STREEBO

XPATH LEARNING

Author : Radha Basida

Revised Version: Akmam Mashkoor

Table of Contents

1. What is Xpath?

XPath, which stands for "XML Path Language", is a syntax or language utilized to locate elements on a web page by employing XML path expressions. It is applied to determine the position of elements on a webpage based on the HTML Document Object Model (DOM) structure.

It is a powerful and versatile language used for navigating and querying XML documents and, in the context of web development, for navigating and interacting with the HTML structure of web pages. It provides a standardized way to traverse through elements and attributes within an XML or HTML document.

1.1 Syntax of Xpath:

Syntax =//tagname[@attribute='value']

//: This symbol denotes the current node in the document, which is typically the starting point for your search.

Tagname: Refers to the HTML tag name of the element you want to locate. For example, if you're looking for a specific div or input element, you would specify the tag name.

@: The "@" symbol is used to indicate that you're about to specify an attribute.

Attribute: This is the name of the attribute you want to target, such as "id" or "class."

Value: The value of the attribute you're looking for, which helps to uniquely identify the element.

XPath allows you to accurately pinpoint elements on a web page by leveraging these components. In addition to XPath, there are other methods, or locators, that can be used to find elements on web pages. These include:

ID: You can locate an element by its unique ID attribute.
Class Name: This method is used to find elements based on their class attribute.
Name: Elements can be located by their "name" attribute.
Link Text: This locator helps you find elements of the text contained within hyperlinks.
CSS Path: This is useful for locating elements that don't have specific name, class, or ID attributes, as it relies on the element's CSS path or selectors.

There are two types of Xpath: Absolute XPath and Relative XPath, and let's break down what each of these means.

Absolute XPath:

Absolute XPath is like providing the full address of an element within the web page's structure. It starts from the very beginning, or the root element of the Document Object Model (DOM). You can think of it as giving directions that include every single step, starting from the root and leading to your desired element.

For example: /html/head/body/table/tbody/tr/th

The issue with Absolute XPath is that it's very specific. If the structure of the web page changes at all, like if someone adds a new element, your Absolute XPath might break. This means you'd have to constantly update it to keep it working.

Relative XPath:

Relative XPath, on the other hand, is a bit more flexible. It's like giving directions starting from a known point, so you don't have to specify the entire path from the root element. It begins with double forward slashes (//), and you can start from an element you know won't change.

