Selenium Básico

class ActionChains.

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_su
bmenu).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()
```

Autor: Reinaldo M. R. Junior

HTML5 Drag and Drop

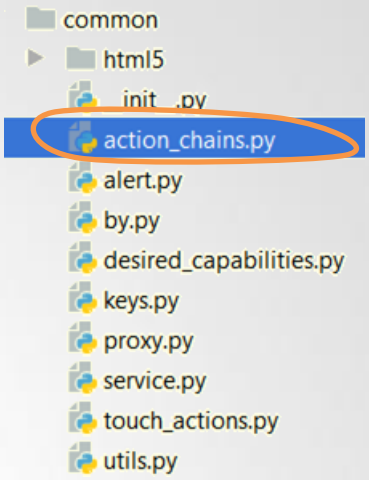
Drag the W3Schools image into the rectangle:

HTML Conteúdo Drag and Drop :

```
<!DOCTYPE HTML>
<html>

<body>

<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)"></div>

<img id="drag1" src="img_logo.gif" draggable="true"
ondragstart="drag(event)" width="336" height="69">


</body>
</html>
```

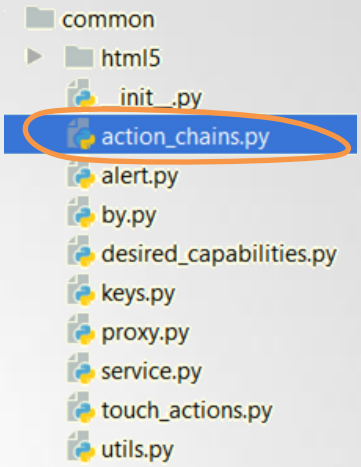
Imagem visível para o usuário.

Sobre a classe ActionChains.



- A classe ActionChains fornece as funções, pra automatizar interações de baixo nível, como movimentos do mouse, ações do botão do mouse, pressionamento de teclas e interações do menu de contexto.
- A classe ActionChains é útil para fazer ações mais complexas, como passar o mouse sobre um determinado elemento arrastando o mesmo e soltar sobre um determinado campo.

Usando a classe ActionChains.



- Primeiro temos que Importar a classe action chains:

```
from selenium.webdriver.common.action_chains import ActionChains
```

- A classe ActionChains faz parte do pacote common do webdriver, ou seja temos que importar a classe e instaciar passando o driver do selenium a mesma.

Driver do Selenium — Web Elemento — Web Elemento

```
ActionChains(driver).move_to_element(menu).click(hidden_submenu).perform()
```

Classe — Funções da Classe

Usando a classe ActionChains.

Opções da classe ActionChains :

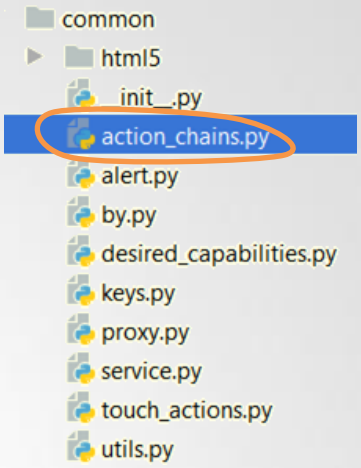
- **perform**() - Executa todas as ações armazenadas.
- **click**() - Clica em um determinado elemento.
- **click_and_hold**() - Mantém pressionado o botão esquerdo do mouse em um elemento.
- **context_click**() - Executa um contexto-clique (clique com botão direito) em um elemento.
- **double_click**() - Clique duas vezes em um elemento.
- **drag_and_drop**() - Mantém pressionado o botão esquerdo do mouse no elemento de origem e, em seguida, move para o elemento de destino e libera o botão do mouse.
- **drag_and_drop_by_offset**() - Mantém pressionado o botão esquerdo do mouse no elemento de origem e, em seguida, move para o deslocamento de destino e libera o botão do mouse.
- **key_down**() - Envia somente uma tecla pressionada, sem soltá-la. Só deve ser usado com teclas modificadoras (Control, Alt e Shift).

Usando a classe ActionChains.

Opções da classe **ActionChains** :

- **key_up**() - Libera uma tecla modificadora.
- **move_by_offset**() - Movendo o mouse da atual posição para uma outra posição.
- **move_to_element**() - Move o mouse para o meio do elemento.
- **move_to_element_with_offset**() - Move o mouse por um deslocamento do elemento especificado. Os deslocamentos são relativos ao canto superior esquerdo do elemento.
- **release**() - Liberar o botão do mouse sobre o elemento.
- **send_keys**() - Envia os caracteres para a atual foco do elemento.
- **send_keys_to_element**() - Envia os caracteres para o elemento.

Usando a classe **ActionChains.**



Opções da classe **ActionChains** via Pycharm:

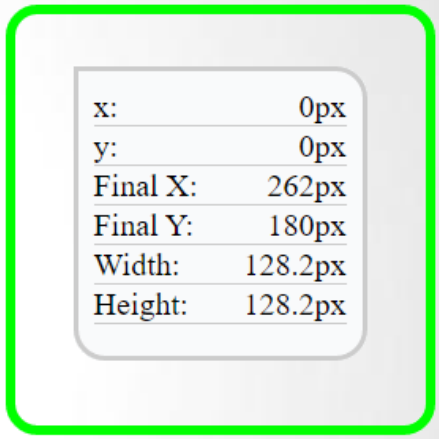
```
from selenium.webdriver.common.action_chains import ActionChains
from selenium import webdriver

