目录

Selenium知识总结

- 最新版本： `v2.1`
- 更新时间： `20210728`

简介

介绍PC端Web自动化最主流工具Selenium。先介绍了如何初始化Selenium开发环境，再介绍基本操作，包括查找元素、输入文字、点击元素、等待元素出现、获取元素属性等；以及总结一些开发经验，包括如何等待元素出现、select不能用于ul、偶尔Chrome右键找到的元素不是你要的，偶尔刷新后才能找到元素等；给出具体实例，比如模拟百度搜索并解析搜索结果；给出相关文档和资料。

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

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
     |████████████████████████████████| 904 kB 1.2 MB/s
Requirement already satisfied: urllib3 in /Users/crifan/.py
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`

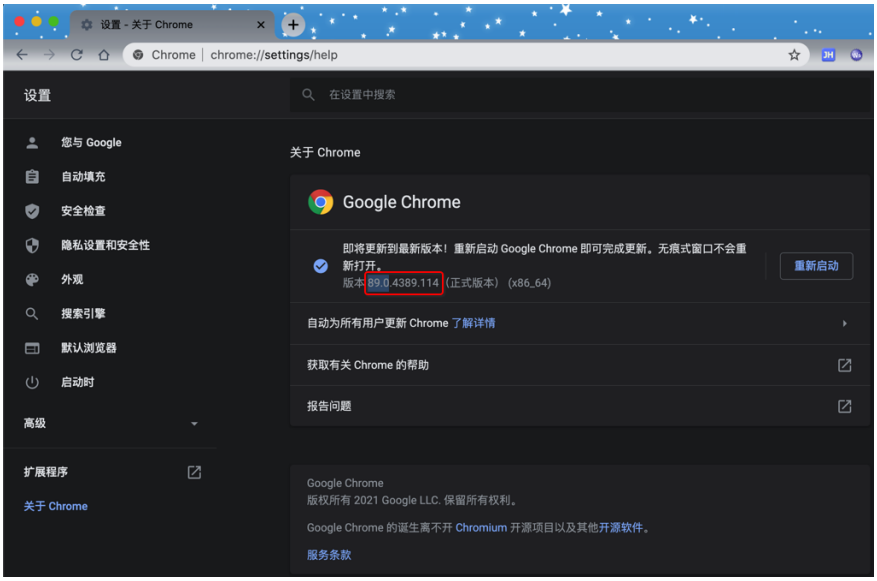
- `Chrome 的 driver：ChromeDriver = chromedriver`

安装Chrome的driver：`ChromeDriver`

下载 `ChromeDriver`

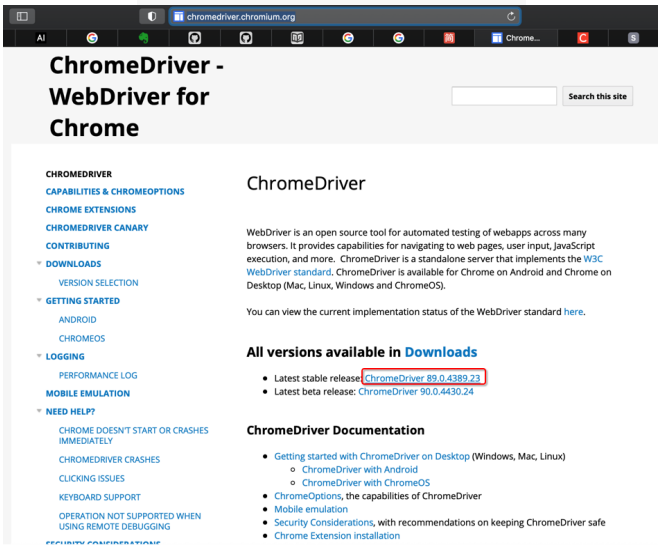
要下载和你的Chrome版本一致的ChromeDriver

此处查看到Chrome的版本是：`89.0`

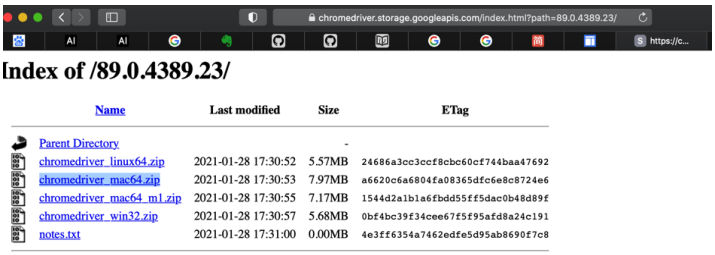


所以要下载的 `ChromeDriver` 也是要于此版本一致的，即下载 `ChromeDriver 89.0` 的版本

- 下载源1：**Chrome官网**
  - ChromeDriver - WebDriver for Chrome
    - 此时最新版是：`ChromeDriver 89.0.4389.23`



  - Index of /89.0.4389.23/
  - chromedriver_mac64.zip



- 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

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`

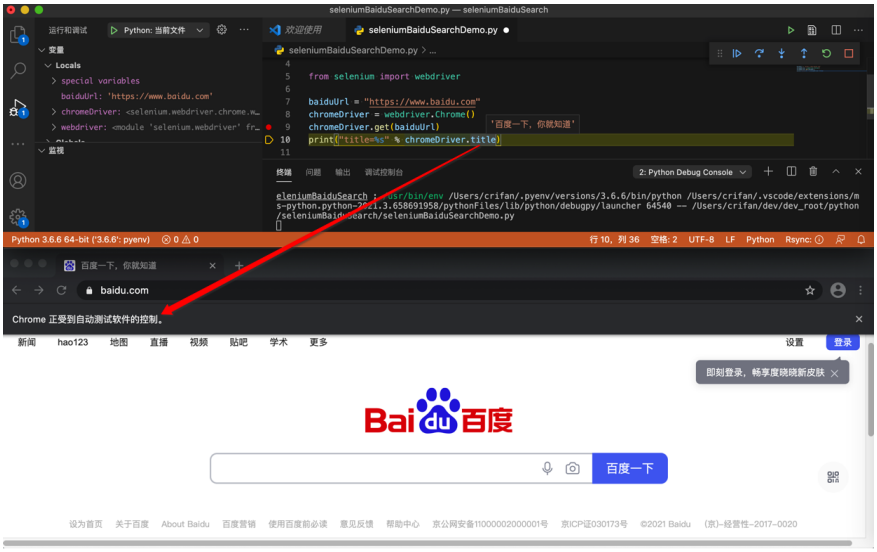
写测试代码，确认环境正常

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`

基本操作

下面介绍Selenium的基本操作。

注：对于元素定位等操作，会涉及到用不同方式，具体语法详见独立教程：

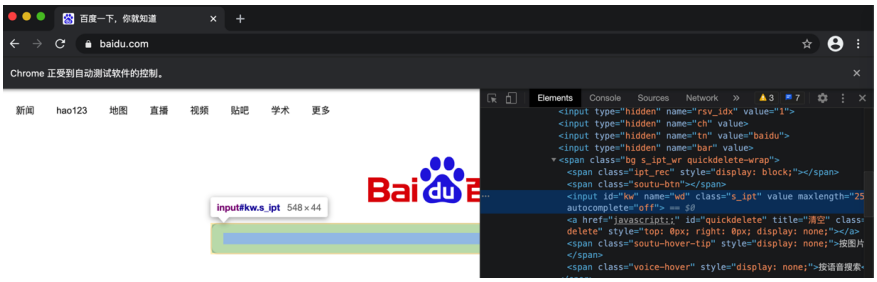
- Xpath
  - XPath知识总结

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="
```

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 (s
    ```

  - 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

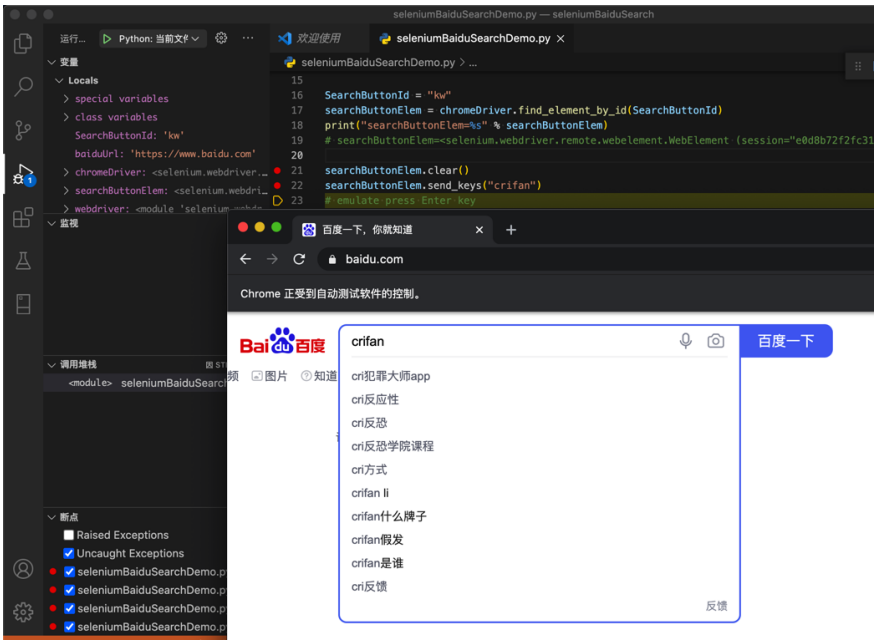
（给元素）输入文字

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()
```

