

# 目录

前言	1.1
初始化	1.2
基本操作	1.3
查找定位元素	1.3.1
输入文字	1.3.2
点击元素	1.3.3
触发搜索	1.3.3.1
文档和教程	1.4
心得和总结	1.5
附录	1.6
参考资料	1.6.1

# Selenium知识总结

- 最新版本: v1.0
- 更新时间: 20210417

## 简介

总结折腾Selenium方面的心得和经验，供参考。

## 源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

### Gitbook源码

- [crifan/selenium\\_summary: Selenium知识总结](#)

如何使用此Gitbook源码去生成发布为电子书

详见：[crifan/gitbook\\_template: demo how to use crifan gitbook template and demo](#)

### 在线浏览

- [Selenium知识总结 book.crifan.com](#)
- [Selenium知识总结 crifan.github.io](#)

### 离线下载阅读

- [Selenium知识总结 PDF](#)
- [Selenium知识总结 ePUB](#)
- [Selenium知识总结 Mobi](#)

### 版权说明

此电子书教程的全部内容，如无特别说明，均为本人原创和整理。其中部分内容参考自网络，均已备注了出处。如有发现侵犯您版权，请通过邮箱联系我 `admin 艾特 crifan.com`，我会尽快删除。谢谢合作。

### 鸣谢

感谢我的老婆陈雪的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

## 更多其他电子书

本人 crifan 还写了其他 100+ 本电子书教程，感兴趣可移步至：

[crifan/crifan\\_ebook\\_readme: Crifan的电子书的使用说明](#)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2021-04-29 07:51:52

# 初始化

核心逻辑：

- Python中安装Selenium库
  - pip install selenium
- 再安装webdriver
  - 比如： Chrome 的 driver : chromedriver
    - 需要下载到二进制的chromedriver，并确保PATH中能找到

## 安装 selenium

```
pip3 install selenium
```

附上完整log

```
pip3 install selenium
Looking in indexes: http://mirrors.aliyun.com/pypi/simple/
Collecting selenium
  Downloading http://mirrors.aliyun.com/pypi/packages/80/d6/4294f0b4bce4de0abf13e17190289f
9d0613b0a44e5dd6a7f5ca98459853/selenium-3.141.0-py2.py3-none-any.whl (904 kB)
    ██████████████████████████████████████████████████████████████████████████████████████████ 904 kB 1.2 MB/s
Requirement already satisfied: urllib3 in /Users/crifan/.pyenv/versions/3.6.6/lib/python3.
6/site-packages (from selenium) (1.25.8)
Installing collected packages: selenium
Successfully installed selenium-3.141.0
```

## 安装driver

Selenium 的运行依赖于具体的浏览器（的内核），此处叫做： driver

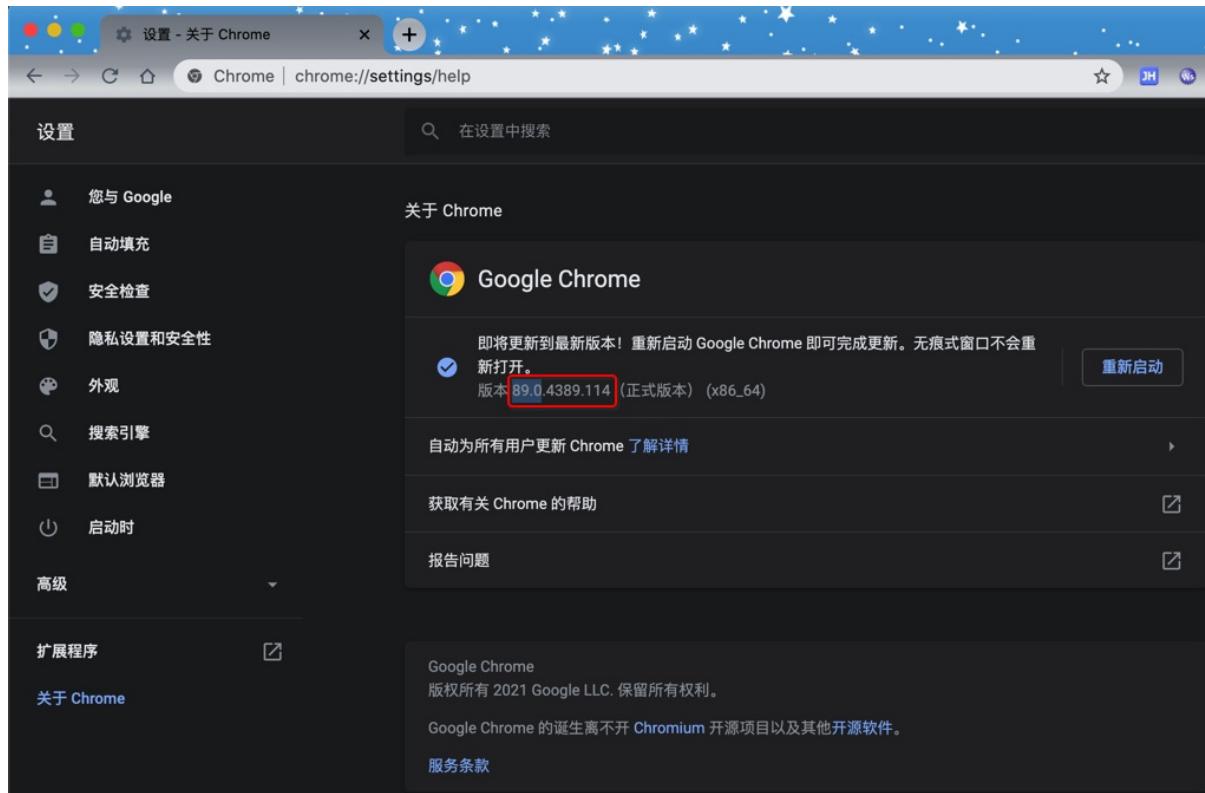
- Chrome 的 driver : ChromeDriver = chromedriver

