当需要模拟键盘或者鼠标操作时，Python需要使用 ActionChains 来处理，Java需要 Actions 来处理。

常用模拟鼠标的行为，比如单击，双击，拖动等。当调用 ActionChains 或者 Actions 的方法时，会将所有操作按顺序存入队列，当调用 perform() 方法时，队列中的事件会依次执行。

引入依赖

- Python 版本

```
# 引入依赖from selenium.webdriver import ActionChains
```

- Java版本

```
import org.openqa.selenium.interactions.Actions;
```

实战演示

点击相关操作

下面代码中，action是模拟键盘或者鼠标的实例对象，on_element 是需要传递一个元素进去，默认值为None。

单击指定元素，如果不指定，会单击当前光标的位置

- Python 版本

```
action.click(on_element=None)
```

- Java版本

```
Actions action = new Actions(webDriver);action.click(on_element=None);
```

长按某个元素

- Python 版本

```
action.click_and_hold(on_element=None)
```

- Java版本

```
Actions action = new Actions(webDriver);action.clickAndHold(on_element=None);
```

执行右键操作

- Python 版本

```
action.context_click(on_element=None)
```

- Java版本

```
Actions action = new Actions(webDriver);action.contextClick(on_element=None);
```

执行左键双击

- Python 版本

```
action.double_click(on_element=None)
```

- Java版本

```
Actions action = new Actions(webDriver);action.doubleClick(on_element=None);
```

拖拽起始的元素到目标元素，即 source 到 target

- Python 版本

```
action.drag_and_drop(source, target)
```

- Java版本

```
Actions action = new Actions(webDriver);action.dragAndDrop(WebElement source,
WebElement target);
```

将目标拖动到指定的位置

- Python 版本

```
# xoffset 和 yoffset 是相对于 source 左上角为原点的偏移量
action.drag_and_drop_by_offset(source, xoffset, yoffset)
```

- Java版本

```
Actions action = new Actions(webDriver);actions.dragAndDropBy(WebElement source, int
xOffset, int yOffset);
```

按键

使用这个方法可以方便的实现某些组合键盘事件，比如按下 ctrl+c 键。

- Python 版本

```
action.key_down(value, element=None)
```

- Java版本

```
Actions action = new Actions(webDriver);actions.keyDown(element, value);
```

松开某个键，可以配合上面的方法实现按下 ctrl+c 并且释放

- Python 版本

```
ActionChains(driver).key_down(Keys.CONTROL)\
.send_keys('c').key_up(Keys.CONTROL).perform()
```

- Java版本

```
Actions action = new
Actions(webDriver);action.keyDown(Keys.CONTROL).sendKeys("c").keyUp(Keys.CONTROL).perfor
m();
```

其他按键请参考：https://python-selenium-zh.readthedocs.io/zh_CN/latest/7.4%20%E7%89%B9%E6%AE%8A%E5%AD%97%E7%AC%A6/

github 参考地址：https://github.com/SeleniumHQ/selenium/blob/916168f403dded05f878fe189d68c0f9152335c9/py/selenium/webdriver/common/keys.py

移动

指定光标移动到某一个位置，需要给出两个坐标位置

- Python 版本

```
# xoffset 和 yoffset 是相对于网页左上角的偏移量action.move_by_offset(xoffset, yoffset)
```

- Java版本

```
Actions action = new Actions(webDriver);action.moveByOffset(xOffset,yOffset);
```

将鼠标移动到指定元素的位置

- Python 版本

```
action.move_to_element(to_element)
```

- Java版本

```
Actions action = new Actions(webDriver);action.moveToElement(to_element);
```

移动鼠标到相对于某个元素的偏移位置

- Python 版本

```
# xoffset 和 yoffset 是相对于 to_element 左上角的偏移量
action.move_to_element_with_offset(to_element, xoffset, yoffset)
```

- Java版本

```
Actions action = new Actions(webDriver);action.moveToElement(to_element, xOffset, yOffset);
```

其它

执行 ActionChains 中的操作

前面介绍的方法会将所有操作按顺序存入队列，要执行这些操作，需要调用 [redacted] 方法。

- Python 版本

```
action.move_to_element_with_offset(to_element, xoffset, yoffset).perform()
```

- Java版本

```
Actions action = new Actions(webDriver);action.moveToElement(to_element, int xOffset, int yOffset).perform();
```

释放按下的鼠标

- Python 版本

```
action.release(on_element=None)
```

- Java版本

```
Actions action = new Actions(webDriver);action.release(on_element=None)
```

向焦点元素位置输入值

焦点元素：使用 tab 键，那些被选中的元素就是焦点元素。

- Python 版本

```
action.send_keys(*keys_to_send)
```

- Java版本

```
Actions action = new Actions(webDriver);action.sendKeys(*keys_to_send)
```

向指定的元素输入数据

- Python 版本

```
action.send_keys_to_element(element, *keys_to_send)
```

- Java版本

```
Actions action = new Actions(webDriver);action.sendKeys(element,keys_to_send);
```