**UI自动化框架使用**

**—基于python与selenium的实现**

**文档所有者**

|  |  |  |
| --- | --- | --- |
| **姓名** | **职位** | **部门** |
|  |  |  |
|  |  |  |

**批准人列表**

|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | **职位** | **批准时间** | **备注** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**参考文档**

|  |  |  |
| --- | --- | --- |
| **文档名称** | **作者** | **内容主题** |
|  |  |  |
|  |  |  |
|  |  |  |

**术语和名词解释**

|  |  |
| --- | --- |
| **术语** | **定义解释** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**文档修订摘要**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **版本日期** | **修改者** | **修改内容** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 简介

## 目标

该文档主要介绍基于该框架下，直接上手编码实施UI自动化。

该文档是动态更新的，根据需求，会对某些章节做内容增加、删除、修改等。

该文档目标读者为项目经理、开发人员、测试人员等。

## 范围

该文档主要覆盖以下内容：

* 编码指导
* 测试运行

1. IDE执行脚本
2. 命令行执行脚本

* 编码姿势示例

## 假设

对本文档的用户做了以下假设：

* 熟悉Python语言
* 熟悉Selenium
* 了解Python unittest框架
* 了解常用的页面元素定位方法，如xpath等
* 拥有一定的UI自动化开发经验
* 初步了解框架设计及实现思路

# 编码指导

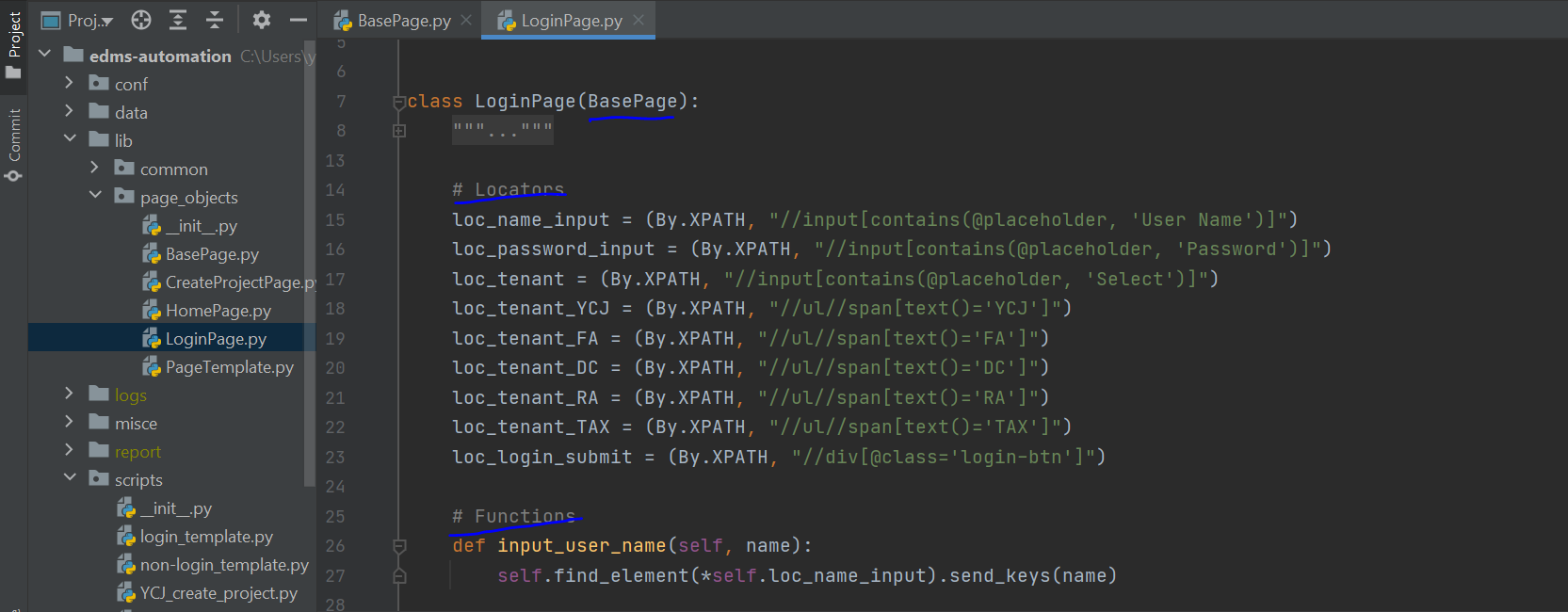
UI自动化本质上就是模拟用户人机交互。常见的人机交互如点击按钮、输入文本、复制内容、访问超链接等等，这些交互都有标的对象，也就是用户操作的对象。因此，概括地说UI自动化实际上做的事，就是找到对象，并针对该对象执行一些动作。

