AirParts航件爬虫总结

一、目标

爬取 "http://www.airparts.cn/Hk\_Function/Hk\_Buy.aspx?lm=17"网站上列出的航件信息，共有109页，每个条目对应一个航件，航件对应不同链接（例如：http://www.airparts.cn/Details/Hk\_Buy\_Details.aspx?Sell\_Id=2291&page\_id=2&lm=17），爬取全部连接的信息。包括：类别、标题、件号、分类、名称、数量、状态、飞机型号、时间。将爬取的信息保存在Excel表里。





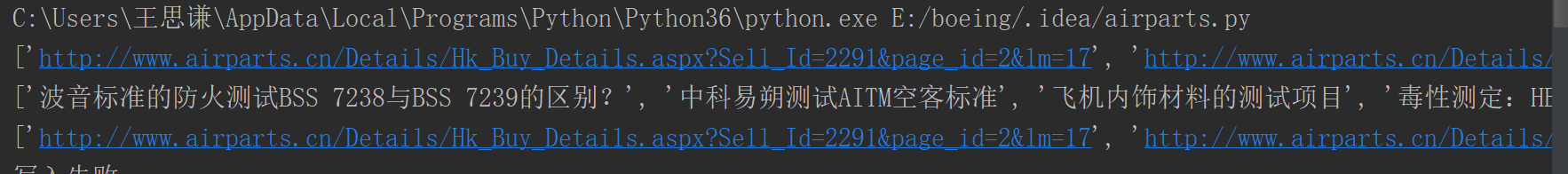
1. 实现

1.在 aripartspage函数中实现将所有航件条目保存在列表中。用selenium 自动化测试工具，定位表项driver.find\_elements\_by\_class\_name('product\_list')这是通过class name 定位到具体元素项目的，还有很多方法，例如driver.find\_elements\_by\_id,driver.find\_elements\_by\_Xpath ...但是，最一开始尝试的时候，使用Xpath定位，感觉不是很好用，拿不到数据，因此，还是通过class name 定位比较准确。

<a href="../Details/Hk\_Buy\_Details.aspx?Sell\_Id=2291&amp;page\_id=2&amp;lm=17" class="product\_list" target="\_blank" title="波音标准的防火测试BSS 7238与BSS 7239的区别？"><span id="ctl00\_ContentPlaceHolder1\_listBuy\_ctl00\_lblTile">波音标准的防火测试BSS 7238与BSS 7239的区别？</span>

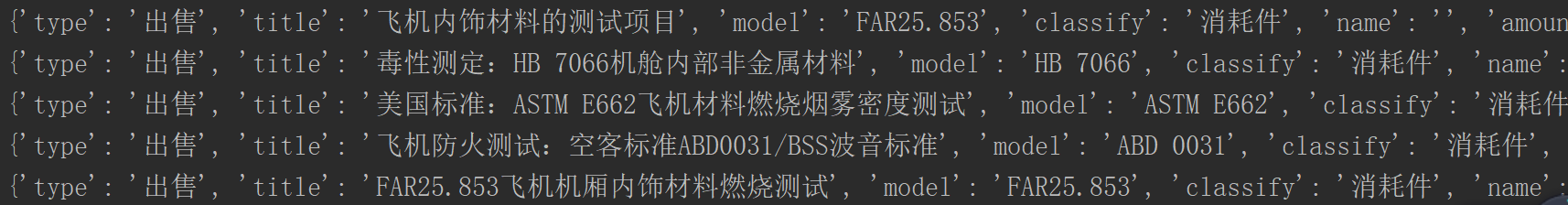
</a>

**def aripartspage**():  
 alist=[]#存放链接  
 blist=[]#存放对应连接的名字  
 airparts=[]  
 #获得页面页数  
 **for** i **in** range(1,110):  
 baseurl="http://www.airparts.cn/Hk\_Function/Hk\_Buy.aspx?lm=17"  
 pageurl="&pageid="+str(i)  
 url=baseurl+pageurl  
 driver.get(url)  
 **for** link **in** driver.find\_elements\_by\_class\_name('product\_list'):  
 href = link.get\_attribute('href')  
 title=link.get\_attribute('title')  
 **if** href **not in** alist:  
 alist.append(href)  
 **if** title **not in** blist:  
 blist.append(title)  
 print(alist)  
 print(blist)  
 artpartclassification(alist)



1. 得到所有条目的“url”,”title”的列表alist后，要调用函数artpartclassification爬取对应链接的一些内容，并将其保存在dict字典中。定位方法是用selenium的driver.find\_elements\_by\_id,准确定位元素。

**def artpartclassification**(alist):  
 airparts=[]  
 print(alist)  
 **for** airparturl **in** alist:  
 **try**:  
 driver.get(airparturl)  
 temp=dict( type=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblType").text,  
 title=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblTitle").text,  
 model=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblModel").text,  
 classify=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblClass").text,  
 name=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblName").text,  
 amount=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblAmount").text,  
 state=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblState").text,  
 aircraftype=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblAircraftType").text,  
 time=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblTime").text,  
 )  
 airparts.append(temp)  
 time.sleep(3)  
 **except**:  
 print("写入失败")  
 **for** item **in** airparts:  
 print(item)  
 export\_excel(airparts)



1. 将爬取到的内容保存在Excel 表中。

首先import 头文件

**import** pandas **as** pd  
**from** openpyxl **import** Workbook  
Lable = ['type','title','model','classify','name','amount','state','aircraftype','time']

