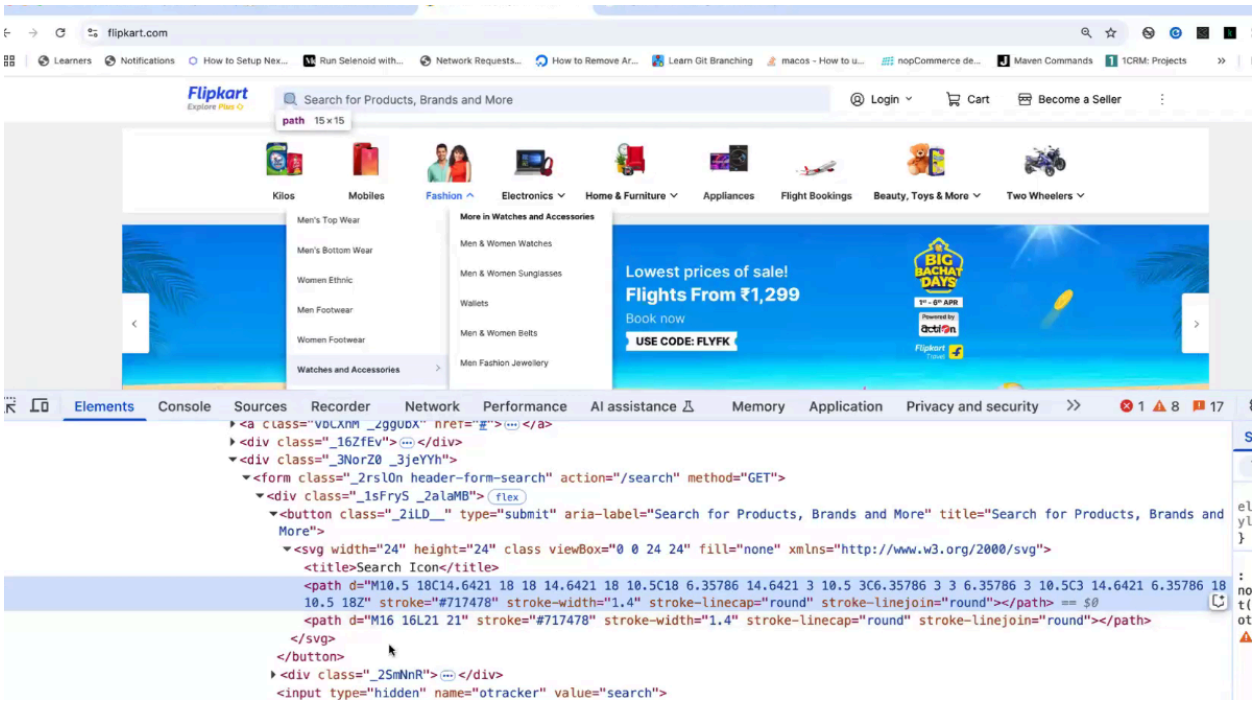
Svg -

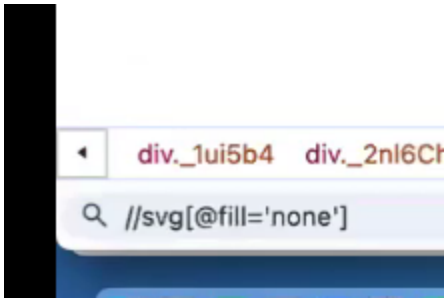
Scalar vector graphs.

Some different shapes like the below will be seen.

Example flipkart search icon-

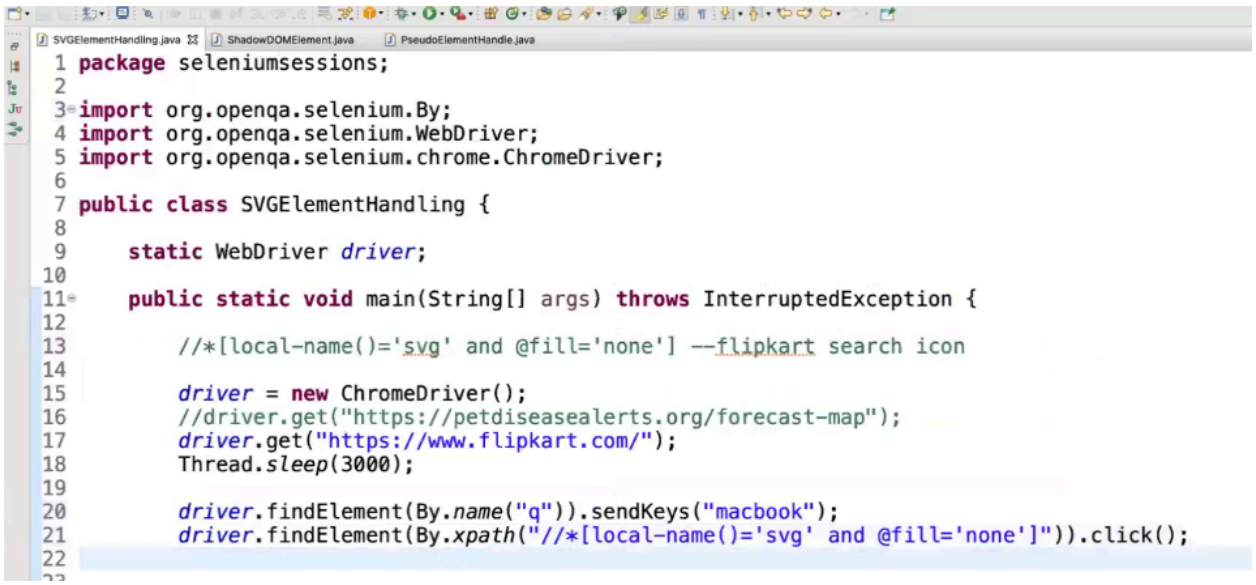


Normal locators wont work with svg.



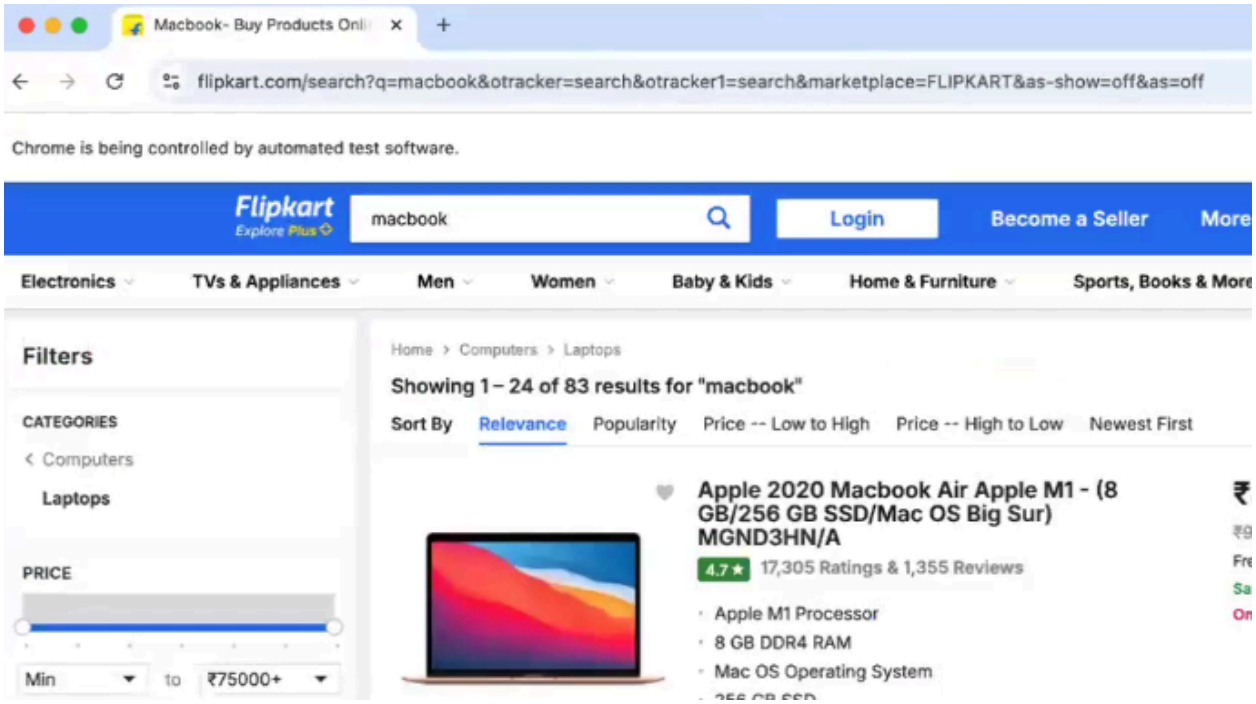
Special way to get svg-

Then add as many attributes as possible.



