



Nome do Campus: Polo Castelo

Nome do Curso: Desenvolvimento FullStack

Nome da Disciplina: Iniciando o caminho pelo Java

Número da Turma: 2025.1

Semestre Letivo: Primeiro Semestre

Nome do Aluno: Roberto Birro Alves Filho

Matrícula: 202402849204

Segundo Procedimento:

Criação do cadastro em modo texto

Título da Prática

Implementação de um cadastro de clientes em modo texto, com persistência em arquivos, baseado na tecnologia Java.

Objetivo da Prática

Utilizar herança e polimorfismo na definição de entidades:

Criar uma hierarquia de classes (Pessoa, PessoaFisica, PessoaJuridica) para demonstrar herança e polimorfismo.

Utilizar persistência de objetos em arquivos binários:

Implementar métodos para salvar e recuperar dados em arquivos binários usando serialização.

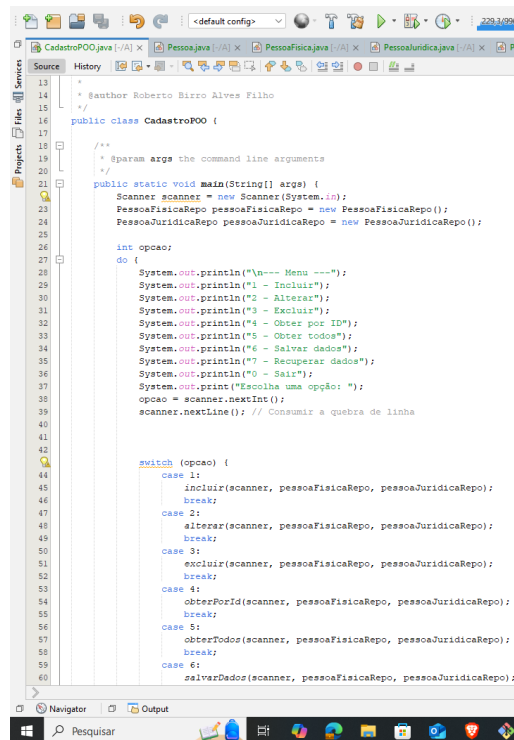
Implementar uma interface cadastral em modo texto:

Desenvolver um menu interativo para interação com o usuário via linha de comando.

Utilizar o controle de exceções da plataforma Java:

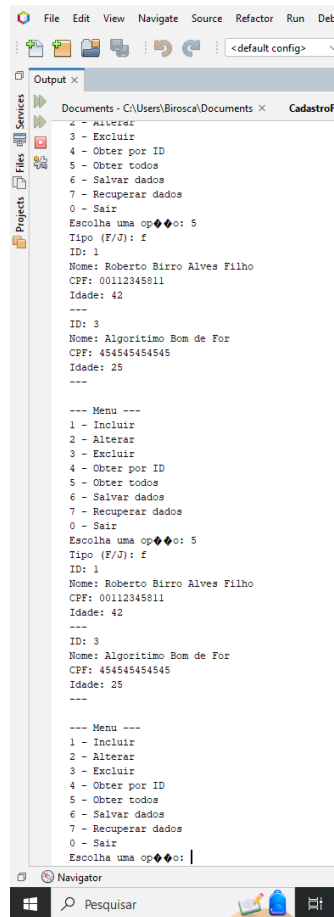
Tratar possíveis erros durante operações como persistência e recuperação de dados.

