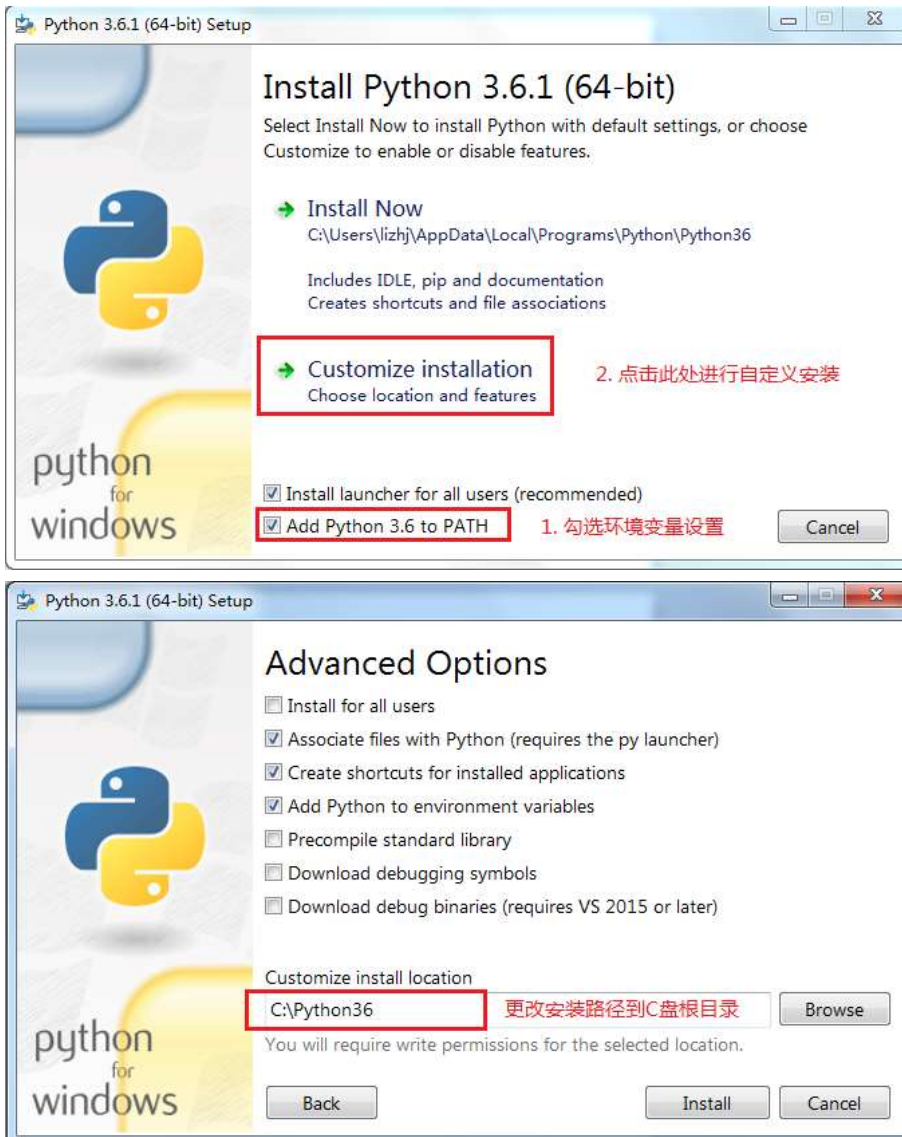


python + selenium自动化测试框架使用说明

1.开发环境安装说明

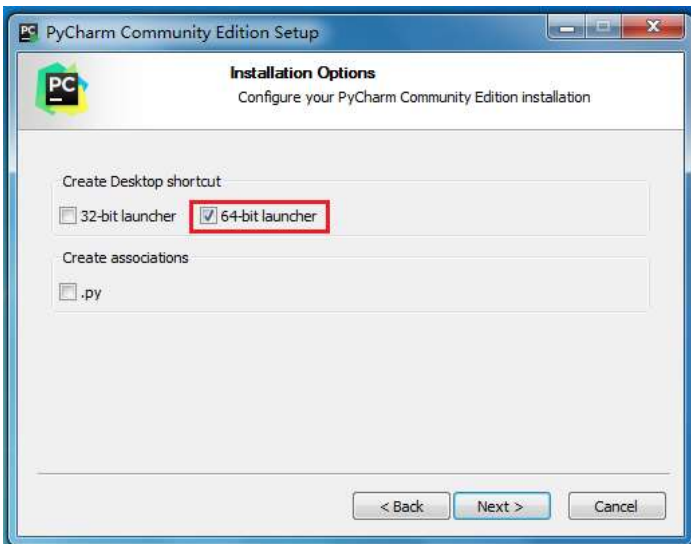
1. 安装python3.6.1
运行python-3.6.1-amd64.exe，完成安装；安装过程中有两处需要设置：



