Linguagem de Programação 2 (IMD0040) - 2018-1

Projeto de Programação - terceira unidade

Este documento traz os detalhes necessários para a implementação do projeto da unidade. Quaisquer dúvidas adicionais **devem** ser tiradas com professor da disciplina. Portanto, não serão aceitas desculpas para a não entrega (total ou parcial), baseados em reclamações sobre o nível de detalhe da especificação.

Observe que este documento poderá ser atualizado no decorrer do semestre para trazer maiores detalhes sobre algum ponto ambíguo ou abstrato. Por gentileza, contribua com sugestões.

1- Introdução

Neste semestre você deverá implementar um tocador de música (*media player*), com funcionalidades para gerenciar um conjunto de músicas. O objetivo é desenvolver um sistema de médio porte, utilizando o paradigma da orientação a objetos na disciplina e conhecimentos adquiridos em EDB2.

Em relação aos conhecimentos específicos da disciplina de Linguagem de Programação 2 (LP2), é esperado que o aluno utilize os seguintes conceitos:

1. Organização de pacotes;
2. Interface (GUI) Swing;
3. Utilização de bibliotecas externas;
4. Documentação JavaDoc;
5. Herança;
6. Polimorfismo;
7. Classe Abstrata;
8. Interfaces;
9. Classes para tratamento de Exceções.
10. Utilizar pelos menos 2 padrões de projetos;
11. Testes unitários

2- Projeto Media Player

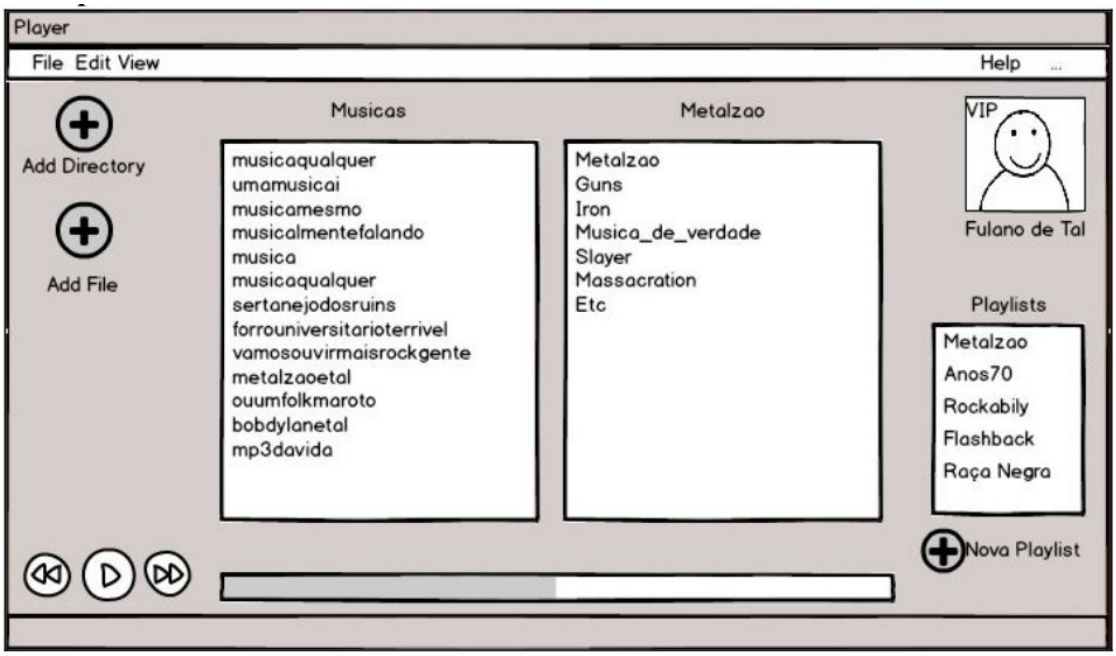
O projeto consiste em um tocador de MP3 (*media player*), capaz de produzir arquivos de áudio individualmente, ou como parte de uma *playlist*. Especificamente, o sistema deverá ter as seguintes funcionalidades:

1. Reproduzir uma música (já cadastrada), ou uma *playlist* de músicas;
2. Adicionar/remover um arquivo de música, ou todos os arquivos de um diretório;
3. Criar uma *playlist* com músicas previamente adicionadas ao media player;
4. Buscar músicas cadastradas;
5. Autenticar um usuário e diferenciar ele em diferentes perfis de acesso;
6. Cadastro de novos usuários;

Observações gerais:

- Qualquer biblioteca poderá ser utilizada para reproduzir as músicas.
    - A biblioteca sugerida é a JLayer: http://www.javazoom.net/javalayer/javalayer.html
- O tocador de áudio deve possuir uma interface gráfica, utilizando Swing ou JavaFX.

