

HTML Object Spy Nested Flow

Creation Date: 2/9/2022 2:21:00 PM

Table Of Contents

Introduction	3
Tool Layout.....	4
Top Bar Menu Functions:.....	4
Bottom Bar Menu Functions:.....	6
Bottom Pane:	8
Example Of a Framed Page with IE	9
Steps:.....	9
Example Of a Framed Page with Other Browsers.....	14
Return Single Object	20
Object Spy – Object Identification Strategy.....	23
Create Code Out of Object Spy Actions	24

Introduction

This document describes the HTML object spy which is integral part of the Nested Flow tool and it is used to perform below tasks

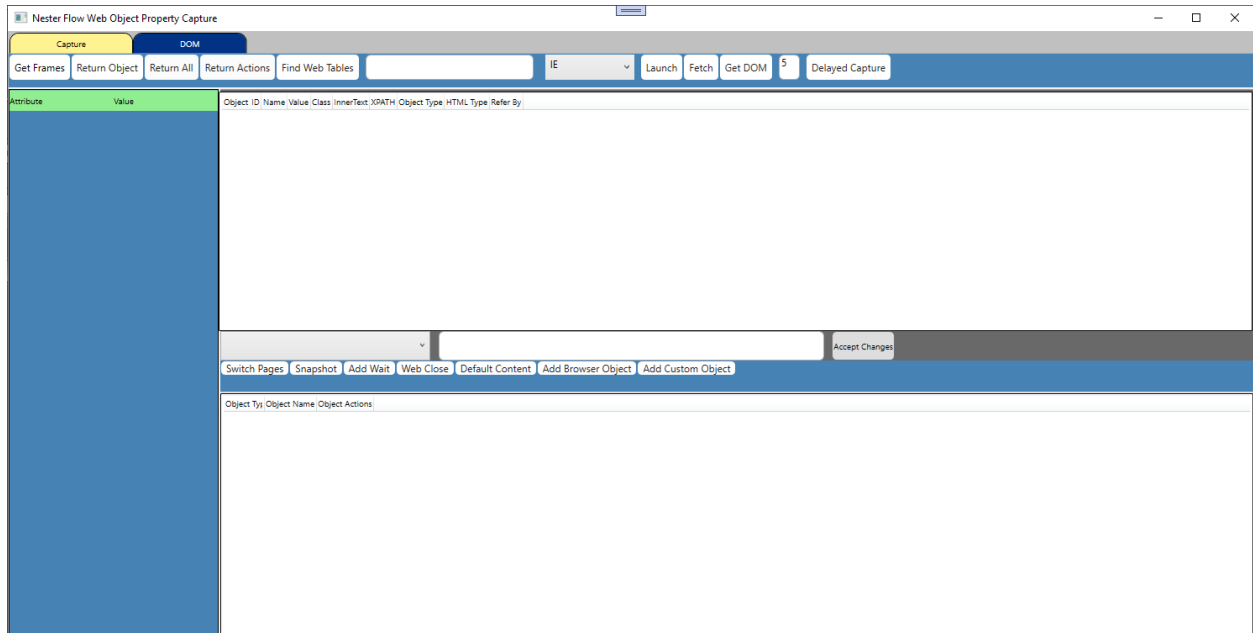
- Capture a single Item
- Capture list of items
- Capture list of items and associated actions
- Capture Object actions as C# script

The tool uses multiple strategies to capture objects:

- Capture Internet explorer objects with mouse right click using .NET framework provided native methods
- Capture any other browser objects with mouse double right click using Selenium (This expects the browser to be opened from the spy itself)

In the upcoming sections we will go over each of these topics in detail

Tool Layout



The Tool has 2 separate menu bars.

The top bar is for helping with object search whereas the bottom bar is to simulate object actions (The tool doesn't support traditional recording instead provides a big list of Selenium webdriver functions to simulate object actions)

Top Bar Menu Functions:


Get Frames → Get All frames in the webpage opened using **Launch** button

Return Object → Returns one single object which is currently displayed in the property grid

Return All → Returns all objects on the grid to Test creation Panel

Return Actions → Returns all object definitions and actions to test case draw panel object list or creates action scripts

Attribute	Value	Object	ID	Name	Value
Browser Title	Wikipedia	HTMLElement::search	searchInput	search	
Object ClassName					
Object ID	searchInput				
Object Inner HTML					
Object Inner Text					
Object Outer HTML	<input name="search" i				
Object Outer Text					
Object Title					
HTML Type	search				
Object TagName	INPUT				
Object isTextEdit Prope	True				
Object Record Number					
Language					
Object onclick					
Object ondblclick					
Object OnbeforeUpdate					
Object OnAfterUpdate					
Object Filters					
Object Style	border-image: none; pa				
Object SourceIndex	99				
Object OffsetLeft	0				
Object OffsetTop	0				
Object OffsetWidth	394				
Object OffsetHeight	44				
Object Value					
Object X	719				
Object Y	465				
Object Name	search				
Object Visibility					
Object Xpath	//html[1]/body[1]/div[3]				
Object Relative XPath	//input[@id='searchInp				
Object CSS Selector	#searchInput				
Object Type	mshtml.HTMLInputElem				
Object href					
Display Name	HTMLElement::search				
Refer By	RefXPath				
Frame Name					
Frame ID					

Step	Type	Step Name
	HTMLElement::search	
	HTMLButton::Search	
	HTMLElement::English	
	HTMLElement:: Wikipedia	
	HTMLElement::The Free Encyclop	
	HTMLButton::js-lang-list-butto	
	HTMLElement::Wikibooks	

Find Web Tables → Get All web Tables in the webpage opened using **Launch** button

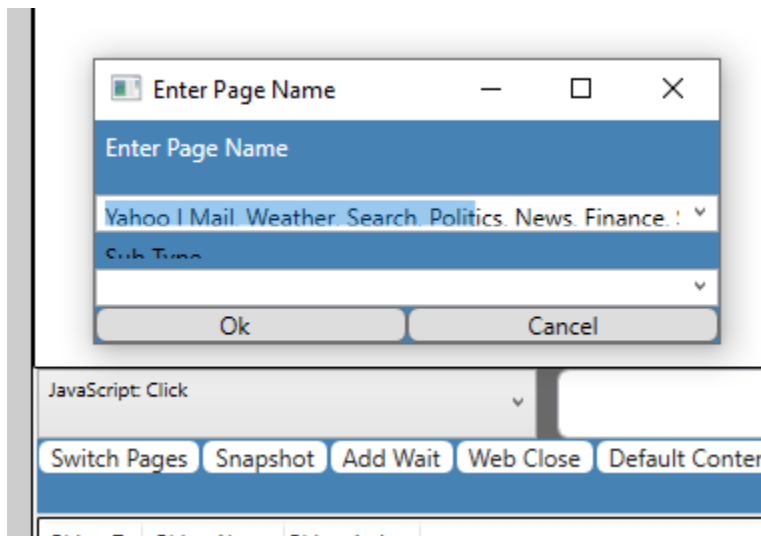


→ This is for launching the browser using Selenium from inside the tool. The initial text box is for entering the URL. Second LOV is to choose type of browser. Supported types are:

- IE → Internet Explorer
- CHROME → Google Chrome
- FF → FireFox
- EDGE → Microsoft chromium-based Edge
- EDGE IE MODE → Microsoft Edge in IE Mode

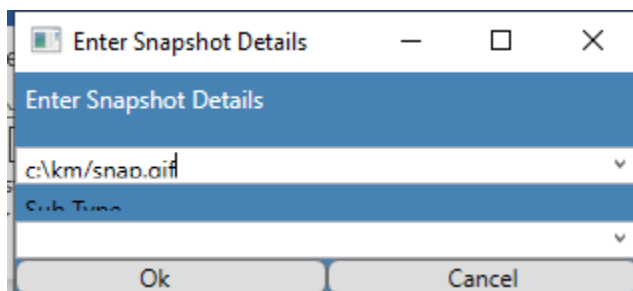
Bottom Bar Menu Functions:

Switch Pages → Moves control to Web Page



Just enter Page title in the text box and click on **Ok** button

Snapshot → Creates snapshot of the web page

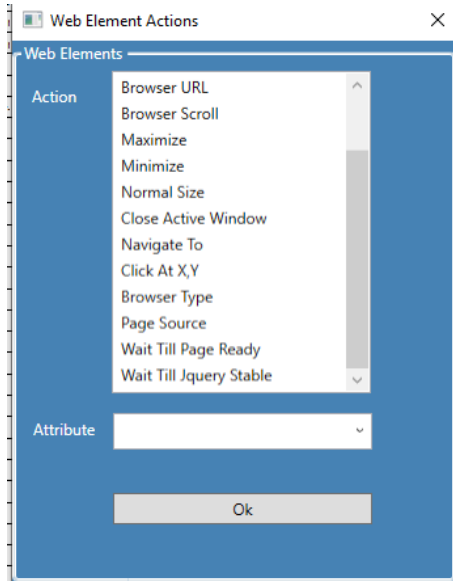


Add Wait → creates a system Delay

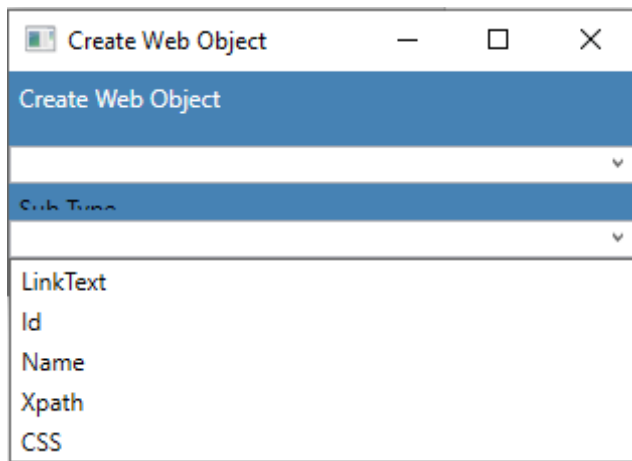
Web Close → Creates a closure of web page

