



**Grupo de Usuários Java
Campina Grande**

Programação Orientada a Testes ***“Test Driven Development - **TDD**”***

Rodrigo Rebouças
rodrigor.com

Somos parceiros do:



O que é TDD?



- É orientar o desenvolvimento aos testes;
- Implementar o teste antes da funcionalidade;
- É uma mudança de comportamento.

Passos de um processo TDD



- O ciclo do TDD é formado pelos seguintes passos [Beck 2003]:
 - Adicione um pequeno teste;
 - Execute todos os testes, mesmo que falhem;
 - Faça uma mudança;
 - Execute todos os testes e eles não devem falhar;
 - Refatore o código para tirar duplicações.

TDD mantra



- Costuma-se resumir esses passos no :
Vermelho/Verde/Refatore
(Red/Green/Refactor):
 1. Vermelho: Faça um pequeno teste que pode até não funcionar;
 2. Verde: Faça o teste funcionar rapidamente;
 3. Refatore: Elimine toda duplicação criada para fazer o teste funcionar;

Eliminando o medo



- TDD é uma forma de gerenciar o medo durante a programação. O medo no sentido de "tome cuidado" é bom, mas pode trazer alguns efeitos desagradáveis:
 - Pode deixar o programador hesitante;
 - Faz com que se queira comunicar menos;
 - Leva os desenvolvedores a evitarem feedback;
 - Pode deixar o desenvolvedor irritado ou nervoso.

Eliminando o medo



- Ao invés de tais efeitos, deve-se buscar:
 - Aprender de forma concreta e o mais rápido possível;
 - Comunicação mais clara;
 - Feedback concreto;
 - Evitar o nervosismo;
- O medo pode ser eliminado através de testes automáticos. Sem estes, chegamos ao seguinte ciclo: Quanto mais estressado estiver o desenvolvedor, menos testes ele irá fazer;
- Quando menos testes, mais erros serão cometidos. Quanto mais erros, maior será o estresse, e assim por diante.

Práticas do TDD



- Os testes devem ser fáceis e rápidos de se fazer, mas para isso, os sistemas testados devem ser pouco acoplados e coesos, levando à necessidade de refatoramentos em alguns casos;
- Ao se refatorar, sugere-se o uso de ferramentas que automatizem esse processo para evitar que os testes sejam quebrados sem necessidade;
- Antes de testar, deve ser feita uma lista do que deve ser testado e das condições que os testes irão verificar;

Práticas do TDD



- Os testes feitos usando TDD não substituem testes de desempenho, estresse e usabilidade;
- Deve-se escrever o teste antes de escrever o código;
- Só é escrito algum código novo quando existir um teste automático falhando;
- Seguir os vários padrões para testes propostos no livro TDD;
- Só fazer refactoring quando se souber que os testes são de fato suficientes e quando ao executá-los nenhum erro acontece (green bar);

Práticas do TDD



- Os testes devem ser capazes de ignorar uns aos outros completamente, inclusive com relação a ordem de execução;
- Os dados utilizados nos testes devem torná-los fáceis de ler e seguir;
- Inclua resultados esperados e os reais nos testes e tente fazer bem aparente essa relação (dados evidentes);
- Quando um erro for encontrado, a primeira coisa a fazer é escrever um teste o mais simples possível que falhe. Uma vez que ele seja executado sem problemas, melhore-o;

Alguns argumentos para se utilizar TDD



- "TDD melhora a velocidade de desenvolvimento e a eficiência, porque encontrar erros mais cedo torna-se barato e rápido." [Rohrl]
- "Se Programação Extrema parece extrema demais, inicie com o desenvolvimento orientado a testes" [Rohrl]

Alguns argumentos para se utilizar TDD



- O estilo de programação Vermelho/Verde/Refatore traz as seguintes vantagens [Beck 2003]:
 - Reduz dramaticamente a densidade de defeitos do código;
 - A garantia de qualidade passa de um trabalho reativo a um trabalho pró-ativo;
 - Reduzem-se as surpresas, ajudando o gerente de projeto a estimar de forma mais segura;
 - É possível obter versões sendo entregues diariamente.
- "Executar testes imediatamente antes de fazer uma alteração e após ela faz a pessoa se sentir bem e reduz o número de erros que ela pode cometer".

Vamos à prática!



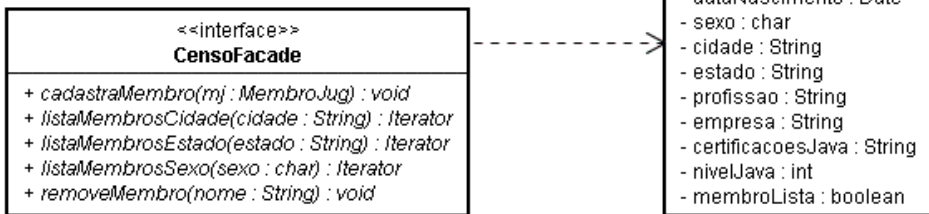
- Precisamos conhecer os membros dos JUGs...
- Vamos implementar um sisteminha de cadastro de membros do JUG!

Vamos à prática!



- Alguns requisitos funcionais...
 - Temos uma entidade “MembroJug”
 - E podemos:
 - Cadastrar um membro dos Jugs
 - Listar os membros de uma cidade
 - Listar os membros de um estado
 - Listar todos os membros dos Jugs
 - Obter a quantidade de membros cadastrados
 - Obter o total de membros da lista de discussão

Vamos à prática!



Referências



- [Rohrl] Agile Development Tools for Test-Driven Development with Java.
Dr. Armin Rohrl Stefan Schmiedl.
- [Beck 2003] Test-Driven Development.
Kent Beck. 2003. Addison-Wesley.



**Grupo de Usuários Java
Campina Grande**

Programação Orientada a Testes

“Test Driven Development - **TDD**”

Rodrigo Rebouças
rodrigor.com

Somos parceiros do:



90% do material destes slides foi gerado por
Ayla Rebouças (ayla@stcursos.com). Obrigado!