Refatoração: Cadastro de Consumidor

Apresente o resultado da refatoração:

–Finalidade da rotina:

Fizemos várias refatorações. Nesse documento destacamos a refatoração da rotina de cadastro de consumidor.
Essa rotina obtém dados informados pelo consumidor vindos de um método POST para o ServletCadastroUsuario, as informações são validadas e gravadas no banco de dados.

–Problemas apresentados nessa rotina que a tornaram alvo de refatoração.

Os problema principais da rotina são:
Pouca clareza;
Pouca simplicidade.

```
96
97          try {
98              if(valida.validarCPF(consumidor.getCpfConsumidor()) == true){
99                  stmt = conexao.createStatement();
100
101                 /**Grava endereço no banco de dados*/
102                 stmt.executeUpdate(enderecoDAO.inserirEndereco(consumidor.getEndereco().getLogradouro(), consumidor.getEndereco().getNumero(),
103                         consumidor.getEndereco().getBairro(), consumidor.getEndereco().getComplemento(), consumidor.getEndereco().getCep(),
104                         consumidor.getEndereco().getCidade(), consumidor.getEndereco().getEstado()));
105
106                 /**Pego o id do endereço gravado antes*/
107                 resultado = stmt.executeQuery(enderecoDAO.selecionarIdEndereco(consumidor.getEndereco().getNumero(), consumidor.getEndereco().getCep(),
108                         consumidor.getEndereco().getCidade()));
109
110                 /**Gravo o consumidor no banco de dados com id do endereço pegado na string sql anterior*/
111                 stmt.executeUpdate(consumidorDAO.inserirConsumidor(consumidor.getCpfConsumidor(), consumidor.getEmail(), consumidor.getSenha(), consumidor.getNomeCompleto(),
112                         consumidor.getTelefone(), consumidor.getCelular(), consumidor.getTipoPerfil(), Integer.parseInt(resultado.getString("id"))));
113
114                 conexao.commit(); //fecha transação, efetiva comandos
115                 stmt.close(); //fecha statement
116                 conexao.close(); //fecha conexao com o banco de dados
117             }
118             else{
119                 throw new SQLException("CPF inválido!"); //cria uma exceção
120             }
121         } catch (SQLException e) {
122             System.out.println("Erro em cadastrar consumidor: "+ e.getMessage() +"\nCodigo do erro: "+ e.getErrorCode());
123             conexao.rollback(); //desfaz toda a transação caso haja exceção
124         }
125     }
126
127     /**
```

Imagem 1 - Parte do cadastro do consumidor a refatorar

ServletCadastroUsuario.java | ValidaDado.java | ConsumidorDAO.java | ProdutoDAO.java | ServletCadastroProduto.java

```
1  /**
4  package br.com.mercadofacil.jdbc;
5
6  /**
7   * @author weryquessantos
8   *
9   */
10 public class ConsumidorDAO {
11     public String inserirConsumidor(String cpf, String email, String senha, String nome, String telefone, String celular, String tipoPerfil, int idEndereco){
12         String insertConsumidor = "INSERT INTO mercadofacil.consumidor ("
13                 + "cpf, email, senha, nome, telefone, celular, tipoPerfil, idEndereco) "
14                 + "VALUES('"+ cpf +"', '"+ email +"', md5('"+ senha +"'), '"+ nome +"', '"+ telefone +"', '"+ celular +"', '"+ tipoPerfil +"', '"+ idEndereco +"')";
15
16         return insertConsumidor;
17     }
18 }
```

Imagem 2 - Classe ConsumidorDAO

–Passos da refatoração que foi realizada.

Na imagem 2 podemos ver: quando o método de inserir é chamado, ele recebe muitos parâmetros, esses parâmetros são montados em uma String SQL e depois essa String é retornada e executada no Servlet (imagem 1) pelo método executeUpdate.