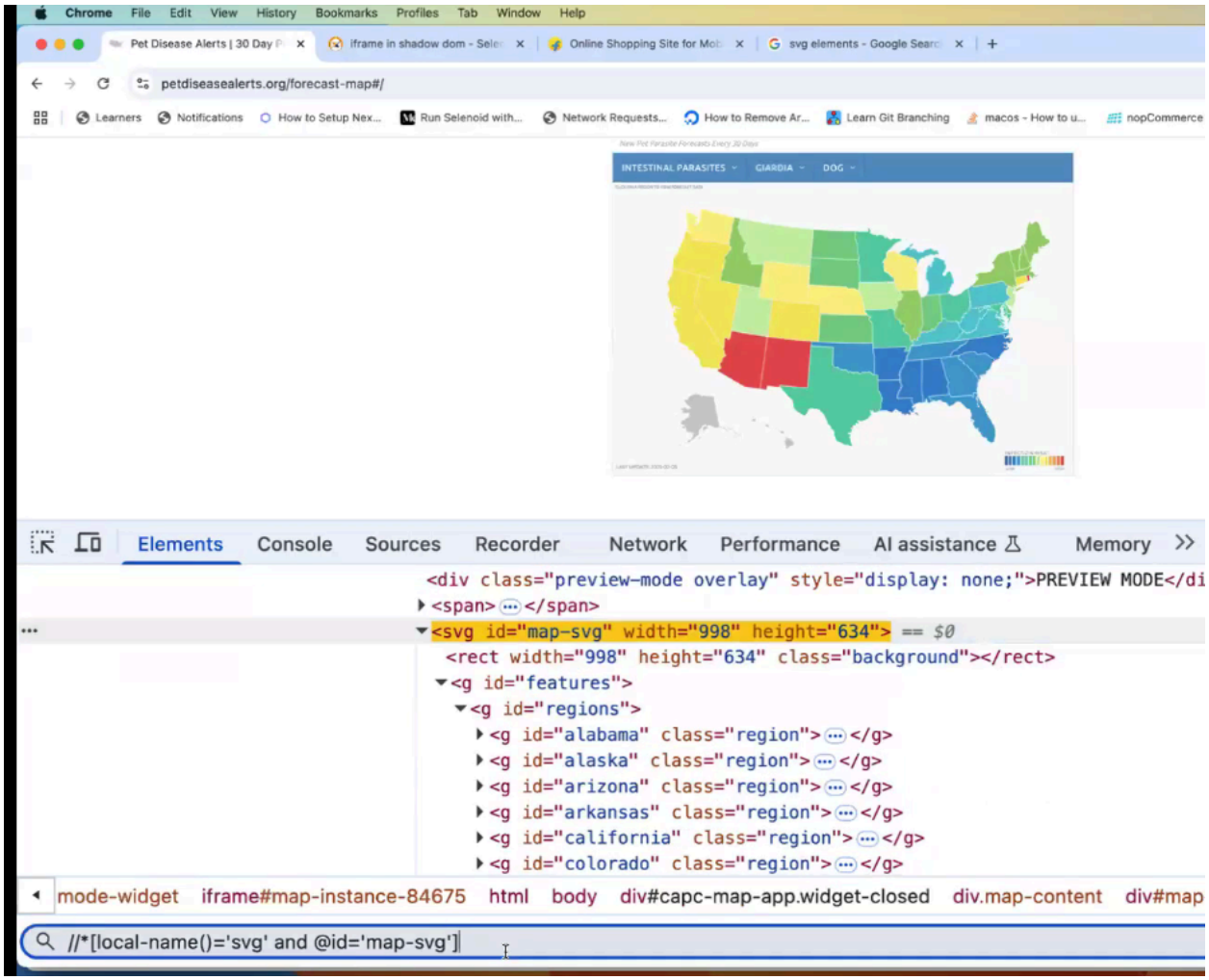
SVGElementHandling.java | ShadowDOMElement.java | PseudoElementHandle.java

```java
package seleniumsessions;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class SVGElementHandling {

    static WebDriver driver;

    public static void main(String[] args) throws InterruptedException {

        //*[local-name()='svg' and @fill='none'] --flipkart search icon

        driver = new ChromeDriver();
        //driver.get("https://petdiseasealerts.org/forecast-map");
        driver.get("https://www.flipkart.com/");
        Thread.sleep(3000);

        driver.findElement(By.name("q")).sendKeys("macbook");
        driver.findElement(By.xpath("//*[local-name()='svg' and @fill='none']")).click();

```

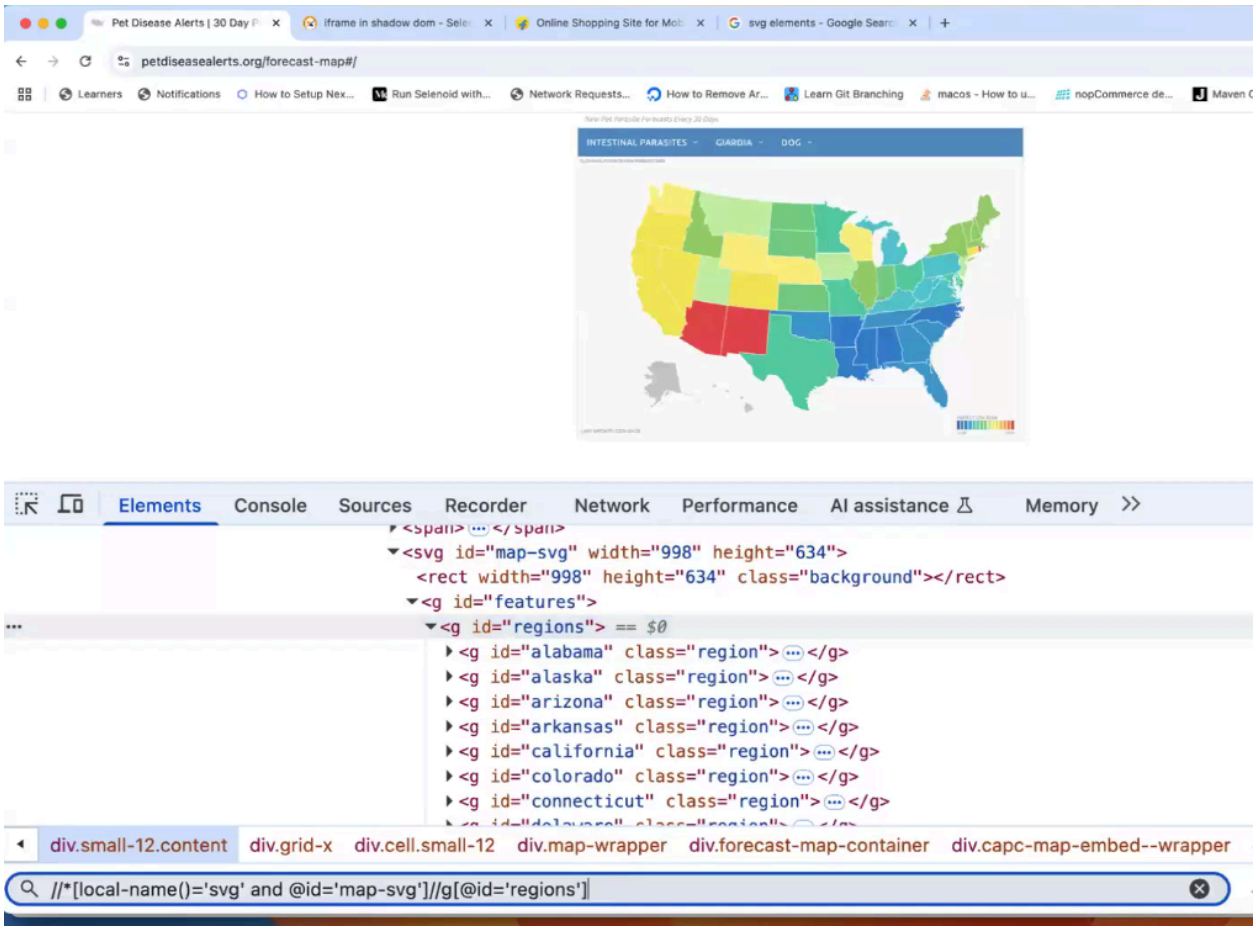


Click california-

All are svg.

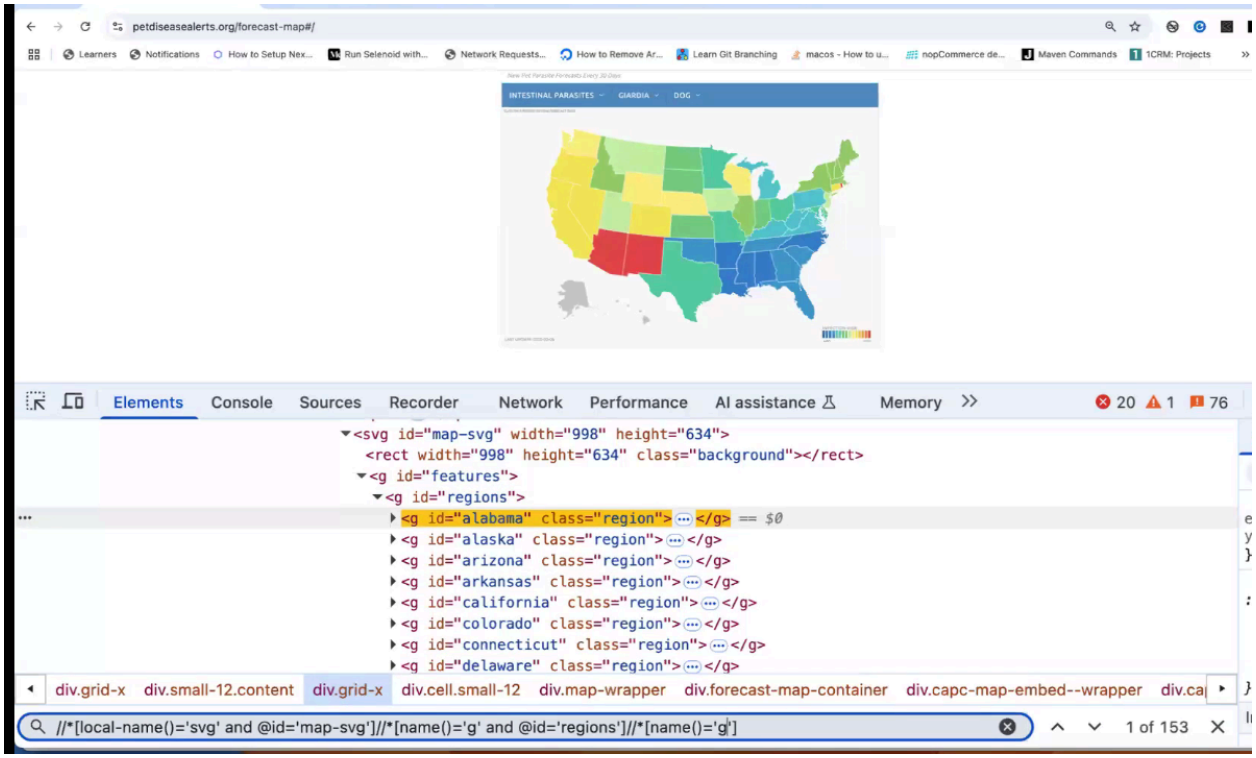


Normal xpaths or any locators wont work.

