**Selenium Relative Locators**

1. <https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/>
2. [Home](https://www.toolsqa.com/" \o ")
3. [Selenium-Webdriver](https://www.toolsqa.com/category/selenium-webdriver/)
4. Selenium Relative Locators

BASICS

* [Java Basics](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Set Up WebDriver with Eclipse](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [WebDriver Commands](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Inspectors Tools & Locators](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)

INTERMEDIATE

* [Java Advance OOPs](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [WebDriver API](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Alerts & Windows](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Action Class](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Robot Class](http://t.com)
* [Tips & Tricks](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)

ADVANCE

* [Data Driven Approach](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Log4j Logging](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [DataBase Connections](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Tips & Tricks](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Grid](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Interview Questions](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)

FRAMEWORKS & DESIGN

* [Hybrid Automation Framework](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Keyword Driven Framework](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)
* [Framework Design principles](https://www.toolsqa.com/selenium-webdriver/selenium-relative-locators/)

In this article, we will cover one of the newly introduced ***Selenium Relative Locators*** of Web Driver API:

**What are Selenium Relative Locators?**

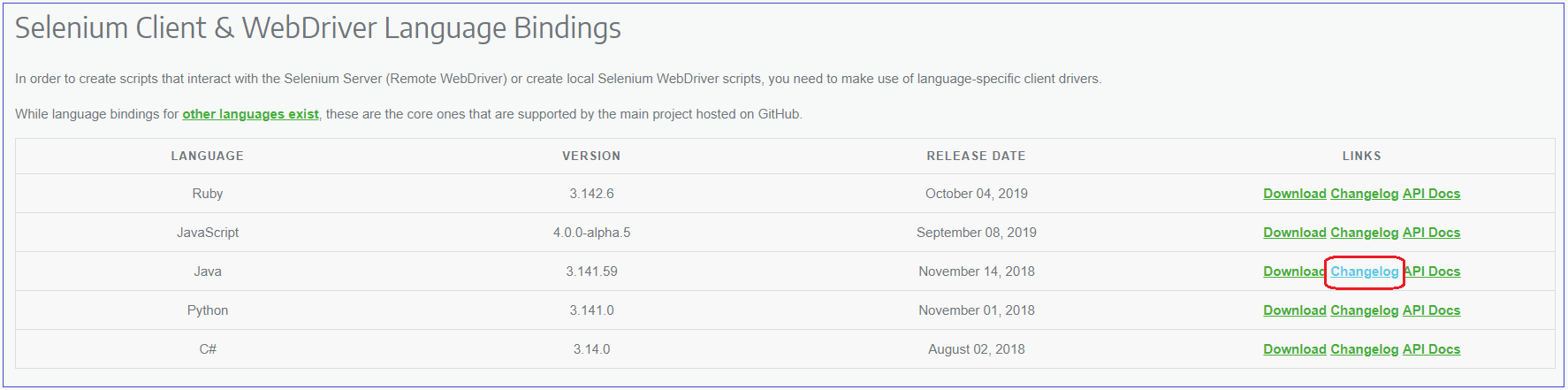
In the Introduction tutorial, we have covered [***Selenium 4.0 new features in Selenium IDE, WebDriver API, and Selenium Grid***](https://www.toolsqa.com/selenium-webdriver/selenium-4/). There, we discussed briefly *Selenium Relative locators*. Subsequently, in this tutorial, we will be covering this new locator in more detail.

In test automation, one of the critical tasks while writing a test script for web UI automation is to identify. That is to say, to locate the elements on the Web page. Selenium Web Driver already provides multiple ways of locating the web element like by id of the element or name of the element etc. Additionally, if you want to know more about existing locators, please refer to the[***locators***](https://www.toolsqa.com/selenium-webdriver/locators/) tutorial.

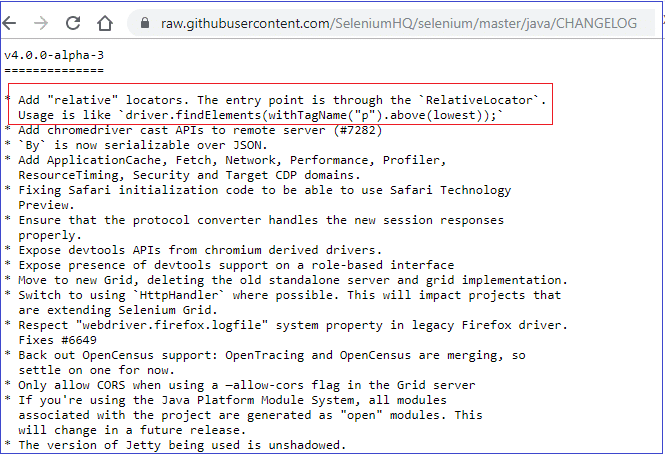
In Selenium  4.0, one more new locator is there, i.e., Friendly or Relative locators. Earlier, its name was Friendly locator. However, later it has been renamed as a Relative locator.