2. 安装selenium组件
双击执行requests_install.bat

3. 安装git客户端
执行Git-2.10.2-64-bit.exe

4. 安装pyCharm
执行pycharm-community-171.3780.47.exe，安装过程中有一处需要勾选



5. putty安全密钥生成工具

执行putty-64bit-0.68-installer.msi，安装完成后打开它，点击按钮Generate生成密钥如下图所示，保存在本地



6. 安装浏览器 (框架目前支持Chrome, Firefox, IE)，下载webdriver (源代码包里已是最新的驱动了)

chromedriver下载地址: <https://code.google.com/p/chromedriver/downloads/list>

Firefox的驱动geckodriver下载地址: <https://github.com/mozilla/geckodriver/releases/>

IE的驱动IEDriver下载地址: <http://www.nuget.org/packages/Selenium.WebDriver.IEDriver/>

注意: 浏览器驱动与客户机安装的浏览器版本版本要对应; 而且浏览器安装时, 建议使用官方版本安装, 避免出现一些莫名其妙的问題 (踩过坑的人友情提示)

###2. 自动化项目工程代码同步

- 以Gitlab管理工具为例说明
- 1.打开gitlab地址, 登录:



GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

Sign in

Username or email

Password

☐ Remember me

[Forgot your password?](#)

Sign in

2.管理员将用户加入项目成员, 成员登录后, 找到测试项目, 进行查看项目详情:

testing > topideal_selenium > Details

T

topideal_selenium

Star0

Fork0

SSH

Global

Files (14.6 MB)

Commits (59)

Branches (2)

Tags (0)

Add Changelog

Add License

Add Contribution guide

Set up CI/CD

Auto DevOps (Beta)

It will automatically build, test, and deploy your application based on a predefined CI/CD configuration.

Learn more in the [Auto DevOps documentation](#)

