

Missão prática | Nível 1 | Mundo 3



Estácio

CAMPUS: POLO DISTRITO JK - ANÁPOLIS - GO

CURSO: DESENVOLVIMENTO FULL STACK

NÚMERO DA TURMA: 2024.3

SEMESTRE LETIVO: Segundo semestre de 2024

ALUNO: Henrique Rodrigues Rabello Vieira

MATRÍCULA: 202301230527

RPG0014 – Iniciando o caminho pelo Java

Objetivos:

- Utilizar herança e polimorfismo na definição de entidades.
- Utilizar persistência de objetos em arquivos binários.
- Implementar uma interface cadastral em modo texto.
- Utilizar o controle de exceções da plataforma Java.

Segundo procedimento:

Códigos solicitados:

Opções:

```
1 package cadastrpoo;
2
3 import java.io.IOException;
4 import java.util.ArrayList;
5 import java.util.Scanner;
6 import model.PessoaFisica;
7 import model.PessoaFisicaRepo;
8 import model.PessoaJuridica;
9 import model.PessoaJuridicaRepo;
10
11
12 public class CadastroPOOsegunda {
13
14     public static void main(String[] args) throws IOException, ClassNotFoundException {
15
16         Scanner in = new Scanner(System.in);
17         PessoaFisicaRepo repoPFisica = new PessoaFisicaRepo();
18         PessoaJuridicaRepo repoPJuridica = new PessoaJuridicaRepo();
19
20         do{
21             System.out.println("=====");
22             System.out.println("1 - Incluir Pessoa");
23             System.out.println("2 - Alterar Pessoa");
24             System.out.println("3 - Excluir Pessoa");
25             System.out.println("4 - Buscar pelo Id");
26             System.out.println("5 - Exibir Todos");
27             System.out.println("6 - Persistir Dados");
28             System.out.println("7 - Recuperar Dados");
29             System.out.println("0 - Finalizar Programa");
30             System.out.println("=====");
31
32             int input = in.nextInt();
```

Opção incluir:

```
34         switch(input){
35             case 1) -> {
36                 System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
37                 String pessoaTipoI = in.next();
38                 if(pessoaTipoI.equals("F")){
39                     System.out.println("Insira os dados...");
40
41                     System.out.print("Nome: ");
42                     String nome = in.next();
43
44                     System.out.print("CPF: ");
45                     String cpf = in.next();
46
47                     System.out.print("Idade: ");
48                     int idade = in.nextInt();
49
50                     int id = repoPFisica.obterTodos().size() + 1;
51
52                     PessoaFisica pessoaFNova = new PessoaFisica(id, nome, cpf, idade);
53                     repoPFisica.inserir(pessoaFNova);
54
55                     System.out.println("Dados adicionados com sucesso.");
56                 }
57                 else if(pessoaTipoI.equals("J")){
58                     System.out.println("Insira os dados...");
59
60                     System.out.print("Nome: ");
61                     String nome = in.next();
62
63                     System.out.print("CNPJ: ");
64                     String cnpj = in.next();
65
66                     int id = repoPJuridica.obterTodos().size() + 1;
67
68                     PessoaJuridica pessoaJNova = new PessoaJuridica(id,nome, cnpj);
69                     repoPJuridica.inserir(pessoaJNova);
70                     System.out.println("Dados adicionados com sucesso.");
71                 }
72             }
73         }
```

Opção alterar:

```

74         case(2) -> {
75             System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
76             String pessoaTipoA = in.next();
77             if(pessoaTipoA.equals("F")){
78                 System.out.print("Insira um id: ");
79                 int id = in.nextInt();
80                 try{
81                     PessoaFisica pessoaAlterar = repoPFisica.obter(id);
82                     pessoaAlterar.exibir();
83
84                     System.out.println("Insira os dados...");
85
86                     System.out.print("Novo Nome: ");
87                     String nome = in.next();
88
89                     System.out.print("Novo CPF: ");
90                     String cpf = in.next();
91
92                     System.out.print("Nova Idade: ");
93                     int idade = in.nextInt();
94
95                     PessoaFisica pessoaFNova = new PessoaFisica(id, nome, cpf, idade);
96                     PessoaFisicaRepo pessoaFisicaRepo = new PessoaFisicaRepo();
97                     pessoaFisicaRepo.alterar(pessoaAlterar, pessoaFNova);
98                     pessoaAlterar.exibir();
99                     System.out.print("Dados alterados com sucesso.");
100                 }catch(Throwable e){
101                     System.out.println("Erro ao obter dados: " + e.getMessage());
102                     e.printStackTrace();
103                 }
104             }

