

Svg -  
Scalar vector graphs.  
Some different shapes like the below will be seen.



Example flipkart search icon-

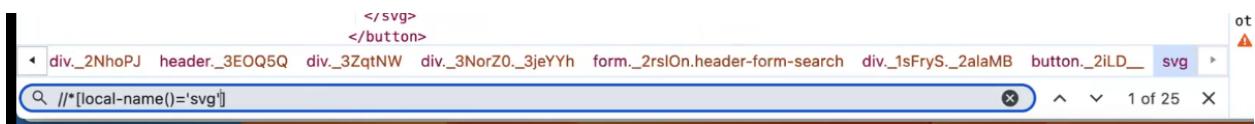
The screenshot shows the Flipkart homepage with a search bar at the top. Below the search bar, there are several category icons: Kilos, Mobiles, Fashion, Electronics, Home & Furniture, Appliances, Flight Bookings, Beauty, Toys & More, and Two Wheelers. A promotional banner for 'BIG BACHAT DAYS' is visible on the right. The developer tools (Elements tab) are open, showing the HTML structure of the search bar. The search button is represented by an SVG element with a width and height of 24px, a viewBox of 0 0 24 24, and a fill of none. The SVG path uses a complex d attribute to define its shape. The code also includes a hidden input field named 'otracker' with the value 'search'.

```
<button class="21LD_" type="submit" aria-label="Search for Products, Brands and More" title="Search for Products, Brands and More">
<svg width="24" height="24" class viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg">
<title>Search Icon</title>
<path d="M10.5 18C14.6421 18 18 14.6421 18 10.5C18 6.35786 14.6421 3 10.5 3C6.35786 3 3 6.35786 3 10.5C3 14.6421 6.35786 18 10.5 18Z" stroke="#717478" stroke-width="1.4" stroke-linecap="round" stroke-linejoin="round"></path> == $0
<path d="M16 16L21 21" stroke="#717478" stroke-width="1.4" stroke-linecap="round" stroke-linejoin="round"></path>
</svg>
</button>
<div class="2SmNnR"></div>
<input type="hidden" name="otracker" value="search">
```

Normal locators wont work with svg.



## Special way to get svg-



Then add as many attributes as possible.



```
SVGElementHandling.java  ShadowDOMElement.java  PseudoElementHandle.java
1 package seleniumsessions;
2
3 import org.openqa.selenium.By;
4 import org.openqa.selenium.WebDriver;
5 import org.openqa.selenium.chrome.ChromeDriver;
6
7 public class SVGElementHandling {
8
9     static WebDriver driver;
10
11    public static void main(String[] args) throws InterruptedException {
12
13        //*[local-name()='svg' and @fill='none'] --flipkart search icon
14
15        driver = new ChromeDriver();
16        //driver.get("https://petdiseasealerts.org/forecast-map");
17        driver.get("https://www.flipkart.com/");
18        Thread.sleep(3000);
19
20        driver.findElement(By.name("q")).sendKeys("macbook");
21        driver.findElement(By.xpath("//*[local-name()='svg' and @fill='none']")).click();
22    }
23}
```

Macbook- Buy Products Online

flipkart.com/search?q=macbook&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=off&as-off

Chrome is being controlled by automated test software.

Flipkart Explore Plus

macbook

Login Become a Seller More

Electronics TVs & Appliances Men Women Baby & Kids Home & Furniture Sports, Books & More

Filters

CATEGORIES

< Computers  
Laptops

PRICE

Min to ₹75000+

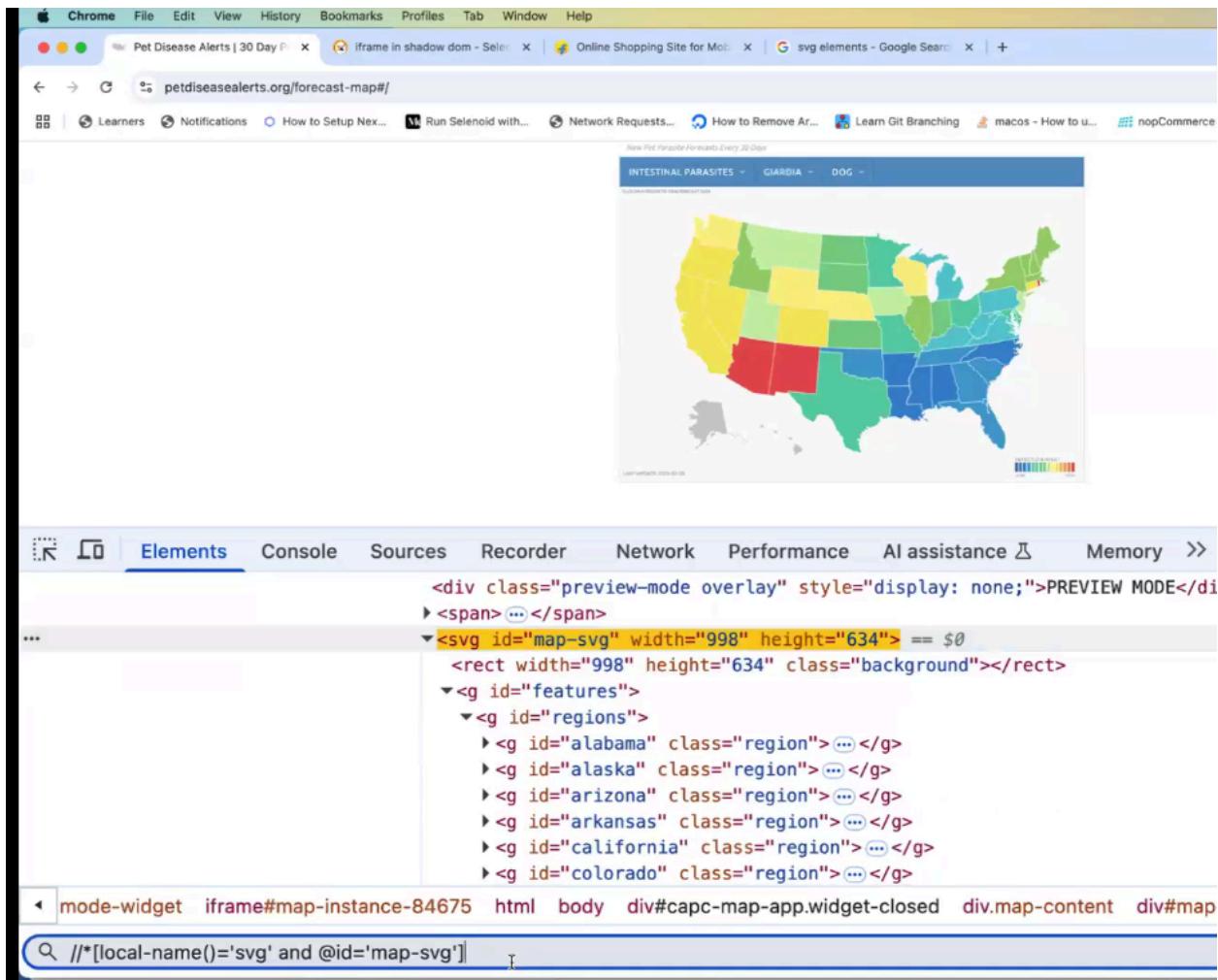
Home > Computers > Laptops

Showing 1–24 of 83 results for "macbook"

Sort By Relevance Popularity Price -- Low to High Price -- High to Low Newest First

 Apple 2020 Macbook Air Apple M1 - (8 GB/256 GB SSD/Mac OS Big Sur) MGND3HN/A ₹  
4.7 ★ 17,305 Ratings & 1,355 Reviews  
· Apple M1 Processor  
· 8 GB DDR4 RAM  
· Mac OS Operating System  
· 256 GB SSD

Click california-  
All are svg.



Normal xpaths or any locators wont work.



The screenshot shows a web page with a map of the United States. The map is divided into states, each colored differently based on a gradient scale. A legend at the bottom right of the map shows the color mapping for the gradient. The map is titled "INTESTINAL PARASITES" and has tabs for "GIARDIA" and "DOG".

