
yamipy

发布 1.1.8

yangjianliang

2021 年 03 月 09 日

Contents

1	yamipy	1
2	工程主页	3
3	一、思路	5
4	二、目录结构	7
5	三、yaml、json 文件说明	9
6	四、运行	13
7	五、打包镜像，运行容器	15
8	索引和表格	17

CHAPTER 1

yamipy

yamipy 接口测试框架

readthedocs: https://yammpy-docs.readthedocs.io/zh_CN/latest/pypi: <https://pypi.org/project/yammpy/github>:
<https://github.com/yjlch1016/yammpy>

yammpy 即为 yaml 文件 +pytest 单元测试框架的缩写可看作是一个脚手架工具可快速生成项目的各个目录与文件支持 MySQL、PgSQL 与 MongoDB 等数据库的增删改查支持 Jenkins、Docker 等 CI/CD 工具支持飞书、钉钉、企业微信等机器人只需维护一份或者多份 yaml (或者 json) 文件即可

与yammlapi 接口测试框架对比, 整体结构仍然保持不变, yaml 文件 (或者 json 文件) 格式仍然保持不变, 可以通用, 抛弃了 python 自带的 unittest 单元测试框架、ddt 数据驱动第三方库、BeautifulReport 测试报告第三方库, 修改了测试类文件, 传参方式由 ddt 的 @ddt.ddt、@ddt.data() 改为 pytest 的 @pytest.mark.parametrize(), 删掉了 tool 工具包里面的 beautiful_report_run.py 文件, 其他文件保持不变。

```
pip install yammpy 安装
```

```
yammpy -h (或 yammpy --help) 查看参数信息
```

```
yammpy -v (或 yammpy --version) 查看版本号
```

```
pip install -U yammpy 安装最新版
```

```
yammpy create --p= 项目名称 创建项目例如在某个路径下执行命令: yammpy create --p=demo_project
```

```
yammpy run --c= 环境缩写运行项目例如在项目的根目录下面执行命令: yammpy run --c=test
```

```
yammpy clean 清理测试报告与日志目录下的所有文件类似于 mvn clean
```

```
pip uninstall yammpy 卸载
```

一、思路

1、 采 用 requests+PyMySQL+DBUtils+psycopg2-binary+pymongo+influxdb+demjson+loguru+PyYAML+ruamel.yaml+pytest+pytest-html+allure-pytest+pytest-reportlog+pytest-assume+pytest-rerunfailures+pytest-instafail+pytest-sugar+pytest-timeout+pytest-parallel+tablib2、requests 是发起 HTTP 请求的第三方库 3、PyMySQL 是连接 MySQL 的第三方库 4、DBUtils 是数据库连接池的第三方库 5、psycopg2-binary 是连接 PostgreSQL 的第三方库 6、pymongo 是连接 Mongo 的第三方库 7、influxdb 是连接 influxDB 的第三方库 8、demjson 是解析非标格式 json 的第三方库 9、loguru 是记录日志的第三方库 10、PyYAML 与 ruamel.yaml 是读写 yaml 文件的第三方库 11、pytest 是单元测试的第三方库 12、pytest-html 是生成 html 测试报告的插件 13、allure-pytest 是生成 allure 测试报告的插件 14、pytest-reportlog 是替换 -resultlog 选项的插件 15、pytest-assume 是多重断言的插件 16、pytest-rerunfailures 是失败重跑的插件 17、pytest-instafail 是实时显示错误信息的插件 18、pytest-sugar 是显示进度的插件 19、pytest-timeout 是设置超时时间的插件 20、pytest-parallel 是多线程的插件 21、tablib 是导出多种格式数据的第三方库

CHAPTER 4

二、目录结构

1、case 是测试用例包 2、report_log 是测试报告和日志的目录 3、resource 是 yaml（或者 json 文件）文件的目录 4、setting 是工程的配置文件包 5、tool 是常用方法的封装包 6、.dockerignore 是在传递给 docker 引擎时需要忽略掉的文件 7、.gitignore 是 .ignore 插件需要排除的文件 8、conftest.py 是全局钩子文件 9、Dockerfile 是构建镜像的文件 10、Jenkinsfile 是 Jenkins Pipeline 文件 11、pytest.ini 是 pytest 的配置文件 12、requirements.txt 是第三方依赖库

三、yaml、json 文件说明

yaml 文件

```
- case_name: 用例名称
  step:
    - step_name: 步骤名称
      mysql:
        -
        -
        -
      pgsql:
        -
        -
        -
      mongo:
        -
        -
        -
      request_mode: POST
      api: /api/test
      file:
        -
        -
        -
      body:
```

(下页继续)

```

    {"key_1":"value_1","key_2":"value_2"}
  headers:
    {"Content-Type":"application/json"}
  query_string:
    {"key_3":"value_3","key_4":"value_4"}
  expected_time: 3
  expected_code: 200
  expected_result:
    {"code":1,"message":"成功"}
  regular:
    variable:
      - name_1
      - name_2
    expression:
      - '"response_1": "(.+)"'
      - '"response_2": "(.+)"'

```

json 文件

```

[
  {
    "case_name": " 用例名称",
    "step": [
      {
        "step_name": " 步骤名称",
        "mysql": [],
        "pgsql": [],
        "mongo": [],
        "request_mode": "POST",
        "api": "/api/test",
        "file": [],
        "body": "{\"key_1\":\"value_1\",\"key_2\":\"value_2\"}",
        "headers": "{\"Content-Type\": 'application/json'}",
        "query_string": "{\"key_3': 'value_3', 'key_4': 'value_4'}",
        "expected_time": 3,
        "expected_code": 200,
        "expected_result": "{\"code\":1, \"message\": \" 成功\"}",
        "regular": {
          "variable": [
            "name_1",
            "name_2"
          ],
          "expression": [
            "\"response_1\": \"(.+)\"",

```

四、运行

1、pytest 模式: `pytest+cmd=` 环境缩写 `pytest --cmd=dev` 开发环境 `pytest --cmd=test` 测试环境
`pytest --cmd=pre` 预生产环境 `pytest --cmd=formal` 生产环境

2、yamipy 模式: `yamipy+run+-c=` 环境缩写 `yamipy run --c=dev` 开发环境 `yamipy run --c=test` 测试环境
`yamipy run --c=pre` 预生产环境 `yamipy run --c=formal` 生产环境

3、运行结果: 会在 `report_log` 目录下生成以下文件 `allure-reportlog` 年月日
`.logreport.htmlreport.xmltest_case.csvtest_case.htmltest_case.jsontest_case.xlsxtest_case.yaml`

五、打包镜像，运行容器

`docker pull registry.cn-hangzhou.aliyuncs.com/yangjianliang/yamlapi:0.0.8` 从阿里云镜像仓库拉取 `yamlapi` 镜像

`docker build -t demo_image .` `docker build -t 镜像名称.本地打包`，`demo_image` 为镜像名称，随便取

`docker run -e cmd=" 环境缩写" 镜像名称:latest` `docker run -e cmd=" dev" demo_image:latest` 开发环境 `docker run -e cmd=" test" demo_image:latest` 测试环境 `docker run -e cmd=" pre" demo_image:latest` 预生产环境 `docker run -e cmd=" formal" demo_image:latest` 生产环境

CHAPTER 8

索引和表格

- `genindex`
- `modindex`
- `search`