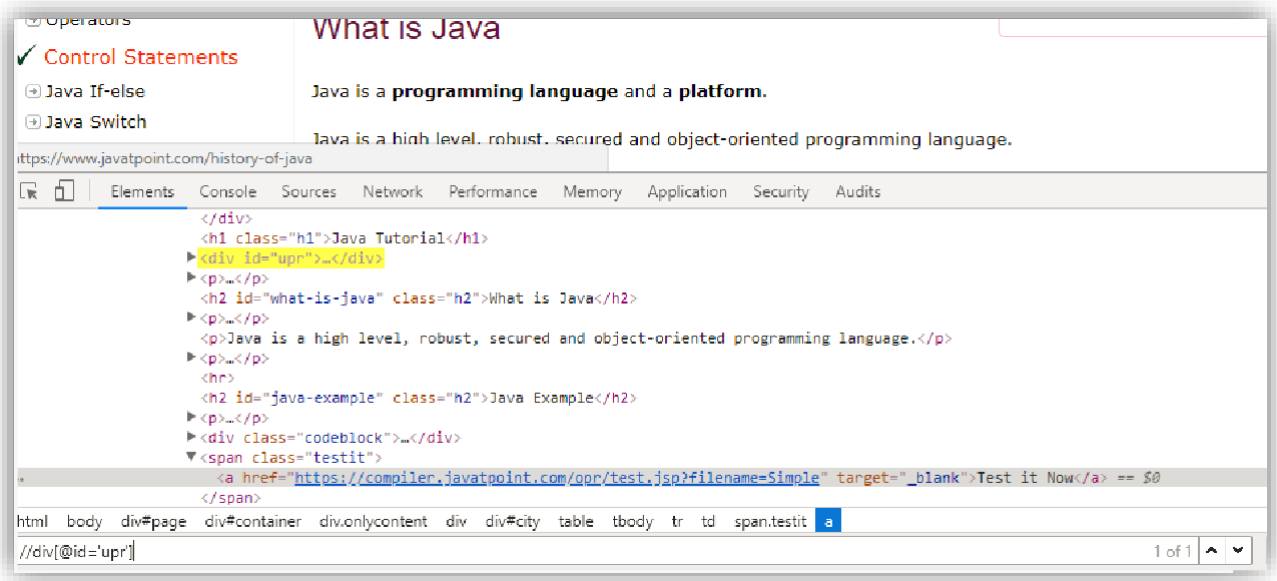
For example: //table/tbody/tr/th

With Relative XPath, you're providing a partial path. The advantage is that it won't break as easily if there are small changes in the structure. However, because it's not as specific, it might take a bit more time for the system to locate the element compared to Absolute XPath. Also, if there are multiple elements that match the same path, it'll usually pick the first one it finds.

In summary, Absolute XPath gives very detailed instructions from the beginning, while Relative XPath starts from a known point and is more flexible but may be a bit slower and might not be as precise if multiple elements match the same description. Beginners often find Relative XPath more manageable because it adapts better to changes in the web page structure.

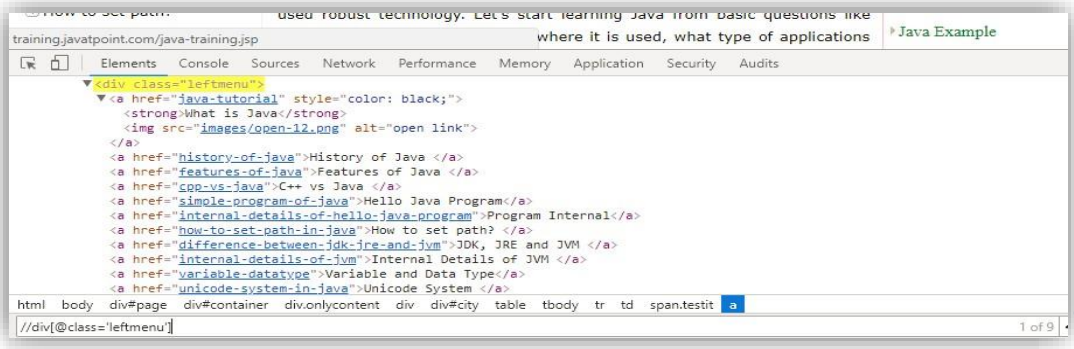
2. Dealing with Complex and Dynamic Elements using XPath in Selenium

