

selenium

21 July 2023 21:48

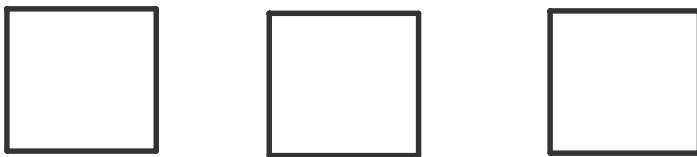
Selenium offical website:
<https://www.selenium.dev/>

Selenium Webdriver:

- One of its component
- It is a module
 - Firefox driver--->Firefox()
 - Chrome Driver ----> Chrome()
 - Edge---> Edge()
- It is a api (application programming interface) why webdriver is api?

Understanding web layers

- 1) GUI/UI or presentation layer- GUI testing
- 2) Application server- contains all business logic- api tetsing
- 3) Database layer- backend layer-database testing



Why webdriver is API?

Automation code -> webdriver -> Application open in browser



Webdrivers contains classes method so we are refering those in our automation. Job of webdriver to understand our code and interact with browser.

That's why we are calling webdriver is an api

Selenium 3.x

Selenium client lib or selenium language binding

Selenium client library(java/python/ruby) ---json wire protocol over http-->webdrivers--->w3c proto-->browsers

- Not stable



Selenium 4.x

Selenium client library(java/python/ruby) ---w3c protocol-->webdrivers--->w3c proto-->w3c proto-->browsers

- Stable and all over architure uses same poto

- Webdrivers are developed by browser companies
- Refer(official website)

Setup and installation:

- Install python latest version -[Welcome to Python.org](https://www.python.org/)
- Install pycharm -[Get Your Educational Tool - JetBrains](https://www.jetbrains.com/pycharm/)

Download webdrivers

Chrome- <https://chromedriver.chromium.org/downloads>

Firefox- <https://github.com/mozilla/geckodriver/releases/download/v0.33.0/geckodriver-v0.33.0-win-aarch64.zip>

Edge- <https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/>

Selenium4 python documentation

<https://www.selenium.dev/selenium/docs/api/py/api.html>

<https://selenium-python.readthedocs.io/>

Starting with selenium

- Open browser
- Webdriver.Chrome(executable_path="")
- Webdriver.Chrome()

- Webdriver.Chrome(service=Service("E:\selenium\drivers\chromedriver.exe"))

<https://pypi.org/project/webdriver-manager/>

```
pip install webdriver-manager
```

Use with Chrome

```
# selenium 3
from selenium import webdriver
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(ChromeDriverManager().install())
```

```
# selenium 4
from selenium import webdriver
from selenium.webdriver.chrome.service import Service as ChromeService
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().install()))
```

Locators

Identify various elements in the locators

Normal locators - can be taken from xml page

- ID-unique-fastest
- Name-unique
- Linktext/partialtext- unique
- Classname- to find more than element
- tagname- to find more than element

Customized locators- can be written or generated from the tool, can write our own xpath

- Css selector
 - Tag/id
 - Tag/class
 - Tag/attr
 - Tag/class/attr

- Xpath
- Absolute
- relative

Understanding Basic html structure

Returning Customer

Email:

Password:

Remember me? [Forgot password?](#)

LOG IN

```

... <div class="form-fields"> == $0
  <div class="inputs">
    <label for="Email">Email:</label>
    <input class="email" autofocus type="email" data-val="true" data-val-email="Wrong email" data-val-required="Please enter your email" id="Email" name="Email">
    <span class="field-validation-valid" data-valmsg-for="Email" data-valmsg-replace="true"></span>
    ::after
  </div>
  <div class="inputs">
    <label for="Password">Password:</label>
    <input class="password" type="password" id="Password" name="Password">
    <span class="field-validation-valid" data-valmsg-for="Password" data-valmsg-replace="true"></span>
    ::after
  </div>
  <div class="inputs reversed">...</div>
</div>
<div class="buttons">
  <button type="submit" class="button-1 login-button">Log in</button>
</div>
<input name="__RequestVerificationToken" type="hidden" value="CfDJ8Jtw2KziYYJGnXbyZNE5L9QvglyADTpULL-v1nqaAmfIQ2XhRtMOpBZxW-r8T1v2Nv
jvzCd1OJW044PzXnZdpq046VGVL1PgBxXSFM0fkDpnSyVHTFW-d1hxhd16WZQJ92k12DVISzqzaYNz3bA">
<input name="RememberMe" type="hidden" value="false">
</form>
</div>
</div>
<div class="external-authentication"></div>

```

```

<input class="email" autofocus="" type="email" data-val="true" data-val-email="Wrong email" data-val-required="Please enter your email" id="Email" name="Email">

```

- Input--> tag
- Class--> attr
- "email"--> value

- Tag for image--> img
- Tag for link--> a(anchor tag)--common tag=href hyperlink_reference

Css selector:(tagname is optional)

- Tag/id---(syntax: tag#valueofid)----input#email---->without tag #email
- Tag/class---syntax: tag.valueofclass-----input.email-->without tag .email
- Tag/attr---syntax: tag[attr=value]----input[data-val=true]---double quotes is optional for value --a[type=button] --

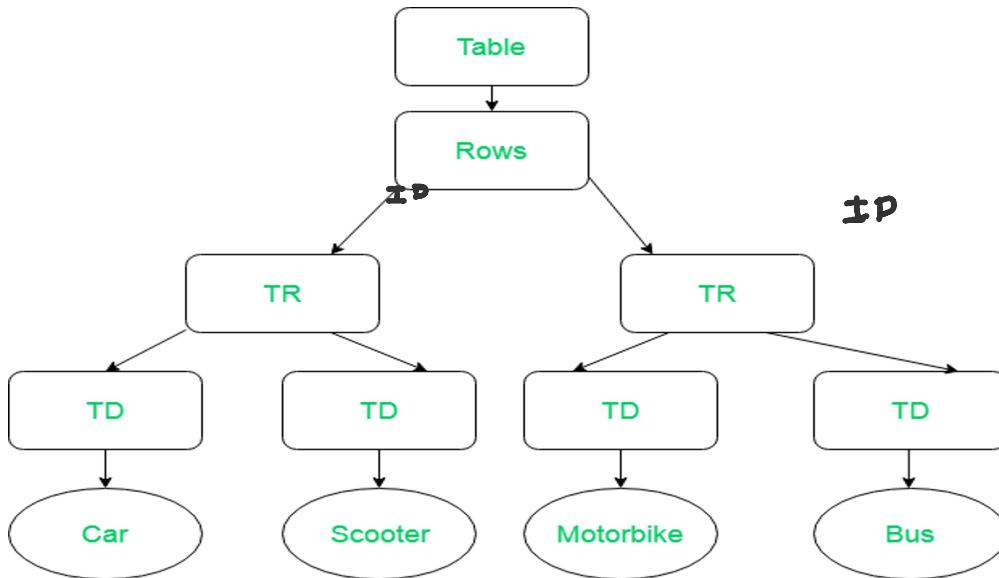
- without tag [type=button]
- Tag/class/attr---syntax: tag.valueofclass[attr=value]----input.email[data-val=true]

