RobotFramework框架

讲师：Pansir

安装

`pip install robotframework==3.1.1`

配置pycharm环境

参考博客文章：http://birdgugu.com/2019/09/03/pycharm-with-rf/

对比分析

分类	python(unittest)	RobotFramework
用例包	python项目包	文件夹
测试套件	xxx.py文件	suite(txt或robot文件)
导入	import	`*** Settings ***`
全局(属性)变量	a = xx 或者 配置文件	`*** Variables ***`
方法(关键字)	def xxx():	`*** Keywords ***`
用例	def test_xxx():	`*** Test Cases ***`
模块	xxx.py(作为模块)	xxx.robot(作为resource)
模块中的关键字	xxx.py里的def	xxx.robot里的Keywords
变量导入	import xxx.py	Variables xxx.py 或 Resource xxx.robot

目录结构

```
09tsms_robot
├── robot_commons  # 业务测试库
|   ├── __init__.py
|   ├── html_test_runner.py
|   ├── read_excel.py
|   ├── tsms_base.py
|   ├── tsms_config.py
|   ├── tsms_db.py
|   ├── tsms_ddt.py
|   ├── tsms_decorator.py
```

```
|   ├── tsms_rds.py
|   └── tsms_web.py
├── tsms_send
|   └── send_api.robot
├── tsms_sign  # 按接口分类
└── tsms_temp
└── test_data  # 测试数据
```

Suite板块

Settings

示例文件：`send_api.robot`

导入自定义类(模块)

```
*** Settings ***
Library    robot_commons.tsms_base.Tsmstest
```

> 报错：no keyword found，如何解决?

- 检查导入格式：`robot_commons.tsms_base.Tsmstest`，{包名}.{模块}.{类名}
- 检查环境变量，需要手动配置环境变量：
    1. 确认自己的环境变量，如：`~/.local/share/virtualenvs/mypython3-Xz6Q90qj/lib/python3.7/site-packages`
    2. 该目录下新建文件：`mypython.pth`
    3. 文件内容中添加包路径，包含：
        - 自定义项目包的路径：`xxx/09tsms_robot/robot_commons`
        - 项目包的父路径：`xxx/09tsms_robot`

导入配置文件(变量)

提供两种导入：py文件导入和robot配置导入

```
*** Settings ***
Documentation    Suite description
Variables  ../robot_commons/tsms_config.py
Resource    ../robot_commons/robot_variable.robot
```

Test Cases

编写用例

Variables

变量管理

Keywords

RF关键字管理

RF参数化(变量)

自定义

rf提供一些自定义方式`${变量名}`

```
test_variable:1
    [Documentation]  rf自带变量定义
    # 1. 定义整型变量
    ${num}    set variable    ${10}
    # 2. 定义字符串变量
    ${name}    set variable    hello
    # 3. 定义列表
    @{data_list}    create list    apple    pen    hello
    log    ${data_list}
    log many    @{data_list}  # for循环使用
    # 4. 定义字典
    ${data_dict}    create dictionary    sign_id=${424}    temp_id=${40}
 mobiles=13988887777
    log    ${data_dict}
```

suite全局变量

在suite最上方定义变量，本suite文件各个用例可使用

```
*** Variables ***
${sign_url}    http://1.1.1.1:5001/v1/sign
```

外部导入

在setting使用`Variables`，配置文件为相对路径，通过`${变量名}`的方式引用

外部导入场景分为三种：

1. 使用python(.py)配置文件的变量
2. 使用robot配置文件(*.robot)的变量
3. 通过自自定义的方法获取参数，这里的方法即可是python方法，也可以是rf方法

```
*** Settings ***
Documentation     测试py文件导入
Library    robot_commons.tsms_base.Tsmstest
Variables  ../robot_commons/tsms_config.py
Resource    ../robot_commons/robot_variable.robot
```

```
test_variable:2
    [Documentation]  外部变量引入
    # 1. 使用python(.py)配置文件的变量
    log    ${test_url}
    # 2. 使用robot配置文件(*.robot)的变量
    log    ${sign_url}
    # 3. 通过自自定义的方法获取参数
    ${mobiles}    create list      18077776666
    log    ${mobiles}
```

练习：分别从py文件和robot文件中导入变量，然后在用例中打印出来

RF方法(关键字)

RF关键字

关键字定义

```
接收参数返回结果
    [Arguments]  ${arg1}    ${arg2}
    ${result}    set variable    ${arg1}+${arg2}
    [Return]    ${result}
```

```
test_keyword:2
    [Documentation]  接收参数的关键字
    ${a}    接收参数返回结果    hello    kitty
    log    ${a}
```

导入和引用

导入场景有2种：

1. 当前文件下定义的关键字，无需导入，直接使用
2. 使用其他文件下导入的关键字，需要导入对应的文件

```
*** Settings ***
Documentation    关键字文件

*** Keywords ***
send_keyword_01
    log    hello

打印一行日志
    log    kitty
```

```
*** Settings ***
Documentation    Suite description
```

```
Resource    send_keywords.robot

test_keyword:1
    [Documentation]  关键字导入
    # 1. 当前文件下定义的关键字，无需导入，直接使用
    当前文件里的关键字
    # 2. 使用其他文件下导入的关键字，需要导入对应的文件，如：Resource
send_keywords.robot
    send_keyword_01
    # 3. rf里的关键字可以用中文
    打印一行日志

*** Keywords ***
当前文件里的关键字
    log    hello
```