Classe Teste Principal:



```
13  *
14  * @author Roberto Birro Alves Filho
15  */
16  public class CadastroPOO {
17
18      /**
19       * @param args the command line arguments
20       */
21      public static void main(String[] args) {
22          Scanner scanner = new Scanner(System.in);
23          PessoaFisicaRepo pessoaFisicaRepo = new PessoaFisicaRepo();
24          PessoaJuridicaRepo pessoaJuridicaRepo = new PessoaJuridicaRepo();
25
26          int opcao;
27          do {
28              System.out.println("\n--- Menu ---");
29              System.out.println("1 - Incluir");
30              System.out.println("2 - Alterar");
31              System.out.println("3 - Excluir");
32              System.out.println("4 - Obter por ID");
33              System.out.println("5 - Obter todos");
34              System.out.println("6 - Salvar dados");
35              System.out.println("7 - Recuperar dados");
36              System.out.println("0 - Sair");
37              System.out.print("Escolha uma opção: ");
38              opcao = scanner.nextInt();
39              scanner.nextLine(); // Consumir a quebra de linha
40
41
42              switch (opcao) {
43                  case 1:
44                      incluir(scanner, pessoaFisicaRepo, pessoaJuridicaRepo);
45                      break;
46                  case 2:
47                      alterar(scanner, pessoaFisicaRepo, pessoaJuridicaRepo);
48                      break;
49                  case 3:
50                      excluir(scanner, pessoaFisicaRepo, pessoaJuridicaRepo);
51                      break;
52                  case 4:
53                      obterPorId(scanner, pessoaFisicaRepo, pessoaJuridicaRepo);
54                      break;
55                  case 5:
56                      obterTodos(scanner, pessoaFisicaRepo, pessoaJuridicaRepo);
57                      break;
58                  case 6:
59                      salvarDados(scanner, pessoaFisicaRepo, pessoaJuridicaRepo);
60                      break;
61              }
62          } while (opcao != 0);
63      }
64
65      private static void incluir(Scanner scanner, PessoaFisicaRepo pessoaFisicaRepo, PessoaJuridicaRepo pessoaJuridicaRepo) {
66          System.out.println("1 - Pessoa Física");
67          System.out.println("2 - Pessoa Jurídica");
68          System.out.print("Escolha o tipo de pessoa: ");
69          int tipo = scanner.nextInt();
70          scanner.nextLine();
71
72          if (tipo == 1) {
73              System.out.print("Nome: ");
74              String nome = scanner.nextLine();
75              System.out.print("CPF: ");
76              String cpf = scanner.nextLine();
77              System.out.print("Endereço: ");
78              String endereco = scanner.nextLine();
79              pessoaFisicaRepo.incluir(nome, cpf, endereco);
80          } else if (tipo == 2) {
81              System.out.print("Nome: ");
82              String nome = scanner.nextLine();
83              System.out.print("CNPJ: ");
84              String cnpj = scanner.nextLine();
85              System.out.print("Endereço: ");
86              String endereco = scanner.nextLine();
87              pessoaJuridicaRepo.incluir(nome, cnpj, endereco);
88          }
89      }
90
91      private static void alterar(Scanner scanner, PessoaFisicaRepo pessoaFisicaRepo, PessoaJuridicaRepo pessoaJuridicaRepo) {
92          System.out.println("1 - Pessoa Física");
93          System.out.println("2 - Pessoa Jurídica");
94          System.out.print("Escolha o tipo de pessoa: ");
95          int tipo = scanner.nextInt();
96          scanner.nextLine();
97
98          if (tipo == 1) {
99              System.out.print("ID: ");
100             String id = scanner.nextLine();
101             System.out.print("Nome: ");
102             String nome = scanner.nextLine();
103             System.out.print("CPF: ");
104             String cpf = scanner.nextLine();
105             System.out.print("Endereço: ");
106             String endereco = scanner.nextLine();
107             pessoaFisicaRepo.alterar(id, nome, cpf, endereco);
108         } else if (tipo == 2) {
109             System.out.print("ID: ");
110             String id = scanner.nextLine();
111             System.out.print("Nome: ");
112             String nome = scanner.nextLine();
113             System.out.print("CNPJ: ");
114             String cnpj = scanner.nextLine();
115             System.out.print("Endereço: ");
116             String endereco = scanner.nextLine();
117             pessoaJuridicaRepo.alterar(id, nome, cnpj, endereco);
118         }
119     }
120
121     private static void excluir(Scanner scanner, PessoaFisicaRepo pessoaFisicaRepo, PessoaJuridicaRepo pessoaJuridicaRepo) {
122         System.out.println("1 - Pessoa Física");
