**Task 2 – Java Multiple threaded Application**

**Title: Music Player**

The Music Player will show a dialog screen for user to select a .wav file. While choosing the file, an information will be shown on the console screen together will all the list of available .wav files from my directory. Once the user selected the .wav file it will continuously play until he/she type “STOP”. If “STOP” was entered, the program will ask the user if he/she wants to play another song again.

**Explanation as to how your program achieved a multithreading functionality. Also, identify the activities that can proceed simultaneously during execution.**

I chose this application because in this sample I can have multiple tasks that can be done

simultaneously using thread. Here are some of the tasks that can be done simultaneously, however, I only selected letter a.-c. to implement in my Java program.

1. *Show dialog box to select specific music you want to play.*
2. *Playing the current song and display its information.*
3. *Show list of tracks available in my Music Player.*
4. Displaying ads in Music Player.
5. Creating a playlist and keep tracking on any song that was added or deleted from files.

**List of all classes, methods, objects, interfaces, etc. that are used in your program.**

1. I created two programs for this: **MainProgram.java** and **MusicPlayer.java**.
2. For **MainProgram.java**, I have a static void main which will call and instantiate **MusicPlayer.**
3. For **MusicPlayer.java**, I have the following:

* MusicPlayer is the main class which extends the use of Thread
* **public** **void** playMusic(), this is the function that executes all thread.
* **thSelectMusic** – thread to show a dialog box to select the specific music you want to play
* **thInfo** – thread to display information on the current song that is playing.
* **thDisplayWavFiles** – thread to show list of tracks available in my Music Player.
* **private void** displayWavFiles(), **thDisplayWavFiles** will use this function, and this is where the actual code to show the list of tracks happened.
* **private** **void** info(), **thInfo** will use this function.
* **private** **void** selectMusic(), **thSelectMusic** will use this function.

1. In **MusicPlayer.java** again,I used Files, Paths, Scanner, JDialog, JFileChooser and Clip.

Files and Paths to get all the .wav files in the directory which I specified

JDialog and JFileChooser – is to show the dialog box

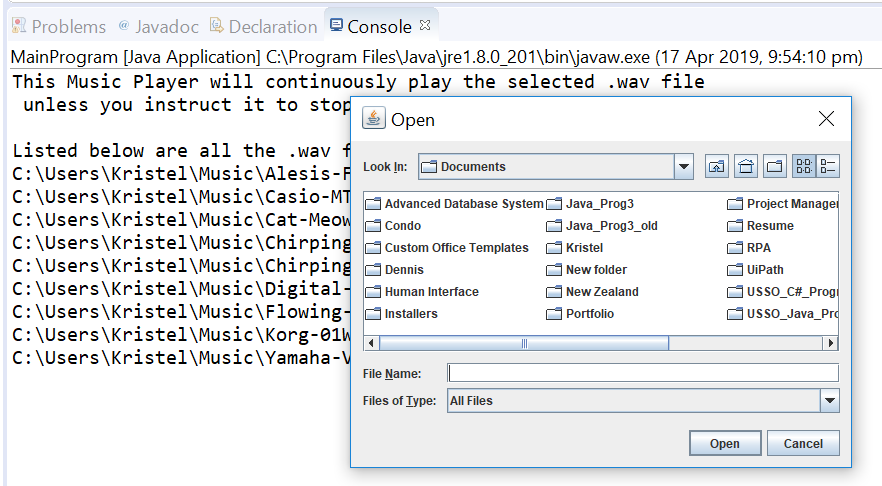
Clip – is to control the audio

**Discussion as to why you implemented Synchronization to your objects.**

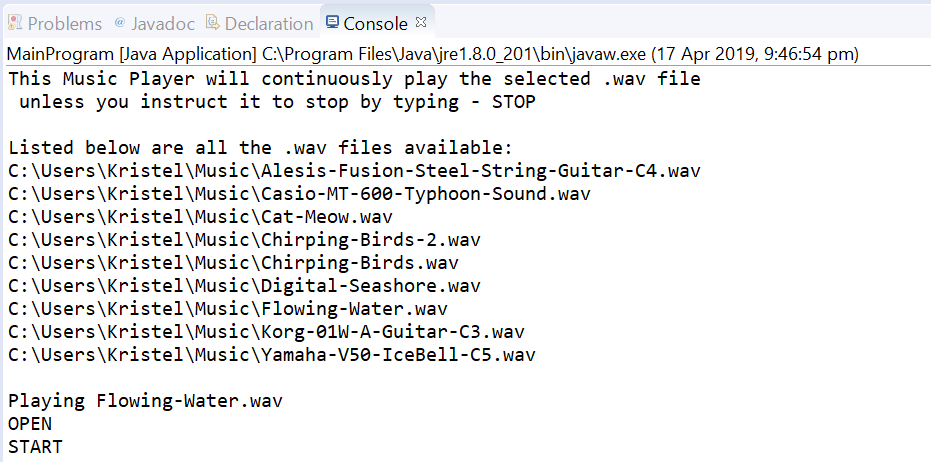
I implemented Synchronization because I want the thread output to be in sequence. I want to display first this information “This Music Player will continuously play the selected .wav file unless you instruct it to stop by typing - STOP” before the list of all the .wav files. I want to achieve the same result as the below screen shots:

**Music Player – Output Screen shots:**

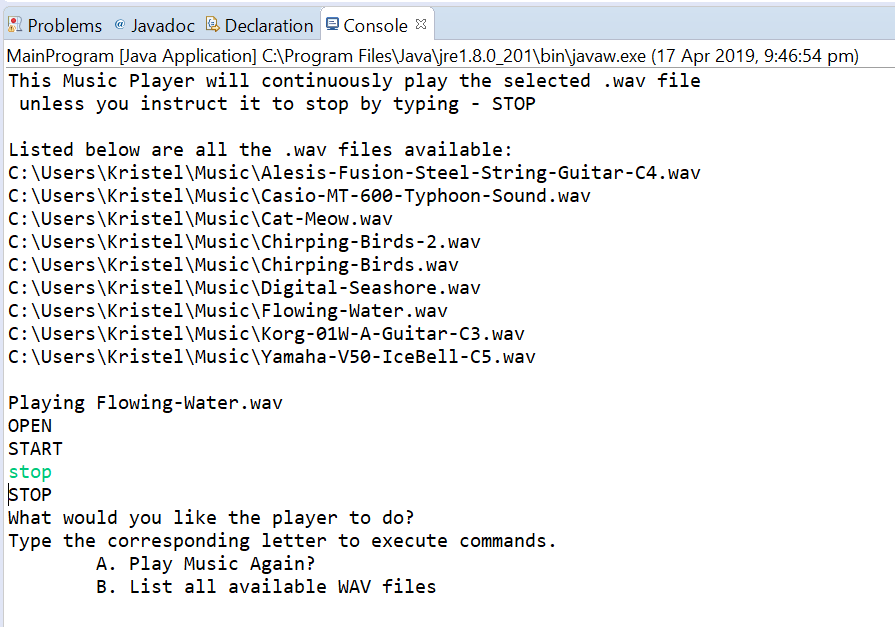
1. When the MainProgram.java was triggered the below screen shot will be shown:



1. After selecting which song to play, the below screen will be shown with the music on the background.



1. When “STOP” was entered the below screen will be shown:



1. When entered “A. Play Music Again?” the below dialog box will be shown:

