

像风一样的自由

人生苦短，我也用python!!!

目录视图 摘要视图 RSS 订阅

个人资料



像风一样的自由

访问： 800493次
积分： 8032
等级：
排名： 第1532名

原创： 142篇 转载： 25篇
译文： 0篇 评论： 111条

文章搜索

文章分类

- python (29)
- 性能 (5)
- 测试 (6)
- 网络 (3)
- 自动化 (15)
- java (15)
- C/C++ (4)
- 白盒 (1)
- python应用 (43)
- python技巧 (12)
- 其它 (18)
- ruby (8)
- python django (2)
- 数据库 (4)
- qt (3)
- mysql (7)
- android (1)
- watir (3)
- linux (5)
- svn (2)
- selenium (15)
- chrome (1)
- html (5)
- hadoop (1)

【免费公开课】Gulp前端自动化教程 走进VR开发世界——我们离开发一款VR大作还有多远？ CSDN发福利啦！C币、京东卡、现金任你选

python单元测试unittest

标签： 单元测试 python 测试 module import 工作

2011-12-26 23:41 44453人阅读 评论(7) 收藏 举报

分类： python应用 (42)

版权声明：本文为博主原创文章，未经博主允许不得转载。

单元测试作为任何语言的开发者都应该是必要的，因为时隔数月后再回来调试自己的复杂程序时，其实也是很崩溃的事情。虽然会很快熟悉内容，但是修改和调试将是一件痛苦的事情，如果你在修改了代码后出现问题的话，而单元测试可以帮助我们很快准确的定位到问题的位置，出现问题的模块和单元。所以这是一件很愉快的事情，因为我们知道其它修改的地方仍然是正常工作的，而我们目前的唯一问题就是搞定眼前这个有点问题的“家伙”。所以工作会在轻松中开始，并且很快将会结束，因为你已经知道很多信息了。

单元测试自然是对程序中最小的可测试模块--函数来进行测试；因为单元测试的对象是函数，也就是说你得被测试对象一定要有输出结果，哪怕就是异常输出，也得有输出，以便单元测试模块能够捕获返回值，并且与预期值进行比较，从而得出测试通过与否。

单元测试的加载方式有2种：一种是通过unittest.main()来启动单元测试的测试模块；一种是添加到testsuite集合中再加载所有的被测试对象，而testsuit里存放的就是单元测试的用例，下面分别列出了2种方法的使用。

1.1 测试模块中的函数：

被测模块：

```
[python]
01.  #!/usr/bin/env python
02.  #encoding: utf-8
03.
04.  def sum( x, y):
05.      return x+y
06.
07.
08.  def sub( x, y):
09.      return x-y
```

单元测试模块：

```
[python]
01.  #!/usr/bin/env python
02.  #encoding: utf-8
03.
04.  import unittest
05.  import myclass
06.
07.  class mytest(unittest.TestCase):
08.
09.      ##初始化工作
10.      def setUp(self):
11.          pass
```

seo (1)

文章存档

2016年03月 (3)

2016年02月 (2)

2015年11月 (2)

2015年06月 (2)

2015年01月 (1)

展开

阅读排行

HTTP协议之multipart/form-data (75202)

python--httplib模块使用 (57494)

python单元测试unittest (44406)

Linux下使用git命令及git (30689)

python操作excel之xlrd (23939)

python中telnetlib模块的用法 (23538)

httplib2--python下的http (21368)

import,reload,__import__ (19409)

putty工具命令行参数 (18792)

Selenium + python的自动化测试 (18371)

评论排行

Selenium2.0中文在线文档 (10)

QTP10-插件破解脚本 (7)

python单元测试unittest (7)

python操作excel之xlrd (6)

HTTP协议之multipart/form-data (6)

python中telnetlib模块的用法 (6)

webpy中使用session (6)

httplib2--python下的http (5)

如何使用Selenium-Grid (5)

nginx+spawn-fcgi+flup (5)

推荐文章

*EventBus的使用与深入学习

*Android 拍照、选择图片并裁剪

*spark性能调优：开发调优

*浅谈android中图片处理之色彩特效处理ColorMatrix(三)

*neutron-server的启动流程(一)

*Hadoop中Map端shuffle源码解析

最新评论

webpy中使用session
vimer25: session中可以设置timeout时间，怎么让每次用户连接，就重新计时？

webpy中使用session
vimer25: session中可以设置timeout时间，怎么让每次用户连接，就重新计时？

关于HOSTS设置不生效的解决小
happyboygd: 注意格式是先ip地址，然后域名1.1.1.1
www.test.com顺序写反是无效的：)