## 安装Chrome的driver: ChromeDriver

下载 ChromeDriver

要下载和你的Chrome版本一致的ChromeDriver

此处查看到Chrome的版本是： 89.0



所以要下载的 ChromeDriver 也是要于此版本一致的，即下载 ChromeDriver 89.0 的版本

- 下载源1：Chrome官网
  - [ChromeDriver - WebDriver for Chrome](#)
    - 此时最新版是：[ChromeDriver 89.0.4389.23](#)

The screenshot shows the official ChromeDriver website at [chromedriver.chromium.org](https://chromedriver.chromium.org/). The main content area includes:

# ChromeDriver - WebDriver for Chrome

**CHROMEDRIVER**

- CAPABILITIES & CHROMEOPTIONS
- CHROME EXTENSIONS
- CHROMEDRIVER CANARY
- CONTRIBUTING
- DOWNLOADS** (highlighted)
  - VERSION SELECTION
- GETTING STARTED
- ANDROID
- CHROMEOS
- LOGGING
- PERFORMANCE LOG
- MOBILE EMULATION
- NEED HELP?
- CHROME DOESN'T START OR CRASHES IMMEDIATELY
- CHROMEDRIVER CRASHES
- CLICKING ISSUES
- KEYBOARD SUPPORT
- OPERATION NOT SUPPORTED WHEN USING REMOTE DEBUGGING
- SECURITY CONSIDERATIONS

**ChromeDriver**

WebDriver is an open source tool for automated testing of webapps across many browsers. It provides capabilities for navigating to web pages, user input, JavaScript execution, and more. ChromeDriver is a standalone server that implements the [W3C WebDriver standard](#). ChromeDriver is available for Chrome on Android and Chrome on Desktop (Mac, Linux, Windows and ChromeOS).

You can view the current implementation status of the WebDriver standard [here](#).

**All versions available in Downloads**

- Latest stable release: [ChromeDriver 89.0.4389.23](#)
- Latest beta release: [ChromeDriver 90.0.4430.24](#)

**ChromeDriver Documentation**

- Getting started with ChromeDriver on Desktop (Windows, Mac, Linux)
  - ChromeDriver with Android
  - ChromeDriver with ChromeOS
- ChromeOptions, the capabilities of ChromeDriver
- Mobile emulation
- Security Considerations, with recommendations on keeping ChromeDriver safe
- Chrome Extension installation

- [Index of /89.0.4389.23/](#)
- [chromedriver\\_mac64.zip](#)

Name	Last modified	Size	ETag
Parent Directory		-	-
<a href="#">chromedriver_linux64.zip</a>	2021-01-28 17:30:52	5.57MB	24686a3cc3ccf8cbc60cf744baa47692
<a href="#">chromedriver_mac64.zip</a>	2021-01-28 17:30:53	7.97MB	a6620c6a6804fa08365dfc6e8c8724e6
<a href="#">chromedriver_mac64_m1.zip</a>	2021-01-28 17:30:55	7.17MB	1544d2a1b1a6fbdd55ff5dac0b48d89f
<a href="#">chromedriver_win32.zip</a>	2021-01-28 17:30:57	5.68MB	0bf4bc39f34cee67f5f95af8a24c191
<a href="#">notes.txt</a>	2021-01-28 17:31:00	0.00MB	4e3ff6354a7462edfe5d95ab8690f7c8

- 下载源2：淘宝的npm源
  - <http://npm.taobao.org/mirrors/chromedriver/>
  - ->
  - <http://npm.taobao.org/mirrors/chromedriver/89.0.4389.23/>
  - ->
  - [http://npm.taobao.org/mirrors/chromedriver/89.0.4389.23/chromedriver\\_mac64.zip](http://npm.taobao.org/mirrors/chromedriver/89.0.4389.23/chromedriver_mac64.zip)

## 确保命令行中能调用到chromedriver

想要让命令行中，可以调用到 `chromedriver`，即：把 `chromedriver` 放到环境变量 `PATH` 中：

下载后解压得到二进制的：`chromedriver`

把 `chromedriver` 放到 `PATH` 中

- 方式1：移动到系统相关目录

```
sudo mv /xxx/chromedriver /usr/local/bin
```

- 方式2：放到某个路径下，把该路径加到PATH中

- 此处放到了：`/Users/crifan/dev/dev_tool/selenium/chromedriver`
- 把路径加到PATH中
  - 编辑启动脚本：

```
vi ~/.zshrc
```

