FACULDADE SANTA TEREZINHA (CEST)
CURSO: SISTEMAS DE INFORMAÇÃO PERÍODO: 5º NOT.
DISCIPLINA: QUALIDADE DE SOFTWARE
PROFESSOR: WAGNER
ALIUNA: SILVILENE CALDAS CAMPOS

FERRAMENTA DE TESTE DE SOFTWARE: NOSETEST

O Nose é uma estrutura de automação de teste muito popular em Python que estende o unitest (teste unitário) para facilitar os testes. Foi lançado em 2005, um ano após o py.test ter recebido sua aparência moderna. Ele foi escrito por Jason Pellerin para suportar os mesmos idiomas de teste que foram lançados pelo py.test, mas em um pacote que é mais fácil de instalar e manter.

A estrutura Nose contém um conjunto de plugins que auxiliam na execução de testes, testes paralelos (ou multiprocessos), registro e relatórios dentre outros. Ele também pode executar doctests, unittests, bem como teste padrão. Os plugins também adicionam suporte para decoradores, fixtures, testes parametrizados, classes e módulos. Algumas vantagens de utilizar o framework Nose são a habilitação da descoberta automática de casos de teste e coleta de documentação.

A versão mais recente é o Nose2, no entanto, ainda existe uma grande quantidade na classe de desenvolvedores e testes que ainda utilizam a versão mais antiga do Nose, ou seja, a versão 1.3.7.

Instalação e teste

É instalado com a ajuda do utilitário ***pip***. Através do comando*: pip install nose*, que fará a instalação do módulo Nose na distribuição atual do Python, bem como um nosetest.exe, assim o teste pode ser executado usando o utilitário e também por meio da opção -m.

Exemplo:

C:\python>nosetests –v test_sample.py

ou

C:\python>python –m nose test_sample.py

O comando **- *nosetests* -** irá procurar automaticamente por quaisquer módulos (e pacotes) com testes se os nomes desses pacotes corresponderem a subpalavra “test”

O nose coleta testes das subclasses *unittest.TestCase*, escreve funções de teste simples, bem como classes de teste que não são subclasses de unittest.TestCase. O nose também fornece várias funções úteis para escrever testes cronometrados, testar exceções e outros casos de uso comuns.

Coleta testes automaticamente, ou seja, não há necessidade de coletar manualmente os casos de teste em suítes de teste. A execução de testes é responsiva, pois o nose começa a executar os testes assim que o primeiro módulo de teste é carregado.

Assim como no módulo de teste unitário, o nose suporta acessórios no nível de pacote, módulo, classe e caso de teste, portanto, inicializações caras podem ser feitas com a menor frequência possível.

Exemplo de Teste

1. Considerando o seguinte script nosetest.py

```
# content of nosetest.py
def func(x):
    return x + 1

def test_answer():
    assert func(3) == 5
```

2. Efetuando o teste

```
C:\python>nosetests –v nosetest.py
```

3. A saída será:

```
nosetest.test_answer ... FAIL
======================================================================
FAIL: nosetest.test_answer
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Python34\lib\site-packages\nose\case.py", line 198, in runTest
    self.test(*self.arg)
  File "C:\Python34\nosetest.py", line 6, in test_answer
    assert func(3) == 5
AssertionError
----------------------------------------------------------------------
Ran 1 test in 0.000s
FAILED (failures = 1)
```