TRABALHO PRÁTICO - FASE 3

Astélio Weber

Leonardo Holtz

Lorenzo Cernicchiaro

Descrição do Programa

O principal objetivo do programa é gerar uma música a partir de um texto que o usuário dará de entrada. O programa se preocupará em ser acessível e objetivo ao usuário, permitindo que ele usufrua da sua funcionalidade, ensinando ele a como usar o programa. O texto a ser lido será a partir de um input direto do usuário e também de um arquivo de texto. A música gerada poderá ser tocada pelo próprio programa ou armazenada em um arquivo de áudio MIDI, com diretório podendo ser escolhido pelo usuário.

A interface do usuário é feita em Java e possui métodos para o usuário poder interagir com o mouse e com o teclado, clicando em botões e digitando o texto de entrada. A caixa de texto serve para o usuário digite o texto que gerará a música. O botão “Search File” permite que o usuário selecione um arquivo de texto em um diretório de escolha. O conteúdo do arquivo imediatamente aparece na caixa de texto para o usuário poder olhar. A caixa de seleção “Select BPM” serve para o usuário escolher o tempo da batida da música. O botão “Play” tocará a música de saída para o usuário. O botão “Generate File” permitirá que o usuário salve um arquivo de áudio MIDI com o nome de sua escolha para salvar a música que foi gerada. Por fim, o botão “Help” mostrará os comandos disponíveis para o usuário.

Classes

Breve descrição das classes implementadas:

- **Interface:** Classe que Interage com o usuário através de um layout intuitivo com o auxilio da API Java Swing.
- **Texto:** Converte o texto inserido pelo usuário em uma “String Musical” (input utilizado pela API JFugue) seguindo as especificações propostas pelo trabalho.
- **ConversaoInputParaStringMusicalTeste:** Classe de Teste em JUnit 4 criada para verificar se o parser de formatação e criação da string musical está funcionando corretamente. Em todos os testes realizados não houve erros.
- **Musica:** Realiza a manipulação da API JFugue para criar um arquivo de som baseado no método de input e retorná-lo ao usuário.

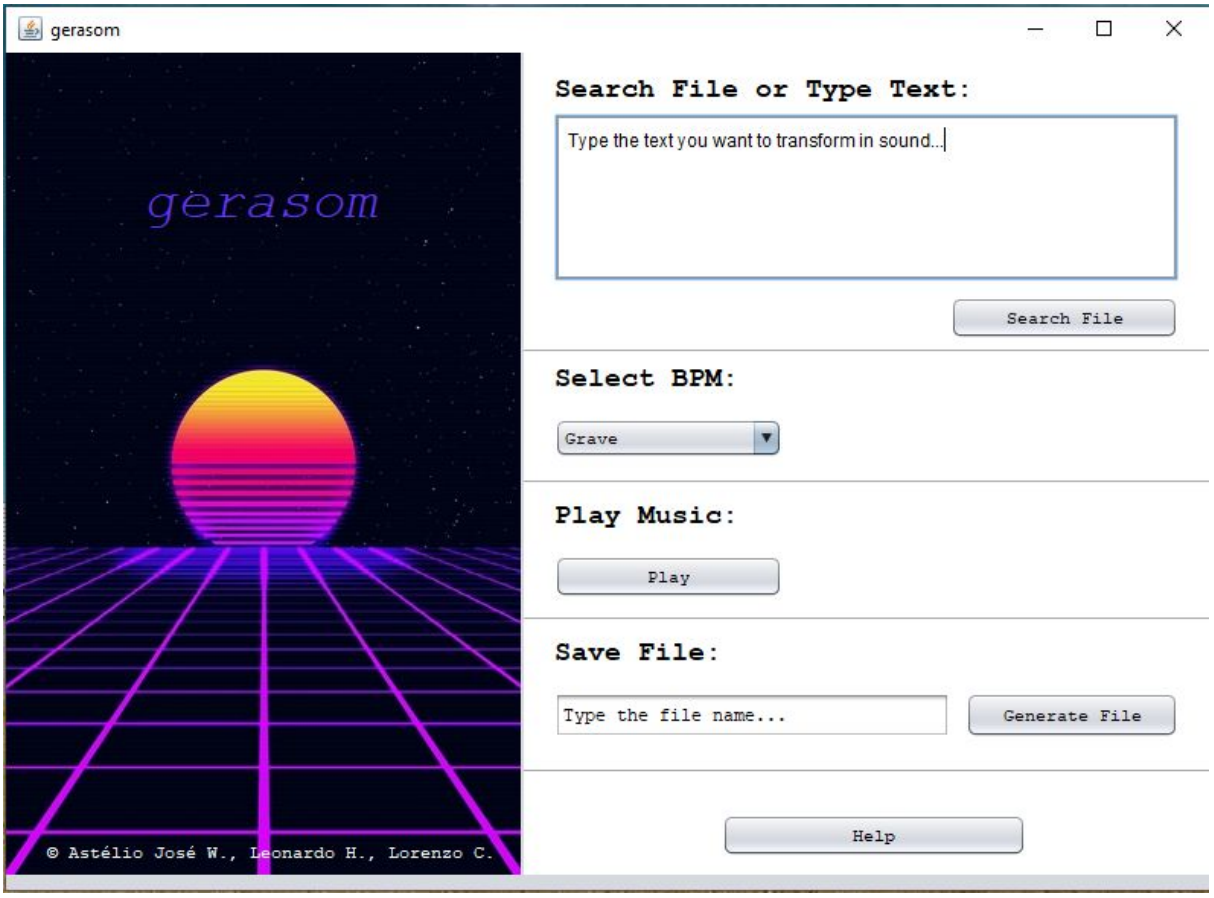
Atributos das classes implementadas:

- **Texto:**
    - private String oitava;
    - private int oitavaInt;
    - private int instrumento;
    - private String instrumentoString;
    - private int volume;
    - private String volumeString;
    - private String OITAVA_DEFAULT = "5";
    - private int INSTRUMENTO_DEFAULT = 0;
    - private int VOLUME_DEFAULT = 127;
    - public String stringMusical;

- **Musica:**
    - public String entradaUsuario;
    - public String tempoBPM;
    - public String nomeArquivo;
    - public Player player = new Player();

Métodos das classes implementadas:

- **Texto:**
    - public boolean charValido(char caractere)
    - private boolean ehNota(char caractere)
    - public String formataTexto(String inputTexto)
    - private boolean ehOutraVogal(char c)
    - public String textoParaMusicString(String textoFormatado, String tempoBPM)
- **Musica:**
    - public void setEntradaUsuario(String entradaUsuario)
    - public void setTempoBPM(String tempoBPM)
    - public void setNomeArquivo(String nomeArquivo)
    - public void setPlayer(Player player)
    - public String getEntradaUsuario()
    - public String getTempoBPM()
    - public String getNomeArquivo()
    - public Player getPlayer()
    - public Pattern textToPattern(String entradaUsuario, String tempoBPM)
    - public void playMusic(String entradaUsuario, String tempoBPM)
    - public void createMidiFile(String entradaUsuario, String tempoBPM, String nomeArquivo)


gerasom
gerasom
© Astélio José W., Leonardo H., Lorenzo C.
Search File or Type Text:
Type the text you want to transform in sound...
Search File
Select BPM:
Grave
Play Music:
Play
Save File:
Type the file name...
Generate File
Help