Locators (Part 3)

This is the next tutorial in the selenium-java series. Please go through the previous tutorial before you start this one. In the last tutorial, we continued looking at more locators. In this tutorial we will continue our exercises with locators!

What you will Learn:
Handle multiple elements having same attribute value
nth-of-type selector
nth-child selector
Custom xpath

Handle multiple elements having same attribute value:
What if multiple elements on a webpage have same attribute value? How will selenium identify the elements? So let's say there are 2 elements on the webpage that have same values for a particular attribute. When you inspect 'Email' and 'Password' fields on facebook page, we notice that both these fields have same value inside the 'class' attribute. Assume that these 2 fields do NOT have any other attribute viz let us assume that they do not have other attributes like: type, name, id, data-testid. So what this means is that, the 2 fields ('Email' and 'Password') only have 'class' attribute and both of the fields have same values, like seen below.

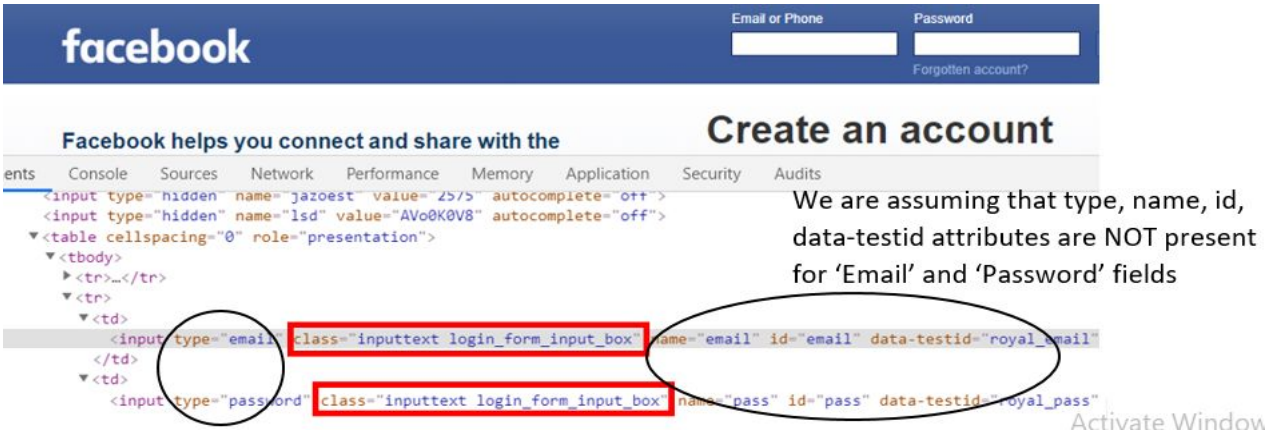

Figure 1

Selenium scans the page from left to right & top to bottom. As soon as it finds the first element that matches the attribute/value, it stops scanning any further, see below

