**WebDriver Elements in Python:**

1. **current\_url:** URL of the currently loaded page.

Usage: driver.current\_url

1. **close():** Closes the current window

Usage: driver.close()

1. **accept**(): Accepts the alert available.

Usage: Alert(driver).accept()

1. **dismiss**(): Dismisses the alert available

Usage: Alert(driver).dismiss()

1. **send\_keys**(*keysToSend*): Send Keys to the field.
2. **back**(): Goes one step backward in the browser history.

Usage: driver.back()

1. **delete\_all\_cookies**(): Delete all cookies in the scope of the session.

Usage: driver.delete\_all\_cookies()

1. **delete\_cookie**(*name*): Deletes a single cookie with the given name.
2. **find\_element**(*by='id'*, *value=None*)

Find an element given a By strategy and locator. Prefer the find\_element\_by\_\* methods when possible.

|  |  |
| --- | --- |
| **Usage:** | element = driver.find\_element(By.ID, ‘foo’) |
| **Return type:** | [WebElement](https://selenium-python.readthedocs.io/api.html#selenium.webdriver.remote.webelement.WebElement) |

1. **find\_element\_by\_class\_name**(*name*): Finds an element by class name.

|  |  |
| --- | --- |
| **Args:** | * name: The class name of the element to find. |
| **Returns:** | * WebElement - the element if it was found |
| **Raises:** | * NoSuchElementException - if the element wasn’t found |
| **Usage:** | element = driver.find\_element\_by\_class\_name(‘foo’) |

1. **find\_element\_by\_css\_selector**(*css\_selector*): Finds an element by css selector.

|  |  |
| --- | --- |
| **Args:** | * css\_selector - CSS selector string, ex: ‘a.nav#home’ |
| **Returns:** | * WebElement - the element if it was found |
| **Raises:** | * NoSuchElementException - if the element wasn’t found |
| **Usage:** | element = driver.find\_element\_by\_css\_selector(‘#foo’) |

1. **find\_element\_by\_id**(*id\_*): Finds an element by id.

|  |  |
| --- | --- |
| **Args:** | * id\_ - The id of the element to be found. |
| **Returns:** | * WebElement - the element if it was found |
| **Raises:** | * NoSuchElementException - if the element wasn’t found |
| **Usage:** | element = driver.find\_element\_by\_id(‘foo’) |

1. **find\_element\_by\_link\_text**(*link\_text*):Finds an element by link text.

|  |  |
| --- | --- |
| **Args:** | * link\_text: The text of the element to be found. |
| **Returns:** | * WebElement - the element if it was found |
| **Raises:** | * NoSuchElementException - if the element wasn’t found |
| **Usage:** | element = driver.find\_element\_by\_link\_text(‘Sign In’) |

1. **find\_element\_by\_name**(*name*):Finds an element by name.

|  |  |
| --- | --- |
| **Args:** | * name: The name of the element to find. |
| **Returns:** | * WebElement - the element if it was found |
| **Raises:** | * NoSuchElementException - if the element wasn’t found |
| **Usage:** | element = driver.find\_element\_by\_name(‘foo’) |

1. **find\_element\_by\_partial\_link\_text**(*link\_text*):Finds an element by a partial match of its link text.

|  |  |
| --- | --- |
| **Args:** | * link\_text: The text of the element to partially match on. |
| **Returns:** | * WebElement - the element if it was found |
| **Raises:** | * NoSuchElementException - if the element wasn’t found |
| **Usage:** | element = driver.find\_element\_by\_partial\_link\_text(‘Sign’) |

1. **find\_element\_by\_tag\_name**(*name*): Finds an element by tag name.

|  |  |
| --- | --- |
| **Args:** | * name - name of html tag (eg: h1, a, span) |
| **Returns:** | * WebElement - the element if it was found |
| **Raises:** | * NoSuchElementException - if the element wasn’t found |
| **Usage:** | element = driver.find\_element\_by\_tag\_name(‘h1’) |

1. **find\_element\_by\_xpath**(*xpath*): Finds an element by xpath.

|  |  |
| --- | --- |
| **Args:** | * xpath - The xpath locator of the element to find. |
| **Returns:** | * WebElement - the element if it was found |
| **Raises:** | * NoSuchElementException - if the element wasn’t found |
| **Usage:** | element = driver.find\_element\_by\_xpath(‘//div/td[1]’) |

1. **find\_elements**(*by='id'*, *value=None*): Find elements given a By strategy and locator. Prefer the find\_elements\_by\_\* methods when possible.

