QA

Selenium

Chapter 05

Neler Öğreneceğiz?

- Window Handle
- Multiple Tabs
- Actions
- JS Executor
- Robot Class

Window Handle

Herhangi bir tarayıcıdaki bir window, kullanıcının bir bağlantıya/URL'ye tıkladıktan sonra geldiği **ana web sayfasıdır**.

Bir kullanıcı bir URL'ye ulaştığında bir web sayfası açılır. Bu ana sayfa, ana penceredir (**parent window**), yani kullanıcının şu anda açtığı ve herhangi bir işlemi gerçekleştireceği ana penceredir.
Bu, Selenium otomasyon komut dosyamız yürütüldüğünde açılacak olan web sayfasının aynısıdır.
Ana pencerenizin içinde açılacak olan tüm pencereler alt pencere (**child windows**) olarak adlandırılır.

Window Handle Methodları

```
driver.getWindowHandle();
```

```
driver.switchTo().window(switch);
```

```
driver.switchTo().newWindow(WindowType.TAB);
```

```
driver.switchTo().newWindow(WindowType.WINDOW);
```

```
driver.getWindowHandles();
```

Window Handle Methodları

driver.getWindowHandle(); → Mevcut sayfanın window handle değerini alır.

driver.switchTo().window(switch); → Window handle değerini kullanarak pencereler arası geçiş yapar.

driver.switchTo().newWindow(WindowType.TAB); → Yeni TAB oluşturarak geçiş yapar.

driver.switchTo().newWindow(WindowType.WINDOW); → Yeni WINDOW oluşturarak geçiş yapar.

driver.getWindowHandles(); → Tüm sayfaların window handle değerini alır.

Multiple Tabs

- **String mainwindow = driver.getWindowHandle():** Parent window değerini benzersiz bir dize türü tanımlayıcısında saklar.
- **Set <String> s = driver.getWindowHandles():** Tüm child window'ları String data type'ta bir sete atar.
- **Iterator <String> itr = s.iterator():** Burada tüm child window'lar yinelenir.
- **if (!mainWindow.equalsIgnoreCase(ChildWindow)):** Parent window ile child window karşılaştırılır.
- **driver.switchTo().window (ChildWindow):** Child window'a geçer ve title okur.

Multiple Tabs – Window Handle Selenium 4

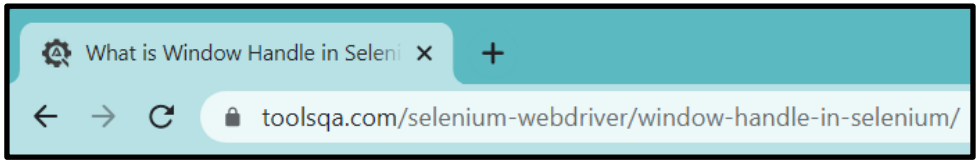
```
WebDriver newWindow = driver.switchTo().newWindow(WindowType.WINDOW);
newWindow.get(URL);
```

New Window

```
WebDriver newTab = driver.switchTo().newWindow(WindowType.TAB);
newTab.get(URL);
```

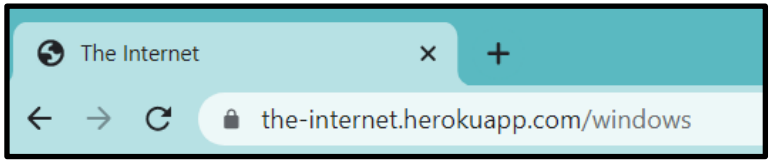
New TAB

Window Handle

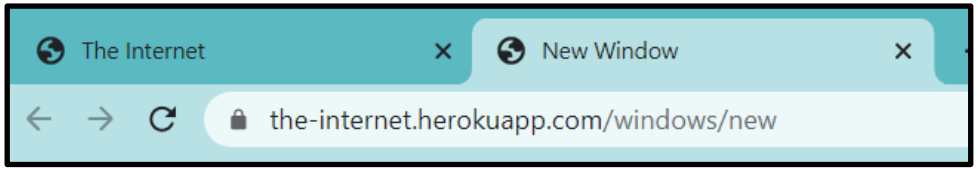


Go to URL: https://www.toolsqa.com/selenium-webdriver/window-handle-in-selenium/
Print the existing windowHandles ids by clicking all the links on the page.
Click on the links that open a new page.
Close other pages other than the home page.
Set the driver back to the main page.

