码同学

Selenium基础

授课老师：沙陌老师
微信:Matongxue_2

Selenium是什么

- Selenium 是ThoughtWorks专门为Web应用程序编写的一个验收测试工具。
- Selenium测试直接运行在浏览器中，就像真正的用户在操作一样。支持的浏览器包括Chrome 、 IE、Mozilla Firefox等。
- Selenium的主要功能包括：
  - 功能验证测试可以模拟用户操作，验证待测系统是否符合预期
  - 兼容性测试，验证待测应用能否正常工作在不同浏览器和操作系统上。

官网：

https://www.selenium.dev/

Selenium环境搭建

1. python安装(已经安装)
2. 安装selenium依赖库

   pip install selenium
3. 安装浏览器chrome(课上采用)，安装完以后请记得禁止chrome的更新
4. 下载浏览器对应的驱动文件(注意必须版本对应)

   chromedriver:

   http://npm.taobao.org/mirrors/chromedriver/

   geckodriver:

   https://firefox-source-docs.mozilla.org/testing/geckodriver/Support.html

   edgedriver:

   https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/#downloads

   iedriver:

   https://selenium-release.storage.googleapis.com/index.html

5. 配置浏览器驱动文件到环境变量path中， 检查chromedriver环境变量配置

```
D:\1码同学\付费课\mock测试>chromedriver
Starting ChromeDriver 76.0.3809.12 (220b19a666554bdcac56dff9ffd44c300842c933-refs/branch-heads/3809@{#83}) on port 9515
Only local connections are allowed.
Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.
```

webdriver对象

web自动化测试中核心对象为WebDriver对象，因此在任何脚本开始之前都必须先创建driver对象，下面是针对chrome浏览器创建的driver对象代码：

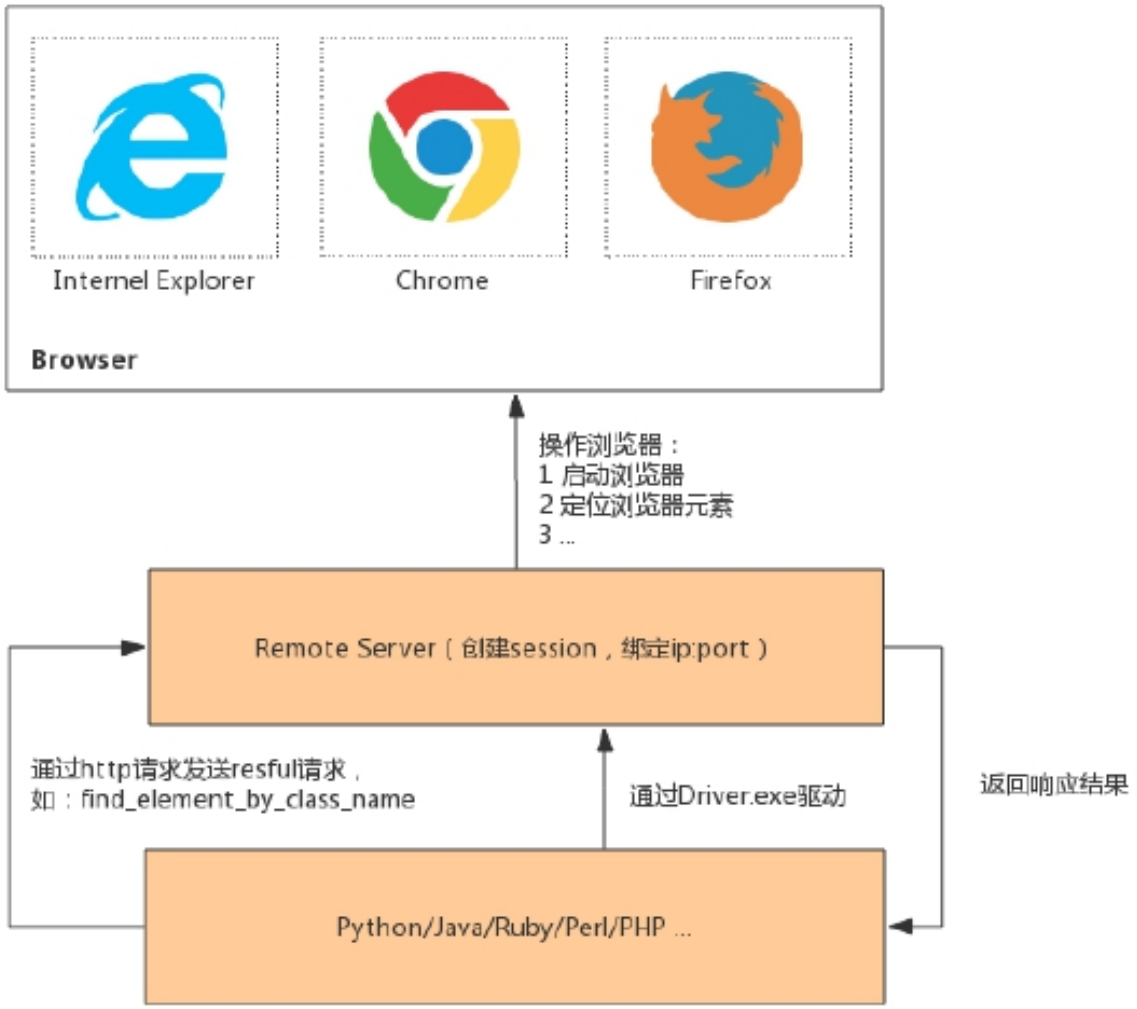
在pycharm中创建项目，执行如下代码，能打开浏览器和百度网站，则说明环境基本没问题

```
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.get('http://www.baidu.com')
```

