Refatoração:
Melhorando código existente

AgilCoop – Cursos de Verão 2010

Mariana Bravo
IME/USP

AgilCoop

Refatoração

- Uma mudança no sistema que **não altera** seu comportamento funcional, mas **melhora** sua estrutura interna
- Limpa o código, minimizando as chances de introduzir erros
- Melhora o *design* depois que o código foi escrito

De onde vem?

- Surgiu na comunidade Smalltalk nos anos ~90
- Desenvolveu-se formalmente na Universidade de Illinois em Urbana-Champaign
- Grupo do Prof. Ralph Johnson
  - Tese de PhD de William Opdyke (1992)
  - John Brant e Don Roberts: *The Refactoring Browser Tool*
- Kent Beck e Martin Fowler na indústria

O espírito da refatoração

O espírito da refatoração

Refatoração

O espírito da refatoração

Refatoração

Primeiro passo: Testes

- Conjunto sólido de testes garante que o comportamento não será alterado
- Refatorações podem adicionar erros
  - Porém, como são feitas em pequenos passos, é fácil recuperar-se de uma falha
- Testes devem ser automatizados e verificarem o próprio resultado

Exemplos

Extrair método

- Transforma um fragmento de código em um método com um nome explicativo

- Motivação:
  - Facilitar o entendimento de trecho de código
  - Aumentar as chances de reutilização do novo método

Renomear variável

- Muda o nome de uma variável para expressar melhor o seu propósito

- Motivação:
  - Facilitar o entendimento do papel da variável em seu escopo

Dinâmica da refatoração

- Cada mudança é simples...
- Mas seu efeito acumulado pode melhorar muito o *design* do código

- É construindo o sistema que podemos descobrir como melhorá-lo

Refatorar para...

- Melhorar o *design* do software
- Facilitar o entendimento do software
- Encontrar falhas mais facilmente
- Programar mais rapidamente

Quando refatorar

- Sempre que você precisar fazer algo e o código atrapalhar mais do que ajudar:
  - Quando adiciona funcionalidade
  - Quando corrige um erro
  - Quando revisa o código
  - Na terceira cópia, refatore (ou na primeira!)

- Quando o código cheira mau
  - *"If it stinks, change it."* (Se feder, troque-o.)
    Vó de Beck, sobre como cuidar de bebês

Alguns maus cheiros

- Nomes de variáveis obscuros
- Código duplicado
- Método muito longo
- Classe muito grande
- Intimidade inapropriada
- Comentários
- Muitos parâmetros

Maus cheiros em testes

- Problemas no código:
  - Testes obscuros
  - Presença de condições
  - Replicação de código
  - Lógica de testes em produção
- Problemas no comportamento:
  - Testes frágeis
  - Testes lentos
  - Intervenções manuais

Exemplos

Mover método

- Move um método que utiliza mais funcionalidades de outra classe do que aquela em que se encontra
- Motivação:
  - Uma classe tem muitos comportamentos
  - O acoplamento entre classes é muito forte
- O velho método delega ou é removido

Usando ferramentas

Substituir temporário por chamada

- Substitui o uso de uma variável por uma chamada a um método que realiza as operações

- Motivação:
  - Variáveis temporárias incentivam seu uso prolongado por terem um escopo limitado

Mais um exemplo

E o desempenho?

"Devemos esquecer as pequenas eficiências em 97% do tempo: otimização prematura é a raiz de todo o mal."

Donald Knuth

- Usar *profiling* para encontrar gargalos
- Otimizar apenas os gargalos
- Código fatorado:
  - Compra tempo para otimizar
  - Aumenta a precisão na otimização

Dois chapéus

- Refatoração

- Nova funcionalidade
- Correção de erros

Mais exemplos: sugestões?

Problemas com refatorações

- Refatoração de sistemas grandes ou enormes
- Refatoração com bancos de dados
- Refatoração de APIs públicas

- Quando **`não`** refatorar?
  - Quando é tão ruim que reescrever é melhor
  - Quando você está próximo de um prazo

Dúvidas?

Mariana Bravo
marivb@agilcoop.org.br

Referências

- M. Fowler, *"Refactoring, Improving the design of existing code"*, Addison-Wesley Professional, 1999
- J. Kerievsky, *"Refactoring to Patterns"*, Addison-Wesley Professional, 2004
- S. Ambler, P. Sadalage, *"Refactoring Databases: Evolutionary Database Design"*, Addison-Wesley Professional, 2006
- www.refactoring.com