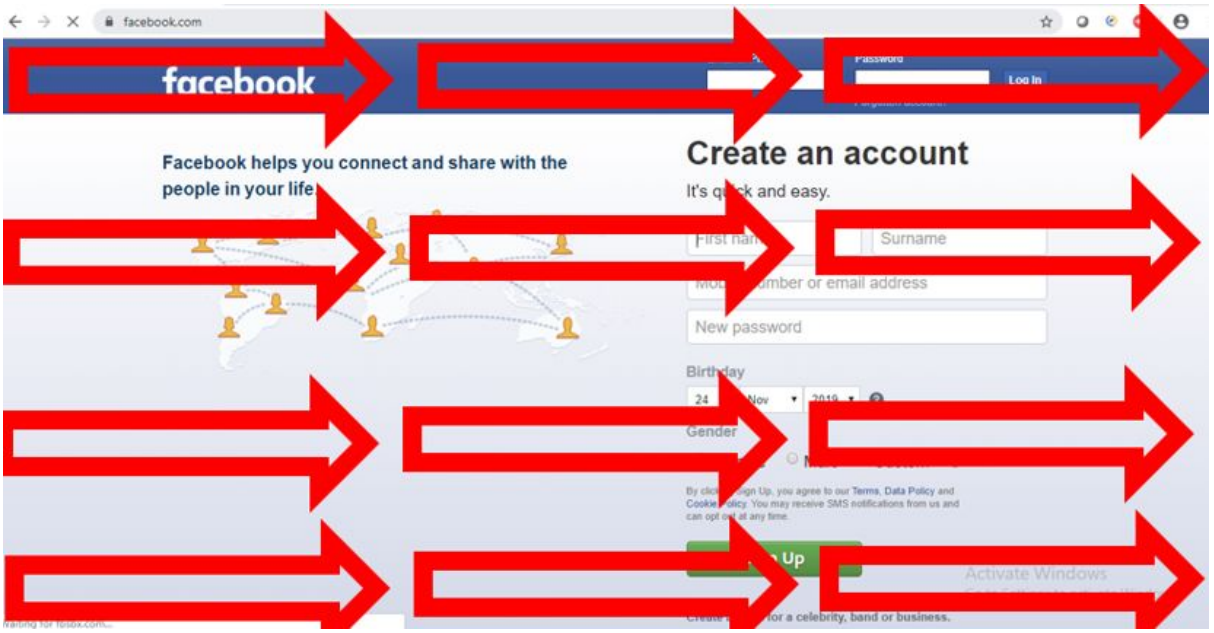

Figure 2

Create a new class. Let us use By.cssSelector locator to identify the text field

```
5  public class MultipleElements {
6
7      public static void main(String[] args) {
8          System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\Desktop\\TRAINING\\
9
10         WebDriver driver = null;
11
12         driver = new ChromeDriver();
13         driver.get("https://facebook.com");
14         driver.findElement(By.cssSelector("input.inputtext")).sendKeys("dummy@gmail.com");
```

Figure 3

When you run the above script, notice that first email field gets populated although the password field too has got the same attribute/value combination as we saw in figure 1.

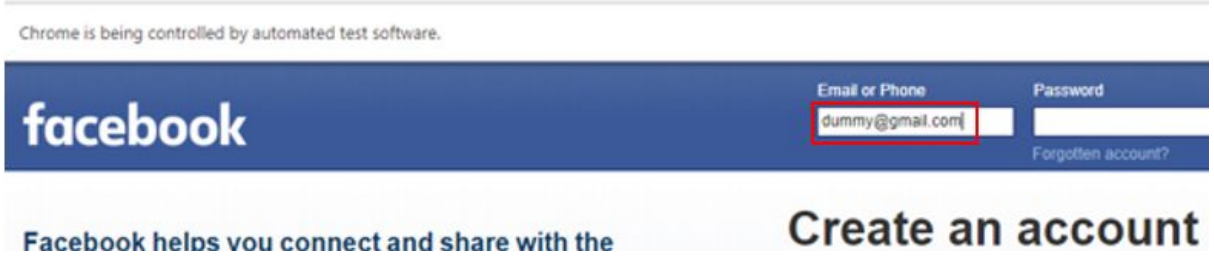

Figure 4

Let us comment line#14

```
5  public class MultipleElements {
6
7      public static void main(String[] args) {
8          System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\Desktop\\TRAINING\\Sof
9
10         WebDriver driver = null;
11
12         driver = new ChromeDriver();
13         driver.get("https://facebook.com");
14         //driver.findElement(By.cssSelector("input.inputtext")).sendKeys("dummy@gmail.com");
```

Figure 5

nth-of-type selector:

Please remember that, we are assuming that the 2 elements on the webpage ('Email' field and 'Password' field in the facebook page) do NOT have any other attribute except 'class' attribute. Both the fields, we know, have same values, see figure 1. We have seen that selenium finds the first matching field ('Email') and enters the text dummy@gmail.com into it. The important question that arises is, how do we enter the text in 'Password' field? So suppose, the business requirement is such that, we have to enter the text in 'Password' field and skip the 'Email' field, how do we achieve that? To visualize it better, look at below figure. We want to enter the text in 'Password' field and keep the 'Email' field empty. How do you achieve that assuming both the fields have only 'class' attribute having same value as seen in figure 1?

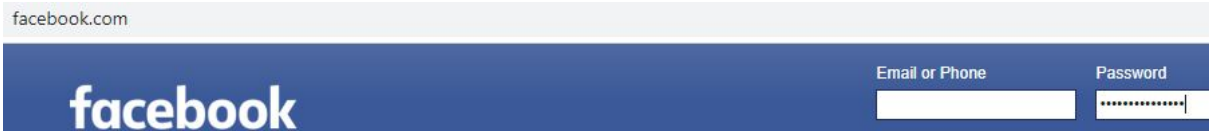

Figure 6

To achieve this requirement, we make use of "nth-of-type" selector.