Selenium工作原理

对于每一条Selenium脚本，一个http请求会被创建并且发送给浏览器的驱动，
浏览器驱动中包含了一个HTTP Server，用来接收这些http请求，
HTTP Server接收到请求后根据请求来具体操控对应的浏览器，
浏览器执行具体的测试步骤，
浏览器将步骤执行结果返回给HTTP Server，
HTTP Server又将结果返回给Selenium的脚本，如果是错误的http代码我们就会在控制台看到对应的报错信息。



元素查找工具-ChromeF12

码同学

Chrome浏览器的开发者工具是web自动化测试查看元素的必备工具

打开方式：

1. 界面上右键-->检查
2. 浏览器设置里-->更多工具-->开发者工具
3. 快捷键 F12

元素查看调试方式：

1. 选中开发者工具的Elements标签，这里所看到的就是当前页面的html源码
2. 点击开发者工具左上角的小箭头，然后选择界面上的某一个元素，则Elements中会自动展示出该元素的源码信息
3. 按快捷键ctrl+f 后在开发者工具下方会显示调试查找框，可以调试xpath和css表达式

八大定位

码同学

定位方式	对应元素	代码
id	元素的id属性:	driver.find_element_by_id('username')
name	元素的name属性:	driver.find_element_by_id('pwd')
className	元素的class属性:	find_element_by_class_name('text')
linktext	a标签的文字:<a>登录	driver.find_element_by_link_text('登录')
partialLinkText	使用a标签的文字进行模糊匹配定位:<a>这个链接很长	driver.find_element_by_partial_link_text('链接')
tagName	标签名称:<button>	driver.find_element_by_tag_name('button')
cssSelector	css 表达式，任意元素	driver.find_element_by_css_selector('xx')
xpath	xpath表达式，任意元素	driver.find_element_by_xpath('xx')

css定位

码同学

	表达式	含义
使用绝对路径定位	html>body>div>nav	从根html一路查找到nav，>表示路径层级
使用相对路径定位	nav div nav	下的所有div，空格表示相对层级
使用class名称定位	.form-control	定位class属性为form-control的元素
使用id值定位	#top-search-form	定位id值为top-search-form的元素
使用属性定位元素	input[name='word']	定位name属性为word的input标签元素
	input[name='word'][type='text']	多个属性一起定位
使用属性值的一部分定位元素	div[id^='sendTo']	以sendTo开头的id属性的div元素
	div[id$='message']	以message结尾的id属性的div元素
	div[id*='model']	id属性值包含model的div元素
定位同级兄弟元素	button + div	定位与button同级的元素，button之后的
使用伪类定位元素	li:first-child	所有同级li元素中的第一个
	li:nth-child(2)	所有同级li元素中的第n个
	li:last-child	所有同级li元素中的最后一个

