Normally in Selenium WebDriver we execute only Java methods. But sometimes we need to execute javaScript statements in our selenium programm. By using executeScript() method we can execute javaScript in Selenium.

JavaScriptExecutor - это интерфейс у него есть один метод executeScript()

Когда в Selenium мы вызываем метод и передаём в него какието параметры, дак вот этот метод в своей реализации выполнен на JavaScript-е

Element Entersapted Exception - это когда мы используем какой либо метод из Selenium, этот метод соответственно идёт в свою реализацию и там выполняется т.е. происходит взаимодействие с веб страницей (посредством javaScript). Но иногда возникает задержка между методом который мы вызвали и его реализацией, т.е. имплементация не может осуществить взаимодействие с веб страницей. Поэтому не смотря на то что метод корректный и с локаторами всё в порядке мы всё равно получим исключение. Эта ошибка не решается time synchronization потому что метод даже не доходит до веб страницы. Решением является не вызывать метод в Selenium-е а напрямую идти в реализацию написанную на javaScript-e. So it's kind of bypassing the Selenium command. **Когда получаешь Entersapted Exception надо использовать javaScriptExecutor**

Есть 2 разновидности имплементации javaScriptExecutor-a

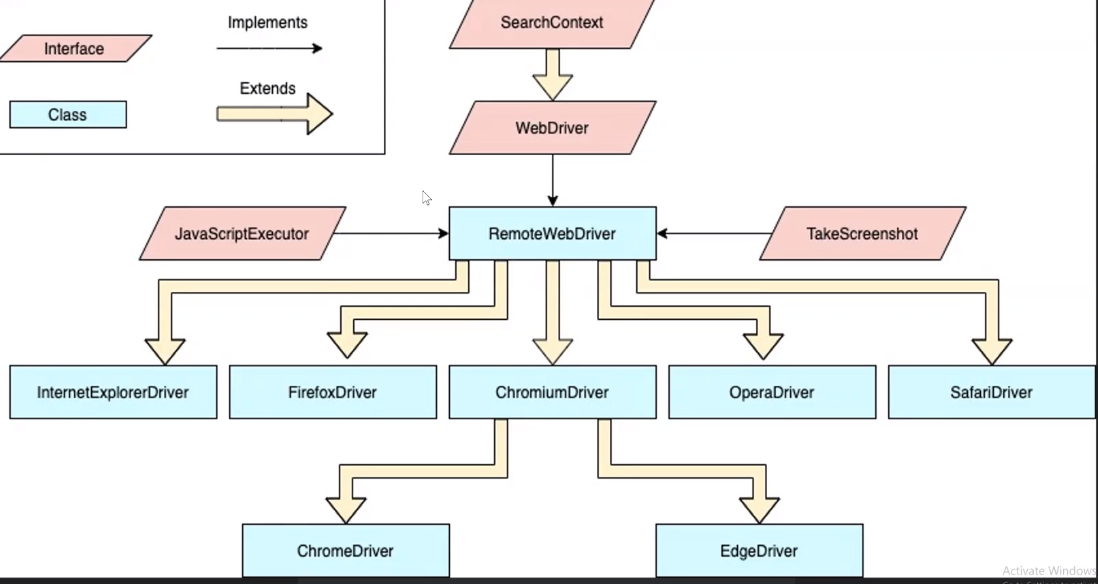
WebDriver driver = new ChromeDriver();

javaScriptExecutor js = (javaScriptExecutor )driver ← с приведением типов

ChromeDriver driver = new ChromeDriver()

javaScriptExecutor js = driver

Почему так происходит потому что WebDriver интерфейс не знает ничего про javaScriptExecutor интерфейс. Но у ChromeDriver-а родитель класс RemoteWebDriver реализует все 3 интерфейса поэтому приведение не нужно.



**How to handle input webElement with javaScript**

JavascriptExecutor javascriptExecutor = driver;

**javascriptExecutor.executeScript(“arguments[0].setAttribute(‘value’, ‘John Travolta’)”, webelement)**

The first parameter we have to define java script statement.

**arguments[0]** - это наш webElement, т.е. это что то типа массива, так как у нас один webElement, а намерация наченается с 0 то мы и передаём этот элемент. Наверно можно иметь массив элементов и просто указывать индекс того элемента который должен быть задействован.

**setAttribute(‘value’, ‘John Travolta’)** - мы сетим(или добавляем) атрибут ‘value’ оно = ‘John Travolta’ для webelement-а

**Handle clickable element (checkBox, radioButton)**

**javascriptExecutor.executeScript("arguments[0].click()",element);**

Тут такая же логика как и в примере выше, только метод заточен под клик.

Чтобы проскролить страницу нужно использовать javaScriptExecutor так как скроллинг опция (вертикальная) не является веб элементом

—---------------------------------------**навести страницу браузера на элемент**—---------------------------

**javascriptExecutor.executeScript("arguments[0].scrollIntoView()",element);**

← наводит страницу браузера на указанный элемент (элемент должен быть загружен на странице т.е. работает при условии что веб страница загружается целиком). Передаём элемент (arguments[0]), вызываем метод scrollIntoView().

-------------------**проскролить вниз (загрузить) динамически расширяемую страницу**—--------------

1)Создать коллекцию, залить туда все присутствующие элементы на странице (те, число которых, увеличивается при скроллинге вниз)

2)создать бесконечный цикл, в нём выполнить пиксельный скроллинг, например:

**((JavascriptExecutor) driver).executeScript("javascript:window.scrollBy(0,2500)")**; ВАЖНО доскролить до момента когда будут подгружаться новые элементы.

3)Остановить поток или сделать ожидание, с целью чтобы всё загрузилось на странице, иначе упадёт.

4)Создать новую коллекцию прямо в цикле (она нужна новая/пустая на каждой итерации) залить в неё все присутствующие элементы на стронице. Так как была загрузка новых элементов, то ожидаем, что в новой коллекции элементов больше чем в старой. Если это true то значит мы не дошли до конца страницы, переписываем элементы из новой коллекции в старую и повторяем.

5)Когда коллекции по количеству элементов будут равны значит мы проскролили до конца, break;

Использовал этот приём в ботфаиндере

—-------------------------------------------**вернутся на самый верх страницы**—-----------------------------------------

(JavascriptExecutor) driver).executeScript("window.scrollTo(0, -document.body.scrollHeight)"); ←сверху до самого низа

**скроллинг путём нажатия клавиш на клавиатуре**

**driver.findElement(By.cssSelector("body")).sendKeys(Keys.CONTROL, Keys.END);** ← (имитация с помощью класса Keys)

**Проскролить передав количество пикселей**

Оба варианта работает одинаково хорошо

**javascriptExecutor.executeScript("javascript:window.scrollBy(0,500)");**

**javascriptExecutor.executeScript("window.scrollBy(0,1500)","");**

**javascriptExecutor.executeScript("return window.pageYOffset;”); ←**вернёт текущее количество пикселей на странице (на каких пикселях находится страница)