A Maneira de Testivus

Menos Dogma de Testes
Mais Karma de Testes

Boas orientações sobre desenvolvimento e testes unitários, apresentados como doze trechos de sabedoria Ocidental.

Traduzido por Alberto Savoia

www.agitar.com/downloads/TheWayOfTestivus.pdf

* Versão em português por Marcelo Andrade <http://mfandrade.wordpress.com>

Introdução do Tradutor

Em Maio de 2006, uma bem-preparada expedição internacional à cordilheira do Himalaia perdeu seu rumo. Após duas semanas andando a esmo - famintos e sedentos, tais como meros novatos inexperientes perdidos à duas semanas - acabaram chegando a uma antiga caverna.

Dentro da caverna eles viram um labirinto de cubículos. Cada cubículo tinha uma espécie de escrivaninha de madeira, uma cadeira de bambu ergonomicamente correta, um calendário do Dilbert™ e um estranho dispositivo mecânico parecido com um computador. Em um canto da caverna eles encontraram barris de um líquido negro (identificado mais tarde como uma bebida carbonada e altamente cafeinada) e uma mesa de ping-pong. Eles concluíram ser a caverna um antigo ambiente de desenvolvimento de software. O mais antigo já descoberto. Mais antigo até que a Netscape.

Junto com os vários itens surpreendentes que eles encontraram na caverna estava o mais incrível de todos: um bilhete deixado por um dos programadores. O guia da expedição, apesar de não ser dos melhores, sabia ler os escritos antigos e traduziu a anotação no bilhete para eles:

"Concluímos o release antes do prazo - de novo. Todos os testes passaram, então decidimos tirar o resto da semana de folga. Estamos indo velejar. Por ser uma atividade em equipe, acreditamos que certamente será muito proveitoso para nós."

Os exploradores se entreolharam, atônitos. Não apenas tinham descoberto o ambiente de desenvolvimento de software mais antigo do mundo como também tinham descoberto uma equipe de programadores que, ao que parece, concluíam regularmente seu trabalho antes dos prazos.

Qual o segredo desses programadores ancestrais? E o que aconteceu com eles? Os exploradores procuraram por pistas em cada um dos cubículos, e encontraram dois pequenos livros já bem desgastados. Um deles intitulado "Aprenda A Velejar Em 30 Minutos", que explicava o destino dos programadores. Você tem em suas mãos uma tradução do segundo livro, "A Maneira de Testivus". Quem escreveu este livrinho misterioso? O que é Testivus? Apenas o Google™ sabe com certeza.

Seria o conteúdo deste texto a razão pela qual os tais programadores antigos concluíam seus projetos antes do prazo? Não temos como saber com certeza, mas acreditamos que incrível bravura desses programadores devia-se provavelmente à combinação da filosofia Testivus e do consumo de grandes quantidades do tal líquido escuro e cafeinado encontrado num dos cantos da caverna.

Siga a leitura deste livrinho e tire suas próprias conclusões.

Alberto Savoia, CTO/Cofundador da Agitar Software
Abril de 2007, Mountain View, Califórnia

Se você escreve código, escreva testes

O pupilo perguntou ao mestre programador:
"Quando devo parar de escrever testes?"

E o mestre respondeu:
"Quando você parar de escrever código."

O pupilo perguntou:
"Quando devo parar de escrever código?"

O mestre respondeu:
"Quando você se tornar um gerente."

O pupilo suou frio e então perguntou:
"E quando me tornarei um gerente?"

E o mestre então respondeu:
"Quando parar de escrever testes."

O pupilo então correu para escrever alguns testes
Tão rápido que seu rastro ficou marcado no chão.

**Se o código merece ser programado,
ele então merece ser testado.**

Não se prenda a dogmas de testes

O Dogma diz:

"Faça isso.
Faça apenas isso.
Faça apenas isso, desse jeito.
E faça-o porque eu lhe digo para fazer."

Dogma é inflexível.
Testar precisa de flexibilidade.

Dogma mata a criatividade.
Testar precisa de criatividade.

Adote o karma de testes

O Karma diz:

"Faça coisas boas e coisas boas acontecem a você.
Faça do jeito que você sabe.
Faça do jeito que você gosta."

Karma é flexível.
Testar precisa de flexibilidade.

Karma é afeito à criatividade.
Testar precisa de criatividade.

Pense no código e teste-o como código

Ao escrever o código, pense no teste.
Ao escrever o teste, pense no código.

Quando você pensa no código e o testa como código,
o teste se torna fácil e o código se torna belo.

O teste é mais importante que a unidade

O pupilo perguntou ao grande mestre programador Penas Esvoaçantes:
"O que faz com que um teste seja um teste de unidade?"