XPATH:

- - XML path
- Syntax / language for finding any element on the webpage
- Finds the location of the element in the page using HTML DOM structure
- Xpath can be used to navigate through the elements and attributes in DOM structure
- Xpath is an address of the element

DOM is an API interface provided by the browser, when a webpage is loaded browser creates a **document object model**

```
<table>
  <ROWS>
    <tr>
      <td>Car</td>
      <td>Scooter</td>
    </tr>
    <tr>
      <td>MotorBike</td>
      <td>Bus</td>
    </tr>
  </ROWS>
</table>
```



- Absolute xpath (full xpath) /html/body/div[1]/div[1]/ul[2]/li[4]/a
- Relative xpath (partial xpath) //*[@id="userProfileId"]/a

Difference between absolute and relative xpath

abs xpath starts from root html node	Rlt xpath directly jumps to element on the dom
/	//
Uses Tags/nodes	Uses attributes

How to write own xpath?

- Abs xpath. Start from bottom to top
/html/body/div[1]/div[1]/ul[2]/li[4]/a
- Rlt xpath- //tagname[@attr="value"]
//input[@id="email"] or //*[@id="email"]

Using tools we can automatically capture automatically

- Firebug/firepath---firefox ---deprecated /not available due to security reasons
- Rightclick element->inspect->highlight html node--> right click--copy path(to check xpath ctrl+f)
- Chropath
- Selector hub

Xpath options:

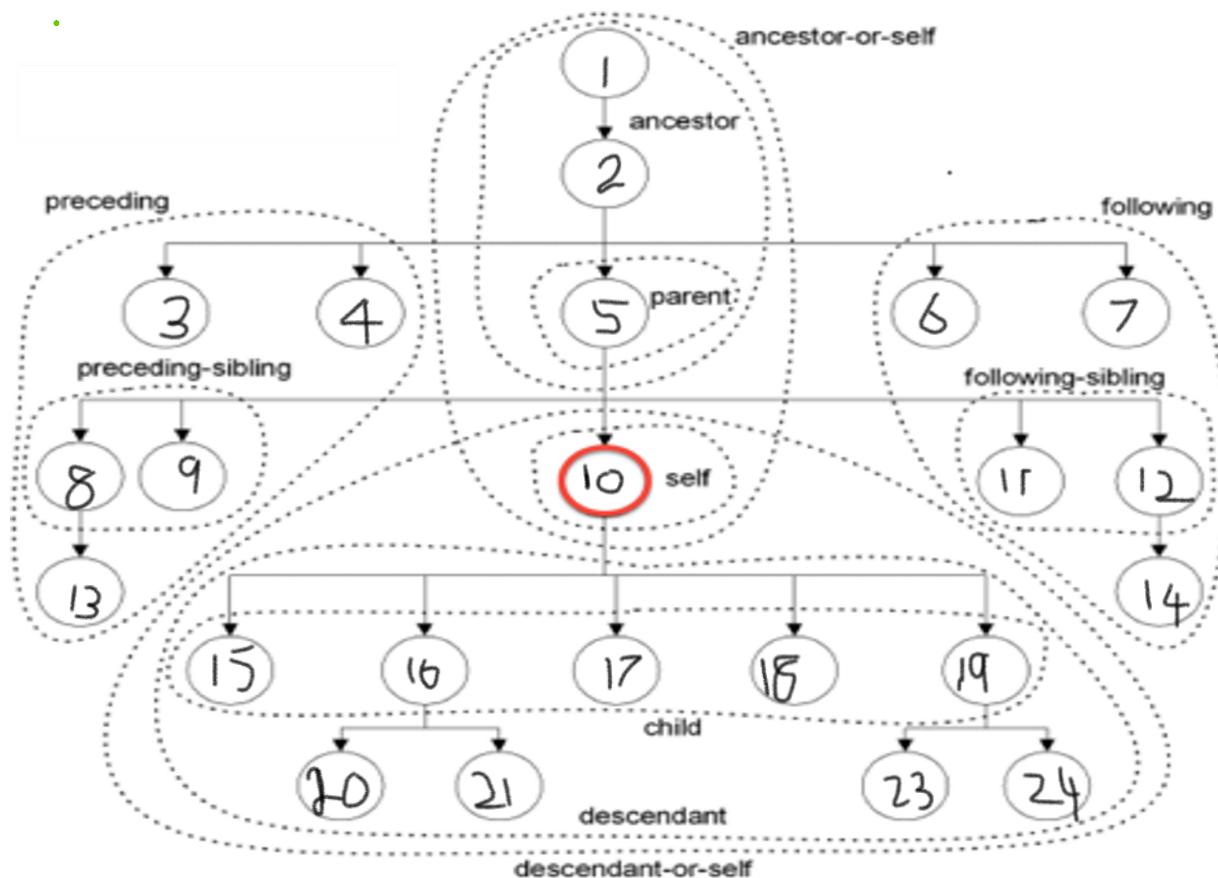
Eg: <input class="email" autofocus="" type="email" data-val="true" data-val-email="Wrong email" data-val-required="Please

enter your email" id="Email" name="Email">

- Or -----> //input[@id="email" or @data-val="true"]
- And-----> //input[@id="email" and @data-val="true"]
- ** Contains()--->
 - When button id is changing.id="start" and id="stop"
//*[contains(@id,st)]
 - //a[contains(text(),'Rajasthan Cylinders')]
- ** Starts-with()
//*[starts-with(@id,st)]
- Also Works with contains operator
- **Ends-with()
- Text()
//tag[text()=>innertext<"]

Xpath axes: (traverse top to bottom or bottom to top)

- Self
- Parent
- Child
- Ancestor
- Descendent
- Following
- Following-sibling
- Preceding
- Preceding-sibling
- Ancestor-or-self
- Descendent-or-self



AxisName*	Result
child	Selects all children of the current node
parent	Selects the parent of the current node
ancestor	Selects all ancestors (parent, grandparent, etc.) of the current node

descendant	Selects all descendants (children, grandchildren, etc.) of the current node	<code>//*[attr='value']/descendant::tagname</code>
following	Selects everything in the document after the closing tag of the current node	<code>//*[attr='value']/following::tagname</code>
preceding	Selects all nodes that appear before the current node in the document, except ancestors, attribute nodes and namespace nodes	<code>//*[attr='value']/parent::tagname</code>
following-sibling	Selects all siblings after the current node	<code>//current html tag[@attr='value']/following-sibling:: sibling tag[@attr='value']</code>
preceding-sibling	Selects all siblings before the current node	<code>//current html tag[@attr='value']/preceding-sibling::previous tag[@attr='value']</code>
self	Selects the current node	<code>//*[attr='value']/self::tagname</code>

