Good Code

Treat your code like any other composition, such as a poem, an essay, a public blog, or an important email.

Peter Sommerlad, in "Only The Code Tells the Truth", from "97 things Every programmer should know", O'Reilly, 2010

Instituto Politécnico de Beja
Escola Superior de Tecnologia e de Gestão
João Paulo Barros

Beja, 23 de Novembro de 2012 - 1 de Novembro de 2014

Erros nos programas. O que fazer?

1. Corrigi-los
2. Evitar que surjam

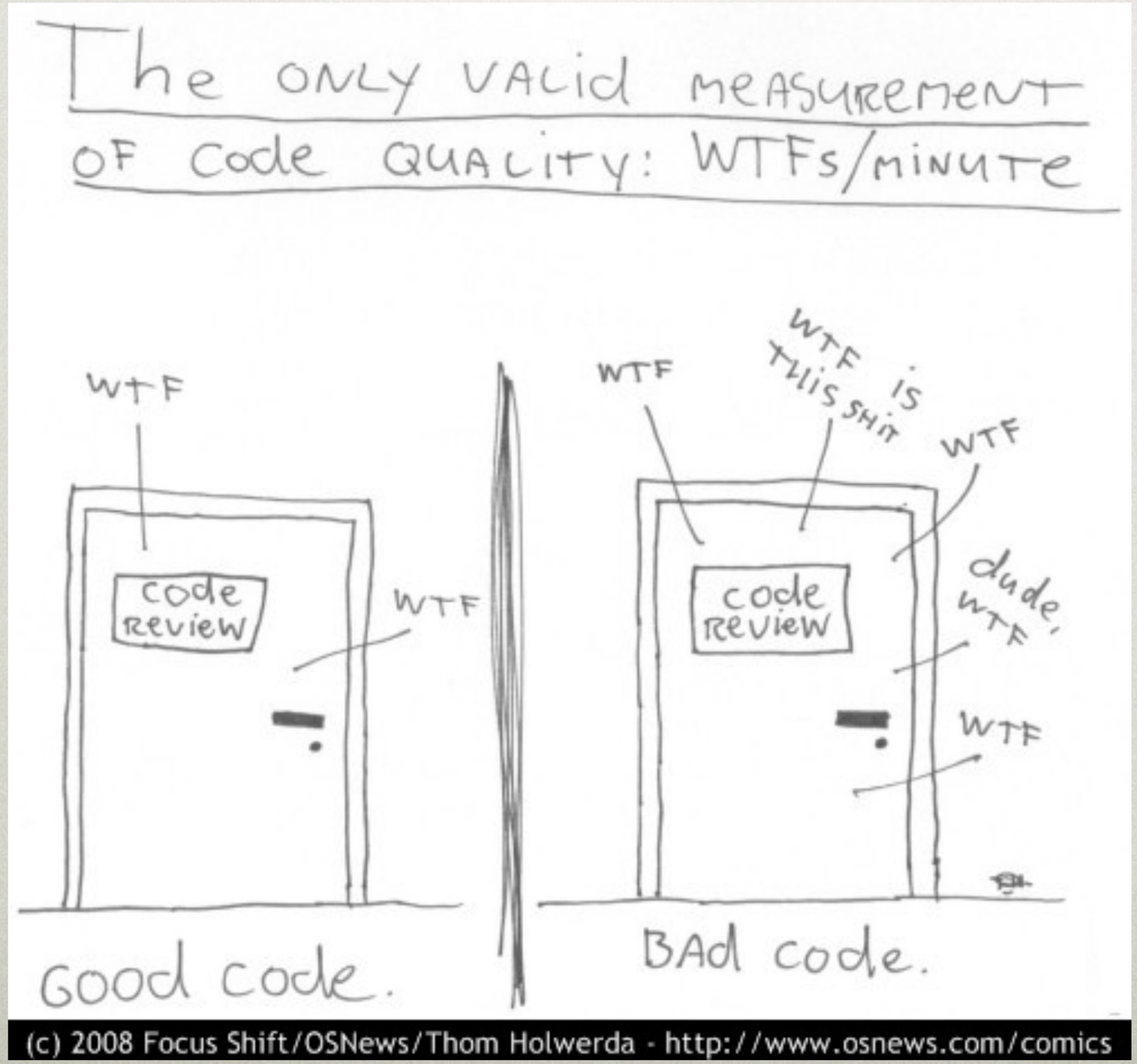
- "Clean Code", e.g. Regras de Estilo
- Asserções e programação por contrato
- ...
- Programação conduzida por testes

