# WebUI 自动化编写说明

*文档修改记录*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 修订人 | 修订说明 | 批准人 | 修改日期 |
| 0.1 | 张正 | 初稿 |  | 2018.06.11 |
| 0.2 | 张正 | 补充测试计划配置.xlsx配置说明  补充多进程并发运行配置说明 |  | 2018.07.12 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## 一、配置(config)

### 1.1 说明

1. 设置自动化案例运行时的属性值。
2. 安排自动化案例的执行顺序。

所在路径：

…\Project\_Selenium\config

### 1.2 文件

#### 1.2.1 config.ini

**目录：**

…\Project\_Selenium\config\config.ini

**配置字段：**

1. [PROJECT]
   1. browserType

说明：

浏览器类型

示例：

browserType = Chrome

browserType = FireFox

1. [TEST]
   1. timeOut

说明：

隐形等待时间，单位秒

示例：

timeOut = 10

1. [LOG]
   1. loggingLevel

说明：

输出日志级别

支持debug, info, warn, warning, error, fatal, critical，级别有低到高，配置级别会记录>=该级别的日志信息

示例：

loggingLevel = info

loggingLevel = error

1. [EMAIL]
   1. sendEmail

说明：

自动化执行完毕后是否发送邮件

示例：

sendEmail = False

* 1. SMTPLoginName

说明：

SMTP服务器登录名

示例：

SMTPLoginName = zhangsan@outlook.com

* 1. SMTPLoginPasswd

说明：

SMTP服务器登录密码

示例：

SMTPLoginPasswd = 12345678

* 1. senderAddress

说明：

发件地址

示例：

senderAddress = zhangsan @outlook.com

* 1. SMTPServerAddress

说明：

SMTP服务器地址

示例：

SMTPServerAddress = smtp-mail.outlook.com

* 1. SMTPPort

说明：

SMTP服务器地址端口

示例：

SMTPPort = 587

* 1. emailTitle

说明：

邮件主题

示例：

emailTitle = Web自动化测试报告邮件

* 1. emailText

说明：

正文内容

示例：

详细测试报告请查看附件...

1. [receiversAddress]

说明：

收件地址 可配置多个

注意：

选项名不能重复

* 1. address1

说明：

地址1

示例：

address1 =123456789@qq.com

* 1. address2

说明：

地址2

示例：

address2 = [zhangsan@outlook.com](mailto:zhangsan@outlook.com)

#### 1.3.2 测试案例运行配置.xlsx

**目录：**

…\Project\_Selenium\config\测试案例运行配置.xlsx

**配置说明：**

如下表格为<测试案例运行配置.xlsx>的配置样式(Sheet页)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **配置方式** | **路径** | **文件名** | **类名** | **方法名** | **是否执行** | **场景描述** | **说明** |
| 1-按方法运行 | test | 案例\_1.py | Test\_1 | test\_baidu\_1\_search | Y |  |  |
| 1-按方法运行 | test | 案例\_2.py | Test\_2 | test\_baidu\_2\_search | Y |  |  |
| 1-按方法运行 | test | 案例\_2.py | Test\_2 | test\_baidu\_3\_search | N |  |  |
| 2-按类名运行 | test | 案例\_2.py | Test\_2 | - | Y |  |  |
| 3-按路径运行 | \test\测试集\_1\ | - | - | - | Y |  |  |
| 3-按路径运行 | \test\ | - | - | - | Y |  |  |

