CS 112-01

Project 4: Mp3 Player.

Due: May 8th, 11:59pm

50 pts

For this project you will write an mp3 player with an interface similar to iTunes. The program should be able to load mp3 files from your local **mp3** directory (and its subdirectories), extract tags from the mp3 files (artist, title and album) using an external library called `jAudiotagger`, and then display the sorted list of songs on the panel. The user will be able to select a given song, and then play it by clicking the Play button.

User Interface

This is how the user interface of your program will look like when you start your program (before the files are loaded into the program):

Figure 1: Mp3 player GUI when you start the program.

Most of the GUI code is provided to you in class MPlayerPanel.java. Download it from the course website.

When the user clicks the "Load mp3" button, your program will find all the mp3 files in the local mp3 directory and its subdirectories, extract the title, the artist and the album tags, create an array of song objects and display them in the panel. This is how the user interface will look like after this step (it shows four songs):

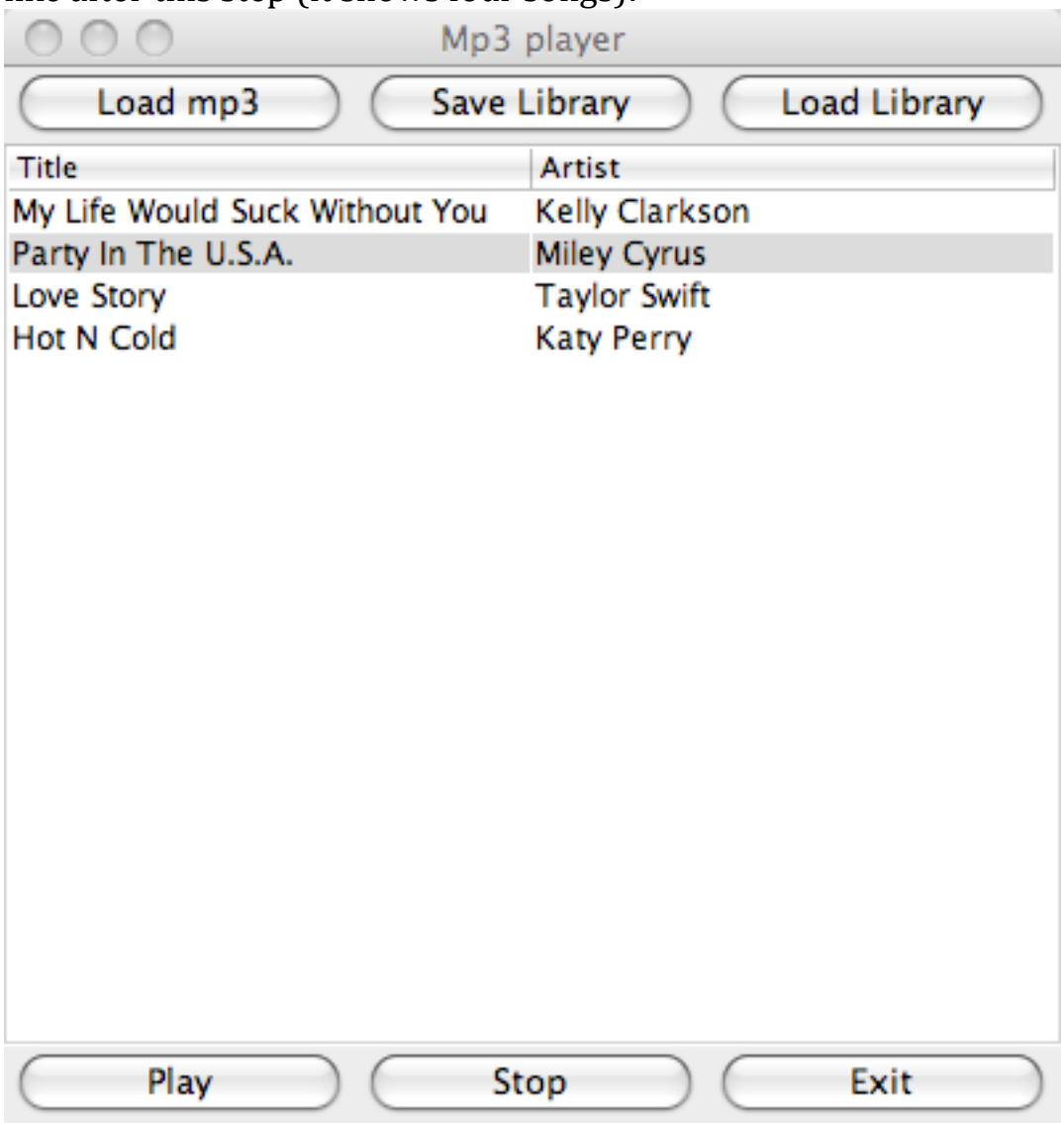


Figure 2: Mp3 player GUI after loading songs.

The user can now select a song and press the "play" button to play it. Pressing the "Stop" button should stop playing the song. "Exit" would exit the program.

The song library can be saved to a file (called "songs") by clicking the "Save Library" button. The format of the file is described later in this handout. Finally, the library of songs can be loaded from the "songs" file (so that we do not have to traverse mp3 files every time we run the program).

Implementation

- **Loading mp3 files:**

Store your mp3 files[1] in the project subdirectory called **mp3**. Create several subdirectories inside this directory and place several mp3 files in each subdirectory.