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

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_b
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(Ba
print("baiduSearchButtonElem=%s" % baiduSearchButtonElem)
baiduSearchButtonElem.click()
print("Clicked button %s" % baiduSearchButtonElem)
```

等待元素出现

很多时候，会遇到类似问题：

查找元素，点击触发搜索类的操作后，页面刷新后，继续去寻找搜索结果元素，调试时代码逻辑没问题，但直接运行却找不到搜索结果

这时候，往往是由于：没有等待搜索完毕

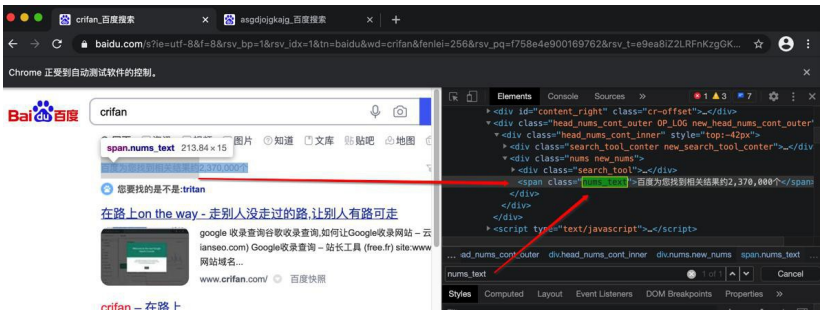
而典型的解决办法是，去研究搜索结果完毕后，肯定会显示的某些元素。然后去判断和等待该元素是否出现

- 该元素显示了：说明搜索结果完成了
- 该元素还没显示：说明搜索结果还没完成，需要继续等待

对于**等待元素出现**的操作，`Selenium`中有专门的函数实现这个逻辑。

举例：等待百度搜索结果完成后的某元素出现

对于百度页面搜索结果，完成时，必然会出现的一个元素是：

- 

对应html：

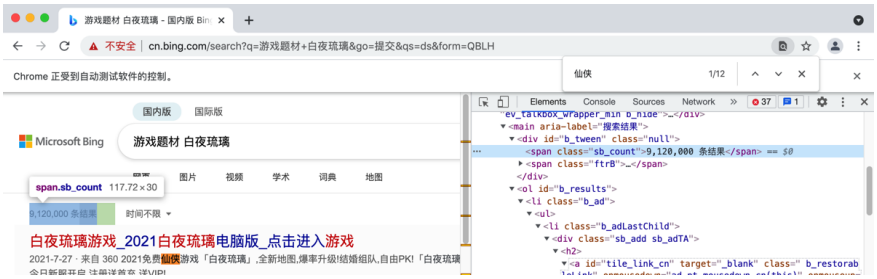
```
<span class="nums_text">百度为您找到相关结果约2,370,000个</spar
```

等待该元素出现的对应代码是：

```
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.common.by import By
from selenium.webdriver.support import expected_conditions

MaxWaitSeconds = 10
numTextElem = WebDriverWait(chromeDriver, MaxWaitSeconds).u
    EC.presence_of_element_located((By.XPATH, "//span[@clas
)
print("Search complete, showing: %s" % numTextElem)
```

举例：等待bing必应搜索结果完成后的某元素出现

对于bing必应搜索结果完成后，必然会出现的一个元素是：



html:

```
<span class="sb_count">9,120,000 条结果</span>
```

等待该元素出现的对应代码是：

```
# wait bing search complete -> show some element
# <span class="sb_count">9,120,000 条结果</span>
MaxWaitSeconds = 10
searchCountElem = WebDriverWait(driver, MaxWaitSeconds).unt
    EC.presence_of_element_located((By.XPATH, "//span[@clas
)
```

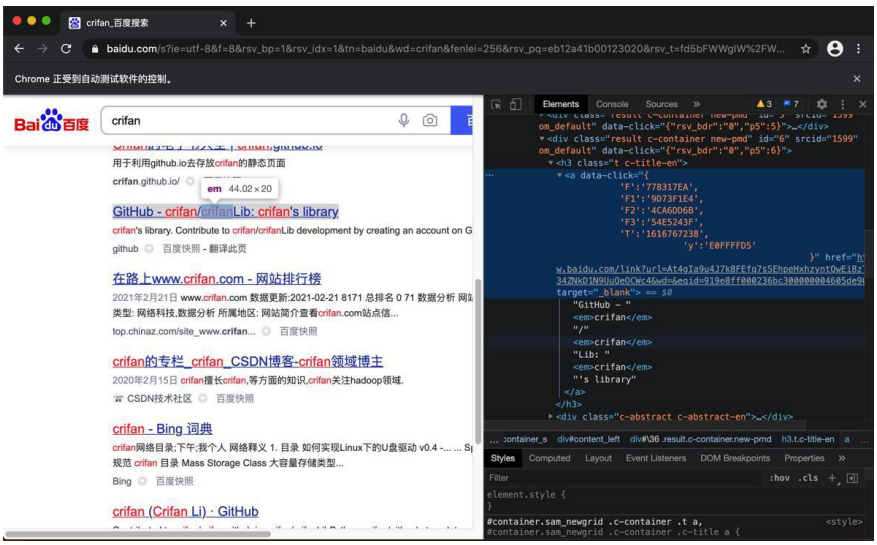
获取元素属性

获取元素属性的典型用法，比如：

- a元素
  - 获取a的href链接
    - `aElement.get_attribute("href")`
  - 获取a的文本值
    - `aElement.text`

举例：解析提取百度搜索结果

对于页面：



的html是：

```
<h3 class="t c-title-en"><a data-click="{
    'F':'778317EA',
    'F1':'9D73F1E4',
    'F2':'4CA6DD6B',
    'F3':'54E5243F',
    'T':'1616767238',
        'y':'E0FFFFD5'
            }" href="https://www.baidu.com/link?url=At4gIa
```

已经通过代码：

```
searchResultAList = chromeDriver.find_elements_by_xpath("/
```

然后就可以用 `htmlElement.get_attribute("href")` 去获取 `href` 的url链接：

```
for curIdx, curSearchResultAElem in enumerate(searchResultA
  print("%s [%d] %s" % ("-"*20, curIdx, "-"*20))
  aHref = curSearchResultAElem.get_attribute("href")
  print("aHref=%s" % aHref)
```

类似的，想要获取文本值，用 `text` ：

```
  aText = curSearchResultAElem.text
  print("aText=%s" % aText)
```

此处输出：

```
-------------------- [0] --------------------
aHref=http://www.baidu.com/link?url=LMF5vQH-QgOuEhaq5huV3bI
aText=在路上on the way - 走别人没走过的路,让别人有路可走
-------------------- [1] --------------------
aHref=http://www.baidu.com/link?url=n4QoZVrJ5gncFIpJZhRcdmc
aText=crifan - 在路上
...
```

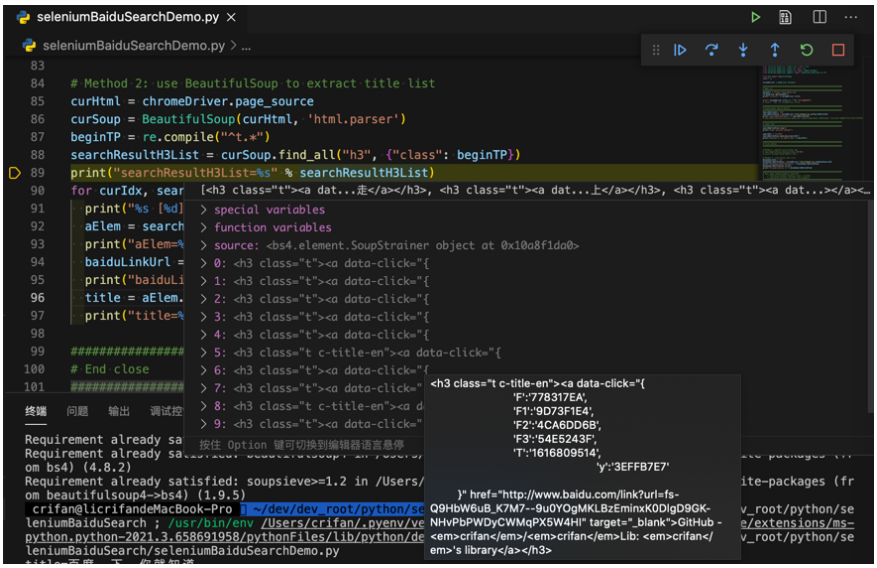
特殊：对于html的解析，一般更常用专用的库： `BeautifulSoup`

对于html的解析，元素的获取等操作，往往会换专用的html解析库： `BeautifulSoup`

举例，此处对应代码：

```
# Method 2: use BeautifulSoup to extract title list
curHtml = chromeDriver.page_source
curSoup = BeautifulSoup(curHtml, 'html.parser')
beginTP = re.compile("^t.*")
searchResultH3List = curSoup.find_all("h3", {"class": begi
print("searchResultH3List=%s" % searchResultH3List)
for curIdx, searchResultH3Item in enumerate(searchResultH3
  print("%s [%d] %s" % ("-"*20, curIdx, "-"*20))
  aElem = searchResultH3Item.find("a")
  # print("aElem=%s" % aElem)
  baiduLinkUrl = aElem.attrs["href"]
  print("baiduLinkUrl=%s" % baiduLinkUrl)
  title = aElem.text
  print("title=%s" % title)
```

调试效果：



最终，同样的输出：