O grande mestre programador respondeu:
"Se fala com um banco de dados, não é um teste de unidade.
Se se comunica por uma rede, não é um teste de unidade.
Se lida com sistema de arquivos, não é um teste de unidade.
Se não pode rodar junto com outros testes de unidade,
não é um teste unidade.
Se é preciso fazer alguma coisa especial em seu ambiente para rodá-lo,
não é um teste unidade."

Outros mestres programadores intervieram e começaram a discutir.
"Desculpe pela pergunta", disse o pupilo. Mais tarde naquela noite, ele recebeu um bilhete do grande mestre programador. No bilhete, lia-se:

"A resposta do grande mestre Penas Esvoaçantes é um excelente guia.
Siga-a e você fará o certo na maioria das vezes.
Mas não se prenda a nenhum dogma.
Escreva o teste que precisa ser escrito."

O pupilo dormiu bem naquela noite.
Os outros mestres madrugaram discutindo.

O melhor momento para testar é quando o código está fresco

Teu código é como o barro.
Quando está fresco é macio e maleável.
Conforme o tempo passa, se torna duro e quebradiço.

Se tu escreves testes quando teu código está fresco e
fácil de se modificar, testar então será fácil,
e tanto o código quanto os testes ficarão consistentes.

Testes não eliminam problemas

Roda teus testes com frequência.
Não permitas que fiquem defasados.
Alegra-te quando eles passarem.
Alegra-te quando eles falharem.

Um teste imperfeito hoje é melhor que um teste perfeito algum dia

O perfeito é o inimigo do bom.
Não esperes o melhor para fazer o ainda melhor.
Não esperes o ainda melhor para fazer o bom.
Escreve hoje o teste que podes fazer hoje.

Um teste feio é melhor que nenhum teste

Quando o código for feio, os testes podem ser feios.

Você não gosta de escrever testes feios,
mas código feio precisa de testes à altura.

Nunca deixe que código feio lhe impeça de escrever testes,
mas faça com que código feio lhe impeça de gerar mais código feio.

Alguma vezes, o teste justifica os meios

O pupilo perguntou a dois mestres programadores:
"Não posso testar este código sem fazer mocks e nem violar o encapsulamento. O que devo fazer?"

O primeiro mestre programador respondeu:
"Mocks são ruins, e você nunca deveria violar o encapsulamento.
Reescreva o código para poder testá-lo adequadamente."

O segundo mestre programador respondeu:
"Mocks são bons, e o teste é mais importante que o encapsulamento."

O pupilo, confuso, saiu para beber. No bar, encontrou o grande mestre programador tomando cerveja e comendo asinhas de frango.

"Magnânimo grande mestre", disse o pupilo, "achava que tu não bebias.
E também não és tu um vegetariano?"

O magnânimo grande mestre sorriu e retrucou:
"Há momento em que tua sede é melhor saciada pela cerveja,
e tua fome, pelas asinhas de frango."

E, de repente, o pupilo não mais estava confuso.

Somente o tolo não usa ferramentas

O fazendeiro que não usa um arado
não é um bom fazendeiro.

O matemático que não usa um ábaco
não é um bom matemático.

Algumas tarefas são melhor realizadas a mãos limpas.
Há outras que são melhor feitas com ferramentas.

Não há nobreza em se fazer à mão
o que pode ser feito com uma ferramenta.

Não é sábio usar sua cabeça
quando sua cabeça não é necessária.

Bons testes falham

O pupilo chegou para o mestre programador e disse-lhe:

"Todos os meus testes passam todas as vezes. Não mereço um aumento?"

O mestre estapeou o pupilo na face e lhe respondeu:

"Se todos os teus testes passam, todas as vezes, precisas
escrever testes melhores."

Com uma marca vermelha no rosto, o pupilo foi se queixar ao RH.
Mas esta já é outra história.

A Maneira de Testivus

Se você escreve código, escreva testes.

Não se prenda a dogmas de testes.

Adote o karma de testes.

Pense no código e teste-o como código.

O teste é mais importante que a unidade.

O melhor momento para testar é quando o código está fresco.

Testes não eliminam problemas.

Um teste imperfeito hoje é melhor que um teste perfeito algum dia.

Um teste feio é melhor que nenhum teste.

Algumas vezes, o teste justifica os meios.

Somente o tolo não usa ferramentas.

Bons testes falham.

Esta tradução e impressão de
"A Maneira de Testivus"
foi trazida até você por:

JUnit Factory

Se você gosta da filosofia de Testivus,
e acredita em menos dogma de testes
e em mais karma de testes,
por favor, visite:

www.JUnitFactory.com

JUnit Factory ajuda você a aumentar seu
karma de testes amplificando e
automatizando seus esforços de testes.

E o melhor de tudo, é grátis.