Python单测框架pytest详解

 2018-05-14T13:34:51     6430     0     0

pytest是Python最流程的单测框架之一。

在本文中，我们将会介绍pytest的特点，功能和使用。

pytest有哪些优点？

* 允许直接使用assert进行断言，而不需要使用self.assert\*;
* 可以自动寻找单测文件、类和函数;
* Modular fixtures可以用于管理小型或参数化的测试信息;
* 与unittest和nose单测框架兼容;
* 兼容性较好，支持Python 2.7，Python 3.4+。
* 丰富的插件支持，共计有超过315个插件支持;

Demo

在开始详解讲解pytest的使用之前，我们首先从一个简单的示例开始了解一下：   
准备一个文件test\_sample.py，内容如下：

1. def inc(x):
2. return x + 1
3. def test\_answer():
4. assert inc(3) == 5

在这个文件中，我们定义了一个inc自增函数，同时编写了一个单测函数test\_answer。   
下面，我们可以直接在命令行执行如下命令来运行单测用例：

1. pytest

运行完成后，可以得到如下结果：

1. =========================== test session starts ============================
2. platform linux -- Python 3.x.y, pytest-3.x.y, py-1.x.y, pluggy-0.x.y
3. rootdir: $REGENDOC\_TMPDIR, inifile:
4. collected 1 item
5. test\_sample.py F [100%]
6. ================================= FAILURES =================================
7. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ test\_answer \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
8. def test\_answer():
9. > assert inc(3) == 5
10. E assert 4 == 5
11. E + where 4 = inc(3)
12. test\_sample.py:5: AssertionError
13. ========================= 1 failed in 0.12 seconds =========================

怎么样？是不是非常简单呢？

使用说明

了解了pytest的基本使用后，我们详细学习一下pytest的使用方式。

安装

开始讲解的第一步，还是先来了解一下如何进行安装~   
pytest可以直接使用Python的包管理工具pip来进行安装。

1. pip install -U pytest

执行上述命令后，即可安装完成pytest。

编写单测用例

在pytest框架中，有如下约束：

* 所有的单测文件名都需要满足test\_\*.py格式或\*\_test.py格式。
* 在单测文件中，可以包含test\_开头的函数，也可以包含Test开头的类。
* 在单测类中，可以包含一个或多个test\_开头的函数。

此时，在执行pytest命令时，会自动从当前目录及子目录中寻找符合上述约束的测试函数来执行。

示例：test\_case1.py文件如下：

1. class TestClass(object):
2. def test\_one(self):
3. x = "this"
4. assert 'h' in x
5. def test\_two(self):
6. x = "hello"
7. assert hasattr(x, 'check')

兼容unittest与nose

如果之前你的单测全部是基于unittest或者nose来编写的。   
不用担心，执行pytest命令同样可以正常运行这些用例并得到结果。   
因此无需担心迁移单测框架从而带来额外的人工成本。

常用插件

在之前的内容中，我们提到了pytest的一个优点是有大量的插件支持，例如：

* pytest-django: 针对Django框架的单测框架
* pytest-twisted: 针对twisted框架的单测框架
* pytest-cov: 产生覆盖率报告
* pytest-instafail: 发送错误时报告错误信息
* pytest-bdd 测试驱动开发工具
* pytest-konira 测试驱动开发工具
* pytest-timeout: 支持超时功能
* pytest-pep8: 支持PEP8检查
* pytest-flakes: 结合pyflakes进行代码检查