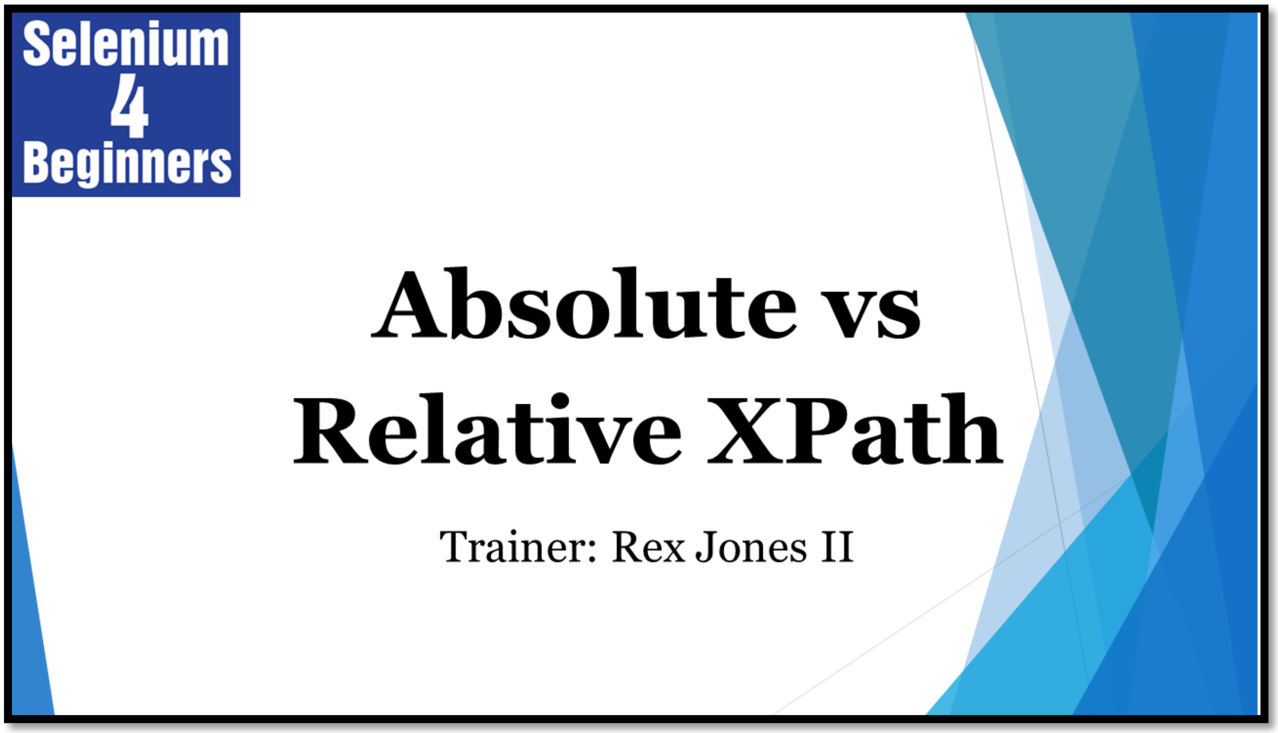
Rex Jones II

(Transcript) Absolute XPath vs Relative XPath



Introduction

We are going to discuss How To Create Our Own Customized XPath Values. The Transcript, Presentation, and Cheat Sheet will be available on github at RexJonesII/Selenium4Beginners and https://tinyurl.com/SeleniumLocatorsForWebElements.

We will look at the difference between Absolute and Relative XPath Values, XPath Functions, and XPath Axes/Axis. We are going to use 4 different applications to walkthrough all of these XPath scenarios.

Absolute vs Relative

What is XPath? XPath is used to navigate through elements and attributes. XPath is short for XML Path and XML stands for e**X**tentsible **M**arkup **L**anguage. XML is different from HTML. XML is used to carry data and HTML is used to display data. With regards to Selenium, XPath is a locator used to extract or pull Tag Attributes. There are 2 types of XPaths: Absolute and Relative

Here’s the syntax for an Absolute XPath. The Absolute XPath always begins with the root element which is html. It can start with a forward slash before html or without a forward slash before html. However, a forward slash will always come after html followed by head or body as the next element and stop with the last element.

Relative XPath always start with 2 forward slashes which selects a direct Tag Name followed by a condition. The condition is located inside 2 square brackets using an at symbol, Attribute equal to the value surrounded by single quotes. Our first application is Demo Web Shop. Let's inspect the Search button. As automation engineers, we should not copy the xpath values from Chrome DevTools. That's when we right click, select Copy and Copy XPath. CTRL + F then paste the value and we see an absolute xpath value in the locator. Sometimes Chrome DevTools generate a Relative XPath value. I will admit, I have used Chrome DevTools to copy XPath but that's because I had not created a video showing how to customize our own xpath values. From this point on, I will not copy xpath values.

Let's breakdown this Absolute XPath. It starts with a forward slash html then we see body. A forward slash provides access to the next element which is a child element. Within body, we see 4 div tags. 1, 2, 3, 4. The XPath shows div[4] which means the fourth div tag. Maximize div 4 and we 2 div tags. The XPath shows div[1]. Maximize div[1]. Now, we see 4 div tags but we need the 3rd div tag because XPath shows div[3]. Next is the form tag and finally the input tag. The XPath shows input[2] and we see all of the attributes for the Search button. That's a lot of navigation to find one element. It's not a good practice to write an Absolute XPath. If anything change between html and input[2] such as div[1] changing to div[2] then Selenium will no longer find the Search button.

That's one of the reasons why we should not use Absolute XPath. I recommend we use Relative XPath. Here's a cheat sheet for Relative XPath, I created it as a reference for you to download. First syntax, for the Search button, we write 2 forward slashes and the Tag Name input. There are 15 WebElements with input as their Tag Name. We can write any Tag Name and see how many elements contain that Tag Name. The Search button is 2 of 15 for input tag. Continue with an opening and closing square bracket, at symbol, then Attribute Name. The Attribute Name can be any attribute within this element. We want to use an attribute that has a unique value. Let's write type followed by an equal sign, after the equal sign, then write 2 single quotes, and the Attribute Value submit. We see the element shows a green highlighted background which means success. If we hover over the element then the WebElement is highlighted in the application. That's the difference between Absolute and Relative XPath.