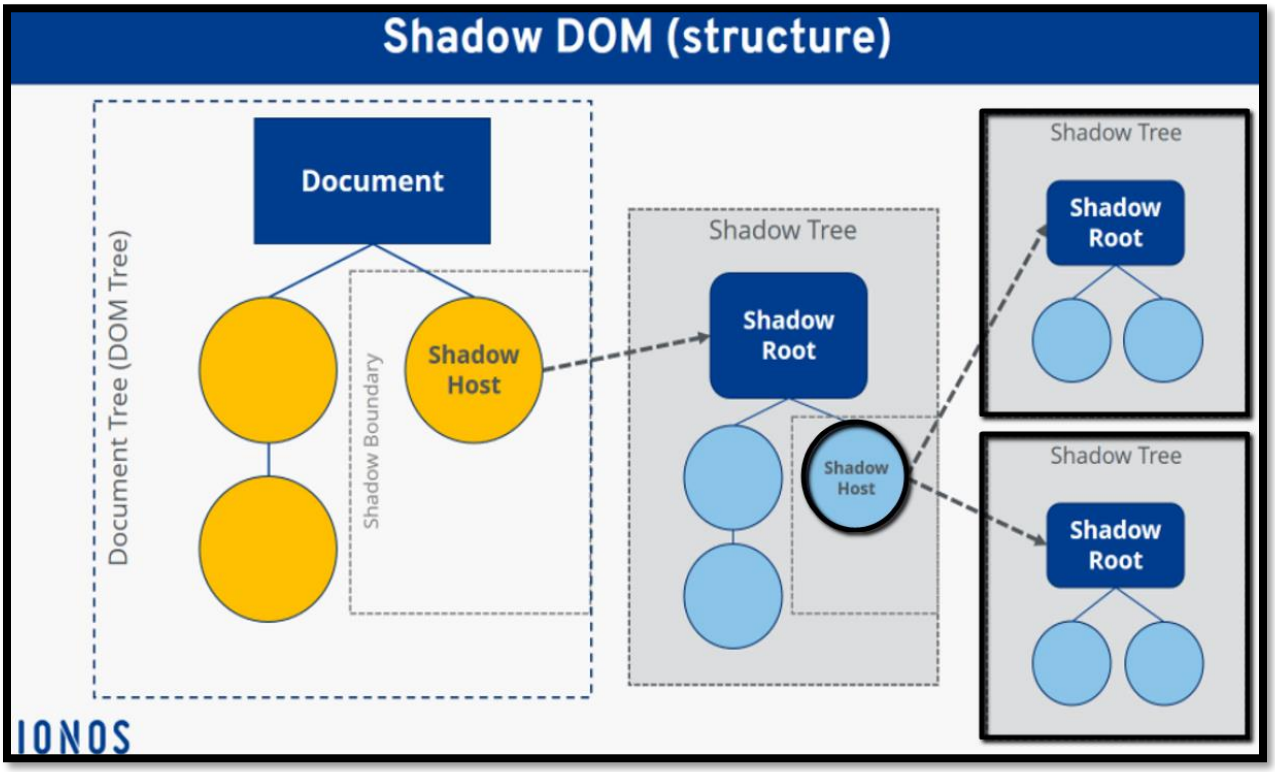
Rex Jones II

How To Locate An Element In Nested Shadow DOM

In this session, we are going to find an element in a Nested Shadow DOM. The previous 2 sessions showed different ways of finding an element. We found an element using Selenium's findElement method and without the findElement method. I am going to locate the nested Shadow DOM element using a JavaScript expression.

A nested Shadow DOM is when the Shadow DOM is located inside of another Shadow DOM. For example, in this diagram we see the last 2 gray boxes are extended from the first gray box. The first gray box is a Shadow DOM and has a Shadow Host. It splits into more Shadow DOMs.



For our code, it is set up to use Chrome, maximize the window, and to load the application.