123         System.out.println("2 - Pessoa Jurídica");
124         System.out.print("Escolha o tipo de pessoa: ");
125         int tipo = scanner.nextInt();
126         scanner.nextLine();
127
128         if (tipo == 1) {
129             System.out.print("ID: ");
130             String id = scanner.nextLine();
131             pessoaFisicaRepo.excluir(id);
132         } else if (tipo == 2) {
133             System.out.print("ID: ");
134             String id = scanner.nextLine();
135             pessoaJuridicaRepo.excluir(id);
136         }
137     }
138
139     private static void obterPorId(Scanner scanner, PessoaFisicaRepo pessoaFisicaRepo, PessoaJuridicaRepo pessoaJuridicaRepo) {
140         System.out.println("1 - Pessoa Física");
141         System.out.println("2 - Pessoa Jurídica");
142         System.out.print("Escolha o tipo de pessoa: ");
143         int tipo = scanner.nextInt();
144         scanner.nextLine();
145
146         if (tipo == 1) {
147             System.out.print("ID: ");
148             String id = scanner.nextLine();
149             PessoaFisica pessoaFisica = pessoaFisicaRepo.obterPorId(id);
150             if (pessoaFisica != null) {
151                 System.out.println("Nome: " + pessoaFisica.getNome());
152                 System.out.println("CPF: " + pessoaFisica.getCpf());
153                 System.out.println("Endereço: " + pessoaFisica.getEndereco());
154             }
155         } else if (tipo == 2) {
156             System.out.print("ID: ");
157             String id = scanner.nextLine();
158             PessoaJuridica pessoaJuridica = pessoaJuridicaRepo.obterPorId(id);
159             if (pessoaJuridica != null) {
160                 System.out.println("Nome: " + pessoaJuridica.getNome());
161                 System.out.println("CNPJ: " + pessoaJuridica.getCnpj());
162                 System.out.println("Endereço: " + pessoaJuridica.getEndereco());
163             }
164         }
165     }
166
167     private static void obterTodos(Scanner scanner, PessoaFisicaRepo pessoaFisicaRepo, PessoaJuridicaRepo pessoaJuridicaRepo) {
168         System.out.println("1 - Pessoa Física");
169         System.out.println("2 - Pessoa Jurídica");
170         System.out.print("Escolha o tipo de pessoa: ");
171         int tipo = scanner.nextInt();
172         scanner.nextLine();
173
174         if (tipo == 1) {
175             List<PessoaFisica> pessoasFisicas = pessoaFisicaRepo.obterTodos();
176             for (PessoaFisica pessoaFisica : pessoasFisicas) {
177                 System.out.println("ID: " + pessoaFisica.getId());
178                 System.out.println("Nome: " + pessoaFisica.getNome());
179                 System.out.println("CPF: " + pessoaFisica.getCpf());
180                 System.out.println("Endereço: " + pessoaFisica.getEndereco());
181             }
182         } else if (tipo == 2) {
183             List<PessoaJuridica> pessoasJuridicas = pessoaJuridicaRepo.obterTodos();
184             for (PessoaJuridica pessoaJuridica : pessoasJuridicas) {
185                 System.out.println("ID: " + pessoaJuridica.getId());
186                 System.out.println("Nome: " + pessoaJuridica.getNome());
187                 System.out.println("CNPJ: " + pessoaJuridica.getCnpj());
188                 System.out.println("Endereço: " + pessoaJuridica.getEndereco());
189             }
190         }
191     }
192
193     private static void salvarDados(Scanner scanner, PessoaFisicaRepo pessoaFisicaRepo, PessoaJuridicaRepo pessoaJuridicaRepo) {
194         System.out.println("1 - Pessoa Física");
195         System.out.println("2 - Pessoa Jurídica");
196         System.out.print("Escolha o tipo de pessoa: ");
197         int tipo = scanner.nextInt();
198         scanner.nextLine();
199
200         if (tipo == 1) {
201             System.out.print("ID: ");
202             String id = scanner.nextLine();
203             System.out.print("Nome: ");
204             String nome = scanner.nextLine();
205             System.out.print("CPF: ");
206             String cpf = scanner.nextLine();
207             System.out.print("Endereço: ");
208             String endereco = scanner.nextLine();
209             pessoaFisicaRepo.salvarDados(id, nome, cpf, endereco);
210         } else if (tipo == 2) {
211             System.out.print("ID: ");
212             String id = scanner.nextLine();
213             System.out.print("Nome: ");
214             String nome = scanner.nextLine();
215             System.out.print("CNPJ: ");
216             String cnpj = scanner.nextLine();
217             System.out.print("Endereço: ");
218             String endereco = scanner.nextLine();
219             pessoaJuridicaRepo.salvarDados(id, nome, cnpj, endereco);
220         }
221     }
222 }
```