The browser's developer tools are open, specifically the Elements tab. The DOM tree is displayed, showing the structure of the map. The search bar at the bottom of the developer tools contains the XPath expression `//*[@local-name()='svg' and @id='map-svg']//g[@id='regions']`.

```
<span>...</span>
<svg id="map-svg" width="998" height="634">
  <rect width="998" height="634" class="background"></rect>
  <g id="features">
    <g id="regions"> == $0
      <g id="alabama" class="region">...</g>
      <g id="alaska" class="region">...</g>
      <g id="arizona" class="region">...</g>
      <g id="arkansas" class="region">...</g>
      <g id="california" class="region">...</g>
      <g id="colorado" class="region">...</g>
      <g id="connecticut" class="region">...</g>
      ...
```

Capture all region names.

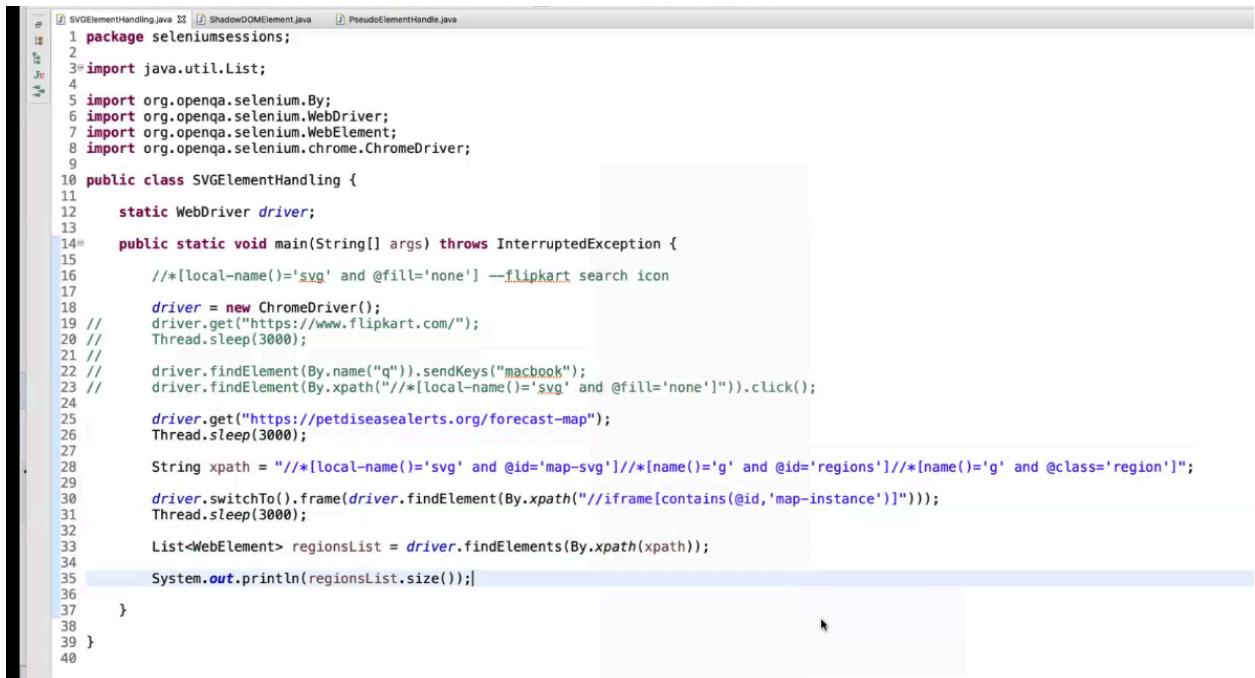
First level depth.

The screenshot shows a web page with a map of the United States. The map is color-coded by state, representing different regions. The browser's developer tools are open, and the Elements tab is selected. The code in the Elements tab shows the structure of the SVG map, including a root `<svg>` element with an `id="map-svg"`, a `<rect>` element with a `class="background"`, and a `<g id="regions">` element containing multiple `<g id="state-name">` elements for each state. The search bar at the bottom contains the XPath query `//*[local-name()='svg' and @id='map-svg']//*[name()='g' and @id='regions']//*[name()='g']`.

Another level deep. We want all the region names.

The screenshot shows the same web page and developer tools as the previous one, but with a different XPath query in the search bar: `//*[local-name()='svg' and @id='map-svg']//*[name()='g' and @id='regions']//*[name()='g' and @class='region']`. This query targets the `<g class="region">` element within the `<g id="regions">` element. A tooltip over one of the state nodes in the map indicates its path as `#region-1.child`. The developer tools also show the detailed SVG path for the Alabama region.

We can use local name or just name for svg.

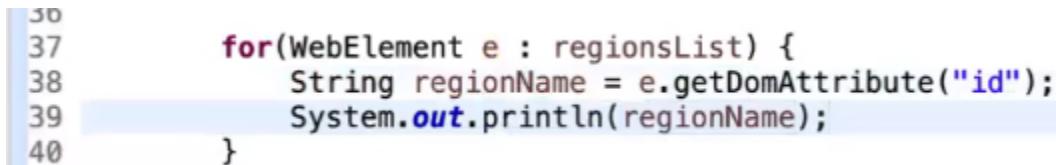


```
1 package seleniumsessions;
2
3 import java.util.List;
4
5 import org.openqa.selenium.By;
6 import org.openqa.selenium.WebDriver;
7 import org.openqa.selenium.WebElement;
8 import org.openqa.selenium.chrome.ChromeDriver;
9
10 public class SVGElementHandling {
11
12     static WebDriver driver;
13
14     public static void main(String[] args) throws InterruptedException {
15
16         /*[local-name()='svg' and @fill='none'] --flipkart search icon
17
18         driver = new ChromeDriver();
19         driver.get("https://www.flipkart.com/");
20         Thread.sleep(3000);
21
22         driver.findElement(By.name("q")).sendKeys("macbook");
23         driver.findElement(By.xpath("/*[local-name()='svg' and @fill='none']")).click();
24
25         driver.get("https://petdisseasealerts.org/forecast-map");
26         Thread.sleep(3000);
27
28         String xpath = "/*[local-name()='svg' and @id='map-svg']/*[name()='g' and @id='regions']/*[name()='g' and @class='region']";
29
30         driver.switchTo().frame(driver.findElement(By.xpath("//iframe[contains(@id,'map-instance')]")));
31         Thread.sleep(3000);
32
33         List<WebElement> regionsList = driver.findElements(By.xpath(xpath));
34
35         System.out.println(regionsList.size());
36
37     }
38
39 }
40
```



```
<terminated> SVGElementHandling
Apr 03, 2025
WARNING: Unable to start chrome
51
```

