pithy-test Documentation

Release 0.0.1

coolfish

Jul 03, 2020

Contents

CHAPTER 1

:

1.
2. http client
3. thrift client
4.
5. json
6.

CHAPTER 2

&

,pithy-test.

2.1 pip

pip:

```
>>> pip install pithy-test
>>> pip install -U pithy-test
```

2.2

```
>>> pip uninstall pithy-test
```

2.3

```
>>> git clone https://github.com/yuyu1987/pithy-test.git
```

CHAPTER 3

3.1

3.1.1

```
>>> pithy-cli
Usage: pithy-cli [OPTIONS] COMMAND [ARGS]...

pithy-cli,

Options:
--version  Show the version and exit.
--help     Show this message and exit.

Commands:
init   : $ pithy-cli init #
```

3.1.2

```
>>> pithy-cli init
,apiapp: api
,pithy-api-test: pithy-api-test
pithy-api-test
...
 api/.gitignore                  []
 api/apis/__init__.py            []
 api/apis/pithy_api.py           []
 api/cfg.yaml                    []
 api/db/__init__.py              []
 api/db/pithy_db.py              []
 api/README.MD                   []
 api/requirements.txt            []
```

(continues on next page)

(continued from previous page)

```
 api/test_suites/__init__.py      []
 api/test_suites/test_login.py    []
 api/utils/__init__.py            []
,
```

Attention: app

3.1.3

```
>>> tree pithy-api-test
pithy-api-test
├── README.MD
├── apis
│   ├── __init__.py
│   └── pithy_api.py
├── cfg.yaml
├── db
│   ├── __init__.py
│   └── pithy_db.py
├── requirements.txt
├── test_suites
│   ├── __init__.py
│   └── test_login.py
└── utils
    └── __init__.py

4 directories, 10 files
```

3.2 HTTP API

python requestsapi

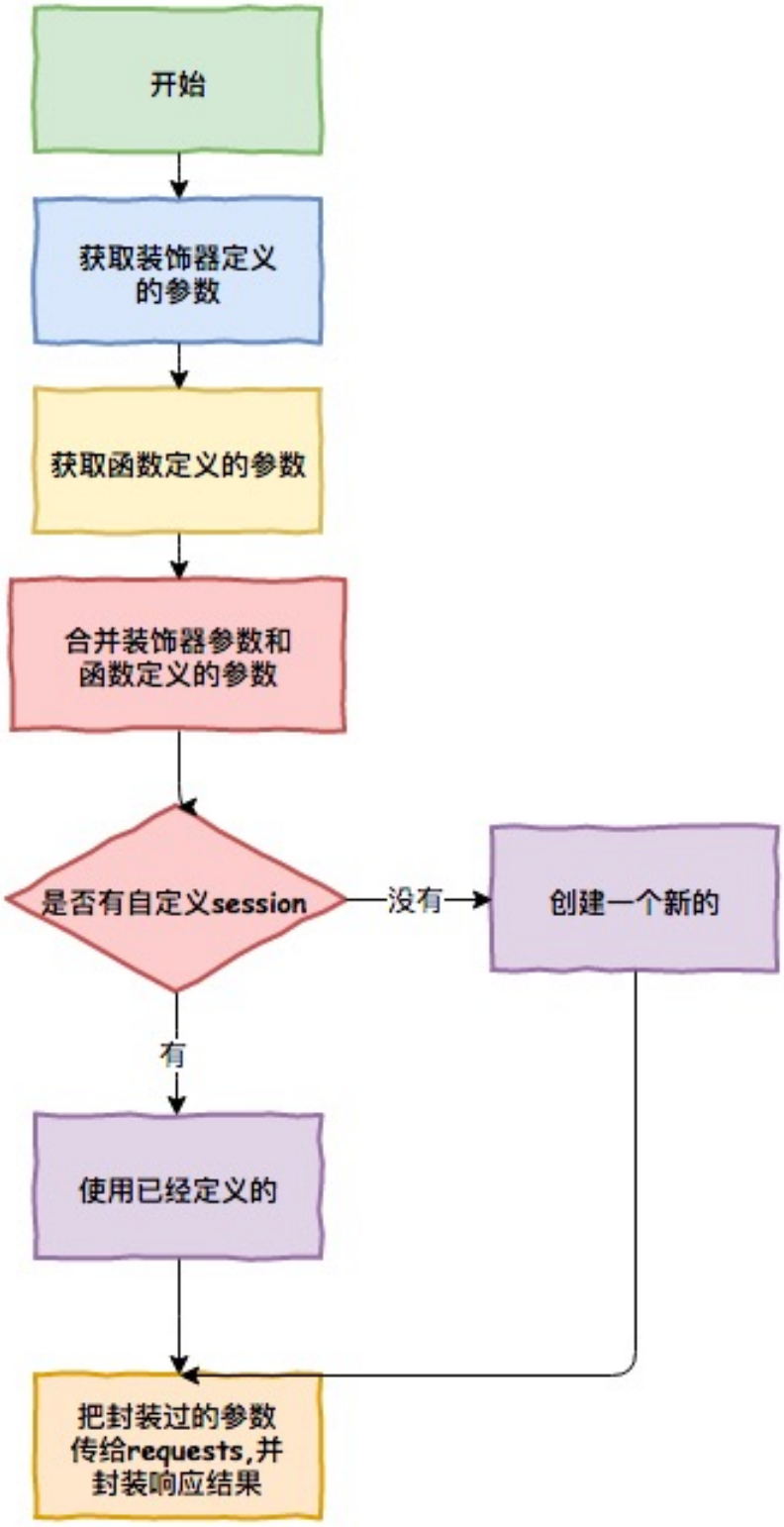
```
import requests
from pithy import request

# requetsapi
data = {'key': 'value'}
requests.get('http://www.xxx.com', data=data)


# request
@request(url='http://www.xxx.com')
def get(value):
    data = {'key': value}
    return {'data': data}
```