<https://money.rediff.com/gainers/nse>

self

`//a[text()="Topic-wise Practice"]/self::a`

Parent

`//a[contains(text(),'Heritage Foods')]/parent::td`

child

`//a[contains(text(),'Heritage Foods')]/ancestor::tr/child::td`

ancestor

`//a[contains(text(),'Heritage Foods')]/ancestor::tr`

Desendents

`//a[contains(text(),'Heritage Foods')]/ancestor::tr/descendant::td`

following

`//a[contains(text(),'Heritage Foods')]/ancestor::tr/following::td`

Preceding

`//a[contains(text(),'Heritage Foods')]/ancestor::tr/preceding::tr`

Preceding sibling

`//a[contains(text(),'Heritage Foods')]/ancestor::tr/preceding-sibling::tr`

`//a[contains(text(),'Heritage Foods')]/ancestor-or-self::tr`

findelement VS findelements:

Findelement <code>//ul[@id="hslider"]//a[contains(text(),"Python")]</code>	Findelements <code>//ul[@id="hslider"]//a</code>
With locator matching single web element Returns single web element	With locator matching single web element Returns single web elements in list
With locator matching multiple web elements Matches with first element and returns that elem	With locator matching multiple web elements Returns multiple web elements in list
NoSuchElementException error	Will not throw an exceptions. returns a empty list
Clear()	Clears the text in the input box [eg: https://www.makemytrip.com/]
Text	Will not Return the text in the input box. This method will work only on innertext. To get inner text Mostly linktext will have innertext.
Get_attribute()	Returns the text in the input box. Returns value of any attribute.
What is Innertext?	value between > < tag <code><button type="submit" class="button-1 login-button">Log in</button></code>

Example:

```
#gettethetexttypedintheinputboxtextvsgt.getAttribute()  
importtime  
  
fromseleniumimportwebdriver  
fromselenium.webdriver.chrome.serviceimportService  
fromselenium.webdriver.common.byimportBy  
  
servcie_obj=Service("E:\selenium\drivers\chromedriver.exe")  
driver=webdriver.Chrome(service=servcie_obj)  
driver.get("https://www.geeksforgeeks.org/")  
driver.maximize_window()  
  
driver.find_element(By.XPATH,'//*[@id="gcse-form"]/button/i').click()  
google_search=driver.find_element(By.XPATH,'//*[@id="gcse-search-input"]')  
google_search.send_keys("selenium")  
print("usingtextmethod:",google_search.text)  
print("usinggetattributemethod:",google_search.get_attribute("value"))  
  
time.sleep(10)  
driver.find_element(By.XPATH,'//*[@id="gcse-search-input"]').clear()  
google_search.send_keys("python")  
  
print("Afterclearingtext...")  
print("usingtextmethod:",google_search.text)  
print("usinggetattributemethod:",google_search.get_attribute("value"))
```

Example2:

```
#gettethetexttypedintheinputboxtextvsgt.getAttribute()  
importtime  
  
fromseleniumimportwebdriver  
fromselenium.webdriver.chrome.serviceimportService  
fromselenium.webdriver.common.byimportBy  
  
servcie_obj=Service("E:\selenium\drivers\chromedriver.exe")  
driver=webdriver.Chrome(service=servcie_obj)  
driver.get("https://www.facebook.com/")  
driver.maximize_window()  
  
#driver.find_element(By.XPATH,'//*[@id="gcse-form"]/button/i').click()  
google_search=driver.find_element(By.XPATH,'//input[@id="email"]')  
google_search.send_keys("selenium")  
print("usingtextmethod:",google_search.text)  
print("usinggetattributemethod:",google_search.get_attribute("value"))  
  
driver.find_element(By.XPATH,'//input[@id="email"]').clear()  
google_search.send_keys("python")  
  
print("Afterclearingtext...")  
print("usingtextmethod:",google_search.text)  
print("usinggetattributemethod:",google_search.get_attribute("value"))
```

Selenium Commands:

➤ Application Commands

Works only after opening the browser. Throws error if the browser not opened

- .title - capture title of current page
- .current_url - capture current url
- .page_source - caputre source code of the page
- .get() - opens application url

Example:

```
fromseleniumimportwebdriver  
fromselenium.webdriver.chrome.serviceimportService  
fromselenium.webdriver.common.byimportBy  
  
driver=webdriver.Chrome()  
driver.get("https://www.geeksforgeeks.org/")  
driver.maximize_window()  
print(f"title:{driver.title}")  
print(f"currenturl:{driver.current_url}")  
print(f"pagesource:{driver.page_source}")
```

➤ Conditional commands

Returns true when the below condition satisfied

`is_displayed()`
`is_enabled()-->//button[@data-cy="continueBtn"]--> makemytrip example`
`is_selected()--> checkbox, radio buttons`

Example:

```
#is_displayed()is_enabled()is_selected()

importtime
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy

driver=webdriver.Chrome()
driver.get("https://www.geeksforgeeks.org/")
driver.maximize_window()
driver.implicitly_wait(10)
#is_displayed()
sign_elem=driver.find_element(By.XPATH,'//*[@id="userProfileId"]/a')
print(sign_elem.is_displayed())
sign_elem.click()

remember_elem=driver.find_element(By.XPATH,'//input[@name="rem"and@type="checkbox"]')
print(remember_elem.is_selected())
print(remember_elem.is_enabled())
remember_elem.click()
print(remember_elem.is_selected())
```

➤ Browser Commands

<code>Driver.close()</code>	<code>Driver.quit()</code>
For single page - closes the browser	Closes the page and quits the browser
Internally - Backend process will continue to run	Ensures Backend Process will be killed
For multiple page - closes the one page (ie) parent window mostly (driver focused)	Closes all the page and quits the browser

Example:

```
importtime
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy

driver=webdriver.Chrome()
driver.get("https://www.geeksforgeeks.org/")
driver.maximize_window()

elem_cookie=driver.find_element(By.XPATH,'//button[@class="consent-btn"]')
if elem_cookie.is_displayed():
    elem_cookie.click()
driver.find_element(By.XPATH,'//*[@id="RA-root"]/a/span').click()

time.sleep(10)

#driver.close()#driverfocusedonlygetclosed

driver.quit()#allprocesswillalsogetkilled
```

➤ Navigator commands

`.back()`
`.forward()`
`.refresh()`

Example:

```
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy

driver=webdriver.Chrome()
```

```

driver.get("https://www.geeksforgeeks.org/")
driver.maximize_window()
print(f"title:{driver.title}")
print(f"currenturl:{driver.current_url}")
#print(f"pagesource:{driver.page_source}")
driver.get("https://www.facebook.com/")
driver.back()
driver.refresh()
driver.forward()

```

➤ Wait Commands

- To solve Synchronization problem - no sync btwn code and browser
- Implicit wait
- Explicit wait
- Time.sleep()---related to python

Time.sleep()-->

Static/hard coding the timing/ poor performance

Need to mention every time when the wait for element required

If the element is available within the time it will wait upto the time mention in the function

If the element is not available with in the time it illl not find the element script will fail

Implicit wait----->

Driver.implicitly_wait(<time in sec>)

Act as wait until element found. Applies for overall script.