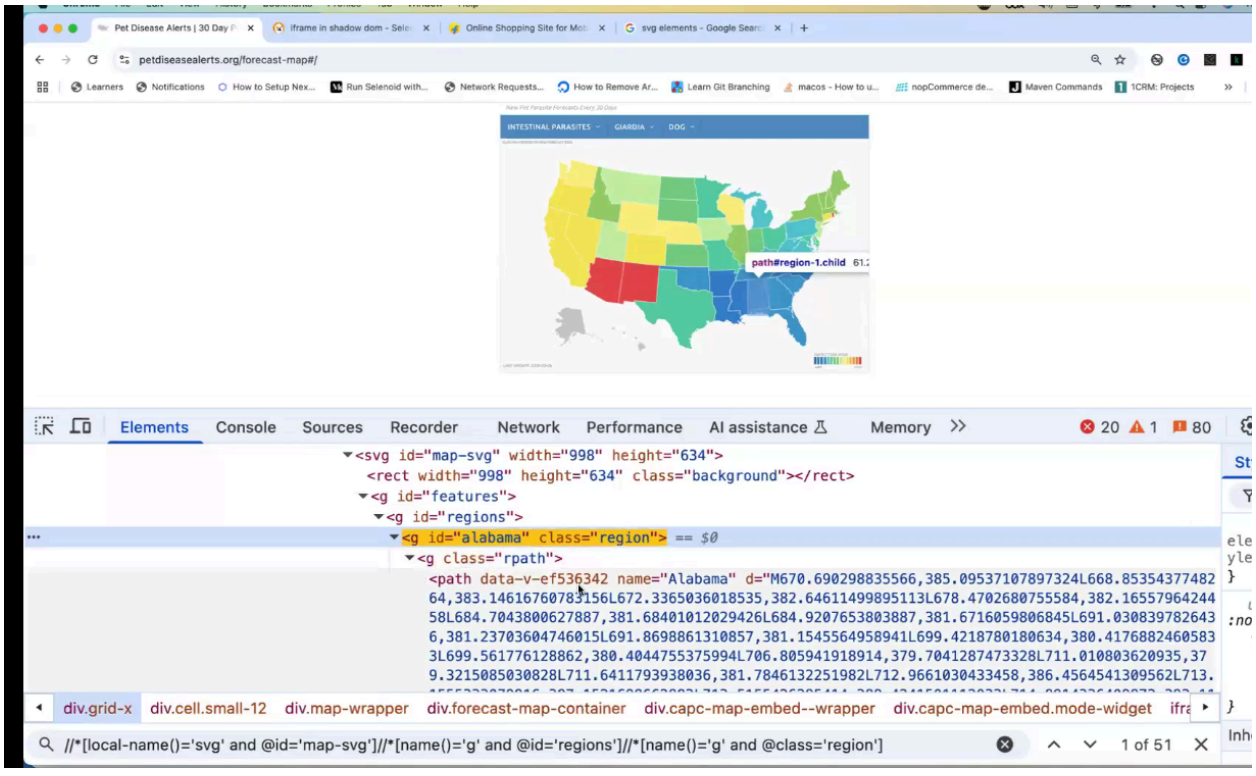


Capture all region names.

First level depth.



Another level deep. We want all the region names.



We can use local name or just name for svg.

```java
package seleniumsessions;

import java.util.List;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class SVGElementHandling {

    static WebDriver driver;

    public static void main(String[] args) throws InterruptedException {

        //*[local-name()='svg' and @fill='none'] --flipkart search icon

        driver = new ChromeDriver();
//      driver.get("https://www.flipkart.com/");
//      Thread.sleep(3000);
//
//      driver.findElement(By.name("q")).sendKeys("macbook");
//      driver.findElement(By.xpath("//*[local-name()='svg' and @fill='none']")).click();

        driver.get("https://petdiseasealerts.org/forecast-map");
        Thread.sleep(3000);

        String xpath = "//*[local-name()='svg' and @id='map-svg']//*[name()='g' and @id='regions']//*[name()='g' and @class='region']";

        driver.switchTo().frame(driver.findElement(By.xpath("//iframe[contains(@id,'map-instance')]")));
        Thread.sleep(3000);

        List<WebElement> regionsList = driver.findElements(By.xpath(xpath));

        System.out.println(regionsList.size());

    }
}
```

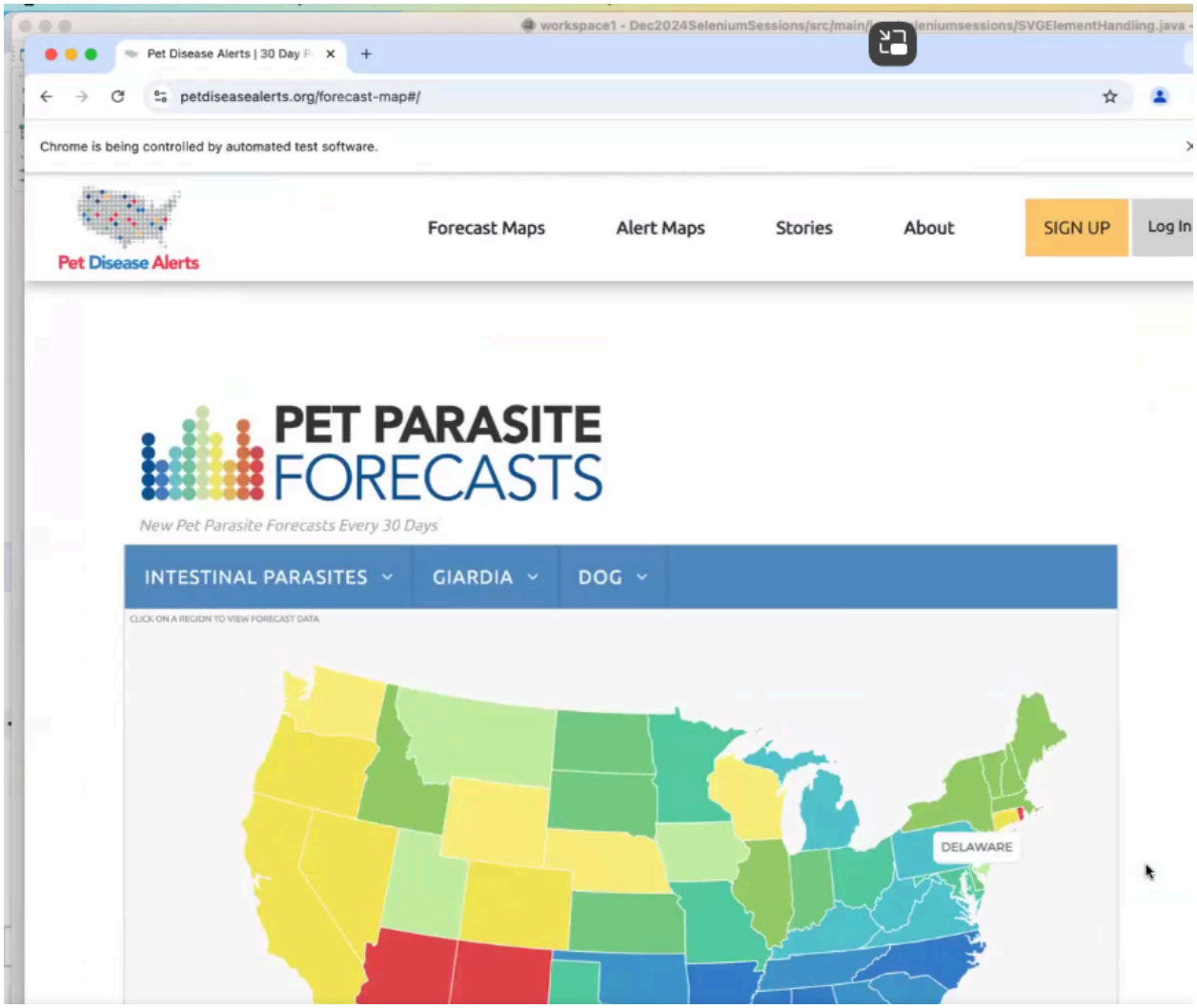
```
Console  Probl
<terminated> SVGElementH
Apr 03, 2025
WARNING: Unab
51
```

Print the ids for every svg-

```java
        for(WebElement e : regionsList) {
            String regionName = e.getDomAttribute("id");
            System.out.println(regionName);
        }
```

Mouse hover every region-

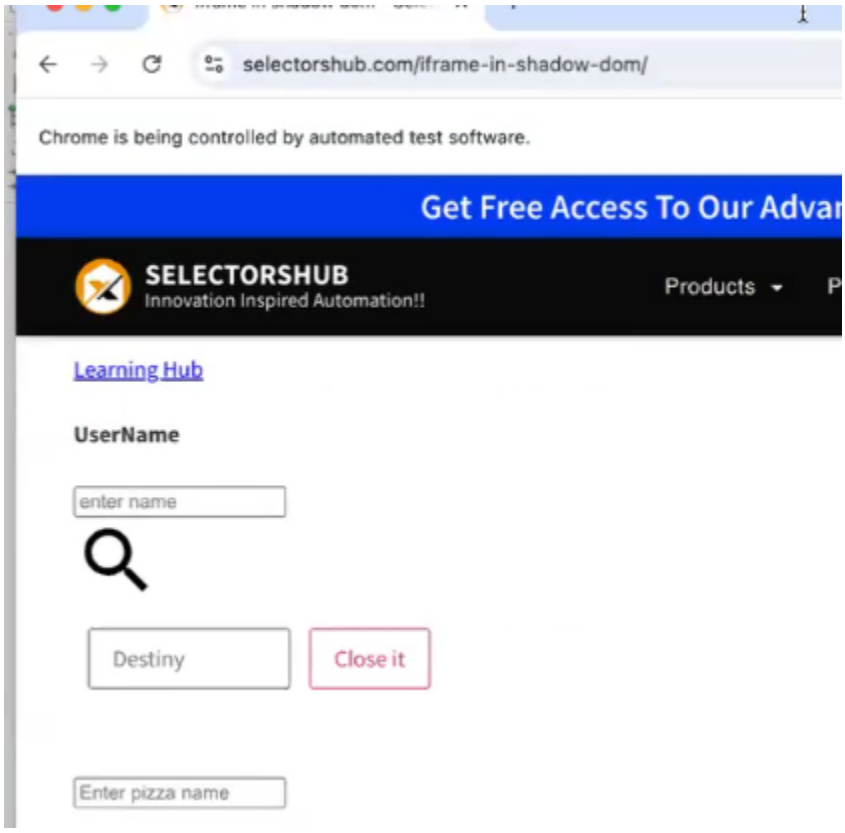
```
37
38          Actions act = new Actions(driver);
39          for(WebElement e : regionsList) {
40              String regionName = e.getDomAttribute("id");
41              System.out.println(regionName);
42              act.moveToElement(e).perform();
43              Thread.sleep(500);
44          }
45
```



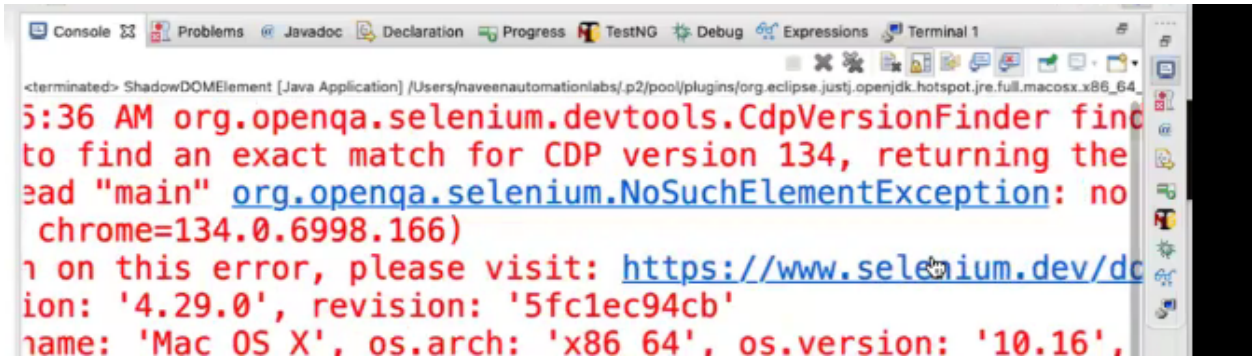
Shadow dom-
Many elements will be present inside the shadow for security purpose. Like iframe.
Selectors hub website.
Enter pizza.

```java
package seleniumsessions;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ShadowDOMElement {

    public static void main(String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.get("https://selectorshub.com/iframe-in-shadow-dom/");
        Thread.sleep(3000);

        driver.findElement(By.id("pizza")).sendKeys("veg pizza");

    }

}
```

Not entering any value.

