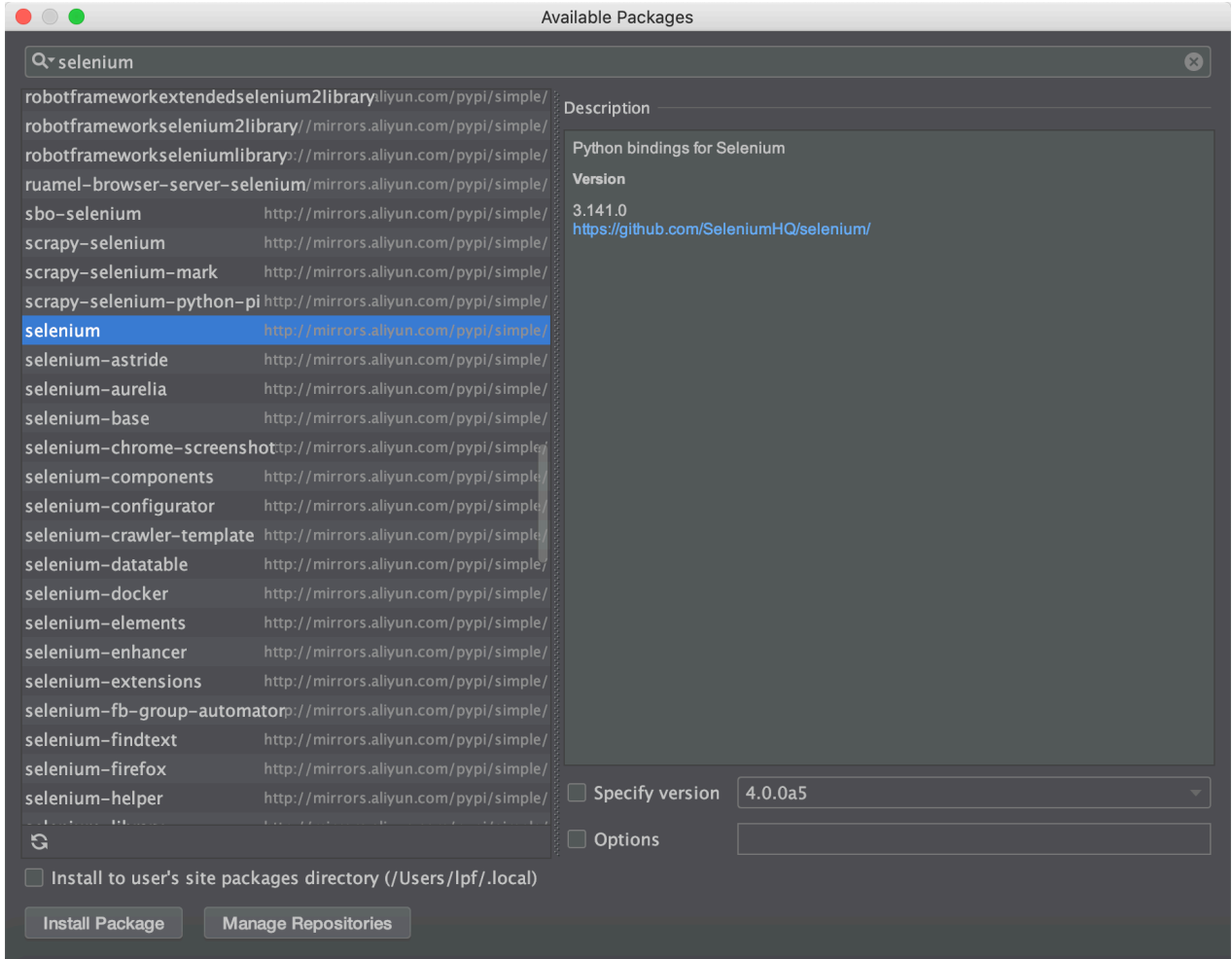


安装

cmd环境下, 用`pip install selenium`

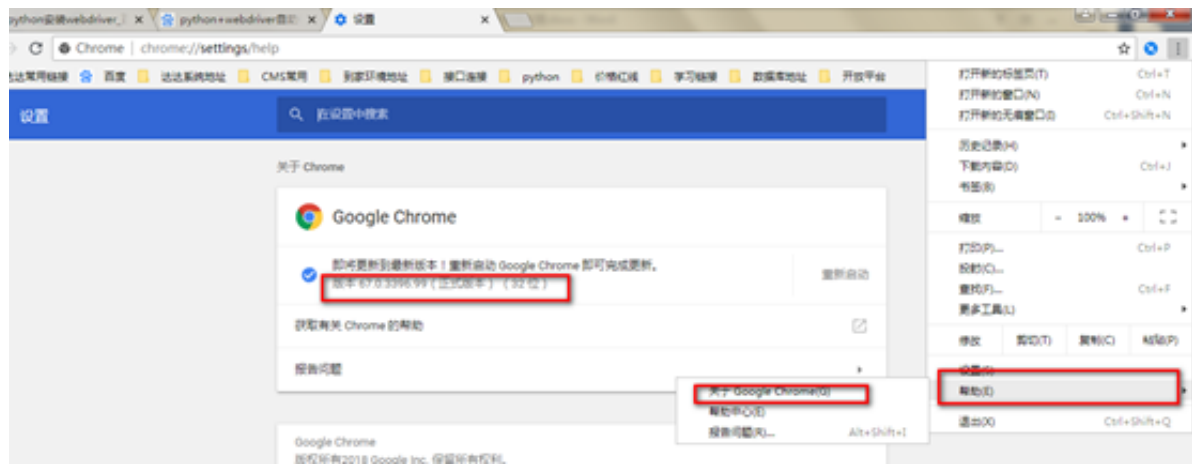
在pycharm的inteptrter里面搜索selenium下载



配置浏览器驱动

1. 确定驱动版本

浏览器驱动, 必须与浏览器版本相符, 首先需要检查浏览器的版本号



2. 下载驱动

下载地址：<http://chromedriver.storage.googleapis.com/index.html>

网址中下载与本机chrome浏览器对应的驱动程序，驱动程序名为chromedriver；

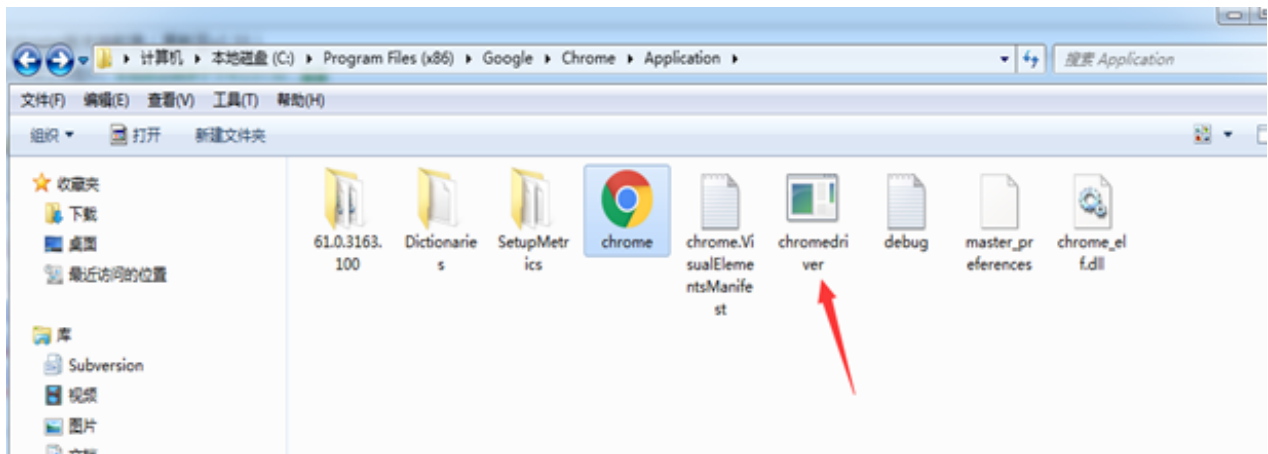
参考下载版本：

chromedriver版本	支持的Chrome版本
v2.40	v66-68
v2.39	v66-68
v2.38	v65-67
