**What is a selector?**

UiPath Studio offers a functionality called selectors which is used to interact with the Graphical User Interface (GUI) elements by identifying their tags and attributes. Some examples of tag-attribute pairs are: –

* wnd-app
* wnd-cls
* html-url
* html-title
* webctrl-idx
* webctrl-aaname
* sap-id

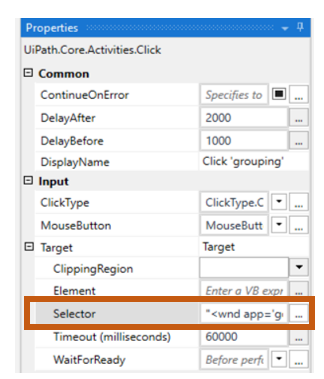
The selector to your activity can be found in the Target section of the Properties pane, as shown below: –

A selector is composed of several nodes and each node has one or more attributes associated with it. Modifications to the attributes can be done as required.

For example, if the title of the window is not constant, the value of the attribute can be changed to make it dynamic. This is done to correctly identify the target elements UiPath Studio wants to access.

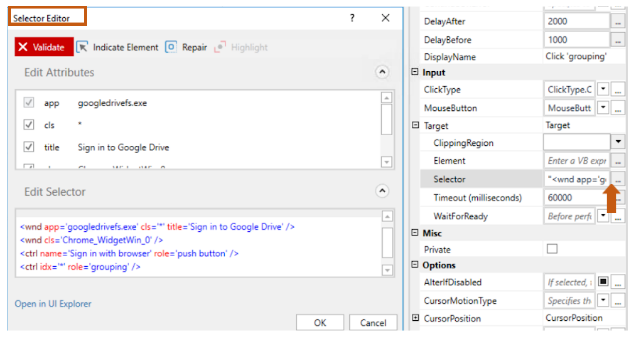
**Selector Editor**

A detailed description of an entire selector can be viewed and modified by opening the selector editor: –



There are several options in the selector editor such as: –

* Validate: This button has the following states: –



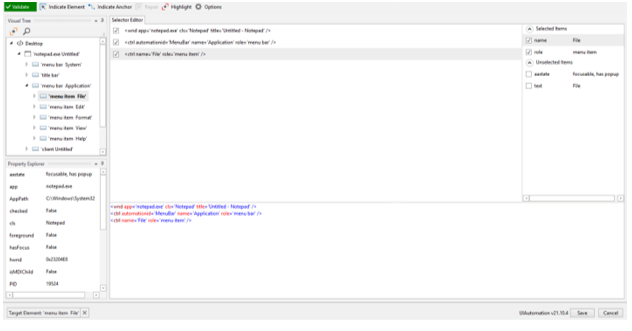
* Indicate element: Used to interact with a new element on screen.
* Repair: Used to repair an already indicated element.
* Highlight: Place a red rectangle around the indicated element to confirm.
* Edit Attributes: Specifies all application related components.
* Edit Selector: Contains the current selector information (tags and attributes).
* Open in UI Explorer: Opens UI Explorer for detailed view of the selector.

**UI Explorer:**

UI Explorer is a tool offered by Studio that enables you to customize your selector for a specific UI element. You can add/remove attributes, tags or make your selector dynamic.

An anchor can also be indicated in case the element’s position is relative to it and not constant.

The UI Explorer window looks like this: –



**Selectors with Wildcards**

Wildcards help to replace single/multiple characters, including spaces in a string inside a selector. They are useful when we have varying attributes in a selector.

For example, the title of a window can be different at every run, so instead of placing the entire title in the attribute, we can add a wildcard to replace some or all portions of the title string.

Two types of wildcards can be used: –

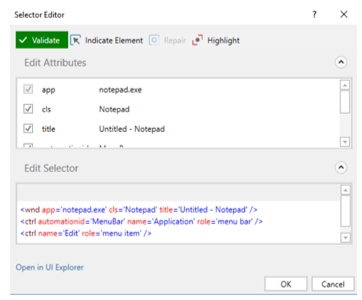
1. Asterisk (\*): used to replace zero or more than zero characters.
2. Question mark (?): used to replace a single character.

In the above UI Explorer window, we can replace the title attribute from “Untitled – Notepad” to “\* – Notepad”.

* **Type of Selectors**

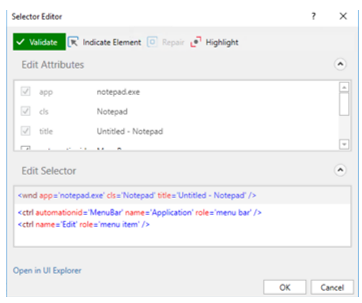
1. Full Selectors: –

Contain detailed information about the element, including the description of the outer container or window. These are useful when multiple windows/screens must be identified. The top-level window’s selector is editable.



1. Partial Selectors: –

Do not contain description of the top-level window. Activities with partial selectors are enclosed in a container such as Attach window, that contains a full selector of the outer window. These are useful when actions need to be performed on a single window. The top-level window’s selector is fixed and not editable.



1. Dynamic Selectors: -

This kind of selector utilizes a variable or an argument in place of a fixed tag value. These are useful when finding an element whose tag value can change during run time. You can directly choose the variable from selector editor. As shown below, “Option” is a variable which can hold values other than “Edit”.

<ctrl name=’Edit’ role=’menu item’ />

<ctrl name=’{{Option}}’ role=’menu item’ />

1. Fuzzy Search: –

A fuzzy search in a selector allows to identify text based on a pattern rather than an exact match. It is useful when dealing with change prone activities in your automation process. For example, use fuzzy search when identifying a web page title:

<html app=’firefox.exe’ title=’Sign in to Google Drive’

matching:title=’fuzzy’ fuzzylevel:title=’0.9’ />

Here, fuzzy level value closer to zero would return lower similarity and fuzzy level value closer to one would return higher similarity.

1. RegEx Search: –

A RegEx search in a selector enables you to identify multiple combinations of the target element in a single search execution. The element is identified based on a regular expression. For example, the digit 5 in the below tag can be replaced with a regular expression to identify every digit combination on the target: –

name=’ID is 5’ role=’text’

name=’ID is \d’ role=’text’ matching:name=’regex’

1. Non-Greedy Search: –

This search in a selector makes it possible to interact with the target element from multiple windows of a single application with matching attributes. For example, if you want to perform simultaneous actions on two similar application windows and want only one selector to work for both, you can modify the value of ‘idx’ attribute to ‘\*’ so that the windows are chosen independent of their place: –

title=’Calculator’ idx=’1’

title=’Calculator’ idx=’\*’

1. Case sensitive Selectors: –

Selectors having case added property identifies an exact match to the target, taking into consideration the casing of an attribute’s value. You can add this property as: –

casesensitive:<Name\_of\_attribute>=’true’