Print the ids for every svg-

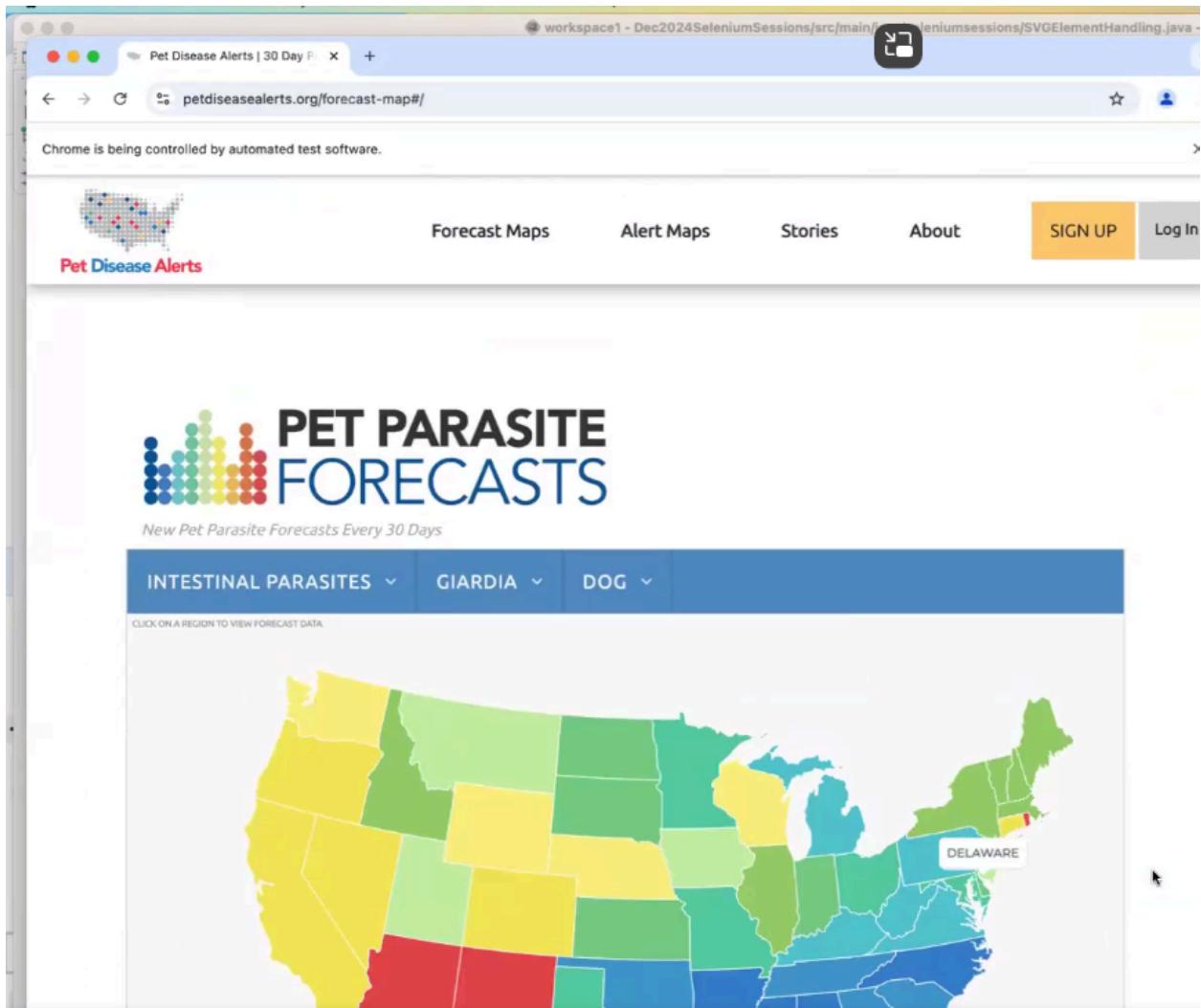


```
30
31
32
33
34
35
36
37     for(WebElement e : regionsList) {
38         String regionName = e.getDomAttribute("id");
39         System.out.println(regionName);
40     }
```

```
/java/seleniumsessions/SVGEElementHandling.java
<terminated> SVGEElementHandling (1) [Java App]
new-mexico
new-york
north-carolina
north-dakota
ohio
oklahoma
oregon
pennsylvania
rhode-island
south-carolina
south-dakota
tennessee
texas
utah
vermont
virginia
washington
west-virginia
wisconsin
wyoming
```

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-

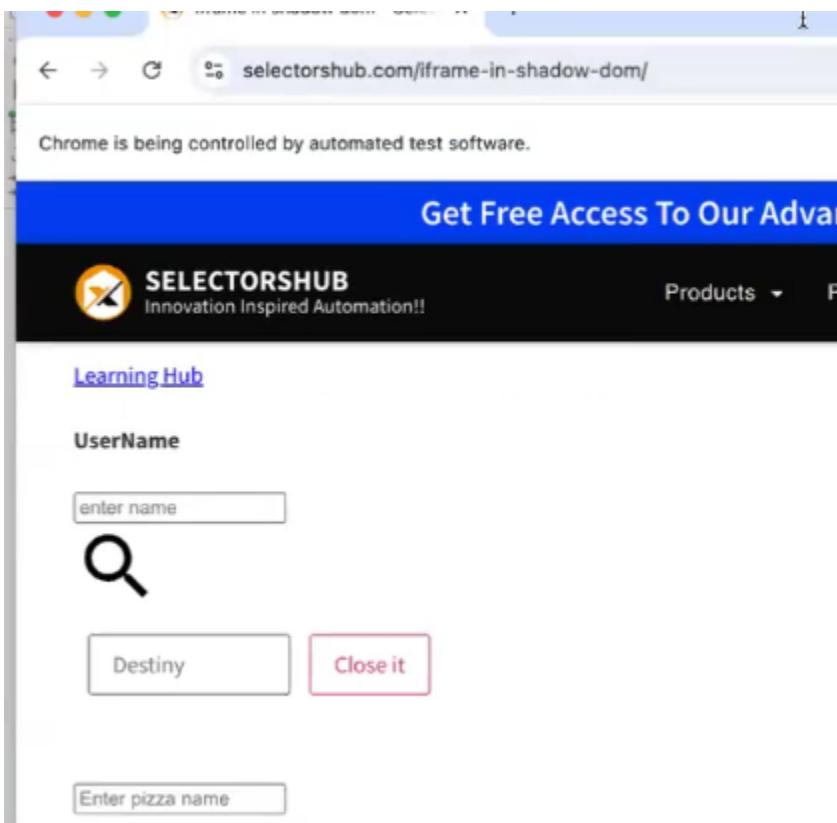
Many elements will be present inside the shadow for security purpose. Like iframe.

Selectors hub website.

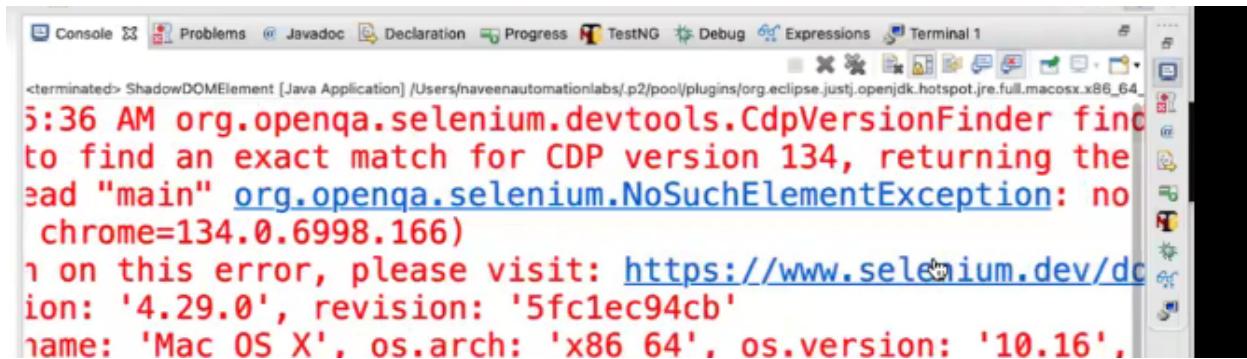
Enter pizza.

```
SVGELEMENTHandling.java  ShadowDOMELEMENT.java  PseudoELEMENTHandle.java
1 package seleniumsessions;
2
3 import org.openqa.selenium.By;
4 import org.openqa.selenium.WebDriver;
5 import org.openqa.selenium.chrome.ChromeDriver;
6
7 public class ShadowDOMELEMENT {
8
9     public static void main(String[] args) throws InterruptedException {
10         WebDriver driver = new ChromeDriver();
11         driver.get("https://selectorshub.com/iframe-in-shadow-dom/");
12         Thread.sleep(3000);
13
14         driver.findElement(By.id("pizza")).sendKeys("veg pizza");
15     }
16
17 }
18
19 }
```

Not entering any value.



