Documentação trabalho de OO

Savio Ribeiro

December 2023

App.java

Este é o ponto de entrada principal do aplicativo.

view

Contém todas as telas do sistema.

Herança

As classes `TelaTicket` e `TelaLogin` estendem a classe `JFrame` do Swing, demonstrando o uso de herança na construção de interfaces gráficas.

Polimorfismo

O método `actionPerformed` da interface `ActionListener` é sobrescrito em várias partes do código para lidar com eventos de botão, demonstrando o polimorfismo na manipulação de eventos.

Interfaces

As interfaces `ActionListener` e `ListSelectionListener` são implementadas para lidar com eventos de botão e seleção de listas, mostrando o uso de interfaces na interação com a interface gráfica.

Coleções

A classe `List` é usada para armazenar e manipular conjuntos de objetos `Ticket`, mostrando o uso de coleções para gerenciar dados na interface gráfica.

Tratamento de exceções

Blocos `try-catch` são utilizados para lidar com possíveis exceções durante a exclusão de um usuário ou ticket, demonstrando o tratamento de exceções na interação com a interface gráfica.

Interface gráfica

A biblioteca Swing é amplamente utilizada para criar a interface gráfica do usuário, incluindo janelas, botões, painéis e outros componentes.

persistence

Contém as classes relacionadas à persistência de dados.

Polimorfismo

As classes `UsuarioPersistence`, `TicketPersistence` e `CategoriaPersistence` implementam a interface `Persistence`, demonstrando o uso de polimorfismo na manipulação de diferentes tipos de objetos de persistência.

Coleções

As classes `UsuarioPersistence`, `TicketPersistence` e `CategoriaPersistence` usam listas de `Usuario`, `Ticket` e `Categoria`, respectivamente, em vários métodos, mostrando o uso de coleções para manipular dados de persistência.

Leitura e escrita em arquivos

As classes `UsuarioPersistence`, `TicketPersistence` e `CategoriaPersistence` realizam a leitura e escrita de dados em arquivos JSON, demonstrando o uso de leitura e escrita em arquivos para persistência de dados.

exception

Contém classes relacionadas ao tratamento de exceções.

Herança

A classe `ValidacaoException` estende a classe `Exception`, mostrando o uso de herança na criação de exceções personalizadas.

Tratamento de exceções

A classe `ValidacaoException` é lançada quando ocorre uma falha de validação, demonstrando o tratamento de exceções na validação do sistema.

model

Contém o CRUD de usuário e ticket.

Coleções

A classe `InitCategoria` usa uma lista de categorias, mostrando o uso de coleções para inicialização de dados.

Motivação para Utilização dos Conceitos

- **Herança:** Utilizada na classe `TelaTicket` e `TelaLogin` para herdar características básicas da classe `JFrame` do Swing, facilitando a construção de interfaces gráficas.
- **Polimorfismo:** Utilizado na implementação da interface `ActionListener` para lidar com diferentes eventos de botão de forma polimórfica.
- **Interfaces:** Amplamente utilizadas na interação com a interface gráfica, implementando interfaces como `ActionListener` e `ListSelectionListener` para tratar eventos.
- **Coleções:** Utilizadas em várias partes do código para armazenar e manipular conjuntos de objetos, seja na interface gráfica ou na persistência de dados.
- **Tratamento de exceções:** Implementado para lidar com possíveis falhas durante a exclusão de usuários ou tickets, melhorando a robustez do sistema.
- **Leitura e escrita em arquivos:** Utilizado nas classes de persistência para armazenar e recuperar dados em arquivos JSON, proporcionando persistência de dados entre execuções do aplicativo.
- **Interface gráfica:** Ampla utilização da biblioteca Swing para criar uma interface gráfica amigável e interativa.

Instruções para Compilar e Executar

1. Certifique-se de ter a ferramenta de build Swing instalada (por exemplo, Apache Ant ou Maven).
2. Navegue até o diretório raiz do projeto.
3. Execute o comando de compilação específico da ferramenta de build escolhida (por exemplo, `ant compile` para Apache Ant ou `mvn compile` para Maven).
4. Execute o aplicativo usando o comando apropriado da ferramenta de build (por exemplo, `ant run` para Apache Ant ou `mvn exec:java` para Maven). Certifique-se de especificar corretamente a classe `App` como ponto de entrada.

5. Certifique-se de ter as dependências corretas instaladas para a interface gráfica (por exemplo, a biblioteca Swing para Java).

Essas instruções podem variar dependendo da ferramenta de build específica que você escolheu para o seu projeto. Certifique-se de ajustar os comandos conforme necessário.