Linguagem de Programação II

Player - Uma aplicação JAVA baseada em um tocador MP3

Edna Juvêncio Nunes, Mateus Alves de Oliveira

Instituto Metrópole Digital – Universidade Federal do Rio Grande do Norte(UFRN)
Av. Cap. Mor Gouveia, S/N - Lagoa Nova, Natal - RN, 59078-900

edna.nunes.707@ufrn.edu.br, mateus.oliveira.111@ufrn.edu.br

Abstract. *The Media Player project is an MP3 player developed in Java, using JavaFX and Scene Builder for the graphical interface. It has access control for ordinary users and VIP users, where each one has different permissions. The project follows the MVC pattern and uses data structures such as lists to store and list directories and songs. It also makes use of exception handling, especially when reading files. The project was successfully implemented, including login functionality, music playback and an extra feature, the Progress Bar, to speed up music playback. Concepts such as inheritance, polymorphism, abstract classes, package organization and exception handling were applied. Overall, the project met the proposed requirements and objectives.*

Resumo. *O projeto Media Player é um tocador de MP3 desenvolvido em Java, utilizando JavaFX e Scene Builder para a interface gráfica. Ele possui controle de acesso para usuários comuns e usuários VIP, onde cada um tem permissões diferentes. O projeto segue o padrão MVC e utiliza estruturas de dados como listas para armazenar e listar diretórios e músicas. Também faz uso de tratamento de exceções, especialmente na leitura de arquivos. O projeto foi implementado com sucesso, incluindo a funcionalidade de login, reprodução de músicas e um recurso extra, o Progress Bar, para adiantar a reprodução da música. Foram aplicados conceitos como herança, polimorfismo, classes abstratas, organização de pacotes e tratamento de exceções. No geral, o projeto atendeu aos requisitos e objetivos propostos.*

1. Introdução

O projeto Media Player consiste em um tocador de MP3 capaz de produzir arquivos de áudio. Foi desenvolvido em Java, utilizando JavaFX e Scene Builder para interface. No projeto, está implementado e funcional a interface, o controle de acesso de usuários, sendo eles do tipo Comum e Vip, tal qual, os usuários do tipo Comum podem adicionar diretórios e ouvir apenas as músicas desse diretório. Já usuários Vip, podem adicionar diretórios, músicas, playlists e ainda, criar novos usuários e definir o seu tipo. Referente ao tocador, seleciona-se a música e depois clica em Play para começar. Além disso, foi implementado a pontuação extra no que se refere ao *Progress Bar* para adiantar a música.

2. Descrição da abordagem de solução do problema

O projeto possui os conceitos estudados tais como: Herança, Polimorfismo, Interface, Classes Abstratas, organizações de pacotes, tratamento de exceções e outros.
No que se refere a organização de pacotes, o projeto está seguindo o padrão MVC. Logo, temos, br.imd.controle, br.imd.modelo, br.imd.visao e data onde possui os arquivos .txt que são os arquivos do banco de dados. Dentro de br.imd.controle temos as seguintes classes: Main, DiretorioService, LoginService, MusicaService, UsuarioVipService; Em br.imd.modelo temos: Diretorio, Musica, Playlist, RetornoLogin, Usuario, UsuarioAdmin, UsuarioComum e UsuarioVip (todas as classes que iniciam com Usuario herdam a classe Usuario); Já dentro de br.imd.visao está organizado os arquivos .FXML das telas da interface, são elas: TelaCadastroUsuario.fxml, TelaLogin, TelaUsuario, TelaUsuarioVip; E por último, dentro de data estão os arquivos do tipo .txt simulando um banco de dados. Logo, possui logins que é o arquivo responsável por guardar os dados de login dos usuários; musicas, responsável por guardar os dados e caminho das músicas adicionadas; e o diretorios que segue a mesma lógica de armazenamento de músicas.