1. 配置方式

说明：

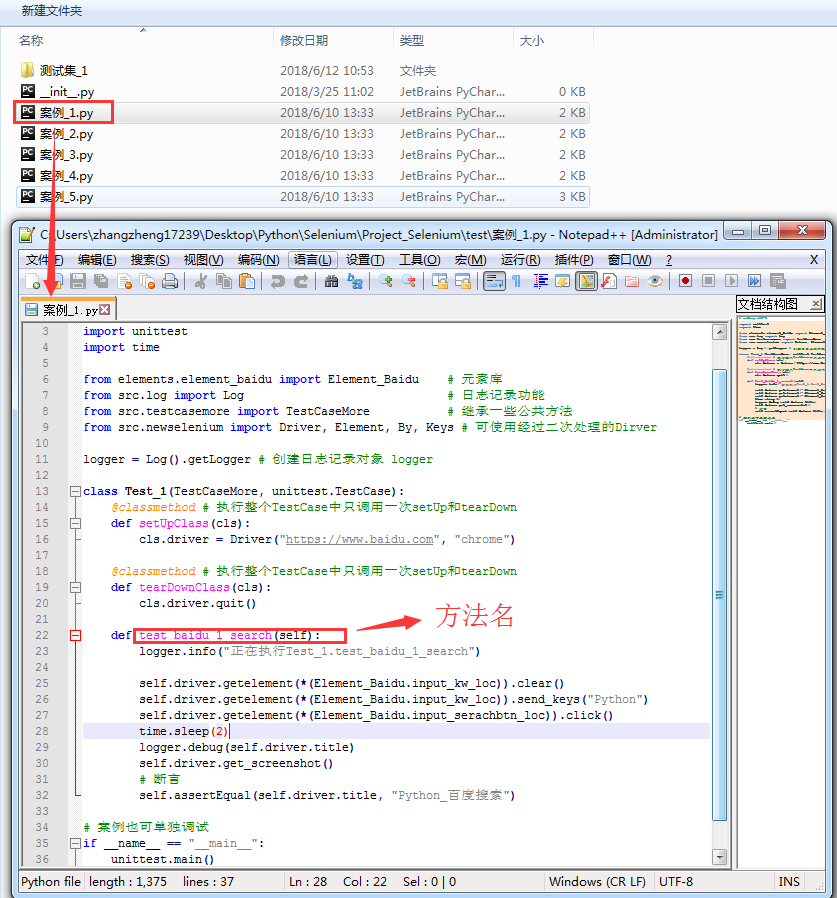
1. 组织测试案例、运行测试案例的方式。
2. 运行时会按照案例顺序从上往下执行。
3. 多进程并发执行只需要在该案例中配置其他Sheet页即可，同一Excel中的不同Sheet页并发执行。
   1. 1-按方法运行

