JavaScriptExecutor

This is the next tutorial in selenium-java series. Please go through the previous tutorials before you start this one. In the last tutorial, we learned how to handle ajax based fields and dropdowns. In this tutorial we will learn about Java script executor!

What you will Learn:

1. JavaScriptExecutor
2. JavaScriptExecutor Exercise
3. Another Javascript exercise

JavaScriptExecutor

JavaScriptExecutor is used when Selenium Webdriver fails to click on any element due to any reason or due to some issue. For example, if an element is hidden, selenium webdriver might fail to identify it.

The core of selenium is built on java script. The wrapper around this core can be C#, java etc. The javascript language is understood by the browser. So basically as a user we would be writing the script in C#, java etc, it will then be converted to java script which would then be understood by browser. Earlier people used javascript commands to directly interact with browser since not much functionality was there in selenium. With the help of JavaScriptExecutor class you can fire any javascript command.

Navigate to https://ksrtc.in/oprs-web/guest/home.do

Inspect 'Leaving From' field, you will notice that the value of class attribute contains 'hidden' text: 'ui-helper-**hidden**-accessible'. So in such cases, selenium webdriver might face issues in identifying elements.

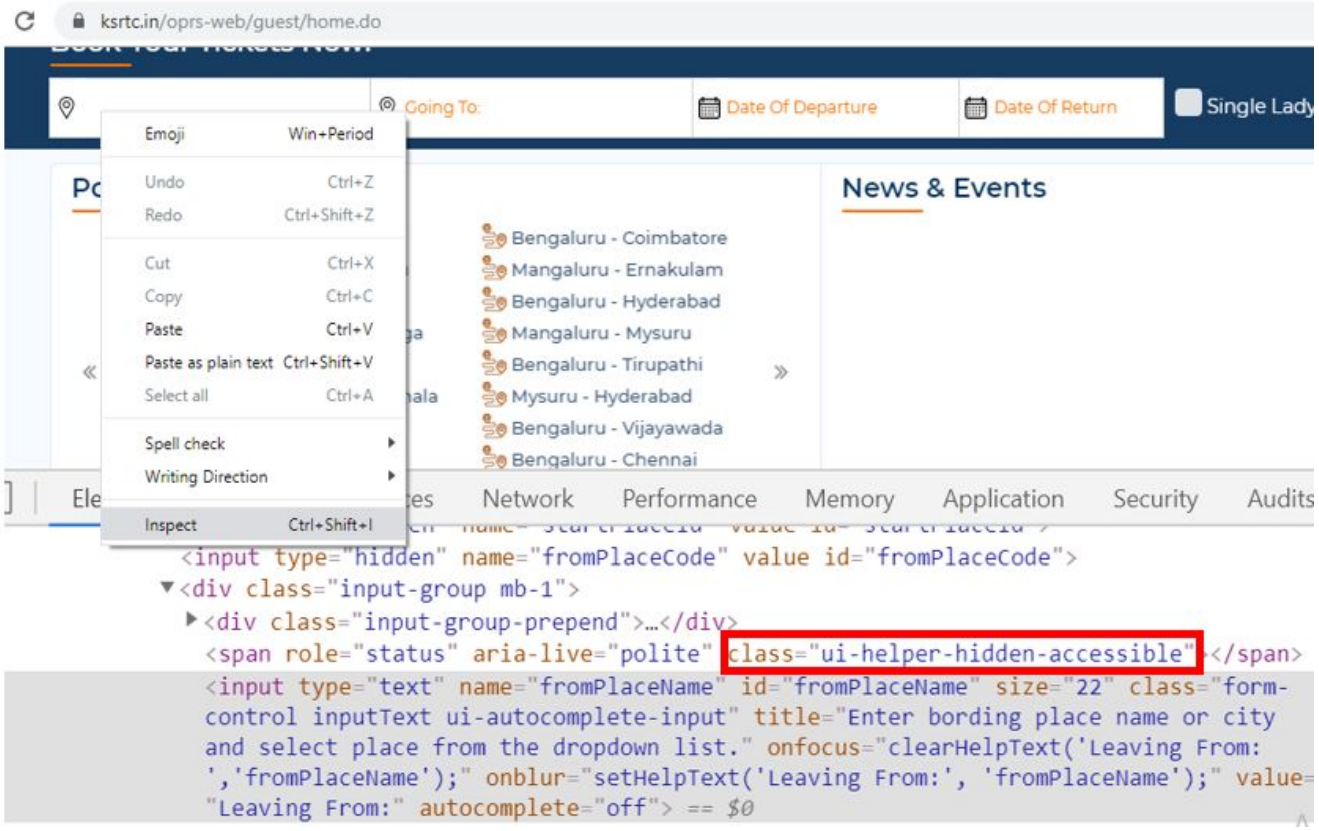

Figure 1

See figure 2. Drag the scroll bar slightly at the bottom so that you can see 'Leaving From' field.
Next, click 'Console' tab
Next, scroll down the console section so that you start seeing the > mark