If the element found within the time limit it will pass on to the next line in the script. Good performance

Demerit: if the element is not seen within the time it will throw the error.

Explicit wait--->

Works based on condition(presence_of_element_located)

Need to create separate class

```

From selenium.webdriver.support.
Wait_obj=webdriver.WebDriverWait(driver_obj,
                               <time>,
                               <ignore_exception=[exceptions to be ignored]>,
                               <poll_frequency=2>----> polls every 2 sec, checks every 2 sec interval

)----declaration

```

And use the wait_obj in script where explicit wait needed

```

From selenium.webdriver import expected_condition as EC
Wait_obj.until(EC.<conditions ie element to be located>(<locator>))---no need of find element

```

Returns the element and perform the action

What if the condition not satisfied? - If the element is not satisfied it will wait until the time set during wait_obj creation

Explicit works based on element. Have to insert in script wherever the wait is needed.

```

import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.support import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

driver=webdriver.Chrome()
driver.get("https://www.redbus.in/")
driver.maximize_window()
#driver.implicitly_wait(10)
wait_obj=WebDriverWait(driver,10)

```

```

driver.find_element(By.ID,'cab_rental_vertical').click()
wait_obj.until(EC.presence_of_element_located((By.XPATH,'//*[@class="xrTxoXbGnuocVMadiJO"]')))

sign_elem=driver.find_element(By.XPATH,'//*[@class="xrTxoXbGnuocVMadiJO"]')
print(sign_elem.text)

driver.quit()

```

Working with different types of elements:

➤ Handling Checkboxes.(<https://itera-qa.azurewebsites.net/home/automation>)

Select specific checkbox---> get locator of specific check box and click
 Select all check box-----> get all elem in list and click. Use xpath that finds multiple element in list and access using for loop [using individual elements, using index]

To select single and multiple check box use xpath condition or python condition

Clear check_box also uses click method before that we need to check condition `is_selected()` to make sure all the check box are cleared

```

fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriver.support.waitimportWebDriverWait
fromselenium.webdriver.supportimportexpected_conditionsasEC

```

```

driver=webdriver.Chrome()
driver.get("https://itera-qa.azurewebsites.net/home/automation")
driver.maximize_window()

#radio_btn_elem=driver.find_element(By.CSS_SELECTOR,'input#female')
#radio_btn_elem.click()
#print(radio_btn_elem.is_selected())

#checkboxtoselectoneelement
#checkbox_elem=driver.find_element(By.CSS_SELECTOR,'input#sunday')
#checkbox_elem.click()
#print(checkbox_elem.is_selected())

#toselectallcheckbox
checkbox_elems=driver.find_elements(By.XPATH,'*[contains(@id,"day")]')
print(len(checkbox_elems))

forelemincheckbox_elems:
ifnotelem.is_selected():
elem.click()

forelemincheckbox_elems:
print(elem.is_selected())

```

➤ Links.

Internal - nav in same page/page open internally
 External - nav to other page

```

importtime
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriver.support.waitimportWebDriverWait
fromselenium.webdriver.supportimportexpected_conditionsasEC

driver=webdriver.Chrome()
driver.get("https://www.geeksforgeeks.org/")
driver.maximize_window()
print(driver.title)
#driver.find_element(By.LINK_TEXT,"InterviewPreparation").click()
#print(driver.title)
driver.find_element(By.LINK_TEXT,"Communityishere").click()
print(driver.title)

#driver.quit()

```

broken links - future development dev place the links in webpage

Deadlinkcity.com---> brokenlinks
Broken links returns error codes- which ever links are giving error codes are brokenlinks.
Request module is required for checking broken links

```
http://www.domaindoesnot.exist/
Traceback (most recent call last):
  File "C:\Users\user\AppData\Local\Programs\Python\Python38\lib\site-packages\urllib3\connection.py", line 174, in
    _new_conn
    conn = connection.create_connection(
  File "C:\Users\user\AppData\Local\Programs\Python\Python38\lib\site-packages\urllib3\util\connection.py", line 72,
in create_connection
    for res in socket.getaddrinfo(host, port, family, socket.SOCK_STREAM):
  File "C:\Users\user\AppData\Local\Programs\Python\Python38\lib\socket.py", line 918, in getaddrinfo
    for res in _socket.getaddrinfo(host, port, family, type, proto, flags):
socket.gaierror: [Errno 11001] getaddrinfo failed
```

- 1) Get the brokenlink from href using get_attribute() method
- 2) Use res=request.head(url)
- 3) Res.status_code()>=400:
- 4) Return bad url

```
import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.support import wait import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

driver=webdriver.Chrome()
driver.get("http://www.deadlinkcity.com/")
driver.maximize_window()
print(driver.title)

driver.find_element(By.LINK_TEXT,"Errorcode401").click()
print(driver.title)

#driver.quit()
```

➤ Dropdown

How to select value from dropdown:
Mostly stated with select tag---option tag
Dropdown is a webelement in that all the options are also webelement
We have select method in python.