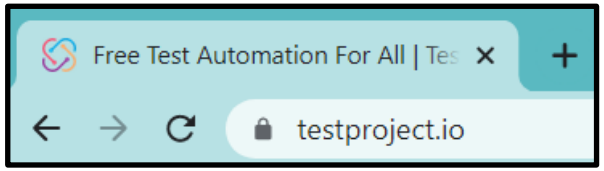
Window Handle



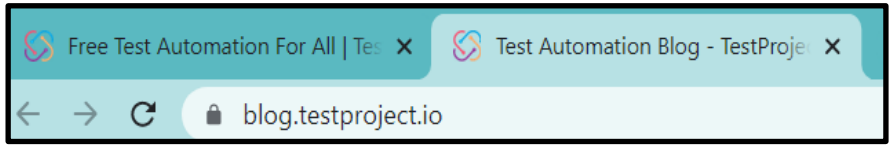
Go to URL: https://the-internet.herokuapp.com/windows
Verify the text: “Opening a new window”
Verify the title of the page is “The Internet”
Click on the “Click Here” button
Verify the new window title is “New Window”
Go back to the previous window and then verify the title: “The Internet”



New Window Handle

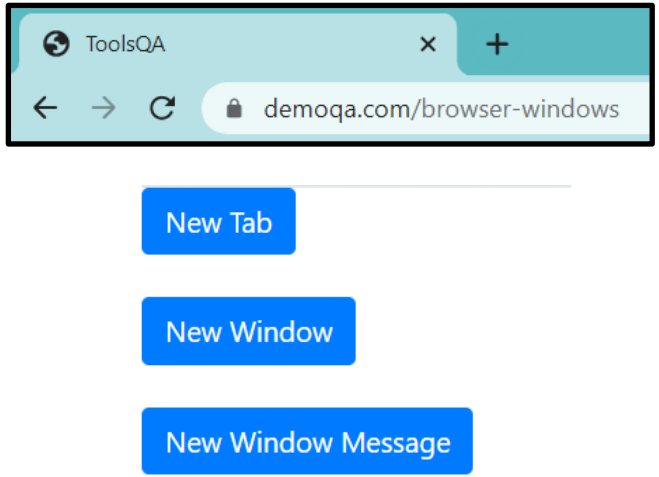


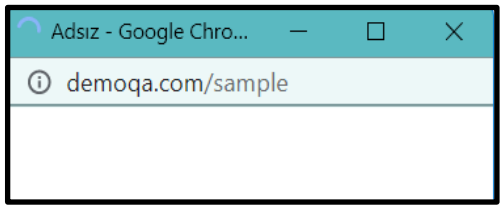
Go to URL: https://testproject.io/
Selenium's 4 newWindow() method to open a new Window for TestProject's Blog page.
https://blog.testproject.io/

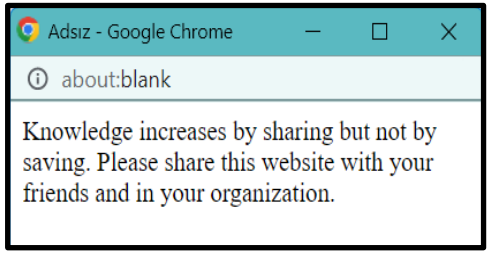


Window Handle

Go to URL: https://demoqa.com/browser-windows
Click on the windows - "WindowButton"
Click on all the child windows.
Print the text present on all the child windows in the console.
Print the heading of the parent window in the console.







Review

iFrame: iFrame etiketi kullanılarak oluşturulur. Sayfa içinde sayfa - iframe'i -index, id/class, WebElement arasında geçiş yaparak ele alınır. - **driver.switchto().frame (3 Seçenek)**

Alerts: JavaScript pop-up'larıdır. Teste devam etmek için uyarılar kabul edilmeli veya iptal edilmelidir.
accept()-clicking ok **/** dismiss()-clicking cancel **/** getText () – Alertten text alma
sendKeys()-alerte text gönderme - **driver.switchTo().alerts() (4 Seçenek)**

Multiple Tabs/Windows: Bazı sayfalar yeni pencereler açar, sonrasında 2 pencere ile karşılaşırız.
getWindowHandle(): Sayfanın window handle değerini String olarak döndürür.
getWindowHandles(): Tüm açık sayfaların window handle değerini Set<String> olarak döndürür.
driver.switchTo().window(Geçilmek istenen sayfanın WH değeri**);**