|  |  |
| --- | --- |
| **Usage:** | elements = driver.find\_elements(By.CLASS\_NAME, ‘foo’) |
| **Return type:** | list of WebElement |

1. **find\_elements\_by\_class\_name**(*name*): Finds elements by class name.

|  |  |
| --- | --- |
| **Args:** | * name: The class name of the elements to find. |
| **Returns:** | * list of WebElement - a list with elements if any was found. An empty list if not |
| **Usage:** | elements = driver.find\_elements\_by\_class\_name(‘foo’) |

1. **find\_elements\_by\_css\_selector**(*css\_selector*): Finds elements by css selector.

|  |  |
| --- | --- |
| **Args:** | * css\_selector - CSS selector string, ex: ‘a.nav#home’ |
| **Returns:** | * list of WebElement - a list with elements if any was found. An empty list if not |
| **Usage:** | elements = driver.find\_elements\_by\_css\_selector(‘.foo’) |

1. **find\_elements\_by\_id**(*id\_*): Finds multiple elements by id.

|  |  |
| --- | --- |
| **Args:** | * id\_ - The id of the elements to be found. |
| **Returns:** | * list of WebElement - a list with elements if any was found. An empty list if not |
| **Usage:** | elements = driver.find\_elements\_by\_id(‘foo’) |

1. **find\_elements\_by\_link\_text**(*text*): Finds elements by link text.

|  |  |
| --- | --- |
| **Args:** | * link\_text: The text of the elements to be found. |
| **Returns:** | * list of webelement - a list with elements if any was found. an empty list if not |
| **Usage:** | elements = driver.find\_elements\_by\_link\_text(‘Sign In’) |

1. **find\_elements\_by\_name**(*name*): Finds elements by name.

|  |  |
| --- | --- |
| **Args:** | * name: The name of the elements to find. |
| **Returns:** | * list of webelement - a list with elements if any was found. an empty list if not |
| **Usage:** | elements = driver.find\_elements\_by\_name(‘foo’) |

1. **find\_elements\_by\_partial\_link\_text**(*link\_text*): Finds elements by a partial match of their link text.

|  |  |
| --- | --- |
| **Args:** | * link\_text: The text of the element to partial match on. |
| **Returns:** | * list of webelement - a list with elements if any was found. an empty list if not |
| **Usage:** | elements = driver.find\_elements\_by\_partial\_link\_text(‘Sign’) |

1. **find\_elements\_by\_tag\_name**(*name*): Finds elements by tag name.

|  |  |
| --- | --- |
| **Args:** | * name - name of html tag (eg: h1, a, span) |
| **Returns:** | * list of WebElement - a list with elements if any was found. An empty list if not |
| **Usage:** | elements = driver.find\_elements\_by\_tag\_name(‘h1’) |

1. **find\_elements\_by\_xpath**(*xpath*): Finds multiple elements by xpath.

|  |  |
| --- | --- |
| **Args:** | * xpath - The xpath locator of the elements to be found. |
| **Returns:** | * list of WebElement - a list with elements if any was found. An empty list if not |
| **Usage:** | elements = driver.find\_elements\_by\_xpath(“//div[contains(@class, ‘foo’)]”) |

1. **forward**(): Goes one step forward in the browser history.

Usage: driver.forward()

1. **fullscreen\_window**(): Invokes the window manager-specific ‘full screen’ operation

Usage: driver.fullscreen\_window ()

1. **get**(*url*): Loads a web page in the current browser session.

Usage: driver.get (url)

1. **get\_cookie**(*name*): Get a single cookie by name. Returns the cookie if found, None if not.

Usage:driver.get\_cookie(‘my\_cookie’)

1. **get\_cookies**(): Returns a set of dictionaries, corresponding to cookies visible in the current session

Usage:driver.get\_cookies()

1. **get\_log**(*log\_type*): Gets the log for a given log type

|  |  |
| --- | --- |
| **Args:** | * log\_type: type of log that which will be returned |
| **Usage:** | driver.get\_log(‘browser’) driver.get\_log(‘driver’) driver.get\_log(‘client’) driver.get\_log(‘server’) |

1. **get\_screenshot\_as\_base64**(): Gets the screenshot of the current window as a base64 encoded string which is useful in embedded images in HTML.

