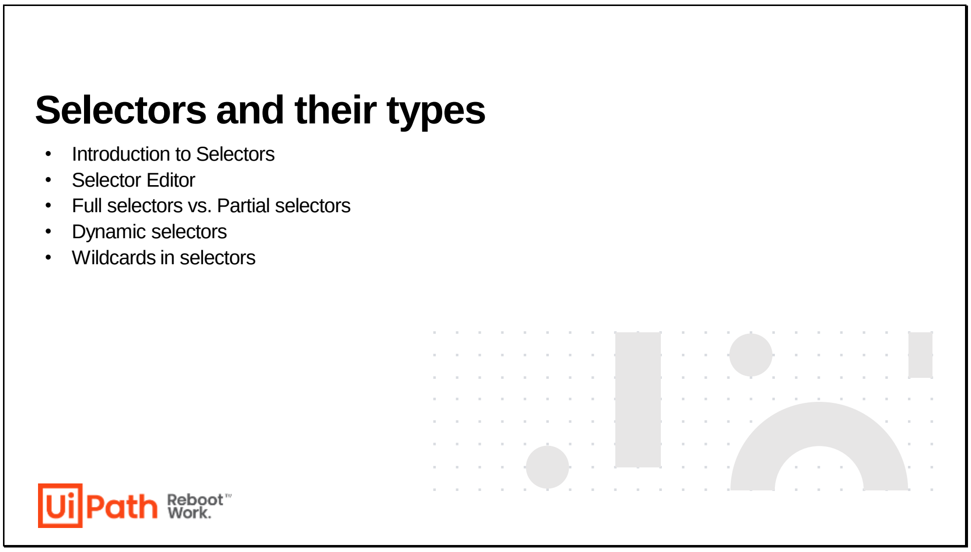
Build a workflow using the Web Recorder in UiPath Studio to Sign in to UiPath's website.

- Ask the user for their email address and password.
- Open the sign in page of UiPath's website
- Sign into the website using the user's credentials

Process Overview

- START
- Use two Input Dialog activities and take the email address and password from the user.
- Store received input in two variables.
- Use an Open Browser activity and open URL – "www.uipath.com".
- Use the Web Recorder in UiPath Studio to:
  - Click the *Try UiPath Free* button.
  - Click the *Sign in* the link on the next page.
  - Type in the user's email address and password.
  - Click on the *Sign in* button.
- STOP

Slide 39



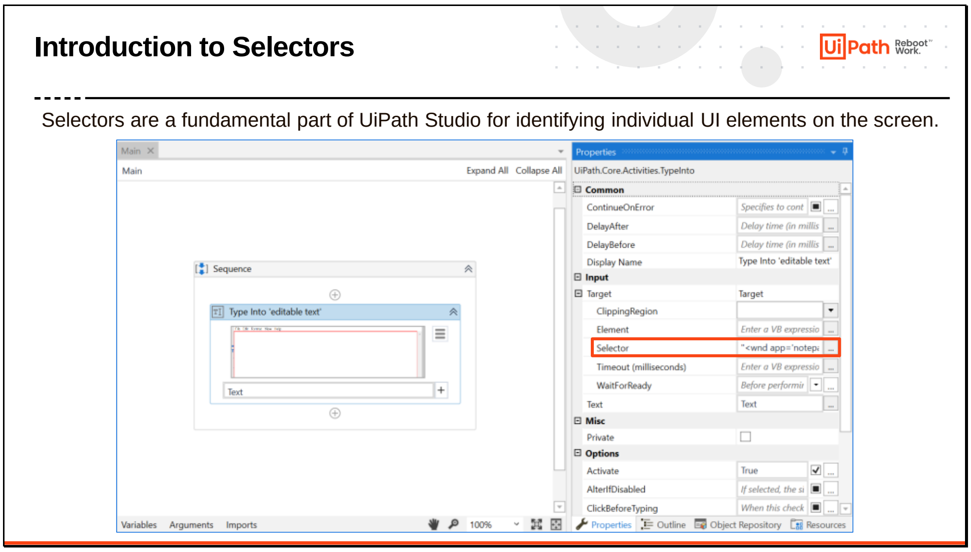
Selectors and their types

- Introduction to Selectors
- Selector Editor
- Full selectors vs. Partial selectors
- Dynamic selectors
- Wildcards in selectors

Reboot™ Work.

This section explains Selectors and their types.

Slide 40



Introduction to Selectors

Selectors are a fundamental part of UiPath Studio for identifying individual UI elements on the screen.

UiPath.Core.Activities.TypeInto

Common	
ContinueOnError	Specifies to cont
DelayAfter	Delay time (in millis
DelayBefore	Delay time (in millis
Display Name	Type Into 'editable text'
Input	
Target	Target
ClippingRegion	
Element	Enter a VB expressio
Selector	"<wnd app='notepa
Timeout (milliseconds)	Enter a VB expressio
WaitForReady	Before performir
Text	Text
Misc	
Private	□
Options	
Activate	True ☑
AlterIfDisabled	If selected, the si
ClickBeforeTyping	When this check

Selectors are a fundamental part of UiPath Studio as they are used to recognize the UI elements on the user screen. A user is required to interact with various windows, buttons, drop-down lists and others on a machine to automate specific actions. This can be done by identifying the position of the element on the screen. However, this can be unreliable. Therefore, UiPath uses selectors. These store the attributes of a graphical user interface element and its parents for improved accuracy. In general, selectors are the address of a UI Element. When a specific task or activity is automated, the user needs to communicate with various UI Elements. The selector is a specific identifier to find a UI Element among multiple applications.

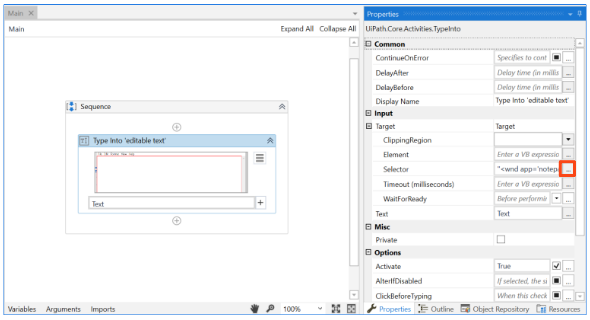
Slide 41

Selector Editor

Enables the user to see the automatically generated selectors and edit their attributes.

To access this window:

- Go to the Designer panel
- Click on the activity whose selector is to be edited
- In the Properties panel of the activity, click the ellipsis button of the Selector property



Properties — UiPath.Core.Activities.TypeInto

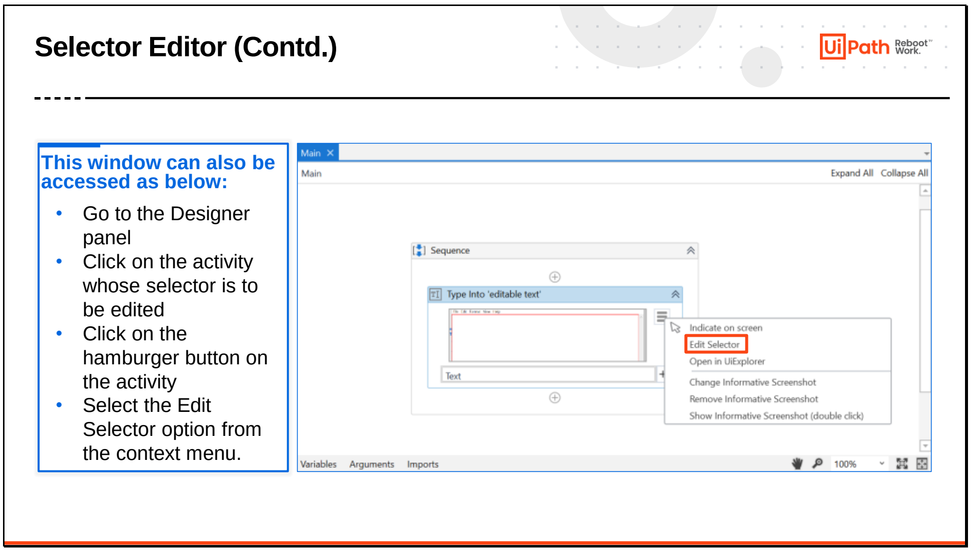
Common	
ContinueOnError	Specifies to cont
DelayAfter	Delay time (in millis
DelayBefore	Delay time (in millis
Display Name	Type Into 'editable text'
Input	
Target	Target
ClippingRegion	
Element	Enter a VB expressio
Selector	"<wnd app='notep
Timeout (milliseconds)	Enter a VB expressio
WaitForReady	Before performir
Text	Text
Misc	
Private	□
Options	
Activate	True ☑
AlterIfDisabled	If selected, the si
ClickBeforeTyping	When this check

Selector Editor enables the user to access the automatically generated selectors and edit their attributes.