- 在文件最后加上：`PATH=$PATH:/Users/crifan/dev/dev_tool/selenium`
- 使其立刻生效：

```
source ~/.zshrc
```

然后去确认命令行中能找到：

```
which chromedriver
```

确保能输出对应了路径，表示找到了。

顺带也可以去：看看版本：

```
chromedriver --version
```

此处输出是： ChromeDriver 89.0.4389.23

## 写测试代码，确认环境正常

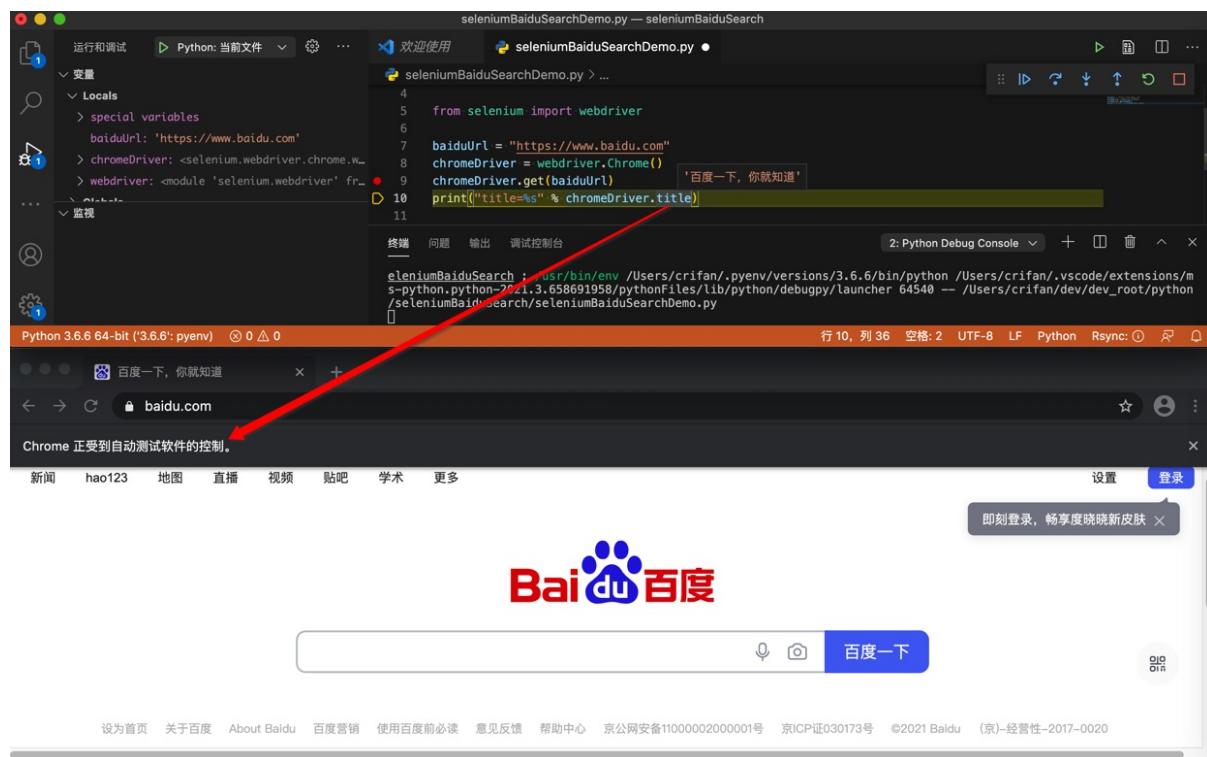
可以用代码：

```
from selenium import webdriver

baiduUrl = "https://www.baidu.com"
chromeDriver = webdriver.Chrome()
chromeDriver.get(baiduUrl)
print("title=%s" % chromeDriver.title)
```