,api,,session

:


开始
获取装饰器定义的参数
获取函数定义的参数
合并装饰器参数和函数定义的参数
是否有自定义session
没有
创建一个新的
有
使用已经定义的
把封装过的参数传给requests,并封装响应结果

3.2.1 POST

```
from pithy import request

@request(url='http://httpbin.org/post', method='post')
def post(self, key1='value1'):
    """
    post method
    """
    data = {
        'key1': key1
    }
    return dict(data=data)

#
response = post('test').to_json()      # json,
response = post('test').json           # json,
response = post('test').to_content()   #
response = post('test').content        #
response = post('test').get_cookie()   # cookie
response = post('test').cookie         # cookie

# , response = {'a': 1, 'b': { 'c': [1, 2, 3, 4]}}
response = post('13111111111', '123abc').json

print response.b.c    # ,[1, 2, 3, 4]

print response('$.a') # object path,1

for i in response('$..c[@>3]'): # object path,c3
    print i
```

3.2.2 POSTJSON

```
from pithy import request

@request(url='http://httpbin.org/post', method='post')
def post(self, key1='value1'):
    """
    post method
    """
    data = {
        'key1': key1
    }
    return dict(json=data)
```

3.2.3 GET,URL

```
from pithy import request

@request(url='http://httpbin.org//get')
def get(self, key1='value1', key2=None):
    """
```

(continues on next page)

(continued from previous page)

```
    get method
    """
    if key2 is None:
        key2 = ['value2', 'value3']

    params = {
        'key1': key1,
        'key2': key2
    }
    return dict(params=params)
```

3.2.4

session,base_url

```
from pithy import request

class PithyAPP(object):

    def __init__(self):
        self.base_url = 'http://httpbin.org

    @request(url='/get')
    def get(self, key1='value1', key2=None):
        """
        get method
        """
        if key2 is None:
            key2 = ['value2', 'value3']

        params = {
            'key1': key1,
            'key2': key2
        }
        return dict(params=params)

    @request(url='post', method='post')
    def post(self, key1='value1'):
        """
        post method
        """
        data = {
            'key1': key1
        }
        return dict(data=data)

    @request(url='post', method='post')
    def json(self, key1='value1'):
        """
        post method
        """
        data = {
            'key1': key1
        }
        return dict(json=data)
```

(continues on next page)

(continued from previous page)

```
    @request(url='login', method='post')
    def _login(username, password):
        """
        api
        : ,,
        """
        data = {
            'username': username,
            'password': password
        }
        return dict(data=data)

    def login(username, password):
        """

        : ,,
        """
        req = self._login(username, password)
        cookies = res.cookies  # cookies
        headers = res.headers  # headers
        self.session.headers.update(xxx=headers.get('xxx')) # sessionheaders,,
        self.session.cookies.set('xxx', cookies.get('xxx')) # sessioncookies,,

# request session
app = PithyAPP()
app.get('value1').to_json()
app.post('value1).to_json()
```