Resultado da execução:



```
Documents - C:\Users\Birosc\Documents x CadastroP
4 - Alterar
3 - Excluir
4 - Obter por ID
5 - Obter todos
6 - Salvar dados
7 - Recuperar dados
0 - Sair
Escolha uma opção: 5
Tipo (F/J): f
ID: 1
Nome: Roberto Birro Alves Filho
CPF: 00112345678
Idade: 42
---
ID: 3
Nome: Algoritmo Bom de For
CPF: 45454545454
Idade: 25
---
--- Menu ---
1 - Incluir
2 - Alterar
3 - Excluir
4 - Obter por ID
5 - Obter todos
6 - Salvar dados
7 - Recuperar dados
0 - Sair
Escolha uma opção: 5
Tipo (F/J): f
ID: 1
Nome: Roberto Birro Alves Filho
CPF: 00112345678
Idade: 42
---
ID: 3
Nome: Algoritmo Bom de For
CPF: 45454545454
Idade: 25
---
--- Menu ---
1 - Incluir
2 - Alterar
3 - Excluir
4 - Obter por ID
5 - Obter todos
6 - Salvar dados
7 - Recuperar dados
0 - Sair
Escolha uma opção: |
```

Análise e Conclusão

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

O que são elementos estáticos?

Elementos estáticos (static) pertencem à classe em vez de pertencerem a uma instância específica da classe.

Isso significa que eles podem ser acessados diretamente pelo nome da classe, sem a necessidade de criar um objeto da classe.

Exemplos de elementos estáticos:

Métodos estáticos (ex.: `Math.sqrt()`).

Variáveis estáticas (ex.: contadores globais).

Blocos estáticos (usados para inicialização).

Por que o método main é estático?

O método main é o ponto de entrada de qualquer programa Java. Ele precisa ser estático porque o Java não cria automaticamente uma instância da classe principal ao iniciar o programa.

Se o método main não fosse estático, seria necessário criar um objeto da classe principal antes de executar o programa, o que complicaria desnecessariamente o processo de inicialização.

Com o modificador static, o método main pode ser chamado diretamente pela JVM (Java Virtual Machine) sem a necessidade de instanciar a classe.

Relação com o projeto:

No seu projeto, o método main na classe Principal foi declarado como static para permitir que o sistema seja iniciado sem a necessidade de criar uma instância da classe Principal.

2. Para que serve a classe Scanner?

O que é a classe Scanner?

A classe Scanner faz parte do pacote java.util e é usada para capturar entradas do usuário via console (linha de comando).

Ela fornece métodos convenientes para ler diferentes tipos de dados, como texto (String), números inteiros (int), números de ponto flutuante (double), etc.

Como funciona?

Um objeto Scanner é criado associado à entrada padrão (System.in), que é o teclado.

Métodos como nextInt(), nextLine(), nextDouble() são usados para ler os dados inseridos pelo usuário.

Importância no projeto:

No menu interativo, a classe Scanner foi essencial para permitir que o usuário escolhesse opções, inserisse dados (como nome, CPF, CNPJ) e interagisse com o sistema.

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

O que são classes de repositório?

Classes de repositório (como PessoaFisicaRepo e PessoaJuridicaRepo) são responsáveis por gerenciar a persistência e recuperação de dados.

Elas encapsulam toda a lógica relacionada ao armazenamento (em memória ou em arquivos) e manipulação dos dados, separando-a da lógica de negócios e da interface com o usuário.

Impacto na organização do código:

Separação de Responsabilidades:

As classes de repositório centralizam as operações de CRUD (Create, Read, Update, Delete) e persistência, enquanto a classe Principal foca na interação com o usuário.

Isso torna o código mais modular e fácil de manter.

Reutilização de Código:

As classes de repositório podem ser reutilizadas em outros projetos ou partes do sistema sem a necessidade de reescrever a lógica de persistência.

Facilidade de Testes:

Como as operações de persistência estão isoladas nas classes de repositório, é mais fácil testá-las independentemente do restante do sistema.

Escalabilidade:

Caso seja necessário alterar a forma de persistência (ex.: migrar de arquivos binários para um banco de dados), basta modificar as classes de repositório, sem impactar o restante do código.

Exemplo no projeto:

A classe PessoaFisicaRepo gerencia todas as operações relacionadas às pessoas físicas, como inserir, alterar, excluir e persistir dados.

Isso permite que a classe Principal fique mais limpa e focada apenas no menu e na interação com o usuário.