This locator helps to locate the elements concerning a given element. *For example, find an element which is* ***toTheLeftOf****a given element. Or many be toTheRightOf that element. Let us see more if it below.*

Moreover, if clicked on the [***Changelog***](https://raw.githubusercontent.com/SeleniumHQ/selenium/master/java/CHANGELOG) link for Java language on[***https://selenium.dev/downloads/***](https://selenium.dev/downloads/) page, *changelog* page gets displayed.



Consequently, as seen in the screenshot below, the changelog mentions about Relative locator under ‘**v4.0.0-alpha-3.’**



***Why New Locator?***

As we know, Selenium already offers eight locators to locate the web element by its id, name, etc. Then why this new locator is required?

An answer to this is, until now, one can locate an element by its attributes like id, name, or XPath, but not by its relative position with other elements. Therefore, Relative Locator provides the way to locate web element by its position concerning other web elements on the page

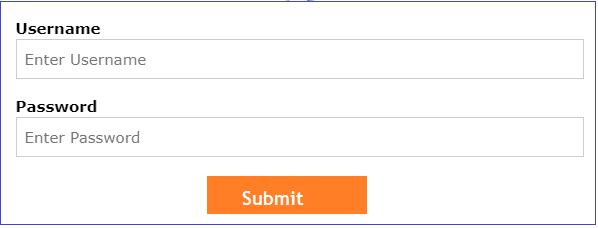
This class provides various methods to consume to locate elements as per the position like

* ***above***
* ***below***
* ***toRightOf***
* ***toLeftOf***
* ***near***

**Use of Relative Locator**

As this is Selenium 4 feature, the build path for the project needs to upgrade to Selenium 4. For this, please refer to the Introduction tutorial.

Let us understand it with the help of the same example seen in the Introduction tutorial. ***For Example****, usually, the following fields are displayed in any Login page:*

So, on the above page, if one has to locate the password text field, use the ***RelativeLocator*** class method. Here is the syntax of RelativeLocator methods :

***driver.findElement(RelativeLocator.withTagName(<tagname>).<method name>(<specified element>));***

* *Here,* ***tagname*** *is the name of the Html tag of the element like input, p, etc. Moreover, in this login page, the* ***tag name*** *is* ***input*** *as the Html tag used for the password field is input.*
* ***method name*** *is the name of the available methods indicating the position of the element to be located concerning specified elements like* ***above****,* ***below****,* ***toRightOf****,* ***toLeftOf****, etc. As seen in the screenshot, the password field is below the* ***Username.*** *Therefore****,*** *the* ***method name*** *to be used is* ***below.***
* ***specified element*** *is the element that one already identifies with existing locators like id, name, etc. And we are trying to locate the web element having a position concerning this specified element. In this example, the* ***specified element*** *is* ***usernameField,*** *which needs to be located by the existing locator strategy, i.e.,* ***By.id****. It could be any locator like name, link, etc.*

So, in simple words, the user is trying to locate the web element ‘**below’** ‘***usernameField’*** having tag name ‘***input.***’

***So code to locate password looks like below :***

*//Locate the specified element*  
*WebElement usernameField = driver.findElement(By.id(“username”));*

*//Locate the element using RelativeLocator*  
WebElement passwordField = driver.findElement(RelativeLocator.withTagName(“input”).below(usernameField));

***RelativeLocator class methods:***

Let’s see methods available in ***RelativeLocator*** class, note that these methods have overloaded methods versions as well:

***methodName(WebElement element):***

In this method, a specified web element reference is passed to the method. For Example, in the above login form example,

*WebElement* ***usernameField*** *= driver.findElement(By.id(“username”));*

*WebElement passwordField = driver.findElement(RelativeLocator .****withTagName****(“input”).****below(usernameField)****);*

Here, *methodName(WebElement element)* method is used. That is to say, specified web WebElement ***usernameField*** is getting passed to the ***below()*** method**.**