Mais prevenção

Treat your code like any other composition, such as a poem, an essay, a public blog, or an important email. Craft what you express carefully, so that it does what it should and communicates as directly as possible what it is doing; so that it still communicates your intention when you are no longer around. Remember that useful code is used much longer than ever intended. Maintenance programmers will thank you. And, if you are a maintenance programmer and the code you are working on does not tell the truth easily, apply the aforementioned guidelines in a proactive manner. Establish some sanity in the code, and keep your own sanity.

Peter Sommerlad, in "Only The Code Tells the Truth", from "97 things Every programmer should know", O'Reilly, 2010

Qualidade



Nomes

- Devem informar
- Não devem confundir
- Substantivos para variáveis.
- Verbos ou expressões com verbos para procedimentos e funções
- Perguntas para funções booleanas
- Pronunciáveis
- Seguir regras (classes, variáveis, constantes, funções)
- Tão longos quanto o necessário
- Utilizar os opostos certos: add/remove, get/set, etc.

Métodos/Funções/Rotinas/...

- Curto. 4, 40, 400 linhas?
    - Excepção: switch/if else if
    - Solução: polimorfismo*
- Deve fazer uma tarefa.
- Nomes informativos mesmo que longos
- Parâmetros por ordem de utilização (in, modify, out)
- Rotinas semelhantes utilizam a mesma ordem para os parâmetros
- Não utilizar os parâmetros como variáveis locais

* http://csis.pace.edu/~bergin/patterns/ppoop.html

Métodos/Funções/Rotinas/...

- Evitar *side effects*. Por exemplo, uma verificação de *password* que também inicia uma sessão.
- Minimize a quantidade de rotinas públicas/visíveis.
- Certifique-se que os parâmetros actuais correspondem aos formais
- Não retornar apontadores para variáveis locais

Comentários

Don't comment bad code - rewrite it

Brian W. Kernighan and P. J. Plaugher in The Elements of Programming Style

- Comentários legais
- Informação
- Intenção
- Clarificação
- Avisar as consequências
- Informar o que está para ser feito
- Evitar o que é inútil (que já está ou deveria estar no código)
- Comentar código: melhor um controlo de versões

Programação Defensiva

SEVOCAB
Software and Systems Engineering Vocabulary

Home/Search | Print Entire Vocabulary | Download ISO Version | Help

Search Results

1 results were found. Print Results

defensive programming. (1) a general approach to programming that assumes that errors will occur during both initial development and maintenance and, as a result, creates code in such a way that the program still operates properly when errors occur *(ISO/IEC/IEEE 24765:2010 Systems and software engineering--Vocabulary)*

- Proteja o programa de inputs inválidos
- Proteja o programa de inputs inválidos
- Proteja o programa de inputs inválidos
- Utilize asserções e programação por testes
- Coloque cada tarefa entre pré-condições e pós-condições.

Tratamento de Erros

- Asserções para o que nunca pode acontecer.
- “Error handling “ para o restante.
- Para código extremamente robusto mesmo as asserções devem ser tratadas.
    - “Desinfecção” de código.

O que Fazer na "Desinfecção de Código"

- Devolver um valor neutro
- Devolver o seguinte
- Repetir a resposta
- Devolver o mais próximo
- Fazer log ou mostrar uma mensagem
- Devolver um código de erro

O que escolher?

Depende...

robustez ou correcção?

...Refactoring

- Fuja das soluções rápidas e simplistas.
- Não deixe arrastar. Melhore já!
- *DRY* principle (*Don't repeat yourself*)
- Tire partido do IDE
- Procure a solução mais geral.

Referências

"Clean Code - A Handbook of Agile Software Craftsmanship", Robert C. Martin, Pearson Education, 2009.

"Code Complete: A Practical Handbook of Software Construction", Steve McConnell, Microsoft Press, 2004.

Específicos:

"Effective Java", Joshua Bloch, Addison-Wesley, 2008.
http://java.sun.com/docs/books/effective/