```

```

104         }
105         else if(pessoaTipoA.equals("J")){
106
107             System.out.print("Insira um id: ");
108             int id = in.nextInt();
109             try{
110                 PessoaJuridica pessoaAlterar = repoPJuridica.obter(id);
111                 pessoaAlterar.exibir();
112
113                 System.out.println("Insira os dados...");
114
115                 System.out.print("Novo Nome: ");
116                 String nome = in.next();
117
118                 System.out.print("Novo CNPJ: ");
119                 String cnpj = in.next();
120
121                 PessoaJuridica pessoaJNova = new PessoaJuridica(id, nome, cnpj);
122                 PessoaJuridicaRepo pessoaJuridicaRepo = new PessoaJuridicaRepo();
123                 pessoaJuridicaRepo.alterar(pessoaAlterar, pessoaJNova);
124                 pessoaAlterar.exibir();
125                 System.out.print("Dados alterados com sucesso.");
126             }catch(Throwable e){
127                 System.out.println("Erro ao obter dados: " + e.getMessage());
128                 e.printStackTrace();
129             }
130         }
131     }

```

Opção excluir:

```

131     }
132     case(3) -> {
133         System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
134         String pessoaTipoE = in.next();
135         if(pessoaTipoE.equals("F")){
136             System.out.print("Insira um id: ");
137             int id = in.nextInt();
138             try{
139                 repoPFisica.excluir(id);
140                 System.out.println("Dados removidos.");
141             }catch(Throwable e){
142                 System.out.println("Erro ao excluir dados: " + e.getMessage());
143                 e.printStackTrace();
144             }
145         }
146         else if(pessoaTipoE.equals("J")){
147             System.out.print("Insira um id: ");
148             int id = in.nextInt();
149             try{
150                 repoPJuridica.excluir(id);
151                 System.out.println("Dados removidos.");
152             }catch(Throwable e){
153                 System.out.println("Erro ao excluir dados: " + e.getMessage());
154                 e.printStackTrace();
155             }
156         }
157     }
158 }

```

Opção obter:

```

159 case(4) -> {
160     System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
161     String pessoaTipoO = in.next();
162     if(pessoaTipoO.equals("F")){
163         System.out.print("Insira um id: ");
164         int id = in.nextInt();
165
166         try{
167             PessoaFisica pessoaObtida = repoPFisica.obter(id);
168             pessoaObtida.exibir();
169         }catch(Throwable e){
170             System.out.println("Erro ao obter dados: " + e.getMessage());
171             e.printStackTrace();
172         }
173     }
174     else if(pessoaTipoO.equals("J")){
175         System.out.print("Insira um id: ");
176         int id = in.nextInt();
177
178         try{
179             PessoaJuridica pessoaObtida = repoPJuridica.obter(id);
180             pessoaObtida.exibir();
181         }catch(Throwable e){
182             System.out.println("Erro ao obter dados: " + e.getMessage());
183             e.printStackTrace();
184         }
185     }
186 }

```

Opção obter todos:

```

187 case(5) -> {
188     System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
189     String pessoaTipoOT = in.next();
190     if(pessoaTipoOT.equals("F")){
191         ArrayList<PessoaFisica> pessoaObtida = repoPFisica.obterTodos();
192         for(PessoaFisica p: pessoaObtida){
193             p.exibir();
194         }
195     }
196     else if(pessoaTipoOT.equals("J")){
197         ArrayList<PessoaJuridica> pessoaObtida = repoPJuridica.obterTodos();
198         for(PessoaJuridica p: pessoaObtida){
199             p.exibir();
200         }
201     }
202 }
203 }
204 }

```

Opção salvar:

```

205 case(6) -> {
206     System.out.print("Insira um prefixo para salvar o arquivo: ");
207     String arquivoNomeS = in.next();
208     try{
209         repoPFisica.persistir(arquivoNomeS + ".fisica.bin");
210         repoPJuridica.persistir(arquivoNomeS + ".juridica.bin");
211         System.out.println("Dados salvos com sucesso.");
212     }catch(IOException e){
213         System.out.println("Erro ao salvar dados: " + e.getMessage());
214         e.printStackTrace();
215     }
216 }

```

Opção Recuperar:

```

218 case(7) -> {
219     System.out.print("Insira um prefixo para recuperar o arquivo: ");
220     String arquivoNomeR = in.next();
221     try{
222         repoPFisica.recuperar(arquivoNomeR + ".fisica.bin");
223         repoPJuridica.recuperar(arquivoNomeR + ".juridica.bin");
224         System.out.println("Dados recuperados com sucesso.");
225     }catch(IOException | ClassNotFoundException e){
226         System.out.println("Erro ao recuperar dados: " + e.getMessage());
227         e.printStackTrace();
228     }
229 }
230 }

```

Resultados:

```
run:
=====
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
=====
```

Análise e conclusão:

O que são elementos estáticos e qual o motivo para o método main adotar esse modificador?

Elementos estáticos são elementos que pertencem a classe e podem ser acessados sem a criação de um objeto. O método main adota esse modificador para que possa ser acessado/utilizado sem ser instanciado.

Para que serve a classe Scanner?

Utilizada para ler dados de entrada, muito usada para permitir interações entre o usuário e o programa.

Como o uso de classes de repositório impactou na organização do código?

O uso de classes repositório impactou positivamente a organização do código deixando ele mais legível.