Enable in settings

master

topideal_selenium /

History

Find file

helper中的增加读取excel数据的方法

linjt authored about 8 hours ago

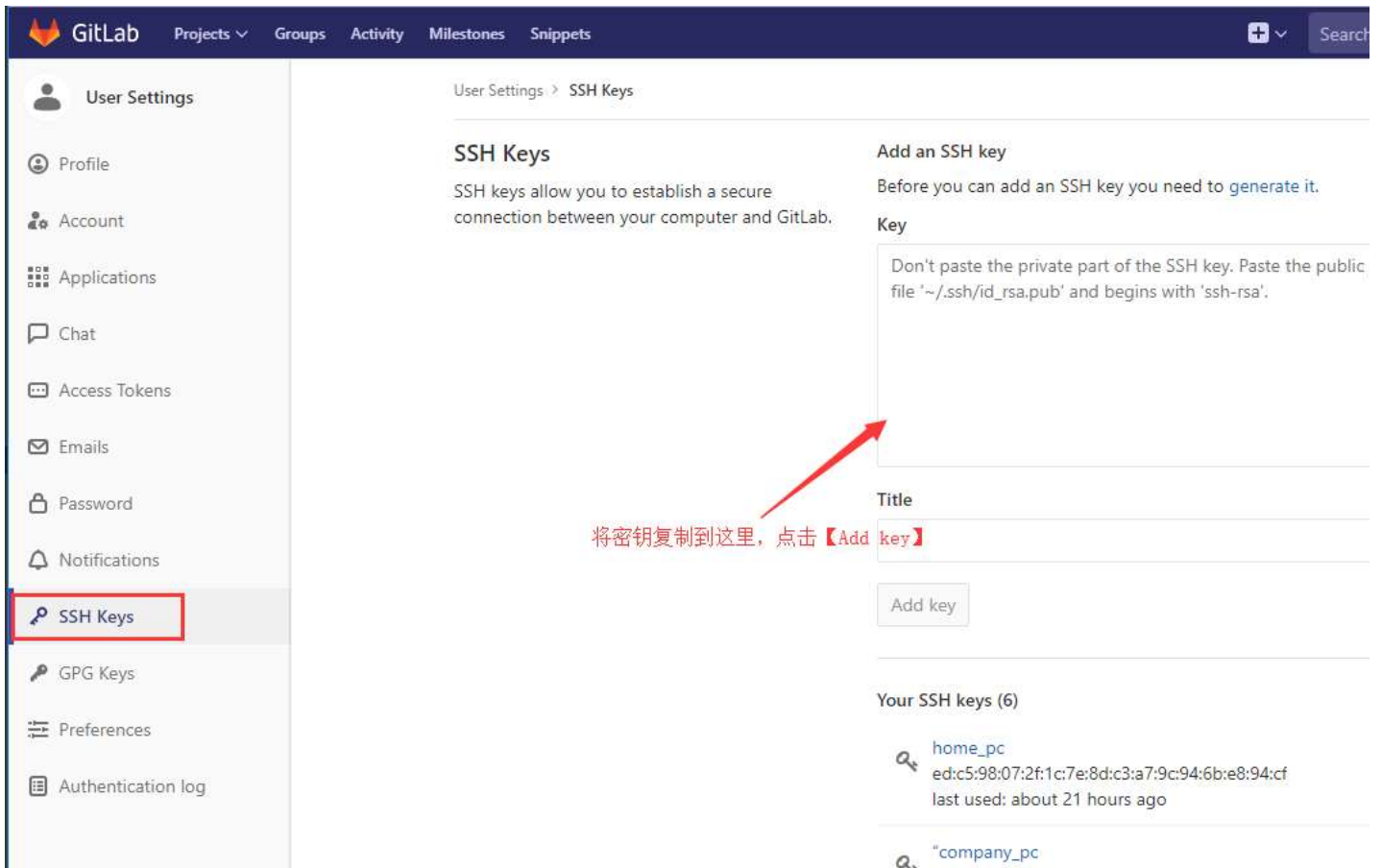
d77705e8

3.添加属于自己的SSH public KEY

密钥在前面开发环境安装步骤中第5步生成, 拷贝到公共密钥页面:

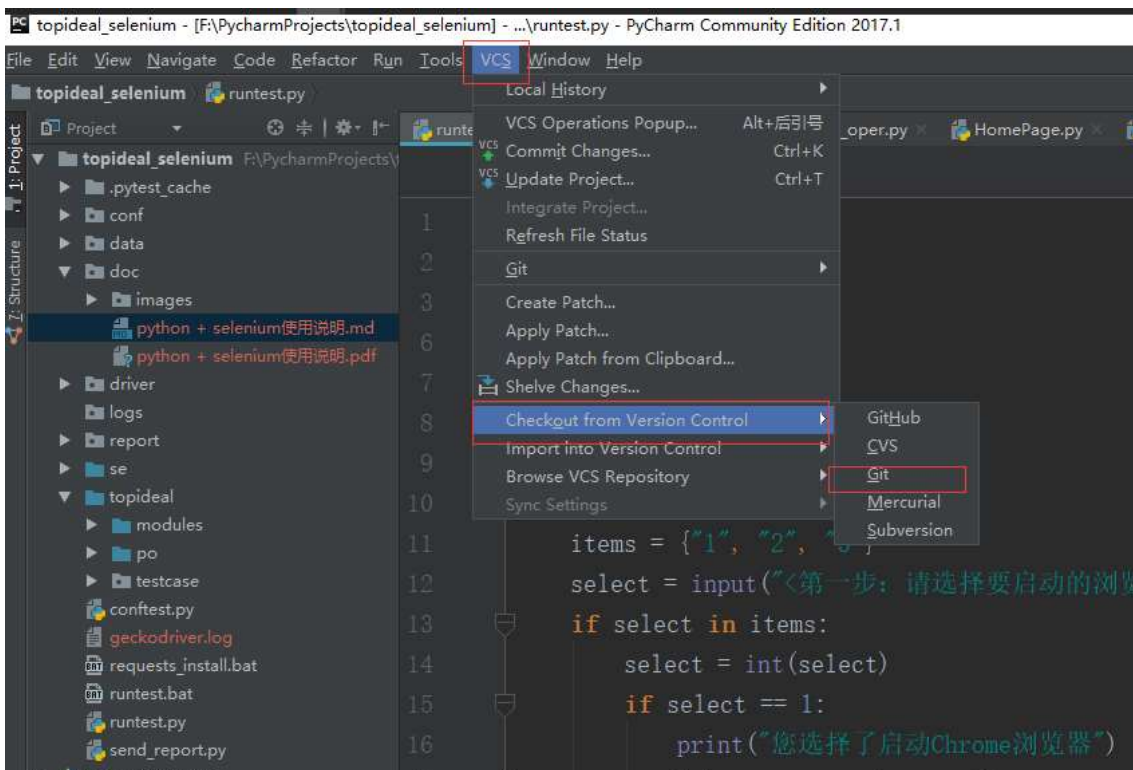
file:///F:/PycharmProjects/topideal_selenium/doc/python%20+%20selenium%E4%BD%BF%E7%94%A8%E8%AF%B4%E6%98%8E.md

