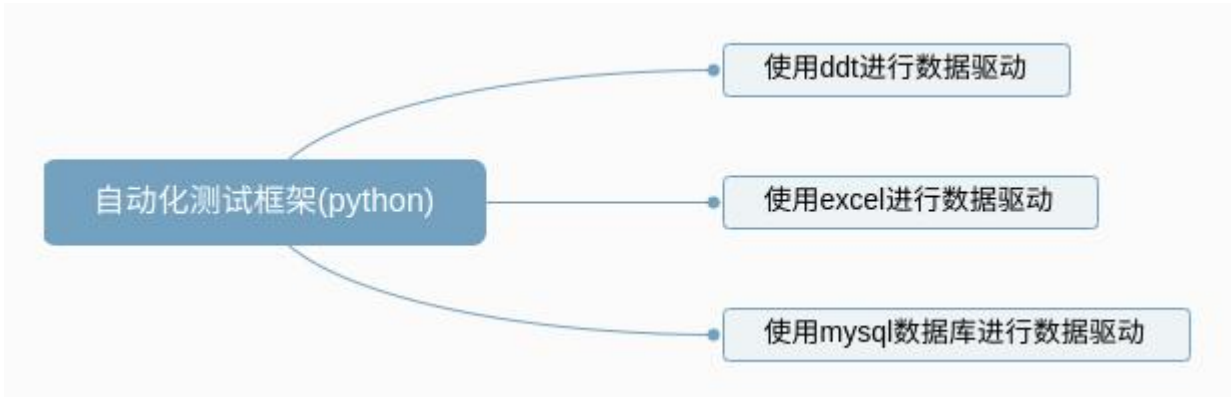
JFZ 金斧子
WWW.JFZ.COM

自动化测试框架（python）-数据驱动测试

讲师：芳姐 | 时间：2018-5-31


自动化测试框架(python)
使用ddt进行数据驱动
使用excel进行数据驱动
使用mysql数据库进行数据驱动

一、数据分离

1、用例设计

用例编号	测试点	前置条件	操作步骤	期望结果	后置条件	测试结果
01	登录	打开登录页	1.输入账号：正确账号（123） 2.输入密码：正确密码（xxx） 3.点击登录按钮	页面跳转到登录成功页， 并且显示当前账号	数据清理操作（如：清空cookie）	
02	登录	打开登录页	1.输入账号：正确账号（123） 2.输入密码：错误密码（xxx） 3.点击登录按钮	不能跳转到登录成功页， 不显示当前账号	数据清理操作（如：清空cookie）	

一、数据分离

2、参数化

```
 def login_case(self, username, password, exc_name):
        '''这里一定要以test开头'''

        self.driver.find_element_by_xpath("//*[@name='username']").send_keys(username)
        self.driver.find_element_by_xpath("//*[@name='password']").send_keys(password)
        self.driver.find_element_by_xpath("//*[@class='submit_wrap']").click()
        self.assertEqual(self.driver.title, '深圳市金斧子网络科技有限公司-ERP', msg='访问erp成
功')
        time.sleep(5)

    def test_login_01(self):
        '''理财师登录'''
        self.login_case('defang1', '123', '德芳理财')

    def test_login_02(self):
        '''项目运维登录'''
        self.login_case('defang3', '123', '德芳运维')
```

一、数据分离

3、dict

```
 lcs_dic = {"username": "defang1", "password": "123", "exc_name": "德芳理财"}
yw_dic = {"username": "defang3", "password": "123", "exc_name": "德芳运维"}


def login_case(self, username, password, exc_name):


 def test_login_01(self):
        '''理财师登录'''
        self.login_case(lcs_dic["username"], lcs_dic["password"], lcs_dic["exc_name"])

    def test_login_02(self):
        '''项目运维登录'''
        self.login_case(yw_dic["username"], yw_dic["password"], yw_dic["exc_name"])
```