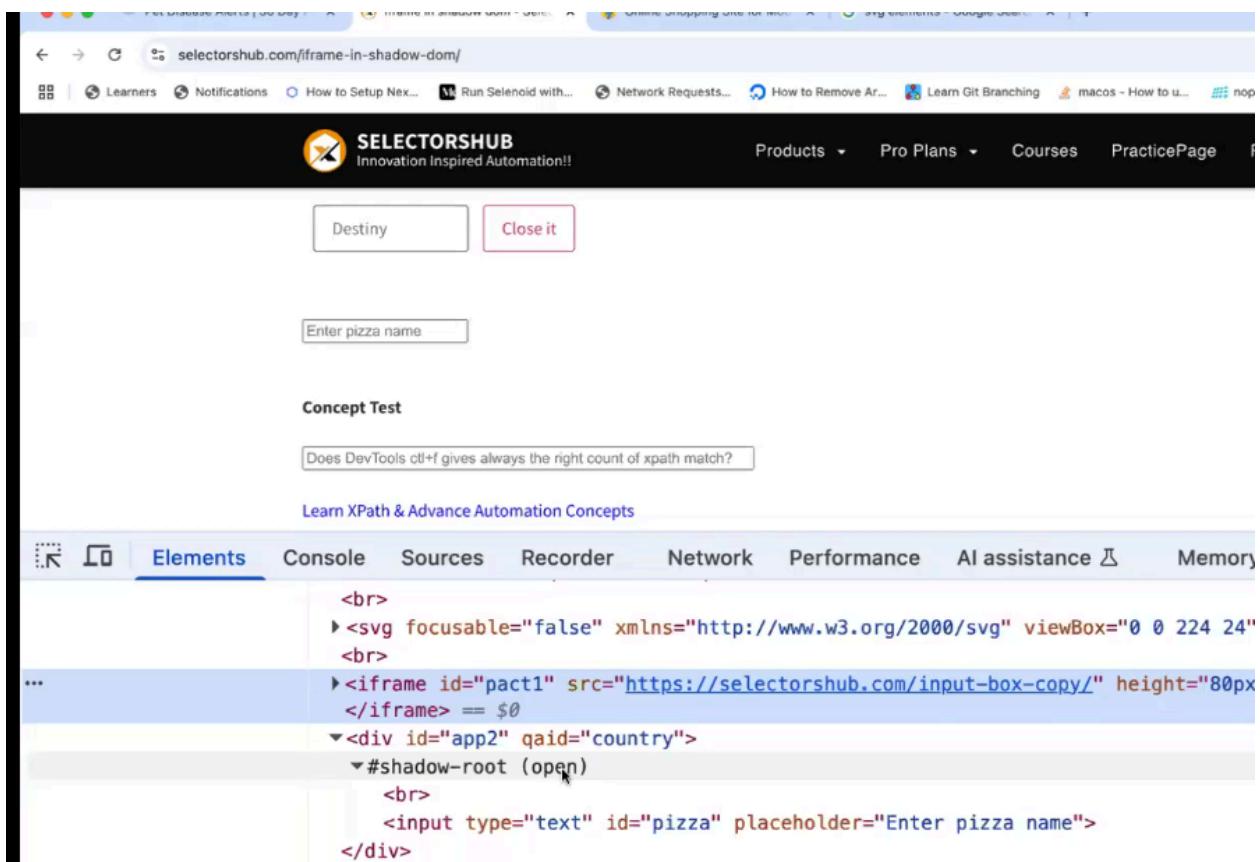
Exception.



```
<terminated> ShadowDOMElement [Java Application] /Users/naveenautomationlabs/p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64
5:36 AM org.openqa.selenium.devtools.CdpVersionFinder find
to find an exact match for CDP version 134, returning the
read "main" org.openqa.selenium.NoSuchElementException: no
such chrome=134.0.6998.166)
on this error, please visit: https://www.selenium.dev/dc
ion: '4.29.0', revision: '5fc1ec94cb'
name: 'Mac OS X', os.arch: 'x86_64', os.version: '10.16'.
```

## Note-

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



## How to get-

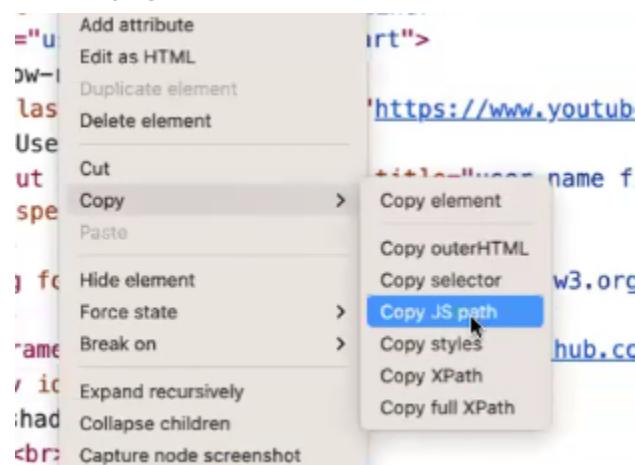
Trick.

Right click on the element you want to interact with.

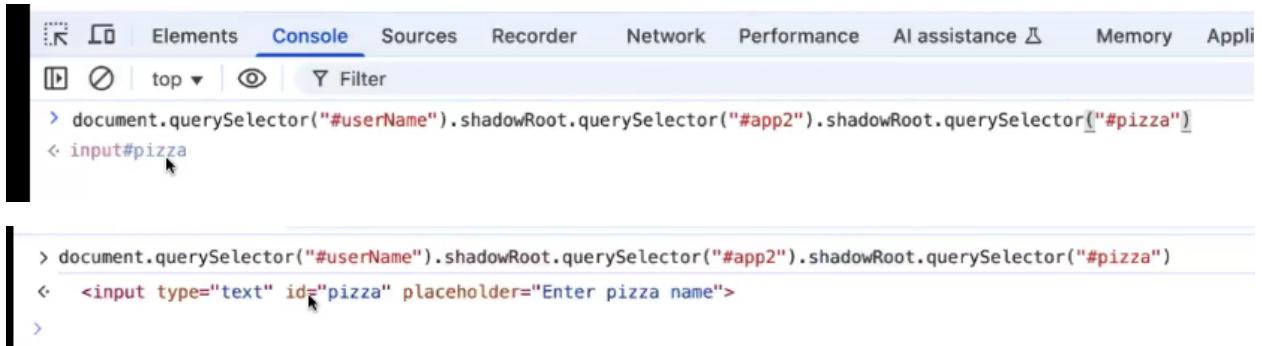
# In our case with the pizza name.

The screenshot shows a browser window with the URL [selectorshub.com/iframe-in-shadow-dom/](https://selectorshub.com/iframe-in-shadow-dom/). The page displays a form with a text input field labeled "Enter pizza name". The browser's developer tools are open, specifically the Elements tab. The DOM tree shows a complex structure involving shadow roots. The element `<input type="text" id="pizza" placeholder="Enter pizza name">` is selected, and its full XPath path is visible in the status bar at the bottom of the DevTools interface: `<input type="text" id="pizza" placeholder="Enter pizza name"> == $0`.

## Copy js path.



## Paste in console.



```
> document.querySelector("#userName").shadowRoot.querySelector("#app2").shadowRoot.querySelector("#pizza")
<- input#pizza

> document.querySelector("#userName").shadowRoot.querySelector("#app2").shadowRoot.querySelector("#pizza")
<- <input type="text" id="pizza" placeholder="Enter pizza name">
>
```

Document.querySelector 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.

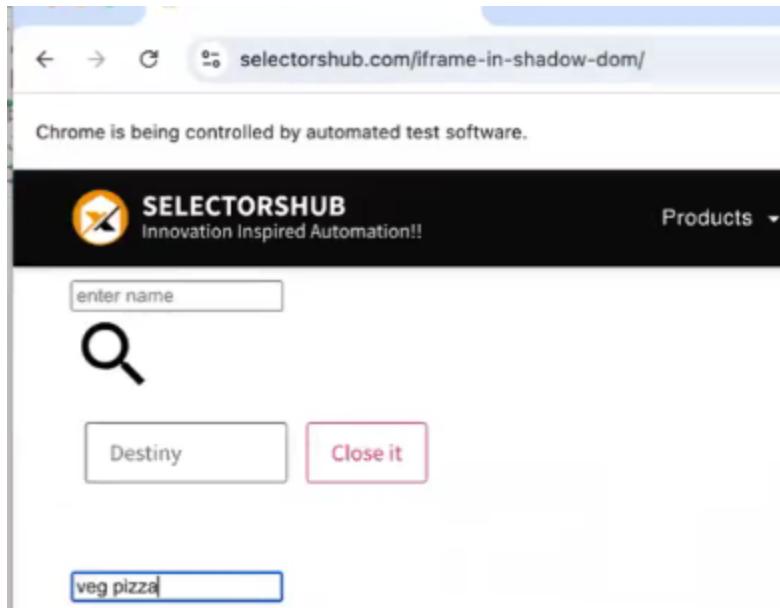
Awesome code ready-

```
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('#pizza')";
16
17     JavascriptExecutor js = (JavascriptExecutor)driver;
18
19
20     WebElement pizza = (WebElement)js.executeScript(script); //html element ----> webElement
21     pizza.sendKeys("veg pizza");
22
23     //driver.findElement(By.id("pizza")).sendKeys("veg pizza");
24 }
```