特殊语法

- if语句

```
test_03
    ${a}    Set variable    59
    run keyword if    ${a}>=90    log    优秀
    ...    ELSE IF    ${a}>=70    log    良好
    ...    ELSE IF    ${a}>=60    log    及格
    ...    ELSE    log    不及格
```

- for循环

```
test_04
    :FOR    ${i}    IN RANGE    10
    \    log    ${i}
```

- Evaluate：导入python库，直接使用python语法

```
test_05
    ${d}    Evaluate    random.randint(1000, 9999)    random
    log    ${d}
```

Python关键字(方法)

从自己的库文件中导入对应的方法

```
*** Settings ***
Documentation    Suite description
Library    robot_commons.tsms_base.Tsmstest

send_01
    [Tags]    DEBUG
    ${mobiles}    create list    18077776666
    ${data}    create dictionary    sign_id=${424}    temp_id=${40}
 mobiles=${mobiles}
    req post    message    ${data}
    check status code    200
```

前置与后置

- 用例前置与后置

```
*** Settings ***
Test Setup    创建一个签名
Test Teardown    log    ok
```

```
创建一个签名
    ${data}    create dictionary    signature=RF创建签名    source=深圳    pics=
[]
    req post    sign    ${data}
```

思考题1：字典传少量参数尚可，当需要传入大量嵌套的参数，则不方便，期望能支持直接传入json串，应当如何修改tsms_base.req_post方法?

新增判断，如果是字符串，则转换成字典

思考题2：rf的前置与后置，不能获取返回结果，如果用例需要使用前置中的数据，应该如何处理?

新增方法，通过属性变量透传

练习：迁移签名审核用例，前置创建，后置删除

- 套件前置与后置(同上)

```
Suite Setup    log    suite start
Suite Teardown    log    suite end
```

结果断言

在RF中断言

通过调用python方法获取返回结果，对返回结果进行断言，需要使用预期结果+实际结果

在Python中断言

在python中进行断言，在rf中进行调用，只需要传入预期结果

```
send_db_rds_01
    [Tags]    DEBUG
    ${mobiles}    create list      18077776666
    ${data}    create dictionary    sign_id=${424}    temp_id=${40}
 mobiles=${mobiles}
    req post    message    ${data}
    ${uuid}    get t id
    check status code    200
    ${real_res}    tsms select    sms_send    mobile,status,consume
 uuid=${uuid}
    check res    mobile    18077776666
    check res    status    failed
    check res    consume    1
```

练习：新增的redis校验到rf用例

命令模式

执行

`*`匹配任意字符，`?`匹配单个字符

`robot`或`pybot`都可

1. 运行一条用例：`pybot --test {用例名} {套件名}`；例如：`robot --test send_01 send_api.robot`；注意报告文件会生成到当前文件下
2. 运行指定文件：`robot {套件名}`；例如：`robot send_api.robot`
3. 运行某类用例：`robot *.robot`；例如：`robot *api*.robot`
4. 运行当前目录下所有用例：`robot ./`
5. 运行某个目录下的用例：`robot {目录名}`；例如：`robot tsms_send`

标签

包含标签：`--include {tag}`；例如：`robot --include Level1 ./`

排除标签：`--exclude {tag}`；例如：`robot --exclude Level1 ./`

标签支持逻辑运算：`AND`和`OR`；例如：`robot --include Level1&Level2 ./`

其他参数

控制文件扩展名：`-F robot:txt`

用例失败重试

robot官方不推荐修改源码，只能外部重试

1. 执行所有用例：`robot --output original.xml robot_case_retry.robot`
2. 重新执行：`robot --rerunfailed original.xml --output rerun.xml robot_case_retry.robot`
3. 汇总结果：`rebot --merge original.xml rerun.xml`
4. 如果要要每次进行重试，需要写一个shell脚本来实现；或者在测试库内部实现重试，然后由RF调用

扩展控制台输出

重写监听类，启动时指定自己的监听：`robot --include Level1 --listener=RobotListener.py ./`

报告

output.xml 记录的测试结果的 XML 文件

log.html 记录 Robot Framework 运行的每一步操作，用于调试

report.html 为测试报告

allure安装

1. 安装java环境
2. 安装scoop：运行powershell输入：`iex (new-object net.webclient).downloadstring('https://get.scoop.sh')`
3. 安装allure：`scoop install allure`

RF集成allure

安装`allure-robotframework`：`pip install allure-robotframework`

1. 执行用例：`robot --listener allure_robotframework --outputdir ./output/robot ./`
    - 指定output目录下会生成两份数据：`allure`测试数据，用作生产allure报告；`robot`为rf自带的测试报告
2. 生成报告：`allure generate ./output/allure/ -o ./allure-report --clean`

RF自带关键字对比

	Robot	**python**
	Set variable	=
	Catenate	+
	Catenate SEPARATOR=,	','.join()
	get time	time.strftime("%Y-%m-%d %H:%M:%S")
	sleep	sleep
	Evaluate	python代码

注意细节(坑)

1. 通过RF直接传参数时，会以字符串传入
2. 列表变量要用`@{list}`才可以引用