2.1 Basic Xpath:

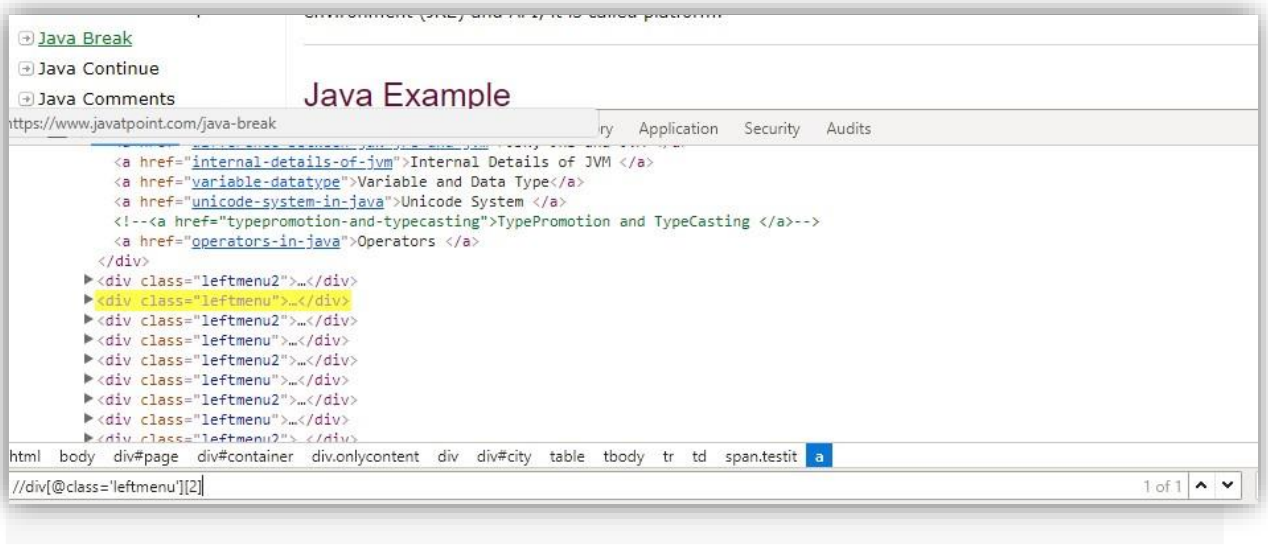


Xpath expression uses attributes and its value to identify the elements.

For instance, consider the XPath: //div[@id='upr'] – it locates the div element with the 'id' attribute set to 'upr'. In this specific case, there is only one element with this unique 'id' attribute. However, there are situations where an XPath expression might match multiple elements. In such cases, by default, Selenium will select and interact with the first element that matches the XPath expression in your script.



Xpath = //div[@class='left menu'] – it will give 9 results. you can also use index to identify element if structure of the DOM is fix as shown in below image



Xpath= //div[@class='leftmenu'][2] – it will give second element from the list identified by given Xpath expression.

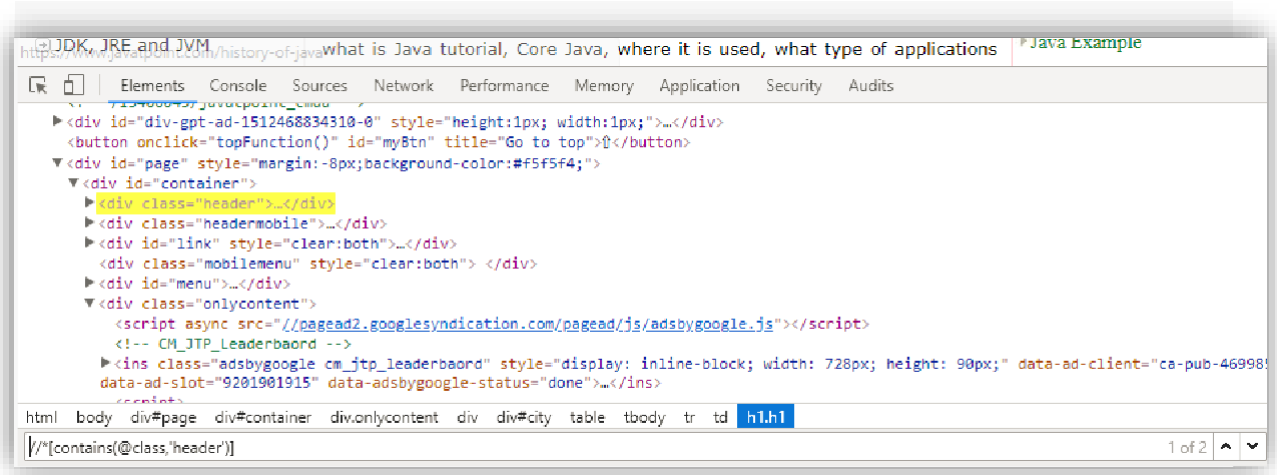
To retrieve not only just div element, but all the element of page having 'leftmenu' class, use '*' instead of tagname(input)

Xpath=//*[@class='leftmenu']

We'll learn about the contains method on next page…….

2.2 Contains() :

The contains() methos is a tool employed within XPath expressions, specifically designed for situations where attribute values undergo dynamic changes or share common characteristics.