xpath定位

码同学

W3C Xpath: http://www.w3school.com.cn/xpath/xpath_syntax.asp

表达式	描述
/	从根节点选取。
//	从匹配选择的当前节点选择文档中的节点，而不考虑它们的位置。
..	选取当前节点的父节点
@	选取属性。
*	匹配任何节点
//title[@lang]	选取所有拥有名为 lang 的属性的 title 元素
/bookstore/book[1]	选取属于 bookstore 子元素的第一个 book 元素
/bookstore/book[last()]	选取属于 bookstore 子元素的最后一个 book 元素
/bookstore/book[position()<3]	选取最前面的两个属于 bookstore 元素的子元素的 book 元素
//*[text()='文本']	选取文字为文本的所有元素
//div[contains(text(),'xxx')]	选取text属性包含xxx的div元素

xpath定位

码同学

轴名称	结果
ancestor	选取当前节点的所有先辈（父、祖父等）
ancestor-or-self	选取当前节点的所有先辈（父、祖父等）以及当前节点本身
parent	选取当前节点的父节点
child	选取当前节点的所有子元素
descendant	选取当前节点的所有后代元素（子、孙等）
descendant-or-self	选取当前节点的所有后代元素（子、孙等）以及当前节点本身
following	选取文档中当前节点的结束标签之后的所有节点
following-sibling	选取当前节点之后的所有同级节点。
preceding	选取文档中当前节点的开始标签之前的所有节点
preceding-sibling	选取当前节点之前的所有同级节点

键盘鼠标操作

案例：新增收货地址(鼠标悬浮)

使用ActionChains类模拟以下操作：

action = ActionChains(driver)

- 模拟键盘输入 action.send_keys('abcd').perform()
- 模拟鼠标右键 action.context_click(element).perform()
- 模拟鼠标悬浮 action.move_to_element(element).perform()
- 模拟双击键 action.double_click(element).perform()
- 模拟Tab键 action.send_keys(Keys.TAB).perform()
- 模拟回车键 action.send_keys(Keys.ENTER).perform()
- 模拟ctrl+c action.send_keys(Keys.CONTROL,'c').perform()
- 模拟拖拽 action.drag_and_drop_by_offset(element,0,50).perform() #向下拖拽

多window切换

案例：提交订单(先搜索"牛奶",然后再点立即购买)

遇到元素在新打开的页面上时，需要先将driver切换至window，然后再定位

1. 先获取所有窗口的句柄

   windows = driver.window_handles

2. 根据窗口顺序索引进行driver切换

   driver.switch_to.window(windows[1])#切换到第二个窗口

3. 如果窗口较多，顺序不好确认，可以遍历所有窗口，根据目标窗口特性进行切换，比如目标窗口的title值、url地址、特性元素或文字

三种等待

- 隐式等待

driver.implicitly_wait(10)

1. 隐式等待可以在指定时间内，不断的查找元素，一旦找到立刻结束查找，继续执行下行代码

2. 隐式等待是一个全局性的等待，仅仅针对findelement查找方法生效，同时可以随时更改

三种等待

- 显式等待

1. 显式等待是针对单一元素或一组元素进行智能等待，通过expected_conditions选择等待条件

wait = WebDriverWait(driver,10)

wait.until(expected_conditions.element_to_be_clickable(By.ID,'xx')) #直到条件满足

wait.until_not(expected_conditions.element_to_be_clickable(By.ID,'xx'))#直到条件不满足

以下条件并不是所有，仅仅列出一些

条件	含义
expected_conditions.element_to_be_clickable(By.ID,'xx')	判断某个元素中是否可见并且是enable的，代表可点击，返回元素对象
expected_conditions.visibility_of_element_located(By.ID,'xx')	判断某个元素是否被添加到了dom里并且可见，可见代表元素可显示且宽和高都大于0，返回元素对象