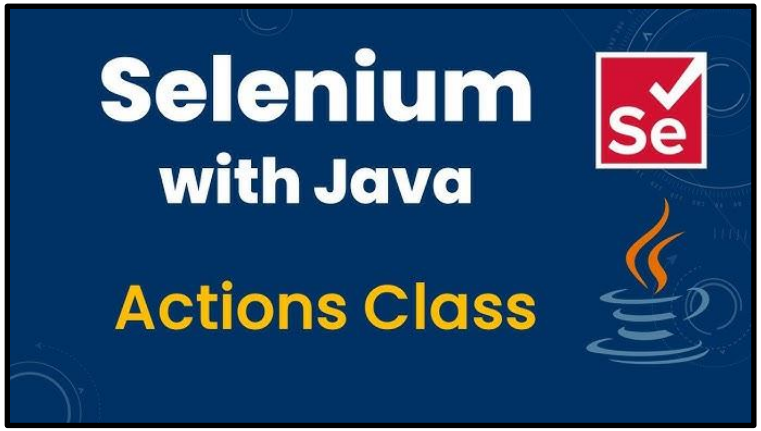
Actions

- Kullanıcı web'i keşfederken herhangi bir butona tıklama, metin girme, çift tıklama, sağ tıklama, sürükle-bırak, açılır menüden seçim yapma, yeniden boyutlandırma vb. gibi çeşitli işlemler gerçekleştirir.

- Bu eylemler web uygulamasında **Actions Class** kullanılarak gerçekleştirilir ve yine bu eylemle Selenium Actions kullanılarak otomatikleştirilir.

- Actions sınıfını kullanarak önce bir dizi bileşik olay oluşturur ve ardından bunu **Action** (tek bir kullanıcı etkileşimini temsil eden bir arabirim- **interface**) kullanarak gerçekleştiririz.

Selenium, web uygulamalarını otomatikleştirmek için en iyi test araçlarından biri olarak kabul edilir. Klavye ve fare ile ilgili her türlü eylemi desteklemek için yerleşik özelliklere sahip güçlü bir araçtır.



Actions

Actions Class & Action Class Farkı

Actions'ın oluşturucu tasarım modeline dayalı bir sınıf olduğu sonucuna varabiliriz. Bu, karmaşık kullanıcı hareketlerini taklit etmek için kullanıcıya yönelik bir API'dir.

Action ise tek bir kullanıcı etkileşimi eylemini temsil eden bir interface-arayüzdür. En yaygın olarak kullanılan **perform()** yöntemlerinden birini içerir.

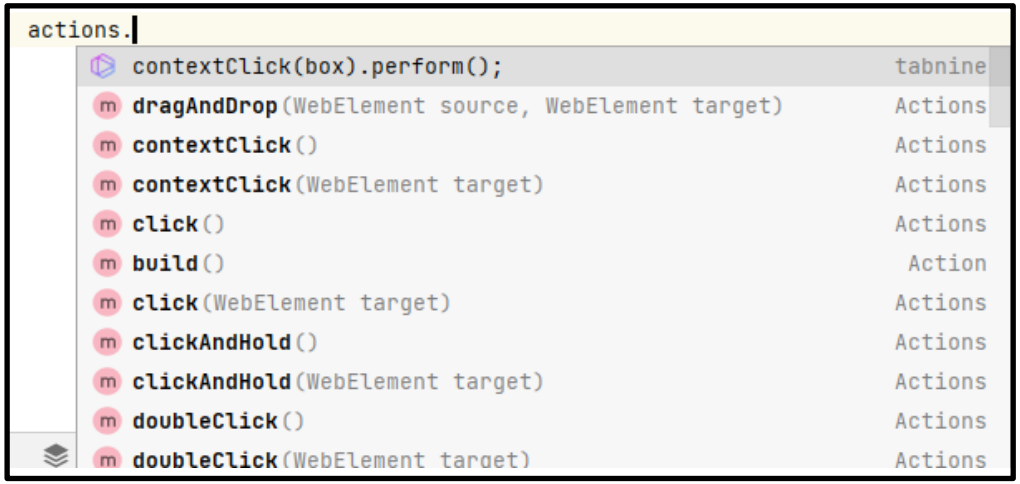
"Generates a composite action containing all actions."

Action is an interface representing 1 user-interaction. It only has 1 method and that method is perform. The Actions class has a lot of methods like build, dragAndDrop, dragAndDropBy, keyDown, keyUp, moveToElement, and it also has perform. This perform() method is different from the one in the Action Interface.