To access the Selector Editor window, follow these steps:

- Go to the Designer panel
- Click on the activity whose selector is to be edited
- In the Properties panel of the activity, click the ellipsis button of the Selector property

Slide 42



Selector Editor (Contd.)

This window can also be accessed as below:

- Go to the Designer panel
- Click on the activity whose selector is to be edited
- Click on the hamburger button on the activity
- Select the Edit Selector option from the context menu.

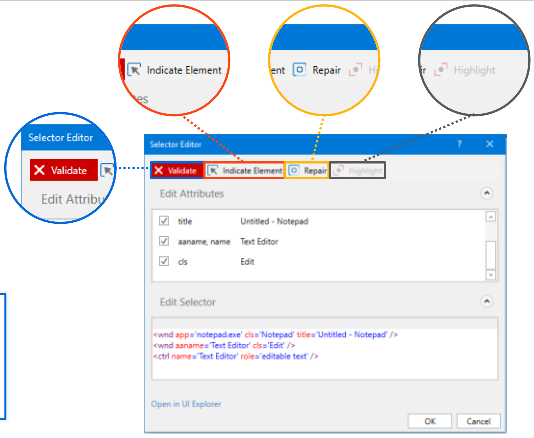
Main
Main
Expand All
Collapse All
Sequence
Type Into 'editable text'
Text
Indicate on screen
Edit Selector
Open in UiExplorer
Change Informative Screenshot
Remove Informative Screenshot
Show Informative Screenshot (double click)
Variables
Arguments
Imports
100%

The Selector Editor window can also be accessed through the below steps:

- Go to the Designer panel
- Click on the activity whose selector is to be edited
- Click on the hamburger button on the activity
- Select the Edit Selector option from the context menu.

Slide 43

Properties of Selector Editor



☑ title Untitled - Notepad
☑ aaname, name Text Editor
☑ cls Edit

```
<wnd app='notepad.exe' cls='Notepad' title='Untitled - Notepad' />
<wnd aaname='Text Editor' cls='Edit' />
<ctrl name='Text Editor' role='editable text' />
```

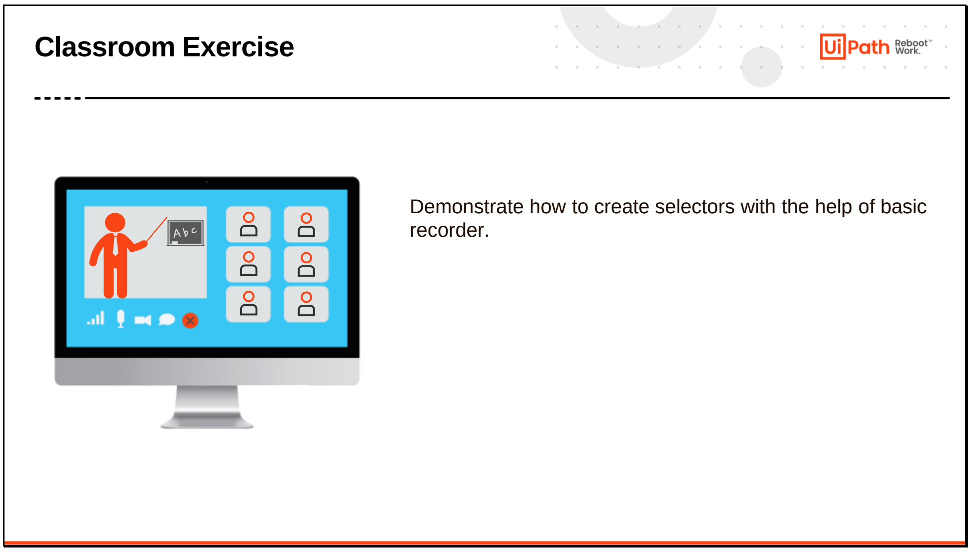
- Validate
- Indicate Element
- Repair
- Highlight

Properties of Selector Editor:

- **Validate**: 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.
  The Validate button has the following states:
  - O (buffer): Selector is being validated
  - √: Valid selector
  - ×: Invalid selector
  - ?: Modified selector, revalidate

  The button is correlated with UI Explorer validation states.
- **Indicate Element:** Indicates a new UI element to replace the previous one.
- **Repair**: Enables the user to re-indicate the same target UI element and repair the selector. The button is available only when the selector is invalid.
- **Highlight**: Brings the target element in the foreground. The highlight stays on until it's switched off. The button is enabled only if the selector is valid.

Slide 44



Classroom Exercise

Demonstrate how to create selectors with the help of basic recorder.

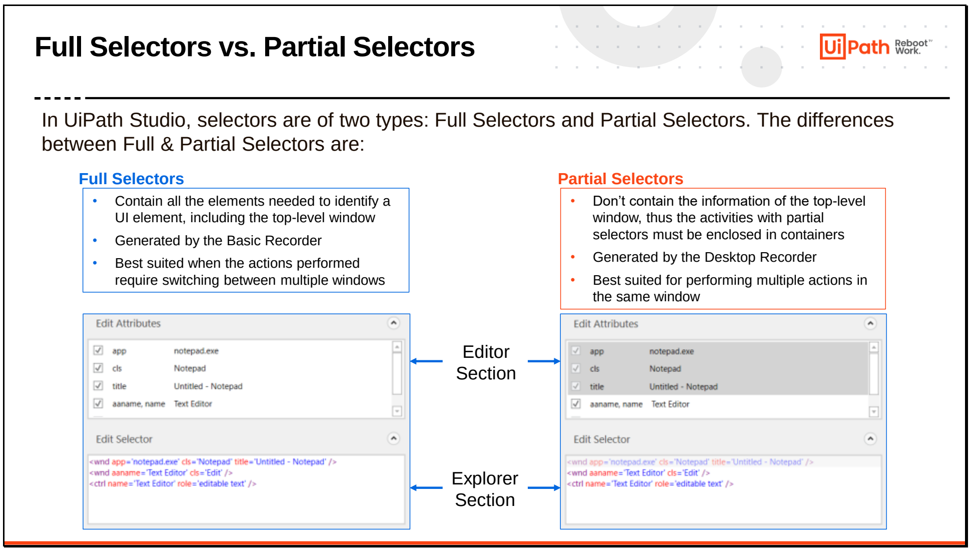
Demonstrate how to create selectors with the help of a basic recorder.

- Open a Notepad window
- In UiPath Studio, click Recording from the Design tab, and select Basic from the list.
- From the Basic Recording toolbar, click Type from the Type drop-down menu. Select notepad editor.
- Enter the text “Automation increases efficiency” in the small pop-up window, and press enter on your keyboard.
- Click the Type drop-down menu and select Send Hotkey. Select the notepad editor.
- From the small pop-up window select Ctrl and enter A as the Key.
- Click OK.
- Select the Click drop-down menu and select Click.
- Select Format from the Notepad menu items.
- In the Use Anchor pop-up window, select the check box, and click No.
- Select Click from the Basic Recording window.
- Press F2 to pause UiPath Studio for 3 seconds.
- Navigate to the Format dropdown of the Notepad menu item and wait for the countdown to go 0.
- Click Font after the countdown ends.
- From the Type drop-down menu of the Basic Recording window, select Type, and select the text area of the Font section.
- Enter a font name Century, check the Empty field, and press enter on your keyboard.
- Repeat this process for the Font style section. Enter the font style name as Bold, check the Empty field, and press enter on your keyboard.

- Again, repeat this process for the Size section. Enter size value as 12, check the Empty field, and press enter on your keyboard.
- Select Click from the Basic Recording window and select the OK button within the Font window.
- Click Save & Exit from the Basic Recording window.
    - Recording session has generated a Sequence of activities in the Designer panel.
- Select the first Type Into activity and open its Select Editor window.
    - Uncheck the title attribute from the window and click Validate and OK.
- Run the process.

Outcome: Process executes successfully as recorded.