To understand this, look at figure below. The 'Email' and 'Password' fields are present under <td> tag. The second 'td' tag represents the 'Password' field.

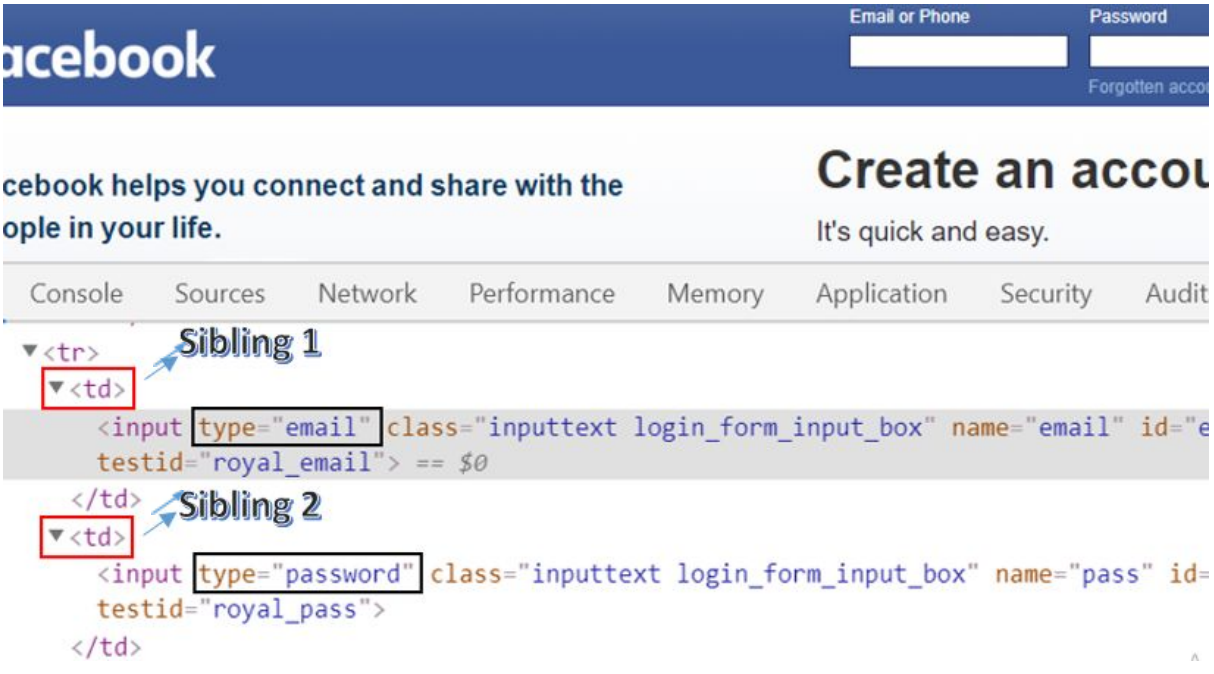

Figure 7

Let us open ChroPath & try to inspect: **td:nth-of-type(2) input.inputtext** by css selector, see below.
Notice that there is only 1 element matching and the 'Password' text box gets highlighted