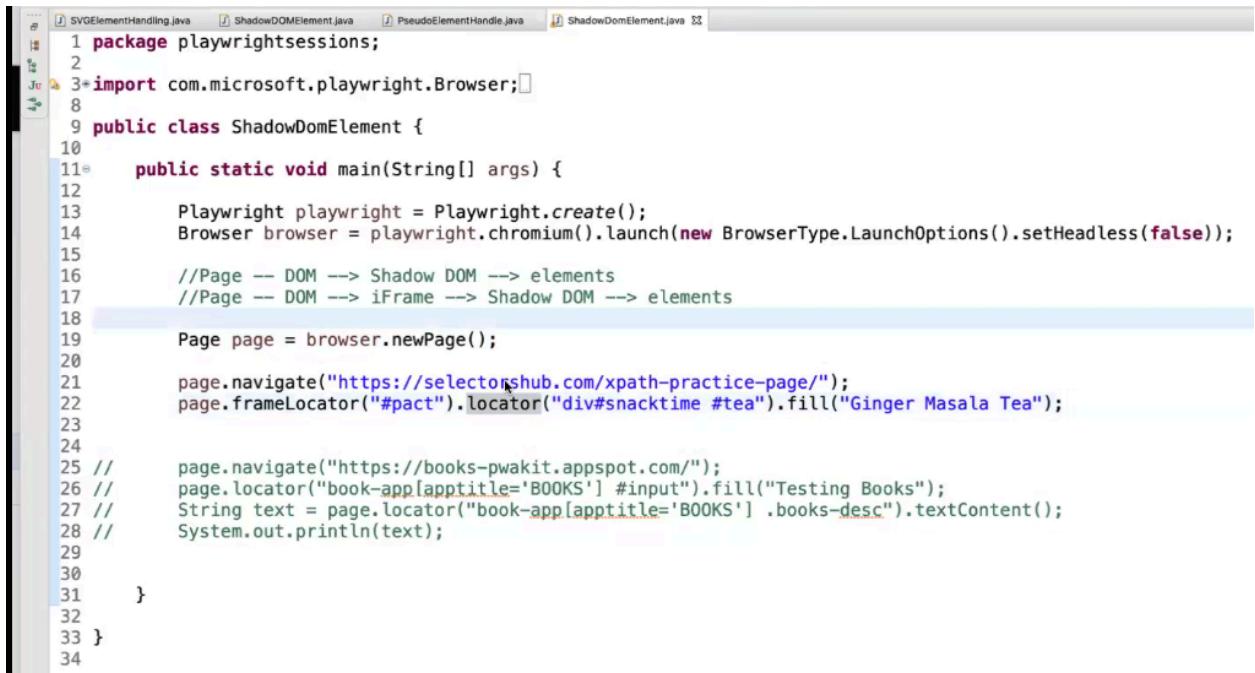
Line 15 clearer version-

```
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.



```
1 package playwrightsessions;
2
3+import com.microsoft.playwright.Browser;
4
5 public class ShadowDomElement {
6
7     public static void main(String[] args) {
8
9         Playwright playwright = Playwright.create();
10        Browser browser = playwright.chromium().launch(new BrowserType.LaunchOptions().setHeadless(false));
11
12        //Page -- DOM --> Shadow DOM --> elements
13        //Page -- DOM --> iFrame --> Shadow DOM --> elements
14
15        Page page = browser.newPage();
16
17        page.navigate("https://selectorshub.com/xpath-practice-page/");
18        page.frameLocator("#pact").locator("div#snacktime #tea").fill("Ginger Masala Tea");
19
20
21        page.navigate("https://books-pwakit.appspot.com/");
22        page.locator("book-app[apptitle='BOOKS'] #input").fill("Testing Books");
23        String text = page.locator("book-app[apptitle='BOOKS'] .books-desc").textContent();
24        System.out.println(text);
25
26
27
28
29
30
31    }
32
33 }
34
```

## Pseudo elements-

The screenshot shows a web browser window with the URL [naveenautomationlabs.com/opencart/index.php?route=account/register](https://naveenautomationlabs.com/opencart/index.php?route=account/register). The page title is "Register Account". Below the title, a sub-header says "If you already have an account with us, please login at the [login page](#)". The main form section is titled "Your Personal Details". It contains two input fields: "First Name" and "Last Name", both with red asterisks indicating they are required fields.

In the bottom right corner of the browser window, the developer tools are open, specifically the "Elements" tab. The DOM tree is visible, showing the HTML structure of the page. A tooltip is displayed over the "First Name" label, highlighting the pseudo-class selector `::before`. The tooltip text reads "label.col-sm-2.control-label::b before" and indicates a size of "9.66 x 16.4".

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.

```

    <div>.required .control-label::before {
        content: '*';
        color: #F00;
        font-weight: bold;
    }
    ::after, ::before {
        -webkit-box-sizing: border-box;
    }

```

Cant use normal locators to reach pseudo. 41.00  
The only way to reach pseudo is like this-

```

> window.getComputedStyle(document.querySelector('label[for="input-firstname"]'), '::before')
< CSSStyleDeclaration {0: 'accent-color', 1: 'align-content', 2: 'align-items', 3: 'align-self', 4: 'alignment-baseline', 5: 'anchor-name', 6: 'anchor-scope', 7: 'animation-name', 8: 'animation-delay', 9: 'animation-direction', 10: 'animation-duration', 11: 'animation-fill-mode', 12: 'animation-iteration-count', 13: 'animation-man-e', 14: 'animation-play-state', 15: 'animation-range-end', 16: 'animation-range-start', 17: 'animation-timeline', 18: 'animation-timing-function', 19: 'app-region', 20: 'appearance', 21: 'backdrop-filter', 22: 'backface-visibility', 23: 'background-attachment', 24: 'background-blend-mode', 25: 'background-clip', 26: 'background-color', 27: 'background-image', 28: 'background-origin', 29: 'background-position', 30: 'background-repeat', 31: 'background-size', 32: 'baseline-shift', 33: 'baseline-source', 34: 'block-size', 35: 'block-block-end-color', 36: 'border-block-end-style', 37: 'border-block-end-width', 38: 'border-block-start-color', 39: 'border-block-start-style', 40: 'border-block-start-width', 41: 'border-bottom-color', 42: 'border-bottom-left-radius', 43: 'border-bottom-right-radius', 44: 'border-bottom-style', 45: 'border-bottom-width', 46: 'border-collapse', 47: 'border-end-end-radius', 48: 'border-image-outset', 49: 'border-image-repeat', 51: 'border-image-slice', 52: 'border-image-source', 53: 'border-image-width', 54: 'border-inline-end-color', 55: 'border-inline-end-style', 56: 'border-inline-end-width', 57: 'border-in-line-start-color', 58: 'border-inline-start-style', 59: 'border-inline-start-width', 60: 'border-left-color', 61: 'border-left-style', 62: 'border-left-width', 63: 'border-right-color', 64: 'border-right-style', 65: 'border-right-width', 66: 'border-start-end-radius', 67: 'border-start-start-radius', 68: 'border-top-color', 69: 'border-top-left-radius', 70: 'border-top-right-radius', 71: 'border-top-style', 72: 'border-top-width', 73: 'bottom', 74: 'box-decoration-break', 75: 'box-shadow', 76: 'box-sizing', 77: 'break-after', 78: 'break-before', 79: 'break-inside', 80: 'buffered-rendering', 81: 'caption-side', 82: 'caret-color', 83: 'clear', 84: 'clip', 85: 'clip-path', 86: 'clip-rule', 87: 'color', 88: 'color-interpolation', 89: 'color-interpolation-filters', 90: 'color-rendering', 91: 'column-count', 92: 'column-gap', 93: 'column-rule-color', 94: 'column-rule-style', 95: 'column-rule-width', 96: 'column-width', 97: 'column-span', 98: 'contain-intrinsic-height', ...}

```

To get only the star icon-