确认Selenium是否正常工作：可以启动Chrome浏览器，打开百度首页。

正常的效果：



其中可以注意到： Selenium 操作的 Chrome 会有提示： Chrome正受到自动测试软件的控制

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新： 2021-04-17 10:28:34



# 基本操作

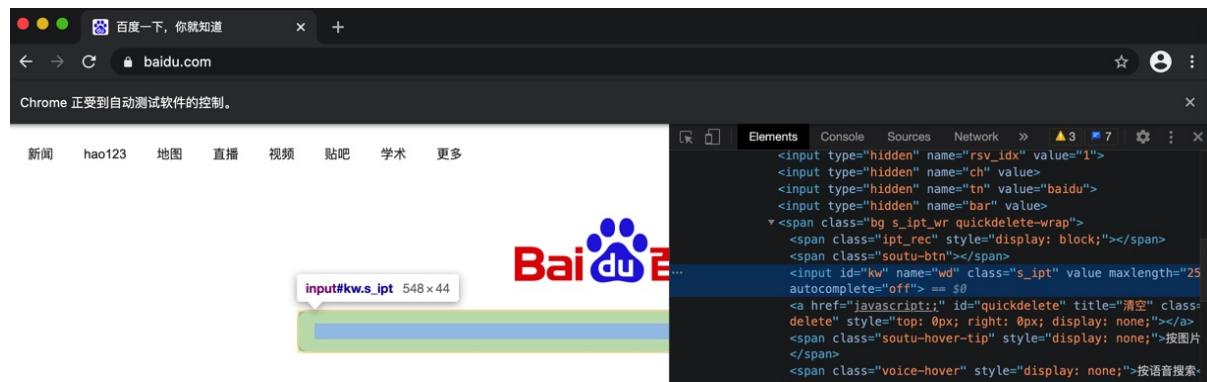
crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2021-04-17 12:14:04

# 查找定位元素

查找元素 = 定位元素

举例：

对于页面：



的html是：

```
<input id="kw" name="wd" class="s_ipt" value="" maxlength="255" autocomplete="off">
```

对应查找该元素的典型方式是：

- `find_element_by_id(id_)`

- 代码

```
driver.find_element_by_id("kw")
```

- 输出

```
<selenium.webdriver.remote.webelement.WebElement (session="5fb657d67aa9c8060f2d0b407b4d40df", element "a879125d-ebb4-4e0f-9659-9c3b07087bac")>
```

- 文档

- 4.1. Locating by Id

- `find_element_by_id(id_)`

- 注意：确保此处的id是唯一的

- `find_element(by='id', value=None)`

- 代码

```
driver.find_element(by="kw")
```

- 文档

- `find_element(by='id', value=None)`

## 查找元素的更详细介绍

去Selenium中定位和查找元素：

方法有很多，常见的有：

- `find_element_by_id`
- `find_element_by_name`
- `find_element_by_xpath`
- `find_element_by_link_text`
- `find_element_by_partial_link_text`
- `find_element_by_tag_name`
- `find_element_by_class_name`
- `find_element_by_css_selector`

如果页面中有多个该元素，则可以用：

- `find_elements_by_name`
- `find_elements_by_xpath`
- `find_elements_by_link_text`
- `find_elements_by_partial_link_text`
- `find_elements_by_tag_name`
- `find_elements_by_class_name`
- `find_elements_by_css_selector`
- 官网文档
  - 中文
    - [4. 查找元素 — Selenium-Python中文文档 2 documentation](#)
  - 英文
    - [4. Locating Elements — Selenium Python Bindings 2 documentation](#)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2021-04-17 12:13:20