Your program should **recursively** traverse **mp3 directory (and its subdirectories)** to find all mp3 files. For each mp3 file, it should extract the artist, the title and the album tags using the external library `jAudiotagger` (see more about the library below). After extracting the artist, the title and the album tags from each mp3 file, store this information about each song in the "song library" (you can use ArrayList of "songs" internally) and display it on the panel using JTable (see Figure 2).

jAudiotagger

Download jAudiotagger here:
https://maven.java.net/content/repositories/snapshots/net/jthink/jaudiotagger/2.2.0-SNAPSHOT/
You need the file called:
jaudiotagger-2.2.0-20130321.142353-1.jar

You need to add this jar file to the build path of your project: follow directions outlined on this website:

---

[1] There are several free legal websites where you can download free mp3 songs.

http://tutoringcenter.cs.usfca.edu/resources/adding-user-libraries-in-eclipse.html

Here is an example of how to use jAudiotagger library to extract the artist tag from file1.mp3:

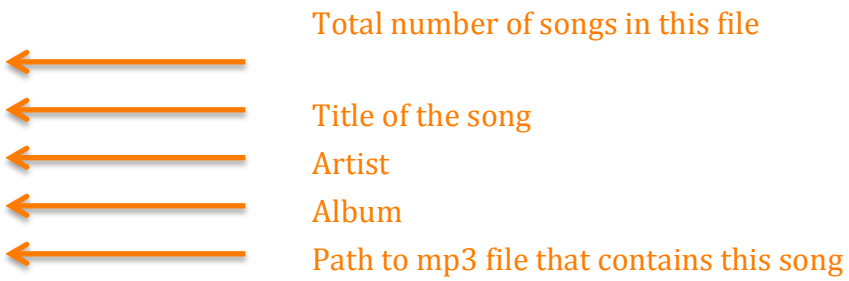
```
AudioFile f = AudioFileIO.read(new File("file1.mp3"));
Tag tag = f.getTag();
String artist = tag.getFirst(FieldKey.ARTIST);
```

- **Saving the library of songs:**

You should be able to save your "library of songs" into a file (by default it should save to the file called "songs" located in your project directory). The format of this file should be the following:

```
#songs
title
artist
album
mp3_file
```

The first line has the total number of songs in the song library, and then for each song the file has the title, the artist, the album and the name of the mp3 file that has the audio for the song. Here is a sample songs file with 2 songs:




```
Gangnam Style
Psy
Psy 6
mp3/psy2.mp3
```

For extra credit, save the library in the XML file (talk to the instructor prior to implementing this). (2pts)

- **Loading the library of songs:**

As opposed to traversing the mp3 directory for mp3 files and extracting the tags each time, we can just load the songs from the library of songs we previously saved in the "songs" file. This involves reading from this file, adding songs to the song library and then displaying the sorted list of songs.

- **Sorting songs**

Whenever you display the list of songs on the panel, the list should be sorted in ascending order by the title. **You need to implement the sorting algorithm yourself** (I recommend insertion sort) for sorting songs. Each song should know how to compare itself with another song – what interface does it need to implement?

- **Playing a song**

Use JLayer library for playing mp3 files from your Java program. Download the jar file from this website and add it to the build path of your project: http://www.javazoom.net/javalayer/javalayer.html

To play an mp3 file using JLayer library, you need to create a Player object (this class is in JLayer library) from the given mp3 file and invoke the play method to begin playback. The close method stops playback. The problem is that the play method will not return until the song is finished playing. In order for your program to respond to other events during playback (for instance, if the user pressed the stop button to stop playing the song), you will need to create a new *Thread* that invokes the play method. Create an inner class PlayerThread, a subclass of Thread, in whatever class will call the play method:

```
class PlayerThread extends Thread {
  Player pl;
  PlayerThread(String filename) {
     FileInputStream file;
     try {
          // filename here contains mp3 you want to play
          file = new FileInputStream(filename);
```

```
            pl = new Player(file);

        } catch (FileNotFoundException e) {
            e.getMessage();
        }
      catch (JavaLayerException e) {
            e.getMessage();
      }

 }
 public void run() {
     try {
         pl.play();
     }
     catch(Exception e) {
         e.getMessage();
     }
 }
}
```

Then, when the button Play is pressed, you can determine what song is selected and then call:

```
currThread = new PlayerThread(filename);
currThread.start();
```

Then the song will start playing.

Design

Before you start coding, spend the time to think about the good design for this project (it will part of the grade!): what classes will you use? How will they interact? Do **not** place all your code in MPlayerPanel class! I encourage students to come talk to me about their design before they start implementing it.

Extra credit

You can earn extra credit if you implement the following operations:

- Search songs by artist and by title. **5 pts**

Add a text box to the UI that allows the user to enter the name of an

artist, click the Search button and the program will display all songs by that artist. The same should be done for the title. Partial search (when the user enters a partial name) will earn additional credit.

- You can implement some extra GUI functionality by using JFileChooser to select a file to save the song library to, and to select a file to load the song library from. **2 pts**

- Using XML file to store the song library. **2 pts**

Submission:

Your code needs to be submitted to svn, to a subdirectory called project4 inside a cs112 directory. Go to https://www.cs.usfca.edu/svn/<username>/cs112/project4 and double check that your java files show up in the correct directory. If your code is not in svn by the deadline, you will not be able to get any credit for it.