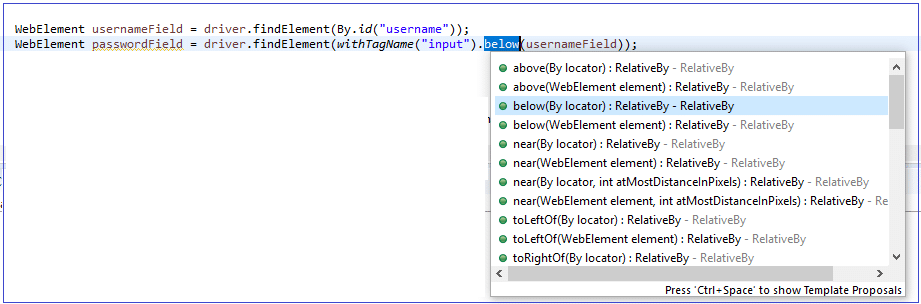
***methodName(By locator):***

In this overloaded version of the method, By method is passed to the method for the web element to be specified. We can write the same code using methodName(By locator) method as follows :

*WebElement passwordField = driver.findElement(RelativeLocator.* ***withTagName****(“input”).****below****(****By.id(“username”)****));*

***Different methods in Selenium Relative Locators***

Let’s see the methods available in **RelativeLocator** class:



1. *above(WebElement element): This method locates web elements appearing above the specified web element ‘element.’*

*above(By locator): This is an overloaded version of the above() method. Additionally, here web element is specified using By methods like By.id, By.name, etc.*

1. *below(WebElement element): This method locates web elements appearing below the specified web element ‘element.’*

*below(By locator): This is an overloaded version of the below() method. Moreover, here web element is specified using By methods like By.id, By.name, etc.*

1. *toLeftOf(WebElement element): This method locates web element appearing left side of the specified web element ‘element.’*

*toLeftOf(By locator): This is an overloaded version of the toLeftOf () method. Additionally, here web element is specified using By methods like By.id, By.name, etc.*

1. *toRightOf(WebElement element): This method locates web element appearing right side of the specified web element ‘element.’*

*toRightOf(By locator): This is an overloaded version of the toRightOf() method. Moreover, here web element is specified using By methods like By.id, By.name, etc.*

1. *near(WebElement element): To locate web elements appearing to near the specified web element ‘element.’ Here, near means the maximum of 50 pixels away from the specific element*

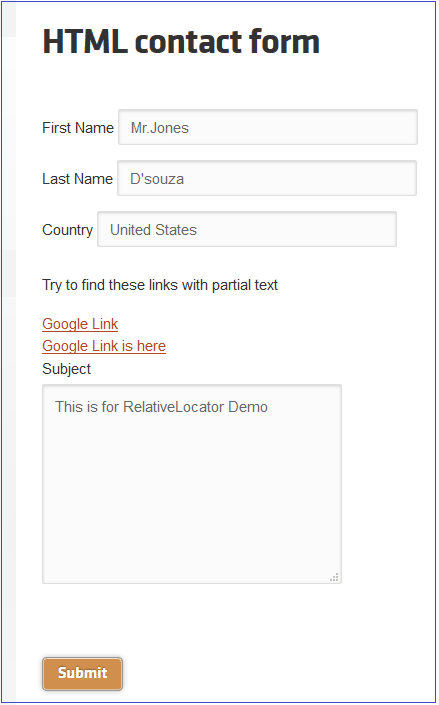
*near(By locator): This is an overloaded version of the near() method. Additionally, here web element is specified using By methods like By.id, By.name, etc.*

1. *near(WebElement element, int atMostDistanceInPixels): To locate web element appearing to near the specified web element ‘element’. Here, the specification of the maximum distance from the specific element is in ‘atMostDistanceInPixels.’*

*near(By locator, int atMostDistanceInPixels): This is an overloaded version of the near() method, here web element is specified using By methods like By.id, By.name, etc.*

**Practice Exercise to use Selenium RelativeLocators class methods**

Let’s take an example from an already available demo page on Toolsqa as[***https://demoqa.com/html-contact-form***](https://demoqa.com/html-contact-form)



Here, to fill up the contact form, the text needs to be entered in all the fields like First Name, Last Name, etc. Moreover, for that, these text fields need to identification or location first. Its achieved using existing locator strategies like By.id, By.name, etc.