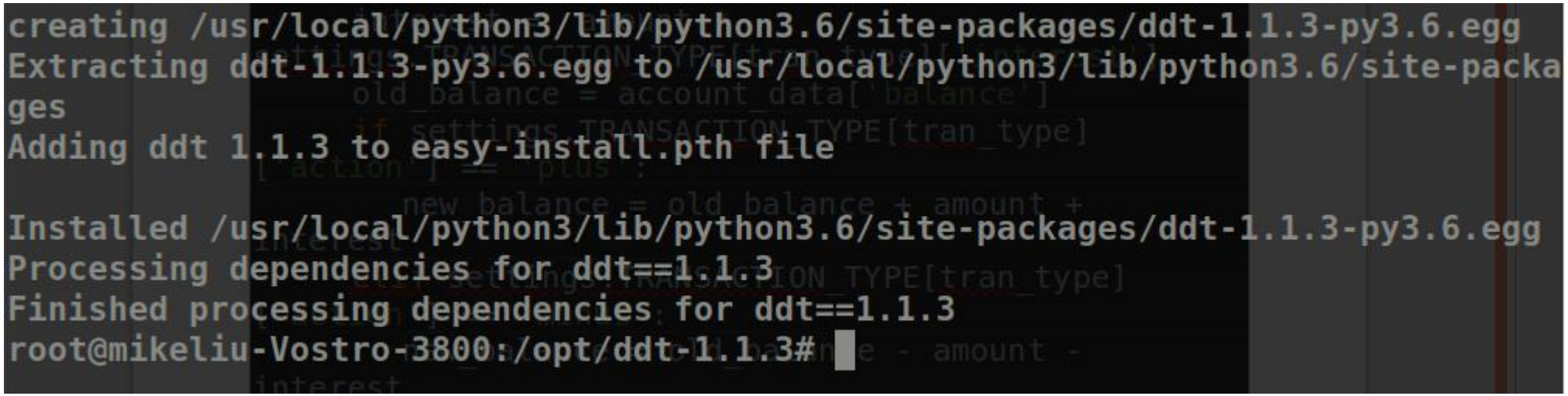
使用ddt进行数据驱动

名词解释：
DDT(Data-driven testing)：数据驱动测试

背景：
　　有些测试用例只是输入参数不一样，其它操作步骤都是一样的，比如登录
如果重复的写这些操作过程，会增加代码量，对应这种多组数据的测试用例，可以用到ddt进行数据驱动

ddt的安装

- 1、安装ddt模块，Windows在cmd里面输入,linux 在命令行终端输入
- 2、输入pip install ddt命令，进行在线安装
- 3、在上面的命令行输入不成功之后，从官网(https://pypi.org/simple/ddt/)下载，在解压文件进行安装，输入：python setup.py install，安装成功如下：



代码见：

ddt数据驱动测试/DataTest.py

日志：

```
logging.basicConfig(
    # 日志级别
    level=logging.INFO,
    # 日志格式
    # 时间，代码所在文件名，代码行号，日志级别名字，日志信息
    format='%(asctime)s %(filename)s[line:%(lineno)d] %(levelname)s %(message)s',
    # 打印日志的时间
    datefmt='%a, %Y-%m-%d %H:%M:%S',
    # 日志文件存放的目录（目录必须存在）及日志文件名
    filename='E:\\Python\\DataDrivenProject\\report.log',
    # 打开日志文件的方式
    filemode='w'

)
```

```
@ddt.ddt
class TestDemo(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome(executable_path="E:\\Python36\\chromedriver")

    @ddt.data(["defang1", "123", "德芳理财"],
              ["defang3", "123", "德芳运维"])
    @ddt.unpack

def test_dataDrivenByObj(self, username, password, expectdata):

try：

except Exception as e:
        print(e)
        logging.error("未知错误，错误信息：" + str(traceback.format_exc()))
else:
        logging.info("登录'%s',期望'%s',通过" % (username, expectdata))
```