- a. Get the locator of drop down use find_element
- b. Select() class
Drp_country_ele= driver.find_element()

```
Drop_county=Select(Drp_country_ele)
Dropcountry.select_by_visible_text("pass_the text")
Drpcountry.Select_by_value("attribute of option")
Drpcountry.select_by_index("index_number")
```

Returns all the options element
Drpcountry.options
Use the for loop to access the options

Without using buildin function how to select dropdown?

```
Alloption=Drpcountry.options
For opt in alloption:
    If opt.text == "name":
        Opt.click()
        Break
```

Works only for select tag. If not select tag ...use the findelement of all options and creatte thefunc with click method.

```
import time
```

```

fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriver.support.waitimportWebDriverWait
fromselenium.webdriver.support.selectimportSelect
fromselenium.webdriver.supportimportexpected_conditionsasEC

driver=webdriver.Chrome()
driver.get("https://itera-qa.azurewebsites.net/home/automation")
driver.maximize_window()

drop_down_elem=driver.find_element(By.XPATH,'//*[@class="custom-select"]')
options_elem>Select(drop_down_elem)

options_elem.select_by_visible_text("Italy")
options_elem.select_by_index("5")
options_elem.select_by_value("3")
print(options_elem.options)

#driver.quit()

```

Switch to commands

Alerts or pop up windows:

https://the-internet.herokuapp.com/javascript_alerts
<https://mypage.rediff.com/>

- 1) Can get text from alert box
- 2) Give input
- 3) Ok and cancel button

Driver.switch_to.alert

```

importtime
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriver.support.waitimportWebDriverWait
fromselenium.webdriver.support.selectimportSelect
fromselenium.webdriver.supportimportexpected_conditionsasEC

driver=webdriver.Chrome()
driver.get("https://the-internet.herokuapp.com/javascript_alerts")
driver.maximize_window()

driver.find_element(By.XPATH,"//button[contains(text(),'ClickforJSPrompt')]").click()
time.sleep(5)
alert_window=driver.switch_to.alert
print(alert_window.text)
alert_window.send_keys("welcome")
#alert_window.accept()#-->clicksopen
alert_window.dismiss()#-->cancel

#driver.quit()

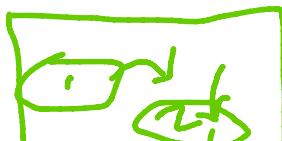
```

Frames or Iframes

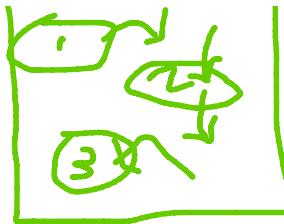
<https://www.selenium.dev/selenium/docs/api/java/index.html?overview-summary.html>

Tags for frame	-----> frame/iframe/form
To get total number of frames	Use ctrl+f in dev tool

- 1) Driver.switch_to_frame() # selenium3
- 2) driver.switch_to.frame() #selenium4
 pass name,id,frameaswebelement
 switch_to.frame(name of the frame)
 switch_to.frame(id of the name)



```
2) driver.switch_to.frame()      #selenium4
pass name,id,frameaswebelement
switch_to.frame(name of the frame)
switch_to.frame(id of the name)
switch_to.frame(webElement)
switch_to.frame(o)--Index
```



driver.switch_to.default_content()---- controls comes to main page

```
importtime
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriver.support.waitimportWebDriverWait
fromselenium.webdriver.support.selectimportSelect
fromselenium.webdriver.supportimportexpected_conditionsasEC

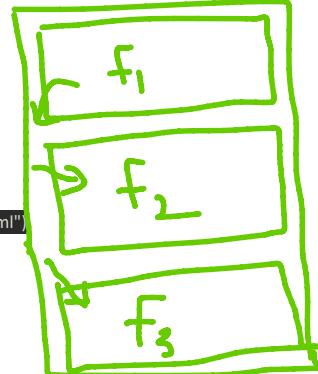
driver=webdriver.Chrome()
driver.get("https://www.selenium.dev/selenium/docs/api/java/index.html?overview-summary.html")
driver.maximize_window()

#framesoriframes

driver.switch_to.frame('packageListFrame')
driver.find_element(By.LINK_TEXT,'org.openqa.selenium').click()
driver.switch_to.default_content()

driver.switch_to.frame('packageFrame')
driver.find_element(By.LINK_TEXT,'WebDriver').click()
driver.switch_to.default_content()

driver.switch_to.frame('classFrame')
driver.find_element(By.XPATH,'//ul[@class="navList"]/li/a[contains(text(),"Help")]).click()
driver.switch_to.default_content()
#driver.quit()
```



Innerframes:

<https://demo.automationtesting.in/Frames.html>
<https://ui.vision/demo/webtest/frames/>

driver.switch_to.parent_frame() ----> control goes to parent frame

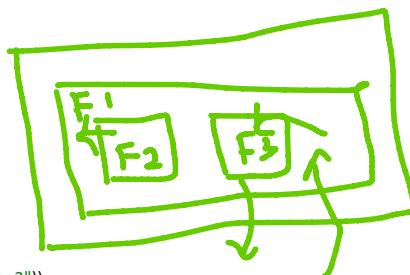
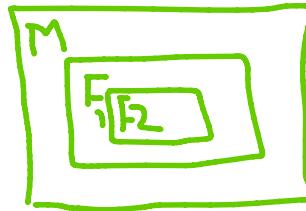
```
importtime
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriver.support.waitimportWebDriverWait
fromselenium.webdriver.support.selectimportSelect
fromselenium.webdriver.supportimportexpected_conditionsasEC

driver=webdriver.Chrome()
driver.get("https://ui.vision/demo/webtest/frames/")
driver.maximize_window()

#framesoriframes
frame_elem=driver.find_element(By.XPATH,'//frameset/frameset/frame[2]')
driver.switch_to.frame(frame_elem)
print(driver.find_element(By.XPATH,'//input[@name="mytext3"]').send_keys("reachedframe3"))
print(driver.find_element(By.XPATH,'//p').text)

inner_frame=driver.find_element(By.XPATH,'/html/body/center/iframe')
driver.switch_to.frame(inner_frame)
print(driver.find_element(By.XPATH,'//div[@class="ahS2Le"]/div').text)

driver.switch_to.parent_frame()
driver.find_element(By.XPATH,'//input[@name="mytext3"]').clear()
driver.find_element(By.XPATH,'//input[@name="mytext3"]').send_keys("reachedframe3again")
print(driver.find_element(By.XPATH,'//input[@name="mytext3"]').get_attribute("value"))
```



➤ Handle windows

Windows are handles based on window id
Window ids are generated by browser

```
driver.switch_to.window(windowid)
driver.current_window_handle-->retunscurrentwindows
driver.window_handles-->returnsmultiplewindows
```

<https://www.myntra.com/tops>

```
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy

servcie_obj=Service("E:\selenium\drivers\chromedriver.exe")
driver=webdriver.Chrome(service=servcie_obj)
driver.get("https://www.geeksforgeeks.org/")
driver.maximize_window()

driver.find_element(By.LINK_TEXT,'Communityishere').click()
driver.find_element(By.LINK_TEXT,'Communityishere').click()
```

```
print(driver.current_window_handle)
print(driver.window_handles)

forhandleindriver.window_handles:
    driver.switch_to.window(handle)
    print(driver.title)
    if"Community"indriver.title:
        driver.close()
```

