Projeto Final 01 – LP2

Media Player com Java

O projeto consiste em um tocador de MP3 utilizando a biblioteca JLayer (http://www.javazoom.net/javalayer/javalayer.html), capaz de produzir arquivos de áudio. Outras bibliotecas de áudio podem ser utilizadas. É necessário que os alunos utilizem conceitos dados na disciplina de LP2, como:

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.

O tocador de áudio deve possuir uma interface gráfica. O sistema deve ser capaz de prever um *login* para seus usuários. Diferenciando níveis entre eles.

Controle de Acesso:

- Usuários comuns podem tocar músicas normalmente, selecionando-as de uma lista;
- Usuários "VIP" podem criar *playlists* e cadastrar usuários para acesso ao Player;
- O Player deve conter um usuário default (*admin*) para iniciar aplicação.
    - Usuário comum:
        - O usuário comum pode adicionar diretórios para que o player exiba as músicas.
    - Usuário VIP:
        - O usuário vip terá a possibilidade de ter, atrelado **à sua conta**, uma ou mais *playlists* personalizadas.
    - Importante: após o usuário adicionar diretórios de músicas ao player, a lista de músicas aparecerá no player, mesmo que o usuário feche o player, após um novo *login* as músicas devem aparecer listadas.

Salvando os dados da aplicação:

Para manter os dados da aplicação salvos, 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. Os primeiros 2 arquivos deverão guardar informações sobre as pastas e sobre as músicas propriamente ditas, 1 para usuários e ***n*** arquivos para *playlists*. Estes arquivos devem ser lidos quando o programa for aberto e devem ser alterados caso os usuários realizem alterações.

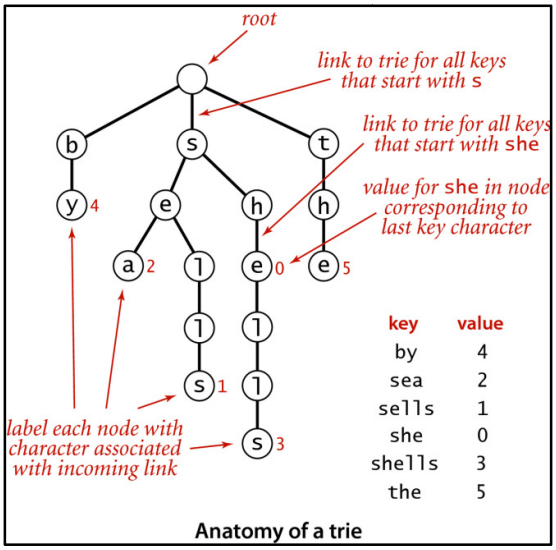
- ***diretorios.txt***: este arquivo será responsável por informar ao *player* quais as pastas serão lidas quando for o software for iniciado. O formato pode ser definido pelo usuário, mas recomendamos que cada linha represente uma pasta. Este arquivo deve ser lido a cada vez que o *player* for aberto e deve ser alterado

quando o usuário desejar adicionar mais pastas. Quando o *player* for executado pela primeira vez, não existirão pastas cadastradas, consequentemente, não haverá lista de músicas no primeiro acesso.

- ***musicas.txt***: este arquivo salvará apenas o nome das músicas e seus respectivos caminhos. É importante que cada linha do arquivo represente 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 comecem com a autenticação do usuário (nome e id).
- ***usuarios***: 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 armazenados nos arquivos;
    - 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



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.

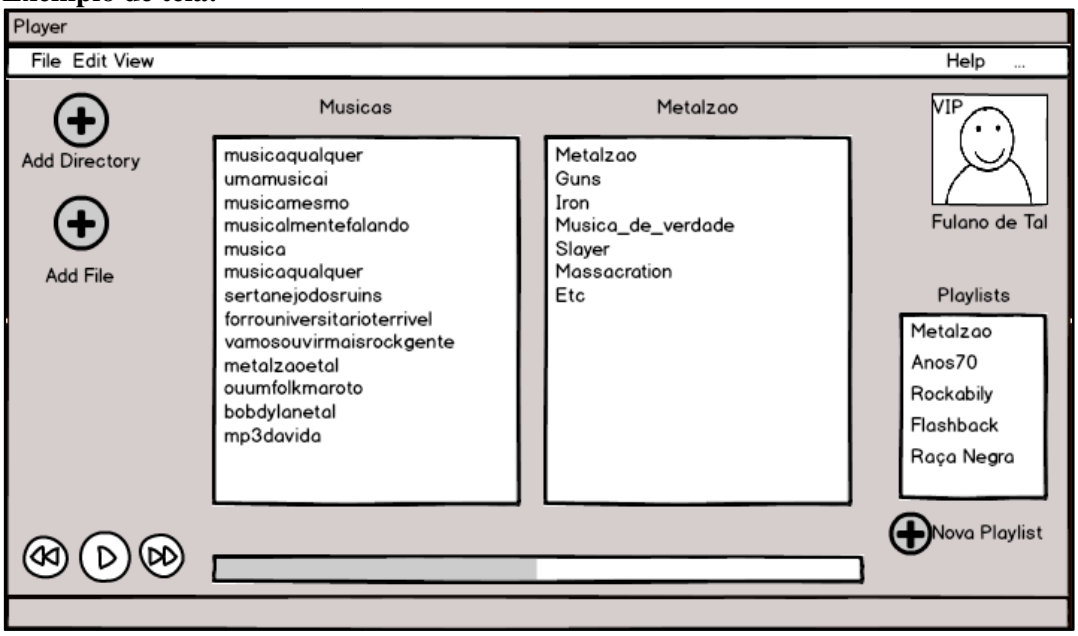
A árvore deve ser gerada durante a inicialização do programa. Como a lista de músicas **não deve exibir** o caminho das mesmas, e sim apenas seu nome, deve-se tratar a *string* antes de inseri-la na árvore. Recomenda-se remover, também, a extensão do arquivo.

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 pra outra, abrindo diretórios e selecionando arquivo por arquivo, etc. Essa adição pode ser feita de qualquer maneira.

Exemplo de tela:



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, uma segunda biblioteca capaz de executar arquivos Lossless audio (FLAC) http://jflac.sourceforge.net/news.html.
- Escolha livre: o aluno poderá propor um recurso extra, que utilize assuntos vistos na disciplina LP2.
- 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*.