* 1.**获取当前页面的Url**

方法：current\_url

实例：driver.current\_url

* 2.**获取元素坐标**

方法：location

解释：首先查找到你要获取元素的，然后调用location方法

实例：driver.find\_element\_by\_xpath("xpath").location

* 3.**表单的提交**

方法：submit

解释:查找到表单（from）直接调用submit即可

实例：driver.find\_element\_by\_id("form1").submit()

* 4.**获取CSS的属性值**

方法：value\_of\_css\_property(css\_name)

实例：driver.find\_element\_by\_css\_selector("input.btn").value\_of\_css\_property("input.btn")

* 5.**获取元素的属性值**

方法：get\_attribute(element\_name)

实例：driver.find\_element\_by\_id("kw").get\_attribute("kw")

* 6.**判断元素是否被选中**

方法：is\_selected()

实例：driver.find\_element\_by\_id("form1").is\_selected()

* 7.**返回元素的大小**

方法：size

实例：driver.find\_element\_by\_id("iptPassword").size

返回值：{'width': 250, 'height': 30}

* 8.**判断元素是否显示**

方法：is\_displayed()

实例：driver.find\_element\_by\_id("iptPassword").is\_displayed()

* 9.**判断元素是否被使用**

方法：is\_enabled()

实例：driver.find\_element\_by\_id("iptPassword").is\_enabled()

* 10.**获取元素的文本值**

方法：text

实例：driver.find\_element\_by\_id("iptUsername").text

* 11.**元素赋值**

方法：send\_keys(\*values)

实例：driver.find\_element\_by\_id("iptUsername").send\_keys('admin')

* 12.**返回元素的tagName**

方法：tag\_name

实例：driver.find\_element\_by\_id("iptUsername").tag\_name

* 13.**删除浏览器所有的cookies**

方法：delete\_all\_cookies()

实例：driver.delete\_all\_cookies()

* 14.**删除指定的cookie**

方法：delete\_cookie(name)

实例：deriver.delete\_cookie("my\_cookie\_name")

* 15.**关闭浏览器**

方法：close()

实例：driver.close()

* 16.**关闭浏览器并且退出驱动程序**

方法：quit()

实例：driver.quit()

* 17.**返回上一页**

方法：back()

实例：driver.back()

* 18.**清空输入框**

方法：clear()

实例：driver.clear()

* 19.**浏览器窗口最大化**

方法：maximize\_window()

实例：driver.maximize\_window()

* 20.**查看浏览器的名字**

方法：name

实例：drvier.name

* 21.**返回当前会话中的cookies**

方法：get\_cookies()

实例：driver.get\_cookies()

* 22.**根据cookie name 查找映射Value值**

方法：driver.get\_cookie(cookie\_name)

实例：driver.get\_cookie("NET\_SessionId")

* 23.**截取当前页面**

方法：get\_screenshot\_as\_file(filename)

实例：driver.get\_screenshot\_as\_file("D:\\Program Files\\Python27\\NM.bmp")

* 24.**获取当前窗口的坐标**

方法：get\_window\_position()

实例：driver.get\_window\_position()

* 25.**获取当前窗口的长和宽**

方法：get\_window\_size()

实例：driver.get\_window\_size()

**ActionChains类鼠标操作的常用方法：**

引入ActionChains类：from selenium.webdriver.common.action\_chains import ActionChains

* 26.**右击**

方法：context\_click()

实例：ActionChains(driver).context\_click(driver.find\_element\_by\_id("id")).perform()

* 27.**双击**

方法：double\_click()

实例：ActionChains(driver).double\_click(driver.find\_element\_by\_name("name")).perform()

* 28：**鼠标拖放**

方法：drag\_and\_drop(source, target)

source：鼠标按下的源元素；target：鼠标释放的目标元素

实例：element = driver.find\_element\_by\_name("name")

target = driver.find\_element\_by\_name("name")

ActionChains(driver).drag\_and\_drop(element, target).perform()

* 29：**鼠标悬停在一个元素上（hover）**

方法：move\_to\_element()

实例：above = driver.find\_element\_by\_xpath("xpath路径")

ActionChains(driver).move\_to\_element(above).perform()

* 30：**按下鼠标左键在一个元素上**

方法：click\_and\_hold()

实例：left = driver.find\_element\_by\_name("name")

ActionChains(driver).click\_and\_hold(left).perform()

**键盘事件：**

引入Keys类包：from selenium.webdriver.common.keys import Keys

* 31：**输入**

方法：send\_keys()

实例：driver.find\_element\_by\_id("id").send\_keys("XXX")

* 32：**输入空格**

方法：send\_keys(Keys.SPACE)

实例：driver.find\_element\_by\_id("id").send\_keys(Keys.SPACE)

* 33：ctrl + a   
  全选输入框的内容   
  ctrl + x   
  剪切输入框的内容   
  ctrl + v   
  粘贴到输入框   
  ctrl + c   
  复制

方法：send\_keys(Keys.CONTROL,'a')

实例：driver.find\_element\_by\_id("id").send\_keys(Keys.CONTROL,'a')

* 34：**回车代替点击**

方法：send\_keys(Keys.ENTER)

实例：driver.find\_element\_by\_id("id").send\_keys(Keys.ENTER)

* 35：**制表键(Tab)**

方法：send\_keys(Keys.TAB)

实例：driver.find\_element\_by\_id("id").send\_keys(Keys.TAB)

* 36：**回退键（Esc）**

方法：send\_keys(Keys.ESCAPE)