```
-------------------- [0] --------------------
baiduLinkUrl=http://www.baidu.com/link?url=DVUbOETLyMZLC5c_
title=在路上on the way - 走别人没走过的路,让别人有路可走
-------------------- [1] --------------------
baiduLinkUrl=http://www.baidu.com/link?url=xA8mzlRBwfRb_I-F
title=crifan - 在路上
...
```

Selenium心得和总结

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(multipleXpa
priceSpanElement.screenshot("priceElement.png")
```

结果报错：

```
selenium.common.exceptions.WebDriverException: Message:
unknown command:
session/7160497aa2d4dc2029bcb5c4d2e8045a/element/0.078535956
38566757-1/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

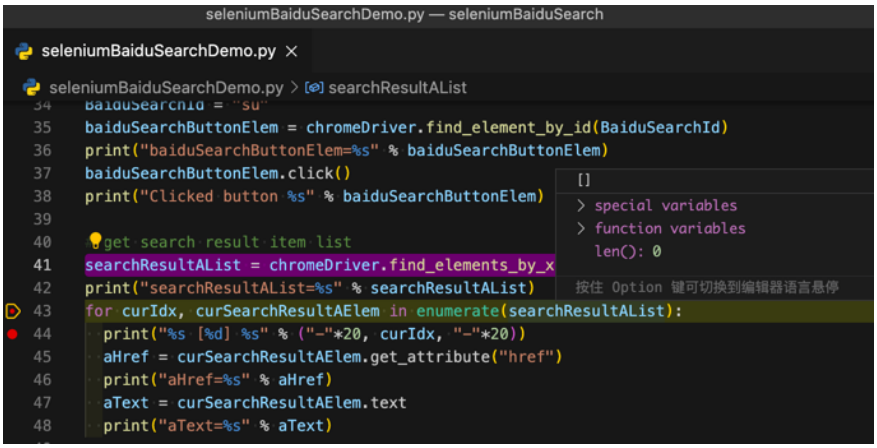
代码运行时却找不到元素

现象：查找元素的代码

```
searchResultAList = chromeDriver.find_elements_by_xpath("//
```

调试时可以找到元素。

但是直接运行代码，却找不到元素，返回是空：



原因：（网页）页面元素重新加载了，比如百度搜索导致页面重新加载，显示搜索结果内容，但是此时代码运行时，搜索结果还没加载出来，导致搜不到。

而调试时，由于有足够的暂停的时间，使得页面已加载新的搜索结果内容，所以再继续调试，可以搜索到。

解决办法：加上等待机制：等待页面加载完毕。

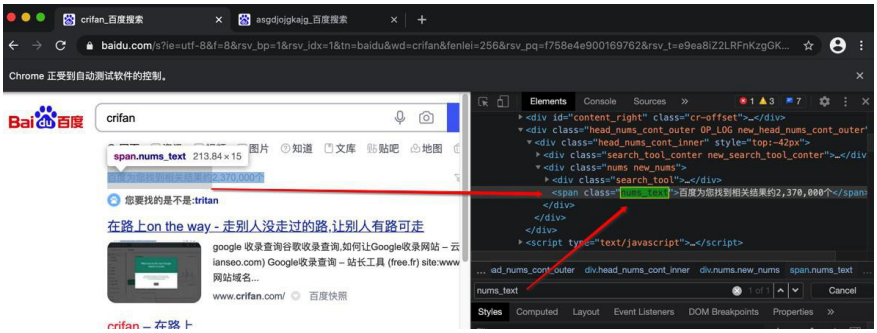
而判断页面加载完毕，则是需要具体问题具体分析。

核心逻辑是：找到加载后的页面中，必然会出现的元素，作为判断的依据，判断该元素可见，则视为页面的确已加载完毕。

操作步骤：

此处经过调试，百度搜索结果中一定会出现：

百度为您找到相关结果约 xxx



对应的html是：

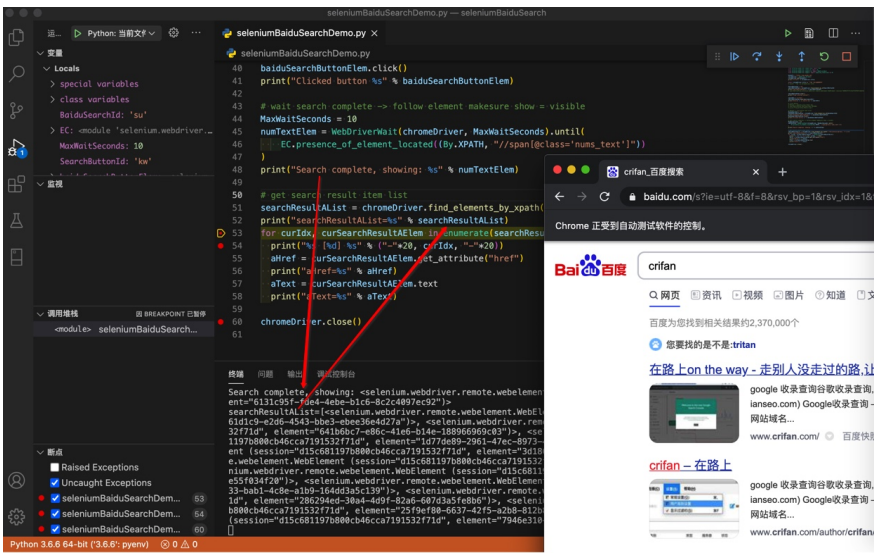
```
<span class="nums_text">百度为您找到相关结果约2,370,000个</span
```

对应的，等待一段时间，确保该元素出现的代码是：

```
# wait search complete -> follow element makesure show = v
MaxWaitSeconds = 10
numTextElem = WebDriverWait(chromeDriver, MaxWaitSeconds).u
    EC.presence_of_element_located((By.XPATH, "//span[@clas
)
print("Search complete, showing: %s" % numTextElem)
```

即可解决问题。

后续代码可以找到元素了：



select不能用于ul

即， `select` 无法用于，非 `select` 的 `option` 下拉选项列表

`select` 只能用于

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

才可以。其他的元素，比如之前遇到的：

【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。

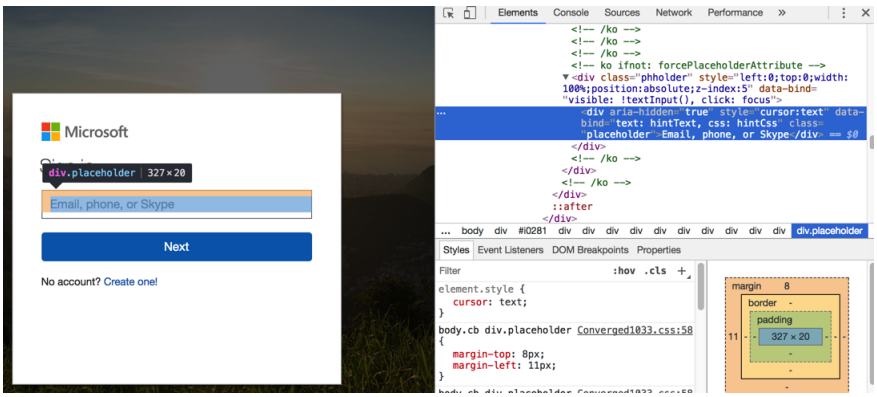
相关代码如下：