Suppose we want div tag whose class name contains header text. You can get it by using below Xpath.

e.g.: //*[contains(@class,'header')]

2.3 Use AND & OR in Xpath:

We can also use OR and AND operation to find the element.
Xpath = //*[@class='next' or text()='next →']

2.4 Start-with function:

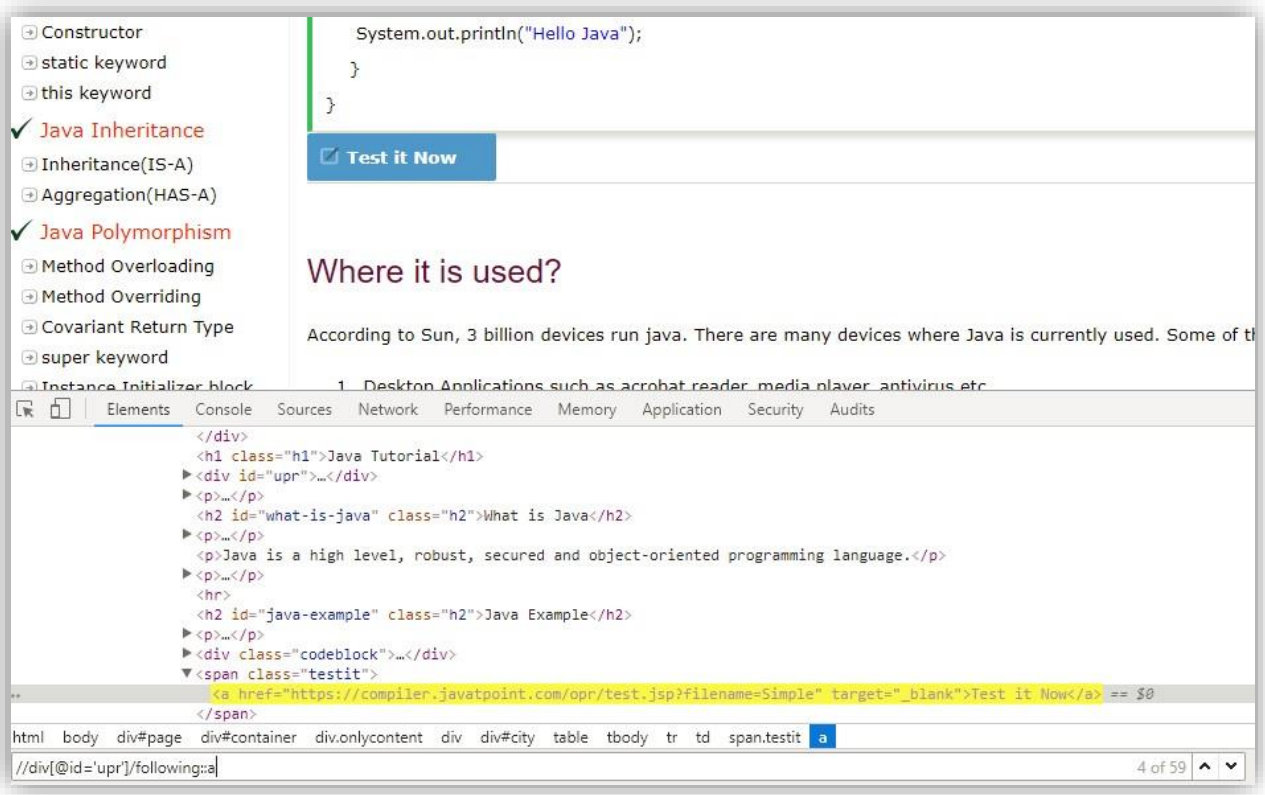
Start-with can be used to find elements whose attribute value changes dynamically and using same prefix value. For example -: Suppose the ID of changes dynamically like: Id=" btn_12" , Id=" btn_123".
Xpath=//*[starts-with(@id,'btn')]

2.5 Text():

Text() function is used to identify element based on the text attribute of the element.
Xpath=//input[text()='body text']

2.6 XPath axes methods:

a) **Following:** this will return all the elements after the current element.

