Universidade Federal de Minas Gerais

Programação Orientado a Objetos

Trabalho Prático 1

Documentação

Davi de Lima Rolim
Lucas Mol Holmquist

Saturday 2nd June, 2018

No arquivo estão anexados dentro da pasta Banco em src, os pacotes **banco** e **interfaceBanco**, que contém os arquivos .java referentes às classes criadas para realizar o trabalho. Dentro da pasta Banco está o arquivo onde se encontram os dados que são armazenados no programa. Tal arquivo possui o nome do banco, no nosso exemplo, **"Bradesco.txt"**.

Ao longo do código utilizamos vários métodos privados auxiliares para facilitar a implementação de certas funções, como para procurar uma conta de um banco dado o seu número e para imprimir um extrato de movimentações.

Os códigos foram feitos utilizando a IDE Eclipse, na ausência dessa, para executar o programa basta abrir o arquivo **Banco.jar** pela linha de comando, usando o comando **"java -jar Banco.jar"**.

A classe interface possui todas as funções necessárias para fazer a integração entre o usuário e o banco sem ser necessário preocupação sobre como as funções do banco são executadas, há independência entre as classes. Interface recebe um argumento **Banco** em seu construtor para a partir dele executar as funções já implementadas nessa classe.

O main se encontra na classe interface, conforme o código abaixo.

```
1
2  public static void main(String[] args) {
3
4      Banco bradesco = new Banco("Bradesco");
5      bradesco.leituraDadosArquivo();
6
7      Interface Iterface_bradesco = new Interface(bradesco);
8      Iterface_bradesco.menu();
9
10   }
```

Código 1: Função main dentro da classe Interface.

Este funciona da seguinte forma: inicialmente um banco é inicializado com o nome Bradesco. O nome do banco coincide com o nome do arquivo de armazenamento txt. Em seguida é feita a leitura do arquivo txt, com o método "leituraDadosArquivo()". Logo após, a classe interface é inicializada, recebendo como parâmtro o banco criado. Finalmente, o menu é aberto usando o método "menu()"pertencente à classe interface. Dentro de menu, através de um switch-case é escolhido o método que vai ser executado e ao fim de cada método há um comando para executar "menu()"outra vez, de modo que o programa é finalizado somente quando for selecionada a opção de deixar o programa. Ao receber o comando para que o programa se encerre, os dados do banco são armazenados no arquivo e ele chega ao fim. Vale ressaltar que os dados só são armazenados no arquivo quando o programa é encerrado adequadamente, com a tecla "q".

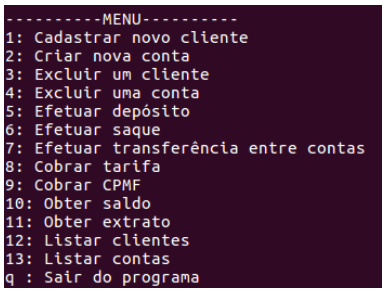

Figura 1: Menu exibido no console. Usuário escolhe o número correspondente à ação e aperta enteder para executá-la.

As classes com seus respectivos atributos e métodos são escritos de forma ilustrativa(a implementação é desconsiderada) nos códigos a seguir:

```
1  package banco;
2
3  public class Cliente {
4
5    private String nomeCliente;
6    private String cpf_cnpj;
7    private String endereco;
8    private String fone;
9
10   //Construtor
11   public Cliente(String nome, String c, String end, String f)
12
13   public String getNomeCliente()
14   public void setNomeCliente(String nomeCliente)
15   public String getCpf_cnpj()
16   public void setCpf_cnpj(String cpf_cnpj)
17   public String getEndereco()
18   public void setEndereco(String endereco)
19   public String getFone()
20   public void setFone(String fone)
21 }
```