利用 pandas包处理数据，这个pandas包对与机器学习很有用，要努力的学习一下。将dict 类型的数据保存在Excel表中是很常见的工作，会应用就行。

**def export\_excel**(dic\_data):  
 # 将字典列表转换为DataFrame  
 pf=pd.DataFrame(list(dic\_data))  
 # 指定字段顺序  
 order=['type', 'title', 'model','classify', 'name','amount','state','aircraftype','time']  
 pf=pf[order]  
 # 将列名替换为中文  
 columns\_map={  
 'type': '类别',  
 'title': '标题',  
 'model': '件号',  
 'classify': '分类',  
 'name': '名称',  
 'amount': '数量',  
 'state':'状态',  
 'aircraftype':'飞机型号',  
 'time':'时间',  
 }  
 pf.rename(columns=columns\_map, inplace=**True**)  
 # 指定生成的Excel表格名称  
 file\_path=pd.ExcelWriter('airparts.xlsx')  
 # file\_csv\_path = pd.read\_csv("compound.csv")  
 # 替换空单元格  
 pf.fillna(' ', inplace=**True**)  
 # 输出  
 pf.to\_excel(file\_path, encoding='utf-8', index=**False**)  
 # pf.to\_csv(file\_csv\_path, encoding='utf-8', index=False)  
 # 保存表格  
 file\_path.save()

1. 源代码

**from** selenium **import** webdriver  
**from** selenium.webdriver.support.ui **import** WebDriverWait  
**import** requests  
**from** bs4 **import** BeautifulSoup  
**import** time  
**import** re  
**import** pandas **as** pd  
**from** openpyxl **import** Workbook  
Lable = ['type','title','model','classify','name','amount','state','aircraftype','time']  
#driver=webdriver.PhantomJS()  
driver=webdriver.Chrome()  
  
**def aripartspage**():  
 alist=[]#存放链接  
 blist=[]#存放对应连接的名字  
 airparts=[]  
 #获得页面页数  
 **for** i **in** range(1,110):  
 baseurl="http://www.airparts.cn/Hk\_Function/Hk\_Buy.aspx?lm=17"  
 pageurl="&pageid="+str(i)  
 url=baseurl+pageurl  
 driver.get(url)  
 **for** link **in** driver.find\_elements\_by\_class\_name('product\_list'):  
 href = link.get\_attribute('href')  
 title=link.get\_attribute('title')  
 **if** href **not in** alist:  
 alist.append(href)  
 **if** title **not in** blist:  
 blist.append(title)  
 print(alist)  
 print(blist)  
 artpartclassification(alist)  
#这个是测试用的，对本实验代码没有作用。  
**def test**():  
 airparturl='http://www.airparts.cn/Details/Hk\_Buy\_Details.aspx?Sell\_Id=2276&page\_id=2&lm=17'  
 driver=webdriver.Chrome()  
 driver.get(airparturl)  
 airparts=[]  
 temp=dict( type=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblType").text,  
 title=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblTitle").text,  
 model=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblModel").text,  
 classify=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblClass").text,  
 name=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblName").text,  
 amount=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblAmount").text,  
 state=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblState").text,  
 aircraftype=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblAircraftType").text,  
 time=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblTime").text,  
 )  
 airparts.append(temp)  
 print(airparts)  
  
  
  
**def artpartclassification**(alist):  
 airparts=[]  
 print(alist)  
 **for** airparturl **in** alist:  
 **try**:  
 driver.get(airparturl)  
 temp=dict( type=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblType").text,  
 title=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblTitle").text,  
 model=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblModel").text,  
 classify=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblClass").text,  
 name=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblName").text,  
 amount=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblAmount").text,  
 state=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblState").text,  
 aircraftype=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblAircraftType").text,  
 time=driver.find\_element\_by\_id("ctl00\_ContentPlaceHolder1\_lblTime").text,  
 )  
 airparts.append(temp)  
 time.sleep(3)  
 **except**:  
 print("写入失败")  
 **for** item **in** airparts:  
 print(item)  
 export\_excel(airparts)  
 #writeDataToExcleFile(airparts,'data.xlsx')  
 #write\_airpart(airparts)  
  
**def export\_excel**(dic\_data):  
 # 将字典列表转换为DataFrame  
 pf=pd.DataFrame(list(dic\_data))  
 # 指定字段顺序  
 order=['type', 'title', 'model','classify', 'name','amount','state','aircraftype','time']  
 pf=pf[order]  
 # 将列名替换为中文  
 columns\_map={  
 'type': '类别',  
 'title': '标题',  
 'model': '件号',  
 'classify': '分类',  
 'name': '名称',  
 'amount': '数量',  
 'state':'状态',  
 'aircraftype':'飞机型号',  
 'time':'时间',  
 }  
 pf.rename(columns=columns\_map, inplace=**True**)  
 # 指定生成的Excel表格名称  
 file\_path=pd.ExcelWriter('airparts.xlsx')  
 # file\_csv\_path = pd.read\_csv("compound.csv")  
 # 替换空单元格  
 pf.fillna(' ', inplace=**True**)  
 # 输出  
 pf.to\_excel(file\_path, encoding='utf-8', index=**False**)  
 # pf.to\_csv(file\_csv\_path, encoding='utf-8', index=False)  
 # 保存表格  
 file\_path.save()  
  
  
**if** \_\_name\_\_=='\_\_main\_\_':  
 aripartspage()

1. 实验结果

见airparts.xlsx,共1087项。