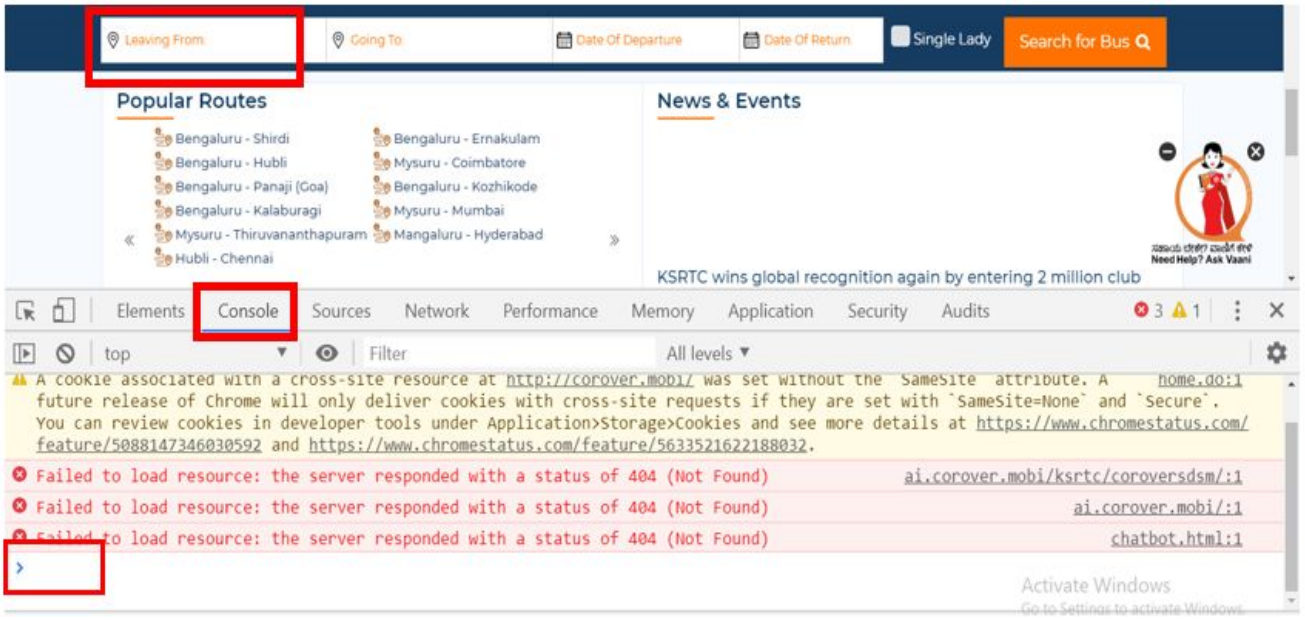

Figure 2

Type **document.getElementById("fromPlaceName")** in the console and hit enter, you should see some result

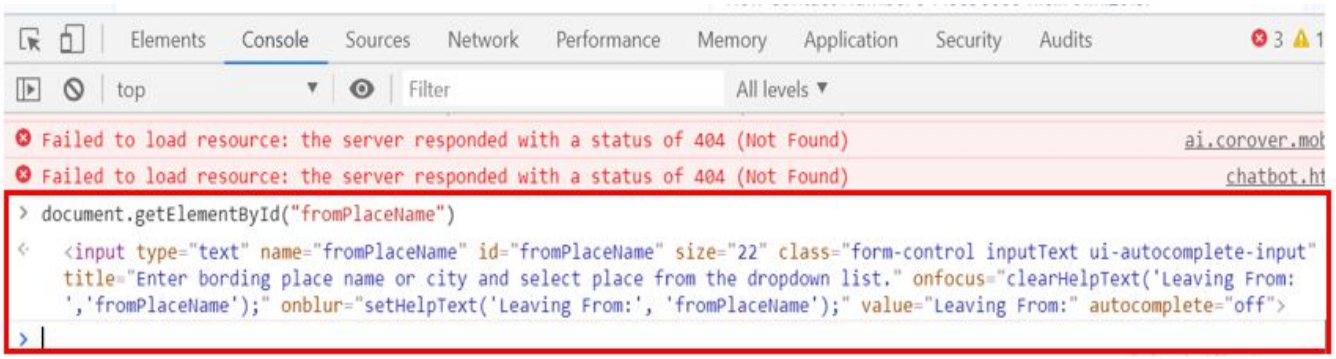

Figure 3

Do mouse hover over the console result, you will see the 'Leaving From' element gets highlighted, see below

