# 切换窗口：

driver.switch\_to.window(handle)

如何获取handle?

通过driver.window\_handles 方法获取当前浏览器所有窗口的handle

通过循环切换窗口判断标题的方法来确定当前窗口是否为目标窗口

保留当前窗口handle:

driver.current\_window\_handle

关闭窗口的方法: driver.close() 用于关闭当前窗口

# 对话框处理：

**driver.switch\_to.alert**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 点击ok | 获取文本 | 点击cancel | 输入内容 |
| alert | accept() | text | N/A | N/A |
| confirm | accept() | text | dismiss() | N/A |
| prompt | accept() | text | dismiss() | send\_keys() |

注意浏览器alter和普通html弹出框的区别

# 上传文件：

# 直接发送键盘消息给 当前应用程序，

# 前提是浏览器必须是当前应用

# 此方法只在windows环境有用

# pip install pypiwin32

import win32com.client

#获取shell对象

shell = win32com.client.Dispatch("WScript.Shell")

# 有的系统要加 '\r'

# 有的系统要加 '\r\n'

#使用shell对象的Sendkeys方法给应用程序发送字符串

shell.Sendkeys(r"d:\button.png" + '\n')

输入法记得调成英文的，默认中文会导致某些字符输入不进去

**注意：这个方法是发送字符串到你当前的焦点程序，所以自动化时不要做其他操作**

# 滚动屏幕

如果窗口太长，改变窗口大小都不够，因为屏幕只有那么大，

需要滚动到元素的方案 ，可以使用最后一招， 直接让浏览器执行javascript脚本

浏览器执行的语言是js， 如果我们能直接传入js让浏览器执行，

就得到了最大的灵活性去控制浏览器。

可以算是终极武器。 比如滚动屏幕，就可以使用 js语言

window.scrollBy(250,0) 就是横向滚动250个像素

让浏览器去执行

selenium里面实用js代码

driver.execute\_script(**'window.scrollBy(200,0)'**)

# 二次渲染

案例：

登录 教管系统，<http://localhost/mgr/login/login.html>

实现一下添加2门课程，再删除2门课程都能正确显示

from selenium import webdriver  
import time  
  
  
driver **=** webdriver.**Chrome**()  
driver.**implicitly\_wait**(5)  
  
driver.**get**('http://localhost/mgr/login/login.html')  
  
driver.**find\_element\_by\_id**('username').**send\_keys**('auto')  
driver.**find\_element\_by\_id**('password').**send\_keys**('sdfsdfsdf')  
  
driver.**find\_element\_by\_tag\_name**('button').**click**()  
  
  
def addCourse(*driver*,*name*,*desc*,*idx*)**:** *driver*.**find\_element\_by\_css\_selector**("a[ui-sref='course']").**click**()  
 time.**sleep**(1)  
  
 *driver*.**find\_element\_by\_css\_selector**(  
 'button[ng-click^=showAddOne]').**click**()  
  
 ele **=** *driver*.**find\_element\_by\_css\_selector**(  
 "input[ng-model='addData.name']")  
 ele.**clear**()  
 ele.**send\_keys**(*name*)  
  
 ele **=** *driver*.**find\_element\_by\_css\_selector**(  
 "textarea[ng-model='addData.desc']")  
 ele.**clear**()  
 ele.**send\_keys**(*desc*)  
  
 ele **=** *driver*.**find\_element\_by\_css\_selector**("input[ng-model='addData.display\_idx']")  
 ele.**clear**()  
 ele.**send\_keys**(*idx*)  
  
 *driver*.**find\_element\_by\_css\_selector**('button[ng-click^=addOne]').**click**()  
  
  
  
  
def DeleteAllTeacher(*driver*)**:** delButtons **=** *driver*.**find\_elements\_by\_css\_selector**(  
'\*[ng-click^=delOne]')  
  
  
 for button in delButtons**:** button.**click**()  
 *driver*.**find\_element\_by\_css\_selector**(  
 '.modal-footer .btn-primary').**click**()  
  
  
  
  
**addCourse**(driver,'数学','数学',1)  
**addCourse**(driver,'语文','语文',2)  
  
time.**sleep**(1)  
**DeleteAllTeacher**(driver)  
  
input()  
driver.**quit**()

