selenium的基本使用

1.无头模式/无界面模式/后台运行

```
from selenium import webdriver
from selenium.webdriver import
ChromeOptions
option = ChromeOptions()
# 无界面启动，也可以直接设置
options.headless=True
option.add_argument("--headless") # 指定无头
模式
browser = webdriver.Chrome(options=option)
# 获取浏览器大小
browser.set_window_size(1920, 1080)
# 访问csdn，发送请求
browser.get("https://wwww.baidu.com")
print(browser.page_source)
```

2.加载网页后的常见操作：

```
from selenium import webdriver
# 创建driver对象
driver = webdriver.Chrome()
# 访问的起始的url地址
start_url = 'https://www.baidu.com'
# 访问
```

```
driver.get(url=start_url)
# 将网页生成为图片
driver.save_screenshot("长城.png")
# 根据element内容，id的属性定位
driver.find_element_by_id("kw").send_keys("
长城")
driver.find_element_by_id("su").click()
# 获取element的源码内容
driver.page_source
# 打印页面的标题
print(driver.title)
# 获取cookie
driver.get_cookies()
# 退出当前页面
driver.close()
# 退出浏览器
driver.quit()
```

3.WebDriver 操作浏览器方式

```
# 最大化浏览器
    driver.maximize_window()
# 刷新
    driver.refresh()
# 后退
    driver.back()
# 前进
    driver.forward()
# 最大化窗口
    driver.maximize_window()
```

```
# 设置浏览器大小
    driver.set_window_size(300,300)
# 设置浏览器位置
    driver.set_window_position(300,200)
# 关闭浏览器单个窗口
    driver.close()
# 关闭浏览器所有窗口
    driver.quit()
```

4.标签定位方法

```
from selenium import webdriver
driver = webdriver.Chrome()
# 窗口最大化
driver.maximize_window()
url = 'https://www.baidu.com'
driver.get(url)
"""根据标签属性定位"""
# 根据标签属性id定位
# send_keys()  赋值，传入值，输入值
driver.find_element_by_id('kw').send_keys('
哥尔赞')
# 根据标签class属性定位
driver.find_element_by_class_name('s_ipt').
send_keys('迪迦')
# 根据标签name属性定位
driver.find_element_by_name('wd').send_keys
('金刚')
"""根据xpath语法定位"""
```

```
driver.find_element_by_xpath('//*
[@id="kw"]').send_keys('迪丽热巴')
"""根据CSS语法定位"""
driver.find_element_by_css_selector('#kw').
send_keys('林允')
```

总结:

```
driver.find_element(By.ID, 'kw')
            返回一个元素
find_element(s)(By.CLASS_NAME, 's_ipt')
  根据类名获取元素列表
find_element(s)(By.NAME, 'wd')          根据
标签的name属性值返回包含标签对象元素的列表
find_element(s)(By.XPATH,
'//input[@id="kw"]')              返回一个包含元
素的列表
find_element(s)_by_link_text          根据连
接文本获取元素列表
find_element(s)_by_partial_link_text  根据链
接包含的文本获取元素列表
find_element(s)(By.TAG_NAME)
根据标签名获取元素列表
find_element(s)_by_css                根据css
选择器来获取元素列表
```

注意:

find_element和find_elements的区别:

- 多了个s就返回列表，没有s就返回匹配到的第一个标签对象
- find_element匹配不到就抛出异常，find_elements匹配不到就返回空列表

by_link_text和by_partial_link_text的区别：

- 全部文本和包含某个文本

以上函数的使用方法:

- driver.find_element_by_id('id_str')

5.WebDriver其他常用方法

```
1. size                          返回元素大小
2. text                          获取元素的文本
3. title                          获取页面title
4. current_url                      获取当前页面URL
5. get_attribute("xxx") 获取属性值;xxx：要获取的属性
6. is_display()                        判断元素是否可见
7. is_enabled()                        判断元素是否可用
```