v2.37	v64-66
v2.36	v63-65
v2.35	v62-64
v2.34	v61-63
v2.33	v60-62
v2.32	v59-61
v2.31	v58-60
v2.30	v58-60
v2.29	v56-58
v2.28	v55-57
v2.27	v54-56
v2.26	v53-55
v2.25	v53-55
v2.24	v52-54
v2.23	v51-53
v2.22	v49-52
v2.21	v46-50
v2.20	v43-48
v2.19	v43-47
v2.18	v43-46
v2.17	v42-43
v2.13	v42-45
v2.15	v40-43
v2.14	v39-42
v2.13	v38-41
v2.12	v36-40
v2.11	v36-40
v2.10	v33-36
v2.9	v31-34
v2.8	v30-33
v2.7	v30-33
v2.6	v29-32
v2.5	v29-32
v2.4	v29-32

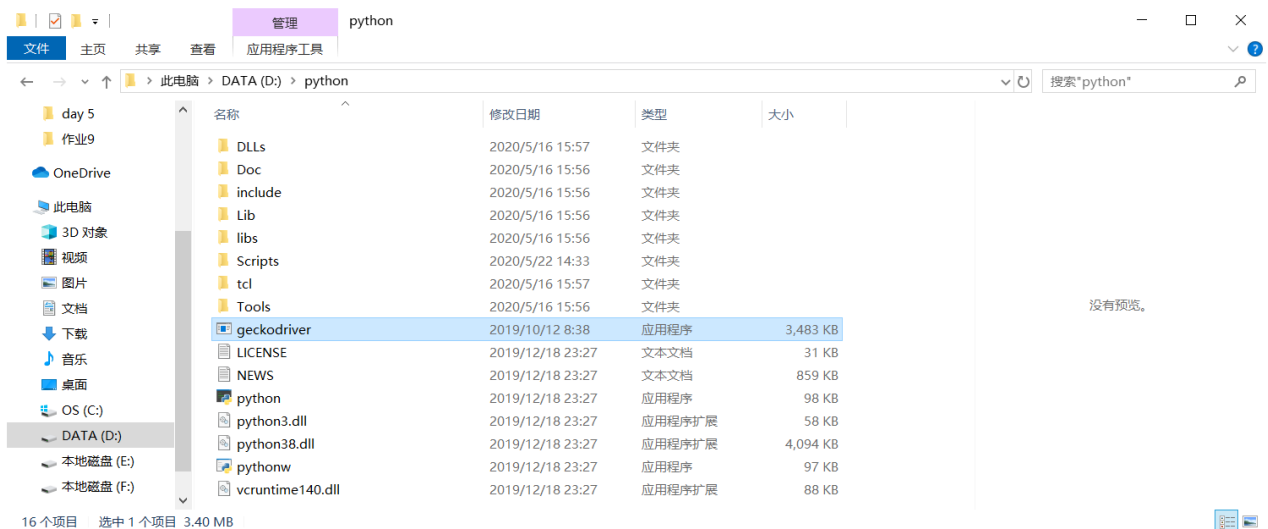
3. 解压到安装路径

下载后把文件解压，然后放到本机chrome浏览器文件路径里，如：

C:\Program Files (x86)\Google\Chrome\Application



2.将解压后文件chromedriver.exe复制到python的安装目录下（我的：D:\python\）



4.调试安装是否成功

操作完后，就可以使用以下代码直接调起浏览器了：

```
driver = new ChromeDriver();
```

```
from selenium import webdriver#导入库
browser = webdriver.Chrome()#声明浏览器
url = 'https://www.baidu.com'
browser.get(url)#打开浏览器预设网址
print(browser.page_source)#打印网页源代码
browser.close()#关闭浏览器
```