```

> window.getComputedStyle(document.querySelector('label[for="input-firstname"]'), '::before').getPropertyValue('content')
< "*"

```

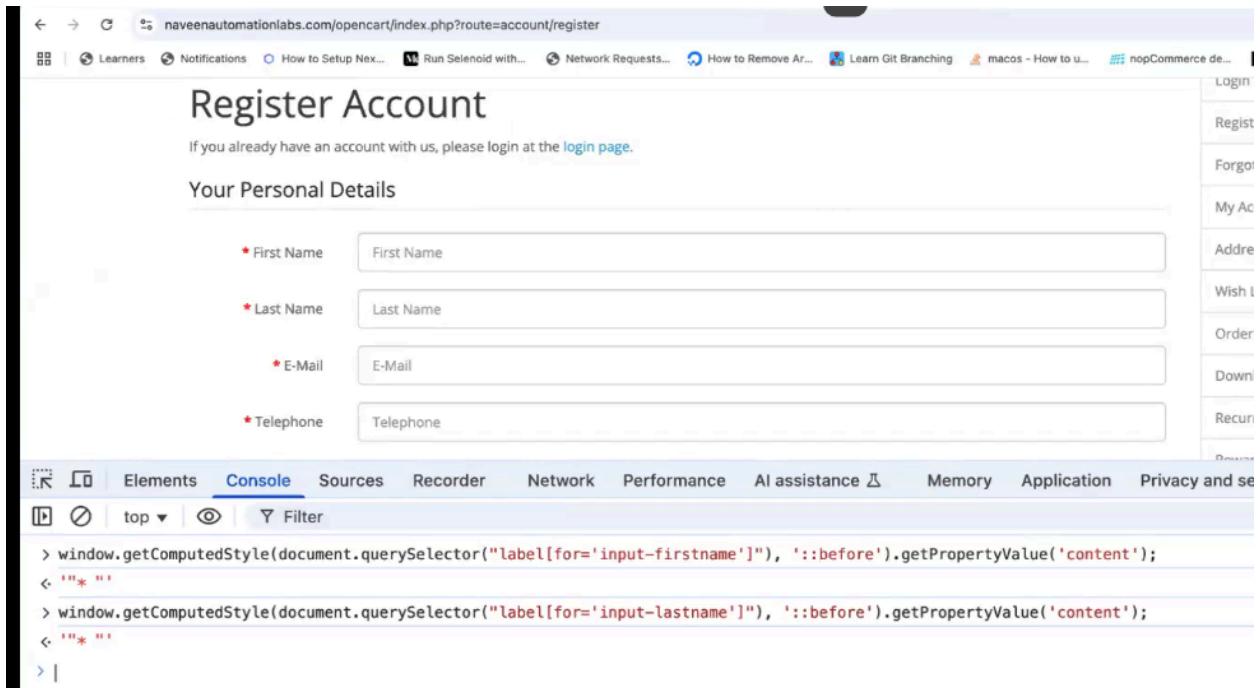
To get color property-

```

> window.getComputedStyle(document.querySelector('label[for="input-firstname"]'), '::before').getPropertyValue('color')
< 'rgb(255, 0, 0)'

```

Get content style for last name field-



naveneautomationlabs.com/opencart/index.php?route=account/register

Learners Notifications How to Setup Next... Run Selenium with... Network Requests... How to Remove Ar... Learn Git Branching macos - How to u... nopCommerce de...

## Register Account

If you already have an account with us, please login at the [login page](#).

### Your Personal Details

\* First Name

\* Last Name

\* E-Mail

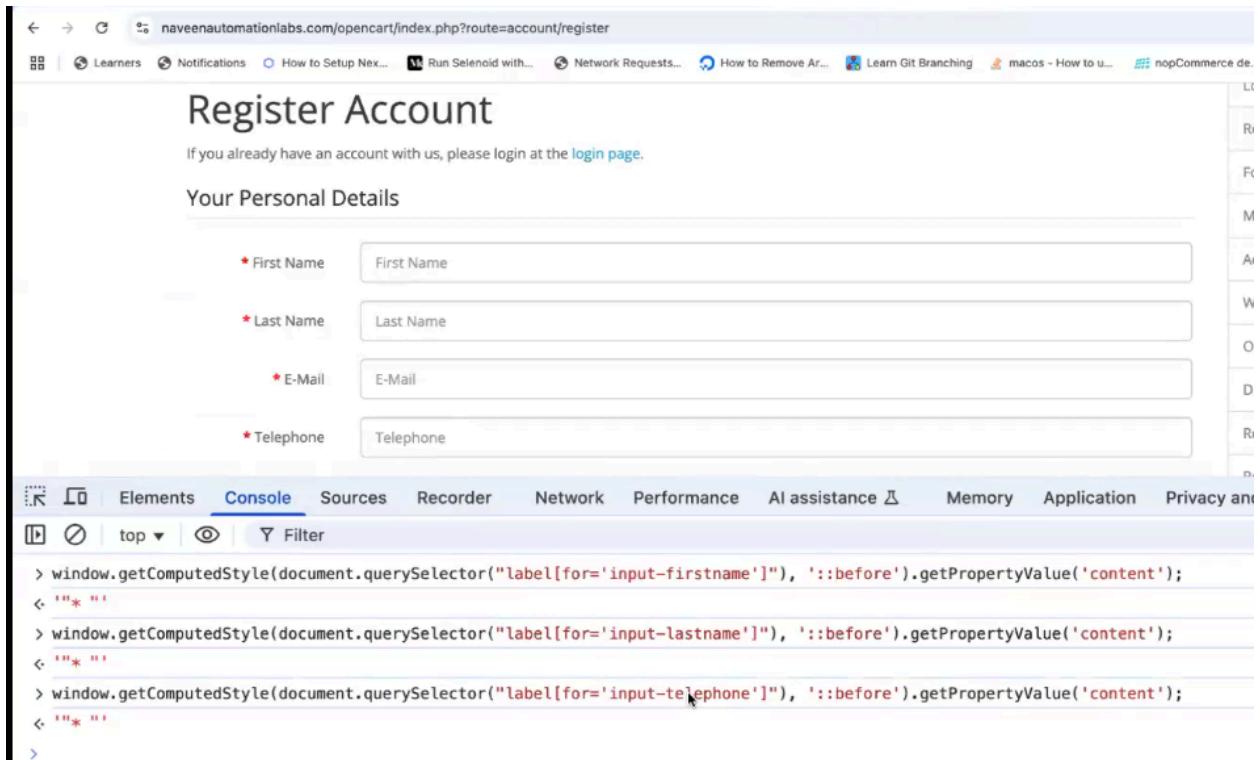
\* Telephone

Elements Console Sources Recorder Network Performance AI assistance Memory Application Privacy and se

top Filter

```
> window.getComputedStyle(document.querySelector("label[for='input-firstname']"), '::before').getPropertyValue('content');
< ""* ""
> window.getComputedStyle(document.querySelector("label[for='input-lastname']"), '::before').getPropertyValue('content');
< ""* ""
> |
```

## Get Star for telephone field-



naveneautomationlabs.com/opencart/index.php?route=account/register

Learners Notifications How to Setup Next... Run Selenium with... Network Requests... How to Remove Ar... Learn Git Branching macos - How to u... nopCommerce de...

## Register Account

If you already have an account with us, please login at the [login page](#).

### Your Personal Details

\* First Name

\* Last Name

\* E-Mail

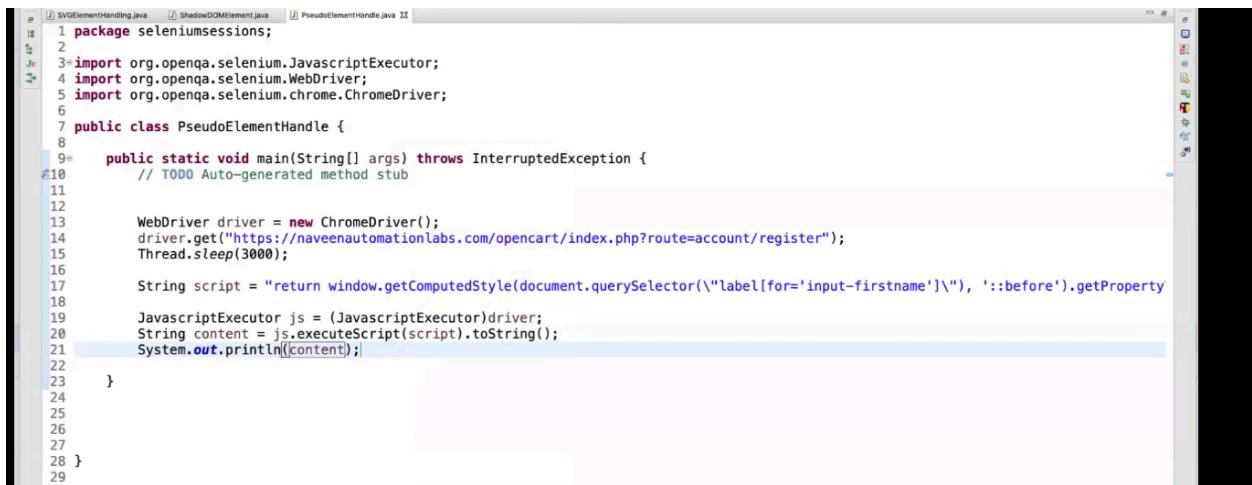
\* Telephone

Elements Console Sources Recorder Network Performance AI assistance Memory Application Privacy and se

top Filter

```
> window.getComputedStyle(document.querySelector("label[for='input-firstname']"), '::before').getPropertyValue('content');
< ""* ""
> window.getComputedStyle(document.querySelector("label[for='input-lastname']"), '::before').getPropertyValue('content');
< ""* ""
> window.getComputedStyle(document.querySelector("label[for='input-telephone']"), '::before').getPropertyValue('content');
< ""* ""
> |
```

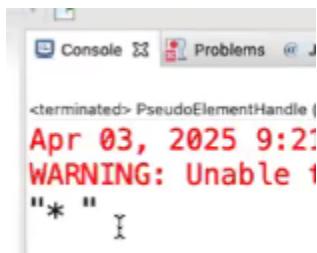
Return keyword for js script needed when it returns something. Like here we get the star. Else not needed. We get js string. Convert to string for java.