Ubuntu搭建subversion+svn环境
Op小剑: 你好，一直在看你的

```
12.
13.     #退出清理工作
14.     def tearDown(self):
15.         pass
16.
17.     #具体的测试用例，一定要以test开头
18.     def testsum(self):
19.         self.assertEqual(myclass.sum(1, 2), 2, 'test sum fail')
20.
21.
22.     def testsub(self):
23.         self.assertEqual(myclass.sub(2, 1), 1, 'test sub fail')
24.
25.
26. if __name__ == '__main__':
27.     unittest.main()
```

测试结果：【F表示一个fail， F前的点表示一个通过，有E的话表示程序自身异常】

```
[python]
01. .F
02. =====
03. FAIL: testsum (__main__.mytest)
04. -----
05. Traceback (most recent call last):
06.   File "C:\Users\xiaowu\workspace\mypython\unitTest.py", line 19, in testsum
07.     self.assertEqual(myclass.sum(1, 2), 2, 'test sum fail')
08. AssertionError: test sum fail
09.
10. -----
11. Ran 2 tests in 0.001s
12.
13. FAILED (failures=1)
```

1.2 测试模块类中的函数：

被测模块：

```
[python]
01. #!/usr/bin/env python
02. #encoding: utf-8
03.
04. class myclass:
05.     def __init__(self):
06.         pass
07.
08.
09.     def sum(self, x, y):
10.         return x+y
11.
12.
13.     def sub(self, x, y):
14.         return x-y
```

单元测试模块：

```
[python]
01. #!/usr/bin/env python
02. #encoding: utf-8
03.
04. import unittest
05. import myclass
06.
07. class mytest(unittest.TestCase):
08.
09.     ##初始化工作
10.     def setUp(self):
11.         self.tclass = myclass.myclass() ##实例化了被测测试模块中的类
12.
13.     #退出清理工作
14.     def tearDown(self):
15.         pass
16.
17.     #具体的测试用例，一定要以test开头
18.     def testsum(self):
19.         self.assertEqual(self.tclass.sum(1, 2), 3)
```

blog, 最近没什么更新啊。还有可以加你的QQ吗? 想问一些关于selenium的问...

python解析xml模块
longhua2014: good

python中telnetlib模块的使用
u014626129: 请问下这段代码往下, 如果想判断telnet不上Linux服务器, 怎样反馈出Linux返回的信息, 比如...

JIRA6.3安装及alige插件破解
qq970131158: 哪里能够下载到jira的安装包, 好像官网现在下载不了, 而且访问速度很慢

webpy中使用session
abaoabao: 上帝De助手?正在学习, 谢谢!

webpy中使用session
abaoabao: python web框架#6正在学习, 谢谢!

Robot framework中支持360浏览
luck1006: 已经搞定了, 谢谢楼主, 是我在贴代码时没有注意空格

```
20.  
21.  
22.     if __name__ == '__main__':  
23.         unittest.main()
```

运行结果:

```
[python]  
  
01. .  
02. -----  
03. Ran 1 test in 0.000s  
04.  
05. OK
```

这种方式执行单个测试文件时使用-v参数可以获得更多的测试结果信息。如: mytest.py -v

2 加载测试套件

好吧, 在运用测试套件进行单元测试之前, 我想还是稍微研究一下unittest模块的内容有哪些, 其大概的运行方式是什么样的。而后在给出根据各种情况如何制定单元测试套件。

首先, 自然是查看unittest模块有哪些成员啦!

```
[python]  
  
01. >>import unittest  
02. >>dir(unittest)  
03. ['FunctionTestCase', 'TestCase', 'TestLoader', 'TestProgram', 'TestResult', 'TestSuite', 'TextTestRunner', '_CmpToKey', '_TextTestResult', '_WriteInDecorator',  
04. '_all_', '_author_', '_builtins_', '_doc_', '_email_', '_file_', '_metaclass_', '_name_', '_package_', '_unittest', '_version_', '_makeLoader', '_strclass', 'defaultTestLoader', 'findTestCases', 'getTestCaseNames', 'main', 'makeSuite', 'os', 'sys', 'time', 'traceback', 'types']  
08.
```

可以看到其自身的成员也不是很多, 大概包括有:

```
['FunctionTestCase', 'TestCase', 'TestLoader', 'TestProgram', 'TestResult',  
'TestSuite', 'TextTestRunner', '_CmpToKey', '_TextTestResult', '_WriteInDecorator',  
'defaultTestLoader', 'findTestCases', 'getTestCaseNames', 'main', 'makeSuite']
```

