Trabalho 3 de Computação Sônica

Amanda Lopes Dantas	13/0100391
Bruno José Bergamaschi Kumer Reis	14/0017666
Ingrid Santana Lopes	14/0083065
Marcos Paulo Cayres Rosa	14/0027131
Rennê Ruan Alves Oliveira	14/0030930

Conversor

```
public class Conversor {
        static private File arquivoMidi = null;
        static private Sequence sequenciaMidi = null;

        private Track[] trilhasDetectadas;
        private double durtique;

        public Conversor(File arquivo){
                try{
                        arquivoMidi = arquivo;
                        sequenciaMidi = MidiSystem.getSequence(arquivoMidi);

                        LerArquivo();
                        CriaArquivoJava();

                }
            catch(InvalidMidiDataException e2){
                System.out.println(e2 + " : Erro nos dados midi.");
            }
            catch(IOException e3){
                System.out.println(e3 + " : O arquivo midi nao foi encontrado.");
            }
        }
```

Cria Arquivo .java

- Inicia o Arquivo:

```
try (PrintWriter in = new PrintWriter(aux + ".java", "UTF-8")){
    in.println("import sintese.*;");
    in.println("\npublic class " + aux + " {");
    in.println("\n\tpublic static void main(String[] args) {");
```

- Para cada trilha decodifica os eventos:

```
for (int j = 0; j < trilhasDetectadas[i].size(); j++){
        double duracao = -1, decadencia = 0;
        MidiEvent evento = trilhasDetectadas[i].get(j);
        MidiMessage msg = evento.getMessage();
        status = msg.getStatus();
        dados = msg.getMessage();
```

Cria Arquivo .java

```
if (inst >= 0 && inst <= 8) {
    tipoInst = "new Instrumento1()";
} else if (inst >= 9 && inst <= 16) {
    tipoInst = "new Instrumento2()";
} else if (inst >= 17 && inst <= 24) {
    tipoInst = "BancoDeInstrumentos.timbreortogonal3()";
} else if (inst >= 25 && inst <= 40) {
    tipoInst = "new Instrumento1()";
} else if (inst >= 41 && inst <= 56) {
    tipoInst = "new Instrumento3()";
} else if (inst >= 57 && inst <= 72) {
    tipoInst = "new Instrumento1()";
} else if (inst >= 73 && inst <= 80) {
    tipoInst = "new Instrumento2()";
} else if (inst >= 81 && inst <= 112) {
    tipoInst = "BancoDeInstrumentos.timbreortogonal3()";
} else if (inst >= 113 && inst <= 120) {
    tipoInst = "BancoDeInstrumentos.timbreortogonal3()";
} else if (inst >= 121 && inst <= 128) {
    tipoInst = "BancoDeInstrumentos.timbre_quasetonal()";
} else {
    throw new RuntimeException("Numero do instrumento invalido");
}

numInst++;
in.print("\n\n\t\tMelodia melodia" + numInst + " = new Melodia();\n" +
            "\t\tmelodia" + numInst + ".addNota(new Nota(0.0001, 0.0001, 0.0001));\n");
```

Instrumentos

Instrumento 1

```
frequencia = 300;

curva1 = constroiCurva(30, 200, 240, 100, 720, 0);
curva2 = constroiCurva(30, 300, 200, 200, 720, 0);
curva3 = constroiCurva(50, 300, 270, 100, 720, 0);
curvaOsc = constroiCurva(10,1300,300, 1000,500,900);
curvaOsc.addPonto(600, 1000);
curvaOsc.addPonto(720, 1100);

envoltoria1 = new Envoltoria(curva1);
envoltoria2 = new Envoltoria(curva2);
envoltoria3 = new Envoltoria(curva3);
envoltoriaOsc = new Envoltoria(curvaOsc);

unidadeH1.setEnvoltoria(envoltoria1);
unidadeH2.setEnvoltoria(envoltoria2);
unidadeH3.setEnvoltoria(envoltoria3);
```

Instrumento 2

```
frequencia = 300;

curva1 = constroiCurva(10, 600, 220, 230, 720, 0);
curva2 = constroiCurva(10, 230, 210, 250, 720, 10);
curva3 = constroiCurva(30, 150, 200, 180, 720, 10);
curvaOsc = constroiCurva(10,1300,300, 1000,500,900);

envoltoria1 = new Envoltoria(curva1);
envoltoria2 = new Envoltoria(curva2);
envoltoria3 = new Envoltoria(curva3);
envoltoriaOsc = new Envoltoria(curvaOsc);

unidadeH1.setEnvoltoria(envoltoria1);
unidadeH2.setEnvoltoria(envoltoria2);
unidadeH3.setEnvoltoria(envoltoria3);
```

Instrumentos

Instrumento 3

```
frequencia = 300;

curva1 = constroiCurva(30, 130, 140, 100, 220, 0);
curva2 = constroiCurva(30, 250, 100, 200, 420, 0);
curva3 = constroiCurva(50, 300, 170, 100, 420, 0);
curvaOsc = constroiCurva(10,1300,300, 1000,500,900);
curvaOsc.addPonto(600, 1000);
curvaOsc.addPonto(720, 1100);

envoltoria1 = new Envoltoria(curva1);
envoltoria2 = new Envoltoria(curva2);
envoltoria3 = new Envoltoria(curva3);
envoltoriaOsc = new Envoltoria(curvaOsc);

unidadeH1.setEnvoltoria(envoltoria1);
unidadeH2.setEnvoltoria(envoltoria2);
unidadeH3.setEnvoltoria(envoltoria3);
```

Resultado

```
import sintese.*;

public class HarryPotter {

    public static void main(String[] args) {


        Melodia melodia1 = new Melodia();
        melodia1.addNota(new Nota(0.5200495049504951, 493.8833012561241, 100));
        melodia1.addNota(new Nota(0.697029702970297, 659.2551138257398, 100));
        melodia1.addNota(new Nota(0.3784653465346534, 783.9908719634985, 100));
        melodia1.addNota(new Nota(0.539108910891089, 739.9888454232688, 100));
        melodia1.addNota(new Nota(0.8603960396039604, 659.2551138257398, 100));
        melodia1.addNota(new Nota(0.5527227722772277, 987.7666025122483, 100));
                                    ○
                                    ○
                                    ○
        melodia1.addNota(new Nota(0.5499999999999999, 932.3275230361799, 100));
        melodia1.addNota(new Nota(0.8658415841584158, 739.9888454232688, 100));
        melodia1.addNota(new Nota(0.5418316831683168, 783.9908719634985, 100));
        melodia1.addNota(new Nota(1.8405940594059405, 659.2551138257398, 100));

        Polifonia p = new Polifonia();
        Voz voz1 = new Voz(new Instrumento2());
        voz1.addMelodia(melodia1);
        p.addVoz(voz1);

        Som s = p.getSom();
        s.visualiza();

    }

}
```

INTERFACE


Player
Arquivo:
Diretorio:
ABRIR MIDI
INFORMACOES
EVENTOS...
Converter .Ja...
75%
00h 00m 00s