

Python & TDD

Python

- Orientado a Objetos
- Tipagem Dinamica, porem forte
- Escopo definido pela indentação

Python

```
class NomeDaClasse :  
  
    def NomeDoMetodo(self):  
        var1 = 'string'  
        var2 = "outra string"  
        return var1 + " - " + var2
```

return var1 + " - " + var2

Python

```
class NomeDaClasse :  
  
    def metodoComIf(self, param):  
        var = 'string'  
        if param == var:  
            return 'igual'  
        else:  
            return 'diferente'
```

LGfUUN , qTt6L6Uf6,

Python

```
class NomeDaClasse :  
  
    def metodoComWhile(self, qtd):  
        while qtd > 0:  
            qtd -= 1  
            print 'teste com while'
```

print 'teste com while'

Python

```
class NomeDaClasse :  
  
    def metodoComLista(self):  
        lista = [1,2,3,4,5]  
        for i in lista:  
            print i
```


Python

```
class NomeDaClasse :  
  
    def metodoComDicionario(self):  
        dic = {'chave' : 2, 'a' : 3, 'b' : 4}  
        for i in dic:  
            print "chave : " + i + " | valor : " + str(dic[i])
```

```
print "chave : " + i + " | valor : " + str(dic[i])
```

Python

```
class NomeDaClasse :  
  
    def NomeDoMetodo(self):  
        ...  
    def metodoComIf(self, param):  
        ...  
    def metodoComWhile(self, qtd):  
        ...  
    def metodoComLista(self):  
        ...  
    def metodoComDicionario(self):  
        ...  
  
obj = NomeDaClasse()  
print obj.NomeDoMetodo()  
print obj.metodoComIf(30)  
obj.metodoComWhile(4)  
obj.metodoComLista()  
obj.metodoComDicionario()
```


TDD

- Ferramenta para proteger o desenvolvedor dele próprio
- Os Testes são escritos antes do código
- *Feedback* instantâneo
- Serve como Documentação