提示：

1. size、text、title、current_url：为属性，调用时无括号；如：xxx.size

2. title、current_url：使用浏览器实例化对象直接调用；　如：driver.title

```
# 获取用户名文本框大小

size=driver.find_element_by_id("userA").size
    print('size:',size)
# 获取a标签内容

text=driver.find_element_by_id("fwA").text
    print('a标签text:',text)
# 获取title
    title=driver.title
    print('title:',title)
# 获取当前页面url
    url=driver.current_url
    print('url:',url)
# 获取a标签href属性值

href=driver.find_element_by_id("fwA").get_attribute("href")
    print('href属性值为:',href)
# 判断span是否显示

display=driver.find_element_by_css_selector('span').is_displayed()
    print('span标签是否显示：',display)
# 判断取消按钮是否可用
```

```
enabled=driver.find_element_by_id('cancelA'
).is_enabled()
    print('取消按钮是否可用：',enabled)
```

6.driver对象的常用属性和方法

```
driver.page_source          当前标签页浏览器渲染之
后的网页源代码
driver.current_url          当前标签页的url
driver.close()              关闭当前标签页，如果只
有一个标签页则关闭整个浏览器
driver.quit()               关闭浏览器
driver.forward()            页面前进
driver.back()               页面后退
driver.screen_shot(img_name)      页面截图
```

7.WebDriver操作鼠标方法

```
1. context_click()                  右击 --> 此方法模拟鼠标右键点击效果
2. double_click()                   双击 --> 此方法模拟双标双击效果
3. drag_and_drop()                  拖动 --> 此方法模拟双标拖动效果
4. move_to_element()                悬停 --> 此方法模拟鼠标悬停效果
5. perform()                        执行 --> 此方法用来执行以上所有鼠标方法
```

示例:

```
    1. 导包：from selenium.webdriver.common.action_chains import ActionChains

        2. 实例化ActionChains对象：Action=ActionChains(driver)
        3. 调用右键方法：element=Action.context_click(username)
        4. 执行：element.perform()
```

8.常用的键盘操作

示例:

- 定位用户名
  element=driver.find_element_by_id("userA")
- 输入用户名 element.send_keys("admin1")
- 删除1 element.send_keys(Keys.BACK_SPACE)
- 全选 element.send_keys(Keys.CONTROL,'a')
- 复制 element.send_keys(Keys.CONTROL,'c')
- 粘贴
  driver.find_element_by_id('passwordA').send_keys(Keys.CONTROL,'v')

9.标签对象提取文本内容和属性值

获取文本 element.text

通过定位获取的标签对象的 text 属性，获取文本内容

获取属性值 element.get_attribute('属性名')

通过定位获取的标签对象的 get_attribute 函数，传入属性名，来获取属性的值

```
from selenium import webdriver
# 创建driver对象
driver = webdriver.Chrome()
# 访问的起始的url地址
start_url = 'https://www.csdn.net/'
# 访问
driver.get(url=start_url)
result =
driver.find_elements_by_tag_name('a')
print(result[1].text)

ret = driver.find_elements_by_link_text('收藏')
print(ret[0].get_attribute('href'))
```

- 使用： 以豆瓣首页为例:https://www.douban.com/

```
  from selenium import webdriver

  driver = webdriver.Chrome()

  driver.get("https://www.douban.com/")

  ret1 = driver.find_element_by_id("anony-nav")
  print(ret1)
```

```
  # 输出为:
<selenium.webdriver.remote.webelement.WebElement
(session="ea6f94544ac3a56585b2638d352e97f3", element="0.5335773935305805-1")>

  ret2 = driver.find_elements_by_id("anony-nav")
  print(ret2)
  #输出为:
[<selenium.webdriver.remote.webelement.WebElement
(session="ea6f94544ac3a56585b2638d352e97f3", element="0.5335773935305805-1")>]

  ret3 = driver.find_elements_by_xpath("//*[@id='anony-nav']/h1/a")
  print(len(ret3))
   #输出为: 1

  ret4 = driver.find_elements_by_tag_name("h1")
  print(len(ret4))
   #输出为: 1

  ret5 = driver.find_elements_by_link_text("下载豆瓣App")
  print(len(ret5))
   #输出为: 1
```

```
  ret6 =
driver.find_elements_by_partial_link_text("
豆瓣")
  print(len(ret6))
   #输出为：28

  driver.close()
```