什么是selenium

selenium 是一个用于Web应用程序测试的工具。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IE（7, 8, 9, 10, 11），Mozilla Firefox，Safari，Google Chrome，Opera等。selenium 是一套完整的web应用程序测试系统，包含了测试的录制（selenium IDE），编写及运行（Selenium Remote Control）和测试的并行处理（Selenium Grid）。

Selenium的核心Selenium Core基于JUnit，完全由JavaScript编写，因此可以用于任何支持JavaScript的浏览器上。

selenium可以模拟真实浏览器，自动化测试工具，支持多种浏览器，爬虫中主要用来解决JavaScript渲染问题。

这里要说一下比较重要的PhantomJS, PhantomJS是一个而基于WebKit的服务端JavaScript API, 支持Web而不需要浏览器支持，其快速、原生支持各种Web标准：Dom处理，CSS选择器，JSON等等。

PhantomJS可以用于页面自动化、网络监测、网页截屏，以及无界面测试

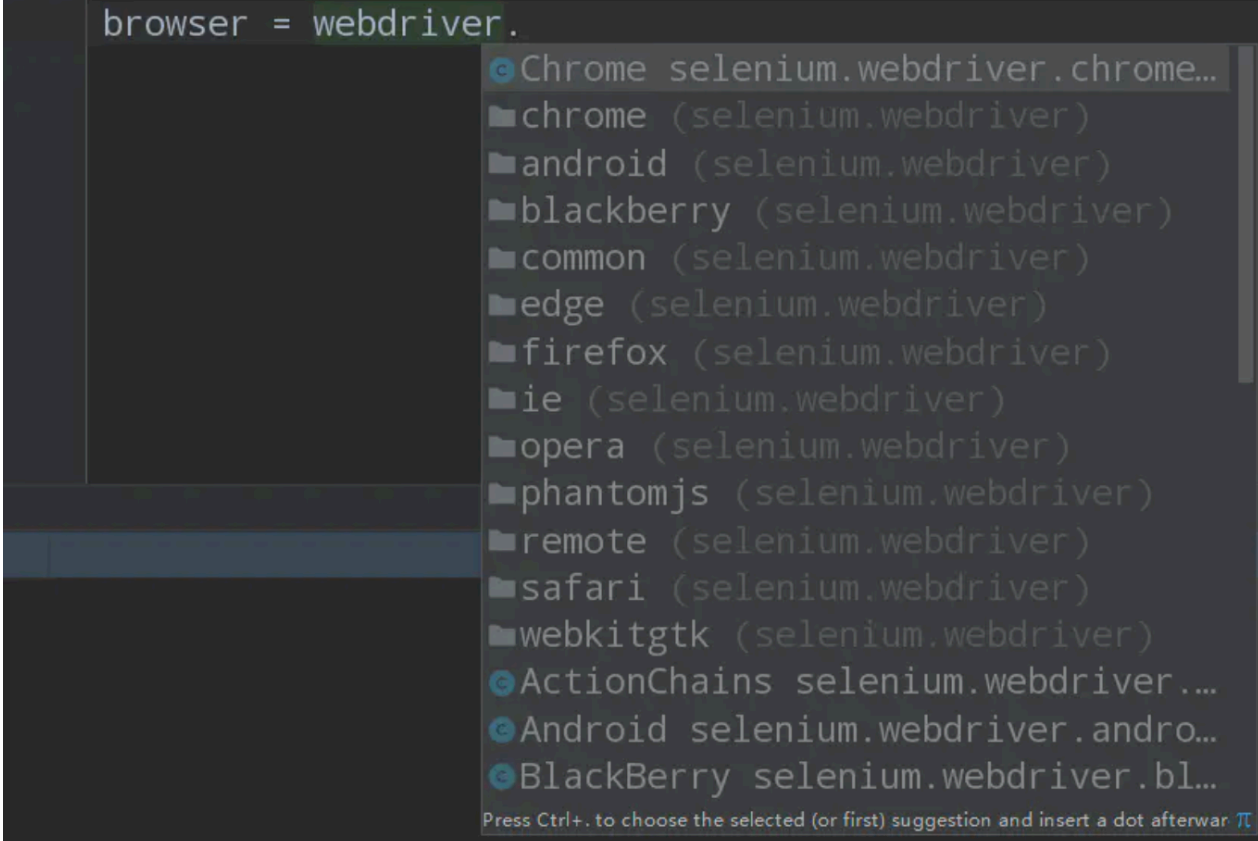
selenium的基本用法

声明浏览器对象

上面我们知道了selenium支持很多的浏览器：

```
from selenium import webdriver

browser = webdriver.
```