好吧, 我们大概看看具体都是干什么的

```
[python]  
  
01. >>memblist = ['FunctionTestCase', 'TestCase', 'TestLoader', 'TestProgram', 'TestResult', \  
02. 'TestSuite', 'TextTestRunner', 'defaultTestLoader', 'findTestCases', 'getTestCaseNames', \  
03. 'main', 'makeSuite']  
04. >>for memb in memblist:  
05.     .. cur = getattr(unittest, memb)  
06.     .. print help(cur)
```

'FunctionTestCase': 函数测试用例, 即给一个函数作为参数, 返回一个testcase实例, 可选参数有set-up, tear-down方法

'TestCase': 所有测试用例的基本类, 给一个测试方法的名字, 返回一个测试用例实例

'TestLoader': 测试用例加载器, 其包括多个加载测试用例的方法。返回一个测试套件

loadTestsFromModule(self, module)--根据给定的模块实例来获取测试用例套件

loadTestsFromName(self, name, module=None)

--根据给定的字符串来获取测试用例套件, 字符串可以是模块名, 测试类名, 测试类中的测试方法名, 或者一个可调用的是实例对象

这个实例对象返回一个测试用例或一个测试套件

loadTestsFromNames(self, names, module=None) --和上面功能相同, 只不过接受的是字符串列表

loadTestsFromTestCase(self, testCaseClass)--根据给定的测试类, 获取其中的所有测试方法, 并返回一个测试套件

'TestProgram': 命令行进行单元测试的调用方法, 作用是执行一个测试用例。其实unittest.main()方法执行的就是这个命令,

而这个类实例时默认加载当前执行的作为测试对象,

原型为 __init__(self, module='__main__', defaultTest=None, argv=None, testRunner=xx, testLoader=xx)

其中module='__main__'就是默认加载自身

'TestResult': 测试用例的结果保存实例, 通常有测试框架调用

'TestSuite': 组织测试用例的实例, 支持测试用例的添加和删除, 最终将传递给testRunner进行测试执行

'TextTestRunner': 进行测试用例执行的实例, 其中Text的意思是以文本形式显示测试结果。显示测试名称, 即完成的测试结果, 其过同执行单元测试脚本时添加-v参数

'defaultTestLoader': 其实就是TestLoader

'findTestCases', 'getTestCaseNames': 这个2个就不用解释了

'main': 其实就是TestProgram

'makeSuite': 通常是由单元测试框架调用的, 用于生产testsuite对象的实例

至此, 我们知道了。其实整个单元测试框架的逻辑出来了, 分三步走: 第一步testloader根据传入的参数获得相应的测试用例, 即对应具体的测试方法,

然后makesuite在把所有的测试用例组装成testsuite, 最后把testsuite传给testrunner进行执行。

而我们通常执行的unittest.main(), 其实就是unittest.testprom方法, 其执行的功能就是上面分析的三步, 在第一步中其传入的参数是自身的模块__main__;

在第二步中把自身模块中的所有测试类中中的测试方法提取出来, 并生成测试套件; 最后再把测试套件传递给testrunner进行具体的测试。

最后给出一个完整的单元测试组织代码, 把该代码放到单元测试用例文件的同一个目录后执行该脚本, 即可执行所有的测试用例文件。

【测试用例文件必须为test开头, 如: testxxx.py, 当然这个文件本身是一个单元测试的文件】

```
[python]
01.  #!/usr/bin/env python
02.  #encoding: utf-8
03.  #该代码源自深入python
04.  import unittest
05.  import myclass
06.  import re
07.  import os
08.  import sys
09.
10.
11.  def testAllinCurrent():
12.      path = os.path.abspath(os.path.dirname(sys.argv[0]))
13.      files = os.listdir(path)
14.      test = re.compile("test\.[0-9]*\.py$", re.IGNORECASE)
15.      files = filter(test.search, files)
16.      filenameToModuleName = lambda f: os.path.splitext(f)[0]
17.      moduleNames = map(filenameToModuleName, files)
18.      modules = map(__import__, moduleNames)
19.
20.      load = unittest.defaultTestLoader.loadTestsFromModule
21.      return unittest.TestSuite(map(load, modules))
22.
23.  if __name__ == "__main__":
24.      unittest.main(defaultTest="regressionTest")
```