Authentication pop ups-

https://the-internet.herokuapp.com/basic_auth
(admin/admin)

we don't have webelement
Inject username password- syntax

- 1) Using url---> this is called bypassing the alert

Syntax

<http://username:password@url>

https://admin:admin@the-internet.herokuapp.com/basic_auth

- 2) Using autoit-install pyautoit -

Pip install pyautoit

```
importtime

fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
importautoit

driver=webdriver.Chrome()
driver.get("https://the-internet.herokuapp.com/basic_auth")
driver.maximize_window()

time.sleep(5)

autoit.win_wait_active("",30)
autoit.send("admin{TAB}")
autoit.send("admin{Enter}")
```

- a. To learn more on autoit refer below links

<https://www.autoitscript.com/autoit3/docs/>

<https://github.com/jacejh/pyautoit/tree/master>

- 1) Allow popup (Whatmylocation.com)

```
Ops=webdriver.ChromeOptions  
Ops.add_arguments("--disable--notifications")  
...  
Driver=wedriver.Chrome(service=serv_obj, options=ops)
```

<https://chromedriver.chromium.org/capabilities>
<https://peter.sh/experiments/chromium-command-line-switches/>

➤ Webletable

- Static - data are same()
 - Dynamic - data updated every time(<https://money.rediff.com/gainers/bse/daily/group>)
- Tags for table is /table/tbody (repr whole body)
/table/thead (repr whole header)

Tr---> table row

Th---> table header available in first row

Td---> table data

- Count number of row and col
- Read row col
- Read all row col
- Read row col based on condition

#Webtables

```
importtime  
fromseleniumimportwebdriver  
fromselenium.webdriver.chrome.serviceimportService  
fromselenium.webdriver.common.byimportBy  
  
driver=webdriver.Chrome()  
driver.get("https://money.rediff.com/gainers")  
driver.maximize_window()  
  
#findingnumberofrowsandcol  
no_of_row=driver.find_elements(By.XPATH,'//table[@class="dataTable"]/tbody/tr')  
no_of_col=driver.find_elements(By.XPATH,'//table[@class="dataTable"]/tbody/tr[1]/td')  
print(f"rows:{len(no_of_row)}\ncolumns:{len(no_of_col)}")  
  
#readsspecificrowandcoloumdata  
#data=driver.find_element(By.XPATH,'//table[@class="dataTable"]/tbody/tr[100]/td[1]').text  
#print(data)  
  
#readallrowandcolodata  
#print("Printingallrowandcoloumdata")  
  
#forinrange(1,len(no_of_row)+1):  
#forinrange(1,len(no_of_col)+1):  
#data=driver.find_element(By.XPATH,f'//table[@class="dataTable"]/tbody/tr[{r}]/td[{c}]').text  
#print(data,"\\t")  
#print("*****")  
  
#readdatabasedoncondition  
#getallinfoofCLIOInfotech  
'''forinrange(1,len(no_of_row)+1):  
stock_name=driver.find_element(By.XPATH,f'//table[@class="dataTable"]/tbody/tr[{r}]/td[1]').text  
ifstock_name=="CLIOInfotech":  
#togetspecificdetialmetioncorrectindex  
#print(driver.find_element(By.XPATH,f'//table[@class="dataTable"]/tbody/tr[{r}]/td[1]').text)  
#print(driver.find_element(By.XPATH,f'//table[@class="dataTable"]/tbody/tr[{r}]/td[4]').text)  
  
#togetalldetialinthatstpecificstoc  
forinrange(1,len(no_of_col)+1):  
print(driver.find_element(By.XPATH,f'//table[@class="dataTable"]/tbody/tr[{r}]/td[{c}]').text,end="\\t")  
break'''
```

```
#getallinfoofstockwhichcurrentpriceislessthan10

for i in range(1, len(no_of_row)+1):

    stock_current_price=float(
driver.find_element(By.XPATH,f'//table[@class="dataTable"]/tbody/tr[{r}]/td[4]').text.replace(",",""))

    #print(type(stock_current_price))

    if stock_current_price<=10:

        for j in range(1, len(no_of_col)+1):
            print(driver.find_element(By.XPATH,f'//table[@class="dataTable"]/tbody/tr[{r}]/td[{c}]').text,end="\t")

        print("\n")

    driver.close()
```

➤ Save screenshot

Takes screenshot of the webpage - useful at the time of failure

```
import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By

driver=webdriver.Chrome()
driver.get("https://money.rediff.com/gainers")
driver.maximize_window()
driver.save_screenshot("./webtables.png")
driver.close()

import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By

driver=webdriver.Chrome()
driver.get( "https://money.rediff.com/gainers")
driver.maximize_window()
#driver.save_screenshot("./webtables.png")
#driver.save_screenshot(r"C:\Users\JS327941\Desktop\webtables.png")
driver.save_screenshot_as_png()# save as png format
driver.save_screenshot_as_base64()# save as binary format
driver.close()
```

➤ Mouse Actions

- Mouse hover
- Right click
- Double click
- Drag and drop

1) Mouse Hover

```
from selenium.webdriver import ActionChains
action=ActionChains(driver_obj)
move_to_element(<web driver>

from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver import ActionChains

driver=webdriver.Chrome()
driver.get("https://www.geeksforgeeks.org/")
driver.maximize_window()

courses_elem=driver.find_element(By.XPATH,'//ul[@class="header-main__list"]/li/span[contains(text(),"Courses")]')
course1_elem=driver.find_element(By.XPATH,'//ul[@class="mega-dropdown"]/li/span[contains(text(),"ForWorkingProfessionals")]')
course2_elem=driver.find_element(By.XPATH,'//ul[@class="mega-dropdown"]/li/ul/li/a[text()="DevOps(Live")']

action=ActionChains(driver)

#Mousehover
action.move_to_element(courses_elem).move_to_element(course1_elem).move_to_element(course2_elem).click().perform()
```

```
print(driver.title)
driver.quit()
```

- 2) Right click----context_click(rclk_elem)
- 3) Double click----double_click(dclk_elem)

```
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriverimportActionChains

driver=webdriver.Chrome()
driver.get("https://demo.guru99.com/test/simple_context_menu.html")
driver.maximize_window()

rclk_elem=driver.find_element(By.XPATH,'//*[@id="authentication"]/span')
paste_elem=driver.find_element(By.XPATH,'//*[@id="authentication"]/ul/li[4]')
dclk_elem=driver.find_element(By.XPATH,'//*[@id="authentication"]/button')

action=ActionChains(driver)

#rightclick
action.context_click(rclk_elem).context_click(paste_elem).perform()
alert=driver.switch_to.alert
print(alert.text)
alert.accept()

#doubleclick
action.double_click(dclk_elem).perform()
alert=driver.switch_to.alert
print(alert.text)
alert.accept()

driver.quit()
```

