三天WEB自动化项目实战训练营

第三天：WEB自动化实战集成篇

金牌讲师：码尚教育-北凡老师
直播时间：2022-04-28 20：00

1. 复杂场景下的元素定位调试
2. 参数化测试提高测试覆盖率
3. allure测试报告
4. 日志及截图的封装
5. POM和关键字驱动的对比
6. Excel+关键字的UI测试框架

上节回顾

1. 引入pytest
2. 利用fixture
3. POM封装

使用PO来编写测试用例：

1. 定义夹具，夹具中使用PO
2. 编写用例，用例中使用PO

为什么会超时：因为没有定位到

1. 复杂元素的定位和调试

定位失败，时UI自动化最大的坑

按照以下几个步骤进行排查：

1. 定位表达式写对了吗？
2. 元素出现了吗？
3. 元素是否在iframe中？

元素在iframe中，就会看到到，但是定位失败

怎么解决？

定位iframe + 切换iframe

- iframe
- 下来选择框
- 动态变化的元素

2. 参数化测试

2.1 什么是参数化测试

对于相似的测试用例，可以使用**参数化的方式**，实现测试代码重用

如果已有测试用例，可以通过参数化快速增加用例，提高测试覆盖率

2.2 实现参数化测试

```python
import pytest


def test_a():
    print("这是一个用例")


@pytest.mark.parametrize(
    "i",  # 参数名
    range(5),  # 参数值
)
def test_b(i):
    print(f"这是{i}个用例")
```

2.3 实战：测试必填字段

1. 准备测试数据
2. 为用例添加装饰器

```python
@pytest.mark.parametrize("name,tel,sheng,shi,qu,address,msg", test_data)
def test_new_address(user_driver, name, tel, sheng, shi, qu, address, msg):

    page = AddressPage(user_driver)
    page.input_adders(name, tel, sheng, shi, qu, address)

    assert msg == page.get_msg()
```

3. 测试报告

1. 生成网页测试报告

allure：

1. 生成测试结果
2. 生成测试报告

安装pytest插件

```
pip install allure-pytest
```

启用pytest插件

```
import os

import pytest

if __name__ == "__main__":
    pytest.main(["--alluredir=./.allure_results"])  # 启用插件
    os.system("allure serve ./.allure_results")  # 生成HTML报告
```

2. 页面截图

截图：selenium

展示截图： allure

什么地方应该截图?

1. PO实例化之后
2. 元素定位之前
3. 除了弹窗之外其他**页面交互**

3. 日志的封装

1. 配置日志
2. 启用配置
3. 添加日志
    1. 每个文件中，创建logger

    ```
    logger = logging.getLogger(__name__)
    ```

    2. 使用logger记录内容

4. 测试框架

pytest+ POM ，前提：封装Page

大型项目来说，Page多，工作量多起来来了

> 各个页面，各有各的不同
>
> 但是用户的操作 总是相似的

用户的操作：

- 鼠标、键盘

大大的减少代码定义量

再配合excel，测试用例甚至不需要写代码了

关键字驱动框架的运行流程：

1. excel列出用户的动作 （关键字）
2. 框架 读取并执行 关键字
3. 加载其他插件，实现并行测试、失败重试、生成报告等等

项目类型

- Web
    - 电商
    - 金融
    - 论坛

- 接口
    - 微信公众号
    - 论坛
    - 外卖任务委派
- 移动端
    - 原生App
    - h5混合app
    - 微信小程序

学习、工作、面试 遇到的所有问题，都可以找北凡老师

学习安排：

直播周期：4个月

直播时间：每周3、5、7 晚上20：00~22：00

永久学习

初级测试 -> 自动化测试 -> 测试开发 -> 测试架构

UI自动化测试： excel

接口自动化测试：yaml

整个自动化测试，不需要写一行代码

至少15k