Default Content → Moves to default content (first page or default frame)

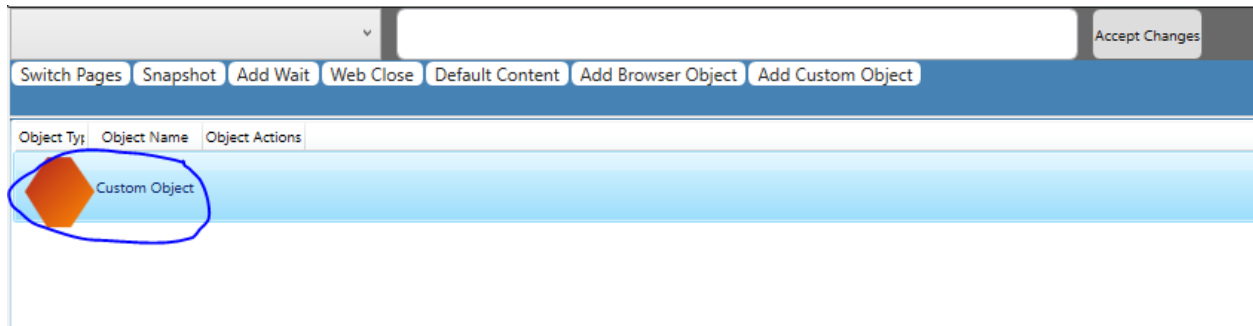
Add Browser Object → Creates a browser action



Add Custom Object → Creates dynamic web object. Just provide identified by and locator value. This function is useful when there is an object created in the run time and for some reason cannot be used with object spy



Bottom Pane:



This section records the actions simulated. These can be returned to the test case builder panel or to test script creator

Example Of a Framed Page with IE

In this example we will examine the ways of recording automation of a framed web page using Internet Explorer.

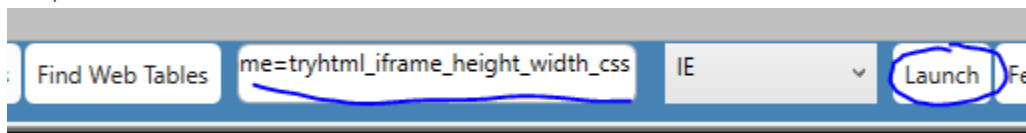
Note: Most modern applications (web pages) have discontinued support for IE as after the advent of HTML5 the entire browser landscape is captured by Google Chrome, MS Edge, Firefox etc. as most features are not supported by IE. IE is still used in legacy applications where a web page opens a form-based windows or java app.

The web page we are considering here is:

https://www.w3schools.com/html/tryit.asp?filename=tryhtml_iframe_height_width_css

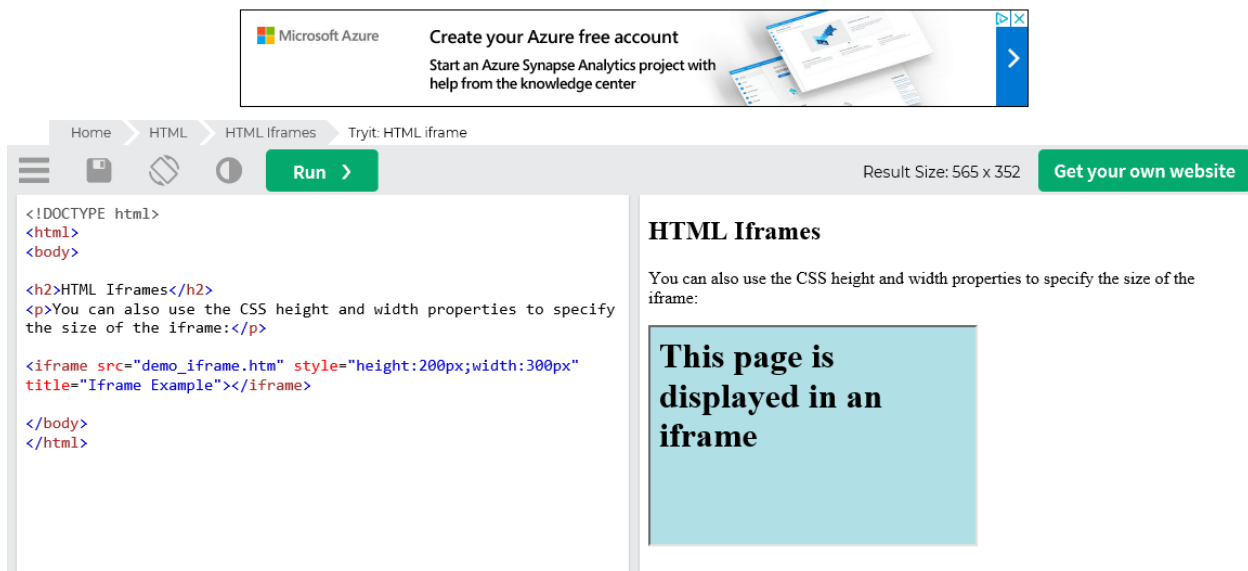
IE pages are automatically captured by the object spy whether they are opened by selenium or not but for actions recording we still need it to be invoked through selenium.

Steps:

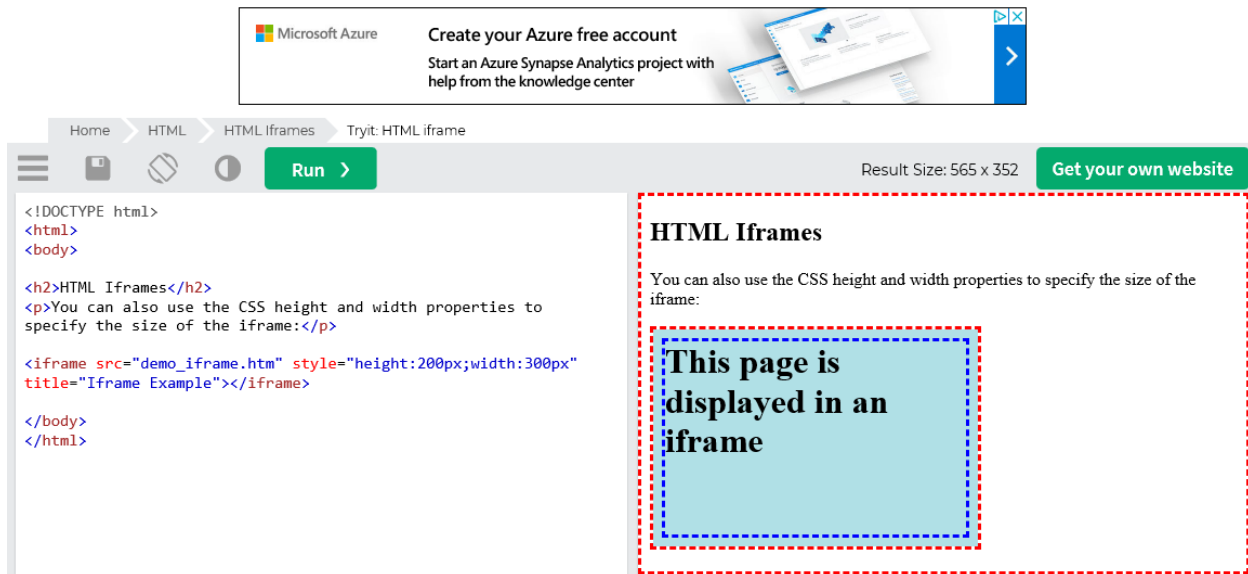


Enter URL in the text box choose IE as the browser and click on **Launch**

Site opens successfully

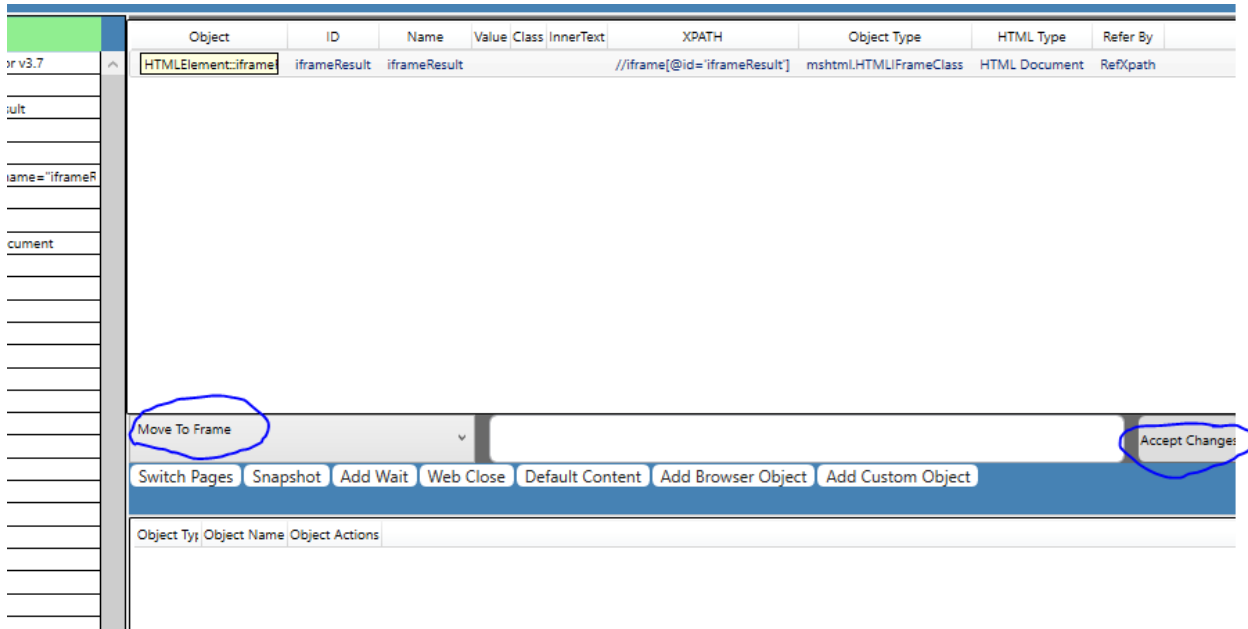


Tool will automatically identify the objects as mouse moves over any IE browser opens.



All iframes are identified as red border and all other items are shown as blue.



For creating a script to access the text in blue border, we need to move to 2 iframes one after another








Choose **Move To Frame** as action and click on **Accept Changes**.

Move To Frame

Switch Pages Snapshot Add Wait Web Close Default Content A

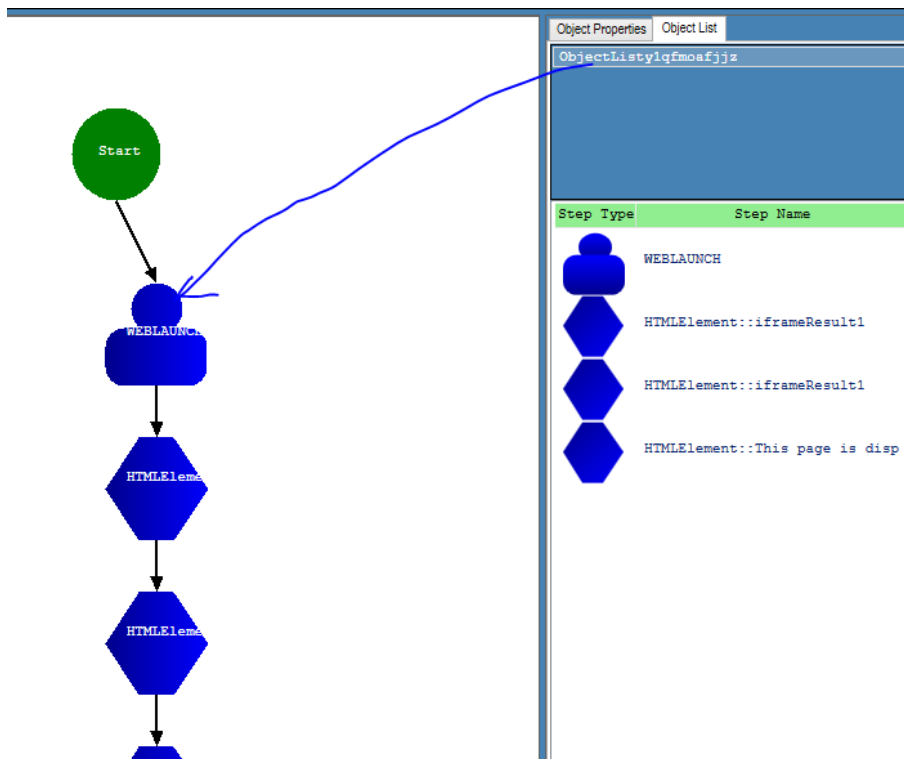
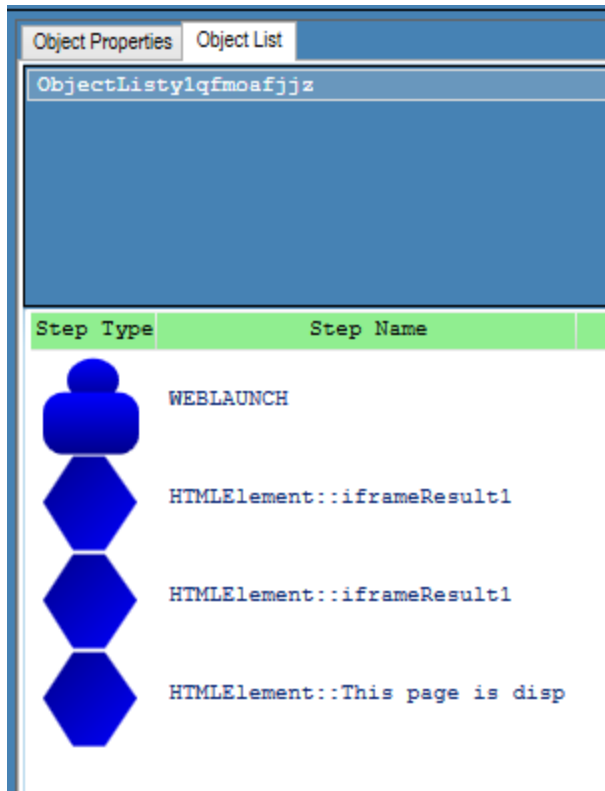
Object Type	Object Name	Object Actions
	WEBLAUNCH IE	
	HTMLElemen	<NewDataSet <Table1> <Action>Mo



Object Type	Object Name	Object Actions
	WEBLAUNCH IE	
	HTMLElemen	<NewDataSet> <Table1> <Action>Move To Frame</Action>
	HTMLElemen	<NewDataSet> <Table1> <Action>Move To Frame</Action>
	HTMLElemen	<NewDataSet> <Table1> <Action>Highlight</Action> <Value

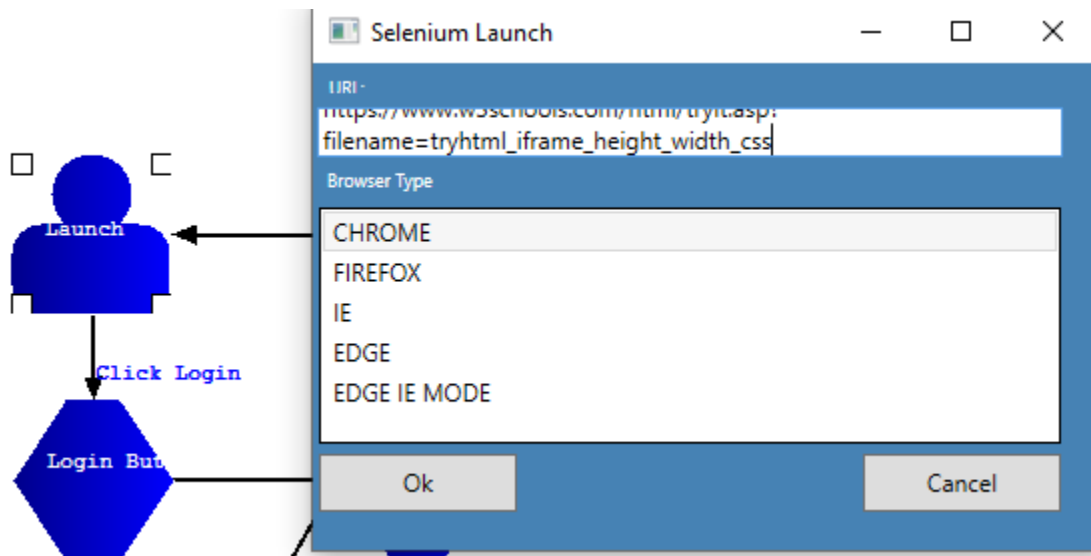
Click on **Return Actions**

Actions returned to test case builder panel



Drag actions to panel.