The screenshot shows a code editor with a dropdown menu for the `webdriver.` attribute. The dropdown lists several browser drivers: `chrome`, `android`, `blackberry`, `common`, `edge`, `firefox`, `ie`, `opera`, `phantomjs`, `remote`, `safari`, and `webkitgtk`. Each item is followed by `(selenium.webdriver)`. Below these are `ActionChains`, `Android`, and `BlackBerry`, each followed by `selenium.webdriver.bl...`. At the bottom of the dropdown, there is a hint: "Press Ctrl+. to choose the selected (or first) suggestion and insert a dot afterward".

但是如果想要声明并调用浏览器则需要：

```
from selenium import webdriver

browser = webdriver.Chrome()
browser = webdriver.Firefox()
```

这里只写了两个例子，当然了其他的支持的浏览器都可以通过这种方式调用访问页面

```
from selenium import webdriver#导入库
browser = webdriver.Chrome()#声明浏览器
url = 'https://www.baidu.com'
browser.get(url)#打开浏览器预设网址
print(browser.page_source)#打印网页源代码
browser.close()#关闭浏览器
```

上述代码运行后，会自动打开Chrome浏览器，并登陆百度打印百度首页的源代码，然后关闭浏览器

查找元素

单个元素查找

```
from selenium import webdriver#导入库
browser = webdriver.Chrome()#声明浏览器
url = 'https://www.taobao.com'
browser.get(url)#打开浏览器预设网址
input_first = browser.find_element_by_id('q')
input_two = browser.find_element_by_css_selector('#q')
print(input_first)
print(input_two)
```

这里我们通过2种不同的方式去获取响应的元素，第一种是通过id的方式，第二个中是CSS选择器，结果都是相同的。

输出如下：

```
<selenium.webdriver.remote.webelement.WebElement
(session="9aaa01da6545ba2013cc432bcb9abfda", element="0.5325244323105505-1")>
<selenium.webdriver.remote.webelement.WebElement
(session="9aaa01da6545ba2013cc432bcb9abfda", element="0.5325244323105505-1")>
```

这里列举一下常用的查找元素方法：

- find_element_by_name
- find_element_by_id
- 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

下面这种方式是比较通用的一种方式：这里需要记住By模块所以需要导入

from selenium.webdriver.common.by import By

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

browser = webdriver.Chrome()
url = 'https://www.taobao.com'
browser.get(url)
input_1 = browser.find_element(By.ID, 'q')
print(input_1)
```

当然这种方法和上述的方式是通用的，browser.find_element(By.ID,"q")这里By.ID中的ID可以替换为其他几个

我个人比较倾向于css

多个元素查找

其实多个元素和单个元素的区别，举个例子：find_elements,单个元素是find_element,其他使用上没什么区别，通过其中的一个例子演示：

```
from selenium import webdriver

browser = webdriver.Chrome()
url = 'https://www.taobao.com'
browser.get(url)
input = browser.find_elements_by_css_selector('.service-bd li')
print(input)
browser.close()
```

输出为一个列表形式：

```
[<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-1")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-2")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-3")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-4")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-5")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-6")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-7")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-8")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-9")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-10")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-11")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-12")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-13")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-14")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-15")>,
<selenium.webdriver.remote.webelement.WebElement
(session="42d192ca36f75170ab489e4839df0980", element="0.73211490098068-16")>]
```

当然上面的方式也是可以通过导入from selenium.webdriver.common.by import By 这种方式实现