3/10

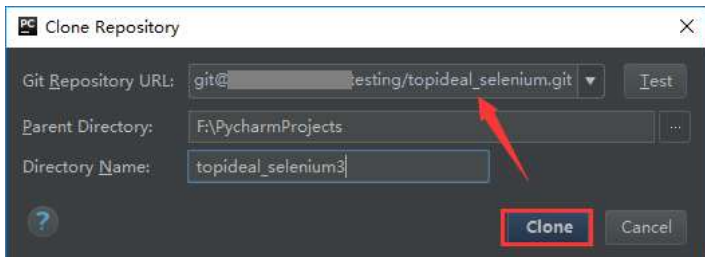


4. 打开pycharm，导出工程项目到本地：

入口：VCS -> Checkout from Version Control --> Git

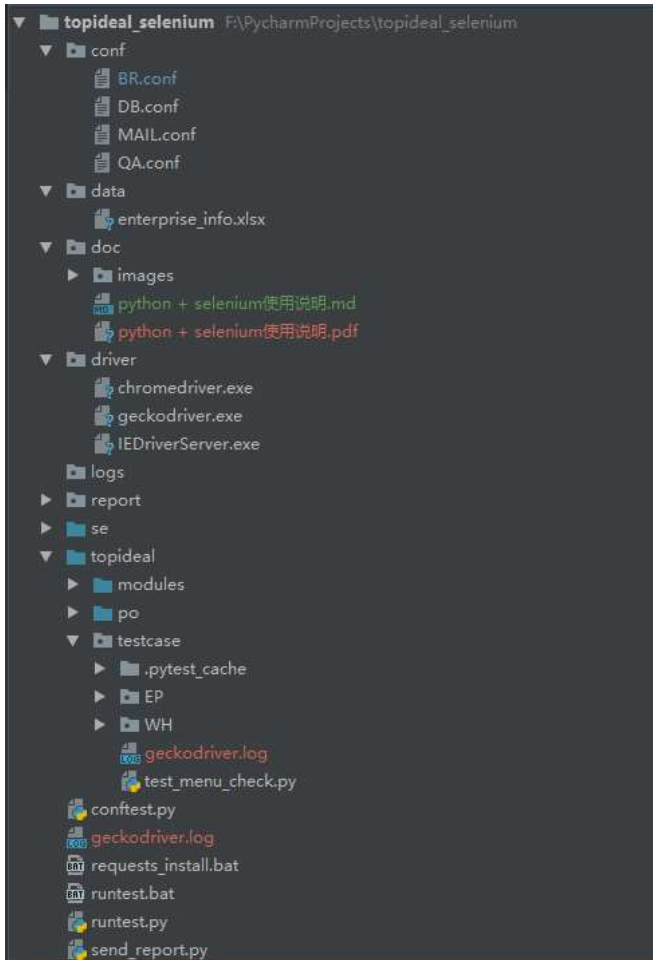


输入SSH地址，点击【Clone】按钮



3.工程目录说明

框架代码目录结构如下：



- conf 配置文件目录：

这个目录主要是存放配置文件的信息，通常会存放系统一些基础信息配置、数据库配置、调用参数等；

```
[base_info]
url = http://[redacted].topideal.com

[account_info]
username = [redacted]
password = [redacted]
```

配置文件信息，可以使用`helper.get_config_item`读取：

例如：`login_url = helper.get_config_item("base_info", "url", "QA.conf")`

- data 测试数据目录：

这个目录主要是存放测试用例数据。通常是一些excel文件。

可以使用`helper.get_excelData_by_key(key, data_file_name, sheetname)`
或者：`helper.get_excelData_by_cell(row, col, data_file_name, sheetname)` 读取数据

- doc 文档目录

主要存放项目或者框架的相关文档。

- driver 浏览器驱动目录： 存放浏览器驱动驱动的文件



- log 日志文件目录：

存放日志文件

- report 报告文件目录：

存放测试报告文件

- se selenium 框架一些公共的封装方法

这个目录主要会存放一些公共的方法，比如：

```
helper 主要封装了一些浏览器的操作、读取配置文件、读取测试文件的方法；  
find 主要封装了一些查找、定位页面元素的方法；  
oper 主要封装了一些常用的页面操作的方法，比如type(),click(),select()等；  
check 主要封装了一些验证的方法wait_element(),wait_element_disappear()等；  
conn 主要封装了一些数据库连接操作的方法
```