Slide 45



Full Selectors vs. Partial Selectors

In UiPath Studio, selectors are of two types: Full Selectors and Partial Selectors. The differences between Full & Partial Selectors are:

Full Selectors

- Contain all the elements needed to identify a UI element, including the top-level window
- Generated by the Basic Recorder
- Best suited when the actions performed require switching between multiple windows

Partial Selectors

- Don't contain the information of the top-level window, thus the activities with partial selectors must be enclosed in containers
- Generated by the Desktop Recorder
- Best suited for performing multiple actions in the same window

Edit Attributes

☑	app	notepad.exe
☑	cls	Notepad
☑	title	Untitled - Notepad
☑	aaname, name	Text Editor

Edit Selector

```
<wnd app='notepad.exe' cls='Notepad' title='Untitled - Notepad' />
<wnd aaname='Text Editor' cls='Edit' />
<ctrl name='Text Editor' role='editable text' />
```

Edit Attributes

☑	app	notepad.exe
☑	cls	Notepad
☑	title	Untitled - Notepad
☑	aaname, name	Text Editor

Edit Selector

```
<wnd app='notepad.exe' cls='Notepad' title='Untitled - Notepad' />
<wnd aaname='Text Editor' cls='Edit' />
<ctrl name='Text Editor' role='editable text' />
```

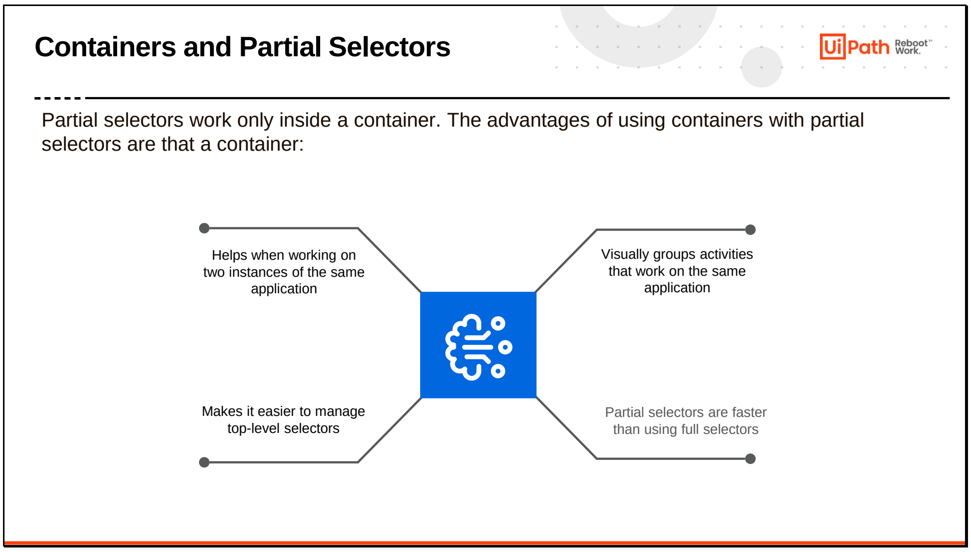
In UiPath Studio, selectors are of two types: Full Selectors and Partial Selectors.

Full and partial selectors are discussed here.

- When automation is created through Basic Recorder, **Full Selectors** are generated. Full selectors start with a window, or an HTML identifier and they contain all the information of the UI element. They are suitable when the Robot has to switch through multiple windows and the use of containers would add unnecessary complexity.

- When automation is created through Desktop Recorder, **Partial selectors** are generated. They do not contain the top-level window information. They are more suitable when the robot has to perform multiple actions in the same window or application. In these cases, the activities will be placed inside a container, such as Attach Browser or Attach Window.

Slide 46



Containers and Partial Selectors

Partial selectors work only inside a container. The advantages of using containers with partial selectors are that a container:

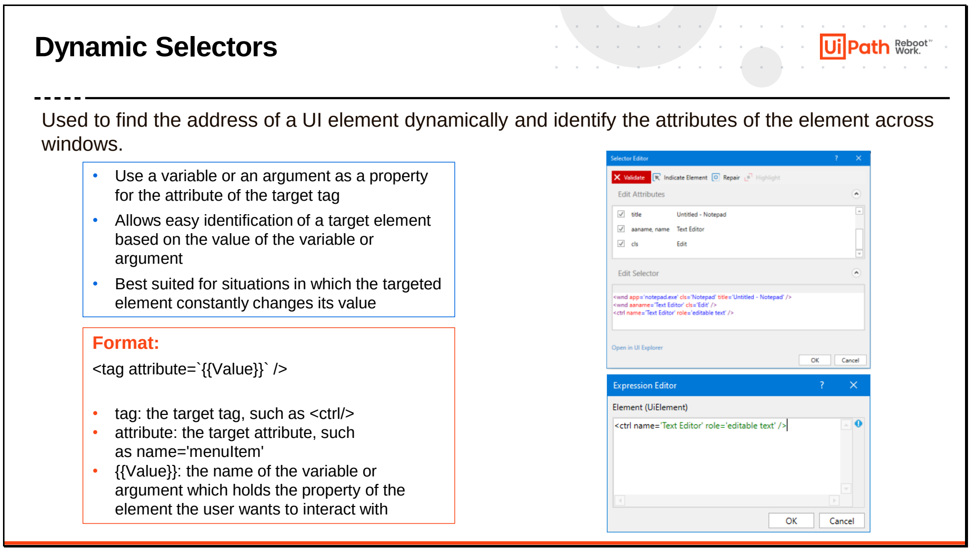
Helps when working on two instances of the same application
Visually groups activities that work on the same application
Makes it easier to manage top-level selectors
Partial selectors are faster than using full selectors

Partial selectors work only inside a container that specifies the top-level window where the elements belong.

The advantages to using containers with partial selectors (instead of full selectors) are:

- Visually groups activities that work on the same application.
- Is slightly faster, not seeking the top window every time.
- Makes it easier to manage top-level selectors in case manual updates are necessary.
- Helps (and is essential) when working on two instances of the same application.

Slide 47



Dynamic Selectors

Used to find the address of a UI element dynamically and identify the attributes of the element across windows.

- Use a variable or an argument as a property for the attribute of the target tag
- Allows easy identification of a target element based on the value of the variable or argument
- Best suited for situations in which the targeted element constantly changes its value

Format:
<tag attribute=`{{Value}}` />

- tag: the target tag, such as <ctrl/>
- attribute: the target attribute, such as name='menuItem'
- {{Value}}: the name of the variable or argument which holds the property of the element the user wants to interact with

☑	title	Untitled - Notepad
☑	aaname, name	Text Editor
☑	cls	Edit

```
<wnd app='notepad.exe' cls='Notepad' title='Untitled - Notepad' />
<wnd aaname='Text Editor' cls='Edit' />
<ctrl name='Text Editor' role='editable text' />
```

Element (UiElement)

```
<ctrl name='Text Editor' role='editable text' />
```

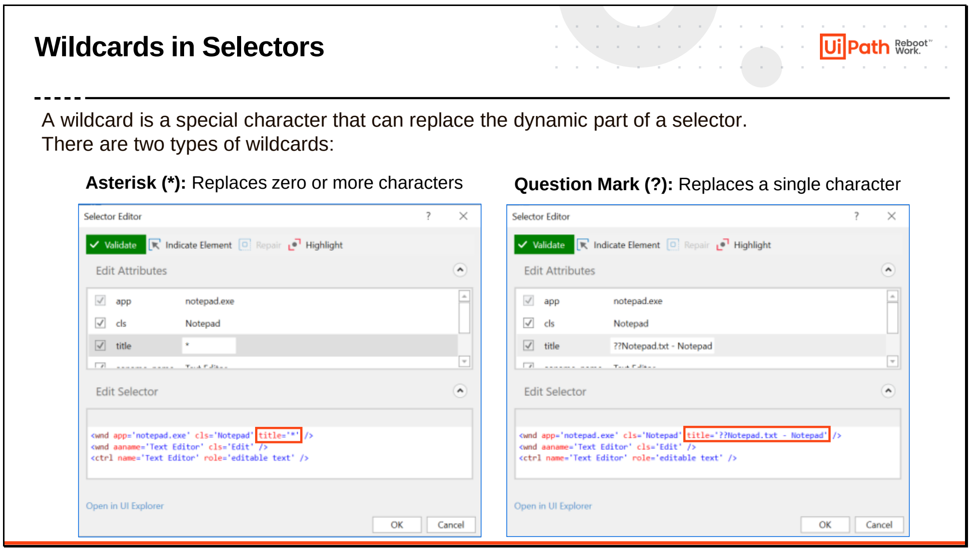
A selector helps to capture the attributes of a particular UI Element on the screen. If the same UI Element is present on another window, then a dynamic selector is used to easily identify its attributes on this window using a variable or an argument.
A dynamic selector uses a variable or an argument as a property for the attribute of the target tag. This allows the selector to easily identify a target element based on the value of the variable or argument. This is best suited for the situations in which the targeted element constantly changes its value as a dynamic selector does not identify the target element based on an exact string, which might change, depending on interactions inside the automation project.
The variable or argument can be changed to interact with a different element, without changing the selector itself.

Format:
<tag attribute=`{{Value}}` />

- tag: the target tag, such as <ctrl/>
- attribute: the target attribute, such as name='menuItem'
- {{Value}}: the name of the variable or argument which holds the property of the element the user wants to interact with

Example: For a calendar on a web page, a user wants to click a specific date. To do this, the user can use the dynamic selector because there is a difference in the location of that particular date due to which selectors attribute may change. So, few changes are required in the attributes to make the selectors of the date to be clicked as dynamic.