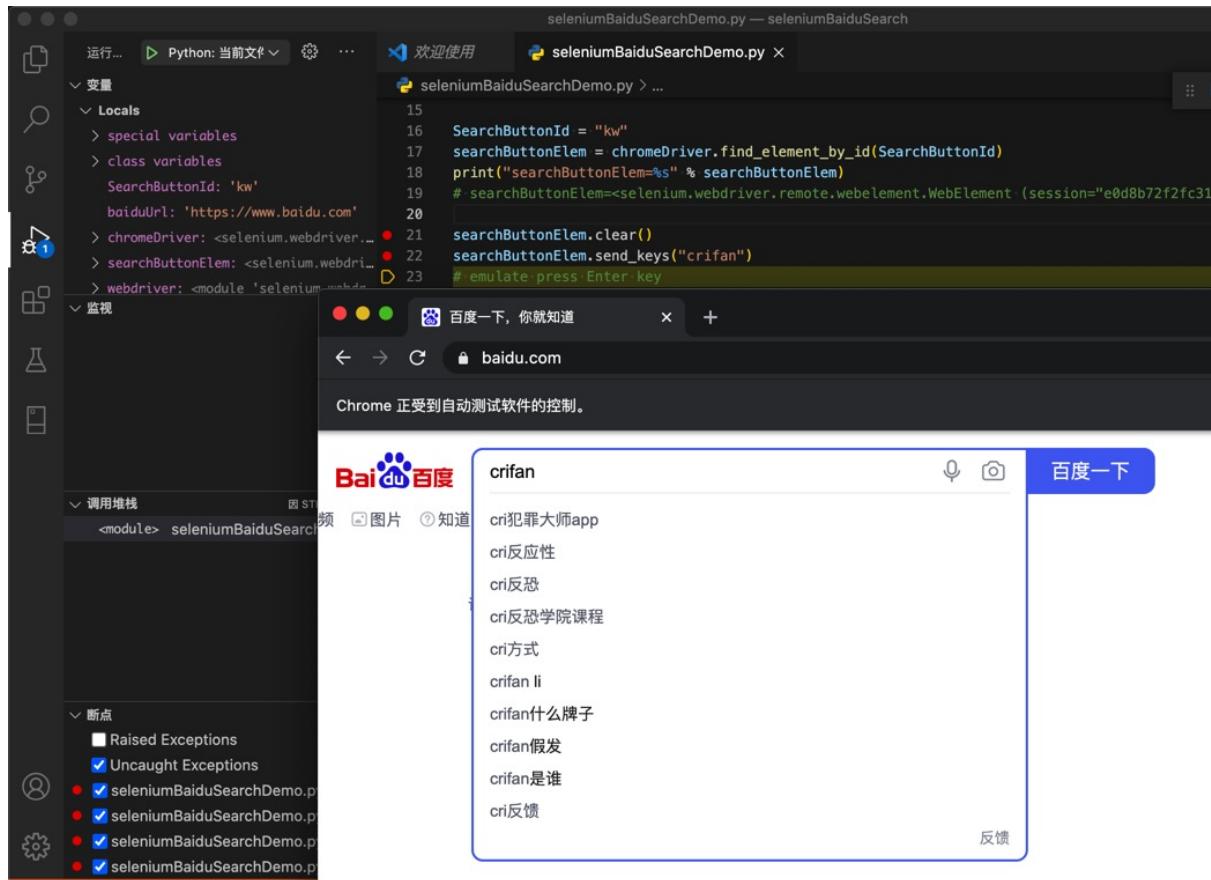
## (给元素) 输入文字

用 `send_keys`

举例：

```
searchButtonElem.send_keys("crifan")
```

效果：



## 相关：清除文字

在输入之前，往往可以或需要 清楚（已输入的）文字：

```
searchButtonElem.clear()
```

输入文字

---

# 点击元素

对于元素直接用 `click()` 即可。

举例：

```
baiduSearchButtonElem.click()
```

即可实现点击百度的 搜索一下 按钮

点击后的 效果：显示搜索结果

The screenshot shows the PyCharm IDE interface. On the left is the 'Variables' tool window. In the center, a code editor window displays a Python script named `seleniumBaiduSearchDemo.py`. The script contains the following code:

```

    # Method 2: find button and click
    BaiduSearchId = "su"
    baiduSearchButtonElem = chromeDriver.find_element_by_id(BaiduSearchId)
    print("baiduSearchButtonElem=%s" % baiduSearchButtonElem)
    baiduSearchButtonElem.click()
    print("Clicked button %s" % baiduSearchButtonElem)

```

The line `baiduSearchButtonElem.click()` is highlighted with a yellow background. Below the code editor is a browser window showing the Baidu search results for the query "crifan". The search bar contains "crifan" and the results page shows approximately 2,220,000 results.

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2021-04-17 12:18:30

## 触发搜索

对于想要触发百度首页中的搜索来说，除了点击元素外，还可以模拟输入回车键。

- 触发百度搜索

- 方式1：点击 百度一下 按钮

```
BaiduSearchId = "su"  
baiduSearchButtonElem = chromeDriver.find_element_by_id(BaiduSearchId)  
baiduSearchButtonElem.click()
```

- 方式2：模拟输入 回车键

```
from selenium.webdriver.common.keys import Keys  
# Method 1: emulate press Enter key  
searchButtonElem.send_keys(Keys.RETURN)
```

附上：模拟百度输入并搜索的相关代码

```
searchStr = "crifan"  
searchButtonElem.send_keys(searchStr)  
print("Entered %s to search box" % searchStr)  
  
# click button  
# Method 1: emulate press Enter key  
# searchButtonElem.send_keys(Keys.RETURN)  
# print("Pressed Enter/Return key")  
  
# Method 2: find button and click  
BaiduSearchId = "su"  
baiduSearchButtonElem = chromeDriver.find_element_by_id(BaiduSearchId)  
print("baiduSearchButtonElem=%s" % baiduSearchButtonElem)  
baiduSearchButtonElem.click()  
print("Clicked button %s" % baiduSearchButtonElem)
```

## Selenium文档和教程

### Selenium文档

#### 官网文档

- [4. Locating Elements — Selenium Python Bindings 2 documentation](#)
- [5. Waits — Selenium Python Bindings 2 documentation](#)
- [6. Page Objects — Selenium Python Bindings 2 documentation](#)
- [7. WebDriver API — Selenium Python Bindings 2 documentation](#)

中文版：

- [7. WebDriver API — Selenium-Python中文文档 2 documentation](#)

### Selenium教程

好的教程：

- [4. 查找元素 — Selenium-Python中文文档 2 documentation](#)

英文：

- [4. Locating Elements — Selenium Python Bindings 2 documentation](#)

API：

- [7. WebDriver API — Selenium Python Bindings 2 documentation](#)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2018-06-12 09:45:39

# Selenium心得和总结

## WebElement有很多函数和属性

整理如下，供有个概念：

- 函数

- `clear()`
- `click()`
- `find_element(by='id', value=None)`
- `find_element_by_class_name(name)`
- `find_element_by_css_selector(css_selector)`
- `find_element_by_id(id_)`
- `find_element_by_link_text(link_text)`
- `find_element_by_name(name)`
- `find_element_by_partial_link_text(link_text)`
- `find_element_by_tag_name(name)`
- `find_element_by_xpath(xpath)`
- `find_elements(by='id', value=None)`
- `find_elements_by_class_name(name)`
- `find_elements_by_css_selector(css_selector)`
- `find_elements_by_id(id_)`
- `find_elements_by_link_text(link_text)`
- `find_elements_by_name(name)`
- `find_elements_by_partial_link_text(link_text)`
- `find_elements_by_tag_name(name)`
- `find_elements_by_xpath(xpath)`
- `get_attribute(name)`
- `get_property(name)`
- `is_displayed()`
- `is_enabled()`
- `is_selected()`
- `screenshot(filename)`
- `send_keys(*value)`
- `submit()`
- `value_of_css_property(property_name)`