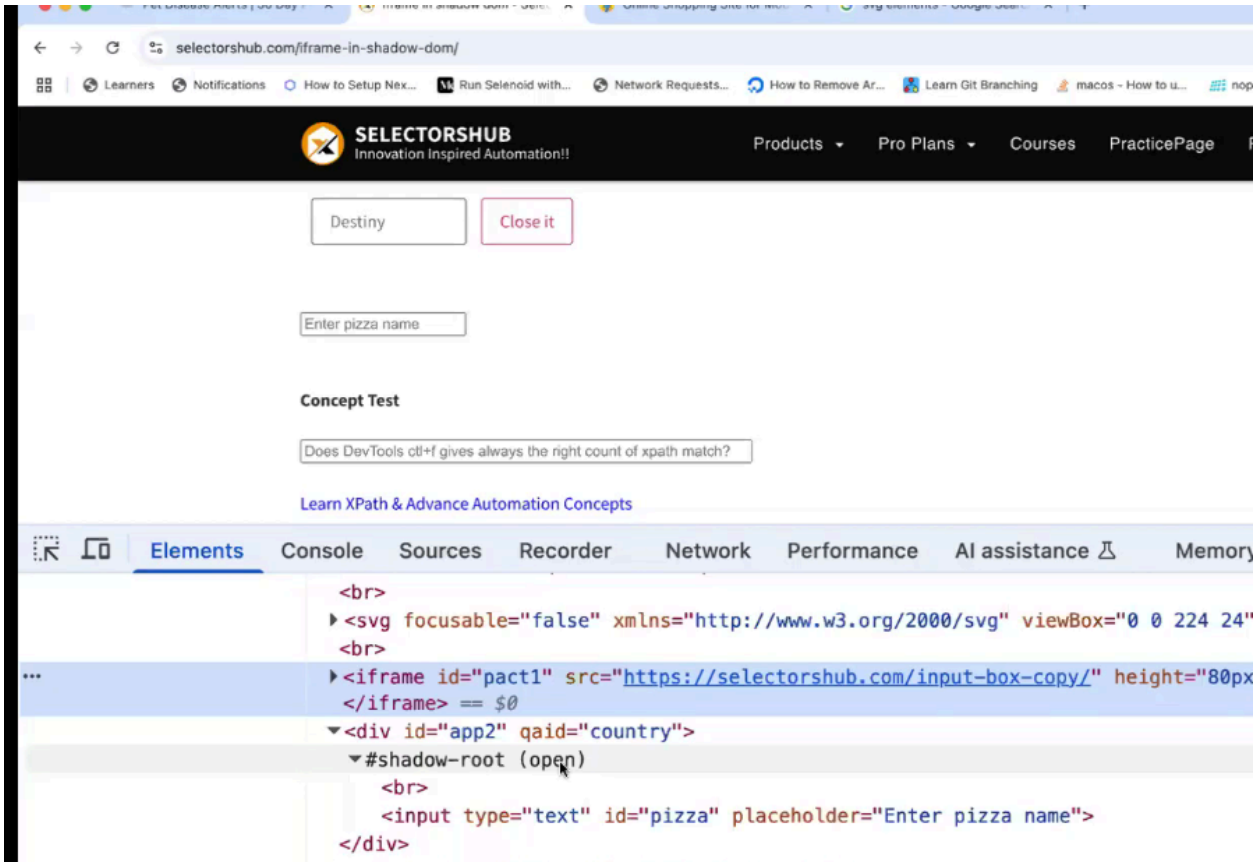


Exception.



Note-

When shadow root is coming with closed word then cannot automate with any tool.

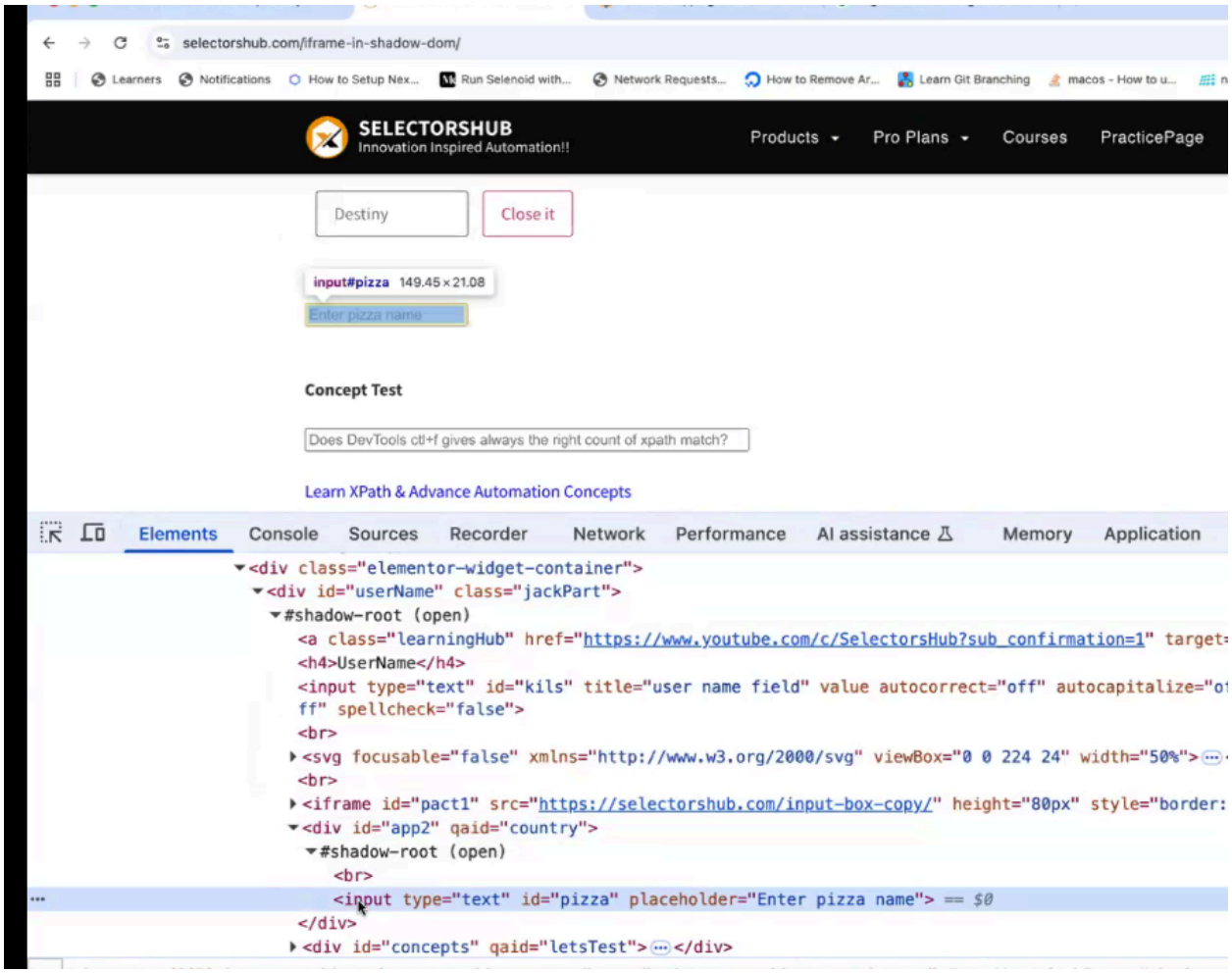


How to get-

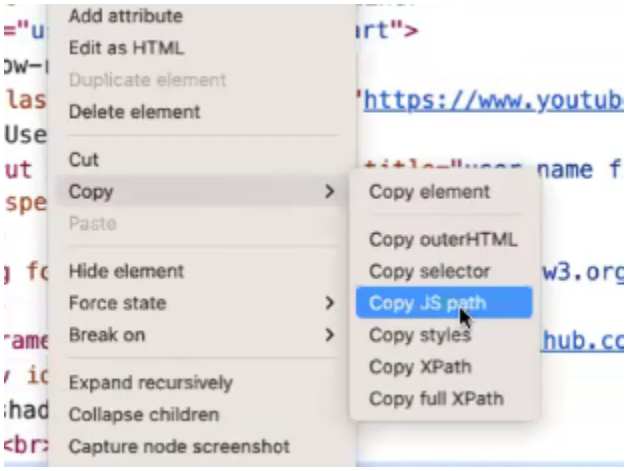
Trick.

Right click on the element you want to interact with.

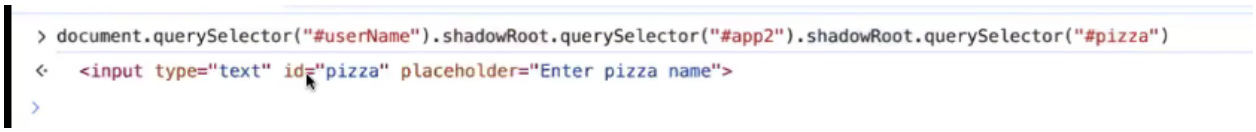
In our case with the pizza name.



Copy js path.



Paste in console.



