**ACTION CHAINS**

MOUSE HOVER ACTIONS

ActionChains are a way to automate low level interactions such as

    mouse movements, mouse button actions, key press, and context menu interactions.

    This is useful for doing more complex actions like hover over and drag and drop.

ActionChains can be used in a chain pattern::

        menu = driver.find\_element(By.CSS\_SELECTOR, ".nav")

        hidden\_submenu = driver.find\_element(By.CSS\_SELECTOR, ".nav #submenu1")

        ActionChains(driver).move\_to\_element(menu).click(hidden\_submenu).perform()

    Or actions can be queued up one by one, then performed.::

        menu = driver.find\_element(By.CSS\_SELECTOR, ".nav")

        hidden\_submenu = driver.find\_element(By.CSS\_SELECTOR, ".nav #submenu1")

        actions = ActionChains(driver)

        actions.move\_to\_element(menu)

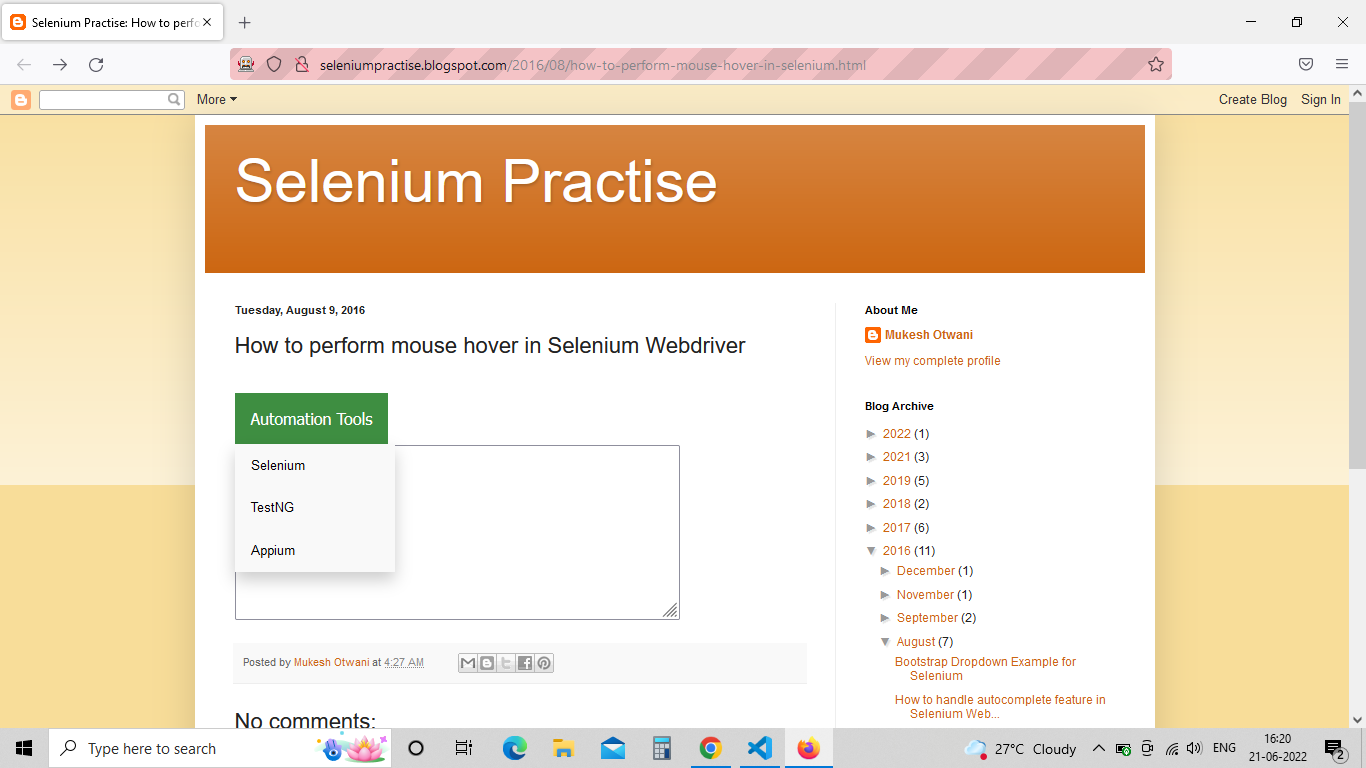
        actions.click(hidden\_submenu)

        actions.perform()

    Either way, the actions are performed in the order they are called, one after

    another.

**MOUSE HOVER ON A DROP DOWN MENU**



From the above website our aim is to --- point the mouse on the Automation Tools drop down --- we get the menu --- click on the Appium.