- 属性

- `id`
- `location`
- `parent`
- `rect`

- screenshot\_as\_png
- size
- text
- tag\_name

更多内容详见官网文档: [WebElement](#)

其中:

- find\_element\_by\_link\_text
- find\_element\_by\_partial\_link\_text

指的是标签 a 的 link , 而其他标签是用不了的。

## 不是 select 和 option 的下拉选项列表

select 只能用于

```
<select>
    <option>
```

才可以。其他的元素，比如我遇到的：

【已解决】Selenium如何点击下拉框并选择某个值

中的

```
<ul>
    <li role="option">
```

是用不了的。

所以最后就是用普通的，去 ul 下找到 li 的列表，通过index获得对应的元素，然后再去操作。

相关代码如下：

```
cartNumOptionElemList = driver.find_elements_by_xpath('//ul[@class="dropdown-menu"]/li[@role="option"]')
cartNumOptionCount = len(cartNumOptionElemList)
logging.info("cartNumOptionElemList=%s, cartNumOptionCount=%s", cartNumOptionElemList, cartNumOptionCount)
if cartNumOptionCount < gCfg["msStore"]["onceBuyNum"]:
    logging.error("Current Cart select max number %s < expected select number %s", cartNumOptionCount, gCfg["msStore"]["onceBuyNum"])
    driver.quit()
toSelectIdx = gCfg["msStore"]["onceBuyNum"] - 1
# carNumSelect = Select(cartNumOptionElemList)
# carNumSelect.select_by_index(gCfg["msStore"]["onceBuyNum"])
carNumSelectElem = cartNumOptionElemList[toSelectIdx]
logging.info("carNumSelectElem=%s", carNumSelectElem)
```

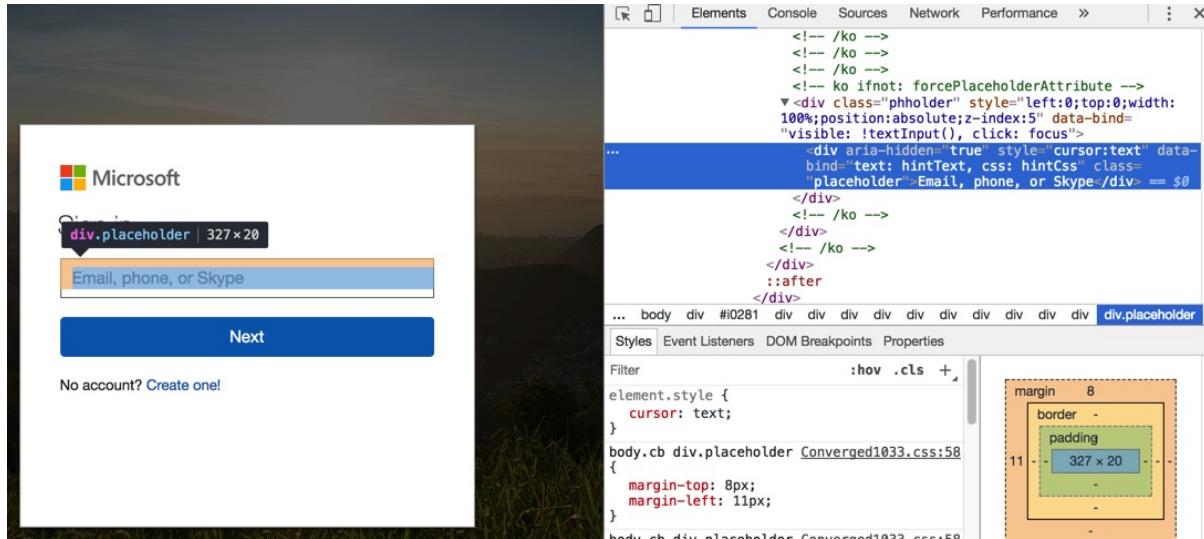
```

carNumSelectElem.click()
# aLinkElem = carNumSelectElem.find_element_by_link_text(str(gCfg["msStore"]["onceBuyNum"]))
())
# logging.info("aLinkElem=%s", aLinkElem)
# aLinkElem.click()