实例：driver.find\_element\_by\_id("id").send\_keys(Keys.ESCAPE)

**等待时间**

导入 WebDriverWait 包   
from selenium.webdriver.support.ui import WebDriverWait   
导入 time 包   
import time

* 37：**固定等待时间**

方法：sleep()

实例：time.sleep(5) # 等待5秒

* 38：**等待一个元素被发现，或一个命令完成，超出了设置时间则抛出异常智能等待。**

方法：implicitly\_wait()

实例：driver.implicitly\_wait(30)

* 39：**在设置时间内，默认每隔一段时间检测一次当前页面元素是否存在，如果超过设置时间检测不到则抛出异常**

"方法：WebDriverWait()"

#WebDriverWait(driver, timeout, poll\_frequency=0.5, ignored\_exceptions=None)

#——driver：WebDriver的驱动程序(Ie, Firefox, Chrome或远程)

#——timeout：最长超时时间，默认以秒为单位

#——poll\_frequency：休眠时间的间隔（步长）时间，默认为0.5秒

#——ignored\_exceptions：超时后的异常信息，默认情况下抛NoSuchElementException异常

实例：

element = WebDriverWait(driver, 10).until(lambda x: x.find\_element\_by\_id("id"))

#一般由unit()或until\_not()方法配合使用，同上:

**此处请详细参考 “Python-Selenium-三种等待.docx”**

调用该方法提供的驱动程序作为一个参数，直到返回值不为False。

——until(method, message=’’)

调用该方法提供的驱动程序作为一个参数，直到返回值为False。

——until\_not(method, message=’’)

# 还可以与expected\_conditions 一起使用

from selenium.webdriver.support import expected\_conditions as EC

# 实例： 判断某个元素是否可见并且是enable的，这样才clickable

WebDriverWait(dr,15,1).until(EC.element\_to\_be\_clickable((By.ID,"EmployeeListMenu")),"Not Find element")

* 40:**选择当前页面上所有tag**

name为input的元素

inputs = driver.find\_elements\_by\_tag\_name(‘input‘)

* 41：**从中过滤出type为checkbox的元素，并勾选上**

for input in inputs:

if input.get\_attribute(‘type‘) == ‘checkbox‘:

input.click()

* 42：**使用CSS定位选择所有type为checkbox的元素，并勾选上**

checkboxes = driver.find\_elements\_by\_css\_selector(‘input[type = checkbox]‘)

for checkbox in checkboxes:

　　checkbox.click()

* 43：**把最后一个checkbox的勾去掉，pop()方法空参数时，默认移除list中的最后一个元素。**

driver.find\_elements\_by\_css\_selector(‘input[type = checkbox]‘).pop().click()

**切换活动对象**

* 44：**切换浏览器handle**

# 切换不同的tab页

方法：driver.switch\_to.window(window\_name)

# 备注：从A页跳转到B页，句柄已经切换过去，但是焦点没有切过去，所以需要switch\_to.window，把焦点也切过去，才可以在当前页进行操作。

# 切换是思路，获取所有的句柄，因为返回是一个list，而且要切换的对象都是最后一个，可以使用[-1]直接切过去

# 例如：

driver.switch\_to.window(driver.window\_handles[-1])

* 45：**返回上一级表单**

方法： driver.switch\_to\_parent\_content()#旧方法

driver.switch\_to.parent\_content#新方法

* 46：**返回最外层表单**

方法： driver.switch\_to\_default\_content()#旧方法

driver.switch\_to.default\_content()#新方法

* 47：**切换到指定frame中**

方法：driver.switch\_to.frame('xxx')

实例：driver.switch\_to.frame('frame\_name')

driver.switch\_to.frame(index)

driver.switch\_to.frame(driver.find\_elements\_by\_tag\_name("iframe")[0])

* 48：**获取当前page的title**

方法：driver.title

实例：driver.title

* 49：**焦点切换到弹窗。**

方法：driver.switch\_to\_alert()

实例：driver.switch\_to\_alert()

* 50：**前进**

方法：

driver.forward()

* 51：**刷新页面**

方法：driver.refresh()

上传文件

upload.py

#coding=utf-8

from selenium import webdriver

import os,time

driver = webdriver.Firefox()

#脚本要与upload\_file.html同一目录

file\_path = 'file:///' + os.path.abspath('upload\_file.html')

driver.get(file\_path)

#定位上传按钮，添加本地文件

driver.find\_element\_by\_name("file").send\_keys('D:\\selenium\_use\_case\upload\_file.txt')

time.sleep(2)

driver.quit()

其它有些应用不好找，所以就自己创建页面，这样虽然麻烦，但脚本代码突出重点。

这里找一139邮箱的实例，有帐号的同学可以测试一下~！

（登陆基础版的139邮箱，网盘模块上传文件。）

139upload.py

#coding=utf-8

from selenium import webdriver

import os,time

driver = webdriver.Firefox()

driver.get("http://m.mail.10086.cn")

driver.implicitly\_wait(30)

#登陆

driver.find\_element\_by\_id("ur").send\_keys("手机号")

driver.find\_element\_by\_id("pw").send\_keys("密码")

driver.find\_element\_by\_class\_name("loading\_btn").click()

time.sleep(3)

#进入139网盘模块

driver.find\_element\_by\_xpath("/html/body/div[3]/a[9]/span[2]").click()

time.sleep(3)

#上传文件

driver.find\_element\_by\_id("id\_file").send\_keys('D:\\selenium\_use\_case\upload\_file.txt')

time.sleep(5)

driver.quit()