```
cartNumOptionElemList = driver.find_elements_by_xpath('//u
cartNumOptionCount = len(cartNumOptionElemList)
logging.info("cartNumOptionElemList=%s,cartNumOptionCount=
if cartNumOptionCount < gCfg["msStore"]["onceBuyNum"]:
    logging.error("Current Cart select max number %s < expe
    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(st
# logging.info("aLinkElem=%s", aLinkElem)
# aLinkElem.click()
```

偶尔Chrome右键元素不是你要的

即：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：

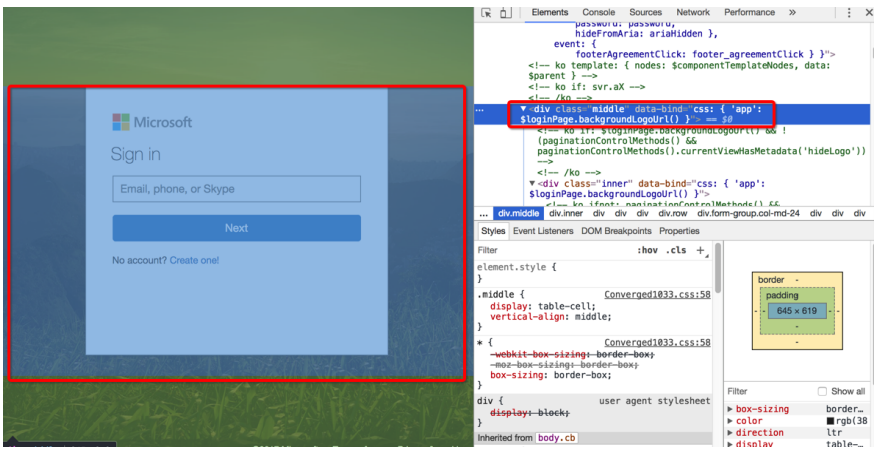


是个：

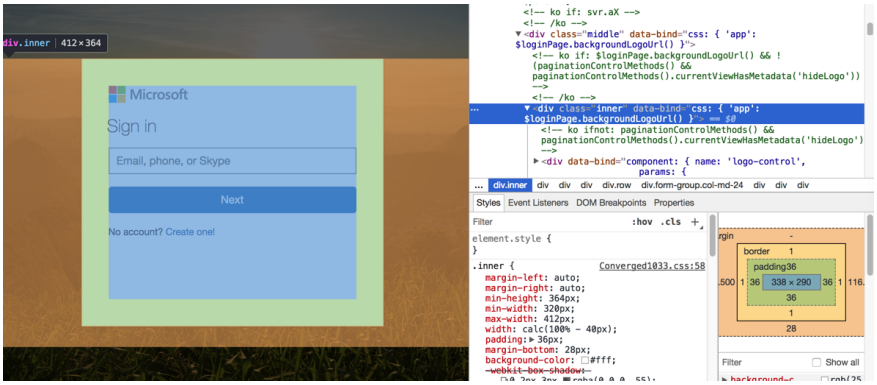
```
<div class="placeholder">
```

但是其实此处要找的是 **可以允许输入的input输入框**

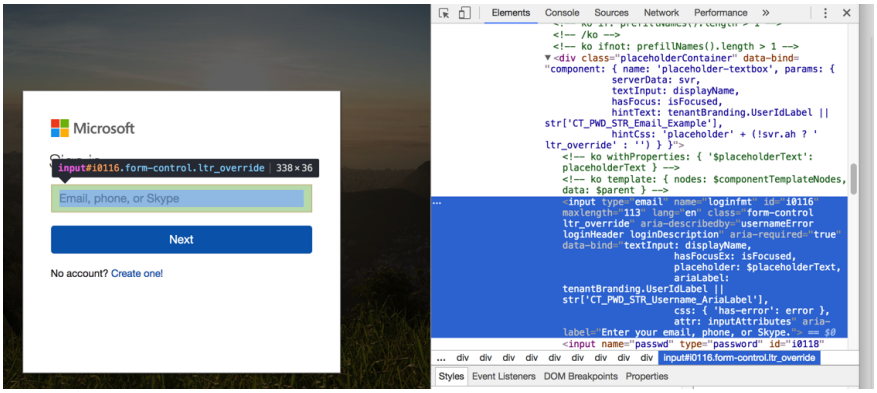
而后来是无意间自己调试，从中间的区域，右键后：



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" ......
                attr: inputAttributes" aria-label="Enter y
```

偶尔刷新后才能找到元素

即：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[@c
```

或：

```
inputEmailElement = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.XPATH, '//div[@class
```

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

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。

找不到元素会抛异常

Selenium的 `find_element_by_xpath` ，是找单个元素，如果找不到，则会抛异常 `NoSuchElementException`

->和常见的逻辑，找不到只返回空 `None` ，不一样

-》代码中要注意，根据你的需要，可能需要捕获异常

举例：

```
for curIdx, eachNormalLi in enumerate(normalLiList):
    try:
        descItem = eachNormalLi.find_element_by_xpath(".//
    except:
        logging.error("Failed to find description(b_captio
        continue
```

而对应的，找多个元素，返回列表的 `find_elements_by_xpath` ，则不会排除异常，如果找不到，则返回空列表 `[]` 而已

总结：

- 找单个元素的 find_element_by_xpath
  - 如果找不到，会抛出异常 `NoSuchElementException`
    - 你需要根据你的情况决定，是否要捕获异常
  - 对应官网解释

    > find_element_by_xpath(xpath)
    >
    > Finds an element by xpath.
    >
    > Args:
    >
    > - xpath - The xpath locator of the element to find.
    >
    > Returns:
    >
    > - WebElement - the element if it was found
    >
    > Raises::
    >
    > - NoSuchElementException - if the element wasn't found
    >
    > Usage:
    >
    > element = driver.find_element_by_xpath('//div/td[1]')

- 找多个元素的 find_elements_by_xpath
  - 如果找不到，只是返回空列表，不会抛异常 -》 无需捕获异常
  - 对应官网解释

> find_elements_by_xpath(xpath)
>
> Finds multiple elements by xpath.
>
> Args:
>
> - xpath - The xpath locator of the elements to be found.
>
> Returns:
>
> - list of WebElement - a list with elements if any was found. An empty list if not
>
> Usage:
>
> elements = driver.find_elements_by_xpath(“//div[contains(@class, ‘foo’)]”)

举例

下面给出Selenium的一些实例，来说明如何使用Selenium实现Web自动化。

模拟百度搜索

此处用Selenium，模拟百度搜索，并提取第一页搜索结果的信息。

代码

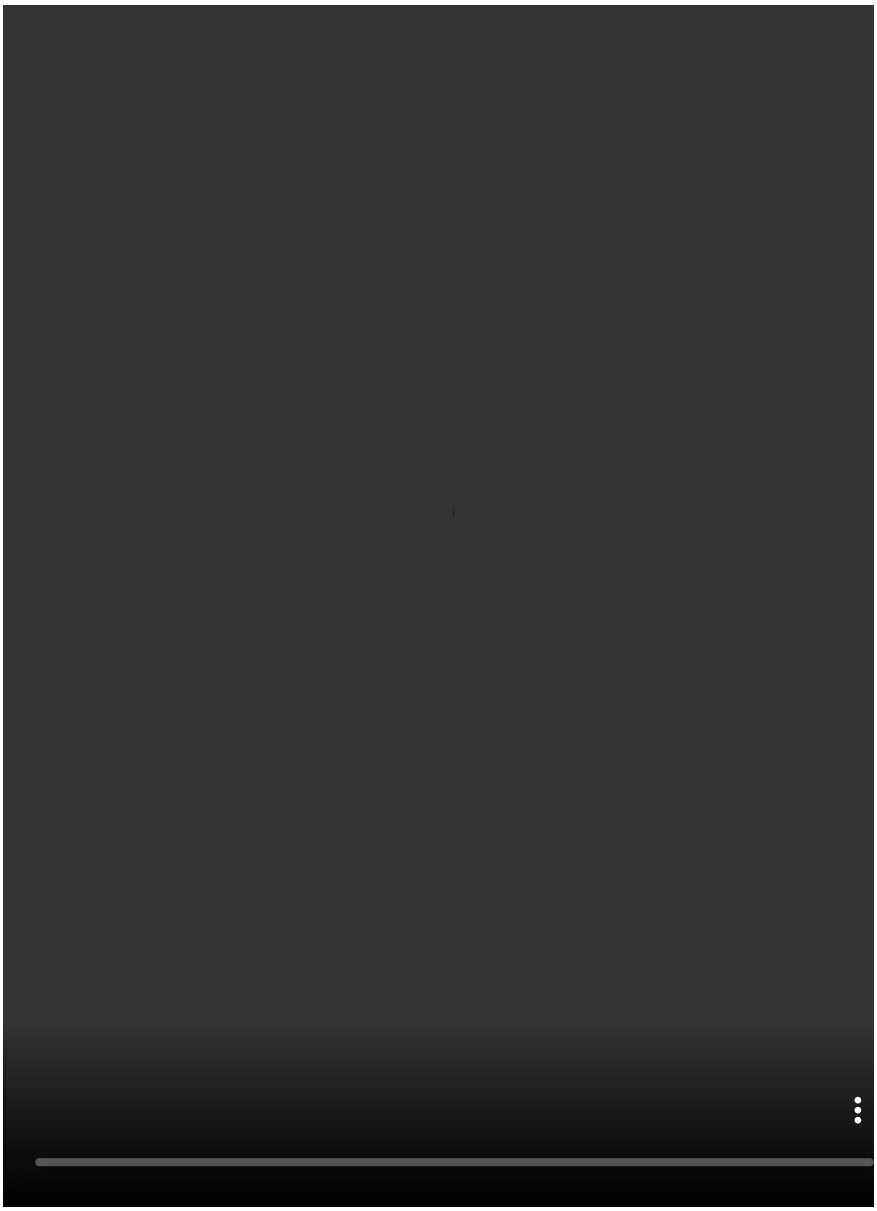
- 文件下载：seleniumDemoBaiduSearch.py
- 直接贴出源码

```
# Function: demo selenium do baidu search and extract resu
# Author: Crifan Li
# Update: 20210327

from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions

from bs4 import BeautifulSoup
import re

chromeDriver = webdriver.Chrome()

################################################################
# Open url
################################################################
baiduUrl = "https://www.baidu.com"
chromeDriver.get(baiduUrl)
print("title=%s" % chromeDriver.title)

assert chromeDriver.title == "百度一下，你就知道"
# assert '百度' in chromeDriver.title

################################################################
# Find/Locate search button
################################################################
SearchButtonId = "kw"
searchButtonElem = chromeDriver.find_element_by_id(SearchBu
print("searchButtonElem=%s" % searchButtonElem)
# searchButtonElem=<selenium.webdriver.remote.webelement.We

################################################################
# Input text
################################################################
searchButtonElem.clear()
print("Clear existed content")

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(Ba
print("baiduSearchButtonElem=%s" % baiduSearchButtonElem)
baiduSearchButtonElem.click()
print("Clicked button %s" % baiduSearchButtonElem)

################################################################
# Wait page change/loading completed
#   -> following element makesure show = visible
#   -> otherwise possibly can NOT find elements
################################################################
MaxWaitSeconds = 10
numTextElem = WebDriverWait(chromeDriver, MaxWaitSeconds).u
    EC.presence_of_element_located((By.XPATH, "//span[@clas
)
print("Search complete, showing: %s" % numTextElem)