框架原生支持2种方式找对象：XPATH， CSS-SELECTOR。XPATH语法相对繁琐一点，但更易上手；CSS-SELECTOR语法相对简洁一点，且效率也更高，但上手要投入更多学习时间。如何选择，因人而异，但只要定位后，方法均支持。

针对对象的操作，框架已经封装好了大部分基本的且常用的操作，维护在lib/page\_objects/BasePage.py中，所有其他页面均继承这些公共方法，以提高编码效率和代码复用。

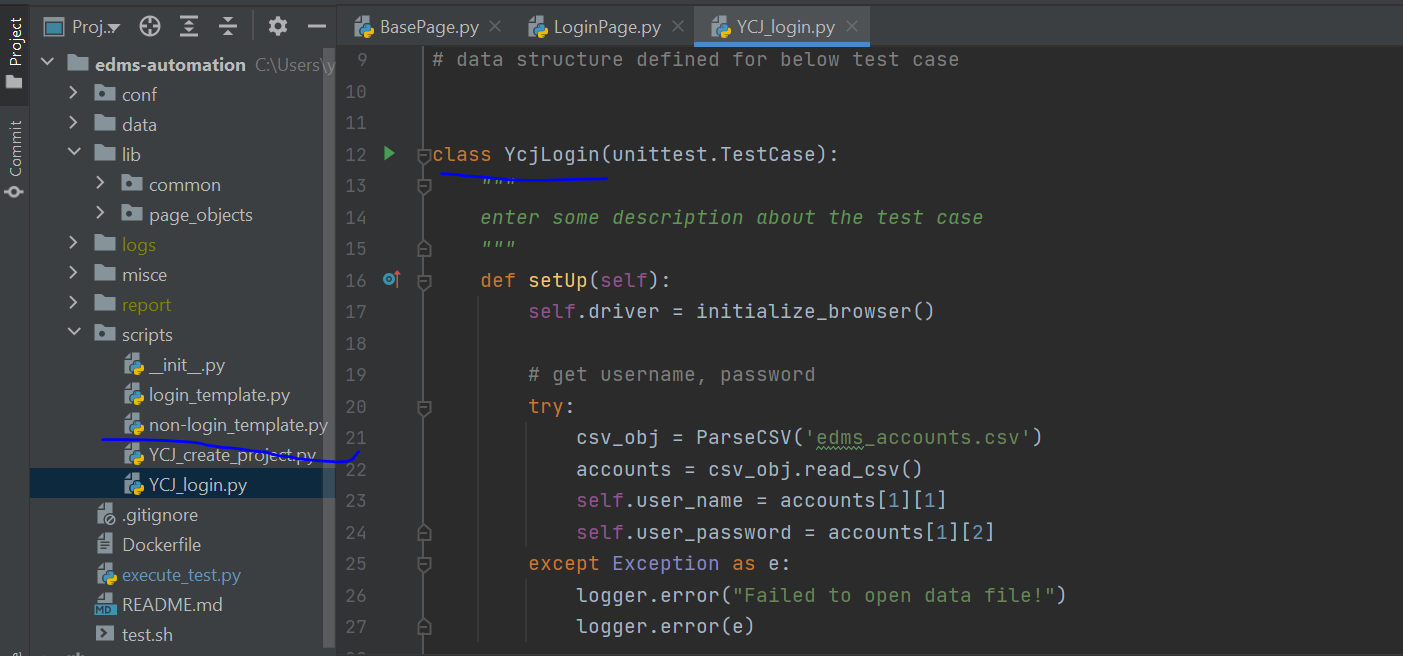
下面就获取该框架后，在不了解该框架实现细节的基础上，怎样快速地写出一个测试用例做简要说明：