Slide 48



Wildcards in Selectors

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

Asterisk (*): Replaces zero or more characters

☑	app	notepad.exe
☑	cls	Notepad
☑	title	*

```
<wnd app='notepad.exe' cls='Notepad' title='*' />
<wnd aaname='Text Editor' cls='Edit' />
<ctrl name='Text Editor' role='editable text' />
```

Question Mark (?): Replaces a single character

☑	app	notepad.exe
☑	cls	Notepad
☑	title	??Notepad.txt - Notepad

```
<wnd app='notepad.exe' cls='Notepad' title='??Notepad.txt - Notepad' />
<wnd aaname='Text Editor' cls='Edit' />
<ctrl name='Text Editor' role='editable text' />
```

Wildcards are symbols that enable the user to replace zero or multiple characters in a string when dealing with dynamically changing attributes in a selector. When a wildcard is used or a variable is added in between selectors, it is known as customizing selectors.
Types:

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

With the help of the below mentioned example, we can understand how to use wildcards while building selectors.

- Open a Notepad file
- Create a new Sequence, give it a proper name and add an annotation
- Use an ‘Attach Window’ and indicate the Notepad file that is opened. Inside the container:
  - Use a ‘Send hotkey’ to send Ctrl + h in order to replace 2 spaces with one
  - Use 2 ‘Type into’ activities – one for the first field (with 2 spaces), and another for the second field (with 1 space)
  - Use 2 ‘Click’ activities, one for the ‘Replace all’ button and the other for ‘Close’
- Open the second Notepad file and run the workflow. It will return the ‘Cannot find element’ error
- Open the selector for the ‘Attach window’ container. Point out that the title attribute contains the name of the first file
- Use the repair function in the Selector Editor and point out that the part of the file name that was different has been replaced by “*”
- Open the third file and re-run the workflow. If it returns another error, replace the entire file name by “*”

The question mark can also be used as a wildcard in a similar manner as the above example.

To know more, visit: https://docs.uipath.com/studio/v2020.10/docs/selectors-with-wildcards

Slide 49

UI Explorer

- Introduction to UI Explorer
- Panels in UI Explorer
- Fields in UI Explorer

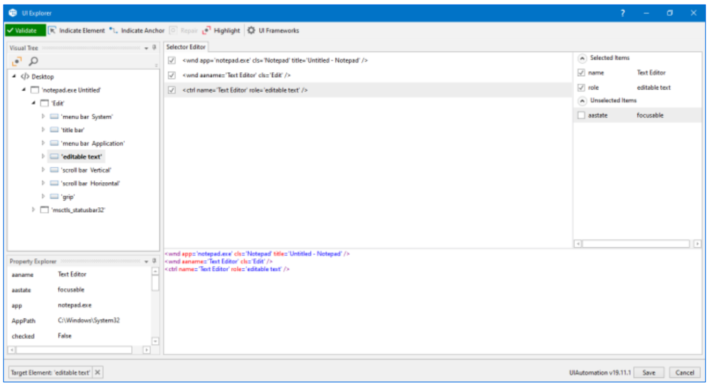
UiPath Reboot Work.

This section gives an overview of UI Explorer.

Slide 50

Introduction to UI Explorer

Used to customize the selector.



```
☑ <wnd app='notepad.exe' cls='Notepad' title='Untitled - Notepad' />
☑ <wnd aaname='Text Editor' cls='Edit' />
☑ <ctrl name='Text Editor' role='editable text' />
```

☑	name	Text Editor
☑	role	editable text
□	aastate	focusable

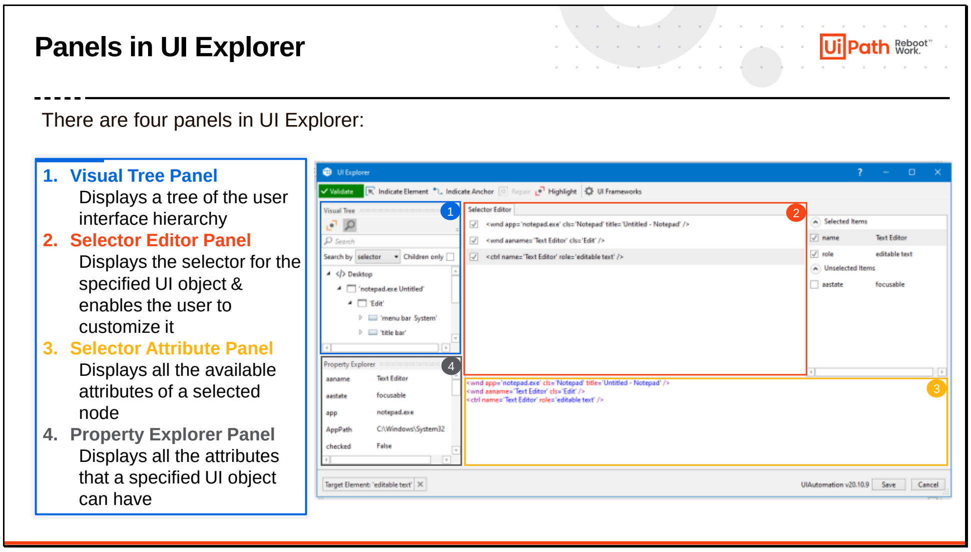
```
<wnd app='notepad.exe' cls='Notepad' title='Untitled - Notepad' />
<wnd aaname='Text Editor' cls='Edit' />
<ctrl name='Text Editor' role='editable text' />
```

aaname	Text Editor
aastate	focusable
app	notepad.exe
AppPath	C:\Windows\System32
checked	False

Target Element: 'editable text'

UI Explorer is an advanced tool for customizing the selector. It enables the user to create a custom selector for a specific UI element. It can be accessed from the Design tab.

Slide 51



Panels in UI Explorer

There are four panels in UI Explorer:

1. **Visual Tree Panel**
   Displays a tree of the user interface hierarchy
2. **Selector Editor Panel**
   Displays the selector for the specified UI object & enables the user to customize it
3. **Selector Attribute Panel**
   Displays all the available attributes of a selected node
4. **Property Explorer Panel**
   Displays all the attributes that a specified UI object can have

```
<wnd app='notepad.exe' cls='Notepad' title='Untitled - Notepad' />
<wnd aaname='Text Editor' cls='Edit' />
<ctrl name='Text Editor' role='editable text' />
```

The interface of UI Explorer consists of four panels and several fields.

The four panels in UI Explorer are:

- **Visual Tree Panel**
  - Displays a tree of the user interface hierarchy and enables the user to navigate through it, by clicking the arrows in front of each node. By default, the first time UI Explorer is opened, this panel displays all opened applications, in alphabetical order.
  - The fields here are:
    - **Highlight**: Highlights the selected element from the Visual Tree in real time.
    - **Show Search Options**: Displays the search box and search filter options. The filter options are:
      - **Search Box**: Enables the users to look for a specific string. If an exact match is not found, nodes containing the nearest match are displayed. Wildcards are supported.
      - **Search by**: Filters the search to a selected attribute or a selector.
      - **Children Only**: Limit the search to the first level children of the selected node. (By default, this check box is not selected.)
- **Selector Editor Panel**
  - Displays the selector for the specified UI object and enables the user to customize it.

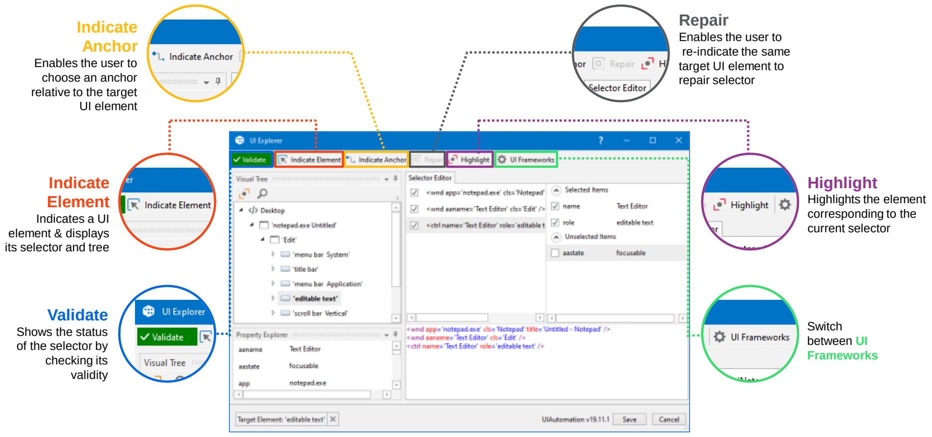
- The bottom part of the panel displays the actual XML fragment that the user has to use in a project.
- The top part of this panel enables the user to view all the nodes in a selector and eliminate the ones that are not necessary by clearing the check box in front of them.
- Selecting a node here displays its attributes in the **Selector Attributes** and **Property Explorer** panels.

- **Selector Attribute Panel**
  - Displays all the available attributes of a selected node
  - The user can add or eliminate some of the node attributes by selecting or clearing the check box in front of each attribute.
  - The user can change the value of each attribute,
- **Property Explorer Panel**
  - Displays all the attributes that a specified UI object can have (including the ones that do not appear in the selector).