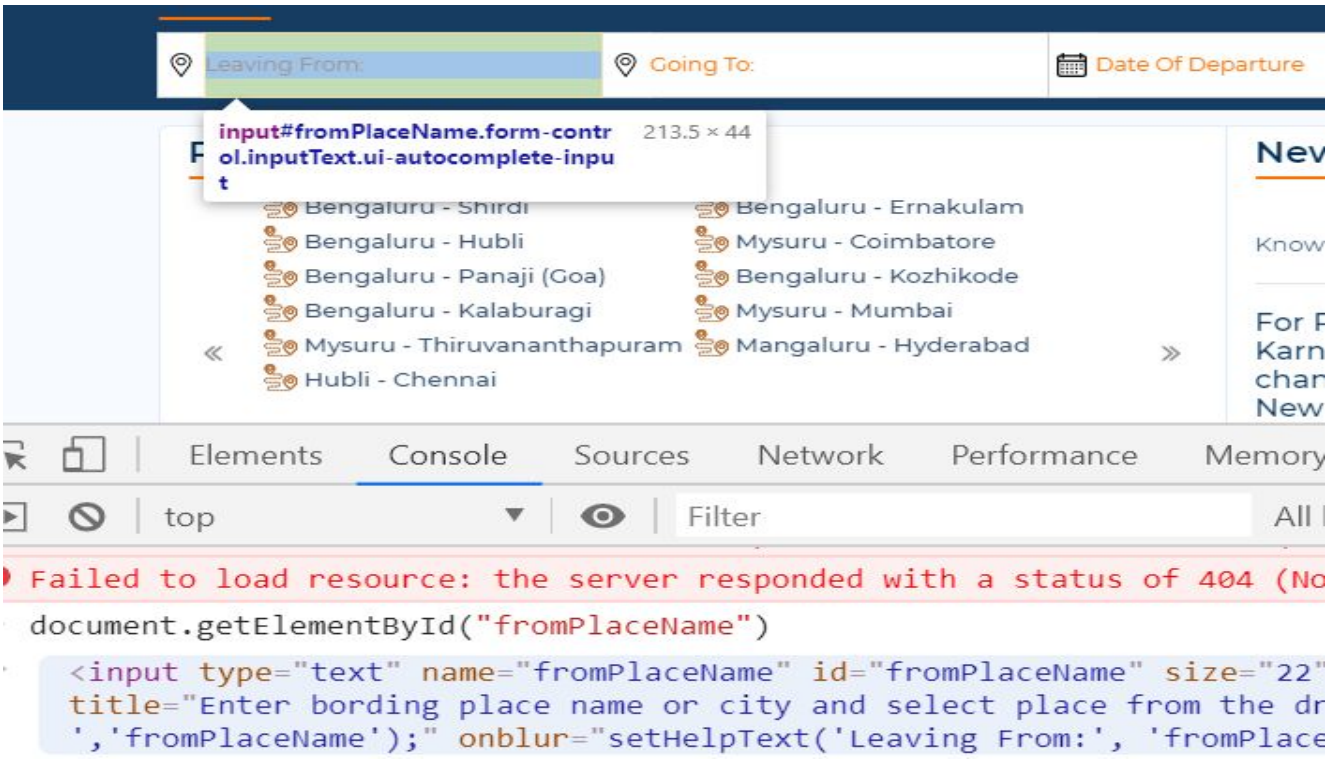

Figure 4

Type 'BENGALURU' in the 'Leaving From' field. You would see some options coming in the dropdown, see below

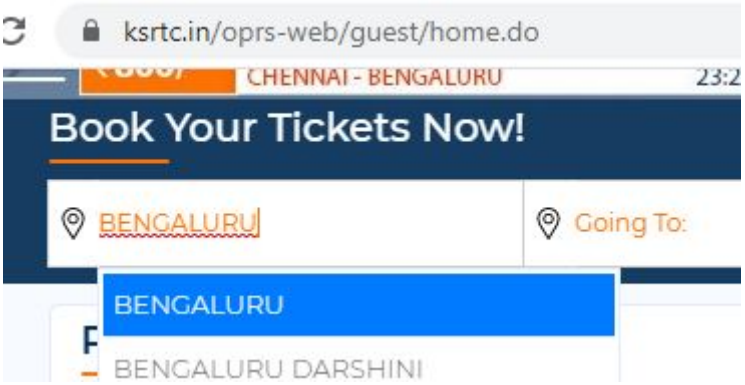

Figure 5

Select 'BENGALURU' from the dropdown, see below

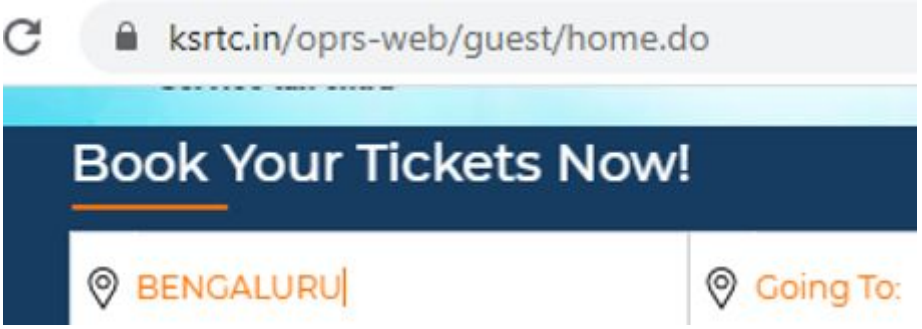

Figure 6

Now type **document.getElementById("fromPlaceName").value** and hit enter, you will see BENGALURU printed in the output