CHAPTER 4

4.1

cfg.yaml file_name .ini .yaml .cfg .conf

4.1.1 yaml

cfg.yaml:

```
db:
    pithy_db:
        host: 127.0.0.1
        port: 3306
        username: user
        password: 111111
```

:

```
from pithy import config_manager

config = config_manager()
print(config.db)
print(config.db.pithy_db)
```

4.1.2 ini

cfg.ini:

```
[pithy_db]

host = 127.0.0.1
port = 5432
```

(continues on next page)

(continued from previous page)

```
username = postgres
password = postgres
```

:

```
from pithy import config_manager

config = config_manager(file_name='cfg.ini')
print(config.db.pithy_db)
```

confcfgini

4.2

4.2.1

```
from pithy import HumanDateTime
print(HumanDateTime())
```

4.2.2

:

```
from pithy import HumanDateTime

#
print(repr(HumanDateTime(1490842267)))
print(HumanDateTime(1490842267000))
print(HumanDateTime(1490842267.11111))
print(HumanDateTime(1490842267111.01))
```

4.2.3

:

```
from pithy import HumanDateTime

#
print(HumanDateTime('2017-02-02'))
print(HumanDateTime('Thu Mar 30 14:21:20 2017'))
print(HumanDateTime(time.ctime()))
print(HumanDateTime('2017-3-3'))
print(HumanDateTime('3/3/2016'))
print(HumanDateTime('2017-02-02 00:00:00'))
```

4.2.4 datetimedate

```
from pithy import HumanDateTime

# datetimedate
print(HumanDateTime(datetime(year=2018, month=11, day=30, hour=11)))
print(HumanDateTime(date(year=2018, month=11, day=30)))
```

4.2.5

```
from pithy import HumanDateTime
#
print(HumanDateTime('2017-02-02').add_day(1))
print(HumanDateTime('2017-02-02').sub_day(1))
print(HumanDateTime('2017-02-02').add_hour(1))
print(HumanDateTime('2017-02-02').sub_hour(1))
print(HumanDateTime('2017-02-02').add(days=1, hours=1, weeks=1, minutes=1, seconds=6))
print(HumanDateTime('2017-02-02').sub(days=1, hours=1, weeks=1, minutes=1, seconds=6))
```

4.2.6

:

```
from pithy import HumanDateTime

#
print(HumanDateTime(1490842267.11111).timestamp_second)
print(HumanDateTime(1490842267.11111).timestamp_microsecond)
print(HumanDateTime('2017-02-02 12:12:12.1111').add_day(1).timestamp_microsecond)
print(HumanDateTime('2017-02-02 12:12:12 1111').add_day(1).timestamp_microsecond)
```

4.2.7

:

```
from pithy import HumanDateTime

#
print(HumanDateTime('2017-02-02 12:12:12 1111') < HumanDateTime('2017-02-02 12:12:11␣
↪1111'))
print(HumanDateTime('2017-02-02 12:12:12 1111') < HumanDateTime('2017-02-02 12:13:11␣
↪1111'))
print(HumanDateTime('2017-02-02 12:12:12 1111') < '2017-02-02 12:11:11')
print(HumanDateTime('2017-02-02 12:12:12 1111') < '2017-02-02 12:13:11 1111')
print(HumanDateTime('2017-02-02 12:12:12 1111') == '2017-02-02 12:13:11 1111')
print(HumanDateTime('2017-02-02 12:12:12 1111') == '2017-02-02 12:13:12 1111')
print(HumanDateTime('2017-02-02 12:12:12 1111') <= '2017-02-02 12:13:11 1111')
print(HumanDateTime('2017-02-02 12:12:12 1111') >= '2017-02-02 12:13:11 1111')
print(HumanDateTime('2017-02-02 12:12:12 1111') != time.time())
print(HumanDateTime('2017-02-02 12:12:12 1111') <= time.time())
print(HumanDateTime('2017-02-02 12:12:12 1111') >= time.time())

#
print(HumanDateTime('2017-02-02 12:12:12 1111').approach('2017-02-02 12:12:11 1111'))
```

(continues on next page)

(continued from previous page)

```
print(HumanDateTime('2017-02-02 12:12:12 1111').approach('2017-02-02 12:12:10 1111'))
print(HumanDateTime('2017-02-02 12:12:12 1111').approach('2017-02-02 12:12:10 1111',␣
↪offset=2))
print(HumanDateTime('2017-02-02 12:12:12 1111').approach('2017-02-02 12:12:14 1111',␣
↪offset=2))
```