################################################################
# Extract result
################################################################

# Method 1: use Selenium to extract title list
searchResultAList = chromeDriver.find_elements_by_xpath("//
print("searchResultAList=%s" % searchResultAList)
searchResultANum = len(searchResultAList)
print("searchResultANum=%s" % searchResultANum)
for curIdx, curSearchResultAElem in enumerate(searchResultA
  curNum = curIdx + 1
  print("%s [%d] %s" % ("-"*20, curNum, "-"*20))
  baiduLinkUrl = curSearchResultAElem.get_attribute("href")
  print("baiduLinkUrl=%s" % baiduLinkUrl)
  title = curSearchResultAElem.text
  print("title=%s" % title)

# # Method 2: use BeautifulSoup to extract title list
# curHtml = chromeDriver.page_source
# curSoup = BeautifulSoup(curHtml, 'html.parser')
# beginTP = re.compile("^t.*")
# searchResultH3List = curSoup.find_all("h3", {"class": beg
# print("searchResultH3List=%s" % searchResultH3List)
# searchResultH3Num = len(searchResultH3List)
# print("searchResultH3Num=%s" % searchResultH3Num)
# for curIdx, searchResultH3Item in enumerate(searchResultH
#   curNum = curIdx + 1
#   print("%s [%d] %s" % ("-"*20, curNum, "-"*20))
#   aElem = searchResultH3Item.find("a")
#   # print("aElem=%s" % aElem)
#   baiduLinkUrl = aElem.attrs["href"]
#   print("baiduLinkUrl=%s" % baiduLinkUrl)
```

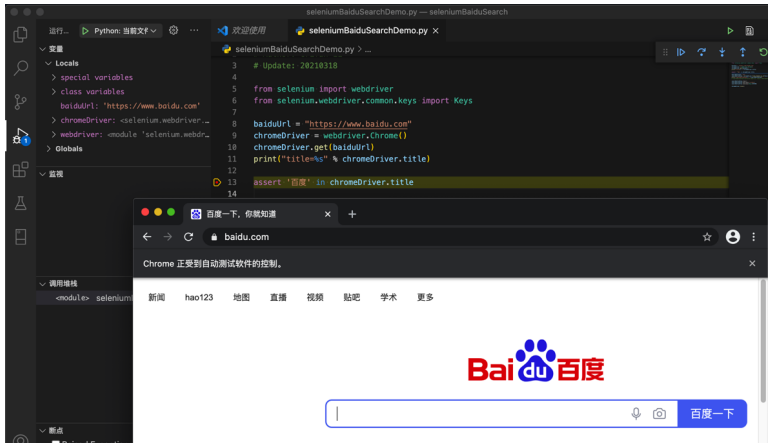
```
#   title = aElem.text
#   print("title=%s" % title)

#############################################################
# End close
#############################################################
chromeDriver.close()
```

视频

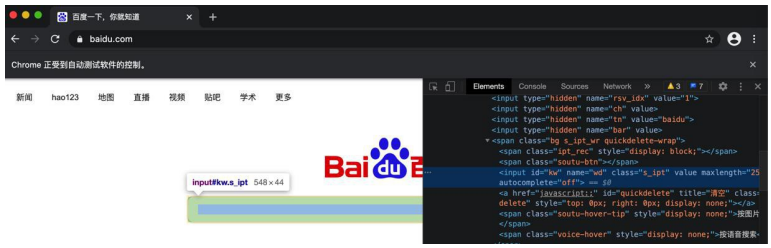


相关图片

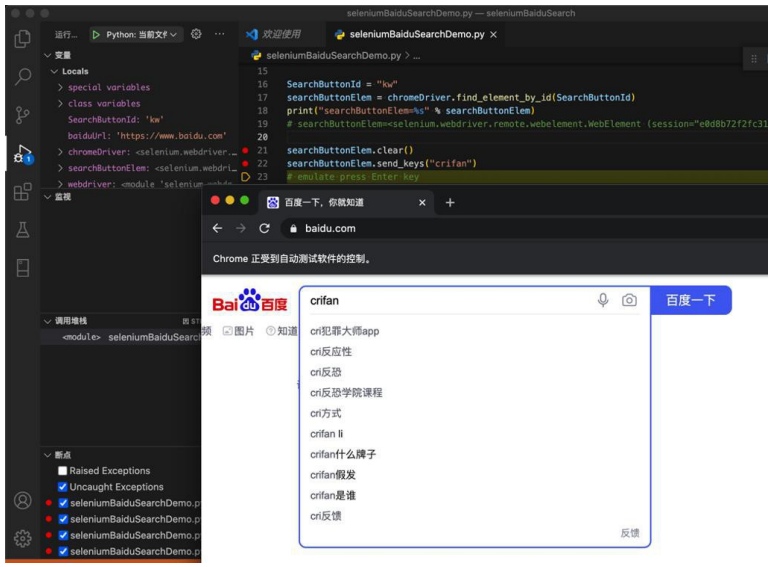
- 打开百度首页



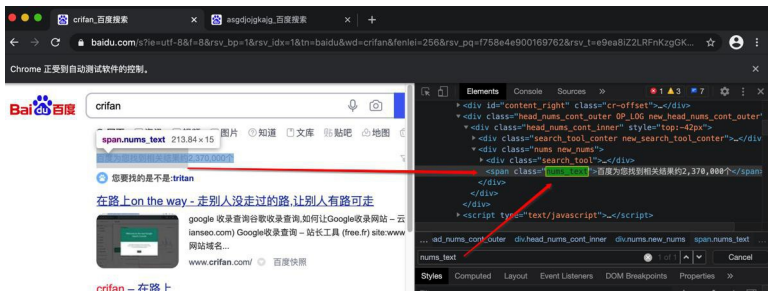
- 调试输入框的html



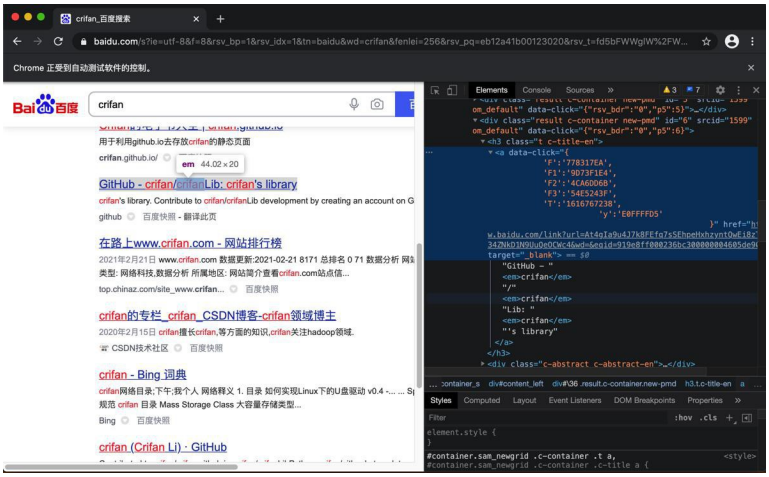
- 输入框中可以输入搜索关键字：crifan



- 研究搜索结果中必然出现的字段

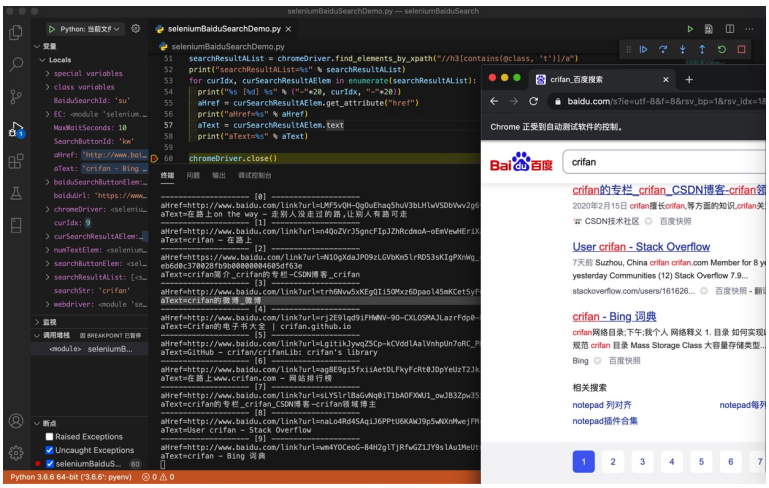


- 调试搜索结果每一项的html



- 点击触发搜索是可以的

- 最终解析出百度搜索结果



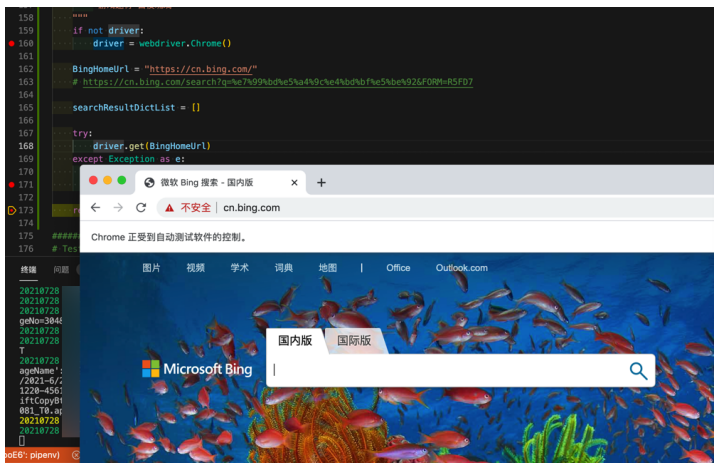
模拟bing必应搜索

此处用Selenium（的Chrome）模拟：

- 打开 bing必应搜索
  - https://cn.bing.com/
- 输入文字
- 点击按钮触发搜索
- 从第一页搜索结果中，解析出搜索结果
  - 标题 `title`
  - 网址 `url`
  - 日期 `date`
  - 描述 `description`

调试页面

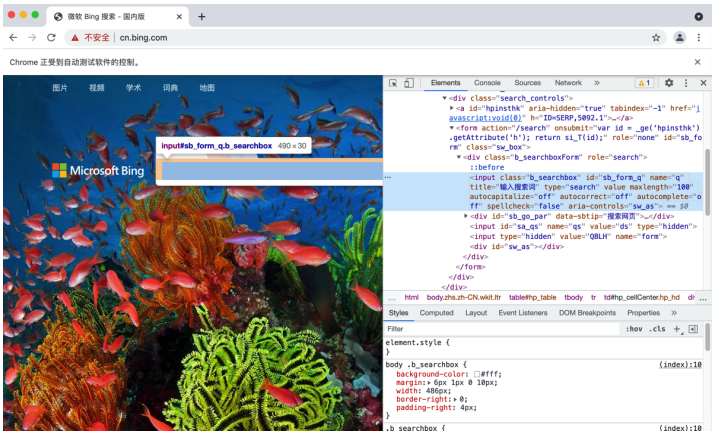
调试期间的相关页面：

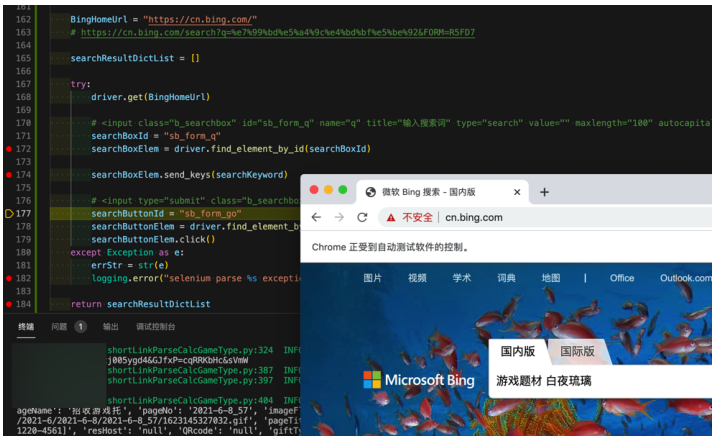
- 打开bing主页
  - 页面
    - 

- 输入框：输入文字
  - html

    ```
    <input class="b_searchbox" id="sb_form_q" name="q"
    ```

  - 页面

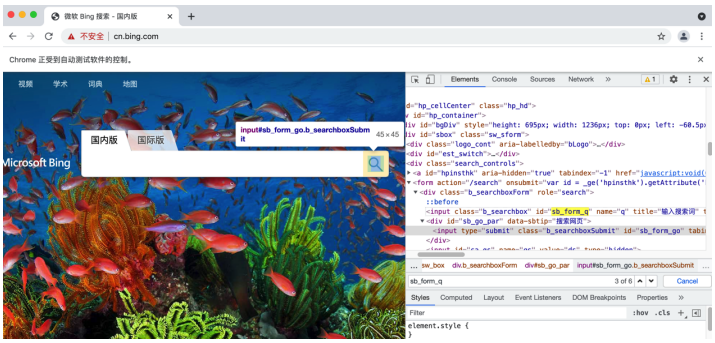




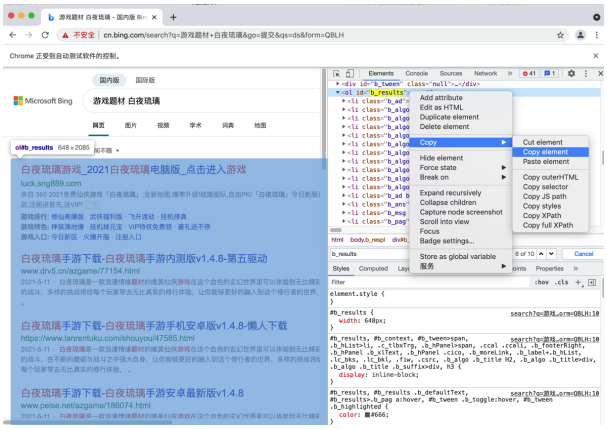
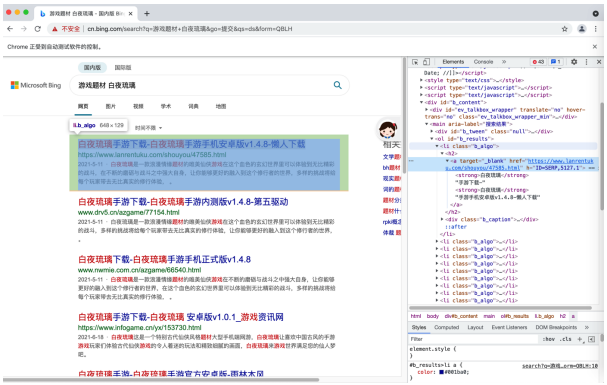
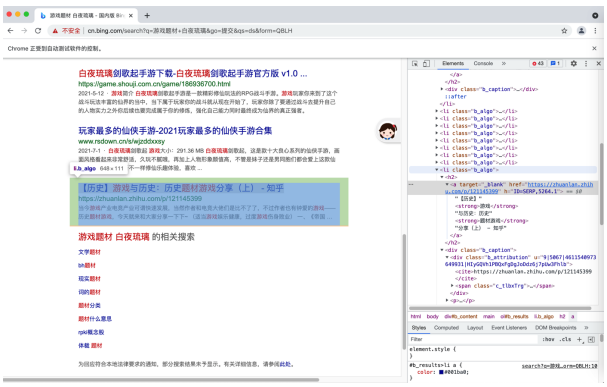
- 点击：搜索按钮
  - html