说明：

我们以案例1.py文件为例，如下图。

执行时只会执行该方法内的案例。

如果一个py文件中有多个测试案例(方法)，而测试时只执行其中的某些，可以通过这一配置实现。



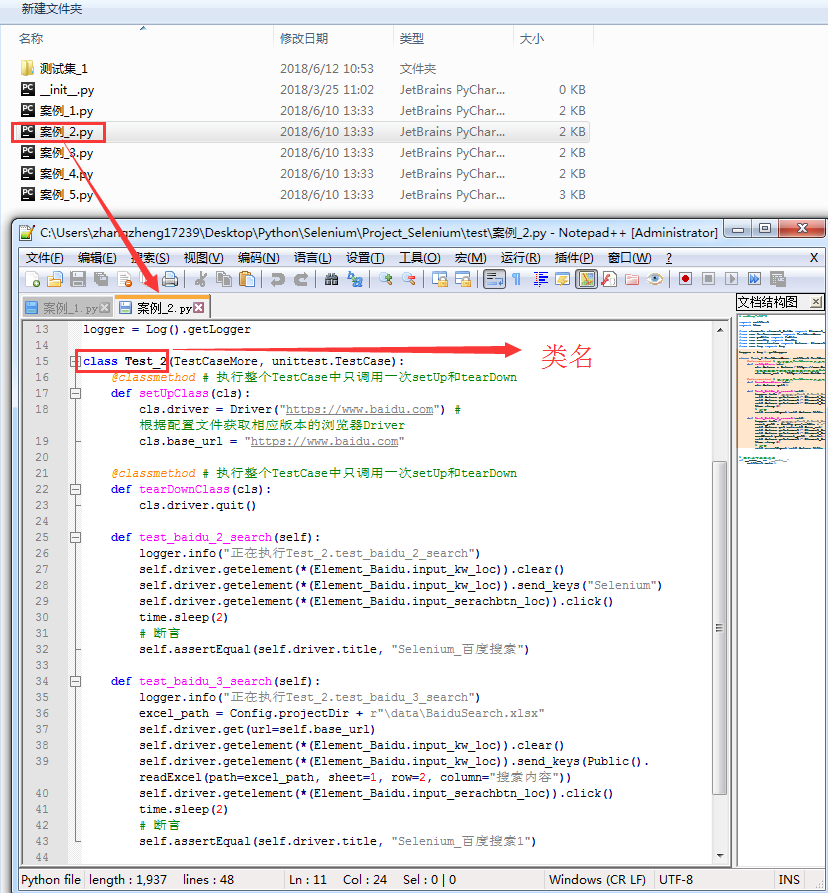
* 1. 2-按类名运行

说明：

我们以案例2.py文件为例，如下图。

配置该文件的类并以‘2-按类名运行’则运行时会执行该案例的所有方法(test\_baidu\_2\_search 和 test\_baidu\_3\_search)。

如果一个py文件中有多个测试案例(方法)，并需要全部执行，可以通过类名配置运行。



* 1. 3-按路径运行

说明：

当前…\Project\_Selenium\test 下的目录文件结构如下：

---->.\test\

---->\测试集\_1\

---->测试集1\_案例1\_zhaopin\_1.py

---->案例\_1.py

---->案例\_2.py

---->案例\_3.py

---->案例\_4.py

---->案例\_5.py

若按照路径配置运行时，例如配置“\test\”，执行时会先执行当前目录下的文件(案例\_1.py到案例\_5.py)，然后再执行当前目录中文件夹下的其他案例(测试集\_1下的测试集1\_案例1\_zhaopin\_1.py)。

1. 路径

说明：

1.路径务必为相对路径

2.以项目中案例目录(test)为起始目录

示例：

test\directory1\directory2

1. 文件名

说明：

此处输入测试用例的文件名，包含后缀。

示例：

案例\_1.py

1. 类名

说明：

文件中定义的类名，区分大小写

1. 方法名

说明：

类中定义的方法名，区分大小写

1. 是否执行

说明：

N-不执行

Y-执行

1. 场景描述

说明：

描述场景

1. 说明

#### 1.3.3 测试计划配置.xlsx

**目录：**

…\Project\_Selenium\config\测试计划配置.xlsx

**配置说明：**

如下表格为<测试计划配置.xlsx>的配置样式

|  |  |  |  |
| --- | --- | --- | --- |
| **顺序执行** | **文件名** | **是否执行** | **说明** |
| 1 | 1-测试案例运行配置.xlsx | Y | 执行前准备 |
| 2 | 2-测试案例运行配置.xlsx | Y | 并发执行\_1 |
| 3 | 3-测试案例运行配置.xlsx | Y | 并发执行\_2 |
| 4 | 3-测试案例运行配置.xlsx | Y | 执行后处理 |

1. 顺序执行

说明：

1. 组织《测试案例运行配置.xlsx》的运行顺序。
2. 运行时会按照案例顺序从上往下执行。
3. 文件名

说明：

配置…\Project\_Selenium\config\路径下“测试案例运行配置.xlsx”的文件名包含后缀格式。

1. 是否执行

说明：

N-不执行

Y-执行

## 二、日志(log)

### 2.1 说明

1. 自动化运行时记录并保存日志为log.log文件

2. 自动保存截图照片文件，文件格式为YYYMMDD-hh24mmss.png，截图方法在Driver类中会介绍

所在路径：

…\Project\_Selenium\log

## 三、数据(data)

### 3.1 说明

1. 支持对\*.csv、\*.xlsx 格式的数据文件读写，可以将测试数据维护在此目录。

所在路径：

…\Project\_Selenium\data

### 3.2 读写方法

**示例文件：**

|  |  |  |
| --- | --- | --- |
| 登录名 | 密码 | 是否通过 |
| 张三 | password | N |
| 李四 | password | Y |