To know more, visit: https://docs.uipath.com/studio/v2020.10/docs/uipath-explorer

Slide 52

Fields in UI Explorer



Indicate Anchor
Enables the user to choose an anchor relative to the target UI element

Repair
Enables the user to re-indicate the same target UI element to repair selector

Indicate Element
Indicates a UI element & displays its selector and tree

Highlight
Highlights the element corresponding to the current selector

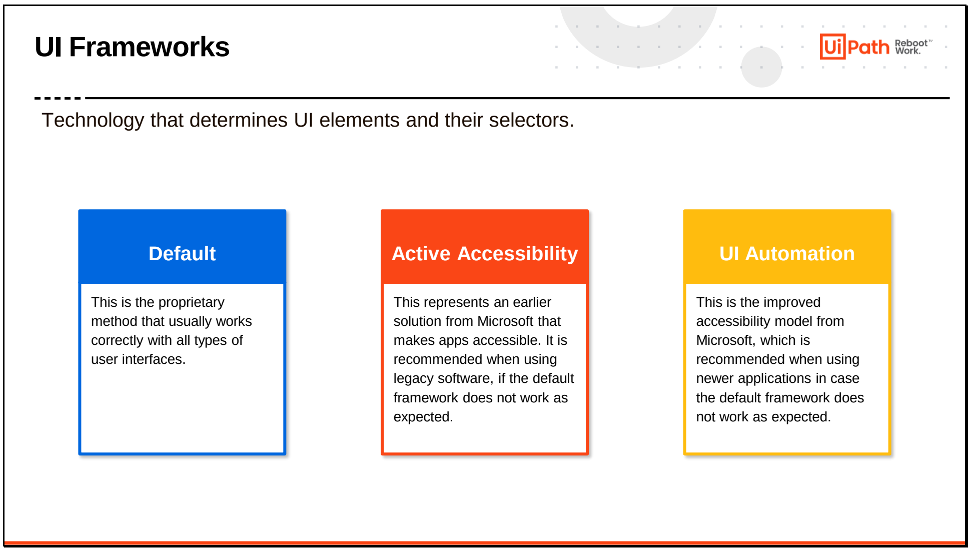
Validate
Shows the status of the selector by checking its validity

Switch between UI Frameworks

The fields in UI Explorer are:

- **Validate**: 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. The Validate button has the following states:
    - O (buffer): Selector is being validated
    - √: Valid selector
    - ×: Invalid selector
    - ?: Modified selector, revalidate
- **Indicate Element:** Indicates a new UI element to replace the previous one.
- **Indicate Anchor:** Enables the user to choose an anchor relative to the target UI element.
- **Repair**: Enables the user to re-indicate the same target UI element and repair the selector. The button is available only when the selector is invalid.
- **Highlight**: Brings the target element in the foreground. The highlight stays on until it's switched off. The button is enabled only if the selector is valid.
- **UI Frameworks**: Changes the technology used to determine UI elements and their selectors.

Slide 53



UI Frameworks

Technology that determines UI elements and their selectors.

Default

This is the proprietary method that usually works correctly with all types of user interfaces.

Active Accessibility

This represents an earlier solution from Microsoft that makes apps accessible. It is recommended when using legacy software, if the default framework does not work as expected.

UI Automation

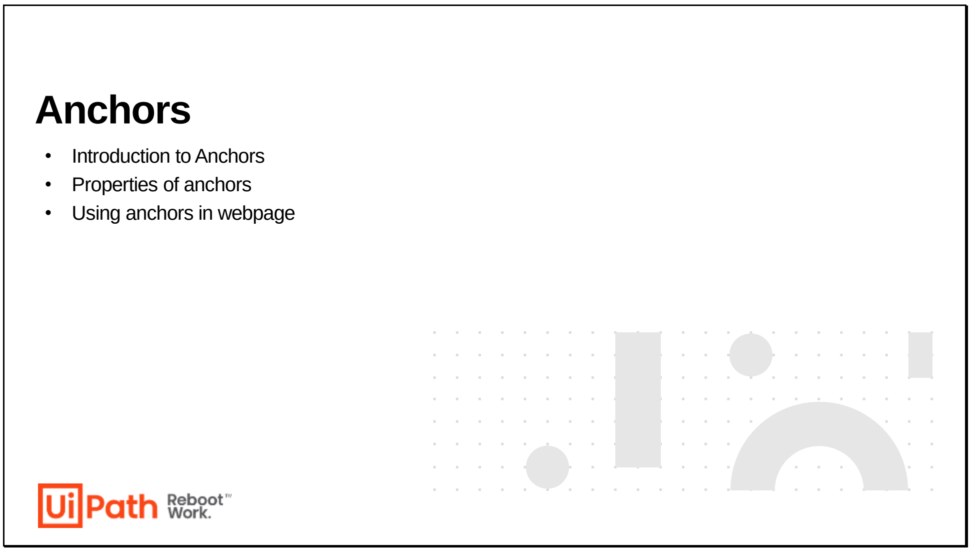
This is the improved accessibility model from Microsoft, which is recommended when using newer applications in case the default framework does not work as expected.

UI frameworks define the technology that determines UI elements and their selectors. The user can switch between three UI Frameworks from the UI Explorer designated button:

- **Default**: This is the proprietary method that usually works correctly with all types of user interfaces.
- **Active Accessibility**: This represents an earlier solution from Microsoft that makes apps accessible. It is recommended when using legacy software, if the default framework does not work as expected.
- **UI Automation**: This is the improved accessibility model from Microsoft, which is recommended when using newer applications in case the default framework does not work as expected.

The selectors generated may be very different from one framework to another.

Slide 54



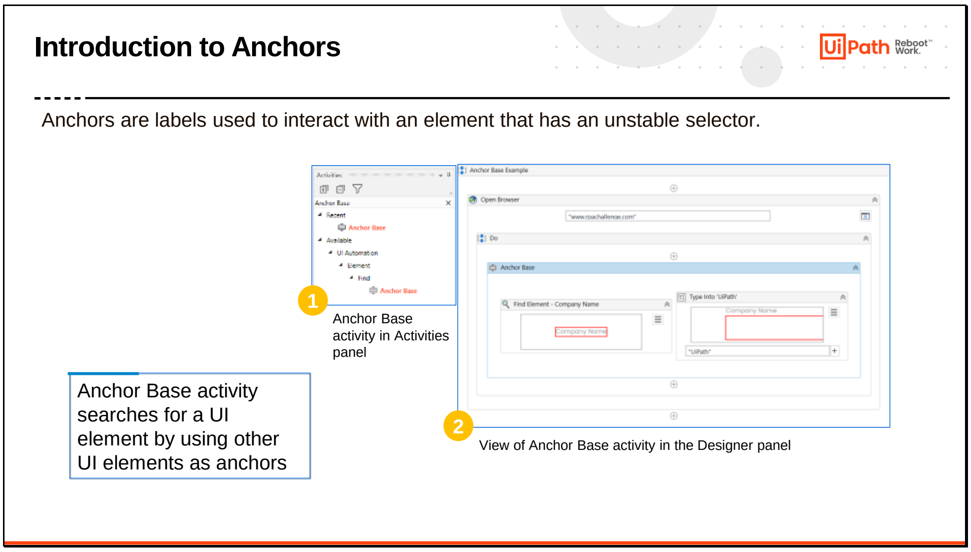
Anchors

- Introduction to Anchors
- Properties of anchors
- Using anchors in webpage

UiPath Reboot™ Work.

This section explains anchors and its properties.

Slide 55



Introduction to Anchors

Anchors are labels used to interact with an element that has an unstable selector.

Anchor Base activity searches for a UI element by using other UI elements as anchors

1 Anchor Base activity in Activities panel

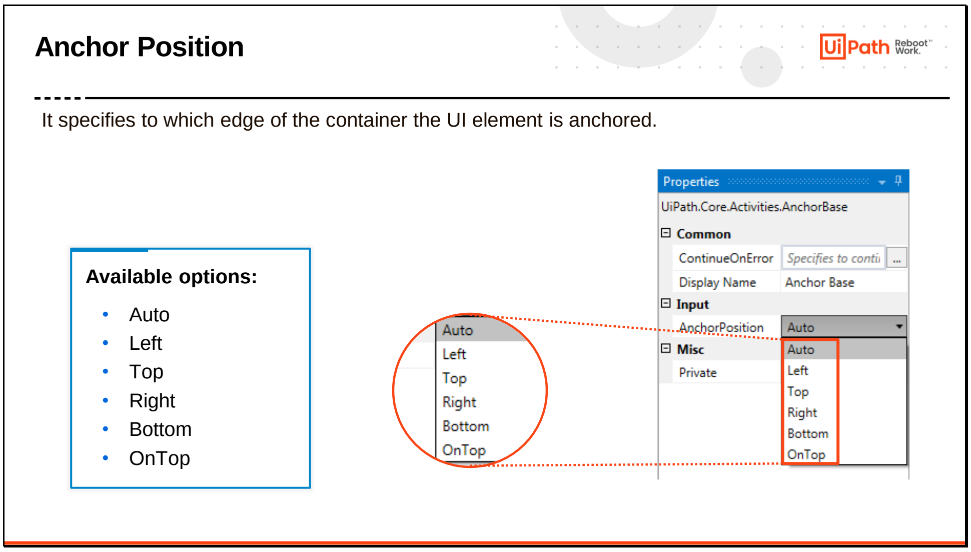
2 View of Anchor Base activity in the Designer panel

Anchors are used when a reliable selector for a UI Element is not available. If a selector keeps changing after every iteration, then the user defines an anchor (or label) that can locate the selector every time it changes. Anchor base activity in UiPath Studio searches for the target UI element by using UI elements (linked to the target UI Element) as anchors. So, the activity locates the anchor and its relative UI element on a screen.