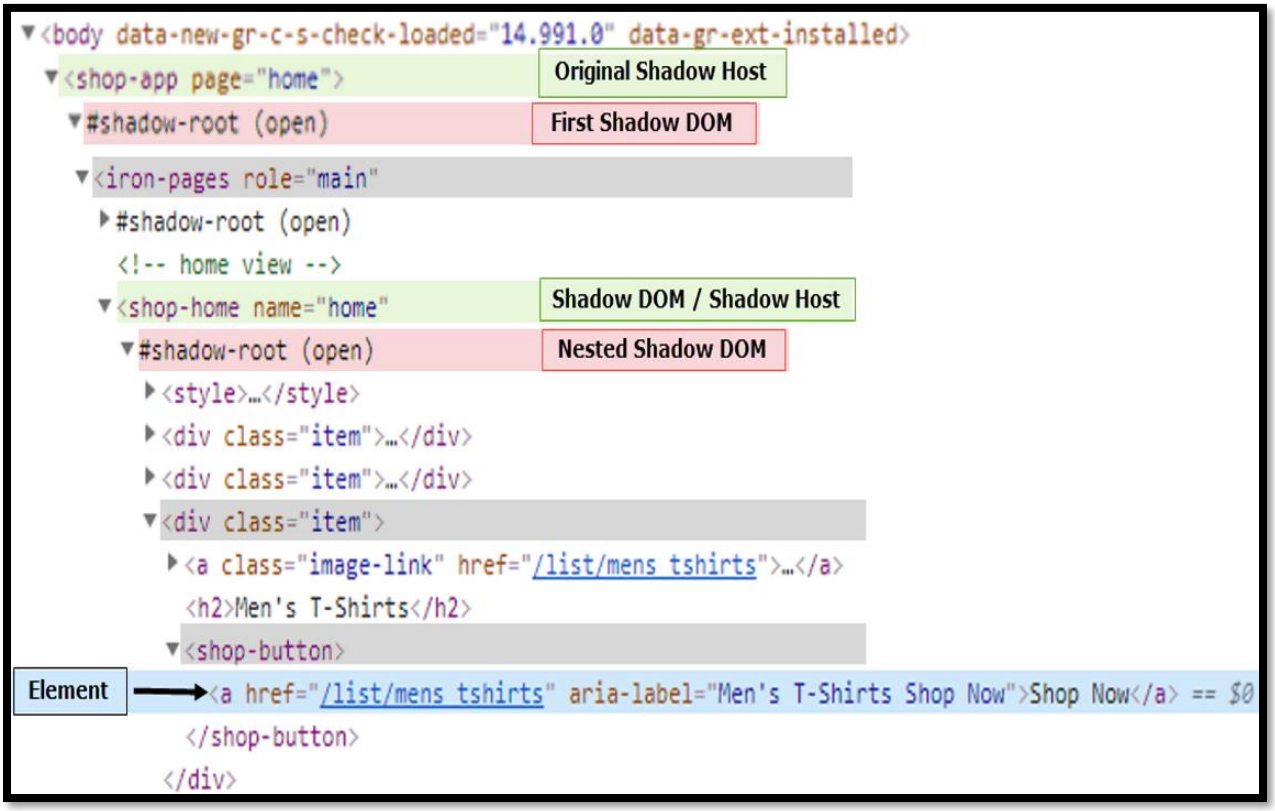
```
@BeforeClass
public void setUp () {
  WebDriverManager.chromedriver().setup();
  driver = new ChromeDriver();
  driver.manage().window().maximize();
  driver.get("https://shop.polymer-project.org/");
}
```

For our test, it starts @Test public void findElementInNestedShadowDOM () {}. There will be 2 main steps // Provide Access To JavaScript and // Find The WebElement So We Can Perform An Action

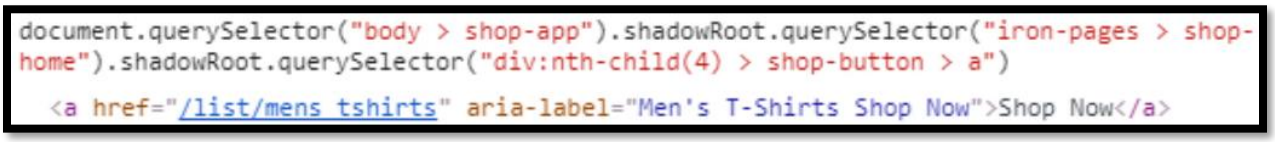
```
@Test
public void findElementInNestedShadowDOM () {
  // Provide Access To JavaScript
  JavascriptExecutor jsExecutor = (JavascriptExecutor) driver;

  // Find The WebElement So We Can Perform An Action

}
```

The (JavascriptExecutor) interface indicates a driver can execute JavaScript. Therefore we cast it then assign it to JavascriptExecutor with a reference jsExecutor =. Now, we must find the WebElement. Let's go to our AUT.