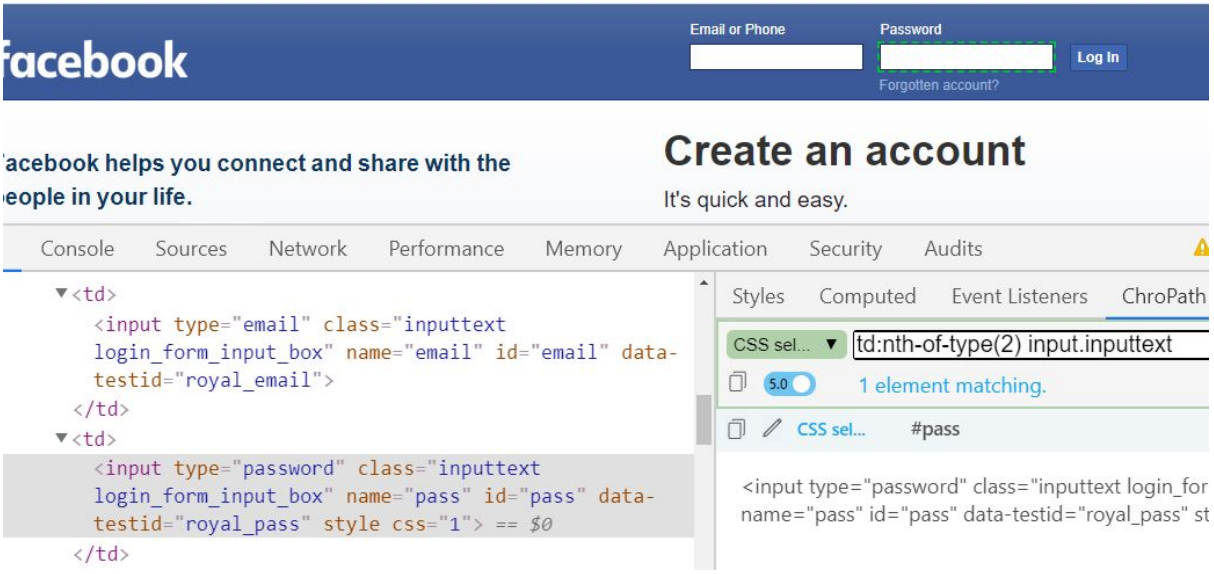

Figure 8

Now let change the number to 1 viz try to inspect: **td:nth-of-type(1) input.inputtext**
Notice that there is only 1 element matching and the 'Email' text box gets highlighted

Figure 9

What does it mean? What does it signify? Why is it that **td:nth-of-type(1) input.inputtext** is focussing to 'Email' field and **td:nth-of-type(2) input.inputtext** focussing to 'Password' field?

The locator **By.cssSelector("td:nth-of-type(2) input.inputtext")** locates an element on the webpage whose a) css path is input.inputtext b) represented by 'td' tag c) should be 2nd sibling. All these 3 conditions are fulfilled by 'Password' field, see figures 7 and 8.
Note that, both the 'td' tags in figure 8 are siblings to each other. The 1st sibling would be 'Email' field and 2nd would be 'Password' field. Now see below. Line#15 makes use of this css path.

```
5  public class MultipleElements {
6
7      public static void main(String[] args) {
8          System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\Desktop\\TRAINING\\Software\\chromedr
9
10         WebDriver driver = null;
11
12         driver = new ChromeDriver();
13         driver.get("https://facebook.com");
14         //driver.findElement(By.cssSelector("input.inputtext")).sendKeys("dummy@gmail.com");
15         driver.findElement(By.cssSelector("td:nth-of-type(2) input.inputtext")).sendKeys("dummy@gmail.com");
```

Figure 10

Make sure to comment line#14. Let us run the script. Notice that the text gets entered in 'Password' field. So we are able to skip the 'Email' field and thus able to meet our requirement.

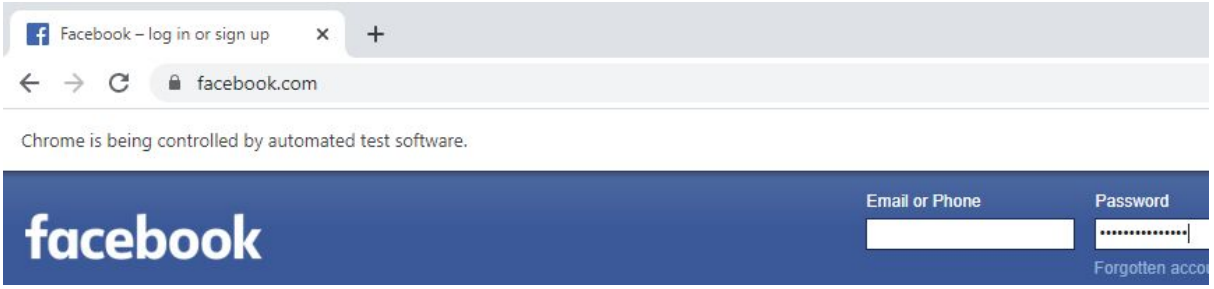

Figure 11

Now comment line#15. Add line#16

```
5  public class MultipleElements {
6
7      public static void main(String[] args) {
8          System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\Desktop\\TRAINING\\Software\\chromedri
9
10         WebDriver driver = null;
11
12         driver = new ChromeDriver();
13         driver.get("https://facebook.com");
14         //driver.findElement(By.cssSelector("input.inputtext")).sendKeys("dummy@gmail.com");
15         //driver.findElement(By.cssSelector("td:nth-of-type(2) input.inputtext")).sendKeys("dummy@gmail.com");
16         driver.findElement(By.cssSelector("td:nth-of-type(1) input.inputtext")).sendKeys("dummy@gmail.com");
```

Figure 12

Run the script, notice that, this time, the text gets entered in 'Email' field. So this is how we can make use of nth-of-type selector in identifying multiple elements that have the same attribute values.