#### 1. Excel/CSV读

*def readExcel(self, path: str, column=1, row=1, sheet=1):*

*"""*

*读取Excel某一单元格数据*

*:param path: Excel路径 type=str*

*:param sheet: 第几个sheet页或sheet页名称 type=int or str*

*:param row: 行数 type=int*

*:param column: 列数或列字段名 type=int or str*

*:return: 单元格的值*

*"""*

*def readCsv(self, path: str, column=1, row=1, fields=True):*

*"""*

*说明：读取csv某一单元格数据*

*注意：文件编码建议使用utf-8*

*:param path: csv文件路径 type=str*

*:param row: 行数 type=int*

*:param column: 字段名或列数 type=str or int*

*:param fields: csv文件第一行是否是字段*

*:return: 单元格值*

*"""*

示例：

1. 获取指定列数和行数的单元格数据

*readExcel( path=r”C:\LoginData.xlsx”, column=1, row=1)*

*readCsv ( path=r”C:\LoginData.csv”, column=1, row=1)*

1. 获取指定列名和行数的单元格数据

*readExcel( path=r”C:\LoginData.xlsx”, column=”登录名”, row=1)*

*readCsv ( path=r”C:\LoginData. csv”, column=”登录名”, row=1)*

#### 2. Excel/CSV写

*def writeExcel(self, path, value="", column=1, row=1, sheet=1):*

*"""*

*写入Excel指定单元格数据*

*:param path: Excel路径 type=str*

*:param row: 行数 type=int*

*:param column: 列数或列字段名 type=int or str*

*:param value: 需要写入的数据 type=str*

*:param sheet: 第几个sheet页或sheet页名称 type=int or str*

*"""*

*def writeCsv(self, path: str, value="", column=1, row=1, fields=True):*

*"""*

*说明：写入csv指定单元格数据*

*注意：文件编码建议使用utf-8*

*:param path: csv文件路径 type=str*

*:param value: 需要写入的数据 type=str*

*:param row: 行数 type=int*

*:param column: 字段名或列数 type=str or int*

*:param fields: csv文件第一行是否是字段*

*:return: None*

*"""*

示例：

1. 根据指定行列数设置单元格数值

*readExcel( path=r”C:\LoginData.xlsx” , value=”N” , column=3, row=2)*

*readCsv( path=r”C:\LoginData. csv” , value=”Y” , column=3, row=3)*

1. 根据指定列名和行数设置单元格数值

*readExcel( path=r”C:\LoginData.xlsx” , value=”李四” , column=”是否通过”, row=2)*

*readCsv( path=r”C:\LoginData. csv” , value=”李四” , column=”是否通过”, row=3)*

## 四、驱动(drivers)

### 4.1 说明

管理浏览器对应的驱动文件。

自动化案例执行时使用的是该目录下的驱动。

所在路径：

…\Project\_Selenium\ drivers

## 五、元素(elements)

### 5.1 说明

在该目录下编写页面的元素，可以将元素与案例隔离，分别维护。

**所在路径：**

…\Project\_Selenium\ elements

**我们拿百度首页举例子：**

在elements目录中创建一个“百度.py”，并写入如下代码。

*from selenium.webdriver.common.by import By*

*class 首页():*

*输入框 = (By.CSS\_SELECTOR, "#kw")*

*确定按钮 = (By.CSS\_SELECTOR, "#su")*

然后可以再test的案例中引用该元素

*class 百度首页测试(TestCaseMore, unittest.TestCase):*

*@classmethod*

*def setUpClass(cls):*

*cls.driver = Driver("https://www.baidu.com", "chrome")*

*@classmethod # 执行整个TestCase中只调用一次setUp和tearDown*