```
<input type="submit" class="b_searchboxSubmit" id="
```

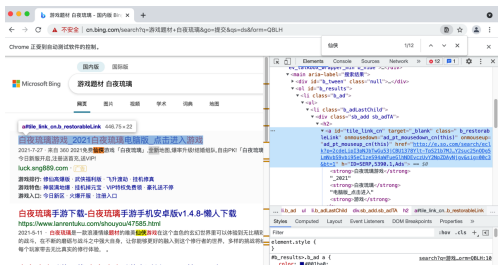
  - 页面



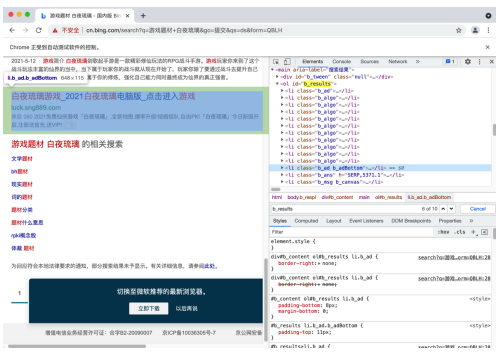
- 搜索结果
  - 中间的搜索结果
    - 列表
      - 页面

- 
- 每条搜索结果
  - 页面
    - 
    - 
  - 普通结果
    - html

```
<li class="b_algo">
    <h2><a target="_blank" href="htt
            h="ID=SERP,5143.1"><stro
    <div class="b_caption">
        <div class="b_attribution" u
            <cite>www.drv5.cn/azgame
                h="BASE:CACHEDPA
            <p>2021-5-11 · 白夜琉璃是一款浪
                。</p>
    </div>
</li>
<li class="b_algo">
    <h2><a target="_blank" href="htt
            h="ID=SERP,5160.1"><stro
    <div class="b_caption">
        <div class="b_attribution" u
            <cite>https://www.lanren
                h="BASE:CACHEDPA
            <p>2021-5-11 · <strong>白夜琉
                。</p>
    </div>
</li>
```

- 之前有：顶部和底部带广告
  - 顶部广告
    - 页面
    - 
    - html

```
<li class="b_ad">
    <ul>
        <li class="b_adLastChild
            <div class="sb_add s
                <h2><a id="tile_
                        onmoused
                        href="ht
                        h="ID=SE
                </h2>
                <div class="b_ca
                    <div class="
                    <p class="">
                                360<
                                clas
                </div>
                <div class="ad_f
                    <p class="b_
                                targ
                                onmo
                                ....
                    <p class="b_
                                targ
                                onmo
                                ...
                                h="I
                                onmo
                                href
                                h="I
                </div>
            </div>
        </li>
    </ul>
</li>
```

- 底部广告
  - 页面
    - 
  - html

```
<li class="b_ad b_adBottom">
    <ul>
        <li class="b_adLastChi
            <div class="sb_add
                <h2><a id="til
                        onmous
                        href="
                        h="ID=
                </h2>
                <div class="b_
                    <div class
                    <p class="
                                36
                                cl
                </div>
            </div>
        </li>
    </ul>
</li>
```

代码

- 注
  - （写代码）之前还有（2个）ad广告结果，此处后续调试没出现。但此处代码已包含相关逻辑。
  - 最新全部代码详见
    - https://github.com/crifan/crifanLibPython/blob/master/python3/crifanLib/thirdParty/crifanSelenium.py

下面贴出对应代码：

模拟打开必应和触发必应搜索