Actions

Actions actions = new Actions(driver); — Create action object.
actions.contextClick(box).perform(); — Perform the action.
perform(): Bir eylemi yürütür. Bu, bir eylemi gerçekleştirmek için sonunda kullanılmalıdır.

perform() KULLANMAYI UNUTMAYIN!



Actions

Mouse Actions

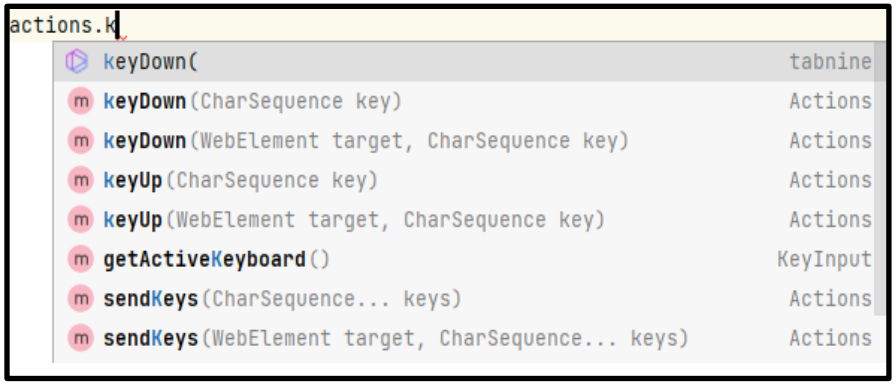
click(): Geçerli konumu tıklamak için kullanılır.
doubleClick(): Fare konumuna çift tıklama gerçekleştirmek için kullanılır.
clickAndHold(): Fare tıklamasını serbest bırakmadan gerçekleştirmek için kullanılır.
contextClick(): Geçerli fare konumuna sağ fare tıklaması gerçekleştirmek için kullanılır.
moveToElement (WebElement target): Fare işaretçisini hedef konumun merkezine taşımak için kullanılır.
dragAndDrop(WebElement source, WebElement target): Öğeyi kaynaktan sürüklemek ve hedef konuma bırakmak için kullanılır.
dragAndDropBy(source, xOffset, yOffset): Geçerli konumu tıklayıp basılı tutmak için verilen ofset değeri ve ardından fareyi bırakmak için kullanılır.
(X = Yatay Kaydır, Y= Dikey Kaydır)
release(): Geçerli konumdaki sol fare düğmesini serbest bırakmak için kullanılır.

Actions

Keyboard Actions

sendKeys (): Metin kutusuna, text yazmak için kullanılır.
keyDown (): Bir tuşu basılı tutmak için kullanılır. Tuşlar Shift, Ctrl ve Alt anlamına gelir.
keyUp (): keyDown() yönteminden sonra zaten basılmış olan bir tuşu serbest bırakmak için kullanılır yani hedefe odaklandıktan sonra tuşu serbest bırakır.

Keyboard actions 2 parametre alır.

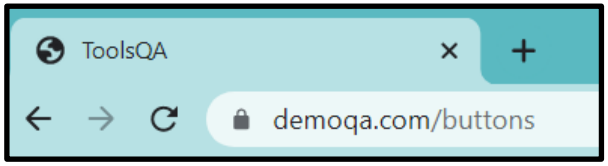


Actions

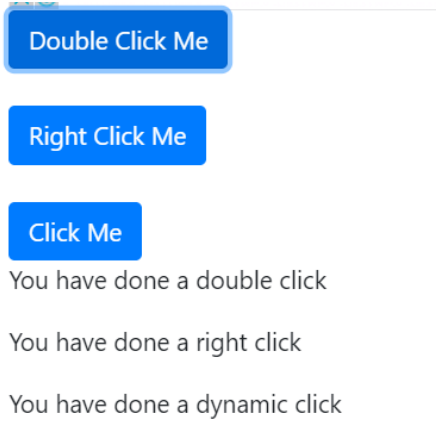
Actions sınıfını yürütmek ve derlemek için build.perform() yöntemi de kullanılır.

action.moveToElement(element).build().perform();
action.moveToElement(element).perform(); → Direkt kodu execute eder.

