Pruebas unitarias con Python Unit

Jesús Espino García

Kaleidos
Python Madrid 2011

15 de Julio de 2011

¿Que son las pruebas unitarias?

- Son pruebas individuales de métodos o clases.
- Son pruebas automáticas (no por definición, sino por practicidad)
- Consiste en muchos casos de prueba
  - Cada uno de ellos prueba un aspecto determinado
  - Para probar un método hacen falta varios casos de prueba

¿Por qué hacer pruebas?

- Un programa no es aceptable si no cumple su especificación.
- Un programa no es aceptable si hace lo que no debe hacer.
- Nos da sensación de seguridad y tranquilidad.
- Evita la inclusión recurrente del mismo error.

"Un programador jamás debería entregar un programa sin haberlo probado. Igualmente, quien recibe un programa de otro jamás debería aceptarlo sin haberlo probado. Cualquier funcionalidad de un programa sin una prueba automatizada, simplemente no existe" ("Extreme Programming Explained", de Kent Beck)

Estrategias de pruebas

- Ninguna: "Soy tan bueno que todo lo que hago funciona a la primera, sin ningún tipo de bug"
- Al final: Voy desarrollando y al final del proyecto me paso una semana probando, o dejo que pruebe el cliente.
- Constante 1: Voy haciendo las pruebas según desarrollo (Desarrollo ⇒ Prueba).
- Constante 2: Voy haciendo las pruebas según desarrollo (Prueba ⇒ Desarrollo). TDD
- Test reactivos: Ante un error, construimos un test que reproduzca el error y lo corregimos.

Introducción

- Provee de un framework para realizar las pruebas unitarias.
- Forma parte de la distribución base de python desde la versión 2.1.
- Inspirado en JUnit.

Conceptos

Unittest soporta 4 conceptos importantes

- test fixture: Acciones de preparación previas a los test (por ejemplo, poblado de bases de datos).
- test case: Unidad básica de tests, comprueba que el resultado sea el correcto ante unos parámetros.
- test suite: Una colección de "test cases" y/o "test suites".
- test runner: Componente encargado de la ejecución de los tests y la presentación de los resultados.

Estructura de un test case

- Es una clase que hereda de TestCase.
- Tendrá una serie de métodos que empiecen por "test_" que serán ejecutados automáticamente.
- Los métodos "test_" deben ejecutar el código que desean probar y luego "afirmar un estado".
- Los métodos "test_" no devuelven ningún valor, se consideran validos aquellos que haya terminado su ejecución.

Afirmando un estado

- Para afirmar los estados usamos "aserciones".
- Unittest nos provee de una serie de aserciones, las mas habituales son:
  - assertEqual
  - assertNotEqual
  - assertTrue
  - assertFalse
  - assertIs
  - assertIsNot
  - assertIsNone
  - assertIsNotNone
  - assertIn
  - assertNotIn
  - assertIsInstance
  - assertNotIsInstance
  - assertRaise
- Pueden ver un listado más exhaustivo de las aserciones en http://docs.python.org/library/unittest.html.

Ejemplo de método de test

```
def test_suma(self):
    resultado = 1+1
    assertEqual(rsultado,2)

    resultado = 1+8
    assertEqual(rsultado,9)
```

Ejemplo fichero de test

```
import unittest

class TestSuma(unittest.TestCase):
    def test_suma(self):
        resultado = 1+1
        assertEqual(resultado,2)

        resultado = 1+8
        assertEqual(resultado,9)
```

Inicialización de los datos

- Unittest nos permite definir un "test fixture".
- Para esto definimos los métodos setUp y tearDown.
- setUp se ejecuta justo antes de cada método test.
- tearDown se ejecuta justo después de cada método test.
- Nos servirán para inicializar y limpiar los datos de prueba.
- Además tenemos setUpClass y tearDownClass.
- Y setUpModule y tearDownModule.

Test Suite

- Unittest nos permite agrupar tests.
- Es útil para estructurar los test de diferentes maneras.
  - Por funcionalidad.
  - Por casos de uso.
  - Por situación en la interfaz.
  - Por proximidad de los datos.
  - ...

Test Runner

- Encargado de ejecutar los tests.
- Una clase con un método run, que recibe un TestCase o un TestSuite.
- Unittest viene solo con el TextTestRunner.
- Podemos implementar nuestros propios TestRunners.

Ejemplos

- Test unitario simple.
- Ejemplo de setUp, tearDown.
- Ejemplo de testSuite.
- Ejemplo de test ejecutable.
- Ejemplo de test con TestRunner.

Referencias

- Documentación de unittest: http://docs.python.org/library/unittest.html.
- TDD: http://en.wikipedia.org/wiki/Test-driven_development.
- Pruebas Unitarias: http://en.wikipedia.org/wiki/Unit_testing
- Python-ES: python-es@python.org

Dudas

...