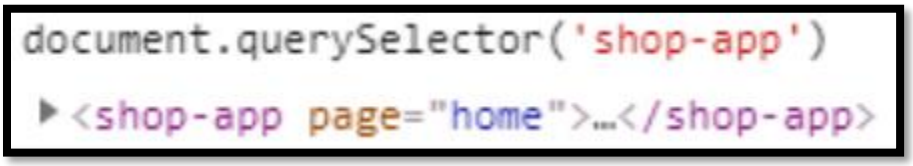
It's a Shop website for Men and Ladies. It has 4 Shop Now buttons and each Shop Now button is located inside of a nested Shadow DOM. We are going to click the Shop Now button for Men's T-Shirts. Inspect the Shop Now button. A fast indicator that the element is in a Shadow DOM is by looking at the bread crumb. We see shadow-root. However, it's a nested Shadow DOM because we see shadow-root 2 times. The complete structure is too big for this screen. Do you see how some of the nodes are collapsed and we are still not able to see both Shadow Hosts, both Shadow DOM's or element in 1 layout? I removed the unnecessary nodes so we can focus on the elements for our Test Script. All of the expanded nodes are highlighted green, red, or gray. Recall from the introduction, a Shadow Host is an element with a Shadow Tree. In this diagram, both Shadow Hosts are green. The Shadow DOM begins with a Shadow Root which is highlighted red. Our nested Shadow DOM is the second shadow-root. Then we see the button element starting with an <a> tag.



Just like in the second session, we can copy the JS Path by right clicking, selecting Copy then Copy JS path. However, in this session, we are going to create the JavaScript Path step-by-step. Go to the Console and clear the console. Know what, I'm going to go ahead and paste it then hit enter.



The JavaScript path starts with document.querySelector and a cssSelector value. This means the element matching body > shop-app will be returned so if I type document.querySelector and the cssSelector ('shop-app'). We see the next output has shop-app as the tag name.



Notice, 2 things about the path I wrote compared to the path that was copied from Elements tab. My path has single quotes and only shop-app as the CSS Selector value. The copied path has double quotes and body > shop-app as the CSS Selector nested values. Either way is okay, I chose to bypass the body

node. If we expand each node then all of the subsequent nodes will make this here look exactly like the Elements tab.

I'm going to collapse the node and go back to the diagram. We see shop-app is the Shadow Host and comes after body. Next, is the shadow-root which let's us know we are at the Shadow DOM. Within the first Shadow DOM is the iron-pages tag followed by the second Shadow Host shop-home. That's all we need for the first Shadow DOM. For CSS Selector, we can skip iron-pages and go directly to shop-home. In the Console, we can click the up arrow button and the up arrow shows the previous statement. To access the shadow root, we write dot .shadowRoot.querySelector('shop-home').

```
document.querySelector('shop-app').shadowRoot.querySelector('shop-home')
▶ <shop-home name="home" class="iron-selected" visible>…</shop-home>
```

shadowRoot represents the shadow root element and is the root for this component. Query Selector works inside of the Shadow Tree. Do you see how we are walking through the DOM step-by-step? In the diagram, we are at the nested Shadow DOM level and it's the same process. For the Shop Now button, I am going to get the shadow-root, bypass the div tag, then return shop-button and the element which starts with the <a> tag. In the Console, we write dot .shadowRoot.querySelector('shop-button > a'). Here's the anchor tag displaying our element.

```
document.querySelector('shop-app').shadowRoot.querySelector('shop-
home').shadowRoot.querySelector('shop-button > a')
  <a href="/list/mens_outerwear" aria-label="Men's Outerwear Shop Now">Shop Now</a>
```

Next we are going to copy the JavaScript path then go back to our code. We have the WebElement which is the shopNowButton =. It's important to convert to a (WebElement) from the jsExecutor.executeScript(" ") method. At this point, we return the JavaScript path. Let me modify the path so it all shows up on the same screen. This is the path we just created.

```
// Find The WebElement So We Can Perform An Action
WebElement shopNowButton = (WebElement) jsExecutor.executeScript(
        script: "return document.querySelector('shop-app')." +
                "shadowRoot.querySelector('shop-home')." +
                "shadowRoot.querySelector('shop-button > a')");
```

First line returns the shadow host from the original DOM. Second line is the shadow host from the nested Shadow DOM. Third line is the Shop Now button. With the shopNowButton, we click().

```
@Test
public void findElementInNestedShadowDOM () {
  // Provide Access To JavaScript
  JavascriptExecutor jsExecutor = (JavascriptExecutor) driver;

  // Find The WebElement So We Can Perform An Action
  WebElement shopNowButton = (WebElement) jsExecutor.executeScript(
          script: "return document.querySelector('shop-app')." +
                  "shadowRoot.querySelector('shop-home')." +
                  "shadowRoot.querySelector('shop-button > a')");
  shopNowButton.click();
}
```

Let's Run. The test passed. We see the button was clicked and the Next page shows up with items.

Contact Info

- ✔ Email Rex.Jones@Test4Success.org
- ✔ YouTube https://www.youtube.com/c/RexJonesII/videos
- ✔ Facebook https://facebook.com/JonesRexII
- ✔ Twitter https://twitter.com/RexJonesII
- ✔ GitHub https://github.com/RexJonesII/Free-Videos
- ✔ LinkedIn https://www.linkedin.com/in/rexjones34/