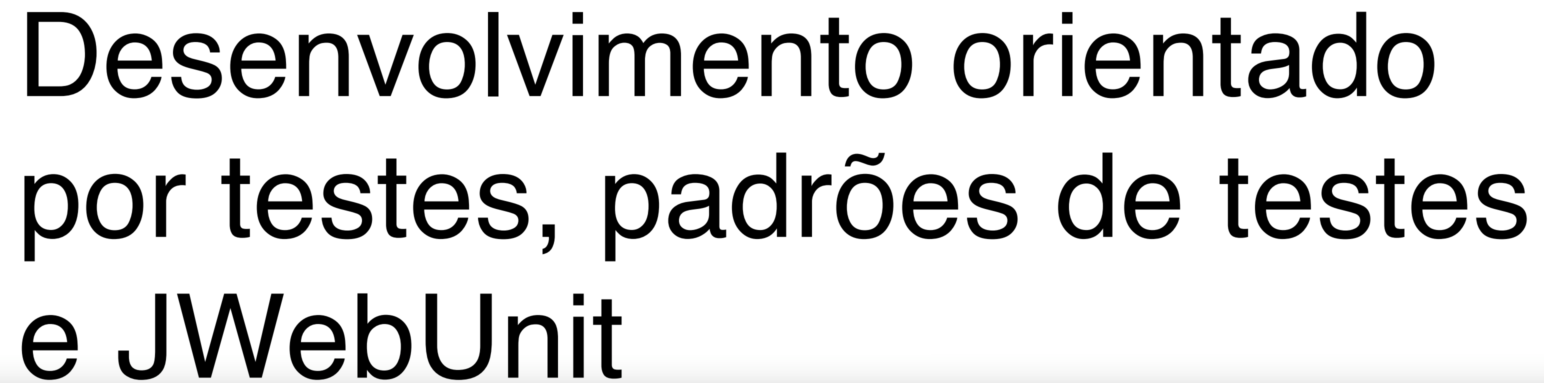
Desenvolvimento orientado por testes, padrões de testes e JWebUnit



… ou por que você quer fazer isso mas sempre deixa pro final?

Copyleft -- Alexandre Freire

Por que testar?

- Precisamos saber se o software funciona
- Todo software (minimamente razoável) é testado
- Testes manuais aumentam as chances de falhas
  - Testar as mudanças não é equivalente a ter testes

Copyleft Alexandre Freire

- Quanto mais stress, menos você testa!
- Quanto menos você testa, mais erros ira cometer.
- Quanto mais erros você comete, mais stress…
- Goto 1

Copyleft Alexandre Freire

Xp e metodologias ágeis

- Entre as práticas se introduz o costume de escrever testes automatizados
- Os testes são escritos preferêncialmente antes do próprio código
- Mesmo dependendo de outras práticas, ter testes é quase sempre benéfico como uma prática isolada

Copyleft Alexandre Freire

Ciclo dos testes automatizados

- Quanto mais stress, rode os testes mais vezes.
- Rodar os testes diminui o stress, diminuindo o número de erros cometidos.
- O que reduz ainda mais o stress!
- Os testes transformam medo em tédio.

Copyleft Alexandre Freire

- Código limpo, que funciona!
- É uma maneira previsível de desenvolver.
- Você sabe quando acabou aquilo que tinha que fazer.
- Não precisa se preocupar (tanto) com bugs.

Copyleft Alexandre Freire

- Primeiro, cuide do “que funciona”.
- Depois, cuide do “código limpo”.
- Mantra do TDD:
  - Vermelho
  - Verde
  - Refatorar para eliminar duplicação (consequentemente dirigindo o design organicamente)

- Código cresce a partir da criação de testes.
- Você realiza pequenas mudanças (podem ser feias).
- Roda os testes (todos) frequentemente.
- Refatore em pequenos passos para melhorar o código.

Copyleft Alexandre Freire

Coragem!

- A existência dos testes da coragem aos desenvolvedores
- Sabemos se o software esta funcionando como devia
- Adicionamos novas funcionalidades sem medo de criar bugs (se acontecer, seremos avisados!)

Copyleft Alexandre Freire

- Estamos falando de testes de unidade, que nos ajudam a desenvolver o software.
- Outros tipos de testes:
  - Performance
  - Stress
  - Usabilidade
  - Aceitação

Copyleft Alexandre Freire

Como sei que os testes são bons

- Você testou toda lógica de negócios do seu software?
  - Existem ferramentas que verificam a cobertura dos testes (http://quilt.sourceforge.net/)
- Inserção de defeitos
  - Ao inserir defeitos no seu código propossitalmente, os testes devem falhar!
  - Existem ferramentas que fazem isso também (http://jester.sourceforge.net/)

Copyleft Alexandre Freire

- Só escreva código “de negócio” quando um teste falhou
- Elimine qualquer duplicação encontrada

Copyleft Alexandre Freire

- Design orgânico, orientado por código que roda!
- Escreva seus próprios testes, não dá para ficar esperando outros escreverem os testes pra você
- O seu ambiente tem que suportar feedback veloz
- Para ser fácil de testar, o seu design tem que ser composto de objetos altamente coesos e com baixo acoplamento

Copyleft Alexandre Freire

Testes de unidade bons

- Rodam rápido (e rodar todos testes deve demorar no máximo 10 minutos)
- Rodam isolados (você deve poder rodá-los em qualquer ordem)
- Usam dados que sejam fáceis de ler e entender
- Usam dados reais quando necessário (cópia de dados de produção)
- Representam um passo em direção ao seu objetivo geral

Copyleft Alexandre Freire

- Faça uma lista (ao programar novas idéias vão surgir, coloque-as no fim da lista, ou até mesmo em uma outra lista)
- Coloque na lista exemplos de todas operações que você sabe que vai implementar
- Coloque a versão “nula” dessas operações na lista
- Liste as refatorações que você acha que terá de fazer para ter código limpo no final.