Let’s understand how to use RelativeLocator class methods to identify these fields:

* Import RelativeLocator withTagName

*import static org.openqa.selenium.support.locators.RelativeLocator.withTagName;*

* Locate the web element with near to some nearby element on the page

*WebElement webElement = driver.findElement(withTagName(<tagname>).<methodname>(<specified element>));*

***Let us automate the following scenario:***

1. *Firstly, launch the web browser and launch our practice page* <https://demoqa.com/html-contact-form>
2. *Secondly, locate web element for the ‘First Name’ label.*
3. *Thirdly, locate the First Name text field which is to the right of ‘First Name’ label and enter the text.*
4. *After that, find the Last Name text field, which is below the ‘First Name’ text field and enter the text.*
5. *Locate the Country text field which is below the ‘Last Name’ text field and enter the text.*
6. *After that, Locate Subject textarea which is below the ‘Country’ text field and enter the text.*
7. *Find texts for all the labels above ‘Subject.’*
8. *After this, retrieve and print the text of the label above the Subject field and to the left of the Country field.*
9. *Then, click on the Submit button.*
10. *Lastly, close the browser to end the program.*



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60 | package com.toolsqa.tutorials.selenium4;    import static org.openqa.selenium.support.locators.RelativeLocator.withTagName;    import java.util.List;    import org.openqa.selenium.By;  import org.openqa.selenium.WebDriver;  import org.openqa.selenium.WebElement;  import org.openqa.selenium.firefox.FirefoxDriver;    public class RelativeLocatorsSample {  public static void main(String[] args) {     System.setProperty("webdriver.gecko.driver","C:\\Selenium\\lib\\geckodriver-v0.24.0-win64\\geckodriver.exe");     WebDriver driver = new FirefoxDriver();     //Launch the web browser and launch our practice page https://demoqa.com/html-contact-form     String URL = "https://demoqa.com/html-contact-form";     driver.get(URL);     //Locate web element for the ‘First Name’ label     WebElement nameLabel = driver.findElement(By.xpath(".//label[contains(text(),'First Name')]"));     //Locate First Name text field which is to the right of ‘First Name’ label and enter the text     WebElement nameText = driver.findElement(withTagName("input").toRightOf(nameLabel));//RelativeLocator.withTagName static method     nameText.sendKeys("Mr.Jones");       //Locate Last Name text field which is below the  ‘First Name’ text field and enter the text     WebElement lastName = driver.findElement(withTagName("input").below(nameText));     lastName.sendKeys("D'souza");     //Locate Country text field which is below the  ‘Last Name’  text field and enter the text     WebElement countryName = driver.findElement(withTagName("input").below(lastName));        countryName.sendKeys("United States");     //Locate Subject textarea which is below the  ‘Country’  text field and enter the text     WebElement subject = driver.findElement(withTagName("textarea").below(countryName));     subject.sendKeys("This is for RelativeLocator Demo");     //Find texts for all the labels above ‘Subject’     List<WebElement> labels = driver.findElements(withTagName("label").above(subject));     for(WebElement e : labels) {  System.out.println("labels-->"+e.getText());     }     //Retrieve and print  the text of the label above Subject field and to the left of Country field     //Note, here, element is located using the combination of above() and toLeftOf() methods     WebElement label = driver.findElement(withTagName("label").above(subject).toLeftOf(countryName));     System.out.println("Label for Country Textfield--->"+label.getText());     //Click on the Submit button     WebElement submit = driver.findElement(withTagName("input").below(subject));     submit.click();     //Close the browser to end the program     driver.close();  }  } |

***Note****: In the sample code above, one can use multiple methods in the same statement for locating the element. For example, in the above sample code, the label for the country is located using the combination of above() and toLeftOf() methods.*

***How does the Selenium Relative Locators work?***

As seen in above section, here is the syntax for using Relative Locators: **driver.findElement(RelativeLocator.withTagName(<tagname>).<methodname>(<specified element>));**

As noticed, here *withTagName* method is invoked on RelativeLocator class directly. It is because withTagName() is a static method. Therefore, the instantiation of ***RelativeLocator*** class is not required. If seen in the source code of ***RelativeLocator*** class, *withTagName* internally creates a new instance of ***RelativeBy***class and returns the same. Consequently, methods like above, below, etc. are invoked on the returned instance of *RelativeBy*.

To find the location of  the element, Selenium leverages the JavaScript function *getBoundingClientRect()*to find the relative elements. *getBoundingClientRect()* function returns properties of an element like left ,right, left, top ,bottom.

***Summary:***

To conclude, in this tutorial, we have covered the new class introduced in Selenium 4.0 WebDriver API, i.e., RelativeLocator. We included essential methods of RelativeLocator:

* *above(WebElement element): Locates web element appearing above the specified element*
* *below(WebElement element): Locates web element appearing above the specified element*
* *toLeftOf(WebElement element): Locates web element appearing to the left of the specified element*
* *toRightOf(WebElement element): Locates web element appearing to the right of the specified element*
* *near(WebElement element): Locates web element appearing to near(maximum 50 pixels away from) of the specified element.*