Universidade: Estácio de Sá

Campus: Estácio Interlagos

Curso: Desenvolvimento Full Stack

Disciplina: Iniciando o caminho pelo Java

Número da Turma: 2023.3

Semestre Letivo: 3° Semestre

Integrantes da Prática: Lucas Henrique Silva Santos

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

Título da Prática: Criação das Entidades e Sistema de Persistência

Objetivo da Prática:

1. Utilizar herança e polimorfismo na definição de entidades.
2. Utilizar persistência de objetos em arquivos binários.
3. Implementar uma interface cadastral em modo texto.
4. Utilizar o controle de exceções da plataforma Java.
5. No final do projeto, o aluno terá implementado um sistema cadastral em Java, utilizando os recursos da programação orientada a objetos e a persistência em arquivos binários.

Códigos da Prática

Neste relatório, descrevi a segunda fase do projeto do sistema cadastral em Java, que envolve a criação de uma interface cadastral em modo texto. O projeto foi desenvolvido como parte de uma prática acadêmica com o objetivo de aprimorar nosso conhecimento em programação orientada a objetos, persistência de dados e interação com o usuário.

Implementação da Interface Cadastral em Modo Texto

A segunda fase do projeto envolveu a criação de uma interface de usuário em modo texto, que permite ao usuário interagir com o sistema de forma amigável e intuitiva. O programa oferece um menu de opções que permite ao usuário

executar diversas operações no sistema. Aqui estão as principais funcionalidades implementadas:

Incluir: O usuário pode adicionar novas entidades ao sistema, escolhendo entre Pessoa Física e Pessoa Jurídica e fornecendo os dados necessários. Esses dados são validados para garantir a integridade das informações.

Alterar: É possível modificar os dados de uma entidade existente, identificando-a pelo ID. O programa exibe os dados atuais e solicita novas informações ao usuário.

Excluir: O sistema permite a exclusão de uma entidade específica pelo seu ID, evitando a exclusão acidental.

Consultar por ID: O usuário pode consultar uma entidade pelo seu ID, e o programa exibe os detalhes dessa entidade.

Listar Todas as Entidades: O sistema permite listar todas as entidades cadastradas no sistema, seja Pessoa Física ou Pessoa Jurídica.

Salvar e Recuperar Dados: Implementei opções para salvar os dados em arquivos binários, permitindo que as informações sejam mantidas entre as execuções do programa. Também é possível recuperar os dados a partir desses arquivos.

Tratamento de Exceções

Durante a implementação da interface cadastral, foi dado destaque ao tratamento de exceções. Isso garante que o programa funcione de maneira robusta, mesmo em situações de erro. Foram implementados blocos try-catch

para capturar exceções e fornecer mensagens de erro adequadas ao usuário, evitando falhas inesperadas no programa.

Conclusão

A implementação da interface cadastral em modo texto tornou o sistema mais acessível e amigável ao usuário. Agora, o programa permite realizar operações de cadastro, alteração, exclusão e consulta de entidades de forma simples e eficaz. O tratamento de exceções garante a integridade do sistema em todas as situações.

Este projeto nos proporcionou uma valiosa experiência de desenvolvimento em Java, combinando conceitos teóricos com aplicação prática. Aprendemos a importância da interação com o usuário e a robustez do tratamento de exceções em sistemas reais.

Análise e Conclusão

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

Elementos Estáticos:

Elementos estáticos em Java, como variáveis e métodos, pertencem à classe em vez de uma instância específica dessa classe. São acessíveis diretamente a partir da classe, sem a necessidade de criar uma instância da classe. O modificador static é usado para definir elementos estáticos. Os elementos estáticos são compartilhados entre todas as instâncias da classe e podem ser acessados usando o nome da classe em vez do objeto. Isso é útil para criar variáveis ou métodos que são comuns a todas as instâncias da classe.

Método Main com Modificador Estático:

O método main em Java é o ponto de entrada para a execução de um programa. Ele é declarado como estático (public static void main(String[] args)) para que possa ser chamado sem a necessidade de criar uma instância da classe que

contém o método main. Isso permite que o sistema operacional ou a máquina virtual Java inicie o programa sem a necessidade de criar um objeto. O método main precisa ser estático porque é chamado antes que qualquer instância da classe seja criada.

Para que serve a classe Scanner?

A classe Scanner em Java é usada para ler dados do usuário ou de outras fontes de entrada, como arquivos. Ela fornece métodos para ler dados de vários tipos, como inteiros, números de ponto flutuante, strings, entre outros. A classe Scanner é amplamente utilizada para obter entrada do teclado durante a interação com o usuário em programas Java, tornando a leitura de dados mais simples e flexível.

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

A introdução das classes de repositório (por exemplo, PessoaFisicaRepo e PessoaJuridicaRepo) teve um impacto significativo na organização do código. Elas ajudaram a:

Separar Responsabilidades: As classes de repositório ficam encarregadas de gerenciar o armazenamento e recuperação de objetos, separando essa responsabilidade do restante do código.

Melhorar a Manutenção: As operações de persistência (salvar e recuperar) são encapsuladas nas classes de repositório, tornando as mudanças futuras nessas operações mais fáceis de serem realizadas sem afetar o restante do código.

Promover Reutilização: As classes de repositório podem ser reutilizadas em diferentes partes do código, permitindo que várias partes do programa acessem os mesmos dados de forma consistente.

Facilitar a Escalabilidade: À medida que o sistema cresce, a organização em classes de repositório ajuda a manter a estrutura do código organizada e escalável.

Em resumo, o uso de classes de repositório melhora a modularidade e a manutenção do código, além de promover boas práticas de programação ao separar as preocupações relacionadas ao armazenamento e recuperação de dados.

Repositório GIT: https: //github.com/LucasHSS904/CadastroPOO

Data de Elaboração: 06/09/2023