Exemplo de tela:



A seguir uma descrição mais detalhada sobre algunas funcionalidades e sobre as estruturas de dados que deverão ser utilizadas no projeto.

Controle de Acesso:

Ao realizar o logon na aplicação, os usuários deverão ser distinguidos entre usuários comuns e usuários VIP, onde:

- Usuários comuns podem:
    - adicionar arquivos de música individualmente, ou um diretórios com arquivos de música ao banco (conjunto) de arquivos do *media player*;
        - após a adição de um (ou vários) arquivos de música, as mesmas estarão visíveis a todos os usuários;
        - músicas adicionadas estarão ainda disponíveis no sistema em próximas execuções do mesmo (mesmo que o usuário feche o sistema e efetue um novo login);
    - tocar uma música previamente cadastrada no media player;
        - músicas podem ser selecionadas a partir de uma lista, ou filtradas pelo nome;
- Adicionalmente, usuários "VIP" podem:
    - criar playlists, que posteriormente serão visualizada apenas por ele;
    - cadastrar usuários para acesso ao player;
- O player deve conter um usuário "VIP" default (admin) para iniciar aplicação.

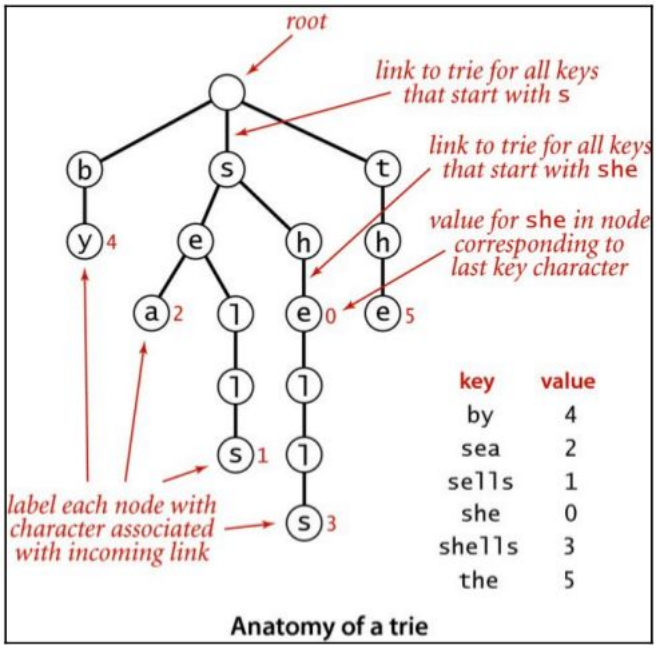
Salvando os dados da aplicação:

Para armazenar os dados da aplicação, utilizaremos arquivos de texto que podem ter a extensão ***.txt** ou qualquer uma que o desenvolvedor preferir. Deve-se manipular estes arquivos de modo a salvar os dados da maneira correta. Um arquivo (*musicas.txt*) deverá guardar informações sobre as músicas propriamente ditas, um para armazenar a lista de usuarios (*usuarios.txt*) e **n** arquivos para playlists (*playlist_xxx.txt*). Estes arquivos devem ser lidos quando o programa for aberto e devem ser alterados caso os usuários realizem alterações.

- musicas.txt: este arquivo salvará apenas o nome das músicas e seus respectivos caminhos. É importante que cada linha do arquivo representa o caminho completo para uma respectiva música.
- playlist_xxx.txt: haverá mais de um arquivo como este, um para cada playlist. Como as playlists serão privadas por usuário (cada usuário VIP pode criar e editar suas playlists), é importante que os arquivos contenham uma identificação do usuário (nome e id).
- usuarios.txt: este arquivo será responsável pelos dados do usuário. Durante a inicialização, o software carregará os usuários para a memória e poderá fazer a autenticação.