Figure 13

nth-child selector:

Now look at figure below. The parent 'tr' tag has 2 child 'td' tags. Please remember that, like we have seen above, the child1 & child2 are siblings to each other.

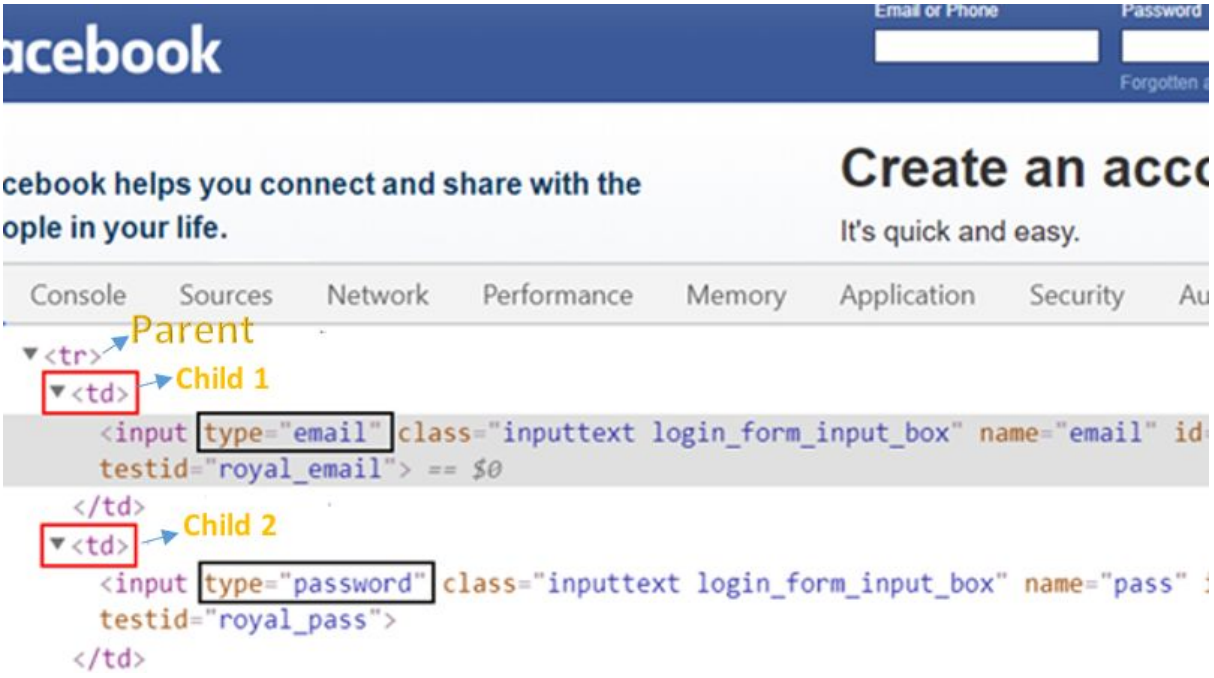

Figure 14

Comment line#16 and add line#17. We can also enter the text inside 'Email' and 'Password' fields by making use of the nth-child selector. Since 'td' is a child tag of 'tr' tag, we have written **tr > td** in line#17. Also, **nth-child(1)** would represent the 'Email' field in this case

```
5  public class MultipleElements {
6
7      public static void main(String[] args) {
8          System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\Desktop\\TRAINING\\Software\\chromedrive
9
10         WebDriver driver = null;
11
12         driver = new ChromeDriver();
13         driver.get("https://facebook.com");
14         //driver.findElement(By.cssSelector("input.inputtext")).sendKeys("dummy@gmail.com");
15         //driver.findElement(By.cssSelector("td:nth-of-type(2) input.inputtext")).sendKeys("dummy@gmail.com");
16         //driver.findElement(By.cssSelector("td:nth-of-type(1) input.inputtext")).sendKeys("dummy@gmail.com");
17         driver.findElement(By.cssSelector("tr > td:nth-child(1) input.inputtext")).sendKeys("dummy@gmail.com");
```

