Refatoramento de Código

Exemplo extraído do livro *Refactoring: Improving the Design of Existing Code* (Fowler, 1999)

Parte 4: criando um extrato com saída em formato HTML

I. Reflexão sobre as mudanças realizadas até aqui

Após realizar uma série de ajustes no código até a última tarefa deste exercício de refatoramento, você deve ter percebido que o código ficou **maior** depois das últimas mudanças. É verdade sim... porém, todas as modificações foram positivas sob as seguintes justificativas:

a) Criamos dois métodos úteis que poderão ser usados mais na frente (**`getValorTotal()` e `getPontosTotaisDeAlugadorFrequente()`**);

b) Tais métodos podem até se tornarem públicos se for necessário que entrem na interface pública de métodos da classe;

c) O código do método **`extrato()`** ficou menor e mais organizado, pois cada pedaço que define as etapas da lógica ficaram "enxutos";

Como contraponto, alguém pode alegar que o código terá o seu desempenho afetado, haja vista que possui mais **loops** do que antes. Essa é uma questão que deve ser discutida a parte, fora do escopo desta tarefa, pois depende de um cenário em que podemos ter muitos clientes e um alto fluxo de locação de carros. Se este não for o caso, os loops adicionais só adicionarão alguns milissegundos ao processamento.

Diante da reflexão apresentada, vejamos como fica fácil criar um método extrato em HTML devido à existência dos dois métodos criados (embora ainda não seja a solução definitiva). Copie o código do método **`extratoHTML()`** em sua classe **`Cliente`**.

```
public class Cliente(){
   .
   .
   .
   public String extratoHTML() {
      final String fimDeLinha = System.getProperty("line.separator");
      int sequencia = 0;

      Iterator<Locacao> locacoes = carrosAlugados.iterator();
      String resultado = "<html><body>"+ fimDeLinha;
      resultado = String.format("<H2>Registro de Locacoes de <EM> %s
</EM></H2>", getNome()) + fimDeLinha;
      resultado += "<table
border=\"1\"><tr><th>Seq</th><th>Automóvel</th><th>Ano</th><th>Diárias</th><t
h>Valor</th></tr>" + fimDeLinha;
```

```
      while(locacoes.hasNext()) {
            Locacao cada = locacoes.next();

            // mostra valores para este Locacao
            sequencia++:
             resultado +=
String.format("<tr><th>%02d.</th><th>%s</th><th>%4d</th><th>%2d</th><th>R$
%8.2f</th></tr>" + fimDeLinha,  sequencia,
cada.getCarro().getDescricao(),cada.getCarro().getAno(),
cada.getDiasAlugado(), cada.valorDeUmaLocacao() );
      } // while

      // adiciona rodapé
      resultado += String.format("<tfoot><tr><td colspan=\"4\">Valor
Acumulado em diárias:</td><td><EM>R$ %8.2f</EM></td></tr></tfoot></table>" +
fimDeLinha, getValorTotal());
      resultado += "<P>Voce acumulou <EM>" +
getPontosTotaisDeAlugadorFrequente() +
            " pontos </EM> de alugador frequente</p></body></html> ";

      return resultado;
   }
}
```

II. Reavaliando decisões tomadas anteriormente

Ao examinarmos o primeiro código do método **extrato(),** percebemos que o cálculo do valor da locação de um **Automovel** deveria ser uma responsabilidade da classe **Locacao**. Vamos analisar novamente a estrutura **switch** para fazermos o seguinte questionamento:

*"O teste do switch está sendo feito em cima dos próprios dados da classe **Locacao** ou em cima dos dados de **outro objeto**?"*.

Podemos perceber que a classe **Locacao** faz um **switch** em cima de dados do **Automovel**. Portanto, faz mais sentido mover o método **valorDeUmaLocacao()** para a classe **Automovel**. Seguindo esse mesmo raciocínio, podemos também transferir a responsabilidade de calcular os pontos de alugador frequente para a classe **Automovel**.

III. Tarefa (refactoring3)

Mova os métodos **valorDeUmaLocacao()** e **getPontosDeAlugadorFrequente()** para a classe **Automovel**. Na classe **Locacao**, mantenha o método de mesmo nome, porém, delegue essa responsabilidade chamando o método correspondente na classe **Automovel**. Realize os ajustes necessários para poder rodar novamente o código. A saída deve ser a mesma e a classe **Locadora** não deve ser afetada pelas mudanças.