数据文件进行数据驱动测试

DataDrivenTest.py

```
@ddt.ddt
class TestDemo(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        # 整个测试过程只被调用一次
        TestDemo.trStr = ""
    def setUp(self):

        self.driver = webdriver.Chrome(executable_path="E:\\Python36\\chromedriver")
        status = None   # 用户存放测试结果的状态，失败：fail，成功：pass
        flag = 0    # 数据驱动测试结果的标志 失败置为：0，成功置为：1

    @ddt.file_data("test_data_list.json")
    def test_dataDrivenByFile(self, value):
        # 决定测试报告中状态单元格中内容的颜色
        flagDict = {0: 'red', 1: '#00AC4E'}
```

```
wasteTime = time.time() - start - 3    # 减去强制等待的3秒
        ''' 每一组数据测试结束后，都将其测试结果信息插入表各行
        的HTML代码中，并将这些行HTML代码拼接到表里trstr变量中
        等所有测试数据都被测试结束后，传入htmlTemplate()函数中
        生成完整测试报告的HTML代码'''
        TestDemo.trStr += '''
        <tr>
          <td>%s</td>
          <td>%s</td>
          <td>%s</td>
          <td>%.2f</td>
          <td style="color: %s">%s</td>
        </tr><br />''' % (username, expectdata, startTime, wasteTime, flagDict[flag], status)

    @classmethod
    def tearDownClass(cls):
        # 写自定义的HTML测试报告
        # 整个测试过程中只被调用一次
        htmlTemplate(TestDemo.trStr)
```

ReporttestTemplate.py

```
def htmlTemplate(trData):
    htmlStr = '''<!DOCTYPE HTML>
    <html>
    <head>
    <tittle>UI自动化测试报告</title>
    <style>
    body {
        width: 80%;  /*整个body区域占浏览器的宽度百分比*/
        margin: 40px auto;
        font-weight:bold;
        font-family:'trebuchet MS','Lucida sans',SimSun;
        font-size:18px;
        color:#000;
    }
```

```
table {
        *border-coolapse:collapse; / *合并表格边框 */
        border-spacing:0; / *表格的边框宽度 */
        width:100%;     / *整个表格相对父元素的宽度 */
    }
    .tableStyle {
        /*border:solid #ggg 1px;*/
        border-style:outset;
        border-width:2px;
        /*border:2px;*/
        border-color:blue;
    }
    .tableStyle tr:hover {
        background:rgb(173,216,230);  /*鼠标滑过一行时，动态显示的颜色146,208,80*/
    }
```

```
 .tableStyle td,.tableStyle th{
        border-left:solid 1px rgb(146,208,80);
        border-top:1px solid rgb(146,208,80);
        padding:15px;
        text-align:center;
    }

.tableStyle th{
        padding: 15px;
        background-color:rgb(146,208,80);
        background-image:-webkit-gradient(linear,left top,left bottom,from(#92D050), to(#A2D668));
        /*rgb(146,208,80)*/
    }
    </style>
    </head>
    <body>
```

```
<center><h1>测试报告</h1></center><br />
        <table class= "tableStyle">
            <thead>
            <tr>
            <th>输入关键词</th>
            <th>预期结果</th>
            <th>开始时间</th>
            <th>耗 时</th>
            <th>Status</th>
            </tr>
            </thead>'''
    endStr = '''
        </table>
    </body>
    </html>'''
    # 拼接完整的测试报告HTML页面代码
    html = htmlStr + trData + endStr
    print(html)
    # 生成HTML文件
    with open(r"E:\\Python\\DataDrivenProject\\testTemplate.html", "w") as fp:
        fp.write(html)
```

test_data_list.json

```
[
  "defang1||123||德芳理财",
  "defang3||123||德芳运维"
]
```

查看结果：

UI自动化测试报告

测试报告

输入关键词	预期结果	开始时间	耗时	Status
defang3	德芳运维	2018-05-30 09:04:47	4.48	pass

作业：

1、学习列表，字典的（增，删，改，查）

2、使用ddt和json数据驱动完成自动化测试用例

Thanks!

科　技　提　升　投　资　品　质

金斧子
WWW.JFZ.COM