Selenium开发笔记

准备工作：

* 检查是否安装PYTHON
* 下载pip
* pip install selenium安装
* 下载对应版本的chrome驱动

# -\*- coding: UTF-8 -\*-

**from selenium import webdriver**

#实例化一个对象，参数是驱动软件的地址，r表示后面是字符串

wd = **webdriver.Chrome(r'D:\Python\selenim\chromedriver.exe')**

#打开网址

wd.get('http://www.baidu.com')

打开浏览器后接下来就是对界面的输入、点击、选择；

首先要选择元素，找到了元素，才能操作它

方法1：打开F12查看元素，原则对应元素

方法2：选中后右键，点击检查

#通过id获取页面元素，返回WebElement对象

* element = **wd.find\_element\_by\_id**(‘kw’)

#通过该WebElement对象就可以对页面元素进行操作了

* element.**send\_keys**('想输入的内容')

#也可以使用点击方法

element = wd.find\_element\_by\_id('su')

element.click()

* 根据元素的class来选择html元素：

find\_elements\_by\_class\_name(‘animal’)，这里会返回多个elements，放在一个list中

for i in elements:

print(i.text) #打印文本内容

* 也可以找一个element：find\_elements\_by\_class\_name(‘animal’)
* 通过标签名寻找元素find\_elements\_by\_tag\_name
* 想要找的元素有重名或者没有id的时候，可以一层层的寻找元素；wd表示整个页面，wd.find\_element就是在整个页面内寻找，找到的element和wd性质相同，又可以继续用来查询，**搜索的范围是元素的内部**

div = wd.find\_element\_by\_class\_name('search-box-flex')

span = div.find\_elements\_by\_tag\_name('span')

for i in span:

print(i.text)

* 代码执行的速度可能比服务器相应的速度更快，导致获取不到页面上的元素：import time time.sleep(1)
* 第二种方法就是设置隐藏等待时间，这个每隔半秒会去扫描一次**wd.implicitly\_wait(10)**，最大等待时间10

------------------------------------------------**选择元素**

element.get\_attribute获取元素属性：element.get\_attribute('class')

webdriver.quit()关闭浏览器窗口和驱动窗口

获取元素对应的整个HTML：element.get\_attribute('outerHTMl') #outerHTML是一个关键属性关键词

获取输入框内容：element.get\_attribute('value') #value是关键词

使用element.text获取展示在界面上的文本内容

如果文本没有展示在界面上，可以使用element.getAttribute('innerText')获取，或者element.getAttribute('textContent')

------------------------------------------------**CSS表达式获取元素**

通过CSS选择器选择元素：find\_element\_by\_css\_selector(CSS Selector参数)

选择所有元素：find\_elements\_by\_css\_selector(CSS Selector参数)

element = wd.find\_element\_by\_css\_selector('.plant') #找到class名为plant的元素

element = wd.find\_element\_by\_class\_name('plant') #与上面这种写法等价

通过标签TAG名寻找元素：find\_element\_by\_css\_selector('标签名')

element = wd.find\_element\_by\_css\_selector('div') #寻找标签名为div的元素

通过id寻找元素：find\_element\_by\_css\_selector('#id')

element = wd.find\_element\_by\_css\_selector('#search') #寻找id为search的元素

------------------------------------------------**CSS选择器选择子元素和后代元素**

子元素vs后代元素

如果 元素2 是 元素1的直接子元素，那CSS Selector选择子元素的语法是： 元素1 > 元素2 也支持多层如：元素1>元素2>元素3>元素4

find\_element\_by\_css\_selector('#search > .em') #id为search的元素下面的类为em的元素

wd.find\_elements\_by\_css\_selector('#container > div') #寻找id为container的元素的div标签的直接子元素

如果元素2是元素1的后代元素，选择后代元素的方法： 元素1 元素2

------------------------------------------------**CSS选择器根据属性选择元素**

使用[]选择属性，('[属性名="属性值"]')

find\_elements\_by\_css\_selector('[href="https://www.baidu.com"]') #选择href属性为baidu的元素

也可以不赋属性值，只靠属性类型找到元素

find\_elements\_by\_css\_selector('[href]') #选择属性有href的元素

find\_elements\_by\_css\_selector('[id="search"]') #寻找id为search的元素

find\_elements\_by\_css\_selector('[class="em"]') #寻找类名为em的值

可以把多种限制条件组合使用：

('.plant div[name="baidu"]') #plant类后代元素div的属性name=baidu的元素

技巧:可以直接把CSS选择器表达式写到chrome的F12 element页签CTRL F搜索框中进行直接验证，这样可以避免把选择器表达式写到代码里面验证

span.copyright #标签名为span,class属性为copyright的值

------------------------------------------------**CSS选择器多条件复合选择**

用逗号连接两种条件 条件1,条件2

.plant,#container #类为plant的节点和id为container的节点

逗号优先级较低 对比#t1>span,p 和 #t1>span,#t1>p CSS选择器没有括号运算

------------------------------------------------**CSS选择器按照次序选择子节点**

选择第几个子节点： (元素1:nth-child(n)) 表示元素1的第n个子节点

(':nth-child(n)')表示第二个子节点,不带父元素

span:nth-child(2) #span的第2个子节点，也可以写作span>:nth-child(2)