Usage: driver.get\_screenshot\_as\_base64()

1. **get\_screenshot\_as\_file**(*filename*): Saves a screenshot of the current window to a PNG image file. Returns False if there is any IOError, else returns True. Use full paths in your filename.

|  |  |
| --- | --- |
| **Args:** | * filename: The full path you wish to save your screenshot to. This should end with a *.png* extension. |
| **Usage:** | driver.get\_screenshot\_as\_file(‘/Screenshots/foo.png’) |

1. **get\_screenshot\_as\_png**(): Gets the screenshot of the current window as a binary data.

Usage: driver.get\_screenshot\_as\_png()

1. **get\_window\_position**(*windowHandle='current'*): Gets the x,y position of the current window.

Usage: driver.get\_window\_position()

1. **get\_window\_rect**(): Gets the x, y coordinates of the window as well as height and width of the current window.

Usage: driver.get\_window\_rect()

1. **get\_window\_size**(*windowHandle='current'*): Gets the width and height of the current window.

Usage: driver.get\_window\_size()

1. **maximize\_window**(): Maximizes the current window that WebDriver is using.

Usage: maximize\_window ()

1. **minimize\_window**(): Invokes the window manager-specific ‘minimize’ operation

Usage: minimize\_window ()

1. **quit**(): Quits the driver and closes every associated window.

Usage: driver.quit()

1. **refresh**(): Refreshes the current page.

Usage: driver.refresh()

1. **save\_screenshot**(*filename*): Saves a screenshot of the current window to a PNG image file. Returns False if there is any IOError, else returns True. Use full paths in your filename.

|  |  |
| --- | --- |
| **Args:** | * filename: The full path you wish to save your screenshot to. This should end with a *.png* extension. |
| **Usage:** | driver.save\_screenshot(‘/Screenshots/foo.png’) |

1. **set\_page\_load\_timeout**(*time\_to\_wait*): Set the amount of time to wait for a page load to complete before throwing an error.

|  |  |
| --- | --- |
| **Args:** | * time\_to\_wait: The amount of time to wait |
| **Usage:** | driver.set\_page\_load\_timeout(30) |

1. **set\_script\_timeout**(*time\_to\_wait*): Set the amount of time that the script should wait during an execute\_async\_script call before throwing an error.

|  |  |
| --- | --- |
| **Args:** | * time\_to\_wait: The amount of time to wait (in seconds) |
| **Usage:** | driver.set\_script\_timeout(30) |

1. **set\_window\_position**(*x*, *y*, *windowHandle='current'*): Sets the x,y position of the current window. (window.moveTo)

|  |  |
| --- | --- |
| **Args:** | * x: the x-coordinate in pixels to set the window position * y: the y-coordinate in pixels to set the window position |
| **Usage:** | driver.set\_window\_position(0,0) |

1. **set\_window\_rect**(*x=None*, *y=None*, *width=None*, *height=None*): Sets the x, y coordinates of the window as well as height and width of the current window.

|  |  |
| --- | --- |
| **Usage:** | driver.set\_window\_rect(x=10, y=10) driver.set\_window\_rect(width=100, height=200) driver.set\_window\_rect(x=10, y=10, width=100, height=200) |

1. **set\_window\_size**(*width*, *height*, *windowHandle='current'*): Sets the width and height of the current window. (window.resizeTo)

|  |  |
| --- | --- |
| **Args:** | * width: the width in pixels to set the window to * height: the height in pixels to set the window to |
| **Usage:** | driver.set\_window\_size(800,600) |

1. **current\_url:** Gets the URL of the current page.

Usage: driver.current\_url()

1. **current\_window\_handle:** Returns the handle of the current window.

Usage: driver.current\_window\_handle()

1. **page\_source:** Gets the source of the current page.

Usage: driver.page\_source()

1. **title:** Returns the title of the current page.

Usage: driver.title()

1. **window\_handles:** Returns the handles of all windows within the current session.

Usage: driver.window\_handles()

1. **clear**(): Clears the text if it’s a text entry element.

Usage: driver.clear()

1. **click**(): Clicks the element.

Usage: driver.click()

1. **get\_attribute**(*name*): Gets the given attribute or property of the element
2. **get\_property**(*name*): Gets the given property of the element.

|  |  |
| --- | --- |
| **Args:** | * name - Name of the property to retrieve. |

1. **is\_displayed**(): Whether the element is visible to a user.
2. **is\_enabled**(): Returns whether the element is enabled.
3. **is\_selected**(): Returns whether the element is selected. Can be used to check if a checkbox or radio button is selected.
4. **submit**(): Submits a form.