```
1 package seleniumsessions;
2
3 import org.openqa.selenium.JavascriptExecutor;
4 import org.openqa.selenium.WebDriver;
5 import org.openqa.selenium.chrome.ChromeDriver;
6
7 public class PseudoElementHandle {
8
9     public static void main(String[] args) throws InterruptedException {
10         // TODO Auto-generated method stub
11
12         WebDriver driver = new ChromeDriver();
13         driver.get("https://naveenautomationlabs.com/opencart/index.php?route=account/register");
14         Thread.sleep(3000);
15
16         String script = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), '::before').getPropertyValue('content')";
17
18         JavascriptExecutor js = (JavascriptExecutor)driver;
19         String content = js.executeScript(script).toString();
20         System.out.println(content);
21
22     }
23
24
25
26
27
28 }
29
```

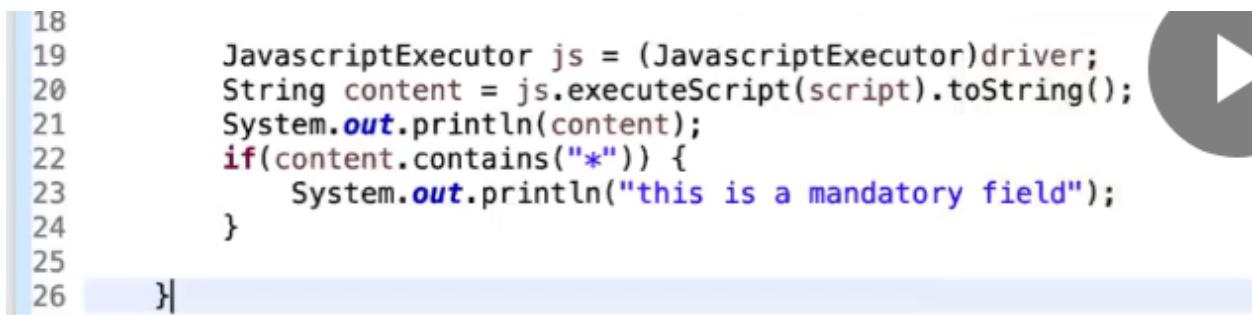
## Clearer line 17-

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



```
<terminated> PseudoElementHandle |
Apr 03, 2025 9:21:14
WARNING: Unable to find element
*
```

## Add if statement-



```
18
19         JavascriptExecutor js = (JavascriptExecutor)driver;
20         String content = js.executeScript(script).toString();
21         System.out.println(content);
22         if(content.contains("*")) {
23             System.out.println("this is a mandatory field");
24         }
25
26     }
27
```

```
Console Problems Javadoc Declaration Program  
<terminated> PseudoElementHandle (2) [Java Application] /Users/naveer  
Apr 03, 2025 9:22:47 AM org.open  
WARNING: Unable to find an exact  
" * "  
this is a mandatory field
```

## Clearer script for line 17 and 18-

```
16 String script = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), ':before').getPropertyValue('content');";  
17 String scriptColor = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), ':before').getPropertyValue('color');";  
18  
19 String script = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), ':before').getPropertyValue('content');";  
20 String scriptColor = "return  
window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), ':before').getPropertyValue('color');";  
21
```

## Line 18 added for color check-

```
16 String script = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), ':before').getPropertyValue('content');";  
17 String scriptColor = "return window.getComputedStyle(document.querySelector(\"label[for='input-firstname']\"), ':before').getPropertyValue('color');";  
18  
19
```

## Clearer picture-

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

## Check color displayed-

```
28 String color = js.executeScript(scriptColor).toString();  
29 System.out.println(color);  
30  
31
```

```
Console Problems Javadoc Declaration Program  
<terminated> PseudoElementHandle (2) [Java Application] /User:  
Apr 03, 2025 9:24:15 AM org.open  
WARNING: Unable to find an exact  
" * "  
this is a mandatory field  
rgb(255, 0, 0)
```

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.



```
1 package seleniumsessions;
2
3 import org.openqa.selenium.By;
4 import org.openqa.selenium.WebDriver;
5 import org.openqa.selenium.WebElement;
6 import org.openqa.selenium.chrome.ChromeDriver;
7
8 public class StaleElementRefException {
9
10    public static void main(String[] args) {
11
12        WebDriver driver = new ChromeDriver();
13        driver.get("https://naveenautomationlabs.com/opencart/index.php?route=account/register");
14
15        WebElement firstname = driver.findElement(By.id("input-firstname"));
16
17        firstname.sendKeys("naveen@gmail.com");
18
19        driver.navigate().refresh();
20
21        firstname.sendKeys("mohit@gmail.com");
22
23    }
24
25
26 }
27
```

Added 21-