顶 踩
1 0

上一篇 【转载】C语言头文件的使用

下一篇 C语言交换两个变量值不利用额外变量

我的同类文章

python应用 (42)

• UEditor在线web编辑器 -...	2015-06-03	阅读 853	• 代码中加入如下内容，可...	2014-08-27	阅读 1517
• python实现tail -f命令功能	2014-08-12	阅读 4046	• python比较2个xml内容	2014-07-31	阅读 1667
• python发送邮件sendmail-...	2014-05-31	阅读 3847	• python循环遍历文件操作	2013-06-27	阅读 3579
• python实现测试脚本的关...	2013-06-08	阅读 2320	• webpy中配置发送邮件服务	2013-01-30	阅读 1365
• python抓网页资源小脚本	2012-11-19	阅读 1174	• python中telnetlib模块的使用	2012-10-22	
• python类型比较的3种方式	2012-10-22	阅读 8148	阅读 23489		

更多文章

参考知识库



Python 知识库
8368 关注 | 812 收录

猜你在找

- Python自动化开发实战视频课程-全新基础篇
软件测试基础
《C语言/C++学习指南》单步调试
Python 零基础到实战
微信公众平台开发入门
- Python单元测试框架-unittest
Python Unittest 自动化单元测试框架Demo
python-unittest模块单元测试
python 单元测试assert 或者 unittest.TestCase
Python 中 unittest单元测试的学习



柯基犬价格



陀飞轮



龙泉剑



在职研究生双



loft公寓出租



魔方公寓



整租一室

查看评论

- 7楼

[sinat_33540202](#)

2015-12-28 14:38发表



楼主还在吗？求解答
我还是不明白你写的第二段 2 加载测试套件 里面那个完整的单元测试组织代码 是怎么加载了测试套件的， 我复制了代码到python里运行 这两行都报错呢

```
test = re.compile("test\py{1}" , re.IGNORECASE)
unittest.main(defaultTest="regressionTest")
```
- 6楼

[sbcelso](#)

2015-10-19 17:17发表



内容详实，谢谢分享！！
- 5楼

[clover1107](#)

2015-09-08 17:47发表



楼主你好啊！看到你python方面的技术博客。我是麦子学院的讲师顾问，目前我们在建设这方面的课程，不知道你是否有兴趣合作视频课程呢？我的qq:2318019782 方便的话可以详谈一下，期待你的回复。
- 4楼

[白熊花田](#)

2015-07-20 10:39发表



Thks
- 3楼

[杰瑞26](#)

2014-01-20 22:36发表



好文章，简单易懂，收藏了。
- 2楼

[as](#)

2013-03-05 10:24发表



我按你给的例子写了一个测试脚本，
如果我在一个testCase子类中写了两个方法testa和testb, 其中testa的执行要依赖于testb产生的结果，那么我如何使testb在testa之前执行呢？ 好像unittest.main()默认按字母优先顺序执行。
- 1楼

[as](#)

2013-02-23 15:02发表



在测试你的第一个例子（测试模块类中的函数）时我除了你给出的报错信息之外还报出如下的信息，请问是什么原因？

```
[python]
01. Traceback (most recent call last):
02.   File "D:/Python Test/unit_test.py", line 26, in <module>
```

```
03.         unittest.main()
04.     File "D:\Python27\lib\unittest\main.py", line 95, in __init__
05.         self.runTests()
06.     File "D:\Python27\lib\unittest\main.py", line 231, in runTests
07.         sys.exit(not self.result.wasSuccessful())
08. SystemExit: True
```

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题	Hadoop	AWS	移动游戏	Java	Android	iOS	Swift	智能硬件	Docker	OpenStack		
VPN	Spark	ERP	IE10	Eclipse	CRM	JavaScript	数据库	Ubuntu	NFC	WAP	jQuery	
BI	HTML5	Spring	Apache	.NET	API	HTML	SDK	IIS	Fedora	XML	LBS	Unity
Splashtop	UML	components	Windows Mobile	Rails	QEMU	KDE	Cassandra	CloudStack				
FTC	coremail	OPhone	CouchBase	云计算	iOS6	Rackspace	Web App	SpringSide				
Maemo	Compuware	大数据	aptech	Perl	Tornado	Ruby	Hibernate	ThinkPHP	HBase			
Pure	Solr	Angular	Cloud Foundry	Redis	Scala	Django	Bootstrap					

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

网站客服 杂志客服 微博客服 webmaster@csdn.net 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 | 江苏乐知网络技术有限公司 提供商务支持
京 ICP 证 09002463 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved 