测试模块：用户注册 测试方法：条件组合覆盖测试人员：吴安之 试代码：test.py

## 1 测试代码

# -\*- coding: utf-8 -\*-

# @Time : 2018/1/8 22:15

# @Author : k\_holmes

# @Email : 31501324@stu.zucc.edu.cn

# @File : TestRegister.py

# @Software: PyCharm

import unittest

from django.http import HttpResponse

from se\_work.models import Customer

import os,django

os.environ.setdefault("DJANGO\_SETTINGS\_MODULE", "se2017\_django.settings")# project\_name 项目名称

django.setup()

class Register(unittest.TestCase):

def setUp(self):

pass

def register(self, user\_id, passwd, passwd\_confirm):

# products = Products.objects()

user\_id = user\_id

user\_passwd = passwd

user\_confirm\_passwd = passwd\_confirm

if len(user\_id) == 0:

return '账号不能为空'

elif len(user\_passwd) == 0 or len(user\_confirm\_passwd) == 0:

return '密码不能为空'

elif user\_passwd != user\_confirm\_passwd:

return "两次密码不一致"

elif len(user\_id) > 20 or len(user\_id) < 6:

return '请输入一个账号长度在6至20位'

elif len(user\_passwd) > 20 or len(user\_passwd) < 6:

return '请输入一个密码长度在6至20位'

elif len(Customer.objects(\_id=user\_id)) != 0:

return HttpResponse('账户已存在')

else:

user\_registed = Customer(\_id=user\_id, password=user\_passwd, sex="男")

user\_registed.save()

# login\_msg = '欢迎:' + request.session['user\_id']

return "注册成功"

def tearDown(self):

return "测试后成功"

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

## 2. 测试数据

def test0(self):

self.assertEqual(self.register("", "11111111", "1111111111"), "账号不能为空")

def test1(self):

self.assertEqual(self.register("123456", "11111111", "13123132131241242132asdsadsasad3"), "两次密码不一致")

self.assertEqual(self.register("123456", "11111112", "1111111111"), "两次密码不一致")

def test2(self):

self.assertEqual(self.register("123456", "", "1111111111"), "密码不能为空")

self.assertEqual(self.register("123456", "1111111111", ""), "密码不能为空")

def test3(self):

self.assertEqual(self.register("cdk", "123456", "123456"), "请输入一个账号长度在6至20位")

self.assertEqual(self.register("cdk2132141241312asdsadsadsadadasdzcdsa", "123456", "123456"), "请输入一个账号长度在6至20位")

def test4(self):

self.assertEqual(self.register("cdk123", "12", "12"), "请输入一个密码长度在6至20位")

self.assertEqual(self.register("cdk123", "1212312421321421321asdsadsad32142141eqe", "1212312421321421321asdsadsad32142141eqe"),

"请输入一个密码长度在6至20位")

def test5(self):

self.assertEqual(self.register("", "", ""), "账号不能为空")

self.assertEqual(self.register("", "asd", ""), "账号不能为空")

self.assertEqual(self.register("", "", "asd"), "账号不能为空")

self.assertEqual(self.register("", "asde", "asd"), "账号不能为空")

self.assertEqual(self.register("", "asde", "asdee"), "账号不能为空")

def test6(self):

self.assertEqual(self.register("cdkcdk123", "12345612312321312414124213132", "12345612312321312414124213132"), "请输入一个密码长度在6至20位")

self.assertEqual(self.register("cdkcdk123", "1234", "1234"), "请输入一个密码长度在6至20位")

def test7(self):

self.assertEqual(self.register("123", "", ""), "密码不能为空")

self.assertEqual(self.register("123131sadasdadsadhjkfafsksad", "", ""), "密码不能为空")

### 代码说明

在程序结构中，首先测试单一情况下对判断结果的正确与否，如代码中，先设置数据字段中账号为空时，密码满足要求的情况下的程序返回，查看返回结果是否是预期。如此连续做几类不同的单一条件的判断。查看运行结果，然后讲这些单一条件判断组合起来丢入程序去跑，看是否依旧符合预期，若符合则满足要求，符合期望内容。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例名称 | 用户注册 | | | |
| 测试目的 | 注册后的用户可以进行账户登录 | | | |
| 测试方法 | 条件组合覆盖 | | | |
| 测试场景 | 测试数据 | 预期结果 | 实际测试结果 | |
| 场景一： | 用户名： | 显示“账号不能为空” | 与预期结果相同 | |
| 账号为空 | 密码：123456 |
|  | 123456 |
| 用户名： | 显示“账号不能为空” | 与预期结果相同 | |
| 密码：12345 |
| 123456 |
| 场景三： | 用户名： llcasd | 显示“密码不能为空” | 与预期结果相同 | |
| 密码为空 | 密码 : “” |
|  | 123123 |
| 用户名：llcasd  密码:123123  “” | 显示“密码不能为空” | 与预期结果相同 | |
| 场景四  输入账号长度小于6位或者大于20位 | 用户名： asd  密码:1231232321  1231232321 | 显示“请输入的账号长度在6至20位” | 与预期结果相同 | |
| 用户名:  Asdsadsadwkjqewqejqwhehw  密码 : 123456  123456 | 显示“请输入的账号长度在6至20位 | 与预期结果相同 | |
| 场景五  输入的密码长度在小于六位或者大于20位 | 用户名: cdk123  密码: 123  123 | 显示“请输入的密码长度在6至20位” | 与预期结果相同 | |
|  | 用户名: cdk123  密码:  12345678901234567890134 12345678901234567890134 | 显示“请输入的密码长度在6至20位” | 与预期结果相同 | |
| 场景六:  输入正确 | 用户名: cdk123  密码: 123456  123456 | 显示:”注册成功 | 与预期结果相同 | |
| 实验七：  输入已注册账户 | 用户名: cdk123  密码: 1234567  1234567 | 显示”账户已存在” | 与预期结果相同 | |
| 场景  输入账号为空，两次密码不一致 | 用户名: “”  密码: 1234567  123456 | 显示“账号不能为空 | 与预期相同 | |
| 场景八  输入账号为空，密码为空 | 用户名: “”  密码: “”  “” | 显示账号不能为空 | 与预期相同 | |
| 场景九  输入账号小于6于20位  密码不一致 | 用户名: cdk  密码: “asdasdad  “1234124 | 显示”请输入一个账号长度在6至20位 | 与预期相同 | |
| 用户名:  cdk1231242142132144211  密码: “asdasdad  “1234124 | 显示”请输入一个账号长度在6至20位 | 与预期相同 | |
| 场景十  输入密码小于6位或20位  账户已存在 | 用户名: cdk123  密码: “asd  “123 | 显示”请输入一个密码长度在6至20位 | 与预期相同 | |
| 用户名: cdk123  密码:  asdasdsadasdsadwqesadasd  asdasdsadasdsadwqesadasd  “ | 显示”请输入一个密码长度在6至20位 | 与预期相同 | |
| 测试结论 | | | | |
| 2状态 | 正常 | | | |
| 错误说明 | 无 | | | |
| 错误重现说明 | 无 | | | |
| 错误重现 | 无 | | | |
| 测试人员 | 吴安之 | | 测试时间 | 2018/1/8 |

## 3. 测试结果

 