```
18
19         driver.navigate().refresh();
20
21     Thread.sleep(2000);
22
```

The screenshot shows a web browser window with the URL [naveenautomationlabs.com/opencart/index.php?route=account/register](http://naveenautomationlabs.com/opencart/index.php?route=account/register). A message at the top states "Chrome is being controlled by automated test software." The header includes a currency dropdown set to "\$", a phone number input field containing "123456789", and a search bar. The main content area features the Naveen Automation Labs logo with the tagline "AUTOMATION SIMPLIFIED". A navigation menu below the logo lists categories: Desktops, Laptops & Notebooks, Components, Tablets, Software, Phones & PDAs, Cameras, and MP3 Players. Below the menu, a breadcrumb navigation shows the path: Home > Account > Register. The main title "Register Account" is displayed prominently.

Chrome is being controlled by automated test software.

\$ Currency ▾ 123456789

Naveen Automation Labs  
AUTOMATION SIMPLIFIED

Desktops Laptops & Notebooks Components Tablets Software Phones & PDAs Cameras MP3 Players

Home Account Register

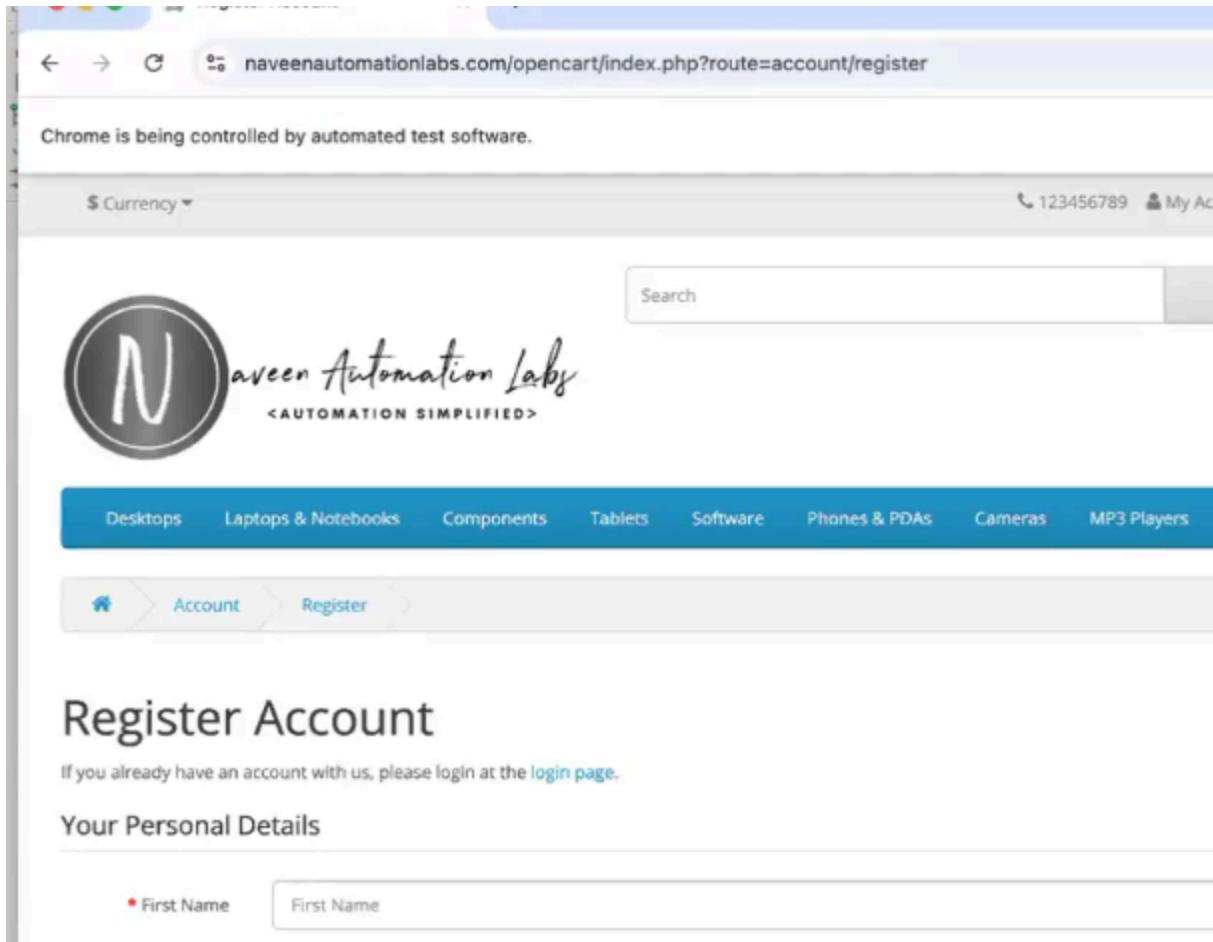
## Register Account

If you already have an account with us, please login at the [login page](#).

### Your Personal Details

\* First Name

Then refreshed.



Stale element exception.  
New value not added.

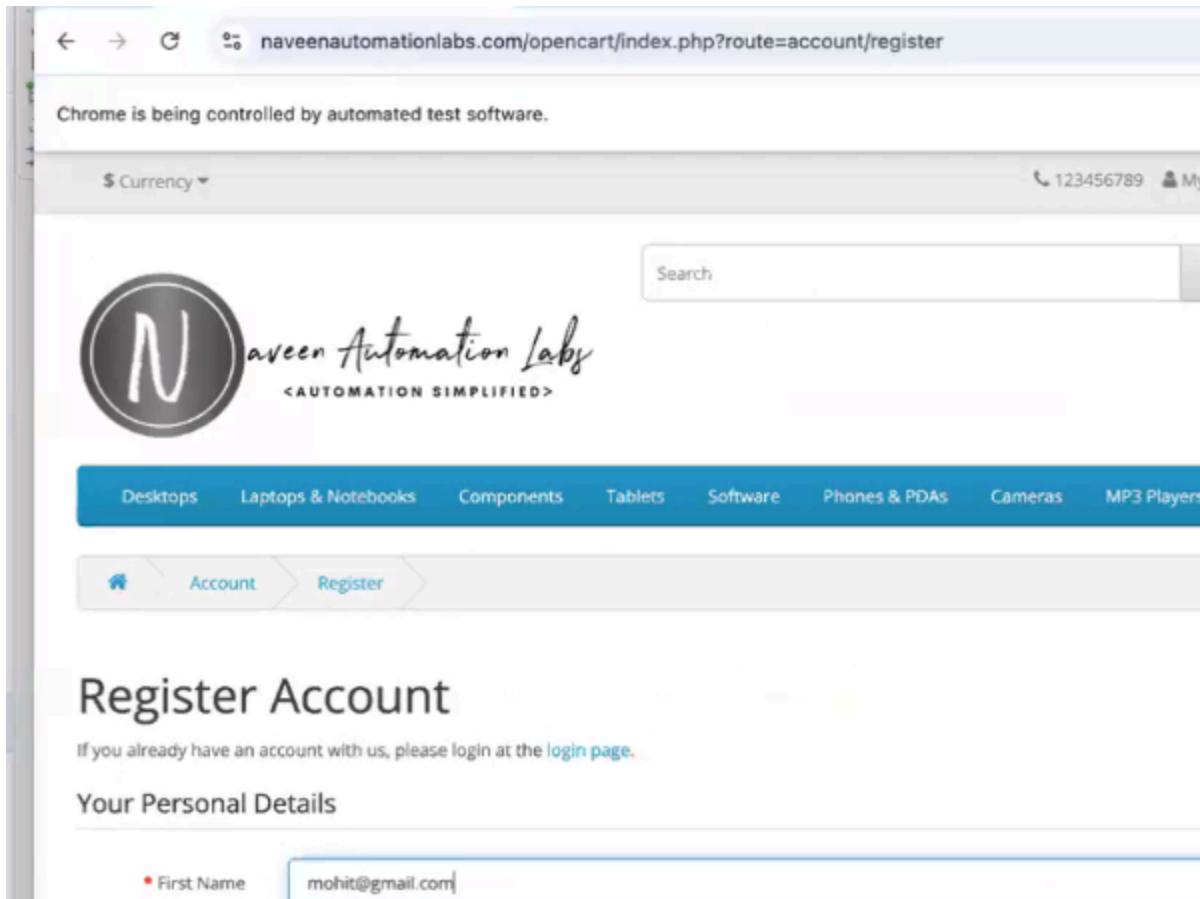
terminated: StaleElementReferenceException (1) [Java Application] /Users/naveenautomationlabs/p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx-x86\_64\_14.0.2.v20200816-0932/jre/bin/java (03-Apr-2025, 9:29:22 am - 9:29:34 am)  
or 03, 2025 9:29:29 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch  
WARNING: Unable to find an exact match for CDP version 134, returning the closest version; found: 133; F  
ception in thread "main" org.openqa.selenium.StaleElementReferenceException: stale element reference:  
(Session info: chrome=134.0.6998.166)

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();           I
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.

```

9
10 public st
11     WebDr
12         drive
13             //DOM
14             WebEl
15             drive
16             //DOM
17             first
18             //DOM
19             first
20             drive
21             //DOM
22             first
23             //DOM
24             first
25             //Dom
26             first
27             first
28
29 }
30 }
31 }
32 }
33

```

The screenshot shows a code editor with Java-like syntax. A context menu is open over a line of code containing 'first'. The menu includes options like Open With, Cut, Copy, Paste, Quick Fix, Source, Refactor, Local History, References, Declarations, Add to Snippets..., All Instances..., Instance Count..., Force Return, Watch (which is highlighted in blue), Inspect, Display, Execute, Run to Line, Step Into Selection, Run As, and Debug As.

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

```

1 v1 dom:
2 e1 -->
3 f.A0B927B9AE85DAE77347FB7E5811363C.d.C353F141316C286357936BB6A1DD61FC.e.6
4
5 f.A0B927B9AE85DAE77347FB7E5811363C.d.C353F141316C286357936BB6A1DD61FC.e.6
6
7 refresh
8
9 v2 dom:
10
11 e1--
12 f.A0B927B9AE85DAE77347FB7E5811363C.d.C353F141316C286357936BB6A1DD61FC.e.6
13
14 new id will be created
15 f.A0B927B9AE85DAE77347FB7E5811363C.d.0C19A67730B573D2B94ECA40ED4337E0.e.84

```

The terminal window displays a sequence of commands and their outputs. It shows two versions of a DOM element ('v1' and 'v2') with different internal IDs ('e1'). After a 'refresh' command, the element is identified by a new ID ('e1--'). The final output indicates a new ID will be created ('new id will be created').

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

```
11
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
```

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

```
v1 dom:
1 v1 dom:
2 e1 -->
3 f.A0B927B9AE85DAE77347FB7E5811363C.d.C353F141316C286357936BB6A1DD61FC.e.6
4
5 f.A0B927B9AE85DAE77347FB7E5811363C.d.C353F141316C286357936BB6A1DD61FC.e.6
6
7 refresh
8
9 v2 dom:
10
11 e1--
12
13 old id -- new dom
14 f.A0B927B9AE85DAE77347FB7E5811363C.d.C353F141316C286357936BB6A1DD61FC.e.6
15
```

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