三种等待

条件	含义
expected_conditions.presence_of_element_located(By.ID,'xx')	判断某个元素是否被加到了dom树里，并不代表该元素一定可见，返回元素对象
expected_conditions.element_to_be_selected(By.ID,'xx')	判断某个元素是否被选中了,一般用在原生select下拉列表，返回布尔值
expected_conditions.element_selection_state_to_be(element,True)	判断某个元素的选中状态是否符合预期,返回布尔值
expected_conditions.text_to_be_present_in_element((By.ID,'xx'),'文本')	判断指定元素文字是否包含了预期的字符串，返回布尔值
expected_conditions.text_to_be_present_in_element_value((By.ID,'xx'),'文本')	判断指定的元素value属性值是否包含了预期的字符串，返回布尔值
expected_conditions.new_window_is_opened(windows)	判端新窗口是否打开，返回布尔值
expected_conditions.presence_of_all_elements_located(By.ID,'xx')	找到至少一个元素，返回元素列表对象
expected_conditions.frame_to_be_available_and_switch_to_it(By.ID,'xx')	判端frame可用并且切换过去

截图方法

码同学

说明：在Selenium中，提供了截图方法，我们只需要调用即可

方法：

整屏截图：driver.get_screenshot_as_file(imgpath) #imgpath：图片保存路径

元素截图：element.screenshot(imgpath)

案例：个人信息里的生日修改

在实际操作过程中有一些控件没有无法直接操作，需要借助js来完成，比如时间控件、隐藏的select元素、只读输入框等等

js1=" document.getElementById('s').value='1987-10-12';" //修改某元素的value值

js2=" document.getElementsByClassName('s')[0].click();" //点击某一个元素

js3=" document.getElementsByClassName('s')[0].readOnly='' ;" //修改只读属性为空，变成可以输入的元素

driver.execute_script(js3) #执行js

文件上传

案例：用户头像上传

adFileUpload=driver.find_element_by_name("file") //定位上传控件

filePath = "C:\\test\\uploadfile \\test.jpg"; //定义了一个本地文件的路径

adFileUpload .send_keys(filePath); //为上传控件进行赋值操作，将需要上传的文件的路径赋给控件

iframe处理

码同学

说明：在Selenium中封装了如何切换frame框架的方法

方法：

1). driver.switch_to.frame(frame_reference) --> 切换到指定frame的方法

frame_reference：可以为frame框架的name、id或者定位到的frame元素

2). driver.switch_to.default_content() --> 恢复默认页面方法

driver.switch_to.parent_frame()--->恢复到上级iframe页面方法

在frame中操作其他页面，必须先回到默认页面，才能进一步操作

选择处理

对下拉框进行操作时首先要定位到这个下拉框，new 一个Selcet对象，然后对它进行操作

```
from selenium.webdriver.support.select import Select
//找到下拉选择框的元素：
select = driver.find_element_by_id("arealD")
//选择对应的选择项：
Select(select).select_by_visible_text("天津市")//通过可见文本去选择
Select(select).select_by_value("shanxi")//通过html中的value值去选择
Select(select).select_by_index(1)//通过index（索引从0开始）选择
```

单选框

单选按钮就当按钮处理理解起来就简单了

r_sex = driver.find_element_by_id("sexID2") //找到单选框元素

r_sex.click() //选择某个单选项

r_sex.is_selected() //判断某个单选项是否已经被选择 //返回的是Boolean类型

复选框

//多选项的操作和单选的差不多：

checkBox = driver.find_element_by_id("u1") checkbox.click() //点击复选框

checkbox.clear() //清除复选框

checkbox .is_selected() //判断复选框是否被选中

checkbox .is_enabled() //判断复选框是否可用

选择处理

码同学

◆ 单选

https://www.w3school.com.cn/tiy/t.asp?f=eg_html_radiobuttons

◆ 复选

https://www.w3school.com.cn/tiy/t.asp?f=eg_html_checkboxes

◆ 下拉框

1. https://www.w3school.com.cn/tiy/t.asp?f=eg_html_dropdownbox

js alert处理

js弹框是页面上一种特殊元素，当需要操作js弹框时，需要按照如下代码进行

```
alert = driver.switch_to.alert #切换得到alert弹框对象
alert.accept() #确认/确定
alert.dismiss() #取消
alert.send_keys('xxx') #向弹框输入内容
alert.text #获取弹框上的提示语文字
```