## 项目以页面为单位封装该页面的元素对象及针对这些对象的操作



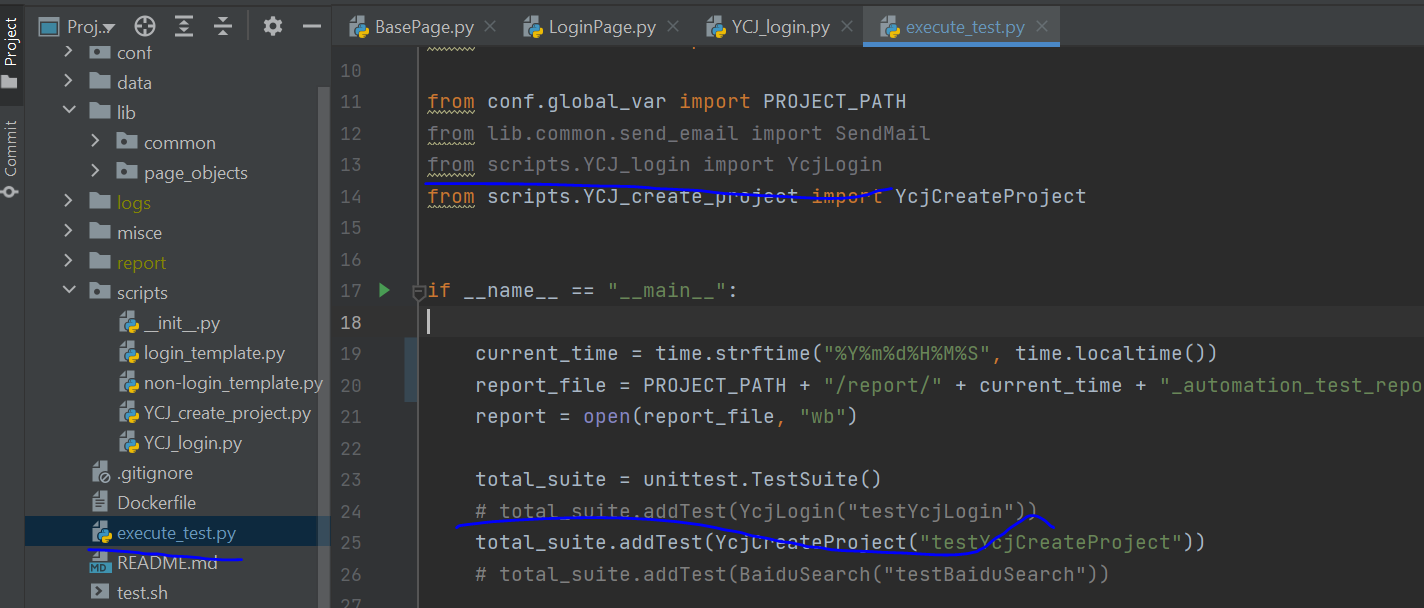
1. 页面类均继承BasePage类以继承公共方法
2. 以XPATH或CSS-SELECTOR方式定位该页面元素
3. 定义针对该页面对象的操作/方法