结果运行发现

selenium.common.exceptions.StaleElementReferenceException: Message: stale element reference: element is not attached to the page document

原来是现在的web应用和以前大不相同，

以前的web应用大都是后端直接产生最终的html界面

而现在呢？很多是页面加载后（也加载javascript前端代码）， 由前端代码动态改变界面的内容，比如从后端获取一些数据后，再动态改变界面的内容。

尤其是很多采用了框架像 angular， react， vue这些都会这样做。

这样就有一个问题，开始的时候获取的html只是一个模板一样的文档，里面的数据没有产生，

而是需要等前端js代码动态的 再次向后端查询数据后， 才动态的向html模板中填入数据或者修改数据。

-----------------------

原因是：

一下子获取了多个删除按钮 元素， 然后用循环点击 每个删除按钮

但是当点击第一个按钮之后，页面有些部分被更新了， 剩余的几个按钮，

虽然看起来还是原来的样子。但是已经是新的界面元素了

所以，你通过原来老的 界面元素点击 ，就会出现这个错误

那么怎么解决这个问题呢？

既然点击 第一个课程的删除按钮 后，页面部分被更新了， 就只好重新获取界面元素。

**def** DeleteAllCourse(*driver*):  
  
 *driver*.implicitly\_wait(**2**)  
  
 **while 1**:  
 delButtons = *driver*.find\_elements\_by\_css\_selector(  
 **'\*[ng-click^=delOne]'**)  
 **if** len(delButtons) < **1**:  
 **break** delButtons[**0**].click()  
 *driver*.find\_element\_by\_css\_selector(  
 **'.modal-footer .btn-primary'**).click()  
 time.sleep(**1**)

# 半自动化

# 半自动化的用途

selenium 属于图形界面自动化的测试，图形界面自动化的难度是比较大的。

模拟难：

有些特殊的输入，比如12306 点击图片的某个部分

打开 https://kyfw.12306.cn/otn/login/init

检查难：

打开 https://www.vmall.com/

比如检查菜单有没有，

我们经常是检查某些输出的字符，

应该还有：

界面的整体是否布局有错乱，

图片显示logo是否正确

为什么会有这些难度， 因为界面操作者是人 ， 有很多操作需要人的智能，才方便输入，检查输出。

比如 难自动化的 操作 包括 模拟输入，检查输入 。 都提示（beep）让人去做

#打开浏览器代码  
  
#系统提示，手动输入验证吗  
**import** time  
**import** winsound  
winsound.Beep(**1500**,**30000**)  
  
time.sleep(**10**)  
  
  
#自动化代码

# 冻结窗口

setTimeout(function(){debugger},3000)

# 上节课作业代码

**from** selenium **import** webdriver  
**from** selenium.webdriver.support.select **import** Select  
  
driver=webdriver.Chrome()  
driver.implicitly\_wait(10)  
  
driver.get(**'https://kyfw.12306.cn/otn/leftTicket/init'**)  
  
*#输入起点和终点*ele=driver.find\_element\_by\_id(**'fromStationText'**)  
ele.click()  
ele.send\_keys(**'南京南\n'**)  
  
ele1=driver.find\_element\_by\_id(**'toStationText'**)  
ele1.click()  
ele1.send\_keys(**'杭州东\n'**)  
  
*#选择发车时段*select=Select(driver.find\_element\_by\_id(**'cc\_start\_time'**))  
select.select\_by\_visible\_text(**'06:00--12:00'**)  
  
*#选择发车时间-当前日期的第二天*driver.find\_element\_by\_xpath(**'//\*[@id="date\_range"]//li[2]'**).click()  
  
xpath=**'//\*[@id="queryLeftTable"]//td[4][@class]/preceding-sibling::td[last()]//a'**xpath2=**'//\*[@id="queryLeftTable"]//td[4][@class]/../td[1]//a'**trains=driver.find\_elements\_by\_xpath(xpath2)  
  
print(**'二等座有座的车次有：'**)  
**for** t **in** trains:  
 print(t.text)  
  
driver.quit()