- topideal 项目目录

实际项目的目录，根据实际的项目进行命名，子目录包含：module，po，testcase

```
module 目录主要是存放跟实际项目相关的一些公共方法；  
po 即pageobject 主要定义一些页面元素即常用的操作；  
testcase 测试用例目录；
```

3.写用例的方法

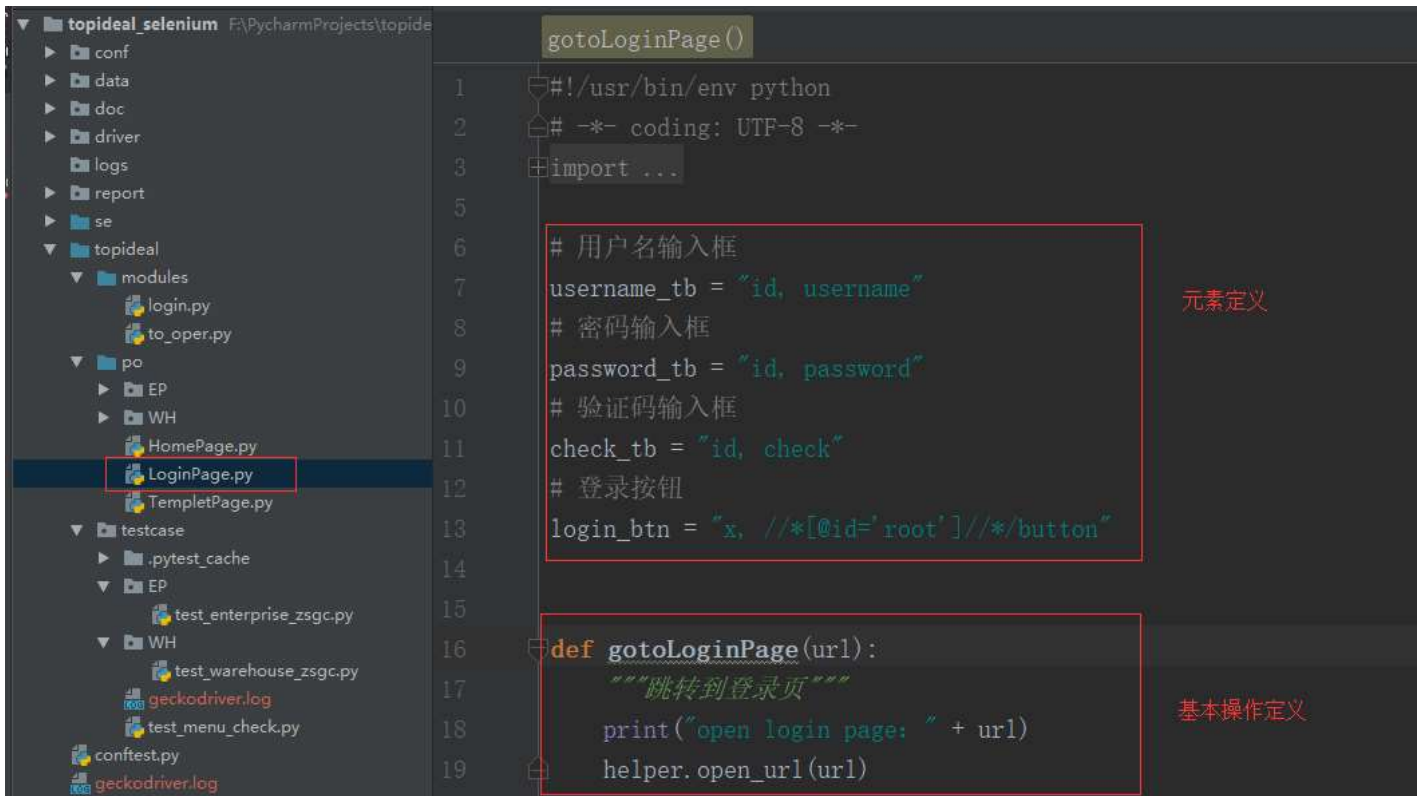
- 参考步骤：

1. 在po目录中建立xxPage.py，定义xx页面的元素和基本操作
2. 如果可以抽离出一些项目通用的方法，就将这些方法提取统一放到module目录下
3. 在testcase目录下创建测试用例