选择最后第个子节点： (':nth-last-child(2)')

p:nth-last-child(2) #p标签且为倒数第二个子节点

某种类型的第n个元素：

span:nth-of-type(2) #第2个span类型的元素

span:nth-child(2) #是第二个子元素，且为span类型；或者理解为父元素的第2个span类型的元素

span:nth-last-of-type(1) #最后一个span

---------------------------------CSS选择器兄弟节点选择

相邻兄弟节点 h3+span #h3后面紧跟的兄弟节点span

后面所有兄弟节点 h3~span #h3后面的所有span节点

---------------------------------frame切换/窗口切换

frame或iframe元素内部可以包含另一份html，但是selenium默认查询当前的Html而不包含里面的Html

如果要操作被嵌入的html中的元素，就必须切换操作范围到被嵌入的文档中

webdriver.switch\_to.frame(frame\_reference) #frame\_reference为frame元素的属性name或ID

webdriver.switch\_to.frame(wd.find\_element\_by\_css\_selector('iframe[src="sample.html"]'))

切回原来的frame：wd.switch\_to.default\_content()

-----------------------------------------------------------窗口切换

打开了新窗口后，如果不进行切换窗口，那操作的还是老窗口

print(webdriver.title) #打印当前窗口的标题栏

所以要操作新开的窗口之前，需要切换到新窗口

webdriver.switch\_to.window(handle)

可以遍历所有窗口的handle，切换到每个窗口判断该窗口是不是想要的窗口，从而切换到想要的窗口

for handle in webdriver.window\_handles:

#先切换到该窗口

wd.switch\_to.window(handle)

#得到该窗口的标题字符串，判断是不是我们想要操作的那个窗口

if 'Bing' in wd.title:

#如果是，那么这时候WebDriver对象就是对应的该窗口，正好，跳出循环

break

可以保存当前窗口的句柄，方便以后再切回来

mainWindow = wd.current\_window\_handle

切到新窗口后，就可以直接像下面这样，将driver对应的对象返回原来的窗口

wd.switch\_to.window(mainWindow)

-----------------------------------------------------------选择框

常见的选择框：radio单选 checkbox多选勾选 select下拉菜单

radio：

<div id="s\_radio">

<input type="radio" name="teacher" value="小明">小明

<input type="radio" name="teacher" value="小江">小江

<input type="radio" name="teacher" value="小胡" checked="checked">小胡

</div>

#获取默认选中的元素

element = wd.find\_element\_by\_css\_selector('#s\_radio input[checked="checked"]')

print('当前选中的是：' + element.get\_attribute('value'));

#点选小江

element = wd.find\_element\_by\_css\_selector('#s\_radio input[value="小江"]')

element.click()

checkbox:

<div id="s\_checkbox">

<input type="checkbox" name="teacher" value="小明">小明

<input type="checkbox" name="teacher" value="小江">小江

<input type="checkbox" name="teacher" value="小胡" checked="checked">小胡

</div>

思路：

把已经选中的先再点一遍取消掉，再勾选想要勾选的选项，比如要勾选小江和小明

wd.find\_element\_by\_css\_selector('#s\_checkbox input[checked]').click()

wd.find\_element\_by\_css\_selector('#s\_checkbox input[value="小明"]').click()

wd.find\_element\_by\_css\_selector('#s\_checkbox input[value="小江"]').click()

select：

selenium为下拉菜单select提供了一种select类，专门处理select元素

<select id="ss\_single">

<option value="小明">小明</option>

<option value="小江">小江</option>

<option value="小胡" selected="selected">小胡</option>

</select>

根据value属性选择：

<option value="foo">foo</option>

三种方法：

1.根据选项的可见文本，选择元素

select\_by\_visible\_text()

2.根据选项的次序（从1开始），选择元素

select\_by\_index()

3.根据选项的value属性值，选择元素

select\_by\_value()

deselect\_all()去掉所有选中的元素

#导入Select类

from selenium.webdriver.support.ui import Select

#创建Select对象

select = Select(wd.find\_element\_by\_id("ss\_single"))

#通过Select对象选中小江

select.select\_by\_visible\_text("小江")

-----------------------------------------------------------更多动作

比如鼠标右键点击、双击、移动鼠标到某个元素、鼠标拖拽等

这些操作可以通过Selenium提供的ActionChains类来实现，这个类提供了一系列特殊动作的模拟

\*\*\*\*此处应有截图\*\*\*\*

下面模拟鼠标放到页面某个元素上：

wd.get("https://www.baidu.com")

from selenium.webdriver.common.action\_chains import ActionChains

#初始化ActionChains对象

ac = ActionChains(wd)

#鼠标放到元素上

ac.move\_to\_element(wd.find\_element\_by\_css\_selector('[name="tj\_briicon"]')).perform()

find\_element可以使用多种方式查询如xpath、css\_selector，默认BY\_ID