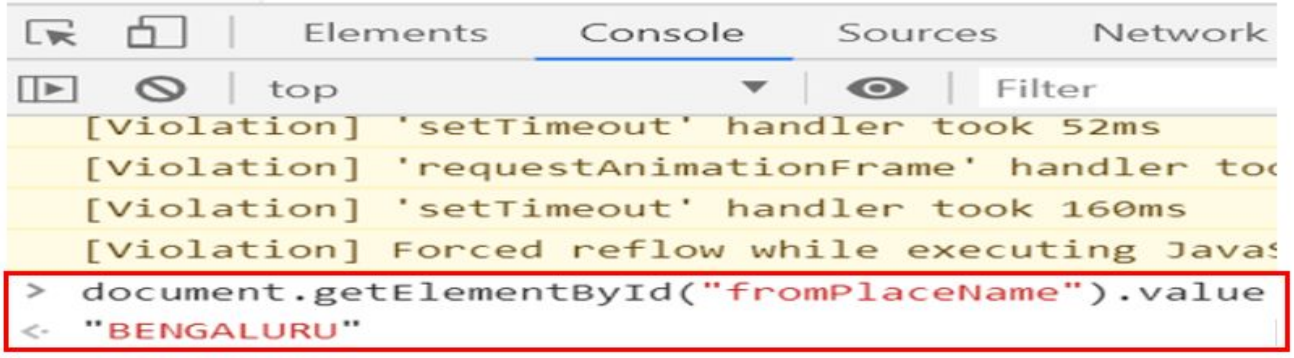

Figure 7

Let us try to automate these steps. Before we do that, the api document shows that JavascriptExecutor is an interface

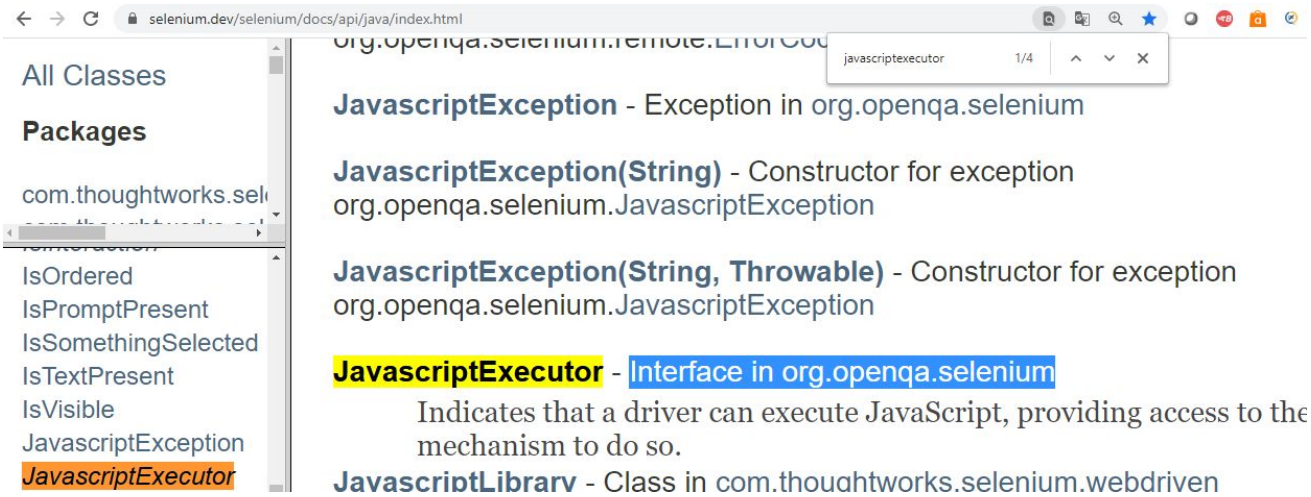

Figure 8

See below, we can see the driver classes that implement this interface

Figure 9

Create new class. Import JavascriptExecutor library (line#2). The line#18 will type BENG in the 'Leaving From' field. The lines#19,20 would perform the key down operation. Add line#22 wherein we are creating an object of JavascriptExecutor interface by Type casting. So we have cast driver object into JavascriptExecutor interface so that we can execute the javascript code in selenium WebDriver

```
1  import org.openqa.selenium.By;
2  import org.openqa.selenium.JavascriptExecutor;
3  import org.openqa.selenium.Keys;
4  import org.openqa.selenium.WebDriver;
5  import org.openqa.selenium.chrome.ChromeDriver;
6
7  public class JavascriptDemo {
8
9      public static void main(String[] args) {
10         System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\
11
12         WebDriver driver = null;
13
14         driver = new ChromeDriver();
15
16         driver.get("https://ksrtc.in");
17
18         driver.findElement(By.id("fromPlaceName")).sendKeys("BENG");
19         driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
20         driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
21
22         JavascriptExecutor js = (JavascriptExecutor)driver;
```