Copyleft Alexandre Freire

- Teste primeiro
- Você não vai testar depois, admita isso.
- Quando escrever as asserções?
  – Escreva as asserções primeiro!
  – Qual é a resposta certa?
  – Como vou verificar isso?

Copyleft Alexandre Freire

Exemplo de asserções primeiro

- Vamos testar comunicação via socket com outro sistema.
- Qual a resposta certa?
  - Depois de tudo o socket deveria estar fechado e eu devo ter conseguido ler a string “foobar”

Copyleft Alexandre Freire

```
testTransacaoCompleta(){
    …
    assertTrue(socket.isClosed());
    assertEquals(“foobar”,
    resposta.toString());
}
```


Copyleft Alexandre Freire

De onde vem a resposta?

```
testTransacaoCompleta(){
   …
   Buffer resposta = socket.contents();
   assertTrue(socket.isClosed());
   assertEquals(“foobar”,
   resposta.toString());
}
```


Copyleft Alexandre Freire

```
testTransacaoCompleta(){
    …
    Socket socket = new Socket("localhost",
    portaDefault());
    Buffer resposta = socket.contents();
    assertTrue(socket.isClosed());
    assertEquals("foobar", resposta.toString());
}
```


Copyleft Alexandre Freire

```
testTransacaoCompleta(){
    Server fooba = new Server(portaDefault,
    “foobar”);
    Socket socket = new Socket(“localhost”,
    portaDefault());
    Buffer resposta = socket.contents();
    assertTrue(socket.isClosed());
    assertEquals(“foobar”, resposta.toString());
}
```


Copyleft Alexandre Freire

- Não espalhe dados por ai.
- Nunca use a mesma constante para mais de uma coisa
  - Se não existe diferença conceitual entre 1 e 2, use o 1.
  - Nunca teste 2+2, prefira testar 2+3

Copyleft Alexandre Freire

```
Banco b = new Banco();
b.adicionaCambio(“$”, “R$”,
  TAXA_DOLAR);
b.comissao(COMISSAO_PADRAO);
Dinheiro saldo = banco.converte(new
  Nota(100, “$”), “R$”);
assertEquals(new Nota(278.54, “R$”),
  saldo);
```


Copyleft Alexandre Freire

Demonstre a intenção dos dados

```
Banco b = new Banco();
b.adicionaCambio(“$”, “R$”, 2.7);
b.comissao(0.01);
Dinheiro saldo = banco.converte(new
  Nota(100, “$”), “R$”);
assertEquals(new Nota(100 / 2.7 * (1 -
  0.01), “R$”), saldo);
```


Copyleft Alexandre Freire

- Comece com uma operação “nula”
  - Começar com um teste realístico dá muito mais trabalho
  - Testes com entrada igual a saída
  - A entrada deve ser a menor possível

```
Redutor r = new Redutor(new Poligono());
assertEquals(0, r.resultado().pontos());
```


Copyleft Alexandre Freire

#

E para continuar…

- Use e abuse de testes como documentação
- Testes de aprendizado (quando for usar uma nova API por exemplo)
- Novas idéias entram na lista como novos testes
- Testes de regressão - se algo falha, escreva o teste!

Copyleft Alexandre Freire

- Ok, você escreveu o teste e ele falha, e agora?
- Faça o teste rodar, 3 técnicas boas:
  – “fake it”
  – Triangularização
  – Implementação óbvia

Copyleft Alexandre Freire

Fake it ('til you make it)

- Retorne uma constante
- Gradualmente transforme a constante em expressões usando variáveis
- Ter algo rodando é melhor do que nada…
- Limpe o código imediatamente

Copyleft Alexandre Freire

```
assertEquals(new Data("5/12/2004"), new
    Data("6/12/2004").ontem());

Data:

public Data ontem(){
    return new Data("5/12/2004");
}
```

Copyleft Alexandre Freire

Data:

```
public Data ontem(){
    return new Data(new Data(“6/12/2004”).dias()
    -1);
}
```

Ainda duplicado!

Copyleft Alexandre Freire

Data:

```
public Data ontem(){
    return new Data(this.dias() -1);
}
```

No meu teste, this = "6/12/2004"!

- Abstrair somente quando você tem dois ou mais exemplos

```
assertEquals(4, soma(3,1));

soma(int a, int b){
   retorna 4;
}
```


Copyleft Alexandre Freire

```
assertEquals(4, soma(3,1));
assertEquals(7, soma(3,4));

soma(int a, int b){
   retorna a + b;
}
```


Copyleft Alexandre Freire

Implementação óbvia

- Quando é óbvio, implemente a solução óbivia!
- Não é óbvio?

Copyleft Alexandre Freire

Padrões de testes

- ChildTest
  - Testes grandes são divididos em testes menores, trabalhe no teste menor, depois retorne ao teste grande
- MockObject
  - Use implementações vazias quando você depende de objetos caros ou complexos

Copyleft Alexandre Freire

+ padrões de testes

- Self shunt
  - Use o próprio teste como MockObject (implementando alguma interface)
- LogString
  - Use uma string para logar chamadas a métodos e verifique o conteudo da String no final
- CrashTestDummy
  - Subclasses para testar casos de erro

Copyleft Alexandre Freire

- Não são impossíveis
- JWebUnit para testar interfaces HTML em Java
  - ```
    assertLinkPresent(″link");
    clickLink(″link");
    ```
  - ```
    assertFormPresent(″form");
    setFormElement("field1",
    "value1");
    submit();
    ```


Copyleft Alexandre Freire

- Direto da fonte…

Copyleft Alexandre Freire