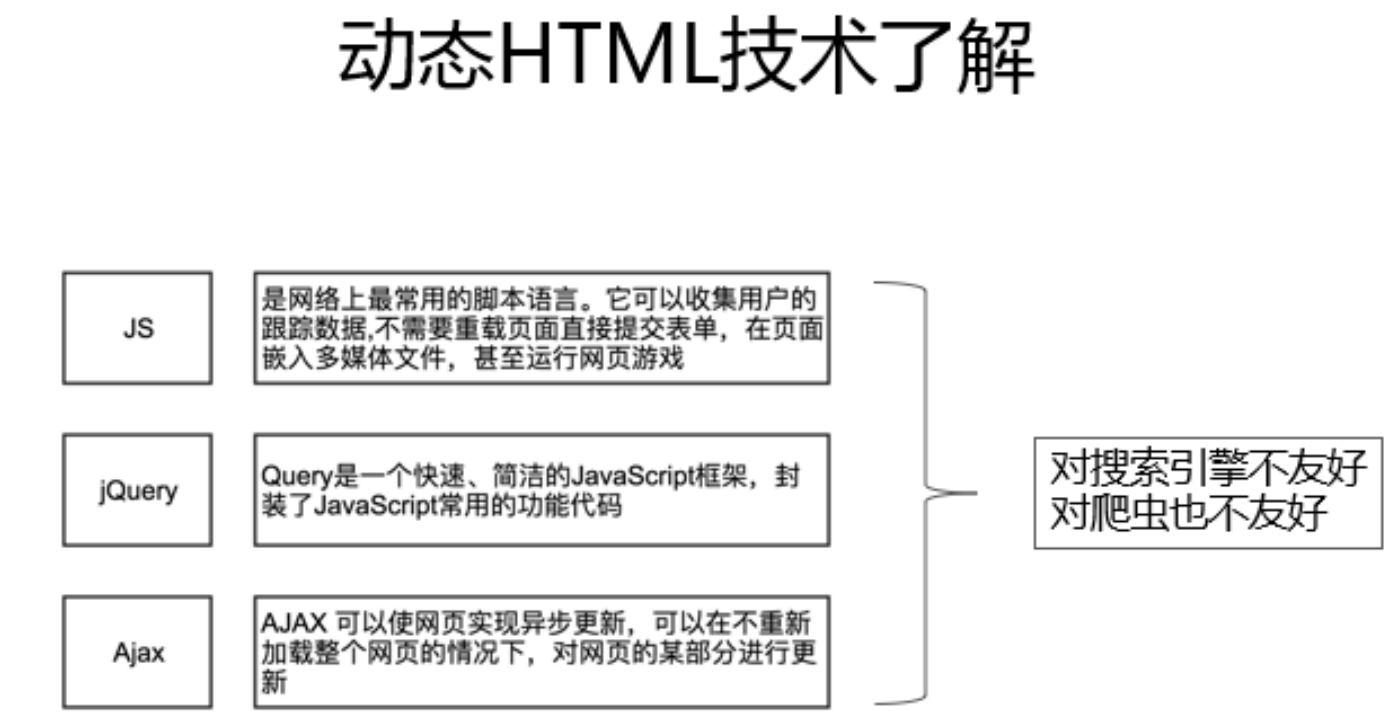

动态页面爬虫



1-Selenium入门

加载页面：

```
from selenium import webdriver
#如果使用windows()，中间传入drvier的路径:"c:/.../pantomjs.exe"
driver = webdriver.Chrome()
# 获取
driver = get("http://www.baidu.com/")
# 截图操作
driver.save_screenshot("baidu.png")

```

定位和操作：

```
# 定位id为kw的标签，输入关键字python
driver.find_element_by_id("kw").send_keys("python")
# 定位id为su的标签，点击，后面会发生跳转
driver.find_element_by_id("su").click()

```

cookie操作

```
# 获取cookie
driver.get_cookies()
# 添加cookie
drive.add_cookie({})
# 删除cookie
driver.delete_cookie("CookieName")
```

查看请求信息：

```
# 查看页面所有内容，是所有的element，包括js,css,ajax请求出来的element
driver.page_source()
# 获取当前的url地址
driver.current_url
# 获取当前页面的cookies值
driver.get_cookies()
```

退出

```
# 退出当前页面
driver.close()
# 退出浏览器
driver.quit()
```

2-selenium更多方法

selenium定位页面元素

- 用法:

```
find_element_by_id                      (根据id返回一个元素，如果没有就报错了)
find_elements_by_xpath                  （根据xpath返回一个列表，如果没有就返回一个空列表）
find_elements_by_link_text              (根据文本内容返回元素列表，如果没有返回空列表)
find_elements_by_partial_link_text          (根据文件部分内容返回元素列表，如果没有返回空列表)
find_elements_by_tag_name           (根据标签名[节点名]返回元素元素列表，如果没有返回空列表)
find_elements_by_class_name         (根据class属性对应值范围元素列表，如果没有返回空列表)
find_elements_by_css_selector       (根据css选择器返回元素列表，如果没有返回空列表)
```

获取元素的数据与操作

获取文本和获取属性

- 先定位到元素，然后调用.text或者get_attribute方法

```
get_attribute(key) :    获取节点上的属性
text属性:               获取该节点上以及子节点的文本
click():               点击控件
```

注意点:

- find_element 和find_elements的区别:

```
find_element:       返回找到第一个元素,如果没有报错
find_elements:          返回找到的所有元素列表，如果没有找到返回空列表
```

- by_link_text和by_partial_link_text的区别:

```
by_link_text:       全部文本都一样
by_partial_link_text: 包含某个文本
```

- **by_xpath只能获取元素, 要获取属性和文本需要使用get_attribute(属性名) 和.text**
- **如果页面中含有iframe、frame，需要先调用driver.switch_to.frame的方法切换到frame中才能定位元素**
- selenium请求第一页的时候会等待页面全部加载完了之后再获取数据，但是在点击翻页之后并不会等待，直接获取可能报错，需要等待time.sleep(3)

3-打开新的标签（了解）

```
打开空白标签页的方式有很多， 在此只演示一种
js='window.open("https://www.jianshu.com/p/4fef4142b33f");'
driver.execute_script(js) # 通过打开新的标签页， 可以节省浏览器打开的时间，减少资源的浪费。
```

4-关闭新打开的标签 driver.window_handles

```
# 1.获取当前标签页句柄
driver.current_window_handle
# 2.获取标签页的句柄
handlesList = driver.window_handles  # 返回一个浏览器中所有标签的句柄列表， 顺序为打开窗口的顺序
# 3. 切换窗口， 关闭标签
driver.switch_to.window(handlesList[0]) # 切换到百度标签(句柄对象)
driver.close()    # 关闭标签，这里必须用 driver.close() ，用driver.quit()会导致浏览器关闭
```

5-切换

- 定位到当前聚集元素上

```
driver.switch_to.active_element()
```

- 切换到alert弹窗

```
driver.switch_to.alert()
```

- 切换到最上层页面

```
driver.switch_to.default_content()
```

- 通过id、name、element(定位的某个元素)、索引来切换到某个frame

```
# 通过frame的id,name,driver.find_element_by_xpath("//a[1]")
driver.switch_to.frame(frame_reference)

```

- 切换到指定标签页

```
driver.switch_to.window()
```

- 切换到上一层的frame，对于层层嵌套的frame很有用

```
driver.switch_to.parent_frame()
```