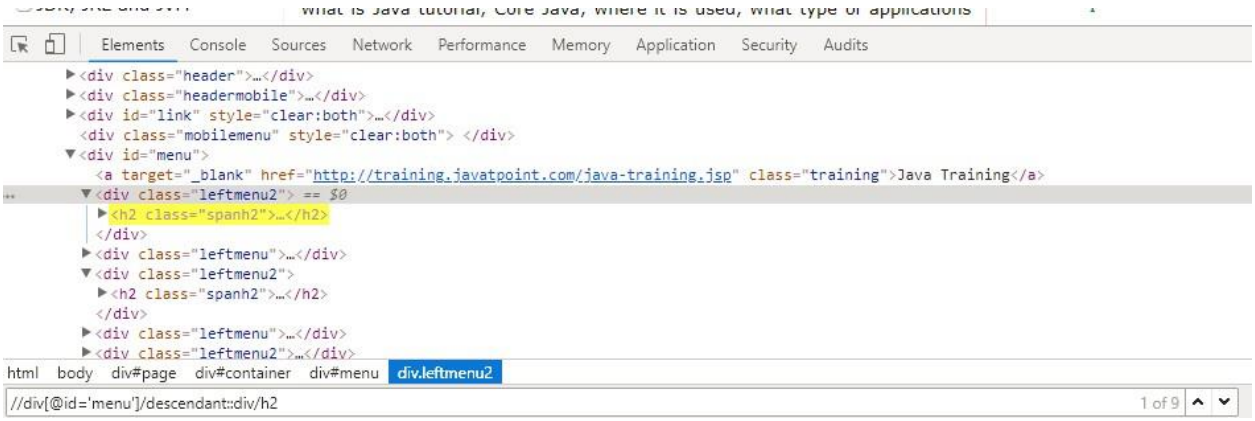


In the above example, it will give all the <a> elements which comes after the div element.
//div[@id='upr']/following::a

b) **Preceding:** In same way preceding will return all the elements coming before the current element.
Xpath: //div[@id='menu']/preceding::a[text()='Training']

c) **Ancestor:** Ancestor will give all the ancestor (parent, grandparent, etc.) of the current element. From the below expression it will give all the parent div element of the input element having id as input.
Xpath: //a[text()='Java Training']/ancestor::div

d) **Descendant:** Descendant will give all the descendant (child, grandchildren, etc.) of the current element.



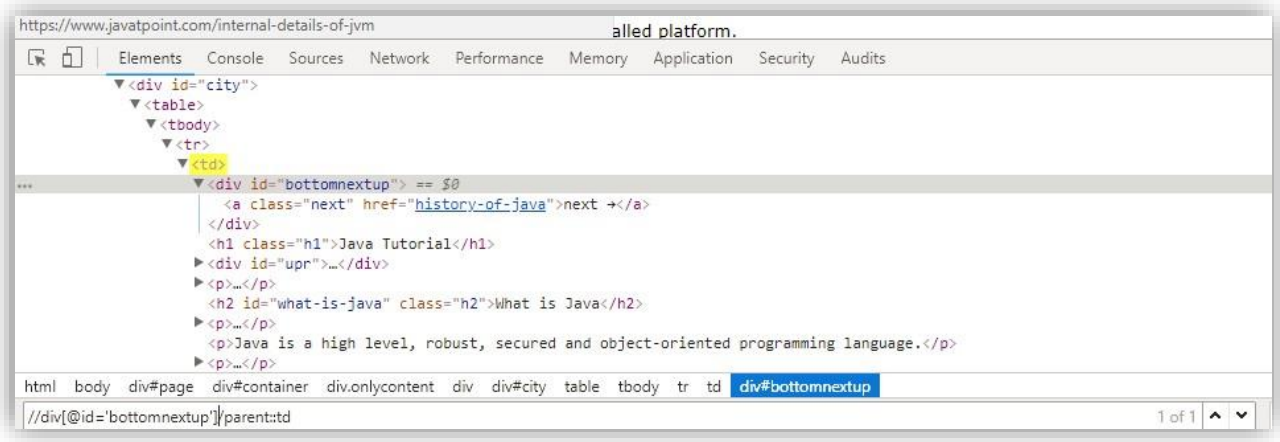
From the above expression it will give all the children div element of the input element having id as input. Xpath: //div[@id='menu']/descendant::div/h2

e) **Child** : Selects all children elements of the current node as shown in the below screen.
Xpath: //div[@class='leftmenu2'][1]/child::h2

f) **Following-sibling:** Select the following siblings of the context node. Siblings are at the same level of the current node as shown in the below screen. It will find the element after the current node.
Xpath: //div[@id='upr']/following-sibling::p

a) **Preceding-sibling:** Select the following siblings of the context node. Siblings are at the same level of the current node as shown in the below screen. It will find the element before the current node.
Xpath: //div[@id='menu']/preceding-sibling::div

h) **Parent:** Selects the parent of the current node as shown in the below screen.