Figure 10

See figure#7 where we had executed the below line and got back the value BENGALURU in console
document.getElementById("fromPlaceName").value

Let us slightly modify above line by adding a semicolon at the end:
document.getElementById("fromPlaceName").value;

Next, let us add backslashes before the 2 double quotes:
document.getElementById(\"fromPlaceName\").value;

Next, let us put it within double quotes
“document.getElementById(\"fromPlaceName\").value;”

Next, put a semicolon at the end:
“document.getElementById(\"fromPlaceName\").value;”;

Next copy the above string and put it in a string variable, line#23 below. Make sure that there is only single back slash before start and closing double quote “

```
16         driver.get("https://ksrtc.in");
17
18         driver.findElement(By.id("fromPlaceName")).sendKeys("BENG");
19         driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
20         driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
21
22         JavascriptExecutor js = (JavascriptExecutor)driver;
23         String script = "document.getElementById(\"fromPlaceName\").value;";
```

Figure 11

In line#23, add ‘return’ keyword to store the value in variable

```
16          driver.get("https://ksrtc.in");
17
18          driver.findElement(By.id("fromPlaceName")).sendKeys("BENG");
19          driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
20          driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
21
22          JavascriptExecutor js = (JavascriptExecutor)driver;
23          String script = "return document.getElementById(\"fromPlaceName\").value;";
```

Figure 12

Next let us add line#24 where we are calling executeScript & storing the o/p in variable. So executeScript() method provided by JavascriptExecutor interface is called by using reference variable js. Without type casting, the executeScript() method of JavascriptExecutor will not be available to use.

```
22          JavascriptExecutor js = (JavascriptExecutor)driver;
23          String script = "return document.getElementById(\"fromPlaceName\").value;";
24          String text = js.executeScript(script);
```

Figure 13

We get an error at line#24. Let us fix this error

```
22          JavascriptExecutor js = (JavascriptExecutor)driver;
23          String script = "return document.getElementById(\"fromPlaceName\").value;";
24          String text = (String) js.executeScript(script);
```

Figure 14

We will now print the ‘text’ in console, line#25

```
16          driver.get("https://ksrtc.in");
17
18          driver.findElement(By.id("fromPlaceName")).sendKeys("BENG");
19          driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
20          driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
21
22          JavascriptExecutor js = (JavascriptExecutor)driver;
23          String script = "return document.getElementById(\"fromPlaceName\").value;";
24          String text = (String) js.executeScript(script);
25          System.out.println(text);
```

Figure 15

Run the script, the text ‘BENGALURU DARSHINI’ is seen printed in console. So this is how we can use JavaScriptExecutor to work directly with browsers.

<terminated> JavascriptDemo [Java App
Starting ChromeDriver
Only local connection
[1584337765.220][WARN
Mar 16, 2020 11:19:27
INFO: Detected dialec
BENGALURU DARSHINI