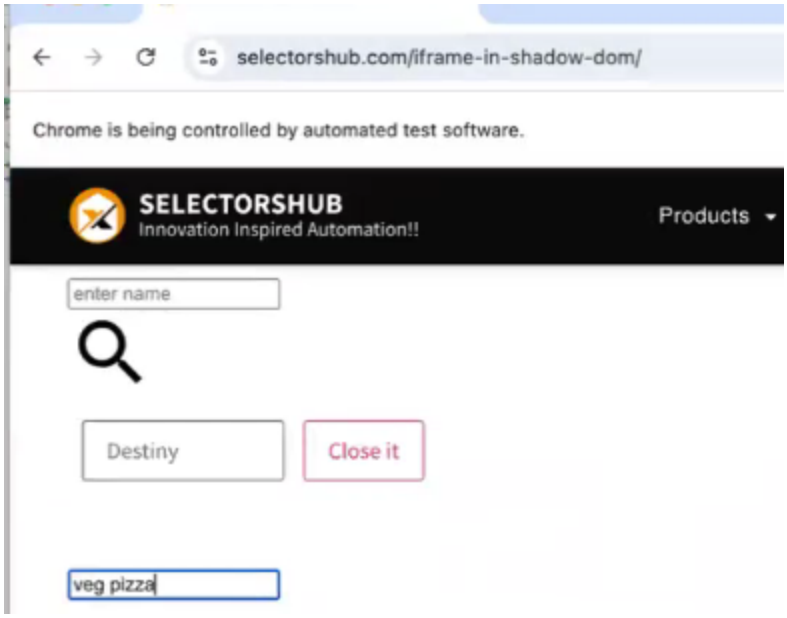
Document.query selector means css selector.
First id locator is username. Then we say that it is a shadow root. Then again query selector(using css locator) for second id which is app2. Then we say its a shadow root. Then again query selector for third element.
Returns complete html element.
Awesome code ready-

```
9
10    public static void main(String[] args) throws InterruptedException {
11        WebDriver driver = new ChromeDriver();
12        driver.get("https://selectorshub.com/iframe-in-shadow-dom/");
13        Thread.sleep(3000);
14
15        String script = "return document.querySelector(\"#userName\").shadowRoot.querySelector(\"#app2\").shadowRoot.querySelector(\"#piz
16
17        JavascriptExecutor js = (JavascriptExecutor)driver;
18
19
20        WebElement pizza = (WebElement)js.executeScript(script);//html element ---> webelement
21
22        pizza.sendKeys("veg pizza");
23
24        //driver.findElement(By.id("pizza")).sendKeys("veg pizza");
```

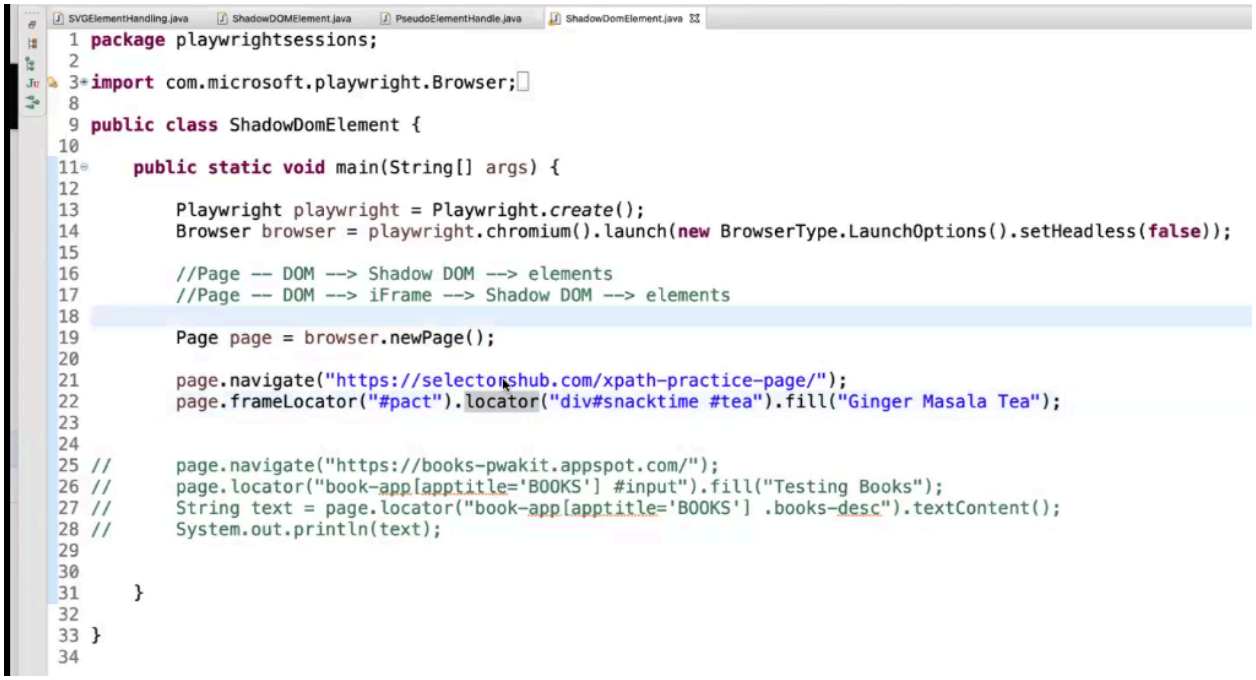
Line 15 clearer version-

String script = "return document.querySelector(\"#userName\").shadowRoot.querySelector(\"#app2\").shadowRoot.querySelector(\"#pizza\")";

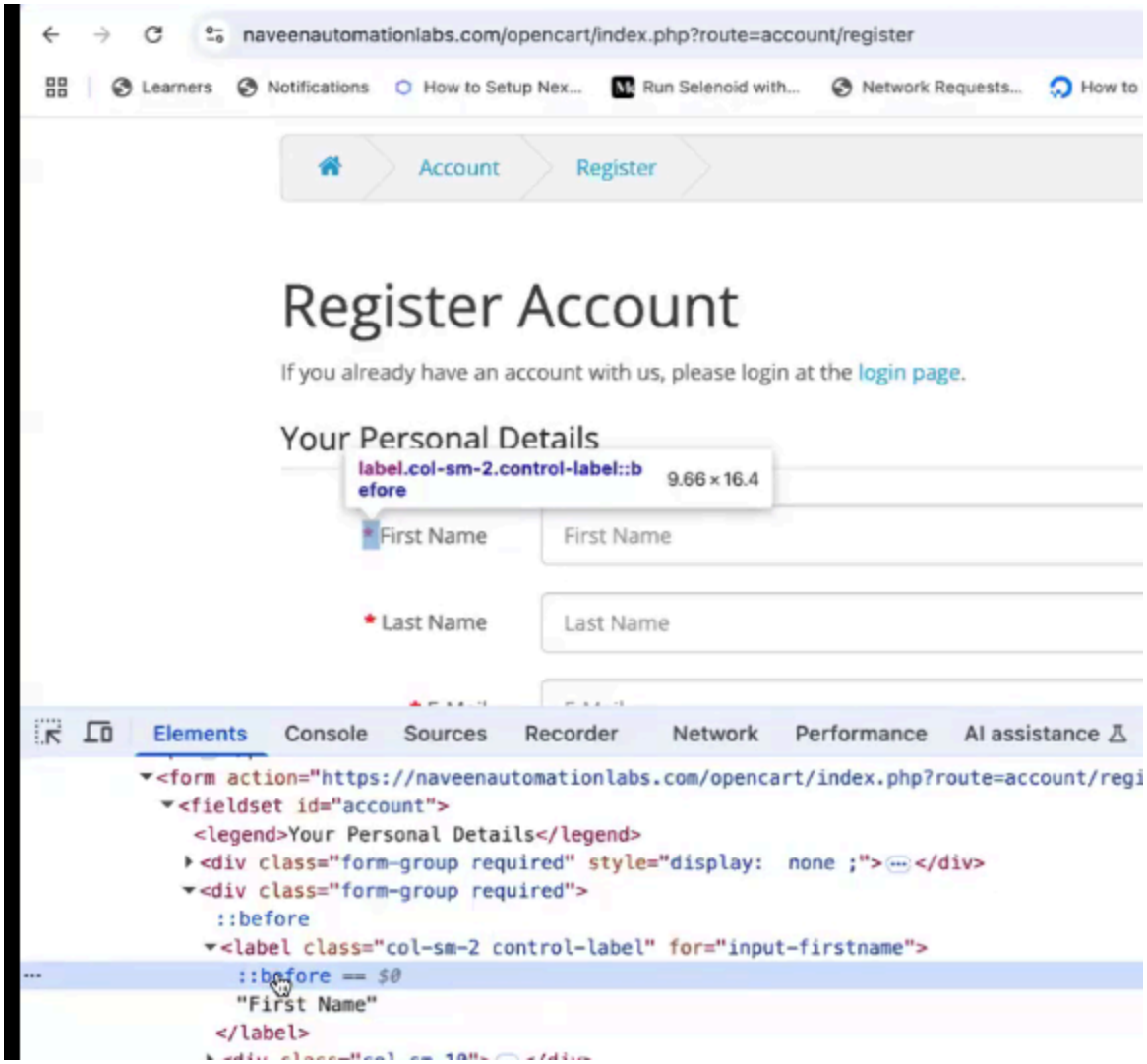
Write return statement and we get the html. To convert html to web element just type cast it to web element.



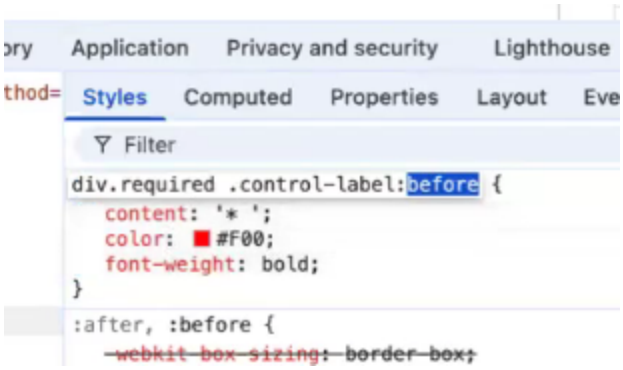
If application is full of shadow dom or loads of them then dont use selenium. Playwright awesome automatically pierces shadow dom and works with it.
Playwright code with shadow dom-
Same selectors hub website.
Passing in masala tea.



Pseudo elements-

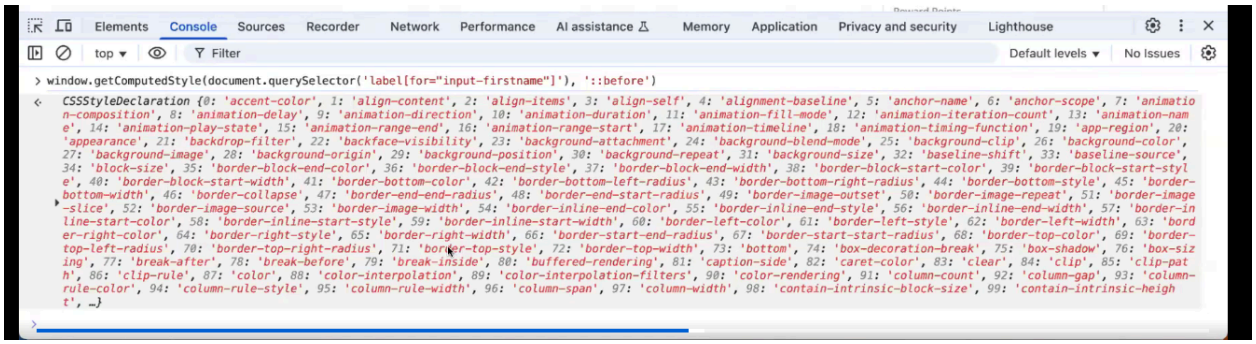


All the asterisk, like this symbols etc.
No html attribute, property, tag etc.
Also called pseudo class. 40.00
Inspect the element and check styles section for pseudo.