Action Class



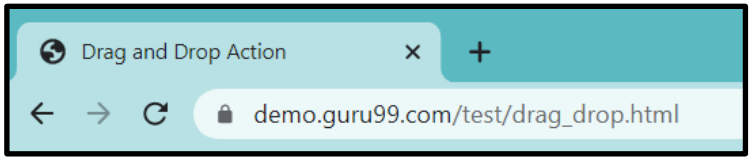
Go to URL: https://demoqa.com/buttons
Run the buttons on the page using the Actions Class.
Verify the texts that appear after the buttons are operated.



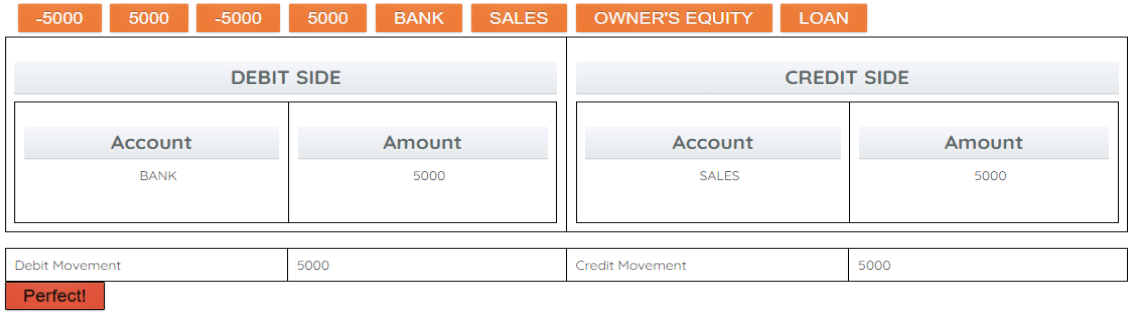
Drag and Drop

Go to URL: http://www.dhtmlgoodies.com/scripts/drag-drop-custom/demo-drag-drop-3.html
Fill in capitals by country.

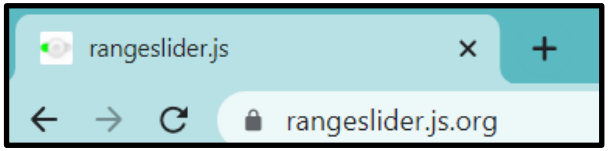
Drag and Drop



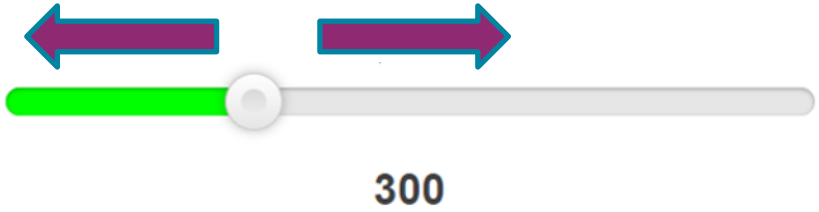
Go to URL: http://demo.guru99.com/test/drag_drop.html
Drag and drop the BANK button to the Account section in DEBIT SIDE
Drag and drop the SALES button to the Account section in CREDIT SIDE
Drag and drop the 5000 button to the Amount section in DEBIT SIDE
Drag and drop the second 5000 button to the Amount section in CREDIT SIDE
Verify the visibility of Perfect text.



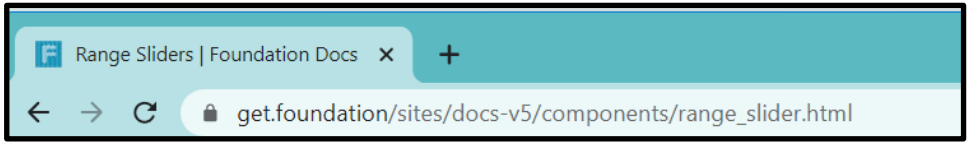
Drag and Drop by Horizontal



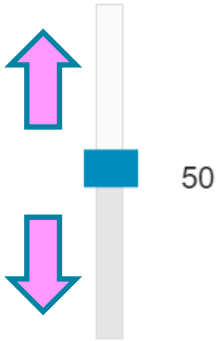
Go to URL: https://rangeslider.js.org/
Shift 100 units to the right and 100 units to the left on the horizontal axis.



Drag and Drop by Vertical

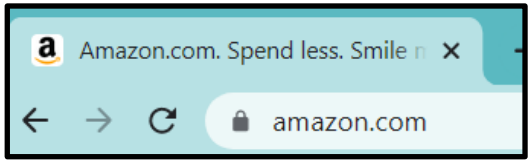


Go to URL: https://foundation.zurb.com/sites/docs/v/5.5.3/components/range_slider.html
Shift 34 units to the right and 34 units to the left on the vertical axis.

