Python selenium commands cheat sheet

The Blog To Learn Selenium and Test Automation

Python selenium commands cheat sheet

admin Python Selenium Test Automation April 13, 2018 | 3

Frequently used python selenium commands – Cheat Sheet

To import webdriver module in python use below import statement

```python
from selenium import webdriver
```

Driver setup:
Firefox:
firefoxdriver = webdriver.Firefox(executable_path="Path to Firefox driver")

To download: Visit GitHub

Chrome:
chromedriver = webdriver.Chrome(executable_path="Path to Chrome driver")

To download: Visit Here

Internet Explorer:
iedriver = webdriver.IE(executable_path="Path To IEDriverServer.exe")

To download: Visit Here

Edge:
edgedriver = webdriver.Edge(executable_path="Path To MicrosoftWebDriver.exe")

To download: Visit Here

Opera:
operadriver = webdriver.Opera(executable_path="Path To operadriver")

To download: visit GitHub

Safari:

SafariDriver now requires manual installation of the extension prior to automation

Browser Arguments:

–headless

To open browser in headless mode. Works in both Chrome and Firefox browser

–start-maximized

To start browser maximized to screen. Requires only for Chrome browser. Firefox by default starts maximized

–incognito

To open private chrome browser

–disable-notifications

To disable notifications, works Only in Chrome browser

Example:

Python

```python
from selenium import webdriver
from selenium.webdriver.chrome.options import Options

options = Options()
options.add_argument("--headless")
options.add_argument("--start-maximized")
options.add_argument("--disable-notifications")
options.add_argument("--incognito")

driver = webdriver.Chrome(chrome_options=options, executable_path="Path to driver")
```

or

Python

```python
from selenium import webdriver
from selenium.webdriver.chrome.options import Options

options = Options()
options.add_argument("--incognito","--start-maximized","--headless")
driver = webdriver.Chrome(chrome_options=options, executable_path="Path to driver")
```

To Auto Download in Chrome:

Python

```python
from selenium import webdriver

options = webdriver.ChromeOptions()
options.add_argument("download.default_directory=")

driver = webdriver.Chrome(chrome_options=options, executable_path="Path to chrome driver")
```

To Auto Download in Firefox:

Python

```python
from selenium import webdriver
from selenium.webdriver.firefox.options import Options

firefoxOptions = Options()
firefoxOptions.set_preference("browser.download.folderList",2)
firefoxOptions.set_preference("browser.download.manager.showWhenStarting", False)
firefoxOptions.set_preference("browser.download.dir","/data")
firefoxOptions.set_preference("browser.helperApps.neverAsk.saveToDisk", "application/octet-stream,application/vnd.ms-excel")

firefoxdriver = webdriver.Firefox(firefox_options=firefoxOptions, executable_path="Path to firefox driver")
```

We can add any MIME types in the list. MIME for few types of files are given below.

1. Text File (.txt) – text/plain

2. PDF File (.pdf) – application/pdf
. CSV File (.csv) – text/csv or “application/csv”
4. MS Excel File (.xlsx) – application/vnd.openxmlformats-officedocument.spreadsheetml.sheet or application/vnd.ms-excel
5. MS word File (.docx) – application/vnd.openxmlformats-officedocument.wordprocessingml.document
   Zip file (.zip) – application/zip

Note:
The value of browser.download.folderList can be set to either 0, 1, or 2.

0 – Files will be downloaded on the user’s desktop.
1 – Files will be downloaded in the Downloads folder.
2 – Files will be stored on the location specified for the most recent download

Disable notifications in Firefox

firefoxOptions.set_preference(“dom.webnotifications.serviceworker.enabled”, false);
firefoxOptions.set_preference(“dom.webnotifications.enabled”, false);

Open specific Firefox browser using Binary:

Python

```python
from selenium import webdriver
from selenium.webdriver.firefox.firefox_binary import FirefoxBinary

binary = FirefoxBinary('path/to/binary')
driver = webdriver.Firefox(firefox_binary=binary)
```

Open specific Chrome browser using Binary:
from selenium import webdriver
from selenium.webdriver.chrome.options import Options

options = Options()
options.binary_location = “”
driver = webdriver.Chrome(chrome_options=options, executable_path=””)
driver.get(‘http://google.com/’)

Read Browser Details:

driver.title
driver.window_handles
driver.current_window_handles
driver.current_url
driver.page_source

Go to a specified URL:

driver.get(“http://google.com”)
driver.back()
driver.forward()
driver.refresh()

Locating Elements:

driver.find_element_by_ – To find the first element matching the given locator argument. Returns a WebElement

driver.find_elements_by_ – To find all elements matching the given locator argument. Returns a list of WebElement

By ID

<input id=”q” type=”text” />

element = driver.find_element_by_id(“q”)

By Name

<input id=”q” name=”search” type=”text” />

element = driver.find_element_by_name(“search”)

By Class Name

<div class=”username” style=”display: block;”>…

element = driver.find_element_by_class_name(“username”)

By Tag Name

<div class=”username” style=”display: block;”>…

element = driver.find_element_by_tag_name(“div”)

By Link Text

Refresh

element = driver.find_element_by_link_text(“Refresh”)

By Partial Link Text

Refresh Here

element = driver.find_element_by_partial_link_text(“Refresh”)

By XPath

<form id=”testform” action=”submit” method=”get”>

Username: <input type=”text” />
Password: <input type=”password” />

</form>

element = driver.find_element_by_xpath(“//form[@id=’testform’]/input[1]”)

By CSS Selector

<form id=”testform” action=”submit” method=”get”>

<input class=”username” type=”text” />
<input class=”password” type=”password” />

</form>

element = driver.find_element_by_css_selector(“form#testform>input.username”)

Important Modules to Import:

from selenium import webdriver
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.ui import Select

from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains

from selenium.common.exceptions import NoSuchElementException

from selenium.webdriver.firefox.firefox_binary import FirefoxBinary

from selenium.webdriver.chrome.options import Options
from selenium.webdriver.firefox.options import Options

Python Selenium commands for operation on elements:

button/link/image:

click()
get_attribute()

is_displayed()
_enabled()

Text field:

send_keys()
clear()

Checkbox/Radio:

is_selected()
click()

Select:

Find out the select element using any element locating strategies and then select options from list using index, visible text or option value.

Python

```python
select = Select(driver.find_element_by_id(""))

select.select_by_index(1)
select.select_by_value("") # pass value
select.select_by_visible_text("") # pass visible text
```

Element properties:

is_displayed()
is_selected()
is_enabled()

These methods return either true or false.

Read Attribute:

get_attribute("")

Get attribute from a disabled text box

driver.find_element_by_id("id").get_attribute("value");

Screenshot:

Python

```python
from selenium import webdriver

driver = webdriver.Firefox(executable_path='[Browser Driver Path]')
driver.get('[URL to Open]')

driver.get_screenshot_as_file('sample_screenshot_2.png')
driver.save_screenshot('sample_screenshot_1.png')
```

Note: An important note to store screenshots is that save_screenshot('filename') and get_screenshot_as_file('filename') will work only when extension of file is '.png'. Otherwise content of the screenshot can't be viewed

Read articles for more details about taking screenshot and element screenshot

The list here contains mostly used python selenium commands but not exhaustive. Please feel free to add in comments if you feel something is missing and should be here.

PYTHON SELENIUM | CHEAT SHEET