**Actions** class is an ability provided by Selenium for handling keyboard and mouse events.

In Selenium WebDriver, handling these events includes operations such as drag and drop, clicking on multiple

elements with the control key and so on…, нажать на элемент правой кнопкой мыши

**Keys** class - represents all keyboard keys. Там много констант **Keys.выбери\_константу**

1) **Implementation of Action Class**

WebDriver driver = new ChromeDriver();

Actions act = new Actions(driver);

2) **Нависание мышкой над элементом**

Actions act = new Actions(driver);

firstElement = driver.findElement(By.xpath("//span[contains(text(),'Услуги')]"));

actions.moveToElement(firstElement).build().perform();

---------------------------------------------------------------------------------------------------------------

3) **Двойной щелчок мыши по выбранному элементу**

WebElement element = driver.findElement(By.xpath("//\*[@id=\"nav-xshop\"]/a[1]"));

Actions action = new Actions(driver);

action.doubleClick(element).build().perform();

**.build(); .perform();**

Есть класс Actions и есть интерфейс Action. В чём отличие?

Можно сначала построить какую то команду используя метод build(), но не выполнять её (такие варианты могут быть), далее можно сохранить эту команду в переменной, и тип переменной должен быть Action но не класс Actions а интерфейс Action. Когда потребуется выполнить команду то берём эту переменную и вызываем на ней метод perform().

Например:

Actions act = new Actions (driver) ← создание объекта от класса Actions

Action delateAction = act.contextClick(webElement).build; ← creating double click команду и записываем в переменную

delateAction.perform(); ← выполнение ранее построенной команды.

Но если выполнить построенну команду требуется в моменте то используют perform() без build()

.build()-отвечает за то чтобы построить сочетание применяющихся методов/метода, для команды

.perform()- выполнение того, что построил build

4) **Нажать правой клавишей мыши по элементу**

Actions act = new Actions(driver);

firstElement = driver.findElement(By.xpath("//div[@class='break']"));

actions.contextClick(firstElement).perform();

-) **Получить значение из определённого атрибута в элименте**

WebElement element = driver.findElement(By.id("SubmitButton"));

String attValue = element.getAttribute("id");

вернёт значение того атрибута который ты указал. Другими словами это когда надо получить не статический текст, типа getText(), а именно значение атрибута.

5) **реализация drag and drop**

WebElement source = driver.findElement(By.xpath("//\*[@id=\"drag1\"]"));

WebElement destination = driver.findElement(By.xpath("//\*[@id=\"div2\"]"));

action.clickAndHold(source).moveToElement(destination).release().build().perform();

**OR**

Actions act = new Actions(driver);

WebElement source = driver…

WebElement destination = driver…

act.**dragAndDrop**(source, destination).perform();

**OR**

Sometimes a web page has specific elements called Sliders. In that case we do not have a destination element. The element is movable only in ‘x’ directory (x=horizontally, y=vertically).

**1**) Capture the element

**2**)Get to know the current location of the element that you want to move, by using the **webElement.getLocation()** method. It will return x and y current parameters of the element. For example we got (x=56, y=110), this current position

**3**) use method **action.drugAndDropBy( x, y)** where x, y is position of the element where we want to move element, since we move only horizontally we need to change only ‘x’. For example action.drugAndDropBy( 156, 110)

If the slider is vertical then use the same method, but move only in ‘y’ direction’.



6) **//holds the SHIFT key and converts the text to uppercase**

action.keyDown(element,Keys.SHIFT).sendKeys("lambdatest").build().perform();

**// Scroll Down using Actions class**

act.keyDown(Keys.CONTROL).sendKeys(Keys.END).perform();

**// Scroll Up using Actions class**

act.keyDown(Keys.CONTROL).sendKeys(Keys.HOME).perform();

---------------------------------------------------------------------------------------------------------------

7)**Метод .sendKeys()- можно отсылать какуюто инф. в input например, а можно имитировать нажатия кнопки какой-то**

action.sendKeys(element, "iphone").build().perform();

action.sendKeys(Keys.ENTER).build().perform(); <-имитирует нажатие клавиши Enter

---------------------------------------------------------------------------------------------------------------

**8) Copy & Paste?**

а) - Сначала надо найти инпут элемент и ввести какой то текст, его будем копировать и вставлять в другое место

**driver.fidnElement(By.xpath(“...”)).sendKeys(“Text that you want to write”);**

**Actions actions = new Actions(driver);**

б) - чтобы не выделять мышкой весь текст(что трудно) мы с имитируем нажатие ctrl+a, но надо понимать что мы с имитируем нажатие кнопок и их отпускание (key down and key up) это важно!

**actions.keyDown(Keys.CONTROL).sendKeys("a").keyUp(Keys.CONTROL).perform();** ← выделить текст, текст выделится там где мы его вводили, так как драйвер сфокусирован на том элементе

**actions.keyDown(Keys.CONTROL).sendKeys("c").keyUp(Keys.CONTROL).perform();**←копируем в буфер обмена

**actions.keyDown(Keys.TAB).keyUp(Keys.TAB).perform(); ←** переходим на соседнее поле ввода. Если поле ввода не пососедству то надо его отдельной командой найти драйвером, иначе вставка текста будет не туда.

**actions.keyDown(Keys.CONTROL).sendKeys("v").keyUp(Keys.CONTROL).perform(); ←** собственно ctrl+v

sendKeys() - имитирует нажатие и отпускание, удобно когда надо нажать на одну клавишу. А если нужна комбинация то имитируется как нажатие кнопки, так и отпускание. Обязательно помни про perform();

9)**обновляем страницу нажимая на f5 (только помоему это работает и без ctrl...)**

action.keyDown(Keys.CONTROL).sendKeys(Keys.F5).build().perform();

10)

11)

12)

13)

14)