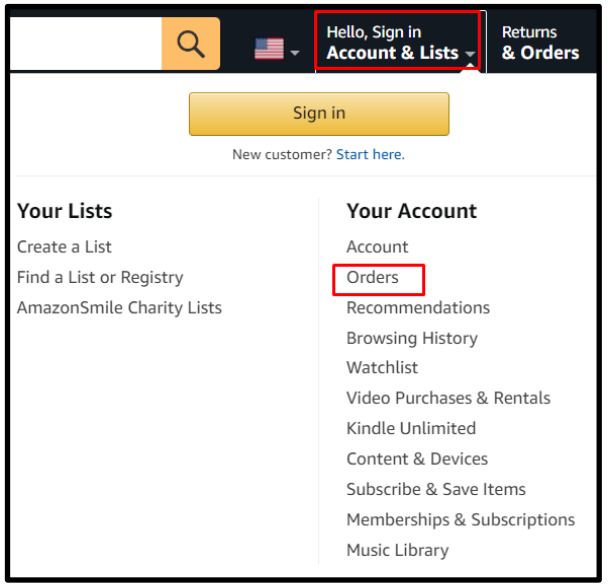


TIME FOR TASK

Hoverover



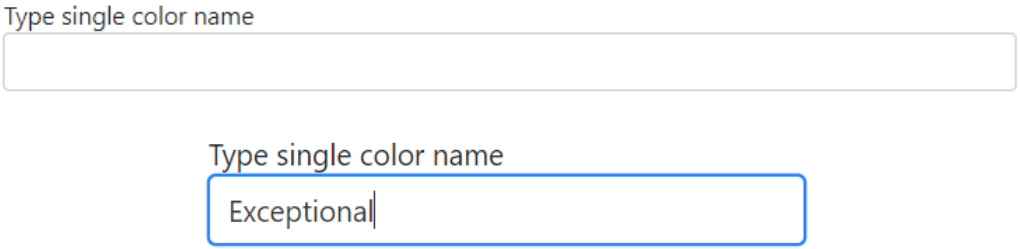
Go to URL: https://www.amazon.com/
Click on “Hello, Sign in Account & Lists” link.
Click on Orders page.
Verify the page title contains “Amazon”.



Keyboard

TIME FOR TASK

Go to URL: https://demoqa.com/auto-complete
In the Type single color name section, print "You are Exceptinal" by using the action methods.



JS Executor

JavaScript Executor’a neden ihtiyacımız var?

Selenium Webdriver'da, bir web sayfasındaki işlemleri tanımlamak ve gerçekleştirmek için XPath, CSS vb. gibi konum belirleyiciler kullanılır.

Bu konum belirleyicilerin çalışmaması durumunda JavaScriptExecutor'ı kullanabilirsiniz. Bir web öğesinde istenen bir işlemi gerçekleştirmek için JavaScriptExecutor'ı kullanabilirsiniz.

Selenium, javaScriptExecutor'ı destekler. Ekstra bir eklentiye gerek yoktur. JavaScriptExecutor kullanmak için komut dosyasında (org.openqa.selenium.JavascriptExecutor) içe aktarmanız yeterlidir.

JavaScript HTML kodlara direk erişip yönetebilen bir script dili olduğundan bize çok fazla kolaylık sağlayabilir.