driver = webdriver.Chrome()
ActionChains(driver).
```

```
click(self, on_element)                    ActionChains
click_and_hold(self, on_element)           ActionChains
context_click(self, on_element)            ActionChains
double_click(self, on_element)             ActionChains
drag_and_drop(self, source, target)        ActionChains
drag_and_drop_by_offset(self, source, x...  ActionChains
key_down(self, value, element)             ActionChains
key_up(self, value, element)               ActionChains
move_by_offset(self, xoffset, yoffset)     ActionChains
move_to_element(self, to_element)          ActionChains
```

Ctrl+Abaixo and Ctrl+Acima will move caret down and up in the editor >>

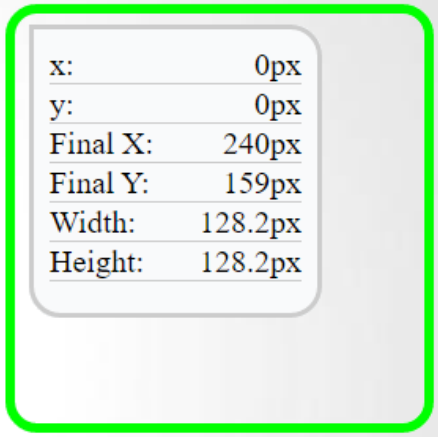
Exemplo 01 - drag_and_drop

```
from selenium.webdriver import ActionChains
from selenium import webdriver

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://jsfiddle.net/davidThomas/DGbT3/3/")
driver.implicitly_wait(15)

driver.find_element_by_css_selector("#run").click()
driver.switch_to.frame("result")
# Elemento que eu desejo arrastar.
source = driver.find_element_by_css_selector("#dragThis")
# Destino do meu elemento.
target = driver.find_element_by_css_selector("#dropHere")

# Instaciamos a classe passando o driver.
actions = ActionChains(driver)
# Utilizamos a funcao drag_and_drop,  passando os elementos.
actions.drag_and_drop(source,  target)
# Executamos as acoes.
actions.perform()
```



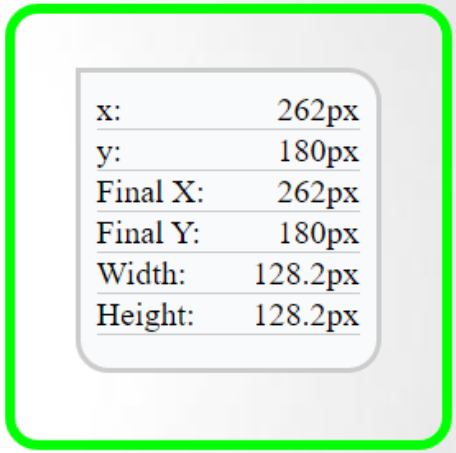
Exemplo 02 - drag_and_drop_by_offset

```
driver.get("http://jsfiddle.net/davidThomas/DGbT3/3/")
driver.implicitly_wait(15)

driver.find_element_by_css_selector("#run").click()
driver.switch_to.frame("result")

source = driver.find_element_by_css_selector("#dragThis")
target = driver.find_element_by_css_selector("#dropHere")
# Pegamos a localizacao x e y do elemento.
location1 = int(target.location['x'])
location2 = int(target.location['y'])
actions = ActionChains(driver)
# Passamos o elemento e a localizacao x e y do elemento de destino.
actions.drag_and_drop_by_offset(source, location1+10, location2+10)
actions.perform()
```



Exemplo 03 – Passo a Passo

```
from selenium.webdriver import ActionChains
from selenium import webdriver

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://jsfiddle.net/davidThomas/DGbT3/3/")
driver.implicitly_wait(15)

driver.find_element_by_css_selector("#run").click()
driver.switch_to.frame("result")

source3 = driver.find_element_by_css_selector("#dragThis")
target3 = driver.find_element_by_css_selector("#dropHere")

actions = ActionChains(driver)
# Clica e segura o elemento.
actions.click_and_hold(source3)
# Move o elemento ateh seu alvo.
actions.move_to_element(target3)
# Libera o mouse sobre o alvo.
actions.release(target3)
actions.perform()
```



Dificuldades

- Em páginas HTML5 não funciona via Selenium o drag and drop, para isso devemos fazer algumas soluções via javascript, vejamos um exemplo no meu github.
- https://gist.github.com/reinaldorossetti/074642e9f954b7119c557748836fcd42

Dificuldades

- No Selenium você não consegue mover o curso do mouse, o selenium somente interage com os elementos HTML, não conseguindo muitas vezes interagir com o browser, com python conseguimos realizar essas ações, exemplo:
- https://github.com/reinaldorossetti/tests_solution/blob/master/test_move_cursor.py

Referências:

[0] http://selenium-python.readthedocs.io/api.html?highlight=actions%20chain#module-selenium.webdriver.common.action_chains

[1] http://www.w3schools.com/html/html5_draganddrop.asp

[2] http://elementalselenium.com/tips/39-drag-and-drop

[3] http://www.guru99.com/keyboard-mouse-events-files-webdriver.html