For example, the form fields on the RPA Challenge website changes every time the webpage is refreshed. Therefore, without using anchors to detect the new position of the form fields, it will be difficult for the process to identify the correct selector and it will give selector errors.

To know more, visit: https://docs.uipath.com/activities/docs/anchor-base

Slide 56



Anchor Position

It specifies to which edge of the container the UI element is anchored.

Available options:

- Auto
- Left
- Top
- Right
- Bottom
- OnTop

Properties

UiPath.Core.Activities.AnchorBase

Common

ContinueOnError: Specifies to conti...

Display Name: Anchor Base

Input

AnchorPosition: Auto

Misc

Private

Anchor Position is the property under the Input category of Anchor base and is one of the key properties of anchors. It refers to the edge of the container where the UI element is anchored.

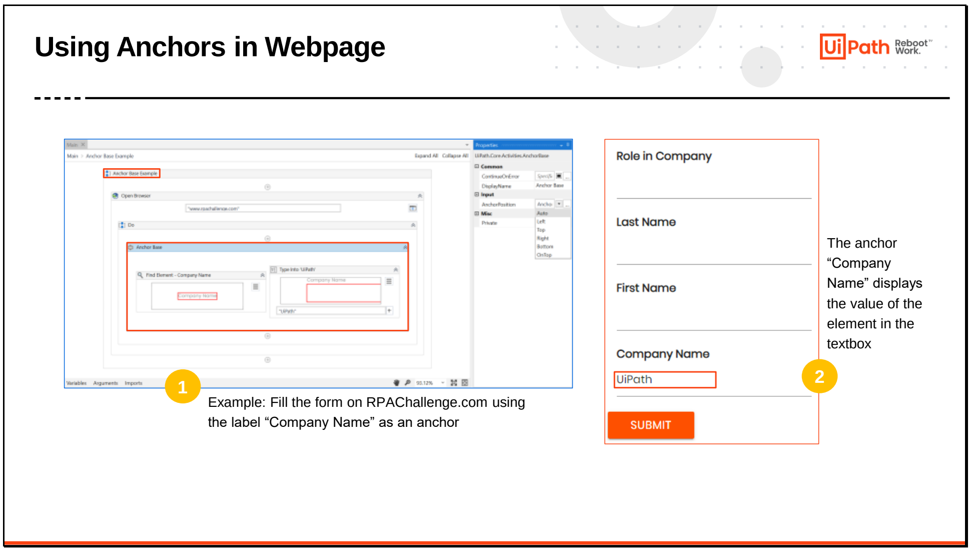
There are six options that can be used by the Anchor Position property:

- **Auto:** UI element closest to the container is searched. When there are UI elements in all directions and at equal distance, then the Anchor base activity will search UI element in the order – Right, Left, Bottom.
- **Left:** UI element on the left of the container is searched.
- **Top:** UI element at the bottom of the container is searched.
- **Right:** UI element on the right of the container is searched.
- **Bottom:** UI element at the top of the container is searched.
- **OnTop:** UI element with a label is searched.

It is good to use anchor position Auto, and the default direction for the anchor position is first left, right and then the bottom. Right, Left and then Bottom directions are generally preferred for an anchor position and then the closest one is selected.

Note: The Top and Bottom options differ from the usual nomenclature. Top searches for UI element in bottom, whereas Bottom searches for UI element in the top.

Slide 57



Using Anchors in Webpage

Example: Fill the form on RPAChallenge.com using the label “Company Name” as an anchor

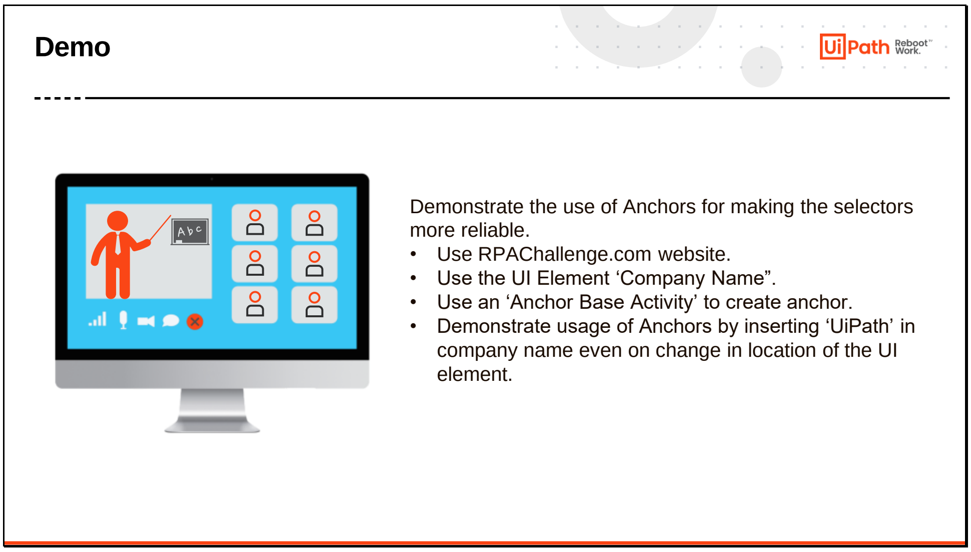
The anchor “Company Name” displays the value of the element in the textbox

Anchor base is a container that searches for a UI element by using other UI elements as anchors and it is used when a reliable selector is not available. The Anchor base activity is used to locate an anchor on a webpage.

Example: Fill a form on **RPAChallenge.com**.

When a user opens the RPAChallenge.com website, a form appears. The workflow writes a particular text string in a field (say Company Name) on this page which changes its position every time the webpage loads. This label **Company Name** can be used as an anchor to type into the value in its respective text box.

Slide 58



Demo

Demonstrate the use of Anchors for making the selectors more reliable.

- Use RPAChallenge.com website.
- Use the UI Element 'Company Name".
- Use an 'Anchor Base Activity' to create anchor.
- Demonstrate usage of Anchors by inserting 'UiPath' in company name even on change in location of the UI element.

Demonstrate the use of Anchors for making the selectors more reliable.

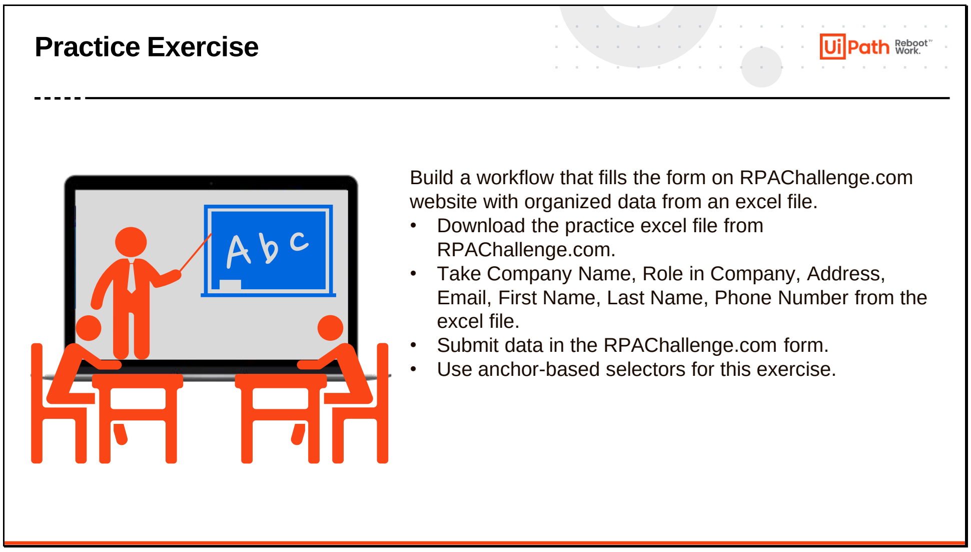
How to use anchors to locate another UI element on the screen:

- In UiPath Studio, drag and drop an **Open Browser** activity into the designer panel.
- Enter website URL "www.rpachallenge.com".
- Run the file…Notice the **location** of the Company Name field in the form. Its location changes every time this webpage is refreshed.
- Insert a **Type Into** activity within the **Do** container of the **Open Browser** activity.
- Click the "Indicate element inside browser" link and indicate the input field of the **Company Name** label on the website.
- Insert the text "UiPath" in the input field of the **Type Into** activity.
- Run the file…We can see that the location of the Company Name field has changed this time.
- Also, the execution will take longer than usual because workflow faces difficulty in finding the Company Name. Wait for few seconds to get an execution error.