Figure 15

Let us run the script, notice that 'Email' field gets populated with text

Figure 16

Next, comment line#17, add line#18

```
5  public class MultipleElements {
6
7      public static void main(String[] args) {
8          System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\Desktop\\TRAINING\\Software\\chromedriver
9
10         WebDriver driver = null;
11
12         driver = new ChromeDriver();
13         driver.get("https://facebook.com");
14         //driver.findElement(By.cssSelector("input.inputtext")).sendKeys("dummy@gmail.com");
15         //driver.findElement(By.cssSelector("td:nth-of-type(2) input.inputtext")).sendKeys("dummy@gmail.com");
16         //driver.findElement(By.cssSelector("td:nth-of-type(1) input.inputtext")).sendKeys("dummy@gmail.com");
17         //driver.findElement(By.cssSelector("tr > td:nth-child(1) input.inputtext")).sendKeys("dummy@gmail.com");
18         driver.findElement(By.cssSelector("tr > td:nth-child(2) input.inputtext")).sendKeys("dummy@gmail.com");
```

Figure 17

Let us run the script, notice that 'Password' field gets populated with text and 'Email' field is skipped. So this is how, we can make use of nth-child selector!

Figure 18

Custom xpath:

When you launch facebook.com, you see 3 radio buttons at the bottom. None of the radio buttons are selected by default.

Figure 19

Assume that we don't have any attribute information for these radio buttons. Let's assume that the only information that we have with regard to radio buttons is that they are represented by 'span' tag. Now how will select the 3rd radio button? Let us inspect the 3 radio buttons. Notice below that the 3 radio buttons are children of 'span' tag. Also, the 3 radio buttons themselves are represented by 'span' tag.

Figure 20

Let us create our own custom xpath in this case. Notice above that the parent of all these radio buttons is 'span' tag having class='_5k_3'. This parent span tag has 3 child span tags. So the custom xpath **//span[@class='_5k_3']/span[1]** would represent first radio button, see below

Figure 21

Similarly the custom xpath **//span[@class='_5k_3']/span[2]** would represent second radio button

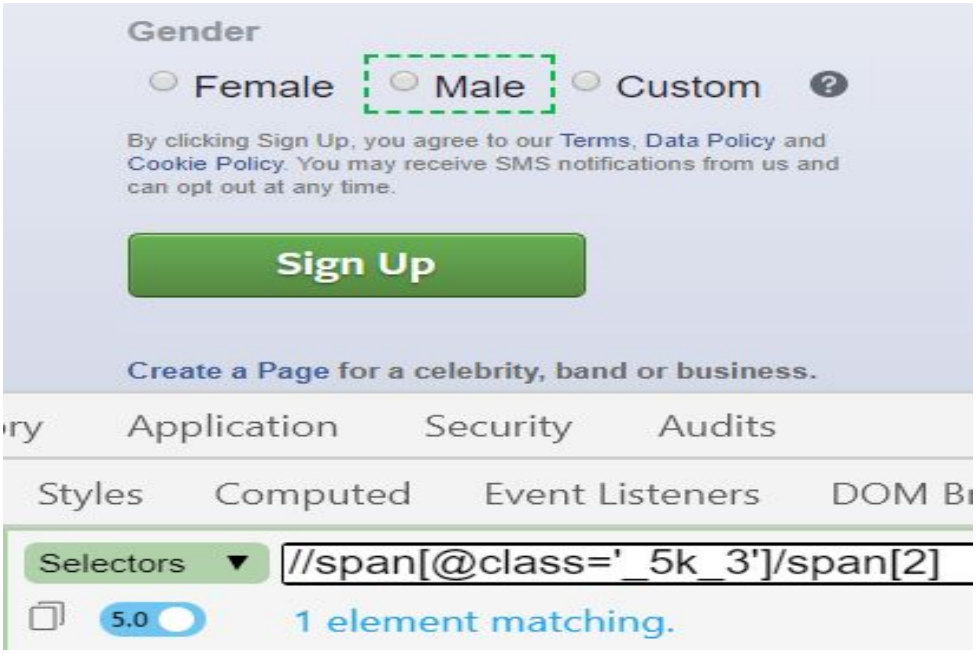

Figure 22

Finally the custom xpath **//span[@class='_5k_3']/span[3]** represents third radio button