JS Executor

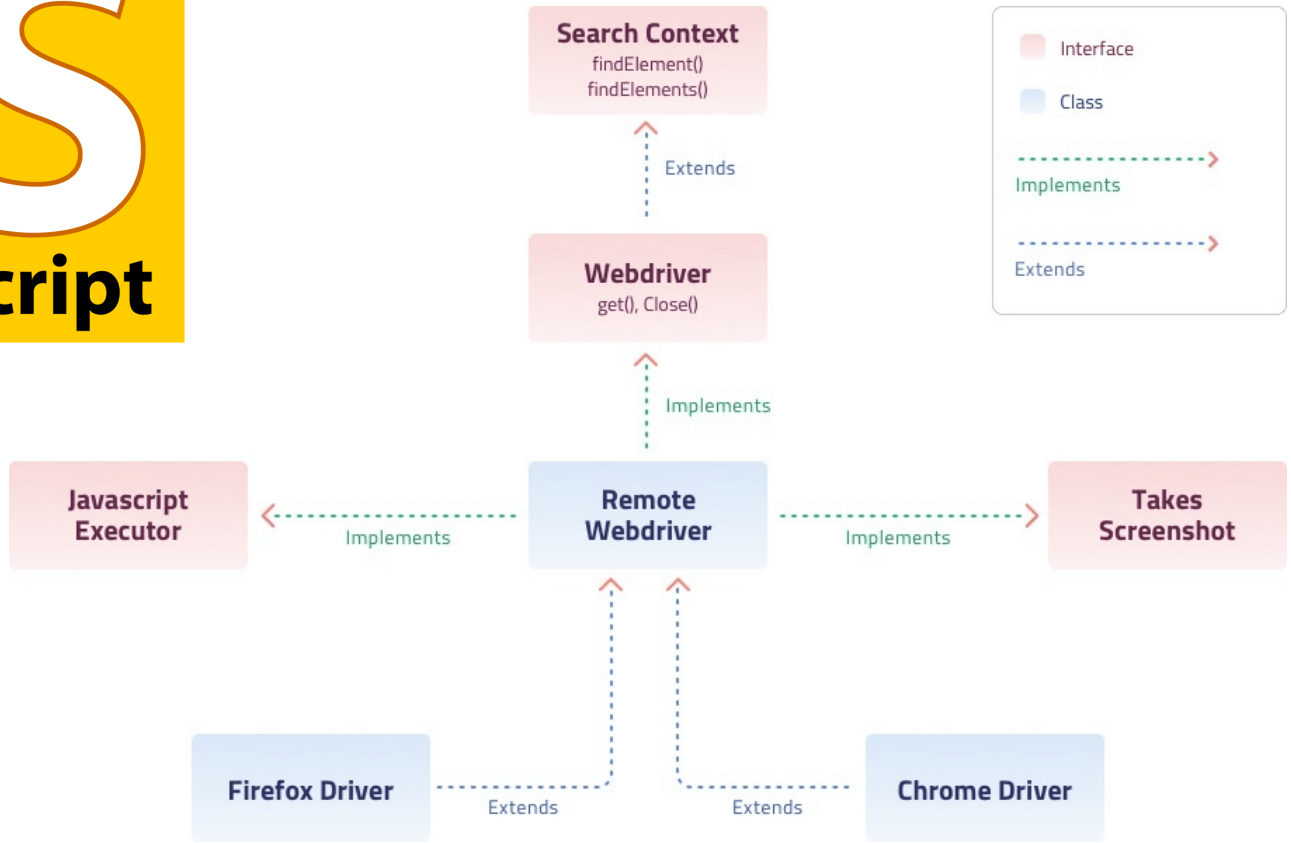
JavascriptExecutor js = (JavascriptExecutor)*driver*;
JavascriptExecutor kullanmak için ilk adım olarak driver'ı JavascriptExecutor interface'sine cast etmektir.
Bu interface sayesinde sayfa kaydırma işlemi ve JavaScript komutları çalıştırılabilir
En yaygın kullanılan methodu, **executescript() methodudur.**

executeScript - ScrollBy () web sayfasını verilen piksel değeri kadar ileri gider.
executeScript("window.scrollBy(x-piksel,y-piksel"); ya da js.executeScript("scroll(x,y);");
x-piksel x eksenindeki sayıdır, sayı pozitifse sola, sayı negatifse sağa hareket eder.
y-piksel y eksenindeki sayıdır, sayı pozitifse ise aşağı doğru,sayı negatif ise yukarı doğru hareket eder.
js.executeScript("window.scrollBy(0,1000)); → Dikey olarak 1000 piksel aşağı kaydırır.

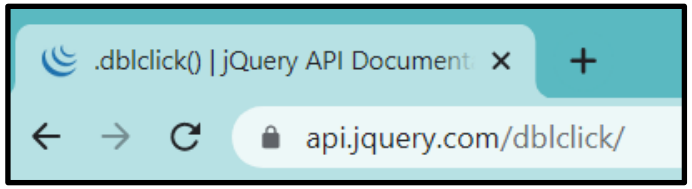
executeScript - ScrollTo () web sayfasını verilen piksel değerine götürür.
js.executeScript("window.scrollTo(0, 0)");
js.executeScript("window.scrollTo(0, document.body.scrollHeight)");

scrollIntoView () web sayfasındaki bir öğenin görünürlüğüne göre kaydırır.
js.executeScript("arguments[0].scrollIntoView();",WebElement);