#mouse hovering action

#here when a mouse is pointed on a drop down menu then the options will come out of that drop down

from threading import activeCount

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.common.action\_chains import ActionChains

driver = webdriver.Firefox()

driver.maximize\_window()

driver.get("http://seleniumpractise.blogspot.com/2016/08/how-to-perform-mouse-hover-in-selenium.html")

#giving the xpath of the drop down menu

automationtool = driver.find\_element(by=By.XPATH,value='/html/body/div[3]/div[2]/div[2]/div[2]/div[2]/div[2]/div[2]/div/div[4]/div[1]/div/div/div/div[1]/div/div/div/div[1]/div[2]/div[1]/div[2]/button')

#passing the xpath stored variable into action chains

act = ActionChains(driver)

#move\_to\_element method is used to move the mouse to the middle of an element.

#always remember at the end of the action, perform() should be called or else the action will not be executed

act.move\_to\_element(automationtool).perform()

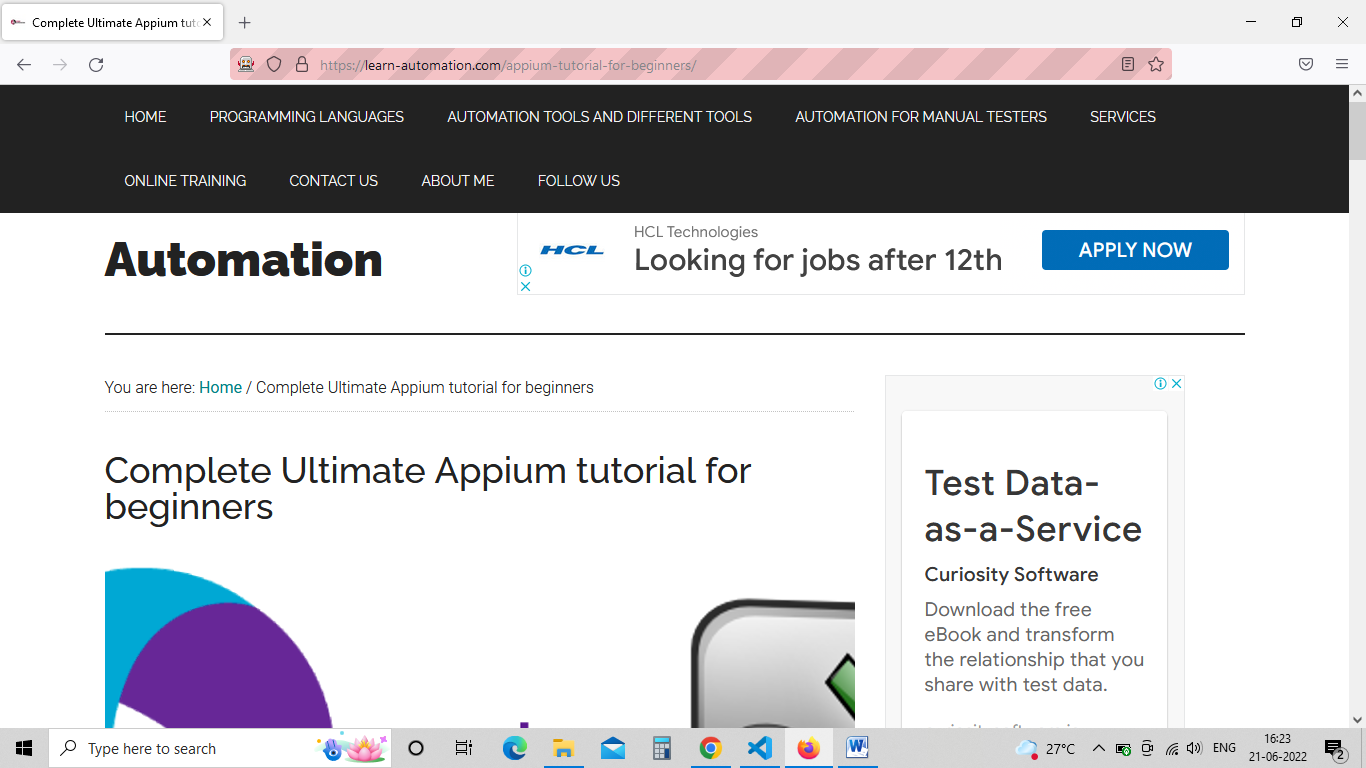
#as the mouse is pointed on the menu , now we will perform the click() method on Appium

driver.find\_element(by=By.XPATH,value='/html/body/div[3]/div[2]/div[2]/div[2]/div[2]/div[2]/div[2]/div/div[4]/div[1]/div/div/div/div[1]/div/div/div/div[1]/div[2]/div[1]/div[2]/div/a[3]').click()