In the Web Launch component right click and choose **Selenium Launch Parameters** option

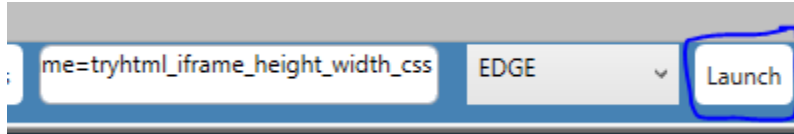


This way whichever browser you record scenario with, you can execute it with any other supported browser

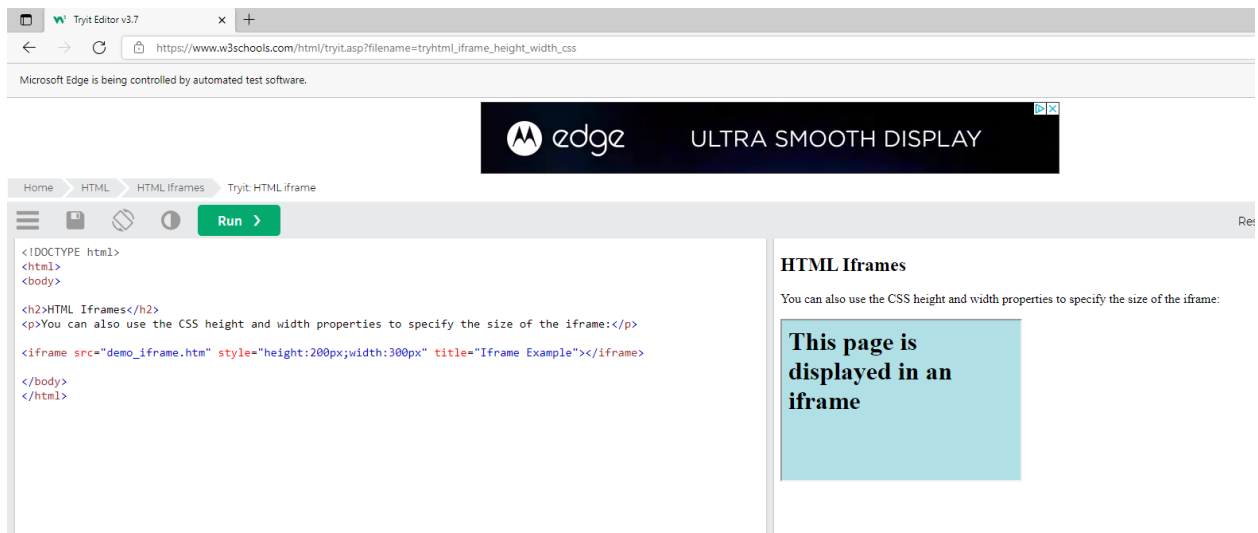
Example Of a Framed Page with Other Browsers

The object spy works with other browsers (Chrome, Firefox, Edge) only if they are triggered from inside the tool. Let us examine the same example with Microsoft Edge (other browsers work the same way)

As shown below, enter the URL in the textbox, choose browser as EDGE and click on **Launch** button.



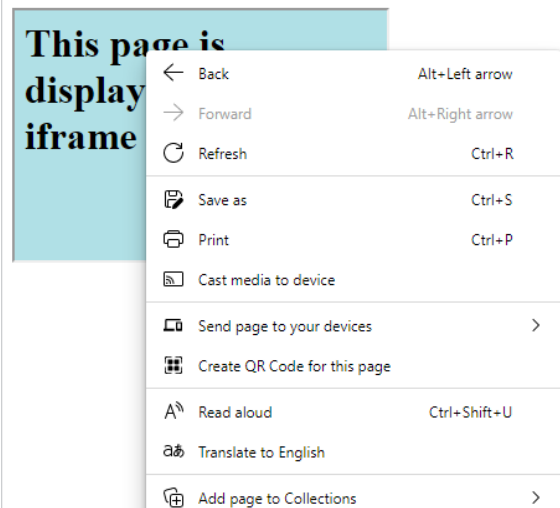
Page opens on edge



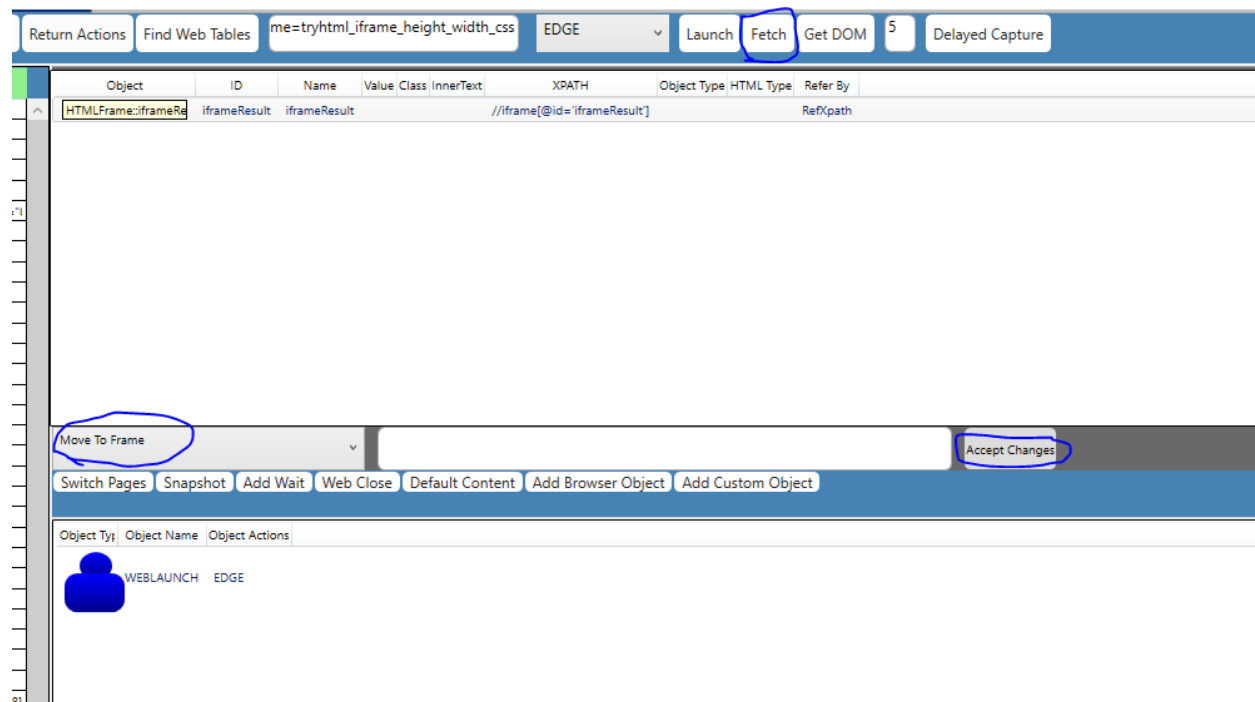
Right click on the text we want to capture on webpage. Instead of capturing it shows default right click menu as it is inside a frame and not visible yet to the object spy

HTML Iframes

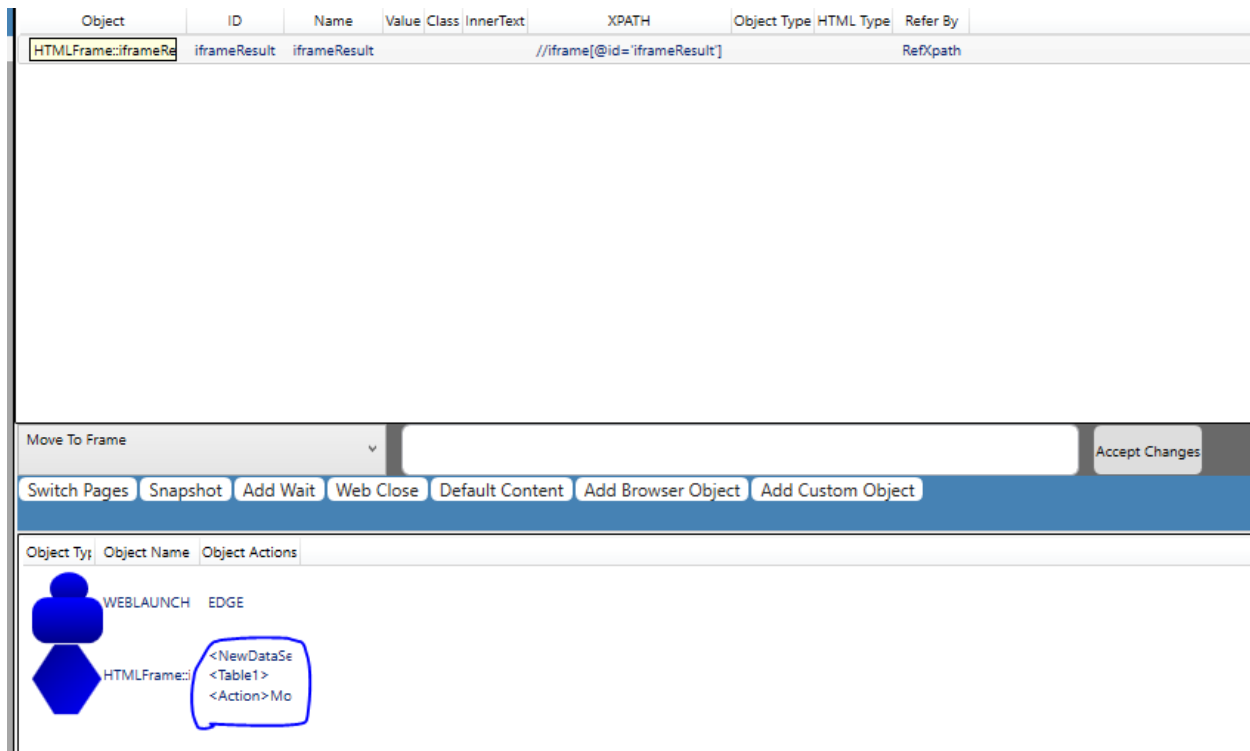
You can also use the CSS height and width properties to specify the size of the iframe:



Click on the text and click on Fetch button in the spy



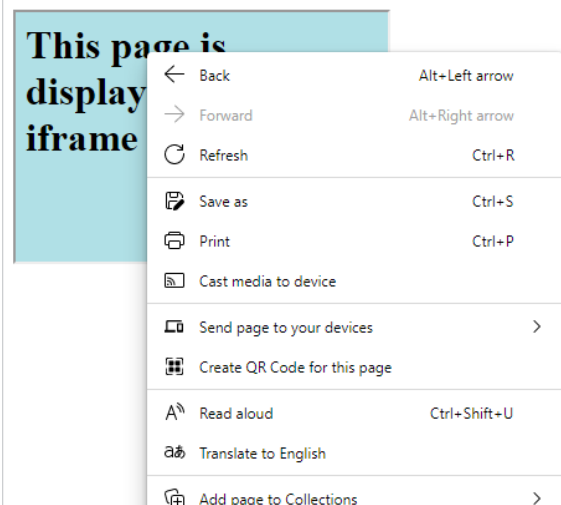
It will capture the frame (parent frame) in which the object is. By default, action is chosen as **Move To Frame** and click on **Accept Changes** button. Automatically focus will be moved to that frame



Once again right click on the text we want to capture on webpage. Instead of capturing it shows default right click menu of Edge again which means it is inside another frame.

HTML Iframes

You can also use the CSS height and width properties to specify the size of the iframe:



Click on **Get Frames** button. All frames on the current frame are displayed

Get Frames Return Object Return All Return Actions Find Web Tables me=tryhtml_iframe_height_width_css EDGE Launch Fetch Get DOM 5 Delayed Capture

Attribute	Value
Browser Title	From Browser Inspect
Object ClassName	
Object ID	
Object Inner HTML	
Object Inner Text	
Object Outer HTML	<iframe src="demo_ifra
Object Outer Text	
Object Title	Iframe Example
HTML Type	
Object TagName	iframe
Object isTextEditable	True
Object Record Number	
Language	
Object onclick	
Object ondblclick	
Object OnBeforeUpdate	
Object OnAfterUpdate	
Object Filters	
Object Style	height: 200px; width: 30
Object SourceIndex	
Object OffsetLeft	8
Object OffsetTop	101
Object OffsetWidth	304
Object OffsetHeight	204
Object Value	
Object X	8
Object Y	101
Object Name	
Object Visibility	True

Object	ID	Name	Value	Class	InnerText	XPATH	Object Type	HTML Type	Refer By
HTMLFrame:iframeRe	iframeResult	iframeResult				//iframe[@id="iframeResult"]			RefXpath
HTMLFrame						//html[1]/body[1]/iframe[1]			RefXpath

Move To Frame

Switch Pages Snapshot Add Wait Web Close Default Content Add Browser Object Add Custom Object

Object Type Object Name Object Actions

WEBLAUNCH EDGE

HTMLFrame: <NewDataSe
<Table1>
<Action>Mo

Try highlighting the frame. You can optionally give a color to highlight it with.

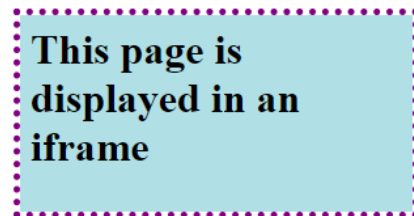
HTMLFrame //html[1]/body[1]/iframe[1] RefXpath

Highlight purple Accept Changes

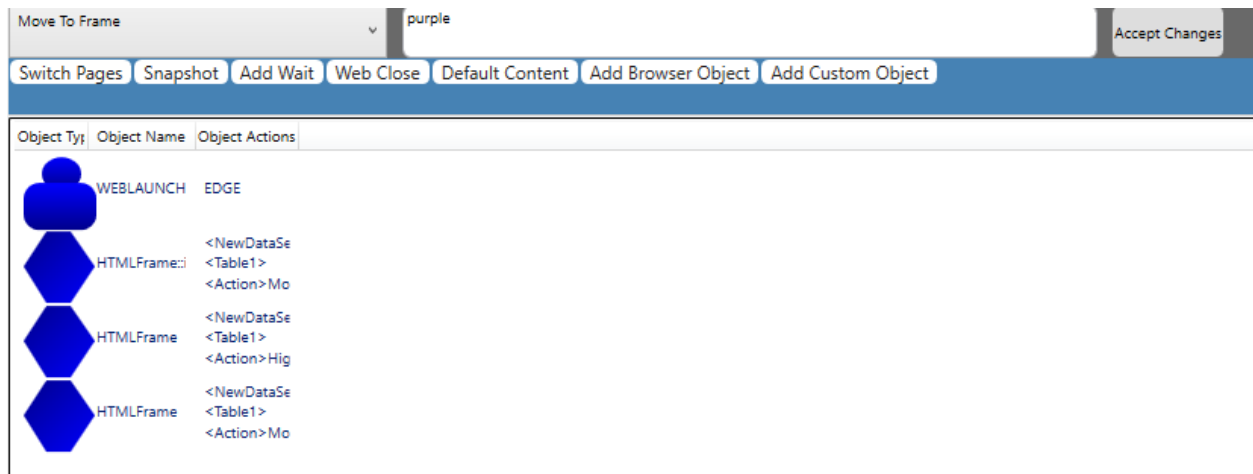
Switch Pages Snapshot Add Wait Web Close Default Content Add Browser Object Add Custom Object

HTML Iframes

You can also use the CSS height and width properties to specify the size of the iframe:

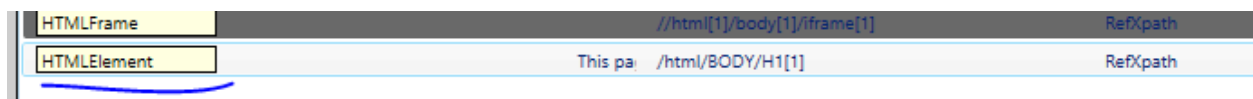


It is evident that the text is inside this frame. Hence, perform a **Move To Frame** action.

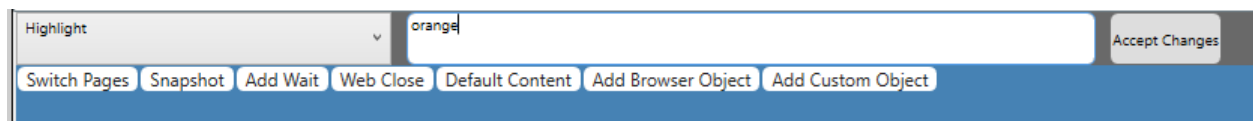


This time, right click on the text will not display default context menu of Edge which means Object spy is seeing it now. Double right click on the object.

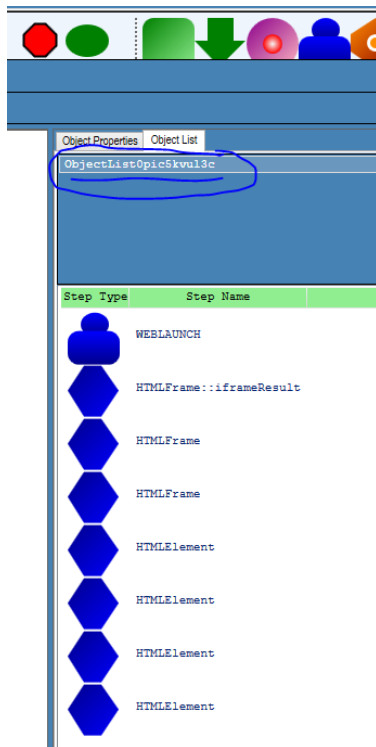
Object is captured.