*def tearDownClass(cls):*

*cls.driver.quit()*

*def test\_baidu\_1\_search(self):*

*self.driver.getelement(\*(首页.输入框)).clear()*

*self.driver.getelement(\*(首页.输入框)).send\_keys("Python")*

*self.driver.getelement(\*(首页.确定按钮)).click()*

## 六、报告(report)

### 6.1 说明

自动化运行结束后生成报告文件，格式为“ResultReport\_YYYYMMDD\_hh24mmss.html”

**所在路径：**

…\Project\_Selenium\ report

## 七、源码(src)

### 7.1 说明：

该目录为项目源码

**所在路径：**

…\Project\_Selenium\ src

## 八、测试用例(test)

### 8.1 说明

该目录为测试用例存放目录

**所在路径：**

…\Project\_Selenium\ test

**示例代码：**

|  |
| --- |
| # coding:utf-8  **import** unittest  **import** time  **from** elements**.**百度 **import** 首页 # 元素库  **from** src**.**log **import** Log # 日志记录功能  **from** src**.**public **import** Public # 继承一些公共方法  **from** src**.**testcasemore **import** TestCaseMore # 集成扩展unittest.TestCase功能的类  **from** src**.**newselenium **import** Driver # 使用经过二次封装的Dirver  logger **=** Log**().**getLogger # 创建日志记录对象,记录日志信息通过logger.info logger.error选择不同级别来记录  **class** **Test\_1(**TestCaseMore**,** unittest**.**TestCase**):** # 继承TestCaseMore 和 unittest.TestCase两个类，必填  *@classmethod* # 执行整个TestCase中只调用一次setUp  **def** setUpClass**(**cls**):** # 案例初始化时执行  cls**.**driver **=** Driver**(**"https://www.baidu.com"**,** "chrome"**)** # 初始化驱动，指定浏览器类型并打开初始url  *@classmethod* # 执行整个TestCase中只调用一次tearDown  **def** tearDownClass**(**cls**):** # 案例结束时执行  cls**.**driver**.**quit**()**  **def** test\_百度搜索测试\_1**(**self**):**  # 测试用例函数 类中至少要有一个  logger**.**info**(**"正在执行Test\_1.test\_baidu\_1\_search"**)** # 记录日志，级别为info  self**.**driver**.**getelement**(\*(**首页**.**输入框**)).**clear**()** # 操作步骤  self**.**driver**.**getelement**(\*(**首页**.**输入框**)).**send\_keys**(**"Python"**)** # 操作步骤  self**.**driver**.**getelement**(\*(**首页**.**确定按钮**)).**click**()** # 操作步骤  time**.**sleep**(**2**)**  logger**.**debug**(**self**.**driver**.**title**)** # 记录日志，级别为debug  self**.**driver**.**get\_screenshot**()** # 截图，自动保存在log目录下  # 断言  self**.**checkElement**(**self**.**driver**,** By**.**CSS\_SELECTOR**,** ".class"**)** # 检查指定元素是否存在  self**.**assertEqual**(**self**.**driver**.**title**,** "Python\_百度搜索"**)** # 断言标题是否为“Python\_百度搜索”  # 案例也可单独调试  **if** \_\_name\_\_ **==** "\_\_main\_\_"**:**  unittest**.**main**()** |

## 九、其他

### 9.1 .idea

**说明：**

项目工程配置文件，可以忽略，请勿删除

### 9.2 tutorial

**说明：**

自动化框架相关手册和说明

### 9.3 runtest.bat

**说明：**

自动化执行bat脚本

### 9.4 runtest.py

**说明：**

自动化开始执行py文件