TDD

```
import unittest

from test import NomeDaClasse

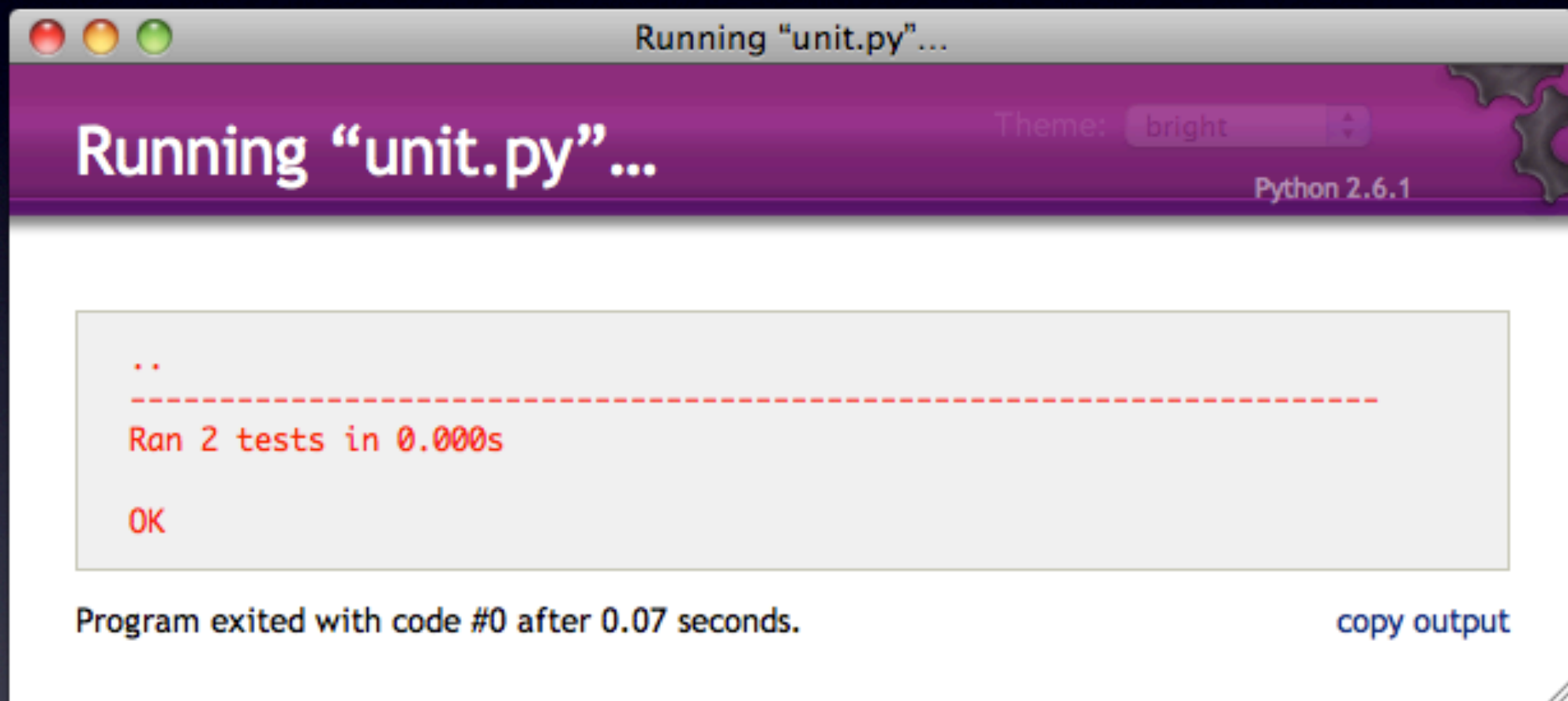
class testa_metodo(unittest.TestCase):
    def test1(self):
        obj = NomeDaClasse()
        self.assertEqual(obj.metodoComIf('string'), 'igual')

    def test2(self):
        obj = NomeDaClasse()
        self.assertEqual(obj.metodoComIf('bazinga'), 'diferente')

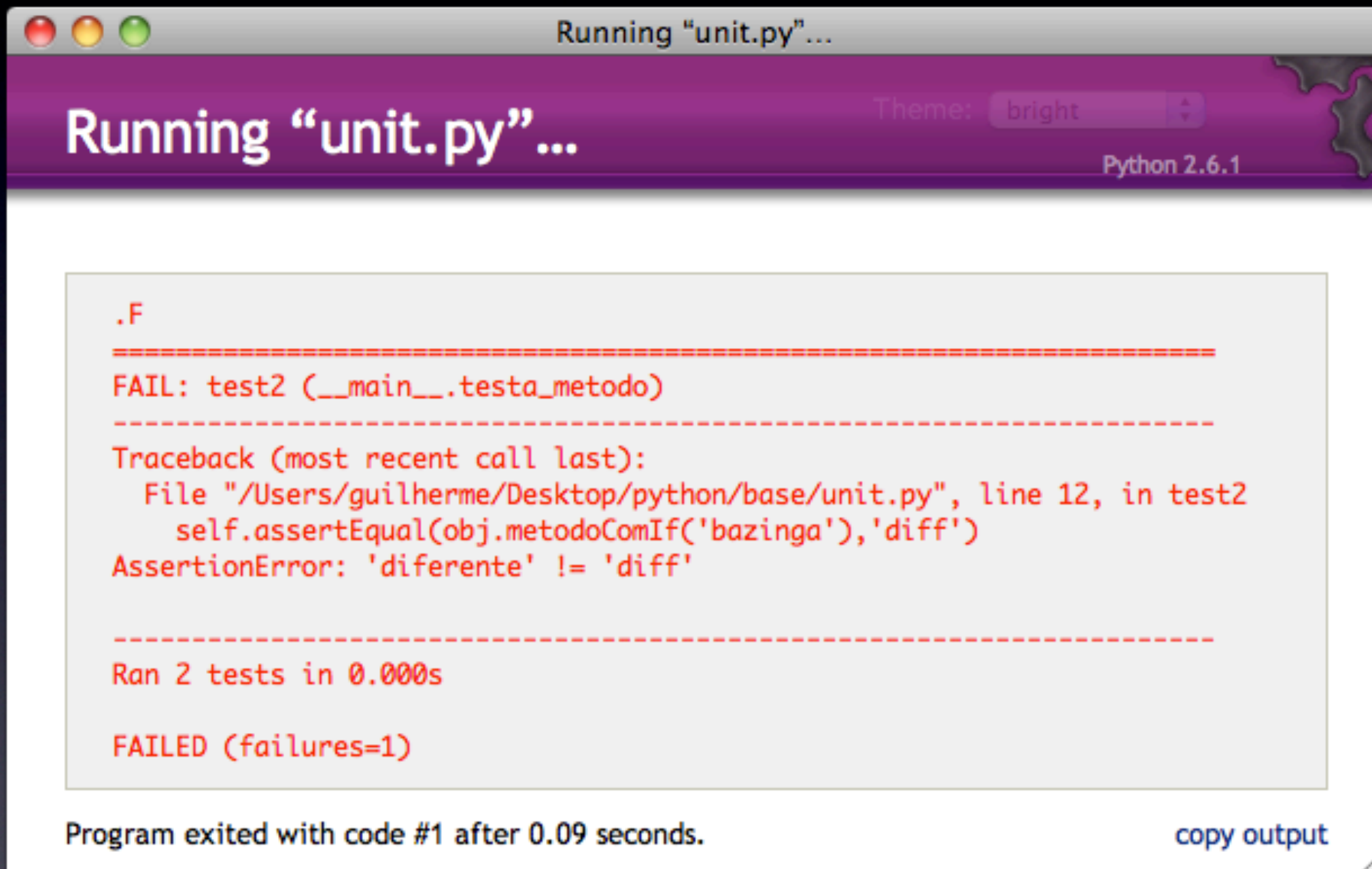
unittest.main()
```

unittest.main()

TDD



TDD



The image shows a terminal window titled "Running 'unit.py'...". The window has a purple header bar with the title "Running 'unit.py'..." on the left, "Theme: bright" in the center, and "Python 2.6.1" on the right. The main content area is white and displays the output of a test run. The output shows a failure for a test named "test2". The failure message is "AssertionError: 'diferente' != 'diff'", indicating that the actual result "diferente" does not match the expected result "diff". The traceback shows the error occurred in "unit.py" at line 12. The test run summary shows "Ran 2 tests in 0.000s" and "FAILED (failures=1)". At the bottom of the window, it says "Program exited with code #1 after 0.09 seconds." and there is a "copy output" link.

```
Running "unit.py"...
Theme: bright
Python 2.6.1

.F
=====
FAIL: test2 (__main__.testa_metodo)
-----
Traceback (most recent call last):
  File "/Users/guilherme/Desktop/python/base/unit.py", line 12, in test2
    self.assertEqual(obj.metodoComIf('bazinga'),'diff')
AssertionError: 'diferente' != 'diff'

-----
Ran 2 tests in 0.000s

FAILED (failures=1)

Program exited with code #1 after 0.09 seconds.
copy output
```