Test Driven Development: TDD Simples e Prático

Maurício Aniche
Professor da disciplina de Projeto de Software da FEA-USP, Pesquisador do Grupo de Qualidade de Software do IME-USP, Desenvolvedor de Software

Versão 1.0 - 2015

Este livro é licenciado sob a Licença Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported. Para ver uma cópia da licença, visite http://creativecommons.org/licenses/by-nc-nd/3.0/br/ ou envie uma carta para Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

Sumário

Introdução
Ciclo de desenvolvimento
Histórias para o seu
Testes Unitários
Nosso funcionalidade
Sendo ciclos, parte I
Mundanças exigem no código
Hello, World do TDD
Ambiente
Nosso Objeto
Documentação para o futuro
Testando exceções
Sendo ciclos, parte II
Testes separados por uma execução
Sendo ciclos, parte III
Refatoração
Mundanças exigem no código
Hello, World do TDD
Ambiente
Nosso Objeto
Documentação para o futuro
Testando exceções
Sendo ciclos, parte II
Testes separados por uma execução
Sendo ciclos, parte III
Refatoração
Desafiadores
Referências

Introdução

Em 1999 Kent Beck publicou **TDD**, que, por sua vez, trouxe para o mundo, para que pudessem conhecer, o conceito de **Test Driven Development**.

Beck é também o criador do **JUnit**, o que significa que ele sabia muito bem como produzir bons testes para software e sua **TDD**.

Dificuldade: Infinita

Mas, o que é um teste? Como escrevê-lo? Como ele pode ajudar o desenvolvimento de software?

Se não sabe muito bem como escrever um teste, não se preocupe. Não é o seu caso, mas muitos programadores não sabem.

Se não sabe o que é um teste, **TDD** é, sim, o que você está pensando. Testes são pequenos pedaços de código que chamam o seu código de produção e verificam se ele está funcionando como esperado.

Se não sabe como escrever