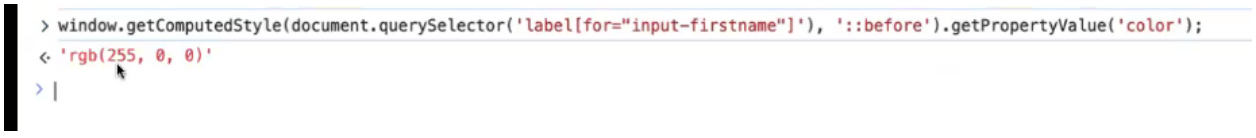
Cant use normal locators to reach pseudo. 41.00

The only way to reach pseudo is like this-

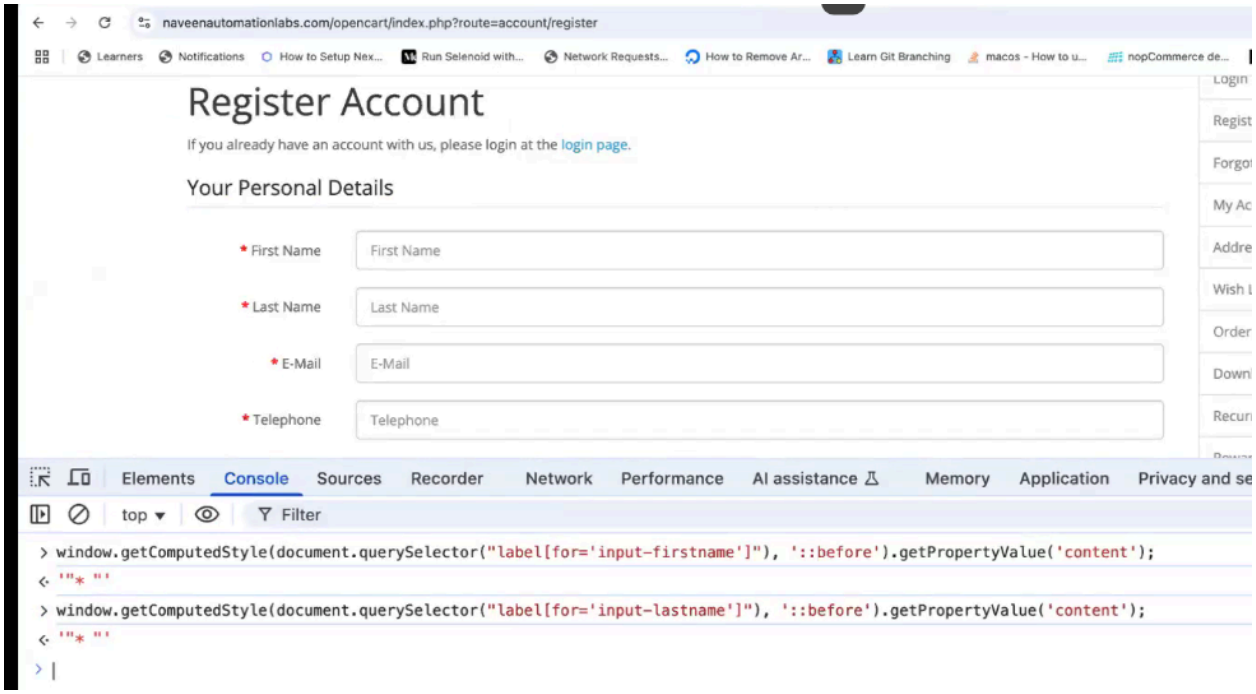


To get only the star icon-

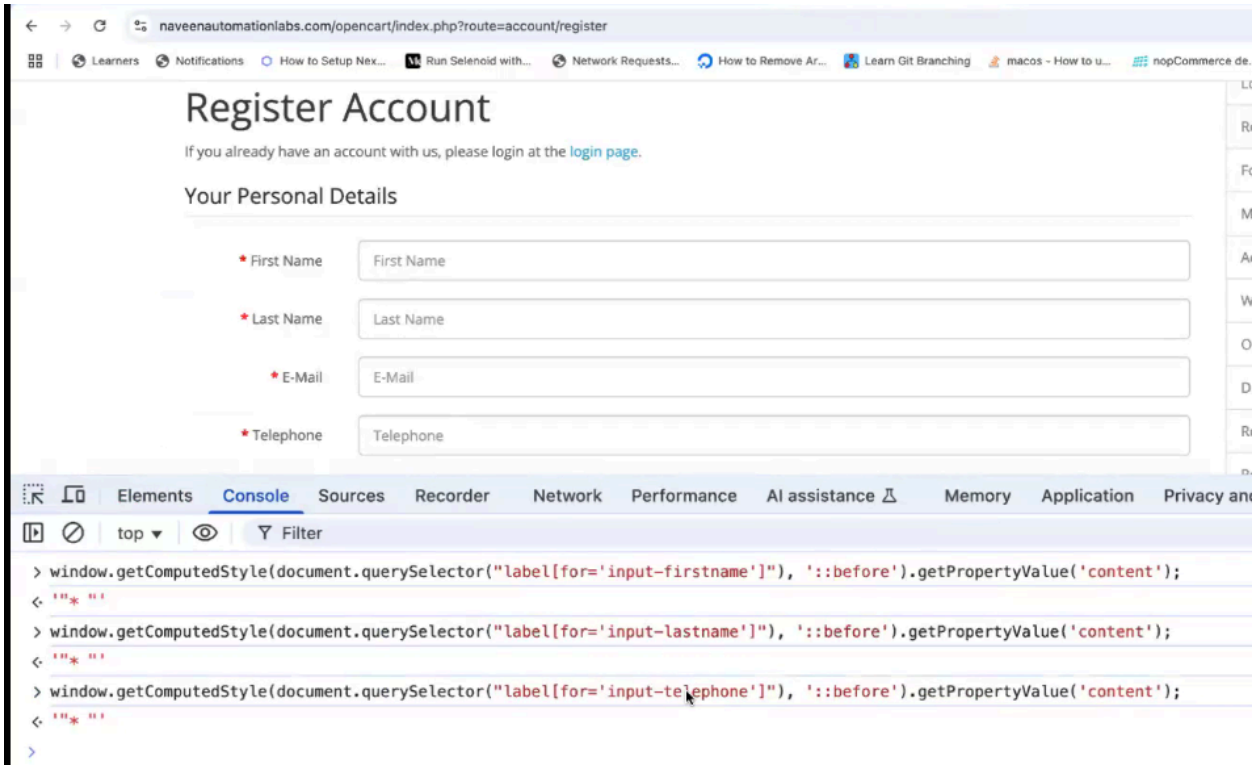
To get color property-



Get content style for last name field-



Get Star for telephone field-

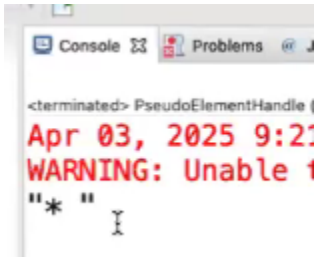


Return keyword for js script needed when it returns something. Like here we get the star. Else not needed. We get js string. Convert to string for java.

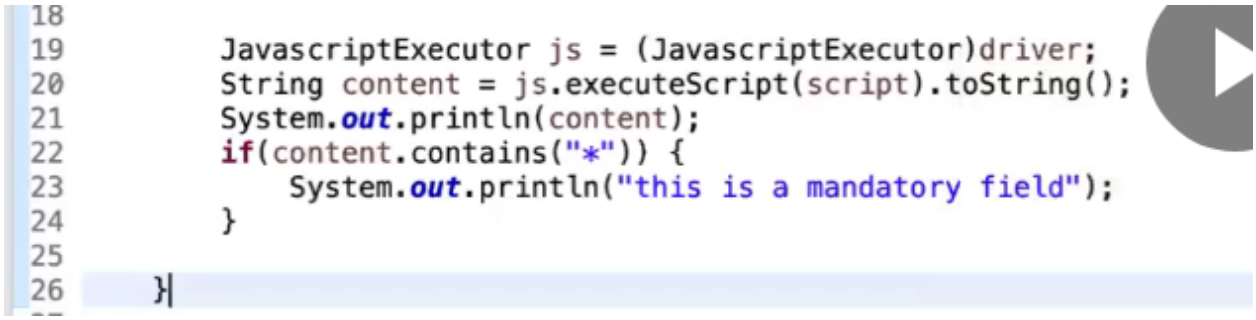
```java
package seleniumsessions;

import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class PseudoElementHandle {

    public static void main(String[] args) throws InterruptedException {
        // TODO Auto-generated method stub


        WebDriver driver = new ChromeDriver();
        driver.get("https://naveenautomationlabs.com/opencart/index.php?route=account/register");
        Thread.sleep(3000);

        String script = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), '::before').getProperty

        JavascriptExecutor js = (JavascriptExecutor)driver;
        String content = js.executeScript(script).toString();
        System.out.println(content);

    }



}
```

Clearer line 17-

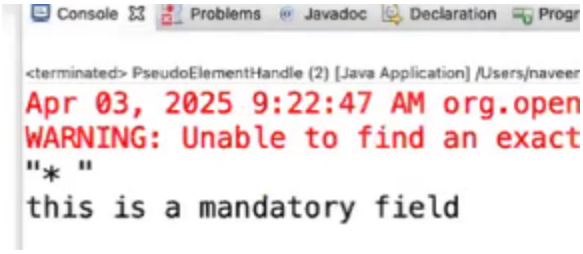
String script = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), '::before').getPropertyValue('content');";



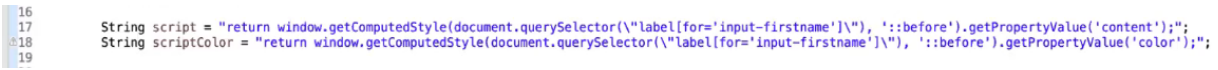
Add if statement-



```java
        JavascriptExecutor js = (JavascriptExecutor)driver;
        String content = js.executeScript(script).toString();
        System.out.println(content);
        if(content.contains("*")) {
            System.out.println("this is a mandatory field");
        }

    }
```

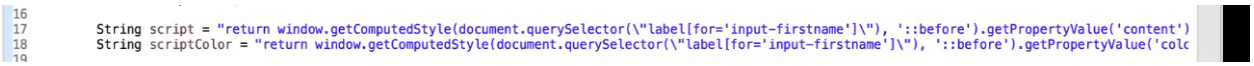


Clearer script for line 17 and 18-



String script = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), '::before').getPropertyValue('content');";
String scriptColor = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), '::before').getPropertyValue('color');";