## 基于template创建脚本



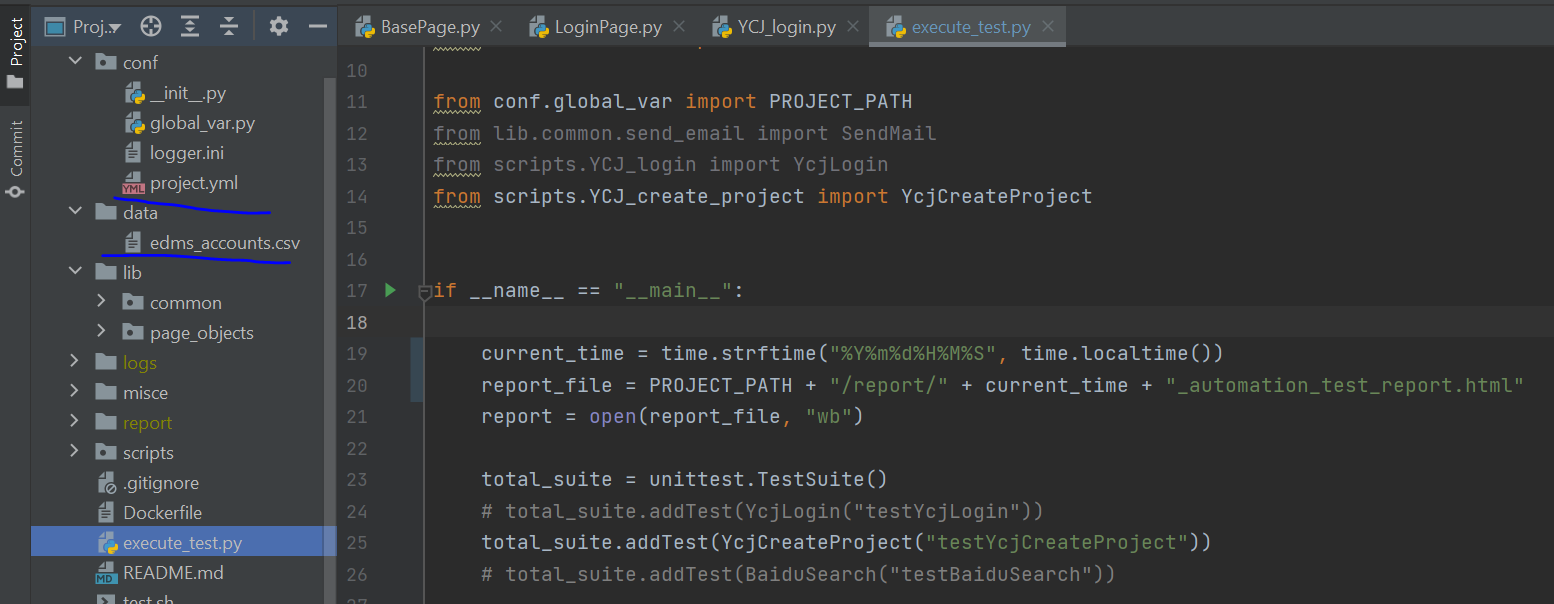
框架scripts目录下已准备需登录和不需登录的2个demo脚本模板，可直接基于他们开启新的脚本编写。

## 将脚本加入testsuite，即加入执行队列



将脚本导入模块并添加至testsuite，那么执行时改脚本即可执行。

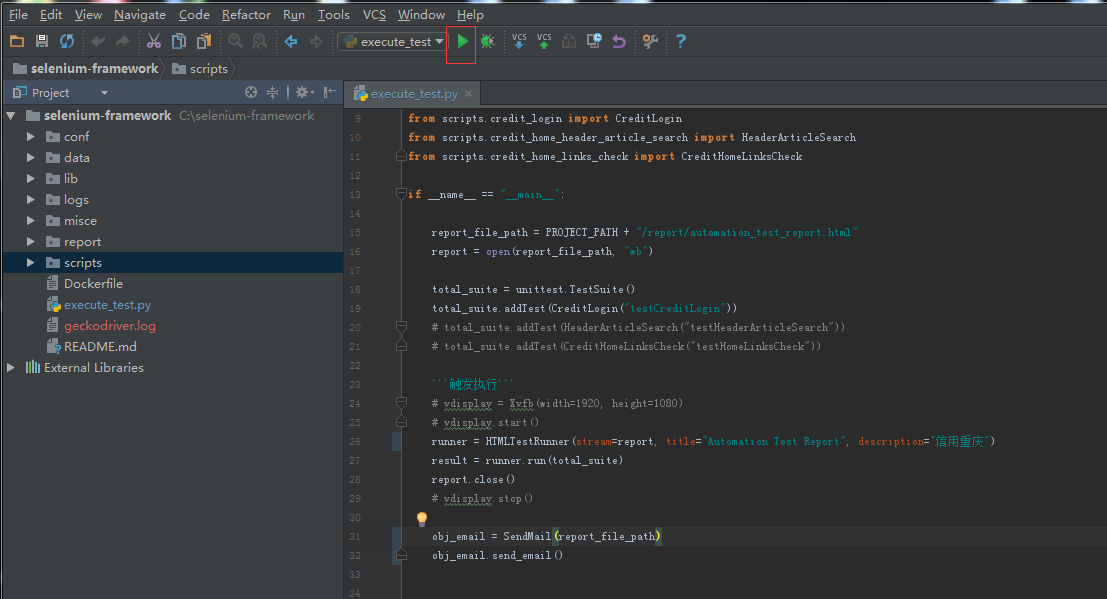
## 测试数据准备



脚本与数据分离，这是自动化测试的基本原则之一。在框架中，如果是项目级别的配置数据则可配置在conf/project.yml文件中；如果是脚本执行所需的业务数据，这可维护在data目录下，建议数据文件名与脚本类名保持一致，方便查找。