# The above code can also be written in different methods as shown in the below

act.move\_to\_element(automationtool).pause(3).click(driver.find\_element(by=By.XPATH,value='/html/body/div[3]/div[2]/div[2]/div[2]/div[2]/div[2]/div[2]/div/div[4]/div[1]/div/div/div/div[1]/div/div/div/div[1]/div[2]/div[1]/div[2]/div/a[3]')).perform()

after performing the action it will result in as shown in below pic



**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**DRAG AND DROP OPERATIONS**

<http://www.dhtmlgoodies.com/scripts/drag-drop-custom/demo-drag-drop-3.html>

<https://testautomationpractice.blogspot.com/>

<https://demos.telerik.com/kendo-ui/treeview/dragdrop>

<https://jqueryui.com/droppable/>

above links are for practice reference

Syntax for drag and drop method :---

    def drag\_and\_drop(self, source, target):

        """

        Holds down the left mouse button on the source element,

           then moves to the target element and releases the mouse button.

        :Args:

         - source: The element to mouse down.

         - target: The element to mouse up.

        """

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.common.action\_chains import ActionChains

driver = webdriver.Firefox()

driver.maximize\_window()

# calling the website

driver.get("https://testautomationpractice.blogspot.com/")

# calling the actionchain method and assigning it to a variable "act"

act = ActionChains(driver)

# xpath of the source item to be dragged

src = driver.find\_element(by=By.ID ,value="draggable")

# xpath of the destination to be dropped

dest = driver.find\_element(by=By.ID ,value="droppable")

# passing the src and dest variables as argu into drag\_and\_drop method and perform()

act.drag\_and\_drop(src,dest).perform()

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**DOUBLE CLICK**

SYNTAX OF DOUBLE\_CLICK :---

def double\_click(self, on\_element=None):

        """

        Double-clicks an element.

        :Args:

         - on\_element: The element to double-click.

           If None, clicks on current mouse position.

        """

        if on\_element:

            self.move\_to\_element(on\_element)

        self.w3c\_actions.pointer\_action.double\_click()

        for \_ in range(4):

            self.w3c\_actions.key\_action.pause()

        return self

Program :---

from selenium import webdriver

from selenium.webdriver.common.action\_chains import ActionChains

from selenium.webdriver.common.by import By

driver = webdriver.Firefox()

driver.maximize\_window()

driver.get("https://demos.telerik.com/kendo-ui/treeview/animation")

# This if condtion is created to handle cookies

# if cookies are present before loading the page this block will execute to click accept cookies

if len(driver.find\_element(by=By.XPATH,value="//button[text()='Accept Cookies']"))>0:

    driver.find\_element(by=By.XPATH,value="//button[text()='Accept Cookies']").click()

    print("Cookies accepted")

act = ActionChains(driver)

# created a variable to store the xpath for double click button

ele = driver.find\_element(by=By.XPATH,value="//span[text()='Furniture']")

# calling the double\_click method

# only one argu will be taken by this method

act.double\_click(ele).perform()

# to check whether double click executed or not from the drop down on double click button

txt = driver.find\_element(by=By.XPATH,value="//span[text()='Tables & Chairs']").text

# printing the text from one of the option

print("captured text is \*\*\*\*",txt,"\*\*\*\*")

# asserting

assert "Tables" in txt

print("assertion is completed")

driver.quit()

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**COPY AND PASTE**

To perform these keyboard operations initially we need to import necessary library such as

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.action\_chains import ActionChains

keys --- library is used for keys class i.e keyboard keys for this program

key\_down 🡪 This is used to press the key and hold

key\_up 🡪 This is used to release the pressed key

Keys.CONTROL 🡪 Is nothing but control button from keyboard

Program :---

from typing import KeysView

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.action\_chains import ActionChains

driver = webdriver.Firefox()

driver.maximize\_window()

driver.get("https://www.facebook.com/")

driver.find\_element(by=By.ID,value='email').send\_keys("santosh")

act = ActionChains(driver)

act.key\_down(Keys.CONTROL).send\_keys("a").key\_up(Keys.CONTROL).perform()

act.key\_down(Keys.CONTROL).send\_keys("c").key\_up(Keys.CONTROL).perform()

act.key\_down(Keys.TAB).perform()

act.key\_down(Keys.CONTROL).send\_keys("v").key\_up(Keys.CONTROL).perform()

act.send\_keys(Keys.ENTER).perform()

"""

    we can also write the above

    code in a single line as shown in below

"""

act.key\_down(Keys.CONTROL).send\_keys("a").key\_up(Keys.CONTROL)\

    .key\_down(Keys.CONTROL).send\_keys("c").key\_up(Keys.CONTROL)\

        .key\_down(Keys.TAB).perform()\

            .key\_down(Keys.CONTROL).send\_keys("v").key\_up(Keys.CONTROL)\

                .send\_keys(Keys.EN

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**RIGHT CLICK ON THE SCREEN**

TER).perform()