To solve this problem, use an Anchor Base activity and use an anchor.

- Delete the Type Into activity and insert an **Anchor Base** activity in the **Do** container.
- Insert a **Find Element** activity in the *Anchor* area of the Anchor Base.
- Click the "Indicate element inside browser" link and indicate the label Company Name as the anchor.
- Insert a **Type Into** activity in the *Action Activity* area. Click the "Indicate element inside browser" link and indicate the input field of Company Name. \
- Insert 'UiPath' in the input field of the Type Into activity.
- Run the project
  - Position of the Company Name field changed again, but the process succeed in inserting the required text in the field. This is the benefit of using anchors.

Slide 59



Practice Exercise

Build a workflow that fills the form on RPAChallenge.com website with organized data from an excel file.

- Download the practice excel file from RPAChallenge.com.
- Take Company Name, Role in Company, Address, Email, First Name, Last Name, Phone Number from the excel file.
- Submit data in the RPAChallenge.com form.
- Use anchor-based selectors for this exercise.

Build a workflow that fills the form on RPAChallenge.com website with organized data from an excel file.

- Download the practice excel file from RPAChallenge.com.
- Take Company Name, Role in Company, Address, Email, First Name, Last Name, Phone Number from the excel file.
- Submit data in the RPAChallenge.com form.
- Use anchor-based selectors for this exercise.

Process Overview

- START
- Use a Read Range activity within an Excel Application Scope activity to read data from an Excel file challenge.xlsx and store it in a data table.
- Use an Open Browser activity to open the URL – “www.rpachallenge.com”.
- Use a Click activity to click on the Start button on the website.
- Use a For Each Row activity to loop through rows in the data table.
- Use Anchor Base activities within the For Each Row activity to locate anchors for each input area of the RPAChallenge.com submission form.
- Use Type Into activities within Anchor Base activities to input data from the data table.
- Use string manipulation methods called Substring and IndexOf to manipulate the names of the users.
- Use a Click activity to submit the form.
- STOP

Slide 60

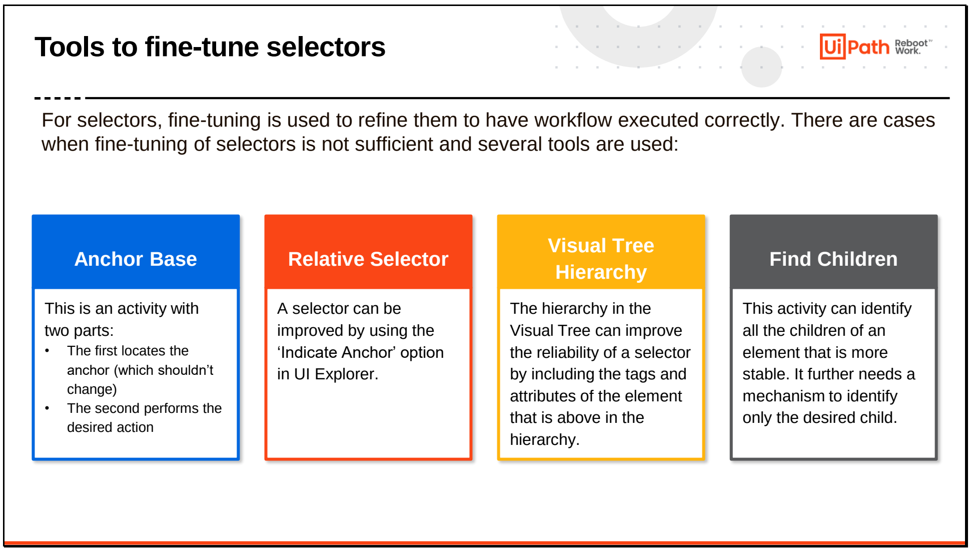
Fine-tuning Selectors

- Tools to fine-tune selectors
- Approach to fine-tune selectors

Reboot™ Work.

This section explains how to fine-tune selectors.

Slide 61



Tools to fine-tune selectors

For selectors, fine-tuning is used to refine them to have workflow executed correctly. There are cases when fine-tuning of selectors is not sufficient and several tools are used:

Anchor Base

This is an activity with two parts:

- The first locates the anchor (which shouldn't change)
- The second performs the desired action

Relative Selector

A selector can be improved by using the 'Indicate Anchor' option in UI Explorer.

Visual Tree Hierarchy

The hierarchy in the Visual Tree can improve the reliability of a selector by including the tags and attributes of the element that is above in the hierarchy.

Find Children

This activity can identify all the children of an element that is more stable. It further needs a mechanism to identify only the desired child.

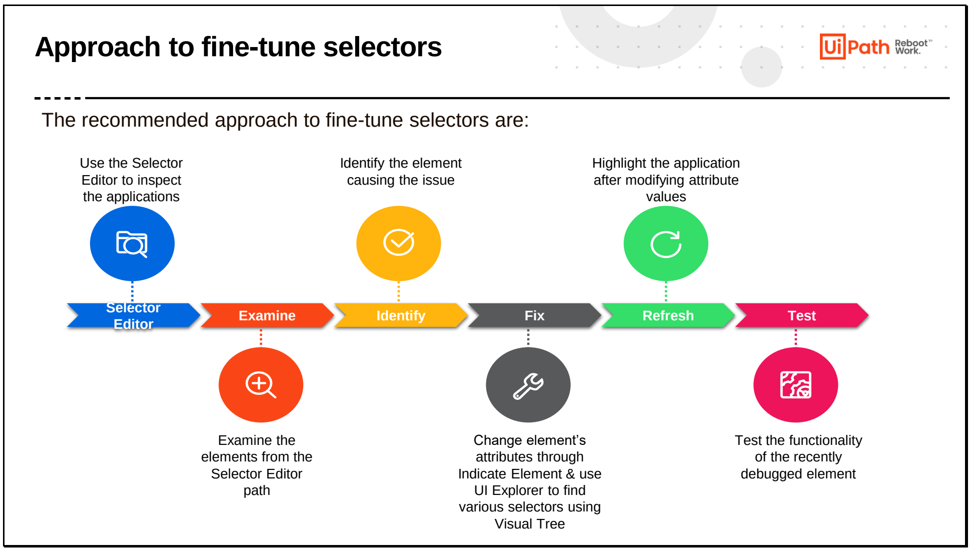
In the case of selectors, fine-tuning is used which is the process of refining selectors in order to have the workflow correctly executed in situations in which the generated selector is unreliable, too specific or too sensitive with regards to system changes. It mainly consists of small simple changes that have a larger impact on the overall process, such as adding wildcards, using the repair function or using variables in selectors.

In most of the cases in which the selectors automatically generated are not reliable enough, fine-tuning will solve the issue. However, there are some difficult situations for which there are other approaches:

- **Anchor Base:** Useful in cases in which the attribute values are not reliable (are generated at each execution, for example), but there is a UI element that is stable and is linked to the target UI element. The Anchor Base activity has two parts, one to locate the anchor UI element (like 'Find Element'), and the second to perform the desired activity.
- **Relative Selector**: It incorporates the information about the anchor's selector in the selector of the target UI element. However, the new selector will probably need additional editing, as the part that made the first selector will still be in the new selector. So that part (like a dynamic ID) needs to be removed, and the selector will stabilize using the anchor's selector.
- **Visual Tree Hierarchy**: Useful when the target UI element's selector is not reliable, but the selector of the UI element right above in the hierarchy is. However, again, the selector needs further editing and validation, as the dynamic part needs to be removed and, at the same time, the user needs to make sure that the target element can be identified with a unique attribute.
- **Find Children**: It identifies the children of a certain UI element. The output of the activity is the collection of children. So, in order to make it work, the user needs to come up with a mechanism to identify only the target UI element (using one of its

attributes, that makes is unique between the children, but wouldn’t be enough to identify it universally).