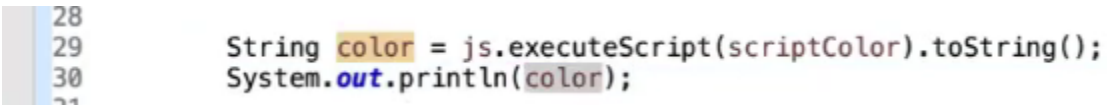
Line 18 added for color check-

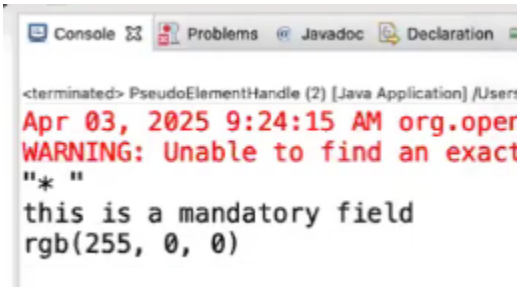


Clearer picture-

String scriptColor = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), '::before').getPropertyValue('color');";

Check color displayed-

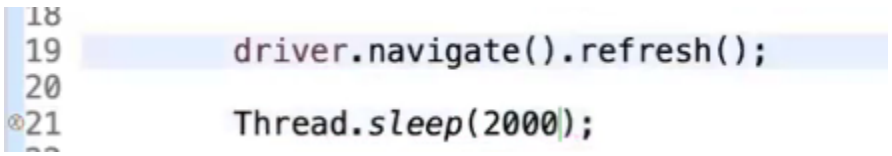


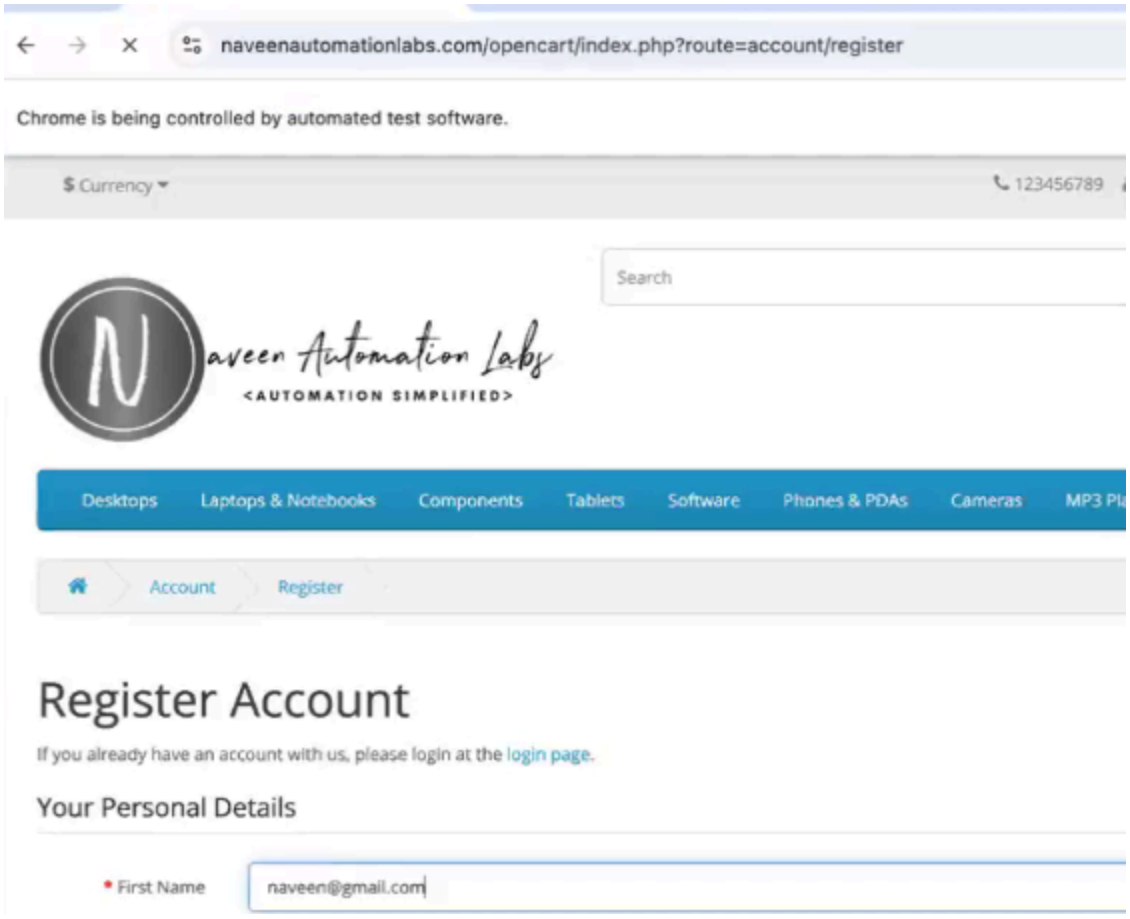


Anything starting with :: is pseudo class.

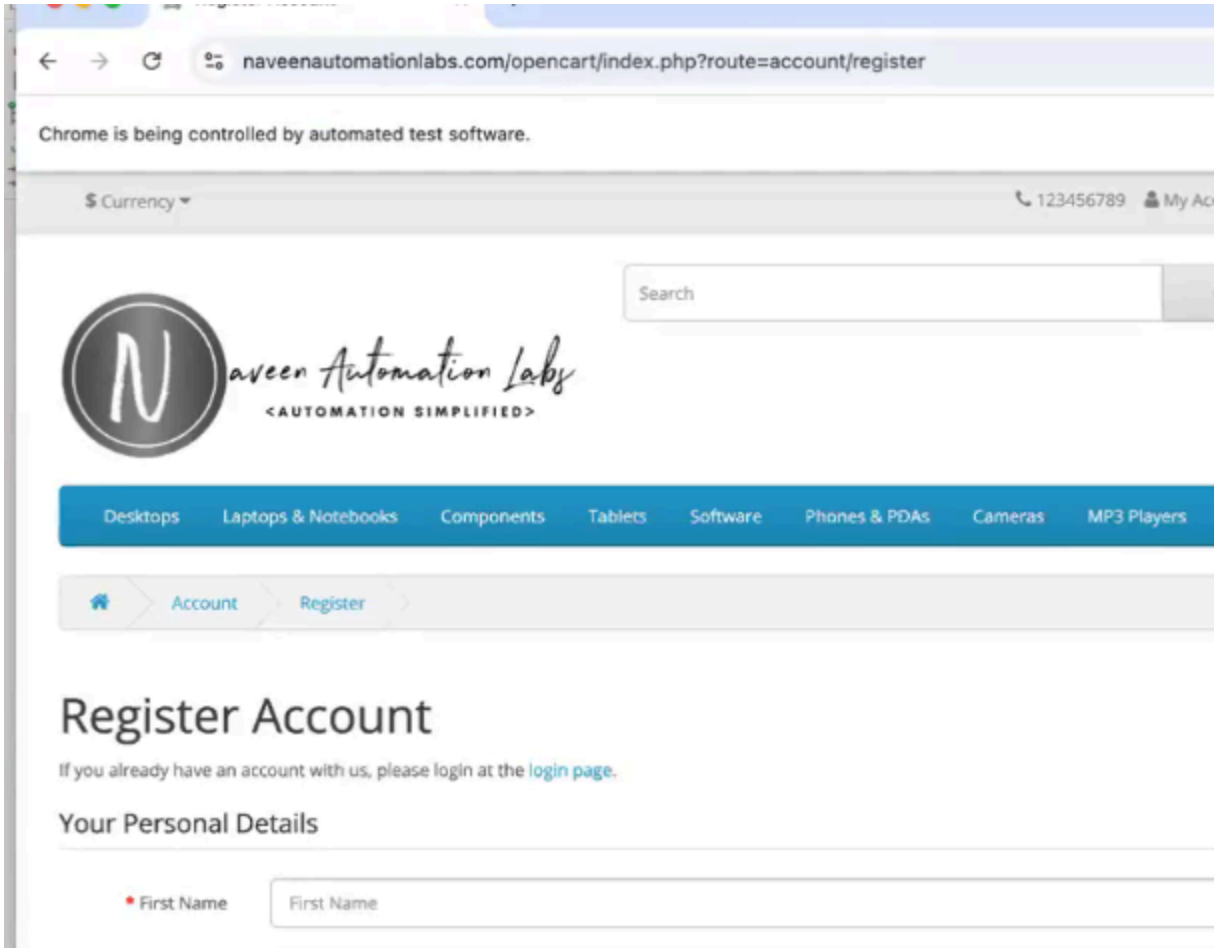
Pseudo cant be handled by pw, cypress and python.
54.00
Only js executor can handle it.
Stale element exception-
Naveen website.
Register page.
Firstname field.
Refresh page.
Again enter name new name.

Added 21-



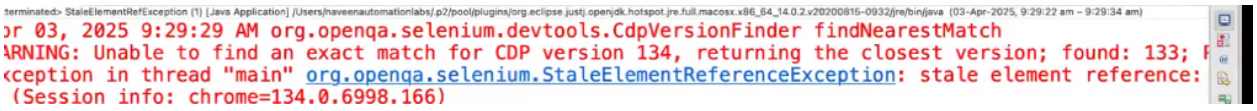


Then refreshed.



Stale element exception.

New value not added.



When you hit refresh, dom and page both updated.

Since after refresh dom refreshed so latest dom not picked up for the same web element.