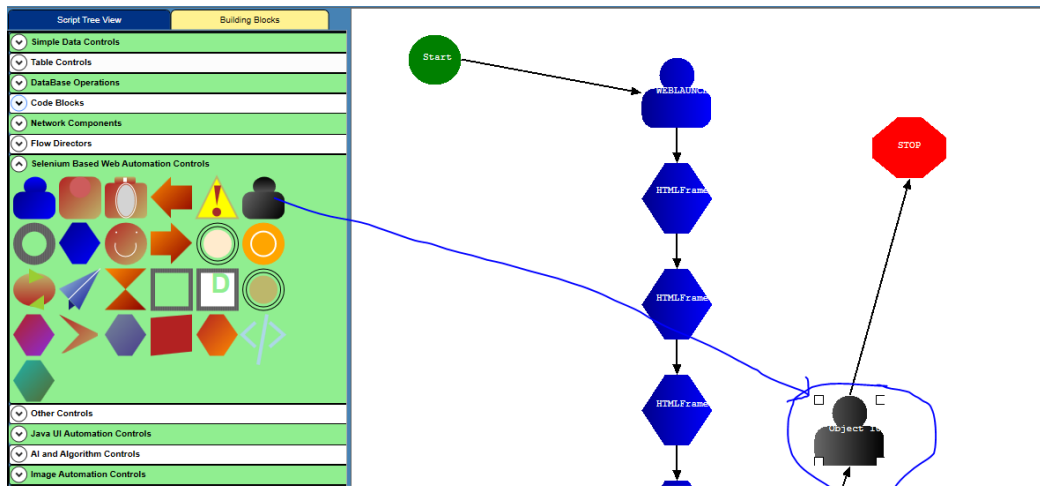
Highlight the object.



Click on **Return Actions** button

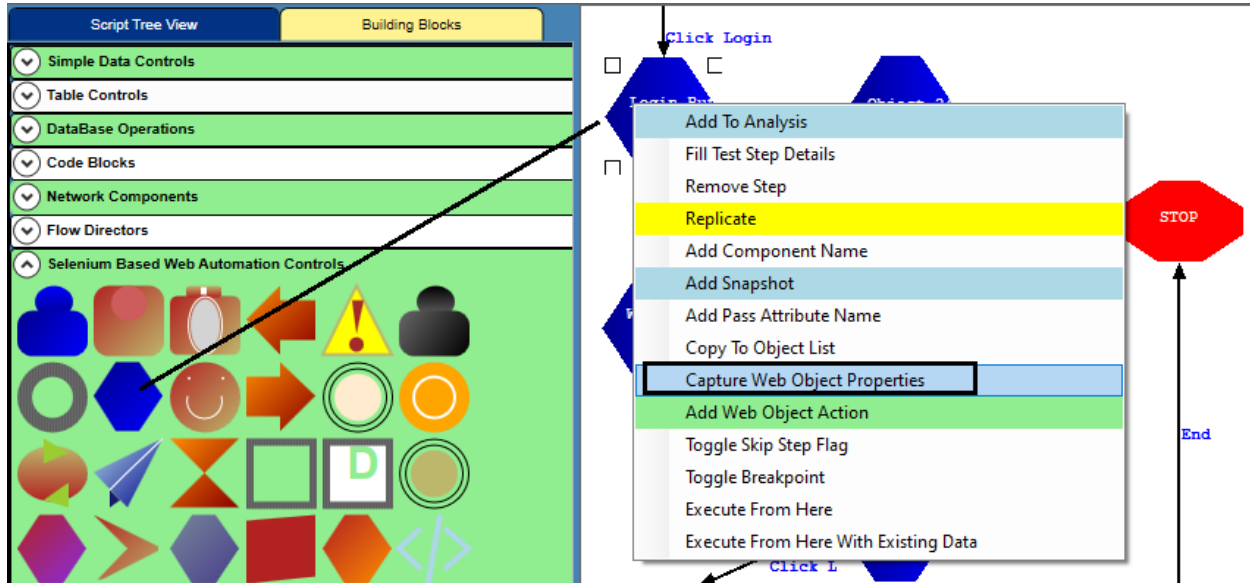


All the actions are returned to the object list. Drag the file to the Test Case Builder. Add a Start and end Block. You have a script ready to be executed now (we optionally added webpage close method before the end of script)

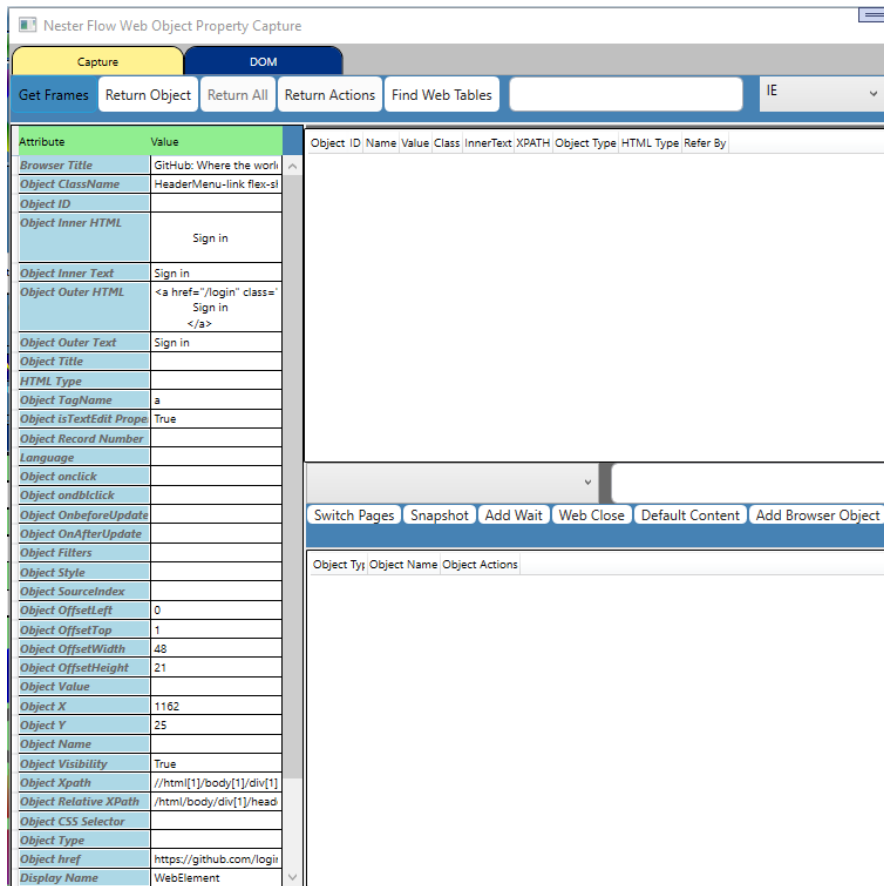


Return Single Object

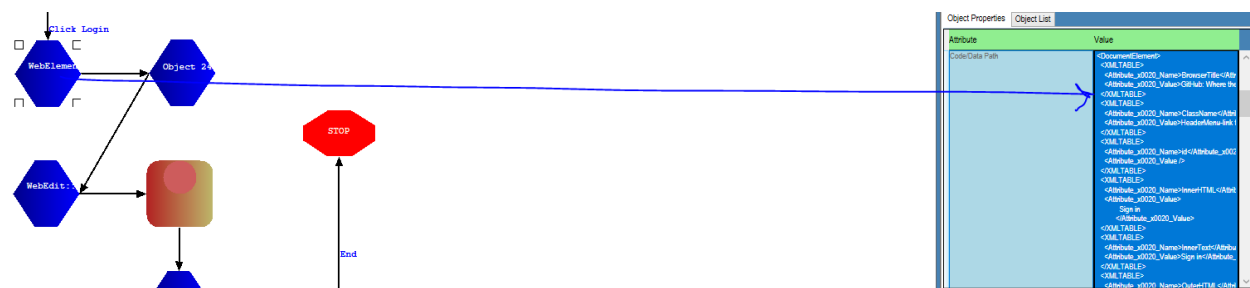
If we already built a flow and either want to assign an object to the web object step or update the step with latest update to the object properties, right click on the step and choose **Capture Web Object Properties** option



HTML Object Spy opens and if the object already has properties assigned, it will display them on property grid.

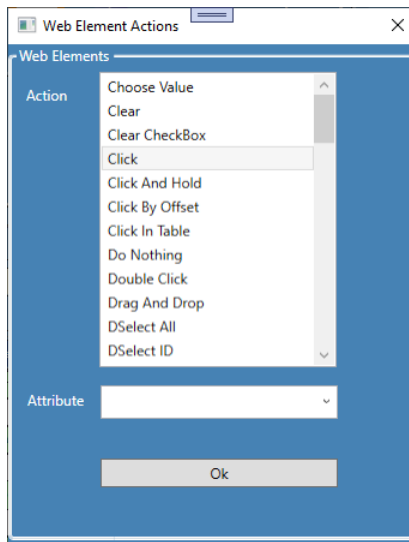


Spy a different object (the most recently captured or the most recently selected item will be shown on property grid). Click on **Return Object** button. Click **Yes** or **No** in the confirmation window (it will not make any difference as this is single object return mode)



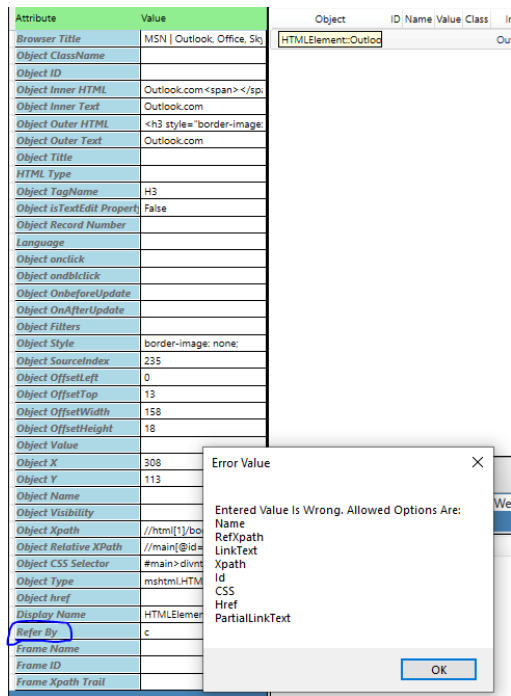
The object property reflects the selected value

Also, when clicking on any web object step on Test Case Builder, **Add Web Object Action** option will enable the user to choose a right object action to be performed with recently chosen action in clicked state.



