import java.io.File;

import java.io.IOException;

import java.util.Scanner;

import javax.sound.sampled.AudioInputStream;

import javax.sound.sampled.AudioSystem;

import javax.sound.sampled.Clip;

import javax.sound.sampled.LineUnavailableException;

import javax.sound.sampled.UnsupportedAudioFileException;

public class Main

{

// to store current position

Long currentFrame;

Clip clip;

// current status of clip

String status;

AudioInputStream audioInputStream;

static String filePath;

// constructor to initialize streams and clip

public Main()

throws UnsupportedAudioFileException,

IOException, LineUnavailableException

{

// create AudioInputStream object

audioInputStream =

AudioSystem.getAudioInputStream(new File("https://drive.google.com/file/d/1OIRO\_wK5lQ2xQf72oqnyHI9pzHH2HvJu/view"));

// create clip reference

clip = AudioSystem.getClip();

// open audioInputStream to the clip

clip.open(audioInputStream);

clip.loop(Clip.LOOP\_CONTINUOUSLY);

}

public static void main(String[] args)

{

try

{

filePath = "https://drive.google.com/file/d/1OIRO\_wK5lQ2xQf72oqnyHI9pzHH2HvJu/view";

Main audioPlayer =

new Main();

audioPlayer.play();

Scanner sc = new Scanner(System.in);

while (true)

{

System.out.println("1. pause");

System.out.println("2. resume");

System.out.println("3. restart");

System.out.println("4. stop");

System.out.println("5. Jump to specific time");

int c = sc.nextInt();

audioPlayer.gotoChoice(c);

if (c == 4)

break;

}

sc.close();

}

catch (Exception ex)

{

System.out.println("Error with playing sound.");

ex.printStackTrace();

}

}

// Work as the user enters his choice

private void gotoChoice(int c)

throws IOException, LineUnavailableException, UnsupportedAudioFileException

{

switch (c)

{

case 1:

pause();

break;

case 2:

resumeAudio();

break;

case 3:

restart();

break;

case 4:

stop();

break;

case 5:

System.out.println("Enter time (" + 0 +

", " + clip.getMicrosecondLength() + ")");

Scanner sc = new Scanner(System.in);

long c1 = sc.nextLong();

jump(c1);

break;

}

}

// Method to play the audio

public void play()

{

//start the clip

clip.start();

status = "play";

}

// Method to pause the audio

public void pause()

{

if (status.equals("paused"))

{

System.out.println("audio is already paused");

return;

}

this.currentFrame =

this.clip.getMicrosecondPosition();

clip.stop();

status = "paused";

}

// Method to resume the audio

public void resumeAudio() throws UnsupportedAudioFileException,

IOException, LineUnavailableException

{

if (status.equals("play"))

{

System.out.println("Audio is already "+

"being played");

return;

}

clip.close();

resetAudioStream();

clip.setMicrosecondPosition(currentFrame);

this.play();

}

// Method to restart the audio

public void restart() throws IOException, LineUnavailableException,

UnsupportedAudioFileException

{

clip.stop();

clip.close();

resetAudioStream();

currentFrame = 0L;

clip.setMicrosecondPosition(0);

this.play();

}

// Method to stop the audio

public void stop() throws UnsupportedAudioFileException,

IOException, LineUnavailableException

{

currentFrame = 0L;

clip.stop();

clip.close();

}

// Method to jump over a specific part

public void jump(long c) throws UnsupportedAudioFileException, IOException,

LineUnavailableException

{

if (c > 0 && c < clip.getMicrosecondLength())

{

clip.stop();

clip.close();

resetAudioStream();

currentFrame = c;

clip.setMicrosecondPosition(c);

this.play();

}

}

// Method to reset audio stream

public void resetAudioStream() throws UnsupportedAudioFileException, IOException,

LineUnavailableException

{

audioInputStream = AudioSystem.getAudioInputStream(

new File(filePath).getAbsoluteFile());

clip.open(audioInputStream);

clip.loop(Clip.LOOP\_CONTINUOUSLY);

}

}