Alert处理

码同学

- 只有确认的弹框

https://www.w3school.com.cn/tiy/t.asp?f=eg_js_alert

- 有确认和取消的弹框

https://www.w3school.com.cn/tiy/t.asp?f=eg_js_confirm

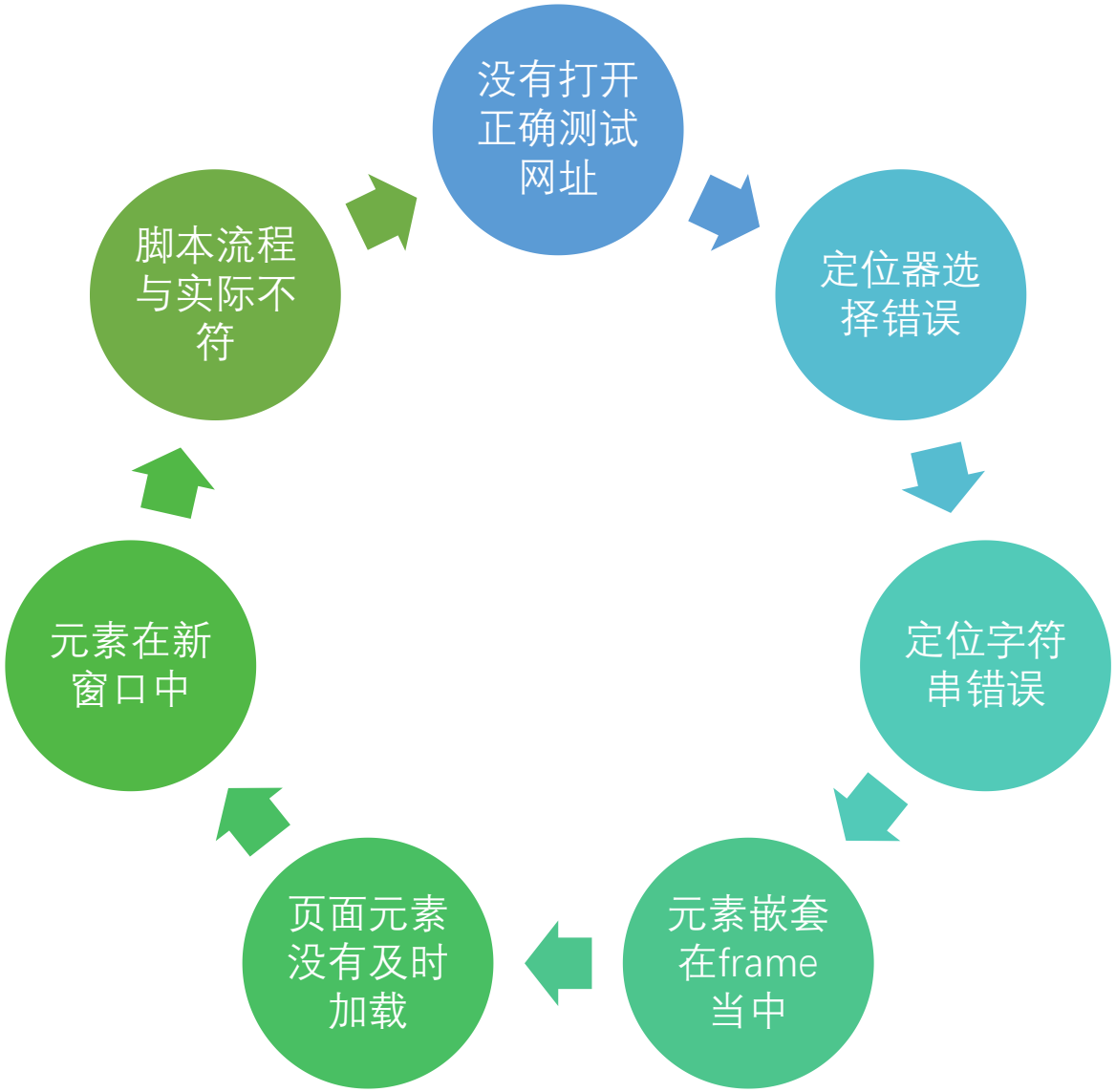
- 有输入框的弹框

https://www.w3school.com.cn/tiy/t.asp?f=eg_js_prompt

- 有提醒语的弹框

https://www.w3school.com.cn/tiy/t.asp?f=eg_js_alert_2

UI元素无法识别的常见原因



码同学

THANK YOU

授课老师：沙陌