```
def getBingSearchResult(searchKeyword, driver=None):
    """
    Emulate bing search, return search result

    Args:
        searchKeyword (str): str to search
        driver (WebDriver): selenium web driver. Default is
    Returns:
        result dict list
    Raises:
    Examples:
        '游戏题材 白夜琉璃'
    """
    if not driver:
        driver = webdriver.Chrome()

    BingHomeUrl = "https://cn.bing.com/"
    # https://cn.bing.com/search?q=%e7%99%bd%e5%a4%9c%e4%bc

    searchResultDictList = []

    try:
        driver.get(BingHomeUrl)

        # <input class="b_searchbox" id="sb_form_q" name="q
        searchBoxId = "sb_form_q"
        searchBoxElem = driver.find_element_by_id(searchBox
        logging.info("Found search box %s", searchBoxElem)

        searchBoxElem.send_keys(searchKeyword)
        logging.info("Inputed text %s", searchKeyword)

        # <input type="submit" class="b_searchboxSubmit" id
        searchButtonId = "sb_form_go"
        searchButtonElem = driver.find_element_by_id(search
        searchButtonElem.click()
        logging.info("Clicked button %s to trigger seach",

        # wait bing search complete -> show some element
        # <span class="sb_count">9,120,000 条结果</span>
        MaxWaitSeconds = 10
        searchCountElem = WebDriverWait(driver, MaxWaitSeco
            EC.presence_of_element_located((By.XPATH, "//sp
        )
        logging.info("Search complete, showing: %s", search

        searchResultDictList = parseBingSearchResult(drive
        searchResultNum = len(searchResultDictList)
        logging.info("Found %d search result", searchResult
    except Exception as e:
```

```
            errStr = str(e)
            # 'Message: timeout: Timed out receiving message fr
            logging.error("selenium open %s exception: %s", Bir

    return searchResultDictList
```

从必应搜索页面中提取搜索结果

其中包含了，从搜索结果页面中解析出相关信息：标题title、网址链接url、日期date、描述description

```
def parseBingSearchResult(driver, isIncludeAd=True):
    """
    Parse bing search result from current (search result) p

    Args:
        driver (WebDriver): selenium web driver
        isIncludeAd (bool): return result is include ad par
    Returns:
        result dict list
    Raises:
    """
    searchResultDictList = []

    # <ol id="b_results">
    resultId = "b_results"

    # driver.find_element_by_xpath("//ol[@id='b_results']")

    resultElem = driver.find_element_by_id(resultId)
    logging.info("resultElem=%s", resultElem)
    if resultElem:

        # allLiXpath = "//li[@class='b_algo' or @starts-wit
        # allLiXpath = '//li[@class="b_algo" or starts-with
        # allLiXpath = "//li[@class='b_algo' or starts-with
        # resultLiList = resultElem.find_elements_by_xpath(
        # logging.info("resultLiList=%s", resultLiList)
        # resultLiNum = len(resultLiList)
        # logging.info("resultLiNum=%s", resultLiNum)

        normlLiXpath = ".//li[@class='b_algo']"
        normalLiList = resultElem.find_elements_by_xpath(no
        normalLiNum = len(normalLiList)
        logging.info("normalLiNum=%s", normalLiNum)

        # normal result item
        """
            <li class="b_algo">
                <h2><a target="_blank" href="http://www.drv
                        h="ID=SERP,5143.1"><strong>白夜琉璃<
                <div class="b_caption">
                    <div class="b_attribution" u="0|5124|45
                        <cite>www.drv5.cn/azgame/77154.htm
                                h="BASE:CACHEDPAGEDEFAULT"
                    <p>2021-5-11 · 白夜琉璃是一款浪漫情缘<stro
                        。</p>
                </div>
            </li>
            <li class="b_algo">
```

```
                <h2><a target="_blank" href="https://www.la
                        h="ID=SERP,5160.1"><strong>白夜琉璃<
                <div class="b_caption">
                    <div class="b_attribution" u="1|5058|45
                        <cite>https://www.lanrentuku.com/sh
                                h="BASE:CACHEDPAGEDEFAULT"
                        <p>2021-5-11 · <strong>白夜琉璃</strong>
                            。</p>
                    </div>
                </li>
            """
            for curIdx, eachNormalLi in enumerate(normalLiList)
                logging.info("%s [%d] %s", "-"*10, curIdx + 1,

                curTitle = ""
                curUrl = ""
                curDate = ""
                curDesc = ""

                try:
                    # h2AItem = eachNormalLi.find_element_by_xp
                    h2AItem = eachNormalLi.find_element_by_xpat
                except:
                    logging.error("Failed to find title(h2 a) e
                    continue

                curTitle = h2AItem.text
                curUrl = h2AItem.get_attribute("href")
                logging.info("curTitle=%s, curUrl=%s", curTitle

                try:
                    descItem = eachNormalLi.find_element_by_xpa

                    curDesc = descItem.text
                    logging.info("curDesc=%s", curDesc) # 2021-
                except:
                    logging.warning("Failed to find description
                    # continue

                if not curDesc:
                    # try to find b_richcard 's tab-content
                    # Note: here for b_richcard only parse firs
                    """
                        <div class="b_caption">
                            <div class="b_attribution" u="8|500
                                <cite>https://zhuanlan.zhihu.co
                                        h="BASE:CACHEDPAGEDEFAU
                            <div class="b_richcard">
                                <div class="rc_herotabheader">
                                ...
                                        <div class="tab-content
```

```
                                        <div id="tab_2" dat
                                            data-priority='
                                            <ul class="b_vl
                                                <li data-pr
                                                    <div><s
                                                </li>
            """
            try:
                firstTabItem = eachNormalLi.find_elemen
                curDesc = firstTabItem.text
                logging.info("curDesc=%s", curDesc) #
            except:
                logging.warning("Failed to find descrip
                # continue

        if not curDesc:
            logging.error("Failed to find description")

        foundDate = re.search("^(?P<curDate>\d+-\d+-\d+
        if foundDate:
            curDate = foundDate.group("curDate") # '202
            pureDesc = foundDate.group("pureDesc") # '|
            logging.info("curDate=%s, pureDesc=%s", cur
            curDesc = pureDesc

        curSearchResultDict = {
            "url": curUrl,
            "title": curTitle,
            "date": curDate,
            "description": curDesc,
        }
        logging.info("curSearchResultDict=%s", curSearc
        searchResultDictList.append(curSearchResultDict

    # first ad item
    """
        <li class="b_ad">
            <ul>
                <li class="b_adLastChild">
                    <div class="sb_add sb_adTA">
                        <h2><a id="tile_link_cn" target
                                onmousedown="ad_pt_mous
                                href="http://e.so.com/s
                                h="ID=SERP,5397.1,Ads">
                        </h2>
                        <div class="b_caption">
```

```
                                <div class="b_attribution">
                                <p class=""><span class="b_
                                        360</span>2021免费仙
                                        class="b_adSlug b_c
                            </div>
                            <div class="ad_fls">
        ...
                            </div>
                        </div>
                    </li>
                </ul>
            </li>
        """

        # bottom ad item
        """
            <li class="b_ad b_adBottom">
                <ul>
                    <li class="b_adLastChild">
                        <div class="sb_add sb_adTA">
                            <h2><a id="tile_link_cn" target
                                    onmousedown="ad_pt_mous
                                    href="http://e.so.com/s
                                    h="ID=SERP,5412.1,Ads">
                            </h2>
                            <div class="b_caption">
                                <div class="b_attribution">
                                <p class=""><span class="b_
                                        360</span>2021免费仙
                                        class="b_adSlug b_c
                            </div>
                        </div>
                    </li>
                </ul>
            </li>
        """
        if isIncludeAd:
            adSearchResultList = []

            adLiXpath = ".//li[starts-with(@class, 'b_ad')]
            adLiList = resultElem.find_elements_by_xpath(ad
            adLiNum = len(adLiList)
            logging.info("adLiNum=%s", adLiNum)

            for eachAdElem in adLiList:
                curAdTitle = ""
                curAdUrl = ""
                curAdDate = ""
                curAdDesc = ""

                try:
```

```
                    titleElem = eachAdElem.find_element_by_
                    curAdTitle = titleElem.text
                    curAdUrl = titleElem.get_attribute("hre
                except:
                    logging.warning("Failed to find ad tit
                    continue

                try:
                    descElem = eachAdElem.find_element_by_x
                    curAdDesc = descElem.text
                except:
                    logging.warning("Failed to find ad tit

                curAdSearchResultDict = {
                    "url": curAdUrl,
                    "title": curAdTitle,
                    "date": curAdDate,
                    "description": curAdDesc,
                }
                logging.info("curAdSearchResultDict=%s", cu
                adSearchResultList.append(curAdSearchResult

            searchResultDictList.extend(adSearchResultList)