Figure 16

Also BENGALURU DARSHINI is seen typed in 'Leaving From' field, see below

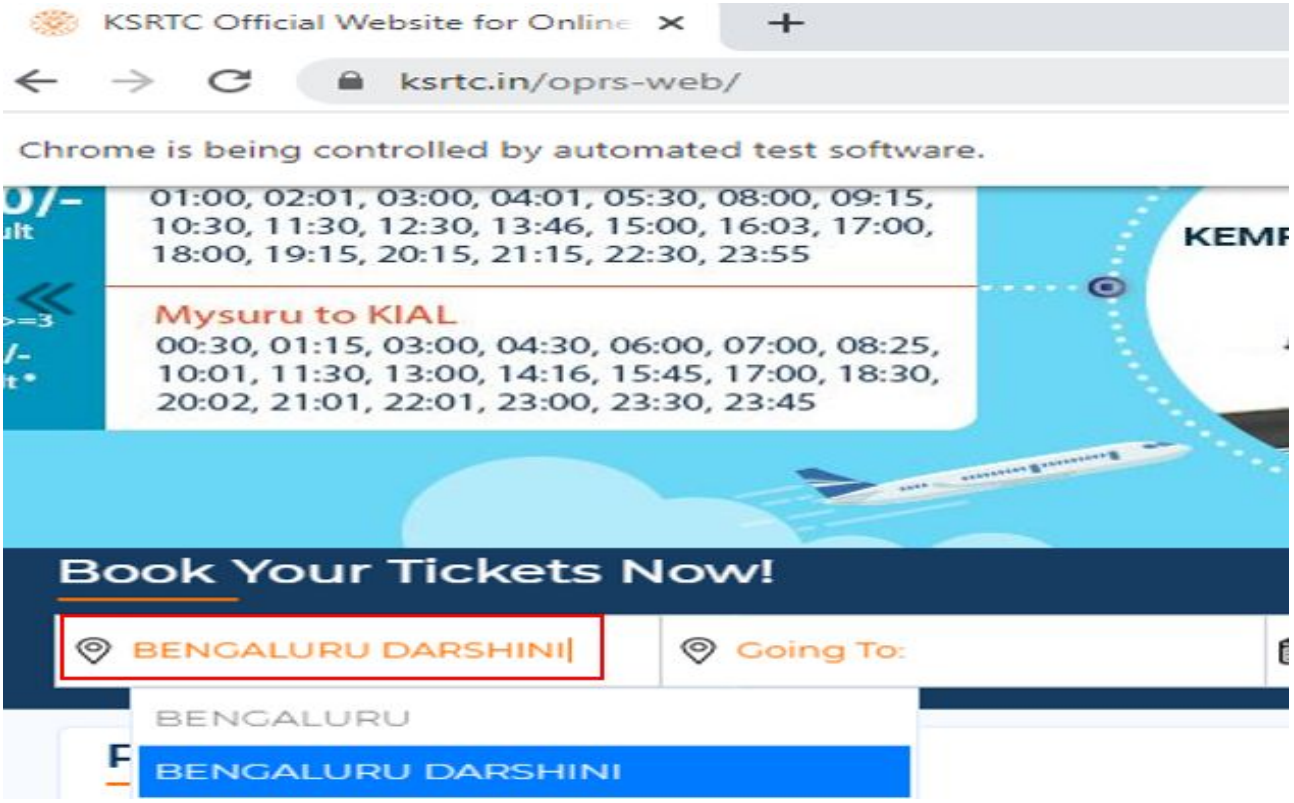

Figure 17

JavascriptExecutor Exercise

See below figure. When we type BENG in the 'Leaving From' field, we get a lot of options in the dropdown. The task here is to keep performing keyDown operation till BENGALURU INTERNATION AIRPORT gets selected.

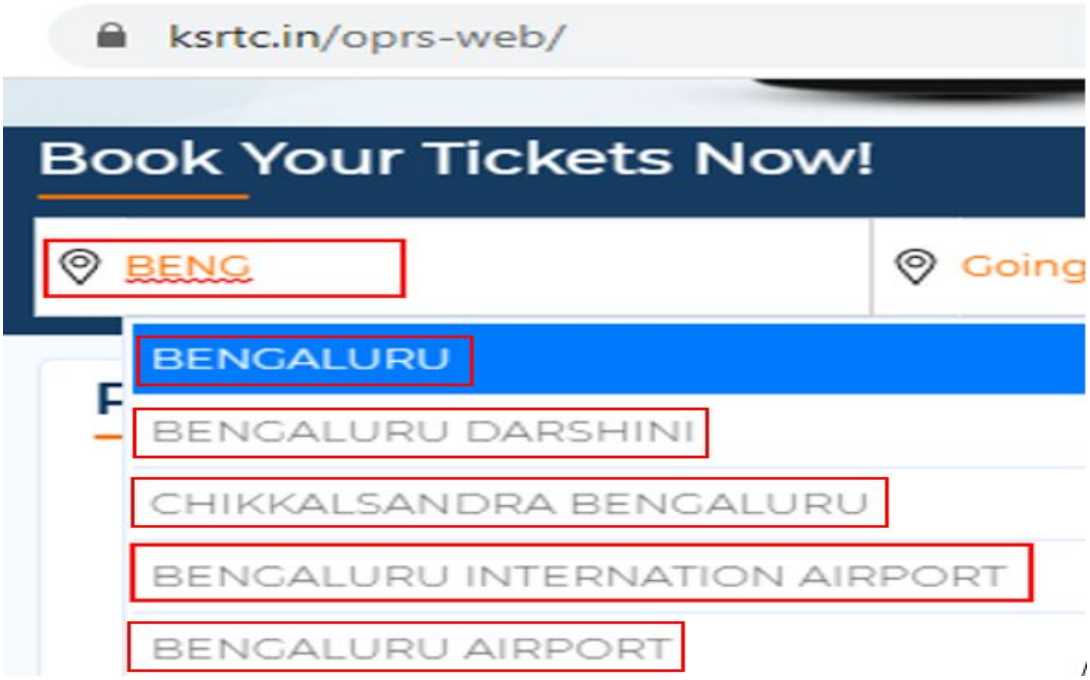

Figure 18

We will be using the 'while' loop to achieve our objective. See line#26. So, as long as the 'text' is not equal to BENGALURU INTERNATION AIRPORT, we will keep executing the code inside the 'while' block. Once the text equals BENGALURU INTERNATION AIRPORT, the control comes out of the 'while' loop. Notice that we have added some wait time in lines 17 and 29.

```
7  public class JavascriptDemo {
8
9      public static void main(String[] args) throws InterruptedException {
10         System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\Desktop\\TRA
11         WebDriver driver = null;
12         driver = new ChromeDriver();
13         driver.get("https://ksrtc.in");
14
15         driver.findElement(By.id("fromPlaceName")).sendKeys("BENG");
16         driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
17         Thread.sleep(1000);
18         driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
19
20         JavascriptExecutor js = (JavascriptExecutor)driver;
21         String script = "return document.getElementById(\"fromPlaceName\").value;";
22         String text = (String) js.executeScript(script);
23         //System.out.println(text);
24
25         //bengaluru internation airport
26         while(!text.equalsIgnoreCase("bengaluru internation airport"))
27         {
28             driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
29             Thread.sleep(1000);
30             text = (String) js.executeScript(script);
31             System.out.println(text);
32         }
```