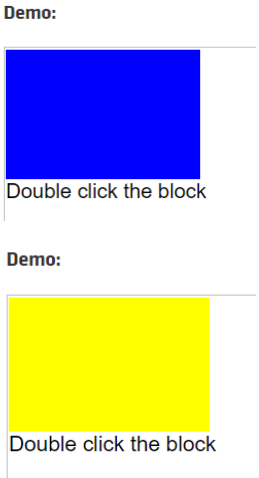
JS Executor


Search Context
findElement()
findElements()
Extends
Webdriver
get(), Close()
Implements
Javascript Executor
Implements
Remote Webdriver
Implements
Takes Screenshot
Firefox Driver
Extends
Extends
Chrome Driver
Interface
Class
Implements
Extends

JS Executor



Go to URL: https://api.jquery.com/dblclick/
Double click on the blue square at the bottom of the page and then write the changed color.



Robot Class

Robot Sınıfı: Selenium komut dosyalarında, tarayıcı ve masaüstü açılır pencerelerini otomatikleştirmek için Robot classını kullanırız .

Genellikle **tarayıcılara/tarayıcılardan dosya yükleme/indirme işlemlerinde** kullanılır. **Fare ve klavye işlemlerini** gerçekleştirmek için Robot classını kullanıyoruz. Robot class otomasyon süreci ile kullanımı çok kolaydır. Java otomasyon frameworkleri ile kolayca entegre edilebilir.

Selenium, pencere tabanlı açılır pencereleri (açılır pencereleri indir, açılır pencereleri yükle gibi) işlemek için destek sağlamaz. Windows iletişim kutusuyla etkileşim, Selenium'da bir sınırlamadır. Bir web öğesi üzerinde herhangi bir işlem gerçekleştirmek için öğe için bir konumlandırıcıya ihtiyacımız vardır. Ancak Windows açılır pencereleri, web sayfasının bir parçası olmadıkları için herhangi bir konum belirleyiciye sahip değildir, bunlar yerel işletim sistemi açılır pencereleridir. İşte bu tür açılır pencereleri işlemek için **Robot Class kavramını kullanabiliriz.**

```
Robot rb = yeni Robot();  rb.<gerekli_yöntem>();
```

Robot Class

Action Class ?

Robot Class ?

Actions class bir fareyi simüle eder. Fare imlecini hareket ettirmez.
Oysa Robot class, Selenium'un gerçek bir fare kullanmasını sağlar.

Selenium Robot Class Methodları

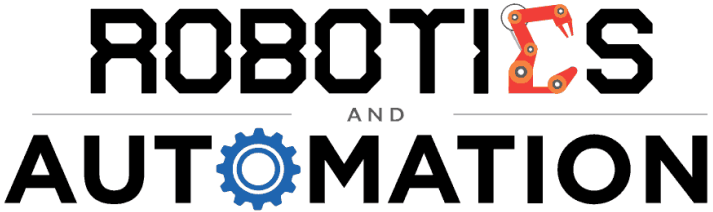
- ***keyPress()***: Belirli bir tuşa basar.
- ***keyRelease()***: Belirli bir klavye tuşunu serbest bırakır.
- ***mousePress()***: Girilen değere göre fare düğmesine basar.
- ***mousePress(1)***: Birincil tuşa basar.
- ***mousePress(2)***: İkincil tuşa basar.
- ***mouseRelease()***: Fare düğmesini serbest bırakır.
- ***mouseMove()***: Fare işaretçisini verilen ekran koordinatlarına taşır.
- ***mouseWheel()***: Verilen değer negatif ise aşağı kaydırılır, pozitif değer çarkı yukarı kaydırılırsa fareyi kaydırır.

Otomasyonda Robot Class Dezantajları

- Pop-up'lara izin verilmezse veya istemci bilgisayarda dosya indirme yetkisi verilmezse robot sınıfı başarısız olabilir.

- Robot sınıfı Gerçek fare komutlarını yürüttüğü için paralel çalıştırmadan kaçınılmalıdır, bu nedenle bir bilgisayarda iki fare olamaz.

- keyPress olayını kullandığınızda keyRelease olayını da kullanmalısınız. keyRelease'i kullanamıyorsanız, basılı kalacak ve arka planda bellek tüketecektir. **Bu en büyük dezavantajdır.**



THANKS!

Any questions?

Elly D. - Full-Stack Automation Engineer

Garry T. - Full-Stack Automation Engineer