4.2.8 datetime

```
from pithy import HumanDateTime

# datetime
print(HumanDateTime('2017-02-02 12:12:12 1111').day)
print(HumanDateTime('2017-02-02 12:12:12 1111').year)
print(HumanDateTime('2017-02-02 12:12:12 1111').second)
print(HumanDateTime('2017-02-02 12:12:12 1111').date())
```

4.3 JSON

JSONpythonjson object path

4.3.1 JSONKEY

```
from pithy import JSONProcessor
dict_data = {'a': 1, 'b': {'a': [1, 2, 3, 4]}}
json_data = json.dumps(dict_data)
result = JSONProcessor(json_data)
print result.a      # 1
print result.b.a    # [1, 2, 3, 4]
```

4.3.2 KEY

```
from pithy import JSONProcessor
dict_data = {'a': 1, 'b': {'a': [1, 2, 3, 4]}}
result = JSONProcessor(dict_data)
print result.a      # 1
print result.b.a    # [1, 2, 3, 4]
```

4.3.3 object path

object path,:http://objectpath.org/reference.html

```
from pithy import JSONProcessor
raw_dict = {
    'key1':{
        'key2':{
            'key3': [1, 2, 3, 4, 5, 6, 7, 8]
```

(continues on next page)

(continued from previous page)

```
        }
    }
}

jp = JSONProcessor(raw_dict)
for i in jp('$..key3[@>3]'):
    print i
```

:

```
4
5
6
7
8
```

4.3.4

```
dict_1 = {'a': 'a'}
json_1 = '{"b": "b"}'
jp = JSONProcessor(dict_1, json_1, c='c')
print(jp)
```

:

```
{
    "a": "a",
    "b": "b",
    "c": "c"
}
```

4.4 JSON

JSON JSONunicodeutf-8

:

```
from pithy import pretty_print
d = {
    "args": {
        "name": ""
    },
    "headers": {
        "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/
↪*;q=0.8",
        "Accept-Encoding": "gzip, deflate, sdch",
        "Accept-Language": "zh-CN,zh;q=0.8,en;q=0.6,zh-TW;q=0.4",
        "Connection": "close",
        "Cookie": "_gauges_unique_day=1; _gauges_unique_month=1; _gauges_unique_
↪year=1; _gauges_unique=1",
        "Dnt": "1",
        "Host": "httpbin.org",
```

(continues on next page)

(continued from previous page)

```
        "Upgrade-Insecure-Requests": "1",
        "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_3) AppleWebKit/
↪537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36"
    },
    "origin": "157.119.234.165",
    "url": "http://httpbin.org/get"
}
print(d)
print('\n\n')
pretty_print(d) # JSON
```

:

```
{'origin': '157.119.234.165', 'headers': {'Dnt': '1', 'Connection': 'close', 'Accept-
↪Language': 'zh-CN,zh;q=0.8,en;q=0.6,zh-TW;q=0.4', 'Accept-Encoding': 'gzip, deflate,
↪ sdch', 'Cookie': '_gauges_unique_day=1; _gauges_unique_month=1; _gauges_unique_
↪year=1; _gauges_unique=1', 'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_
↪12_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36',
↪'Host': 'httpbin.org', 'Accept': 'text/html,application/xhtml+xml,application/xml;
↪q=0.9,image/webp,*/*;q=0.8', 'Upgrade-Insecure-Requests': '1'}, 'args': {'name':
↪'\xe9\xb1\xbc\xe9\xb1\xbc'}, 'url': 'http://httpbin.org/get'}


{
    "args": {
        "name": ""
    },
    "headers": {
        "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/
↪*;q=0.8",
        "Accept-Encoding": "gzip, deflate, sdch",
        "Accept-Language": "zh-CN,zh;q=0.8,en;q=0.6,zh-TW;q=0.4",
        "Connection": "close",
        "Cookie": "_gauges_unique_day=1; _gauges_unique_month=1; _gauges_unique_
↪year=1; _gauges_unique=1",
        "Dnt": "1",
        "Host": "httpbin.org",
        "Upgrade-Insecure-Requests": "1",
        "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_3) AppleWebKit/
↪537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36"
    },
    "origin": "157.119.234.165",
    "url": "http://httpbin.org/get"
}
```