Figure 19

Run the script, notice that BENGALURU INTERNATION AIRPORT is shown in ‘Leaving From’ field.

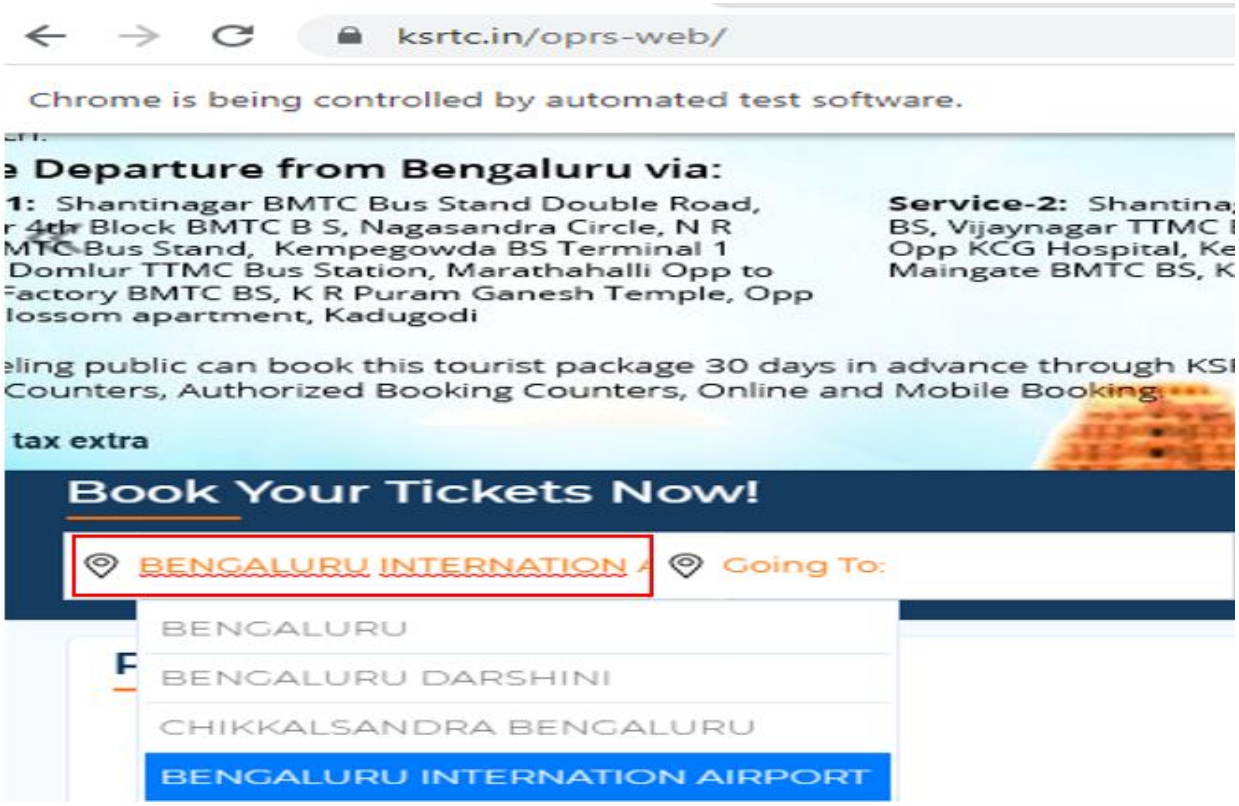

Figure 20

Also see below. Nothing gets printed in the console after BENGALURU INTERNATION AIRPORT.

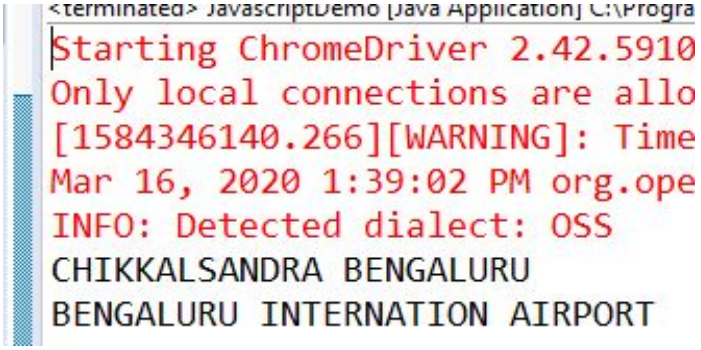

Figure 21

Another Javascript exercise

In continuation of above exercise, notice that, the option “BENGALURU NATION AIRPORT” is not present in ajax list at all