Object Spy – Object Identification Strategy

The object spy allows the right object identification strategy to be chosen.



The property grid provides a field called **Refer By** which allows following values:

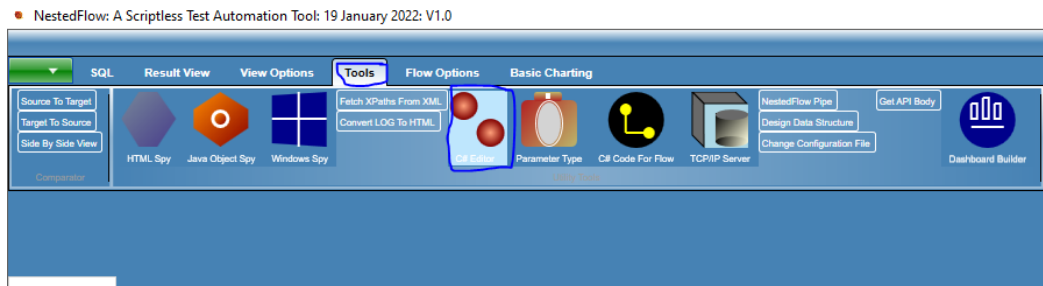
- **Name:** Object is identified using object Name
- **RefXpath:** This is default value. Allows object identification using shorter Xpath value the tool provides
- **LinkText:** Allows object identification using innerText
- **Xpath:** Allows object identification using Absolute Xpath of the object shown by the tool
- **Id:** Object is identified using object Id
- **CSS:** Object is identified using CSS selector value provided by the tool
- **Href:** Object is identified using Href value if existing
- **PartialLinkText:** Allows object identification using partial innerText

Create Code Out of Object Spy Actions

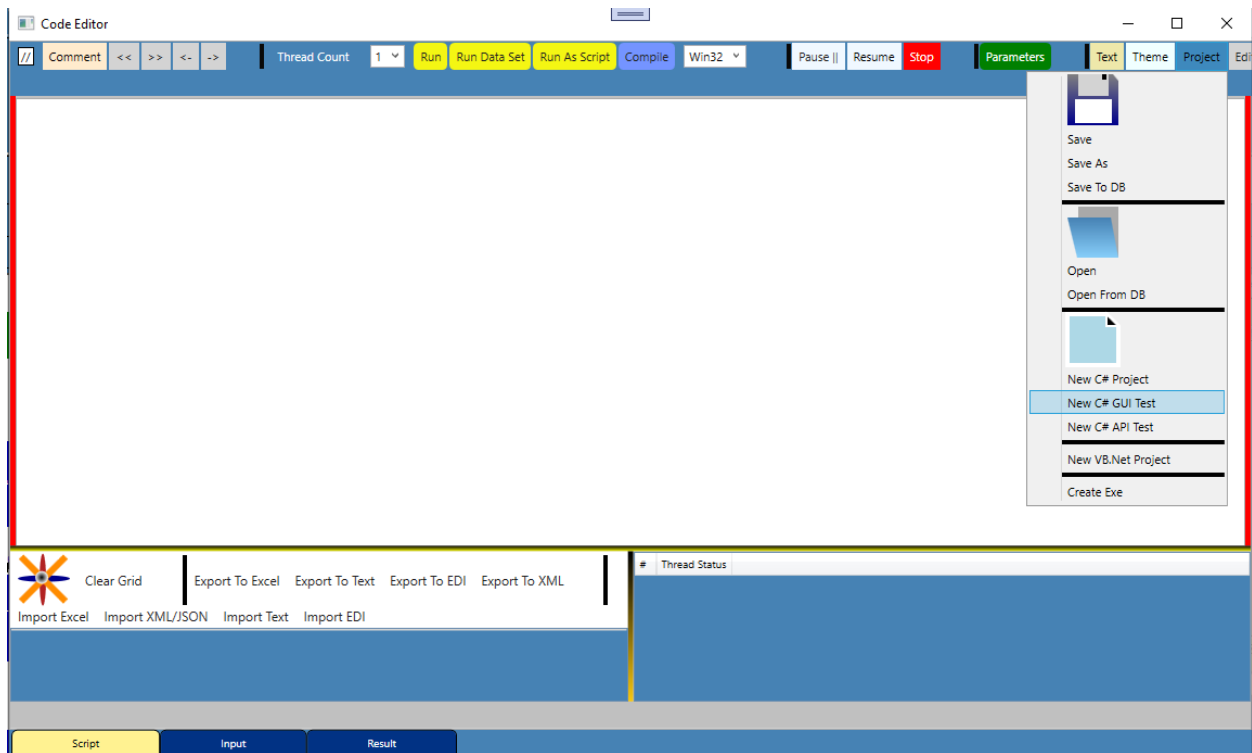
Nested Flow framework exposes all its functions through DLL. Using which fully scripted code can also be created using Visual Studio or using the Code editor available in the tool.

This methodology only copies selenium object and actions to Code it will ignore other functions in spy like Move to frame and other functions. Those functions need to be manually coded in the editor (There will be a separate document for a coded framework)

Click on **Tool** Ribbon Menu and click on **C# Editor**.



Click on Project → New C# GUI Test



This will create a barebone UI testing script:


```

New Project
1 using System;
2 using System.Xml;
3 using System.Data;
4 using System.Data.SqlClient;
5 using System.Windows.Forms;
6 using System.Drawing;
7
8 using System.IO;
9
10 using OpenQA.Selenium;
11
12 using OpenQA.Selenium.Support.UI;
13
14 using OpenQA.Selenium.IE;
15
16 using OpenQA.Selenium.Firefox;
17
18 using OpenQA.Selenium.Chrome;
19
20 using OpenQA.Selenium.Interactions;
21
22 using OpenQA.Selenium.Remote;
23
24 //Return value from Eval Code will be assigned to column
25 //DataTable T1: Whatever Data in Bottom Grid Of Query 1
26 //string s1 : Oracle User Name
27 //string s2 : Oracle Password
28 //string s3 : Second User Name
29 //string s4 : Second Password
30
31 namespace NSUpdateTable
32 {
33     public class CSUpdateTable
34     {
35         //Return value from Eval Code will be assigned to column
36         public object UpdateTable(DataTable T1,string s1,string s2,string s3,string s4,ref string s5)
37         {
38
39             XMLParser.LogFileActions.CreateLogFile();//GetLogFile
40             XMLParser.Web_Automation_Framework.SeleniumFunctions SF = new XMLParser.Web_Automation_Framework.SeleniumFunctions();
41
42             string webtype = "";
43             string pid = "";
44             SF.InitializeWebDriver("CHROME",pid,out webtype);
45             bool retval;
46             SF.LaunchDriver("https://<URL>/", out retval);
47
48             return (T1);
49         }
50     }
51 }

```

Update the URL and browser (default is CHROME)

Put cursor between browser definition and return Table statements as shown below

```

public object UpdateTable(DataTable T1,string s1,string s2,string s3,string s4,ref string s5)
{

    XMLParser.LogFileActions.CreateLogFile();//GetLogFile
    XMLParser.Web_Automation_Framework.SeleniumFunctions SF = new XMLParser.Web_Automation_Framework.SeleniumFunctions();

    string webtype = "";
    string pid = "";
    SF.InitializeWebDriver("EDGE",pid,out webtype);
    bool retval;
    SF.LaunchDriver("http://github.com", out retval);

    return (T1);
}

```

Right Click and choose **Web Object Spy** menu.

```

1 namespace NSUpdateTable
2 {
3     public class CSUpdateTable
4     {
5         //Return value from Eval Code will be assigned to column
6         public object UpdateTable(DataTable T1,string s1,string s2,string s3,string s4,ref string s5)
7         {
8
9             XMLParser.LogFileActions.CreateLogFile();//GetLogFile
10            XMLParser.Web_Automation_Framework.SeleniumFunctions SF = new XMLParser.Web_Automation_Framework.SeleniumFunctions();
11
12            string webtype = "";
13            string pid = "";
14            SF.InitializeWebDriver("EDGE",pid,out webtype);
15            bool retval;
16            SF.LaunchDriver("http://github.com", out retval);
17
18            return (T1);
19        }
20    }
21 }

