

# Testes de Unidade com JUnit

Alberto Costa Neto  
DComp - UFS





# Bugs...

- Debugar e Modificar levam muito mais tempo do que Escrever o código
- Consertar um *bug* é rápido! O problema é encontrá-lo
- Uma correção de um *bug* pode introduzir novos *bugs*



# Como muitos testam...

- “Cada classe deve ter seus métodos de teste”
  - Em Java, normalmente fazemos testes em **métodos main**
    - Separados
    - Muitas vezes são **descartados depois que a classe está “pronta”**
  - Geralmente esses testes devem ter sua **saída interpretada**

# Filosofia dos testes de unidade

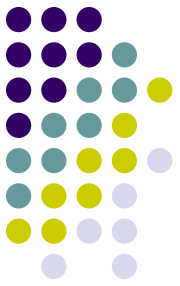


- A identificação de erros nos testes deve ser automática
  - Indicar o teste, o valor esperado e o obtido
  - Caso não tenham havido erros, uma mensagem com OK é o suficiente
- Cada teste deve ser independente

# Filosofia dos testes de unidade



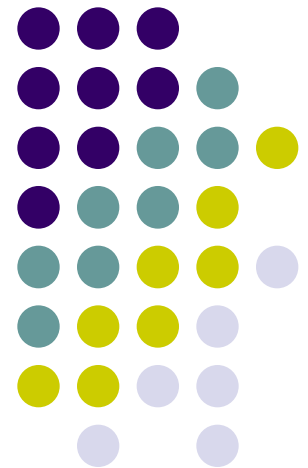
- Fazer o teste antes de implementar
  - Ponto de parada
  - Objetivos mais explícitos
- Ferramenta para executar os testes de uma vez
- Os testes devem ser pequenos e rápidos
  - Permite executá-los freqüentemente



# Obstáculos

- Convencer programadores de que:
  - É interessante
  - Gasta-se menos tempo
- Alguns programadores **não testam**

# JUnit



# JUnit



- É um *framework* de código aberto criado por Erich Gamma (Padrões de Projeto) e Kent Beck (XP)
- Traz três tipos de interface para execução de testes (Swing, AWT e Texto)



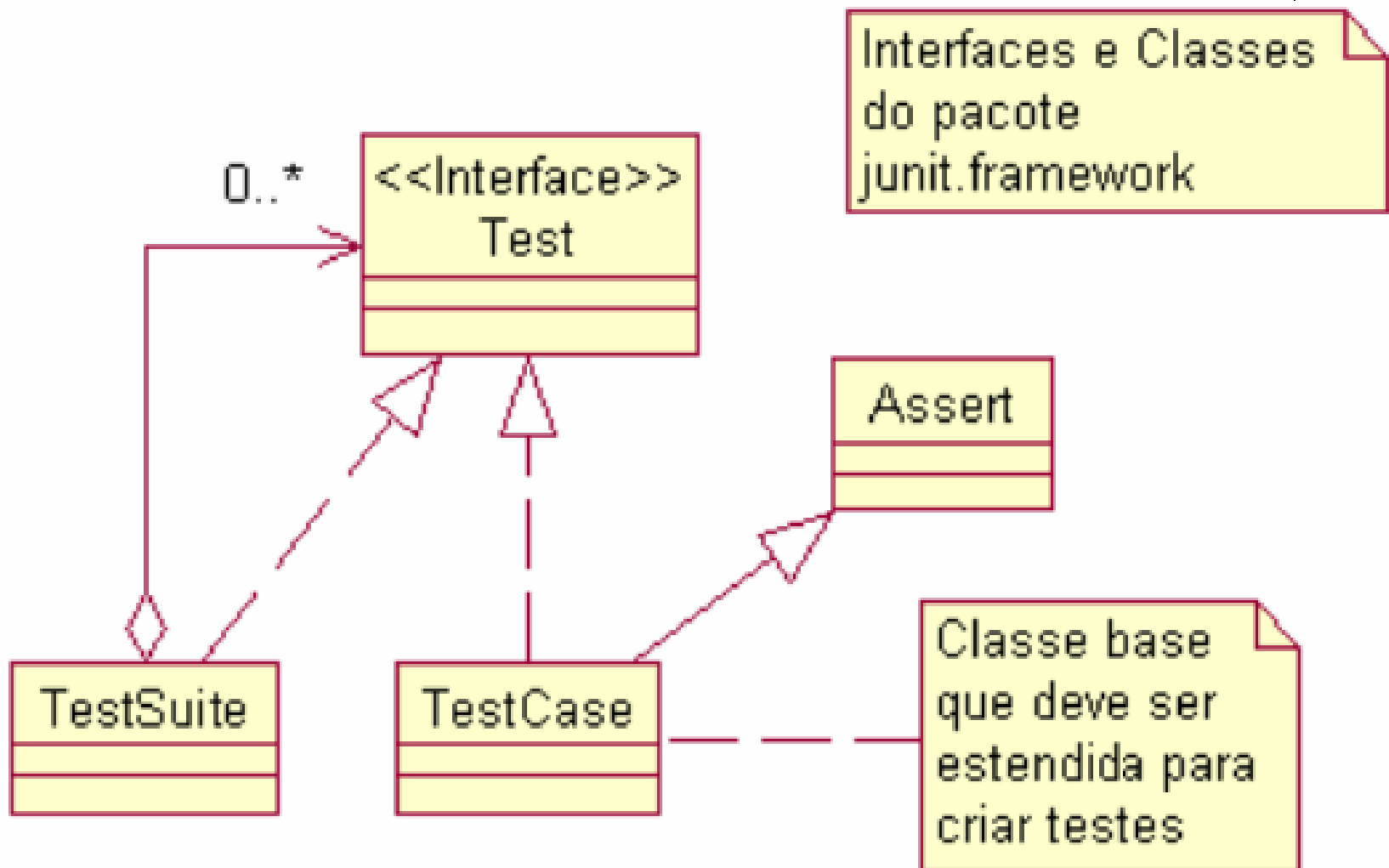


# Teste de Unidade

- Ferramentas
  - <http://www.junit.org> (Java)
  - <http://dunit.sourceforge.net> (Delphi)
  - <http://cppunit.sourceforge.net> (C++)
  - <http://cunit.sourceforge.net> (C)
  - <http://httpunit.sourceforge.net> (Extensão do JUnit)



# Hierarquia de classes





# Principais Métodos

- Métodos que podem ser implementados em subclasses de **TestCase**
  - Construtor
  - test<nome>
  - setUp (antes de cada método de teste)
  - tearDown (depois de cada método de teste)
  - suite



# Principais Métodos

- A classe **Assert** traz um conjunto de métodos, herdados por **TestCase**, os quais podem ser usados nos métodos `test<nome>`
  - `assertEquals` (usa o método `equals`)
  - `assertNull`
  - `assertNotNull`
  - `assertSame` (mesma referência)
  - `assertTrue`
  - `fail` (encerra o teste)



# JUnit 4

- Passou a usar anotações na classe de Testes
- `@Test` (indica que é um método de teste)
- `@Before` (executado antes de cada teste)
- `@After` (executado depois de cada teste)
- `@BeforeClass` (antes de iniciar os testes)
- `@AfterClass` (depois de concluir os testes)