Selenium 4 Beginners

Rex Jones II

Selenium WebElement Methods

Table of Contents

Introduction

Hello everybody, Welcome To Selenium 4 Beginners. My name is Rex Allen Jones II.

In this video, I will cover the WebElement Methods for Selenium. The Browser Methods were covered in a video called Selenium Browser Methods.

I will cover all 5 Method Categories with each category having its own video. The WebElement Methods are a group of methods that perform actions on WebElements. A WebElement is anything you see on a browser such as buttons, text boxes, checkboxes, drop-down menus, and hyperlinks.

If you are interested, you can download this presentation, automation code, and Test Cases at https://tinyurl.com/SeleniumWebElementMethods

In this tutorial, we are going to cover some descriptions for WebElements, demo WebElements, and have a Practice Exercise.

WebElement Descriptions & Methods

Let's start with the descriptions.

There's a total of 16 WebElements but we will cover clear, click, findElement, getAttribute, getText, sendKeys, and submit. Clear – clears or erases a value in a text field. Click – clicks a link, button, checkbox, or radio button. findElement – finds the first WebElement based on the locator type you use to find your WebElement. getAttribute – gets the value based on a certain WebElement attribute. getText – gets the text from your web application. sendKeys – inserts or types text in a text field. Submit – operates like the click method in a form

Now, let's demo the WebElement Methods.

We are going to use SharpSpring as our Application Under Test (AUT). First, we will click the Login hyperlink. Next, we will use the Login page to showcase the WebElement Methods. Let's go to Eclipse.

WebDriver driver / @BeforeTest Annotation We set up our test public void setup then execute on Chrome and maximize our browser window. Driver dot manage dot window dot maximize

Our test method will be demo Selenium WebElement Methods @Test / (public void demoSeleniumWebElementMethods) Let's load SharpSpring using the get method: driver.get https://sharpspring.com/ The first WebElement Method we are going to look at, is findElement: driver.findElement. This method is used for finding WebElements on a browser. However, the findElement method will find the first element if there is more than 1 element with the same Attribute Value.

For example, the Home page has a menu bar with hyperlinks. If we inspect the links, we will see the class Attribute has the same Attribute Value for all of these hyperlink elements.

Let's inspect the Products element. It has mega-menu-link as the value for class. Let's also inspect Login, it has mega-menu-link as the value for class. Therefore, if we use className as the locator, the first element which is Products will get found every time. That's why It's best to use a locator with a unique Attribute Value. The hyperlink's name which is Login, is a unique value

Let me ask you a question, do you see within HTML that Login has a space? Our Selenium script will not work if we have a space before Login. Driver dot findElement By linkText "Login" I will remove the space. Dot. The next WebElement method is click.

We can also see the other WebElement Methods such as: clear, getAttribute, getCSSValue, and there's more WebElement methods in this intellisense. Do you see WebElement at the end of the method's syntax? That gives us a tip; that a particular method is a WebElement Method. I am going to select click.

On the Login page, we are going to type some text for email and password: driver.findElement By id dot sendKeys. sendKeys is a WebElement method that enters text into a field. I'm going to enter RexAllenJones@Hotmail.com for email. Copy and Paste then use this code for Password but change the value for sendKeys to Selenium4Beginners. Let's go back to the application and get the id value for email and password.

The id value for email is username and id value for password is password. username / password

The next WebElement method we are going to use is clear: driver.findElement by id "password" dot clear which will clear Selenium For Beginners.

After clearing the text, let's click the Sign In button. We can use the click or submit method. Recall from our PowerPoint slide, the submit method "operates like the click method if the WebElement is a form or within a form.

For now, let me show you how, click works with the Sign In button: driver.findeElement By id dot click. Let's inspect the button. The value for id is login-button which shows the button tag is located inside of the form tag. The button tag disappears when I collapse the form tag.

Let's Run – Invalid username and/or password. The Click method works. I need to let you know, there are cases when click will not work in a form. If click does not work, then use the submit method.

Now, let's change click to submit then Run. Submit also works. Did you notice the error message is different when using click and submit? The error message after using submit shows "Please enter your password" but the message states "Invalid username and/or password" after using click. You will get no message if the email address is invalid.

Submit will not work if we try to click the Login link because the link is outside a form. I will change click to submit. Let's look at the description for submit which shows it throws a NoSuchElementException if the element is not within a form.

Let's Run / NoSuchElementException; no such element: Element was not in a form, so could not submit. That's the difference between click and submit.

The next WebElement Method is getText. We are going to get the error message that shows up after clicking the Sign In button. First we find the error message: driver.findElement by xpath next we get the error message: dot getText. The description states "Get the visible innerText of this element". This description can be confusing if you don't know HTML.

Let me use W3 Schools to help explain the getText method and HTML. www.w3Schools.com / Learn HTML / Introduction. Do you see how tag name has an opening and closing tag? The only difference is closing tag has a forward slash before tag name. Content goes here. The getText method returns the content between the opening tag and closing tag. Sometimes HTML does not contain a closing tag. HTML tags normally come in pairs. That means the closing tag is not required.

Go back to our Application Under Test. Let's inspect the error message. In this case, the error message is between the opening and closing h4 tags. Copy and Paste xpath value

The getText method returns a String so let's assign the value to String strErrorMessage. Now, let's print the error message. Sysout "What Is The Error Message? " + strErrorMessage. Let's Run / What Is The Error Message? Please enter your password

To summarize, the getText method. It will return most data you see on a web page. It will not return data if there's no content after the opening tag. For example, the Password placeholder will not return data if we use the getText method. Although we see Password on our application, it does not have content. There is no content after the opening tag. The blue highlighted section is the opening tag for input which have 5 Attribute Names and Values.

Let me show you how getText will not work for this element: driver.findElement By id "password" dot getText. Assign the value to String strPlaceHolder. Sysout / getText = "I Will Enter My " + strPlaceHolder. Let's run

The Console does not show data after I Will Enter My. We must use the getAttribute method when we see data on an application but HTML does not have content for that element. I will use the same code and change getText to getAttribute. Copy and Paste. Change getAttribute method and getAttribute within the print statement. Also change the object reference strPlaceHolder to strPasswordLabel.

Add double quotes to the getAttribute value. The description states "Get the value of the given attribute of the element." That mean we can get any value by writing the attribute name. Let's verify which attribute name has the attribute value Password.

The placeholder attribute has the Password value we want to print. There are multiple attributes with password as a value but we need placeholder because it has a capital "P" for Password. Attributes type, id, and name contains a lowercase "p" for Password which does not match the application. Let's run again / getAttribute displays I Will Enter My password.

Let's end by closing the browser @AfterTest (public void teardown) / driver.quit That's all I have for Selenium WebElement Methods which are methods that perform actions on a method. In this video, we covered 4 types of WebElements: a hyperlink, a couple of text fields, a button, and text within the application.

Practice Exercise

You can use Free CRM and SharpSpring for Practice Exercise. Let's look at the Test Cases. We walked through SharpSpring in our tutorial. Here's your Practice Test Case for Free CRM. You can download your Presentation, Automation Code, and Test Cases at
https://tinyurl.com/SeleniumWebElementMethods