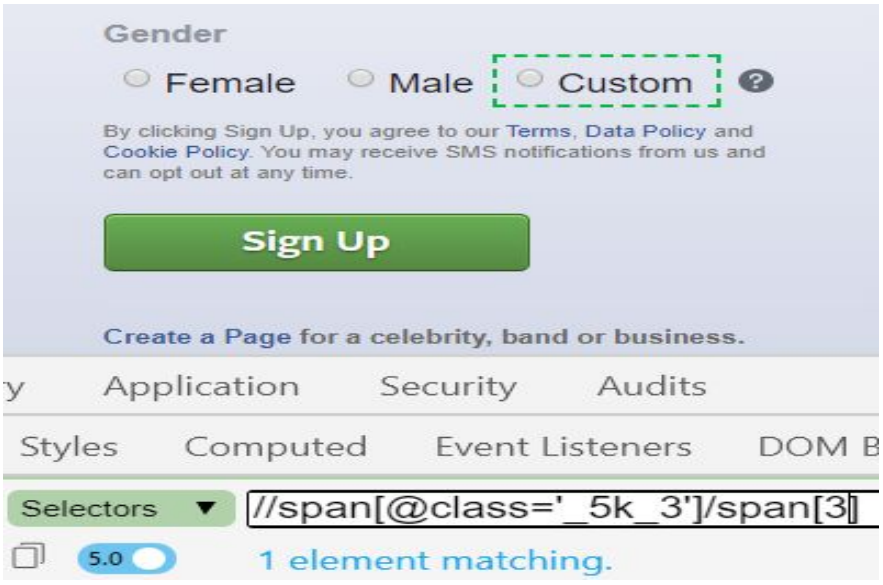

Figure 23

Comment line#18. Add line#19 to select the 3rd radio button

```
5  public class MultipleElements {
6
7      public static void main(String[] args) {
8          System.setProperty("webdriver.chrome.driver", "C:\\Users\\DELL\\Desktop\\TR
9
10         WebDriver driver = null;
11
12         driver = new ChromeDriver();
13         driver.get("https://facebook.com");
14         //driver.findElement(By.cssSelector("input.inputtext")).sendKeys("dummy@gma
15         //driver.findElement(By.cssSelector("td:nth-of-type(2) input.inputtext")).s
16         //driver.findElement(By.cssSelector("td:nth-of-type(1) input.inputtext")).s
17         //driver.findElement(By.cssSelector("tr > td:nth-child(1) input.inputtext")
18         //driver.findElement(By.cssSelector("tr > td:nth-child(2) input.inputtext")
19         driver.findElement(By.xpath("(//span[@class='_5k_3']/span[3])")).click();
```

Figure 24

Run the script, notice that 'Custom' radio button gets selected. Thus without using/knowing the attribute values of radio button, we are still able to select it.