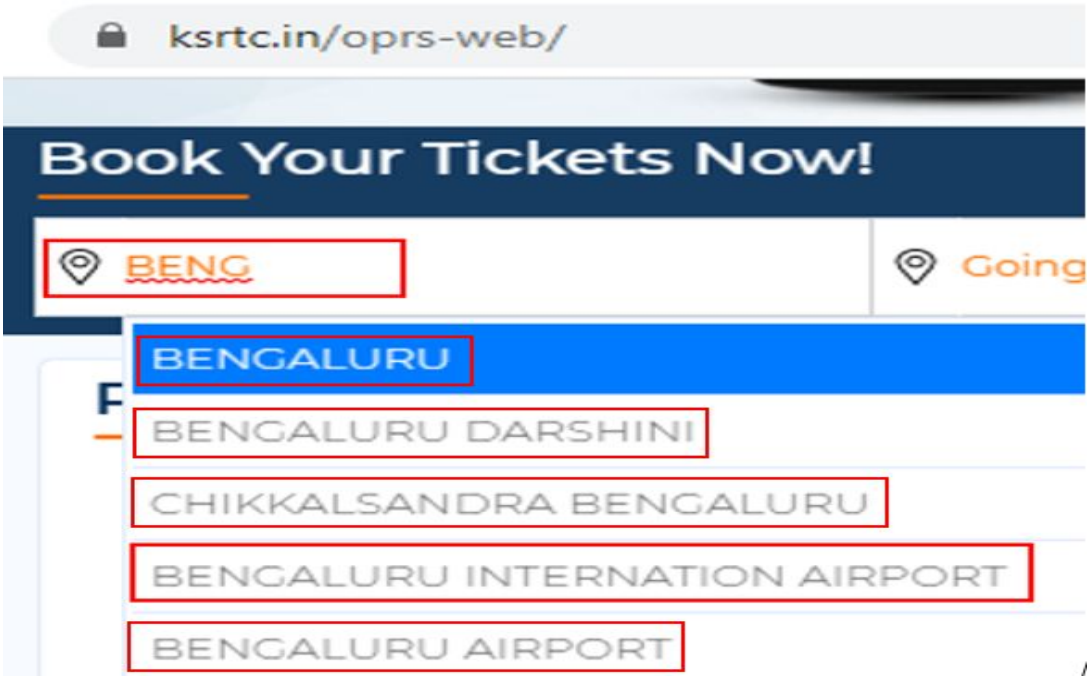

Figure 22

So if we try to search for an invalid text (line#23 in the below figure), than we will never come out of the 'while' loop, it would be an infinite loop. So there should be some mechanism to break the 'while' loop if the text is not found. The approach that we will follow here is that: we will perform keydown operation for only 10 times viz we will check just 10 options. If the desired text does not match any of the 10 options, than we will break out from the 'while' loop (line#31 in the below figure).

We initiate the counter i=0 at line#22, increment it on line#25, put an 'if' condition at line#30, break at line#31, write sop lines 33-37

```
12          driver = new ChromeDriver();
13          driver.get("https://ksrtc.in");
14          driver.findElement(By.id("fromPlaceName")).sendKeys("BENG");
15          driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
16          Thread.sleep(1000);
17          driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
18          JavascriptExecutor js = (JavascriptExecutor)driver;
19          String script = "return document.getElementById(\"fromPlaceName\").value;";
20          String text = (String) js.executeScript(script);
21          //System.out.println(text);
22          int i = 0;
23          while(!text.equalsIgnoreCase("bengaluru nation airport"))
24          {
25              i++;
26              driver.findElement(By.id("fromPlaceName")).sendKeys(Keys.DOWN);
27              Thread.sleep(1000);
28              text = (String) js.executeScript(script);
29              System.out.println(text);
30              if(i>10)
31                  break;
32          }
33          if(i>10)
34          {
35              System.out.println("Element not found");
36          }else
37              System.out.println("Element found");
```

Figure 23

When we run, the script tries for 10 runs, than prints 'Element not found'

```
Console  PyUnit
<terminated> Exercisejavascript2 [Java Application] C:\Progr
Only local connections are allow
[1575448206.163][WARNING]: Timec
Dec 04, 2019 2:00:08 PM org.oper
INFO: Detected dialect: OSS
CHIKKALSANDRA BENGALURU
BENGALURU INTERNATION AIRPORT
BENGALURU AIRPORT
BENG
BENGALURU
BENGALURU DARSHINI
CHIKKALSANDRA BENGALURU
BENGALURU INTERNATION AIRPORT
BENGALURU AIRPORT
BENG
BENGALURU
Element not found
```

Figure 24

So this is how we can make use of JavaScriptExecutor to perform operations using selenium. Thanks you for reading!