3. Descrição geral das estruturas de dados

Foi utilizado estruturas de dados tais como listas para armazenar e listar os diretórios e as músicas adicionadas no Player, como por exemplo:
Na classe MusicaService possui um atributo estático músicas, que é uma lista do tipo ArrayList<Musica>. Essa lista armazena objetos da classe Musica. A classe também possui uma variável estática musicaAtual, que é um inteiro usado para controlar a posição atual na lista de músicas.
O construtor MusicaService inicializa a lista de músicas.
O método estático getMusicas recebe um objeto Diretorio como parâmetro e retorna um ArrayList<Musica>. Esse método lê as músicas a partir de um arquivo de texto e as armazena na lista de músicas.

Além disso, utilizamos também tratamento de exceções e, um exemplo está aplicado na parte de controle de usuários onde a exceção IOException é tratada no método verificarCredenciais(String usuario, String senha). O trecho de código onde a exceção é tratada está dentro de um bloco try-catch. Quando o código tenta ler o arquivo de logins (logins.txt) utilizando o BufferedReader, pode ocorrer uma exceção IOException se houver algum problema ao abrir ou ler o arquivo. Caso aconteça, o fluxo de controle é transferido para o bloco catch correspondente, onde a exceção é tratada.

4. Gráfico UML

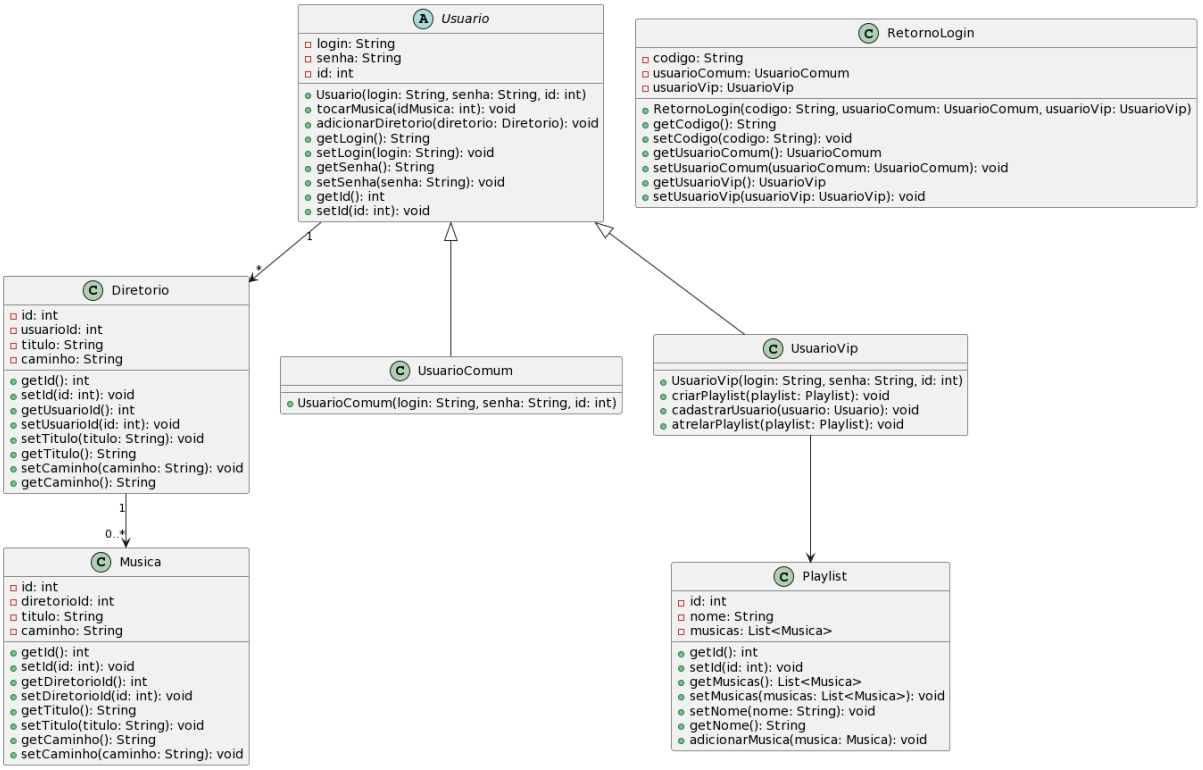

Figura 1. Gráfico UML das classes de modelo e detalhes sobre UsuarioVip

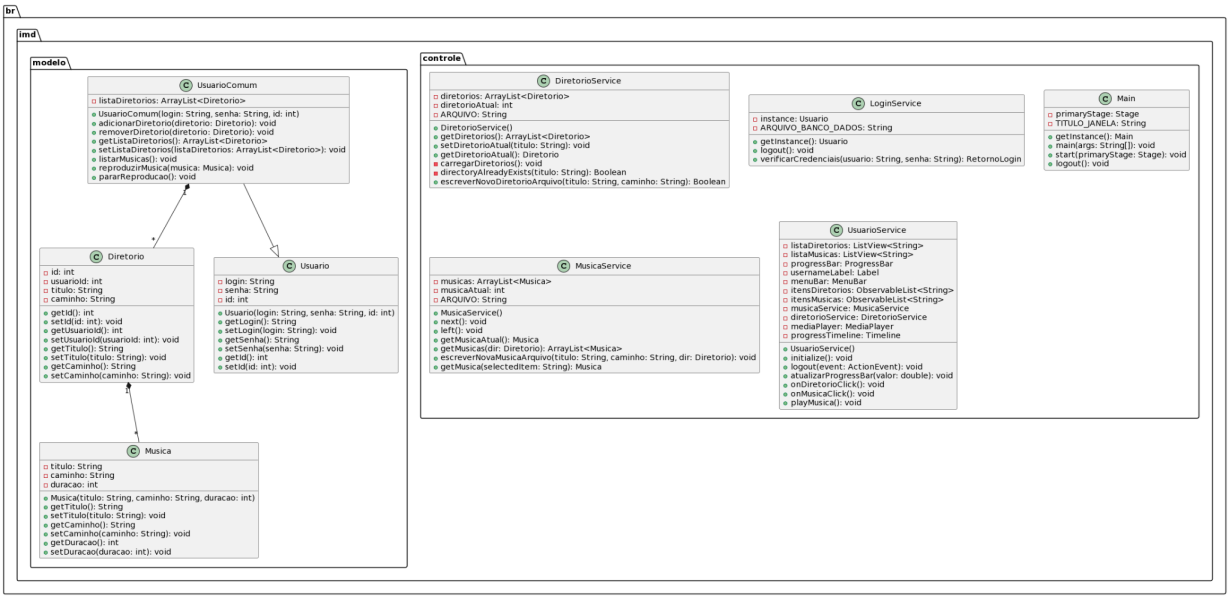

Figura 2. Gráfico UML completo - Possui as classes de modelo e controle