```
lis = browser.find_elements(By.CSS_SELECTOR,'.service-bd li')
```

同样的在单个元素中查找的方法在多个元素查找中同样存在：

```
find_elements_by_name
```

```
find_elements_by_id
```

```
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
```

元素交互操作

对于获取的元素调用交互方法

```

from selenium import webdriver
import time
browser = webdriver.Chrome()
browser.get(url='https://www.baidu.com')
time.sleep(2)
input = browser.find_element_by_css_selector('#kw')
input.send_keys('韩国女团')
time.sleep(2)
input.clear()
input.send_keys('后背摇')
button = browser.find_element_by_css_selector('#su')
button.click()
time.sleep(10)
browser.close()

```

运行的结果可以看出程序会自动打开Chrome浏览器并打开百度页面输入韩国女团,然后删除,重新输入后背摇,并点击搜索

Selenium所有的api文档: http://selenium-python.readthedocs.io/api.html#module-selenium.webdriver.common.action_chains

交互动作

将动作附加到动作链中串行执行

```

from selenium import webdriver
from selenium.webdriver import ActionChains

browser = webdriver.Chrome()

url = "http://www.runoob.com/try/try.php?filename=jqueryui-api-droppable"
browser.get(url)
browser.switch_to.frame('iframeResult')
source = browser.find_element_by_css_selector('#draggable')
target = browser.find_element_by_css_selector('#droppable')
actions = ActionChains(browser)
actions.drag_and_drop(source, target)
actions.perform()

```

更多操作参考: http://selenium-python.readthedocs.io/api.html#module-selenium.webdriver.common.action_chains

执行JavaScript

这是一个非常有用的方法,这里就可以直接调用js方法来实现一些操作,下面的例子是通过登录知乎然后通过js翻到页面底部,并弹框提示


```
from selenium import webdriver
browser = webdriver.Chrome()
browser.get("http://www.zhihu.com/explore")
browser.execute_script('window.scrollTo(0,document.body.scrollHeight)')
browser.execute_script('alert("To Bottom")')
```

selenium webdriver——JS滚动到最底部

JS控制滚动条的位置：

`window.scrollTo(x,y);`

竖向滚动条置顶 `window.scrollTo(0,0);`

竖向滚动条置底 `window.scrollTo(0,document.body.scrollHeight)`

JS控制TextArea滚动条自动滚动到最下部

`document.getElementById("textarea").scrollTop = document.getElementById("textarea").scrollHeight`

获取元素属性

`get_attribute('class')`

```
from selenium import webdriver
import time
browser = webdriver.Chrome()
browser.get("http://www.zhihu.com/explore")
logo = browser.find_element_by_css_selector('.zu-top-link-logo')
print(logo)
print(logo.get_attribute('class'))
print(logo.get_attribute('id'))
time.sleep(2)
browser.quit()
```

输出如下：

```
<selenium.webdriver.remote.webelement.WebElement
(session="b72dbd6906debbca7d0b49ab6e064d92", element="0.511689875475734-1")>
zu-top-link-logo
zh-top-link-logo
```

获取文本值

`text`

```
from selenium import webdriver
browser = webdriver.Chrome()
browser.get("http://www.zhihu.com/explore")
logo = browser.find_element_by_css_selector('.zu-top-link-logo')
print(logo)
print(logo.text)
```

输出如下:

```
<selenium.webdriver.remote.webelement.WebElement
(session="ce8814d69f8e1291c88ce6f76b6050a2", element="0.9868611170776878-1")>
知乎
```

获取ID, 位置, 标签名

id
location
tag_name
size

```
from selenium import webdriver

browser = webdriver.Chrome()
url = 'https://www.zhihu.com/explore'
browser.get(url)
input = browser.find_element_by_css_selector('.zu-top-add-question')
print(input.id)
print(input.location)
print(input.tag_name)
print(input.size)
```

输出如下:

```
0.022998219885927318-1
{'x': 759, 'y': 7}
button
{'height': 32, 'width': 66}
```