- 4) Drag and drop

- a. Get elem of source and dest

```
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriverimportActionChains
```

```
driver=webdriver.Chrome()
driver.get("http://www.dhtmlgoodies.com/scripts/drag-drop-custom/demo-drag-drop-3.html")
driver.maximize_window()

source_elem=driver.find_element(By.ID,'box6')
dest_elem=driver.find_element(By.ID,'box106')

action=ActionChains(driver)

#draganddrop
action.drag_and_drop(source_elem,dest_elem).perform()
```

- b. Offset

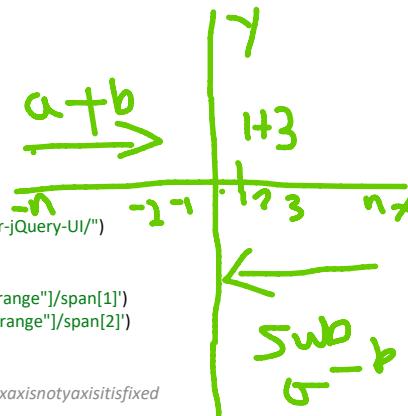
```
fromseleniumimportwebdriver
fromselenium.webdriver.chrome.serviceimportService
fromselenium.webdriver.common.byimportBy
fromselenium.webdriverimportActionChains
```

```
driver=webdriver.Chrome()
driver.get("https://www.jqueryscript.net/demo/Price-Range-Slider-jQuery-UI/")
driver.maximize_window()

min_slider_elem=driver.find_element(By.XPATH,'//*[@id="slider-range"]/span[1]')
max_slider_elem=driver.find_element(By.XPATH,'//*[@id="slider-range"]/span[2]')

print(min_slider_elem.location)#{'x':59,'y':250}
print(max_slider_elem.location)#{'x':545,'y':250}--wecanmoveonlyxaxisnotyaxisifitfixed

action=ActionChains(driver)
# x axis for horizontal slider
# y axis will get changed for vertical slider
action.drag_and_drop_by_offset(min_slider_elem,100,0).perform()
action.drag_and_drop_by_offset(max_slider_elem,-145,0).perform()
```



```

#dependsonresolutionofwebpage
#dependsonbrowser

print(min_slider_elem.location)#{'x':161,'y':250}
print(max_slider_elem.location)#{'x':399,'y':250}

➤ Scrolling page

From browser when the page is long- it cannot be done by mouse action class
    ○ Scroll to some offset
    ○ Script until the element found
    ○ Scroll top to bottom

import time

from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver import ActionChains

driver=webdriver.Chrome()
driver.get("https://www.countries-ofthe-world.com/flags-of-the-world.html")
driver.maximize_window()

#scrolltooffset

driver.execute_script("window.scrollBy(0,3000)","")
print(driver.execute_script("return window.pageYOffset;"))

#scrolluntilelementfound
flag=driver.find_element(By.XPATH,'//img[@alt="FlagofIndia"]')
driver.execute_script("arguments[0].scrollIntoView();",flag)
print(driver.execute_script("return window.pageYOffset;"))

#scrolltoendofpage
driver.execute_script("window.scrollBy(0,document.body.scrollHeight)","")
print(driver.execute_script("return window.pageYOffset;"))

time.sleep(5)
#scrollbacktotopofpage
driver.execute_script("window.scrollBy(0,-document.body.scrollHeight)","")
print(driver.execute_script("return window.pageYOffset;"))

```

FileDownload:

```

#filedownload

from selenium import webdriver
from selenium.webdriver.common.by import By
import os
location=os.getcwd()

def chrome_setup():
    from selenium.webdriver.chrome.service import Service
    #tosavethedownloadsinprefredlocation
    preferences={"download.default_directory":location}
    ops=webdriver.ChromeOptions()
    ops.add_experimental_option("prefs",preferences)

    servcie_obj=Service("E:\selenium\drivers\chromedriver.exe")
    driver=webdriver.Chrome(service=servcie_obj,options=ops)
    return driver

def edge_setup():
    from selenium.webdriver.edge.service import Service

```

```

#tosavethedownloadsinpreferedlocation
preferences={"download.default_directory":location}
ops=webdriver.EdgeOptions()
ops.add_experimental_option("prefs",preferences)
servcie_obj=Service("E:\selenium\drivers\msedgedriver.exe")
driver=webdriver.Edge(service=servcie_obj,options=ops)
returndriver

deffirefox_setup():
fromselenium.webdriver.firefox.serviceimportService
#toskipandownloadthefilewindowinfirefox
#setting
ops=webdriver.FirefoxOptions()
ops.set_preference("browser.helperApps.neverAsk.saveToDisk","application/txt")#/msword,/txt,/pdf
#https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Common_types
ops.set_preference("bowser.download.manager.showWhenStarting",False)
ops.set_preference("browser.download.folderList",2)#0-desktop,1-downloadfolder,2-desiredloc
ops.set_preference("browser.download.dir",location)#onlythefolderlistis2
#servcie_obj=Service("E:\selenium\drivers\geckodriver.exe")
driver=webdriver.Firefox(options=ops)
returndriver

#driver=chrome_setup()
#driver=edge_setup()
driver=firefox_setup()
driver.maximize_window()
driver.get("https://file-examples.com/index.php/sample-documents-download/sample-doc-download/")
driver.find_element(By.XPATH,'//*[@id="table-files"]//tr[1]/td[5]/a').click()


```

For PDF files which uses browser download- this option is not working in edge browser(**known issue with bowser**)

```

#filedownload

fromseleniumimportwebdriver
fromselenium.webdriver.common.byimportBy
importos
location=os.getcwd()

defchrome_setup():
fromselenium.webdriver.chrome.serviceimportService
#tosavethedownloadsinpreferedlocation
preferences={"download.default_directory":location,"plugins.always_open_pdf_externally":True}
ops=webdriver.ChromeOptions()
ops.add_experimental_option("prefs",preferences)

servcie_obj=Service("E:\selenium\drivers\chromedriver.exe")
driver=webdriver.Chrome(service=servcie_obj,options=ops)
returndriver


```

```

defedge_setup():
fromselenium.webdriver.edge.serviceimportService
#tosavethedownloadsinpreferedlocation
preferences={"download.default_directory":location,"plugins.always_open_pdf_externally":True}
ops=webdriver.EdgeOptions()
ops.add_experimental_option("prefs",preferences)
servcie_obj=Service("E:\selenium\drivers\msedgedriver.exe")
driver=webdriver.Edge(service=servcie_obj,options=ops)
returndriver