Slide 62



Approach to fine-tune selectors

The recommended approach to fine-tune selectors are:

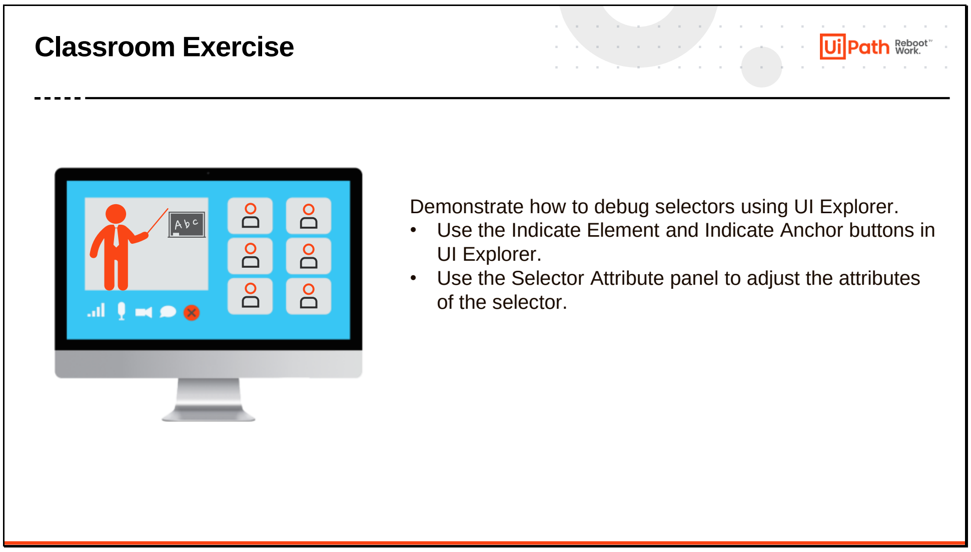
Use the Selector Editor to inspect the applications
Identify the element causing the issue
Highlight the application after modifying attribute values
Selector Editor
Examine
Identify
Fix
Refresh
Test
Examine the elements from the Selector Editor path
Change element's attributes through Indicate Element & use UI Explorer to find various selectors using Visual Tree
Test the functionality of the recently debugged element

Selector Editor can be used for checking selectors and UI Explorer for debugging selectors. UI Explorer enables the user 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 starts. Fine-tuning a selector involves changing the element's attributes, either adding or removing them and using wildcards where specific attributes have variable values inside them.

The recommended approach that can be adapted to fine-tune selectors can have the following steps:

- **Selector Editor:** Inspect the applications
- **Examine:** Examine the elements from the Selector Editor path
- **Identify:** Identify the element causing the issue. Once the issue is identified and resolved, click Validate to ensure that the Validate button turns green.
- **Fix:** Change the element's attributes through Indicate Element or Repair and use UI Explorer to find various selectors using Visual Tree
- **Refresh:** Highlight the application after modifying attribute values
- **Test:** Test the functionality of the recently debugged element

Slide 63



Classroom Exercise

Demonstrate how to debug selectors using UI Explorer.

- Use the Indicate Element and Indicate Anchor buttons in UI Explorer.
- Use the Selector Attribute panel to adjust the attributes of the selector.

Demonstrate how to debug selectors using the UI Explorer.

How to debug a process using UI Explorer?

- In UiPath Studio, insert an Open Browser activity in the designer panel.
- Enter www.rpachallenge.com in the URL field and run the project to open the website. A form with multiple fields can be seen on the webpage. The fields on this form change their position every time the webpage is refreshed.
- Drag and drop a Type Into activity in the Do section of the Open Browser activity.
- Click the “Indicate element inside browser” link and indicate the input field of First Name.
- In the text box of the Type Into activity, enter a name, say John.
- Run the project.

Outcome: UiPath Studio gives an error. It was unable to find the input field of First Name because its position has changed.

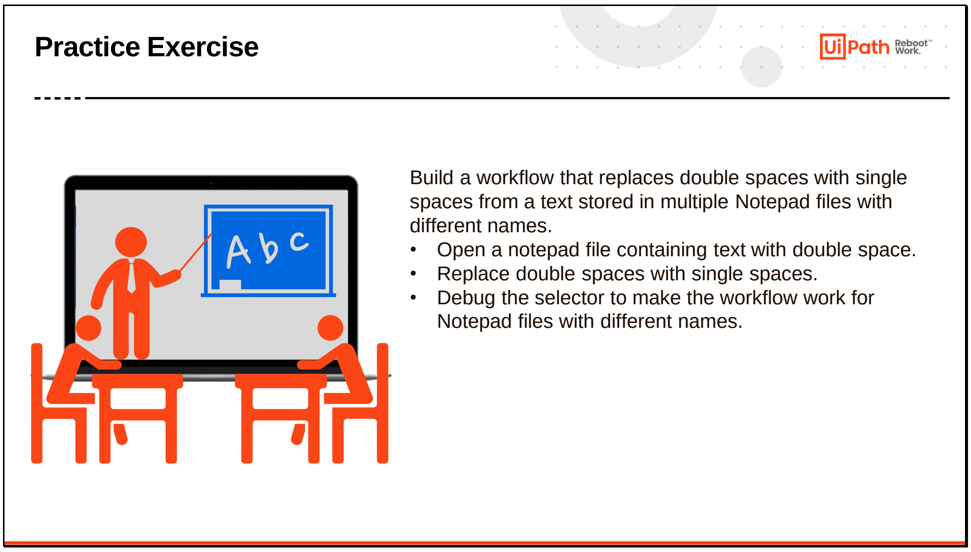
Follow below steps to debug:

- Click the hamburger button within the **Type Into** activity and select **Open in UiExplorer** from the context menu.

- In the UI Explorer window, notice the red **Validate** button. It indicates the error.
- Click on the Indicate Element button and select the input field of First Name.
- Now, click on the Indicate Anchor button and select the First Name.
- In the Selector Attributes panel, uncheck the box of the “id” attribute.
- Click Validate to validate the changes. It shows a green Validate button.
- Click Save.
- Run the project.

Outcome: Project runs without an error.

Slide 64



Practice Exercise

Build a workflow that replaces double spaces with single spaces from a text stored in multiple Notepad files with different names.

- Open a notepad file containing text with double space.
- Replace double spaces with single spaces.
- Debug the selector to make the workflow work for Notepad files with different names.

Build a workflow that replaces double spaces with single spaces from a text stored in multiple Notepad files with different names.

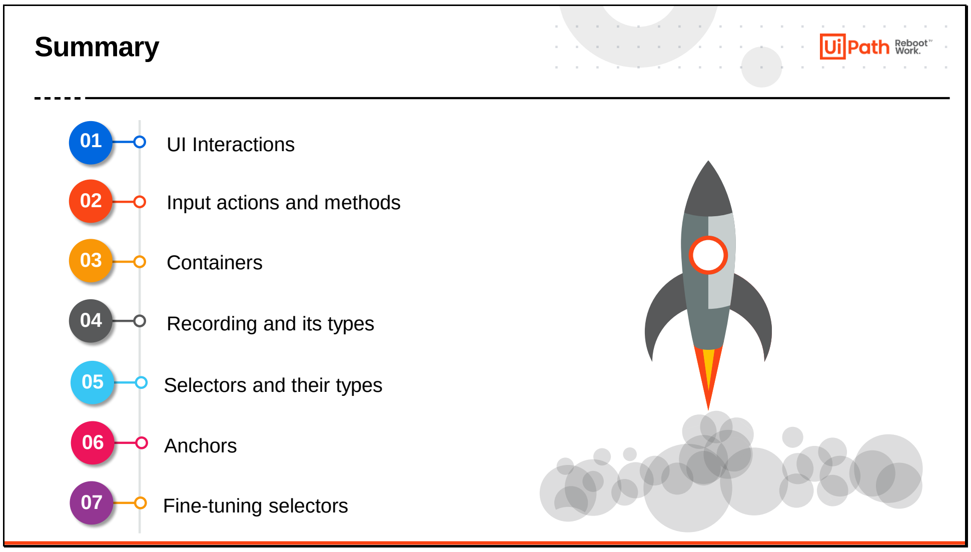
Notes:

- Open a notepad file containing text with double space.
- Replace double spaces with single spaces.
- Debug the selector to make the workflow work for Notepad files with different names.

Process Overview

- START
- Use an Attach window activity and indicate one Notepad file (File_123421.txt)
- Use the wildcard (*) in the Notepad file name using the Selector Editor to make the workflow work for a Notepad with a different name.
- Use Click activities and indicate the Replace option within the Edit option from the Notepad file's toolbar.
- Use Type Into activities to enter double space in the *Find what* text box and single space in the *Replace with* text box of the Replace popup window.
- Use Click activities to click the Replace All button and to close the Replace pop up window.
- Run the program to replace the double spaces of "File_123421.txt" file.
- STOP

Slide 65



Summary

01 UI Interactions

02 Input actions and methods

03 Containers

04 Recording and its types

05 Selectors and their types

06 Anchors

07 Fine-tuning selectors

To summarize, this lesson explained:

- UI interactions
- Input Actions and Input Methods
- Containers
- Recording and its types
- Selectors and their types
- Anchors
- Fine-tuning selectors