# 测试运行

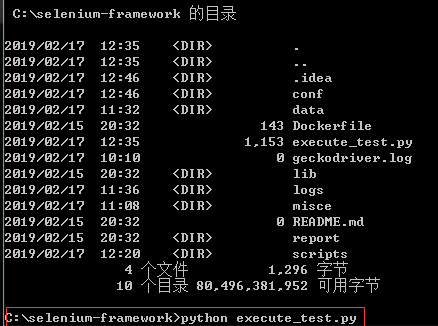
## Pycharm运行测试脚本



## 命令行运行脚本

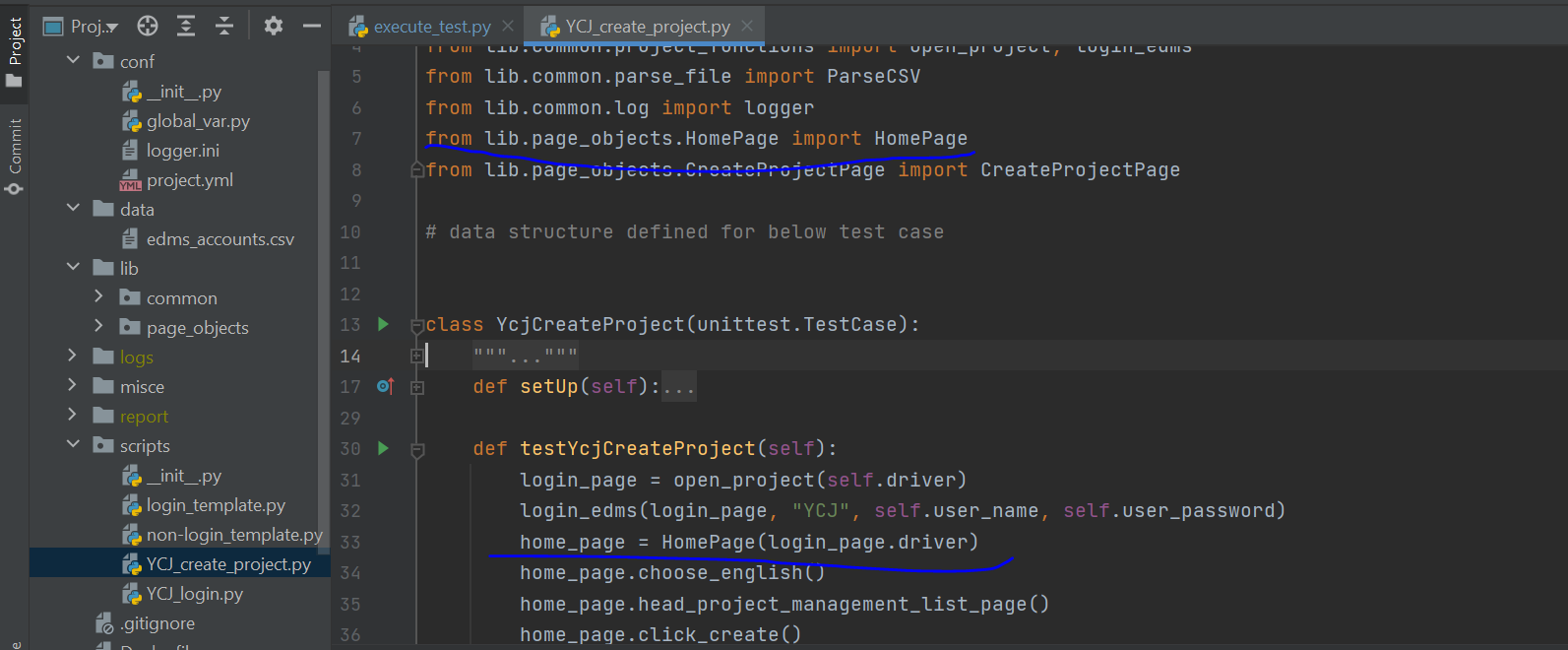
#cd project\_home\_path

#python execute\_test.py



# 编码姿势示例

1. 浏览器同一tab内的页面跳转



在脚本中，用到几个page，就可在文件头部引入几个page。然后，在脚本执行过程中涉及到切换至未实例化过的页面，即可在脚本中实例化以便引用封装在该页面类中的方法。

1. 浏览器tab之间的页面跳转