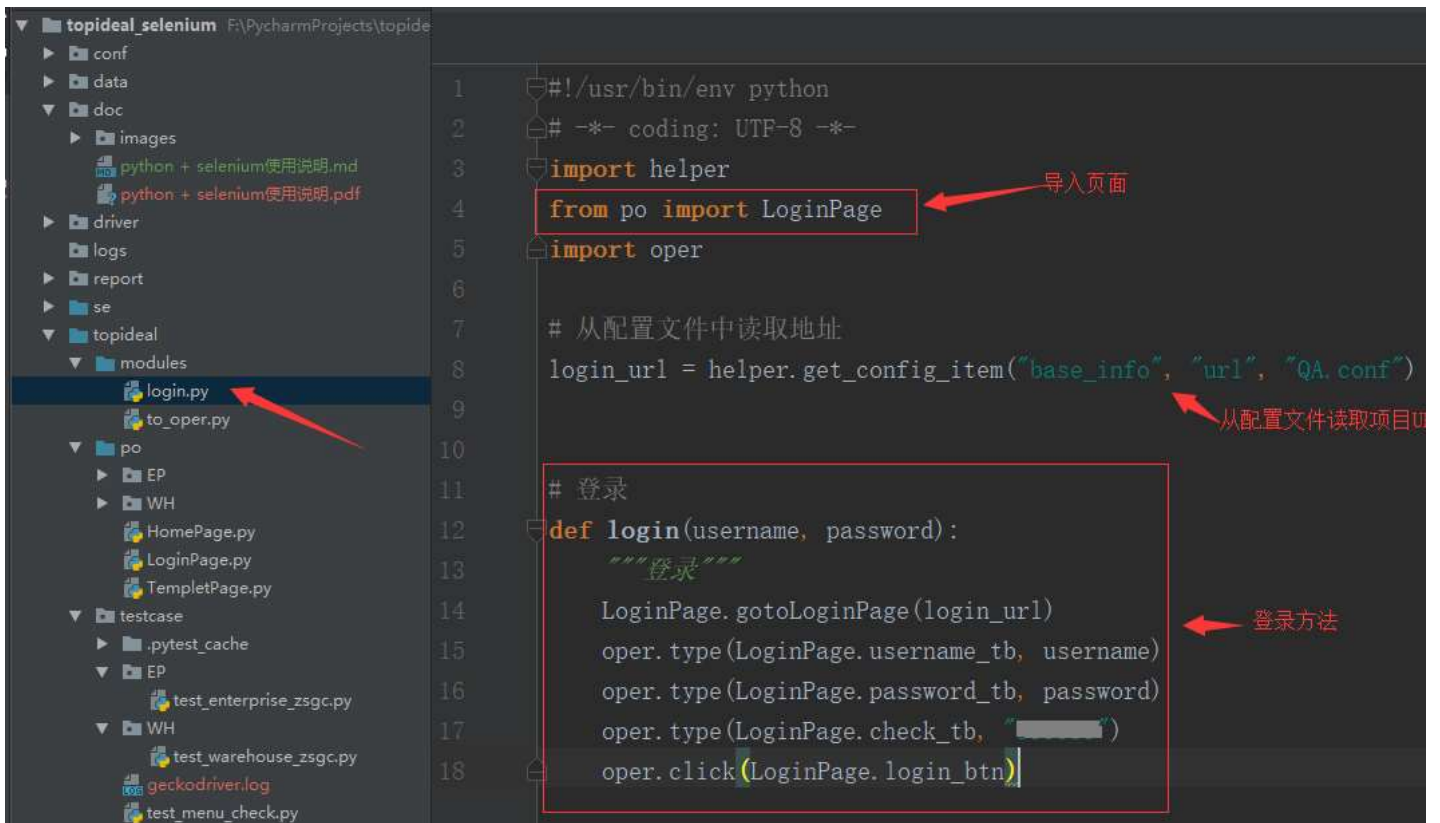
- 参考样例：

用例：xx项目-企业管理模块，企业的增删改查用例

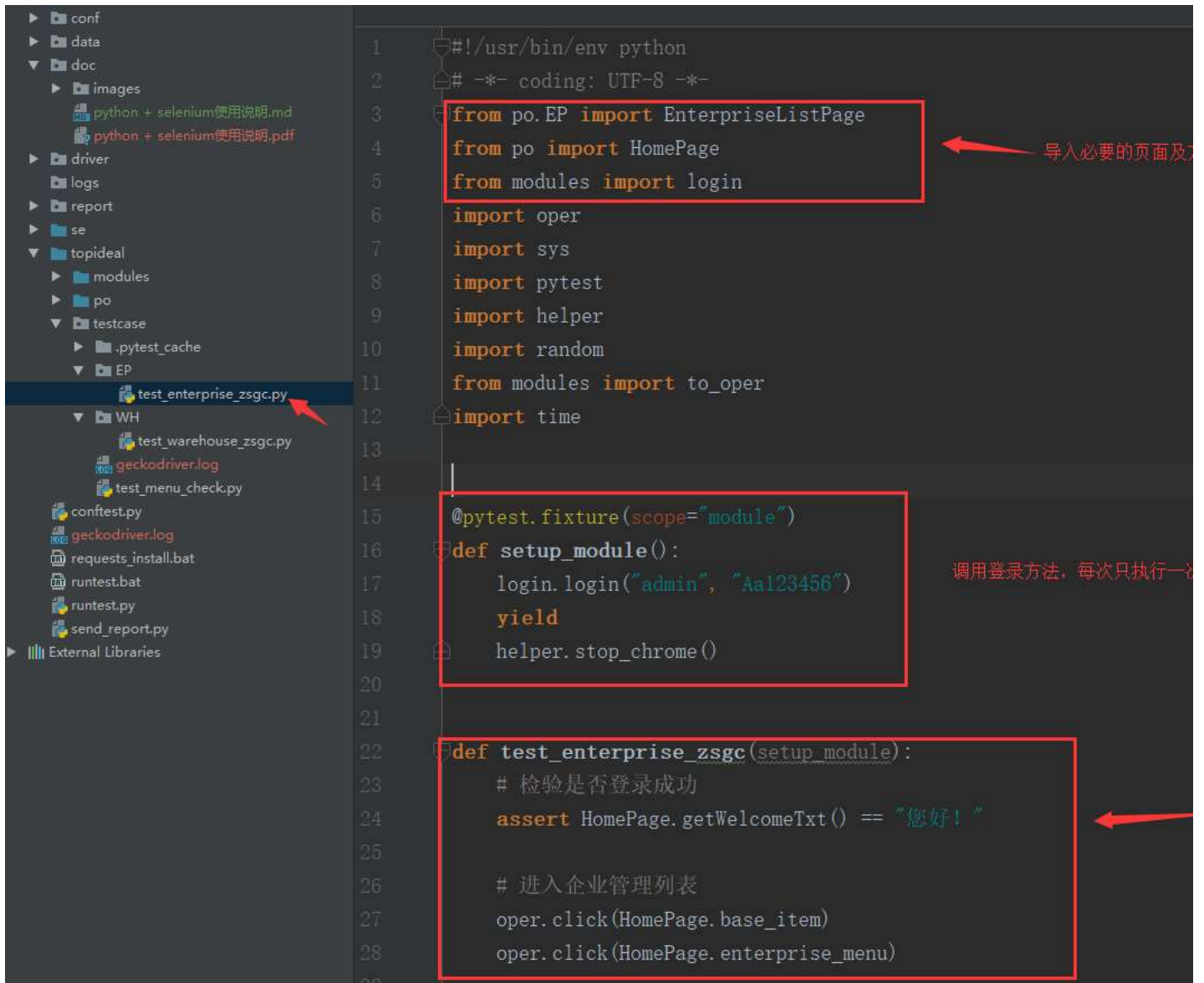
步骤1：定义页面元素和页面基本操作-->LoginPage.py



步骤2: 每个用例要执行都必须先登录, 所以登录我们将它封装公共方法放到module下:



步骤3: 在testcase下编写用例:




```
def test_enterprise_zsgc(setup_module):
    # 检验是否登录成功
    assert HomePage.getWelcomeTxt() == "您好!"

    # 进入企业管理列表
    oper.click(HomePage.base_item)
    oper.click(HomePage.enterprise_menu)

    # 新增物流企业
    company_code = "cc" + str(random.randint(10000, 99999))
    alias = "alias" + str(random.randint(10000, 99999))
    EnterpriseListPage.add_wl_enterprise(company_code, alias)
    time.sleep(1)

    # 根据简称查询企业, 校验是否添加成功
    EnterpriseListPage.search_by_alias(alias)
    time.sleep(1)
    assert to_oper.get_cell_data(1, 2) == "物流企业" and to_oper.get_cell_data(1, 4) == alias

    # 修改企业简称
    alias2 = "alias" + str(random.randint(10000, 99999))
    EnterpriseListPage.edit_wl_enterprise_info(alias=alias2)
    time.sleep(1)


    # 根据新简称查询列表
    EnterpriseListPage.search_by_alias(alias2)
    time.sleep(2)
    assert to_oper.get_cell_data(1, 4) == alias2
```