    return searchResultDictList
```

输出结果

（某次）调试的输出结果：

```
normalLiNum=10
---------- [1] ----------
curTitle=白夜琉璃手游下载-白夜琉璃手游手机安卓版v1.4.8-懒人下载, c
curDesc=2021-5-11 · 白夜琉璃是一款浪漫情缘题材的唯美仙侠游戏在这个
curDate=2021-5-11, pureDesc=白夜琉璃是一款浪漫情缘题材的唯美仙侠
curSearchResultDict={'url': 'https://www.lanrentuku.com/sh
---------- [2] ----------
curTitle=白夜琉璃手游下载-白夜琉璃手游内测版v1.4.8-第五驱动, curU
curDesc=2021-5-11 · 白夜琉璃是一款浪漫情缘题材的唯美仙侠游戏在这个
curDate=2021-5-11, pureDesc=白夜琉璃是一款浪漫情缘题材的唯美仙侠
curSearchResultDict={'url': 'http://www.drv5.cn/azgame/771
---------- [3] ----------
curTitle=白夜琉璃下载-白夜琉璃手游手机正式版v1.4.8, curUrl=http:
curDesc=2021-5-11 · 白夜琉璃是一款浪漫情缘题材的唯美仙侠游戏在不断
curDate=2021-5-11, pureDesc=白夜琉璃是一款浪漫情缘题材的唯美仙侠
curSearchResultDict={'url': 'http://www.nwmie.com.cn/azgam
---------- [4] ----------
curTitle=白夜琉璃手游下载-白夜琉璃 安卓版v1.0.1_游戏资讯网, curUr
curDesc=2021-6-18 · 白夜琉璃这是一个特别古代仙侠风格题材大型手机端
curDate=2021-6-18, pureDesc=白夜琉璃这是一个特别古代仙侠风格题材
curSearchResultDict={'url': 'https://www.infogame.cn/yx/15
---------- [5] ----------
curTitle=白夜琉璃手游-白夜琉璃手游官方安卓版-雨林木风, curUrl=http
curDesc=2021-5-25 · 白夜琉璃是一款能够带给玩家超爽仙灵修道冒险的RP
curDate=2021-5-25, pureDesc=白夜琉璃是一款能够带给玩家超爽仙灵修
curSearchResultDict={'url': 'https://www.ylmfu.com/zhuti/4
---------- [6] ----------
curTitle=白夜琉璃游戏下载-白夜琉璃完整版免费下载v1.0-七度网, curUr
curDesc=2021-5-11 · 白夜琉璃是一款经典趣味的仙侠手游，游戏中的玩法
curDate=2021-5-11, pureDesc=白夜琉璃是一款经典趣味的仙侠手游，游
curSearchResultDict={'url': 'https://m.7do.net/game/108503
---------- [7] ----------
curTitle=白夜琉璃修真诀手游下载-白夜琉璃修真诀安卓版-旭宁下载站, cu
curDesc=2021-7-26 · 白夜琉璃修真诀手游是一款带来有趣体验的战斗主题
curDate=2021-7-26, pureDesc=白夜琉璃修真诀手游是一款带来有趣体验
curSearchResultDict={'url': 'http://www.xuningbo.com/game/
---------- [8] ----------
curTitle=白夜琉璃剑歌起手游下载-白夜琉璃剑歌起手游官方版 v1.0 ...,
curDesc=2021-5-12 · 游戏简介 白夜琉璃剑歌起手游是一款精彩修仙玩法
curDate=2021-5-12, pureDesc=游戏简介 白夜琉璃剑歌起手游是一款精
curSearchResultDict={'url': 'https://game.shouji.com.cn/ga
---------- [9] ----------
curTitle=玩家最多的仙侠手游-2021玩家最多的仙侠手游合集, curUrl=ht
curDesc=2021-7-1 · 白夜琉璃剑歌起 游戏大小：291.36 MB 白夜琉璃剑
curDate=2021-7-1, pureDesc=白夜琉璃剑歌起 游戏大小：291.36 MB
curSearchResultDict={'url': 'http://www.rsdown.cn/s/wjzddx
---------- [10] ----------
curTitle=【历史】游戏与历史：历史题材游戏分享（上） - 知乎, curUrl
curDesc=当今游戏产业电竞产业可谓快速发展，当然作者和电竞大佬们是比不
curSearchResultDict={'url': 'https://zhuanlan.zhihu.com/p/
```

插件

Selenium应用广泛，有人基于Selenium开发出一些插件，扩展实现了特定的功能。

selenium-wire

- `selenium-wire`
  - 是什么：Selenium的一个插件
  - 功能：可以捕获和访问到底层的浏览器发出的请求和响应
  - 主页
    - GitHub
      - https://github.com/wkeeling/selenium-wire

官网示例

```
from seleniumwire import webdriver  # Import from seleniumw

# Create a new instance of the Chrome driver
driver = webdriver.Chrome()

# Go to the Google home page
driver.get('https://www.google.com')

# Access requests via the `requests` attribute
for request in driver.requests:
    if request.response:
        print(
            request.url,
            request.response.status_code,
            request.response.headers['Content-Type']
        )
```

输出：

```
https://www.google.com/ 200 text/html; charset=UTF-8
https://www.google.com/images/branding/googlelogo/2x/google
https://consent.google.com/status?continue=https://www.goog
https://www.google.com/images/branding/googlelogo/2x/google
https://ssl.gstatic.com/gb/images/i2_2ec824b0.png 200 image
https://www.google.com/gen_204?s=webaft&t=aft&atyp=csi&ei=
...
```

基本用法

- 安装

```
pipenv install selenium-wire
```

- 更新代码

把之前的：

```
from selenium import webdriver
```

改为：

```
from seleniumwire import webdriver
```

其他（一般的Selenium的）代码，基本不用变。

- 具体使用：捕获http请求

在用：

```
driver.get(inputUrl)
```

加载页面后，用：

```
# capture http request and response
# Access requests via the `requests` attribute
requestNum = len(driver.requests)
logging.info("Captured %d requests", requestNum)
for curIdx, curReq in enumerate(driver.requests):
    curNum = curIdx + 1
    curUrl = curReq.url
    if curReq.response:
        curResp = curReq.response
        logging.info("[%d] resp: url:%s, code=%s, Content
            curNum, curUrl, curResp.status_code, curResp.
```

即可获取到每一个request（和对应response）。

之后，即可根据自己业务逻辑，去利用（每一个）request（和response）了。

- 说明
  - 初始化期间会打印出log

    ```
    backend.py:51   INFO    Created proxy listening on
    ```

  - 而 `driver.get` 后，会输出相关捕获到的请求和响应

```
20210722 03:33:05 handler.py:57   INFO    Capturing
20210722 03:33:05 handler.py:116  INFO    Capturing
...
20210722 03:33:05 handler.py:57   INFO    Capturing
...
```

其他

禁止捕获请求

可以额外增加参数，（临时）禁止捕获http请求：

```
options = {
    'disable_capture': True  # Don't intercept/store any re
}
driver = webdriver.Chrome(seleniumwire_options=options)
```

附录

下面列出相关参考资料。

Selenium文档

官网文档

- 官网
  - 英文
    - 3 Navigating — Selenium Python Bindings 2 documentation
      - https://selenium-python.readthedocs.io/navigating.html
    - 4 Locating Elements — Selenium Python Bindings 2 documentation
      - https://selenium-python.readthedocs.io/locating-elements.html
    - 5 Waits — Selenium Python Bindings 2 documentation
      - https://selenium-python.readthedocs.io/waits.html
    - 6 Page Objects — Selenium Python Bindings 2 documentation
      - https://selenium-python.readthedocs.io/page-objects.html
    - 7 WebDriver API — Selenium Python Bindings 2 documentation
      - https://selenium-python.readthedocs.io/api.html
  - 中文
    - 4 查找元素 — Selenium-Python中文文档 2 documentation
      - https://selenium-python-zh.readthedocs.io/en/latest/locating-elements.html
    - 5 等待页面加载完成(Waits) — Selenium-Python中文文档 2 documentation
      - https://selenium-python-zh.readthedocs.io/en/latest/waits.html
    - 6 页面对象 — Selenium-Python中文文档 2 documentation
      - https://selenium-python-zh.readthedocs.io/en/latest/page-objects.html
    - 7 WebDriver API — Selenium-Python中文文档 2 documentation
      - https://selenium-python-zh.readthedocs.io/en/latest/api.html

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。

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中Python解析百度搜索结果第一页获取标题列表
- 【已解决】Selenium中如何实现百度搜索结果标题元素的定位
- 【已解决】Mac中用Selenium自动操作浏览器实现百度搜索
- 【已解决】Selenium调试时能搜到元素但是直接运行找不到
- 【已解决】Selenium中如何获取到当前页面的html源码
- 【已解决】用Selenium模拟浏览器去bing必应搜索并返回结果
- 【部分解决】通过参数禁止Selenium-wire捕获请求
- 【已解决】Selenium从bing搜索结果解析出错：特殊的richcard多tab
- 【已解决】Selenium中用xpath查询元素节点找到结果不对
- 【已解决】Selenium如何实现2种class属性值的or或去查找元素
- 【已解决】Selenium获取页面内部调用的http的api请求和返回的json数据
- 【已解决】Selenium的Chrome模拟bing必应搜索结果页面提取搜索结果
-
- 【整理】用Chrome或Chromium查看百度首页中各元素的html源码 – 在路上
- 【已解决】Selenium中如何定位元素：百度首页中的输入框 – 在路上
- 【已解决】Selenium中给百度搜索框中输入文字并触发搜索 – 在路上
- 【整理】用Chrome或Chromium查看百度首页中各元素的html源码
- 【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
-