Explanation for stale element exception-

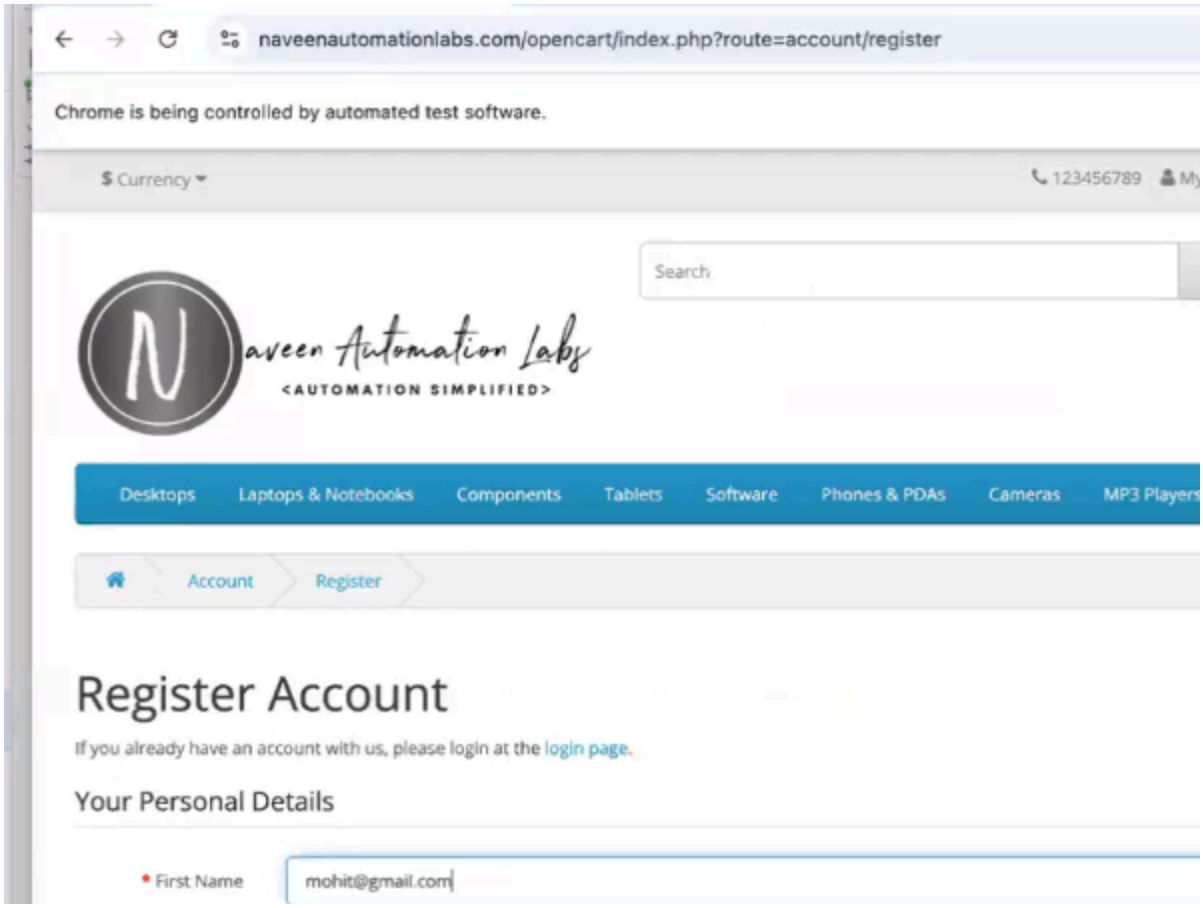
Dom 1 and dom 2 after refresh.

```
12        WebDriver driver = new ChromeDriver();//browser
13        driver.get("https://naveenautomationlabs.com/opencart/index.php?route=account/register");//page DOM-v1
14
15        //DOM v1
16        WebElement firstname = driver.findElement(By.id("input-firstname"));
17
18        //DOM v1
19        firstname.sendKeys("naveen@gmail.com");
20
21        driver.navigate().refresh();
22
23        //DOM v2
24
25        //Dom v1
26        firstname.sendKeys("mohit@gmail.com");
27
28
```

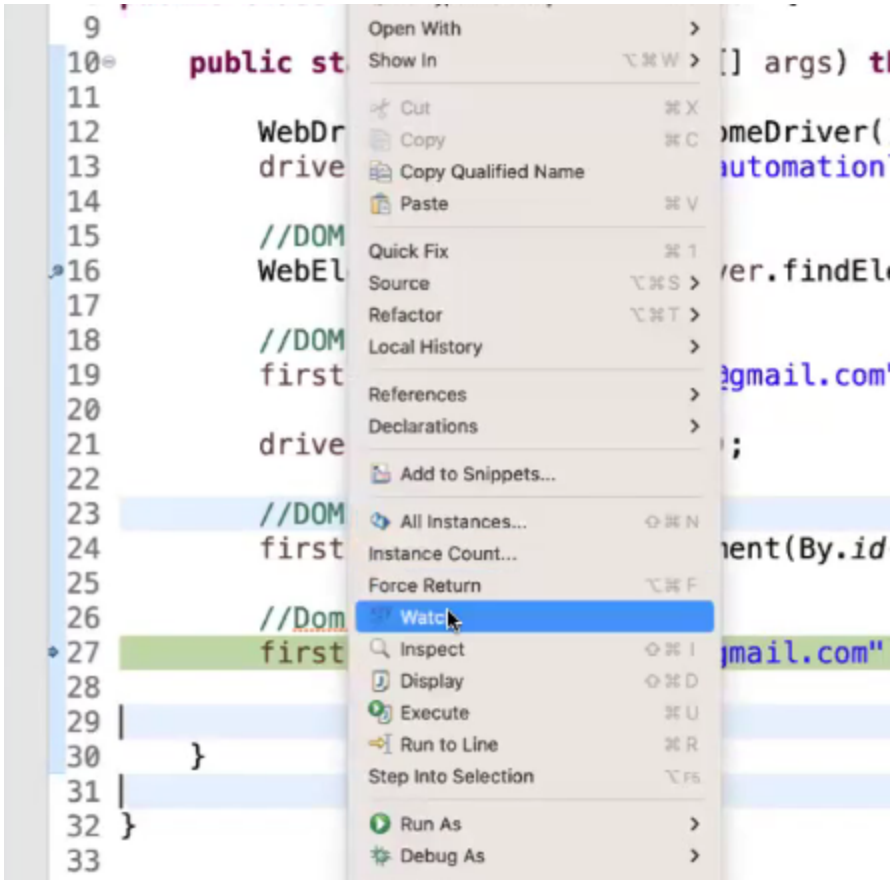
Add line 24 and issue resolved.

We need to recreate the web element.

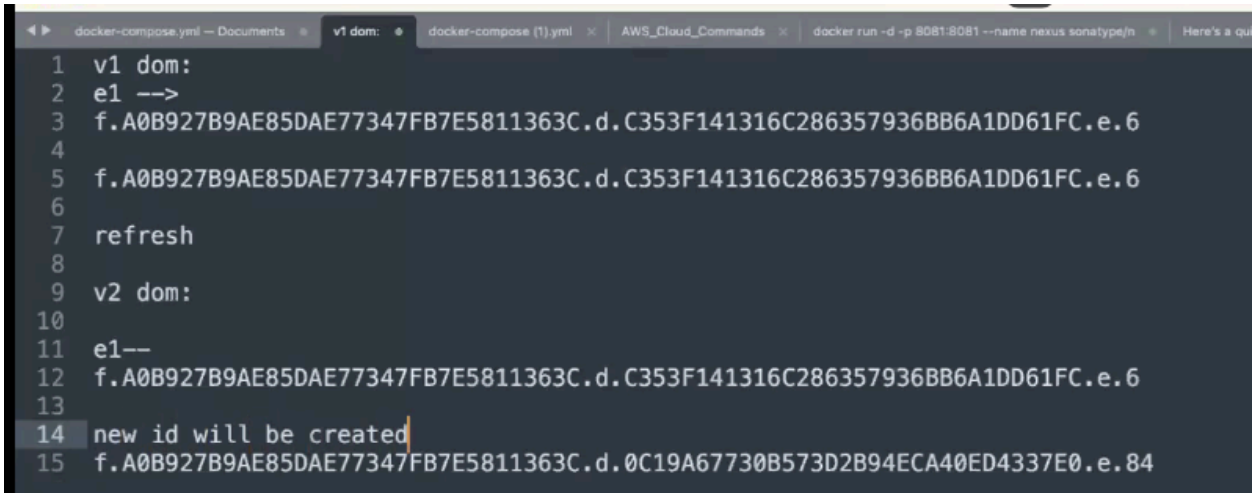
```
12        WebDriver driver = new ChromeDriver();//browser
13        driver.get("https://naveenautomationlabs.com/opencart/index.php?route=account/register");//page DOM-v1
14
15        //DOM v1
16        WebElement firstname = driver.findElement(By.id("input-firstname"));
17
18        //DOM v1
19        firstname.sendKeys("naveen@gmail.com");
20
21        driver.navigate().refresh();
22
23        //DOM v2
24        firstname = driver.findElement(By.id("input-firstname"));
25
26        //Dom v2
27        firstname.sendKeys("mohit@gmail.com");
28
```



Refresh is dangerous in automation.
Thats why use by locator and then pass to method, rather than having web element directly like this.
1.02. Every time we get the element like a fresh one and then pass the values.
Example our pom.
To watch variable-
Add debug point somewhere.
Right click variable name and watch.



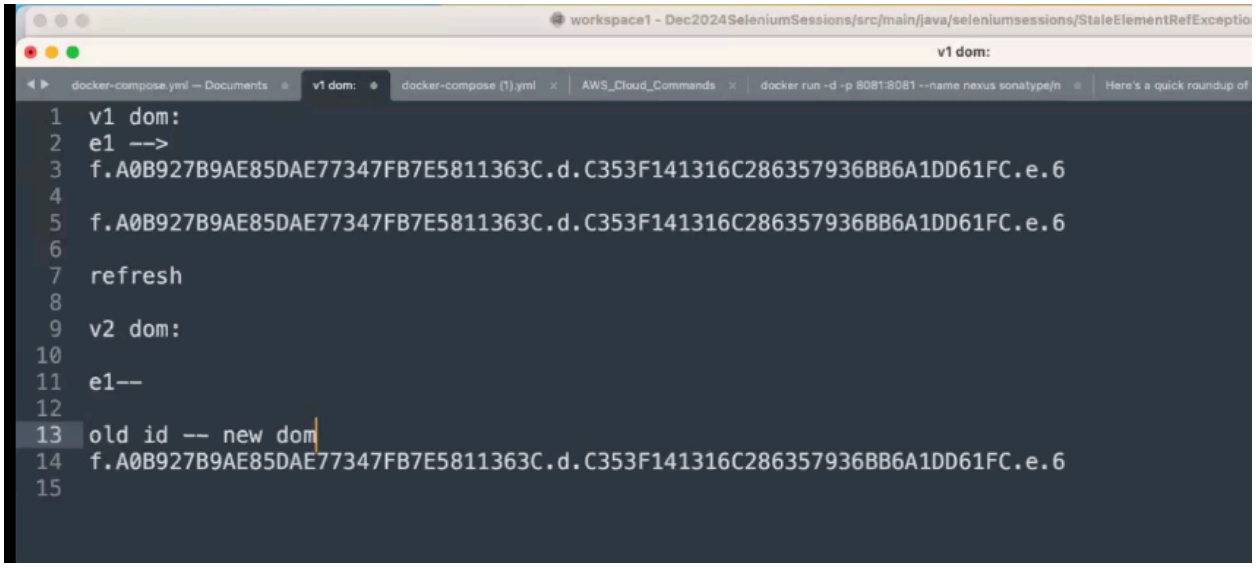
Internal working of the dom elements and versions- Send keys performed on new web element id. Webelement id internally gets changed on every refresh.



When we dont reinitialise web driver again this is how it works-

```
        WebDriver driver = new ChromeDriver();//browser
        driver.get("https://naveenautomationlabs.com/opencart/index.php?route=account/register");//page DOM-v1

        //DOM v1
        WebElement firstname = driver.findElement(By.id("input-firstname"));

        //DOM v1
        firstname.sendKeys("naveen@gmail.com");

        driver.navigate().refresh();

        //DOM v2
        //firstname = driver.findElement(By.id("input-firstname"));

        //Dom v2
        firstname.sendKeys("mohit@gmail.com");
```

Its still the old web element id after refresh and we are not able to enter the values.



Stale element comes when refresh, when hitting back and forward.1.13