Código 2: Classe Cliente

```
1  package banco;
2
3  import java.util.*;
4
5  public class Conta {
6    private int numConta;
7    private double saldo;
8    private Cliente cliente;
9    private static int proximoNumConta = 1;
10   private ArrayList<Movimentacao> listaMov = new ArrayList<Movimentacao>();
11
12   Conta(Cliente cliente)
13   //Construtor adicionado para auxiliar no processo de leitura
14   Conta(Cliente cliente, String saldo,ArrayList<Movimentacao> listaM)
15   public int getNumConta()
16   public double getSaldo()
17   public Cliente getCliente()
18   public ArrayList<Movimentacao> getListaMov()
19   public static int getNumeroDeContas()
20   public void debitaConta(double valor, String descricao)
21   public void creditaConta(double valorCreditado, String descricaoMovimentacao)
22   public ArrayList<Movimentacao> extrato(GregorianCalendar dataInicial)
23   public ArrayList<Movimentacao> extrato(GregorianCalendar dataInicial, GregorianCalendar
24   public ArrayList<Movimentacao> extrato()
25 }
```

Código 3: Classe Conta

```
1  package banco;
2
3  import java.util.GregorianCalendar;
4
5  public class Movimentacao {
6
7    private GregorianCalendar dataMov;
8    private String descricao;
9    private char debitoCredito;
10   private double valor;
11
12   Movimentacao(String desc, char debCre, double val)
13   //Construtor adicionado apenas para auxiliar na leitura de dados de arquivo
14   Movimentacao(String desc, char debCre, double val, GregorianCalendar data)
15   public GregorianCalendar getDataMov()
16   public String getDescricao()
17   public char getDebitoCredito()
18   public double getValor()
19 }
```

Código 4: Classe Movimentacao

```java
package banco;

import java.nio.charset.Charset;
import java.nio.charset.StandardCharsets;
import java.nio.file.*;
import java.io.*;
import java.util.*;

public class Banco {
  private String nomeBanco;
  private ArrayList<Cliente> listaClientes = new ArrayList<Cliente>();
  private ArrayList<Conta> listaContas = new ArrayList<Conta>();

  public Banco(String nomeBanco)
  //Método adicionado para simplificar a implementação de vários métodos da classe banco
  private Conta procuraConta(int numeroConta)
  //Método adicionado para auxiliar no processo de leitura do arquivo
  private void carregarConta(Cliente cliente, String saldo, ArrayList<Movimentacao> lMov)
  public void insereCliente(Cliente cliente)
  public void criaConta(Cliente cliente)
  public void excluiCliente(String cpfOuCnpj)
  public void excluiConta(int numeroConta)
  public void depositaConta(int numeroConta, double valor)
  public void saqueConta(int numeroConta, double valor)
  public void transferencia(int numeroContaOrigem, int numeroContaDestino, double valor)
  public void cobraTarifa()
  public void cobraCPMF()
  public double saldoConta(int numeroConta)
  public ArrayList<Movimentacao> extratoConta(int numeroConta)
  public ArrayList<Movimentacao> extratoConta(int numeroConta,
  GregorianCalendar dataInicial)
  public ArrayList<Movimentacao> extratoConta(int numeroConta,
  GregorianCalendar dataInicial, GregorianCalendar dataFinal)
  public ArrayList<Cliente> getListaCliente()
  public ArrayList<Conta> getListaContas()
  public String getNomeBanco()
  public void gravarDadosArquivo()
  public void leituraDadosArquivo()
}
```

Código 5: Classe Banco

```java
package interfaceBanco;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.Scanner;

import banco.Banco;
import banco.Cliente;
import banco.Conta;
import banco.Movimentacao;

public class Interface {

  //Impletação de main
  public static void main(String[] args) {

    Banco bradesco = new Banco("Bradesco");
    bradesco.leituraDadosArquivo();

    Interface Iterface_bradesco = new Interface(bradesco);
    Iterface_bradesco.menu();

  }
  //Atributo adicionado para auxiliar na implementação de menu
  private static int menuAberto = 0;
  // Atritubo bando da classe Interface
  private Banco banco;
  //Construtor
  Interface(Banco b)
  public void cadastraCliente()
  public void excluiCliente()
  public void criaConta()
  public void excluiConta()
  public void cobraTarifa()
  public void cobraCPMF()
  public void obtemSaldo()
  private void imprimeExtrato(ArrayList<Movimentacao> extrato)
  public void obtemExtrato()
  public void efetuarDeposito()
  public void efetuarSaque()
  public void efetuarTransferencia()
  public void listarClientes()
  public void listarContas()
  public void menu()
  //Método adicionado para simplificar método menu
  private void jumpSpace()
}
```

Código 6: Classe Interface