```

## 有时候Chrome中直接右键找到的元素，并不一定是你想要的

比如：

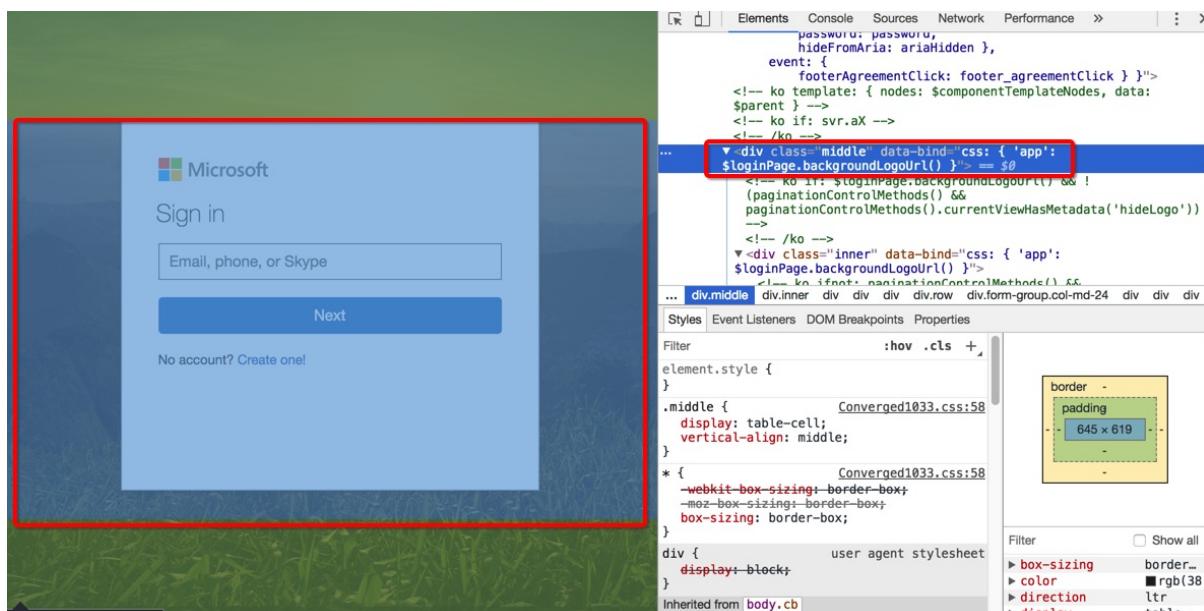


是个：

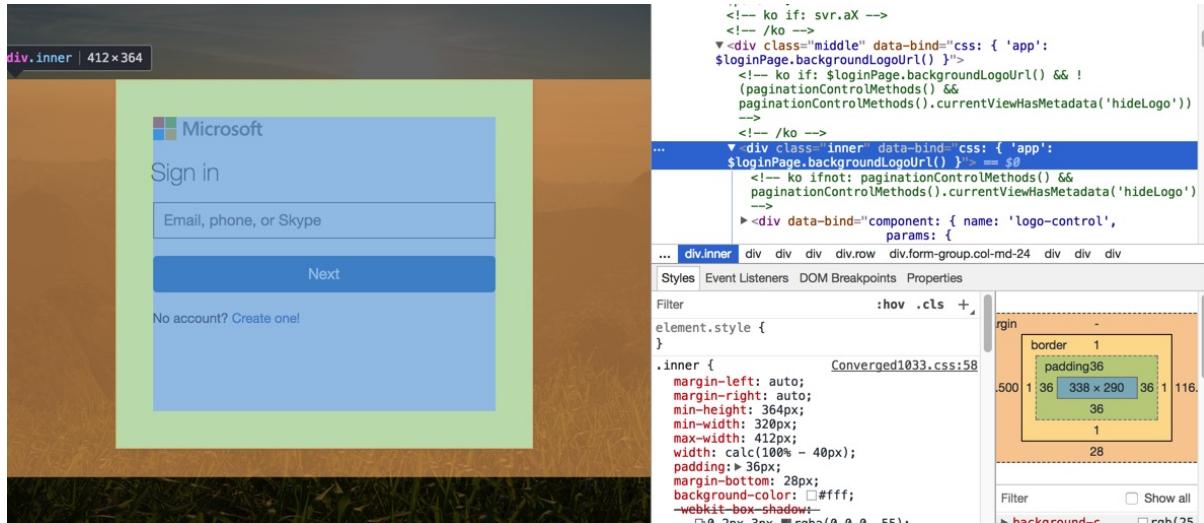
```
<div class="placeholder">
```

但是其实此处要找的是可以允许输入的input输入框

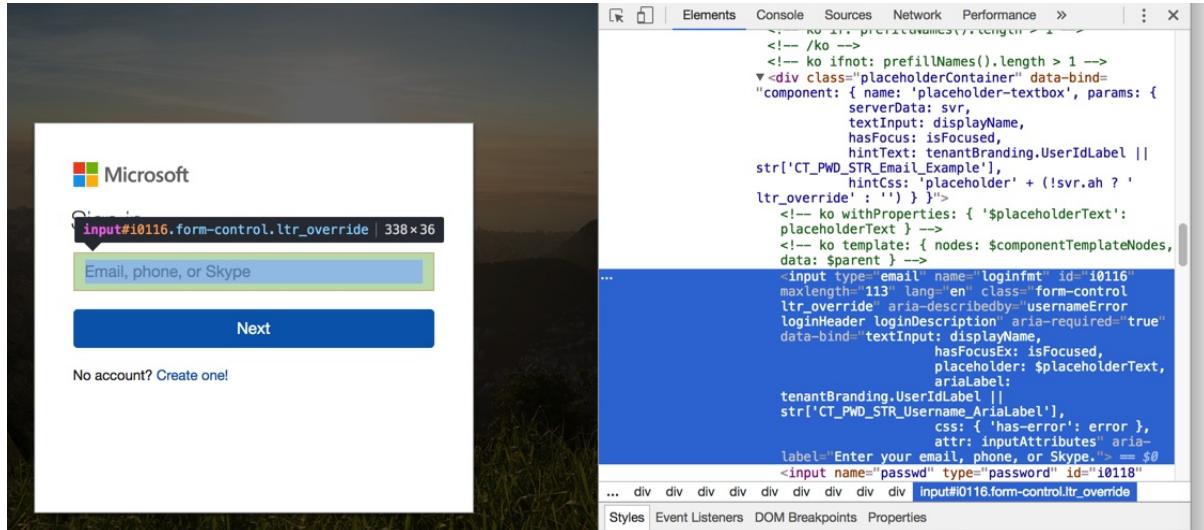
而后来是无意间自己调试，从中间的区域，右键后：



然后一点点点击看子元素：



最后找到真正的input的：



相关代码是：

```
<input type="email" name="loginfmt" id="i0116" .....
    inputAttributes" aria-label="Enter your email, phone, or Skype.">
```

有时候点击按钮后页面刷新且url地址也换了，再去用driver寻找元素之前，先要refresh然后才能找到

但是有时候却又不需要refresh也可以

最后是：

```
driver.refresh()
inputEmailElement = driver.find_element_by_xpath('//div[@class="placeholderContainer"]/inp
```

```
ut[@name="loginfmt"]')
```

或：

```
inputEmailElement = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.XPATH, '//div[@class="placeholderContainer"]/input[@name="loginfmt"]')))
```

好像都可以。

注：

不过还是不知道为何前面的代码：

```
placeholderElement = driver.find_element_by_xpath('//div[@class="phholder"]/div[@class="placeholder"]')
placeholderElement = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.XPATH, '//div[@class="phholder"]/div[@class="placeholder"]')))
placeholderElement = driver.find_element_by_xpath('//div[@class="phholder"]')
phholderElement = driver.find_element_by_class_name('phholder')
logging.info("phholderElement=%s", phholderElement)
placeholderElement = phholderElement.find_element_by_class_name("placeholder")
```

此处已确保 xpath 写的是对的，且查看页面元素的确是存在的，但却还是找不到元素。

## 单个WebElement本身好像不支持截图

详见：[WebElement.screenshot\(filename\)](#) 和：[WebElement.screenshot\\_as\\_png](#)

以为单个WebElement也不支持截图，但是试了试：

```
multipleXPathRule = '//div/.....'
priceSpanElement = driver.find_element_by_xpath(multipleXPathRule)
priceSpanElement.screenshot("priceElement.png")
```