```

```

deffirefox_setup():
fromselenium.webdriver.firefox.serviceimportService
#toskipandownloadthefilewindowinfirefox
#setting
ops=webdriver.FirefoxOptions()
ops.set_preference("browser.helperApps.neverAsk.saveToDisk","application/pdf")#/msword,/txt,/pdf
#https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Common_types
ops.set_preference("bowser.download.manager.showWhenStarting",False)
ops.set_preference("browser.download.folderList",2)#0-desktop,1-downloadfolder,2-desiredloc
ops.set_preference("browser.download.dir",location)#onlythefolderlistis2
ops.set_preference("pdfjs.disabled",True)
#servcie_obj=Service("E:\selenium\drivers\geckodriver.exe")
driver=webdriver.Firefox(options=ops)


```

```

return driver

#driver=chrome_setup()
#driver=edge_setup()
driver=firefox_setup()
driver.maximize_window()
driver.get('https://file-examples.com/index.php/sample-documents-download/sample-pdf-download/')
driver.find_element(By.XPATH,'//*[@id="table-files"]/tr[1]/td[5]/a').click()

#####
#####333

import pyautogui
import time
from selenium import webdriver
from selenium.webdriver.common.by import By
import os
print(os.getcwd())
location=os.getcwd()

def edge_setup():
    from selenium.webdriver.edge.service import Service
    service_obj=Service("E:\selenium\drivers\msedgedriver.exe")

    preferences={"download.default_directory":location,
    "plugins.always_open_pdf_externally":True,
    "download.prompt_for_download":False}
    #preferences={"download.default_directory":r"C:\Users\user\PycharmProjects\pythonselenium\venv\selenium"}

    ops=webdriver.EdgeOptions()
    ops.UseChromium=True
    ops.add_experimental_option("prefs",preferences)
    driver=webdriver.Edge(service=service_obj,options=ops)
    return driver

driver=edge_setup()
driver.get("edge://settings/downloads")

driver.find_element(By.XPATH,'//input[@aria-label="OpenOffice files in the browser"]').click()
time.sleep(2)

driver.get("https://file-examples.com/index.php/sample-documents-download/sample-doc-download/")
driver.maximize_window()

driver.find_element(By.XPATH,'//*[@id="table-files"]/tbody/tr[1]/td[5]/a').click()
time.sleep(10)

##pdf download
#driver.get("https://file-examples.com/index.php/sample-documents-download/sample-pdf-download/")
#driver.maximize_window()
#driver.find_element(By.XPATH,'//*[@id="table-files"]/tbody/tr[1]/td[5]/a').click()
#time.sleep(10)

#pyautogui.hotkey('ctrl','s')
#time.sleep(5)
#pyautogui.typewrite(location+ "/" + "pyautoit_download.pdf")
#pyautogui.press('enter')
#####

Upload file

from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.support.select import Select

service_obj=Service("E:\selenium\drivers\chromedriver.exe")
driver=webdriver.Chrome(service=service_obj)
driver.maximize_window()
driver.get('https://www.foundit.in/')
driver.find_element(By.XPATH,'//*[@id="heroSection-container"]/div[3]/div[2]/div[2]').click()
driver.find_element(By.ID,"file-upload").send_keys(r"C:\Users\user\Downloads\info.txt")

#####
#####33
➤ handling cookies- browsers remember data(username password, suggestion are based on data)--using cookies
import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service

```

```

driver=webdriver.Chrome()
driver.get( https://money.rediff.com/gainers)
driver.maximize_window()

#cookie---> having attribute like name, value, expiredate...which will return dict collection
#print(driver.get_cookie())
cookies = driver.get_cookies() # capture cookies from the browser
print("size of cookies", len(cookies))
# print dict of all cookies
#for c in cookies:
#    #print(c)
#    #print(c["domain"],":",c["name"])

# how to add new cookie
driver.add_cookie({"name": "mycookie", "value":"12345"})
cookies = driver.get_cookies() # capture cookies from the browser
print("size of cookies after adding cookie", len(cookies))

# delete the cookie

driver.delete_cookie("mycookie")
cookies = driver.get_cookies()
print("size of cookies after deleting cookie", len(cookies))
driver.delete_all_cookies()
cookies = driver.get_cookies()
print("size of cookies after deleting all cookies", len(cookies))
driver.quit()

#####
#> Headless mode testing-runs in backend-execution is faster
import time
from selenium import webdriver

def chrome_headless():
    from selenium.webdriver.chrome.service import Service
    ops=webdriver.ChromeOptions()
    ops.add_argument("--headless")
    driver=webdriver.Chrome(options=ops)
    return driver

def edge_headless():
    from selenium.webdriver.edge.service import Service
    ops=webdriver.EdgeOptions()
    ops.add_argument("--headless")
    driver=webdriver.Edge(options=ops)
    return driver

def firefox_headless():
    from selenium.webdriver.firefox.service import Service
    ops=webdriver.FirefoxOptions()
    ops.add_argument("--headless")
    driver=webdriver.Firefox(options=ops)
    return driver

#driver=chrome_headless()
#driver=edge_headless()
driver=firefox_headless()
driver.get( https://money.rediff.com/gainers)
print(driver.current_url)
print(driver.title)
driver.quit()

#####
#> Autocomplete
import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys

driver=webdriver.Chrome()
driver.get( https://www.google.com/)
driver.maximize_window()

```

```

search_elem = driver.find_element(By.XPATH, "//*[@name='q']")
search_elem.click()
# time.sleep(4)
# search_elem.send_keys(Keys.ARROW_DOWN)
# time.sleep(4)
# search_elem.send_keys(Keys.ARROW_DOWN)
# time.sleep(4)
#print(search_elem.get_attribute("value"))

# suggestion_len = len(driver.find_elements(By.XPATH, '//*[@id="Alh6id"]/div[1]/div/ul/li'))
# for sug in range(suggestion_len):
#     time.sleep(4)
#     search_elem.send_keys(Keys.ARROW_DOWN)
#     print(search_elem.get_attribute("value"))

elem = driver.find_elements(By.XPATH, '//*[@id="Alh6id"]/div[1]/div/ul/li/div/div[2]/div[1]/div[1]/span')
for i in elem:
    print(i.text)

```

Tips to freeze or get autosuggestions:

Control+\ or F8 pause the java exe

Go to event listeners-> look for 'blur' option -> click remove remove all property(works for **auto selection**)

Go to event listeners-> look for 'mouseout' option -> click remove remove all property(works for **tooltip**)

#####
#####333

➤ Datepicker-

Ref : [datetime — Basic date and time types — Python 3.11.4 documentation](#)

```

import datetime
#year month date time min sec millisec
date_time = datetime.datetime(2005, 4, 16)
print(date_time.strftime("%B %d, %Y %H:%M %p"))
current_time = datetime.datetime.now()
print(current_time.strftime("%B %d, %Y %H:%M %p"))

```

Shadow DOM