Dos conceitos vistos em EDB2, deve-se utilizar:

- Árvore patrícia (árvores de sufixo):
    - A aplicação deve ser capaz de autocompletar o nome dos arquivos de áudio cadastrados;
        - Árvores de sufixo armazenam, em cada um de seus nós, uma letra e uma flag para definir fim de palavra;
        - Nós folhas obrigatoriamente possuem a flag positiva.
- Árvore binária de busca (BST):
    - A BST será usada para armazenar usuários e será organizada de acordo com o ID dos usuários.
- As implementações das árvores devem conter: inserção, deleção e busca



Anatomy of a trie

A árvore de sufixos deve guardar, em cada nó, um caractere. Os filhos desse nó serão outros caracteres que seguem para a formação da palavra. Veja no exemplo à esquerda. Se fossemos adicionar a palavra "*they*", teríamos de adicionar um filho ("y") ao último nó ("e") e um valor agregado que demarca o fim de palavra. No nosso caso, basta que marquemos com um booleano, que diz se aquele nó é uma palavra ou não. Se fomos buscar pela música "*shells*", ao adicionar "s" no form de busca, é feita uma busca na árvore de sufixo que mostra todas as palavras formadas a partir do primeiro nó "s". Dessa forma, a cada caractere digitado na interface, deve-se rodar a busca na árvore. Para realizar tal tarefa, consulte a documentação das bibliotecas de interface que você vai usar (JavaSwing) e veja como são tratados os eventos de teclado (key events).

O menu de músicas:

A árvore de sufixos será utilizada para facilitar as busca realizadas pelos usuários dentro da aplicação. Digamos que existam muitas strings (que representam músicas) na lista de músicas disponíveis para tocar. Ao digitar na barra de busca a letra 'A', todas as músicas que não comecem com esta letra devem sumir da guia. Caso seja acrescentado um 'b' ("Ab"), todas as músicas que não comecem com "Ab" não deverão mais ser visualizadas.

Fica a critério do aluno o comportamento para quando a música encontrada for executada. A barra de busca pode ser limpa, e ser exibidas novamente todas as músicas disponíveis. Ou aguardar pelo usuário para que este remova o nome da busca.

A árvore deve ser gerada durante a inicialização do programa. Como a lista de músicas **não deve exibir** o caminho das mesmas, e sim apenas seu nome, deve-se tratar a string antes de inseri-la na árvore. Recomenda-se remover, também, a extensão do arquivo.

Além do campo busca rápida por nome da música, o player deve conter a lista de músicas do diretório que foi adicionado, podendo o usuário apenas selecionar e tocar.

O menu de playlists:

Os usuários VIPs poderão cadastrar músicas em uma playlist. Essa playlist deverá ter um nome, podendo tal usuário ter mais de uma. A forma como a playlist será montada deverá ser decidida pelo programador. Pode ser arrastando de uma lista para outra, abrindo diretórios e selecionando arquivo por arquivo, etc. Essa adição pode ser feita de qualquer maneira.

Extras:

O aluno pode melhorar seu projeto, incluindo alguns itens extras, tais como:

- O aluno pode pesquisar uma maneira de mostrar a barra de execução da música. A biblioteca possui um método para pegar o tempo atual da música. Essa barra deve permitir ao usuário avançar e voltar;
- O usuário pode utilizar, ainda, outras bibliotecas de música para ler outros tipos de arquivos.
    - uma segunda biblioteca capaz de executar arquivos Lossless audio (FLAC) é a: http://jflac.sourceforge.net/news.html.
- No momento que o áudio estiver sendo executado, o *player* não precisa executar outras tarefas. Porém, pode-se implementar outras funcionalidades através do uso de *multithread*.
- O sistema poderá registrar os diretórios adicionados (em um arquivo diretorios.txt), para que esta lista de diretórios seja verificada sempre que o software for iniciado.
    - Assim, quando novos arquivos de música forem adicionados ou removidos de um diretório, esta mudança será percebida pelo software (durante a sua inicialização), e, consequentemente, o conjunto de arquivos de música (musicas.txt) será alterado;
    - O formato pode ser definido pelo usuário, mas recomendamos que cada linha represente uma pasta.
    - Quando o player for executado pela primeira vez, não existirão pastas cadastradas, consequentemente, não haverá lista de músicas no primeiro acesso.
- Escolha livre: o aluno poderá propor um recurso extra.