结果报错：

```
selenium.common.exceptions.WebDriverException: Message: unknown command:
session/7160497aa2d4dc2029bcb5c4d2e8045a/element/0.07853595638566757-1/screenshot
```

所以算了，不去管这个了。感觉是单个WebElement本身好像不支持截图

get的url没法back或forward，而navigate的url可以

详见：[URL Loading in Selenium Webdriver: All about get\(\) and navigate\(\) – Make Selenium Easy](#)

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更

新: 2018-01-16 11:58:30

## 附录

下面列出相关参考资料。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2018-01-16 11:50:00

## 参考资料

- 【已解决】Selenium中如何定位元素：百度首页中的输入框
- 【整理】用Chrome或Chromium查看百度首页中各元素的html源码
- 【已解决】Mac中搭建Selenium的Python开发环境
- 【已解决】Mac中下载Selenium的Chrome的driver: ChromeDriver
- 【已解决】Selenium中给百度搜索框中输入文字并触发搜索
- 
- 【已解决】Selenium再次出错：NoSuchElementException: Message: no such element: Unable to locate element
- Mac os上配置selenium并使用python操作web页面\_iluxiaoxiaoniao的博客-CSDN博客\_mac python webdriver
- mac 搭建selenium与ChromeDriver环境 - 简书
- selenium · PyPI
- 2. 快速入门 — Selenium-Python中文文档 2 documentation
- Selenium with Python — Selenium Python Bindings 2 documentation
- Selenium Form WebElement: TextBox, Button, sendkeys(), click()
- java - Selenium Webdriver: Entering text into text field - Stack Overflow
- How to locate and insert a value in a text box (input) using Python Selenium? - Stack Overflow
- java - Using Selenium Web Driver to retrieve value of a HTML input - Stack Overflow
- Selenium WebDriver get text from input field - Stack Overflow
- How to use Selenium to input text in Python - Stack Overflow
- Get value of an input box using Selenium (Python) - Stack Overflow
- 1. Navigating — Selenium Python Bindings 2 documentation
- 

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2021-04-17 12:09:44