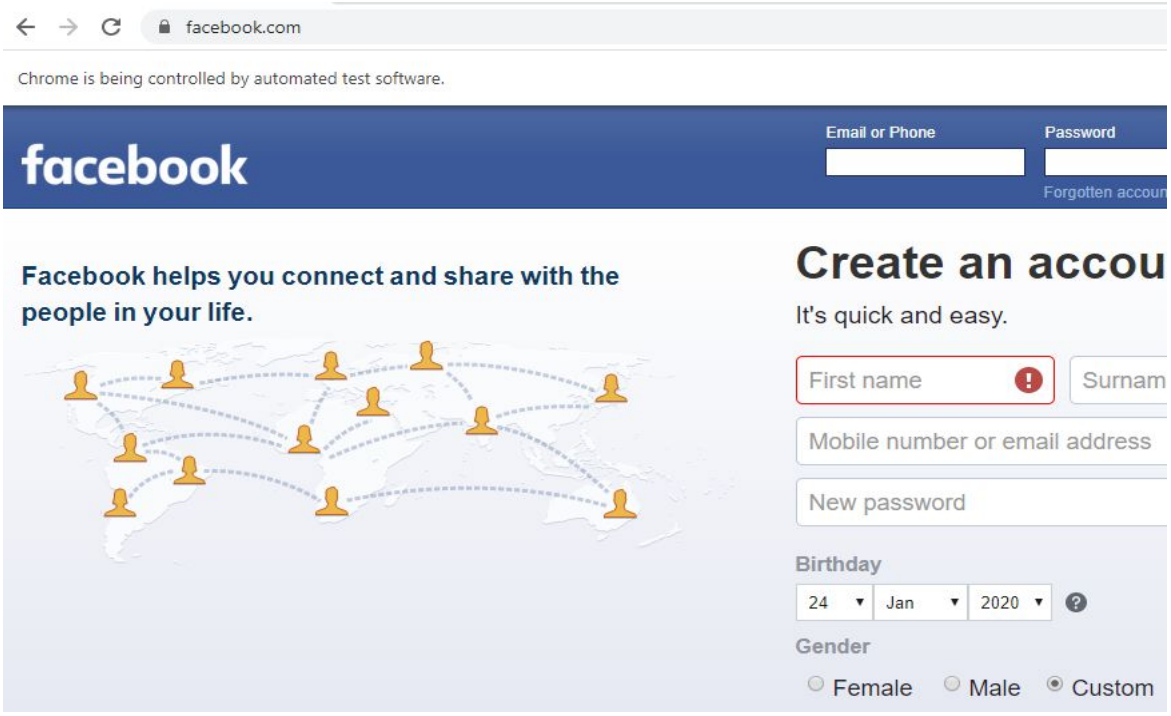

Figure 25

Similarly we can select first radio button

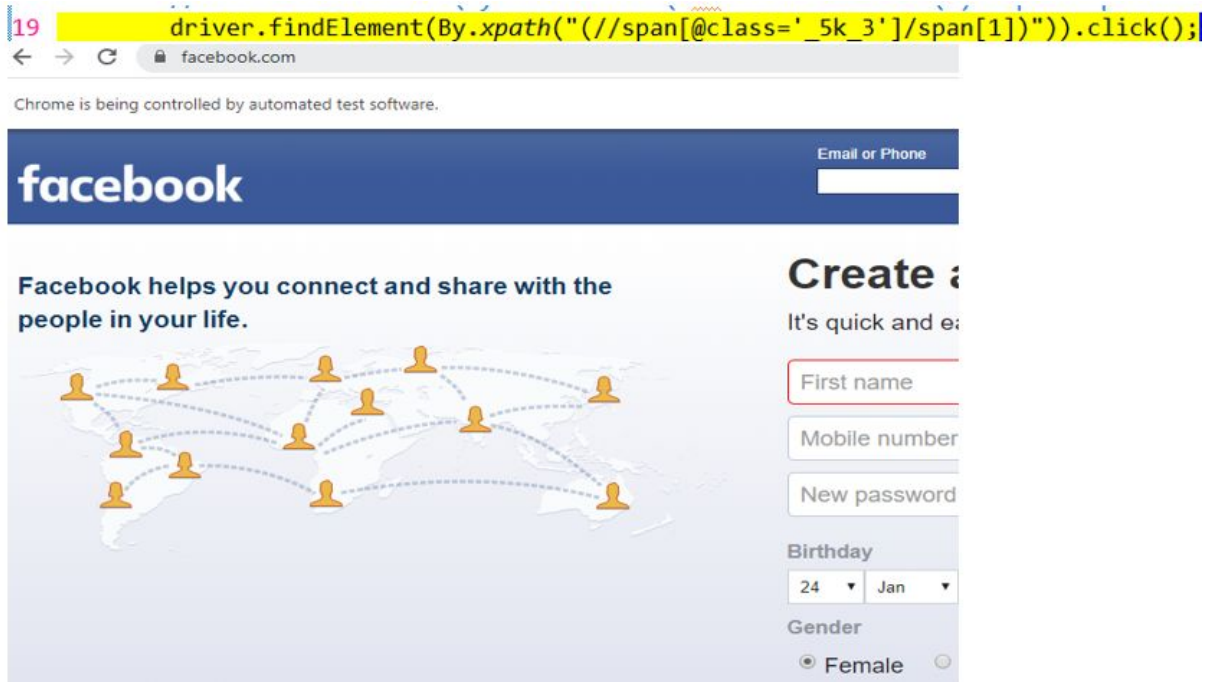

Figure 26

Similarly we can select second radio button

Figure 27

We will continue with locators in our next tutorial. Thank you for reading!