5. Conclusão

Em suma, utilizamos dos conceitos aprendidos em sala para o desenvolvimento do projeto, tais quais: Organização de pacotes; Interface (GUI); Utilização de bibliotecas externas; Documentação JavaDOC; Herança; Polimorfismo; Classe Abstrata; Classes

para tratamento de Exceções. Conseguimos concluir com êxito as seguintes partes solicitadas: Controle de Acesso, toda a parte de login está funcional, de acordo com o banco de dados onde, após o login, cada usuário tem sua tela específica de acordo com o nível de permissão (Comum ou Vip); Tocador MP3 (exceto a criação de playlist devido a um imprevisto) e também foi desenvolvido o extra, onde foi adicionado o Progress Bar para adiantar a barra da musica e ir para o ponto específico selecionado.

6. Referências

Seguimos as referências do Template SBC para realizar a escrita do relatório; Do exemplo de tela explícita no detalhamento do Projeto Final; Gráficos UML; E vídeos sobre a utilização do JAVAFX com Scene Builder.

SBC Templates para Artigos e Capítulos de Livros, http://www.sbc.org.br/documentos-da-sbc/summary/169-templates-para-artigos-e-capitulos-de-livros/878-modelosparapublicaodeartigos.

Especificação do Trabalho (2023) "Final ProjetoFinal_LP-II_TipoA_2023-1"

Diagrama de classes (2023) "O que é um diagrama de classe UML?" https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-classe-uml

Descompila (2017) "JavaFX para iniciantes - #02 - Scene Builder" https://www.youtube.com/watch?v=DpyVnys1nvs&ab_channel=Descompila