Mudamos basicamente a forma em que o método de inserção de consumidor no banco de dados recebe os parâmetros. Em vez de mandarmos todos esses parâmetros, mandaremos apenas o instância de Consumidor.

O método de executeUpdate também passa ser de responsabilidade do ConsumidorDAO.

–Em que a rotina ficou “melhor” após a realização da refatoração.

Após a refatoração obtivemos mais clareza no código, mais simplicidade e diminuímos linhas de código.

```
64    private void cadastrarConsumidor(ServletRequest req) throws SQLException {
65        FabricaConexao conn = new FabricaConexao();
66        Consumidor consumidor = new Consumidor();
67        Connection conexao = null;
68
69        /**Abre conexão*/
70        conexao = conn.getConexao();
71
72        //Abrir transação
73        conexao.setAutoCommit(false); //A transação não "commita" sozinha
74
75        consumidor.getEndereco().setCep(req.getParameter("cep"));
76        consumidor.getEndereco().setCidade(req.getParameter("cidade"));
77        consumidor.getEndereco().setEstado(req.getParameter("estado"));
78        consumidor.getEndereco().setBairro(req.getParameter("bairro"));
79        consumidor.getEndereco().setLogradouro(req.getParameter("logradouro"));
80        consumidor.getEndereco().setComplemento(req.getParameter("complemento"));
81        consumidor.getEndereco().setNumero(Integer.parseInt(req.getParameter("numero")));
82
83        consumidor.setCpfConsumidor(req.getParameter("cpf"));
84        consumidor.setNomeCompleto(req.getParameter("nomeCompleto"));
85        consumidor.setEmail(req.getParameter("email"));
86        consumidor.setSenha(req.getParameter("senha"));
87        consumidor.setTelefone(req.getParameter("telefone"));
88        consumidor.setCelular(req.getParameter("celular"));
89        consumidor.setTipoPerfil("tipoPerfil");
90
91        ConsumidorDAO consumidorDAO = new ConsumidorDAO();
92        EnderecoDAO enderecoDAO = new EnderecoDAO();
93        ValidaDado valida = new ValidaDado();
94
95        try {
96            if(valida.validarCPF(consumidor.getCpfConsumidor()) == true){
97
98                //grava endereco
99                enderecoDAO.inserirEndereco(consumidor.getEndereco());
100
101               int idEndereco = enderecoDAO.selecionarIdEndereco(consumidor.getEndereco());
102
103               //grava consumidor
104               consumidorDAO.inserirConsumidor(consumidor, idEndereco);
105
106               conexao.commit(); //fecha transação, efetiva comandos
107               conexao.close(); //fecha conexao com o banco de dados
108           }
109           else{
110               throw new Exception("CPF inválido!"); //cria uma exceção
111           }
112       } catch (Exception e) {
113           System.out.println("Erro em cadastrar consumidor: "+ e.getMessage());
114           conexao.rollback(); //desfaz toda a transação caso haja exceção
115       }
116   }
```

Imagem 3 - Cadastro de Consumidor após refatorar

```
15  public class ConsumidorDAO {
16      public void inserirConsumidor(Consumidor consumidor, int id) throws SQLException{
17
18          String insertConsumidor = "INSERT INTO mercadofacil.consumidor ("
19                  + "cpf, email, senha, nome, telefone, celular, tipoPerfil, idEndereco) "
20                  + "VALUES('"+ consumidor.getCpfConsumidor() +"', '"+ consumidor.getEmail() +"', md5('"+ consumidor.getSenha() +"'), "
21                          + "'"+ consumidor.getNomeCompleto() +"', '"+ consumidor.getTelefone() +"', '"+ consumidor.getCelular() +"','"
22                                  + " '"+ consumidor.getTipoPerfil() +"', '"+ id +"')";
23
24          Statement stmt = null;
25
26          stmt.executeQuery(insertConsumidor);
27      }
28  }
```

Imagem 4 - Consumidor DAO