```

Web Object Spy

Record Actions and Click on **Return Actions** button

Nester Flow Web Object Property Capture

Capture DOM

Get Frames Return Object Return All Return Actions Find Web Tables CHROME Launch Fetch Get DOM 5 Delayed Capture

Attribute	Value	Object	ID	Name	Value	Class	InnerText	XPATH	Object Type	HTML Type	Refer By
Browser Title	GitHub	HTMLElement				HeaderMenu-link flex-shrink-0 no-underline	Sign in	/html/BODY/DIV[1]/HEADER[1]/DIV[1]/DIV[2]/DIV[2]/A[1]			Ref[path]
Object ClassName	dropdown-item dropdo	HTMLElement		login		form-control input-block js-login-field		//*[@id="login_field"]	text		Ref[path]
Object ID		HTMLElement		pass		form-control form-control input-block js-pa		//*[@id="password"]	password		Ref[path]
Object Inner HTML	Sign out	HTMLElement		com	Sign	btn btn-primary btn-block js-sign-in-button		//*[@id="login"]/DIV[4]/FORM[1]/DIV[1]/INPUT[12]	submit		Ref[path]
Object Inner Text	Sign out	HTMLElement				avatar avatar-small circle		/html/BODY/DIV[1]/HEADER[1]/DIV[7]/DETAILS[1]/SUMMARY[1]			Ref[path]
Object Outer HTML	<button type="submit" Sign out </button>	HTMLElement				dropdown-item dropdown-signout	Sign ou	/html/BODY/DIV[1]/HEADER[1]/DIV[7]/DETAILS[1]/DETAILS-MEN	submit		Ref[path]
Object Outer Text	Sign out										
Object Title											
HTML Type	submit										
Object TagName	button										
Object isTextEdit Prope	True										
Object Record Number											
Language											
Object onclick											
Object onclick											
Object OnbeforeUpdate											
Object OnAfterUpdate											
Object Filters											
Object Style											
Object SourceIndex											
Object OffsetLeft	0										
Object OffsetTop	439										
Object OffsetWidth	178										
Object OffsetHeight	29										
Object Value											
Object X	1692										
Object Y	488										
Object Name											
Object Visibility	True										
Object Xpath	/html[1]/body[1]/div[1]										
Object Relative Xpath	/html/BODY/DIV[1]/HEA										
Object CSS Selector	html:nth-of-type(1)>but										
Object Type											
Object href											
Display Name	HTMLElement										

JavaScript: Click

Switch Pages Snapshot Add Wait Web Close Default Content Add Browser Object Add Custom Object

Object Type Object Name Object Actions

WEBLAUNCH CHROME

HTMLElement <NewDataSe <Table1> <Action>Jav

HTMLElement <NewDataSe <Table1> <Action>Ent

HTMLElement <NewDataSe <Table1> <Action>Ent

HTMLElement <NewDataSe <Table1> <Action>Jav

HTMLElement <NewDataSe <Table1> <Action>Jav

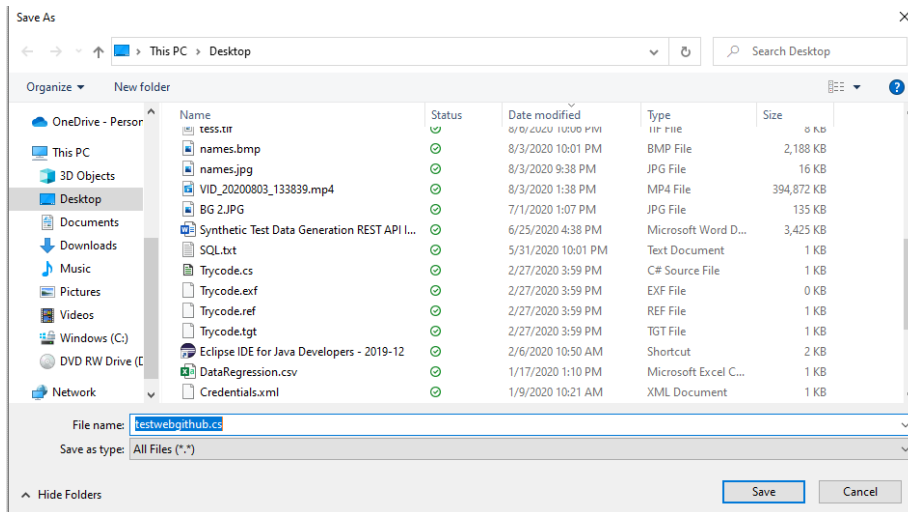
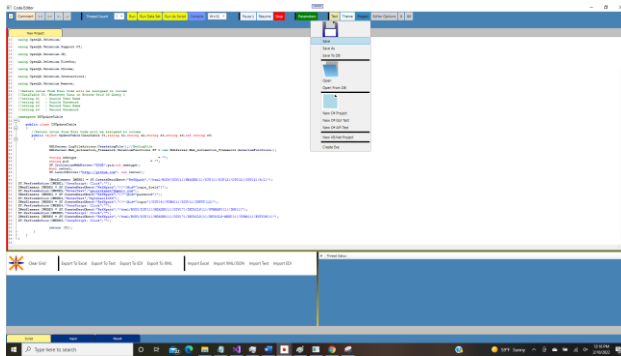
Recorded script returns to editor

```

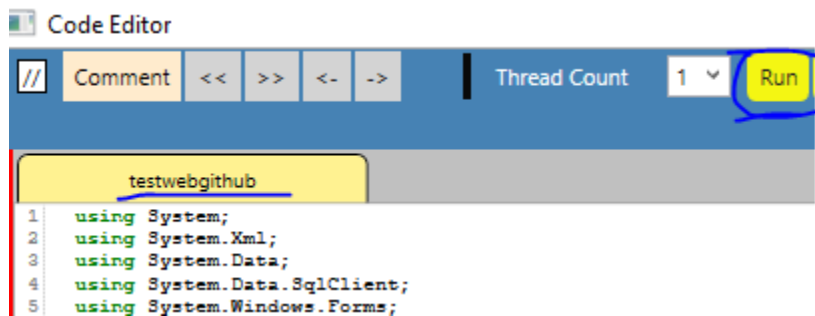
1 using OpenQA.Selenium.Support.UI;
2
3 using OpenQA.Selenium.IE;
4
5 using OpenQA.Selenium.Firefox;
6
7 using OpenQA.Selenium.Chrome;
8
9 using OpenQA.Selenium.Interactions;
10 using OpenQA.Selenium.Remote;
11
12 //Return value from Eval Code will be assigned to column
13 //DataTable T1: Whatever Data in Bottom Grid Of Query 1
14 //string s1 : Oracle User Name
15 //string s2 : Oracle Password
16 //string s3 : Second User Name
17 //string s4 : Second Password
18
19 namespace NSUpdateTable
20 {
21     public class CSUpdateTable
22     {
23         //Return value from Eval Code will be assigned to column
24         public object UpdateTable(DataTable T1, string s1, string s2, string s3, string s4, ref string s5)
25         {
26
27             XMLParser.LogFileActions.CreateLogFile(); //GetLogFile
28             XMLParser.Web_Automation_Framework.SeleniumFunctions SF = new XMLParser.Web_Automation_Framework.SeleniumFunctions();
29
30             string webtype = "";
31             string pid = "";
32             SF.InstallLeNoDriver("EDGE", pid, out webtype);
33             bool retval;
34             SF.LaunchDriver("http://github.com", out retval);
35
36             IWebElement IWEE1 = SF.CreateDescObect("RefXpath", "/html/BODY/DIV[1]/HEADER[1]/DIV[1]/DIV[2]/DIV[2]/A[1]");
37             SF.PerformAction(IWEE1, "JavaScript: Click", "");
38             IWebElement IWEE2 = SF.CreateDescObect("RefXpath", "//*[@id=\"login_field\"]");
39             SF.PerformAction(IWEE2, "EnterText", "");
40             IWebElement IWEE3 = SF.CreateDescObect("RefXpath", "//*[@id=\"password\"]");
41             SF.PerformAction(IWEE3, "EnterText", "");
42             IWebElement IWEE4 = SF.CreateDescObect("RefXpath", "//*[@id=\"login\"]DIV[4]/FORM[1]/DIV[1]/INPUT[12]");
43             SF.PerformAction(IWEE4, "JavaScript: Click", "");
44             IWebElement IWEE5 = SF.CreateDescObect("RefXpath", "/html/BODY/DIV[1]/HEADER[1]/DIV[7]/DETAILS[1]/SUMMARY[1]/IMG[1]");
45             SF.PerformAction(IWEE5, "JavaScript: Click", "");
46             IWebElement IWEE6 = SF.CreateDescObect("RefXpath", "/html/BODY/DIV[1]/HEADER[1]/DIV[7]/DETAILS[1]/DETAILS-MENU[1]/FORM[1]/BUTTON[1]");
47             SF.PerformAction(IWEE6, "JavaScript: Click", "");
48
49             return (T1);
50         }
51     }
52 }

```

Save the script as .cs



Click on **Run** button.



Script will execute in coded mode.