- 采用POM模型进行用例设计, 会使得用例看起来很简洁且容易阅读, 如果后期页面进行修改, 也只要修改对应的pageObject, 减少因为UI改变带来的大量的维护量。


4.用例执行方法

用例执行分为单文件执行、调制, 还有批量执行:

- 单文件执行的方法:

- 1.直接在文件上右键, 执行RUN
- 2.在pycharm右上角点击绿色三角  按钮

- 单文件调试的方法:

- 1.打断点, 直接在文件上右键, 执行Debug
- 2.打断点, 在pycharm右上角点击绿色瓢虫  按钮

- 批量执行用例的方法:

在工程目录下, 找到runtest.py 文件, 执行, 然后根据提示操作:

```
C:\Python36\python.exe F:/PycharmProjects/topideal_selenium/runtest.py
<class 'Exception'>
<第一步: 请选择要启动的浏览器(不输入默认chrome): 1:Chrome 2:Firefox 3:IE >
->请输入序号: 1
您选择了启动Chrome浏览器
<第二步: 选择要执行的用例的文件路径; 输入路径: \testcase\ ; 多个路径写法: \testcase\DEM01;\testcase\DEM02 >
->输入用例路径: \testcase
<第三步: 选择要执行的用例的文件名前缀; 比如: test_ 或者 test_aa_ >
->输入要执行的用例的前缀: test_
```

执行后, 会在工程目录下, 生成runtest.bat文件, 执行它就可以批量执行脚本了:

```
C:\Windows\system32\cmd.exe

F:\PycharmProjects\topideal_selenium>py.test ./topideal/testcase/test_menu_check.py ./topideal/testcase/EP/test_enterpri
se_zsgc.py ./topideal/testcase/WH/test_warehouse_zsgc.py --html= ./report/report.html
===== test session starts =====
platform win32 -- Python 3.6.1, pytest-3.5.0, py-1.5.3, pluggy-0.6.0
sensitiveurl: .*
rootdir: F:\PycharmProjects\topideal_selenium, inifile:
plugins: variables-1.7.1, timeout-1.2.0, selenium-1.9.1, metadata-1.7.0, html-1.14.2, base-url-1.4.1
collected 3 items

topideal\testcase\test_menu_check.py
DevTools listening on ws://127.0.0.1:12392/devtools/browser/9d2f330b-d0fa-45eb-9bb3-42e5bff86138
[ 33%]
topideal\testcase\EP\test_enterprise_zsgc.py
DevTools listening on ws://127.0.0.1:12219/devtools/browser/9934be04-81a2-4d7e-a649-16c3debc75e0
[ 66%]
topideal\testcase\WH\test_warehouse_zsgc.py
DevTools listening on ws://127.0.0.1:12646/devtools/browser/e9f7cf41-df03-4468-bce5-12becf166529
[100%]

generated html file: F:\PycharmProjects\topideal_selenium\report\report.html -
===== 3 passed in 80.29 seconds =====

F:\PycharmProjects\topideal_selenium>
```

查看测试报告:

Report generated on 15-May-2018 at 09:30:44 by [pytest-html](#) v1.14.2

Environment

JAVA_HOME	C:\Program Files (x86)\Java\jdk1.8.0_144
Packages	{'pytest': '3.5.0', 'py': '1.5.3', 'pluggy': '0.6.0'}
Platform	Windows-10-10.0.16299-SP0
Plugins	{'variables': '1.7.1', 'timeout': '1.2.0', 'selenium': '1.9.1', 'metadata': '1.7.0', 'html': '1.14.2', 'base-url': '1.4.1'}
Python	3.6.1

Summary

3 tests ran in 80.22 seconds.

(Un)check the boxes to filter the results.

☒ 3 passed, ☐ 0 skipped, ☐ 0 failed, ☒ 0 errors, ☐ 0 expected failures, ☐ 0 unexpected passes

Results

[Show all details](#) / [Hide all details](#)

▲ Result	▼ Test	▼ Duration
Passed (show details)	topideal/testcase/test_menu_check.py::test_menu_check	13.84
Passed (show details)	topideal/testcase/EP/test_enterprise_zsgc.py::test_enterprise_zsgc	12.16
Passed (show details)	topideal/testcase/WH/test_warehouse_zsgc.py::test_enterprise_zsgc	7.78