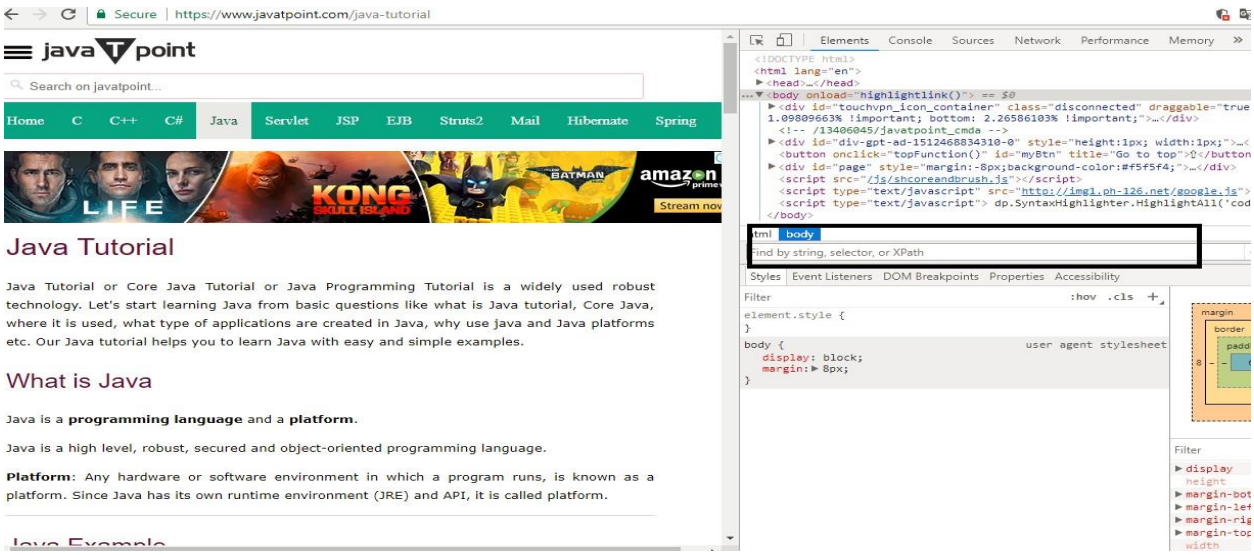


Xpath: //div[@id='bottomnextup']/parent::td

3. Identify and Verify Xpath in Web and Mobile Application

3.1 : Locate an element with Xpath in web (Chrome)

Press F12 to open developer tools and press ctrl + f , you will find one textbox to write Xpath and validate it as shown in below screen.



3.2: Locate an element with Xpath in Appium Studio(IOS)

Appium studio can be used for both android and IOS automation. Follow the steps below to locate an element in mobile screen using Xpath.

1. Connect device to Pc and open it in Appium studio
2. Open the Application in device. Mobile screens will be displayed in Appium studio. Click on th object spy button to display mobile view as shown in below screen.
3. After clicking on spy button, object properties screen will be displayed. You can direct copy Xpath from this screen and also write your own Xpath expression to locate an element. Let's go through both ways.

a) Using copy Xpath : right click on the element and click on the copy unique Xpath to use the Xpath.

b) Write your own Xpath : sometimes it may happen, Xpath generated from Appium studies looks complex because Appium studio mostly uses index and text based Xpath. So, writing own Xpath will be good practice as other users can also understand the Xpath.

3.3: Locate an element using Xpath with Appium Studio(Android) :

We have multiple options to identify elements for Android Devices like UI Automator Viewer, Appium desktop client, Appium studio.

Here we will continue with Appium studio as it will provide Xpath query filter to check whether Xpath is working or not. On the other side using the other two options, we can only view different attributes and their value.

Appium studio works the same for both IOS and Android applications. But the attributes will differ in both devices. Method for writing Xpath will be same as described in section 2.

Below attributes are used widely for android application:

- Resource-id/id
- Class
- Text
- Index

For further information or in case of any obstacles, please reach out to:

jaydip.vora@streebo.com, *akmam.mashkoor@streebo.com*.

THANK YOU