Technical report

Digital Innovations driving media entertainment

1. Software Installation

   1.1 Java Installation

Step 1: Open your internet browser

Step 2: Search for the latest version of Java JDK, JRE and download it

http://www.oracle.com/technetwork/java/javase/downloads/index.html.

Step 3: Install the package of java (JDK and JRE)

JDK is installed in this path "C:\Program Files\Java\jdk-10.0.{x}", where {x} is the java version
JRE is installed in this path "C:\Program Files\Java\jre-10.0.{x}".

Step 4: Set the Environment variables

- Go to control panel → System and Security→System→Advanced System settings→Environment variables→New variables→In "Variable Name", enter "JAVA_HOME" ⇒ In "Variable Value", enter your JDK installed directory path →ok→ok
- Here path of the JDK means
  "c:\Program Files\Java\jdk-10.0.xx", where xx is the version number

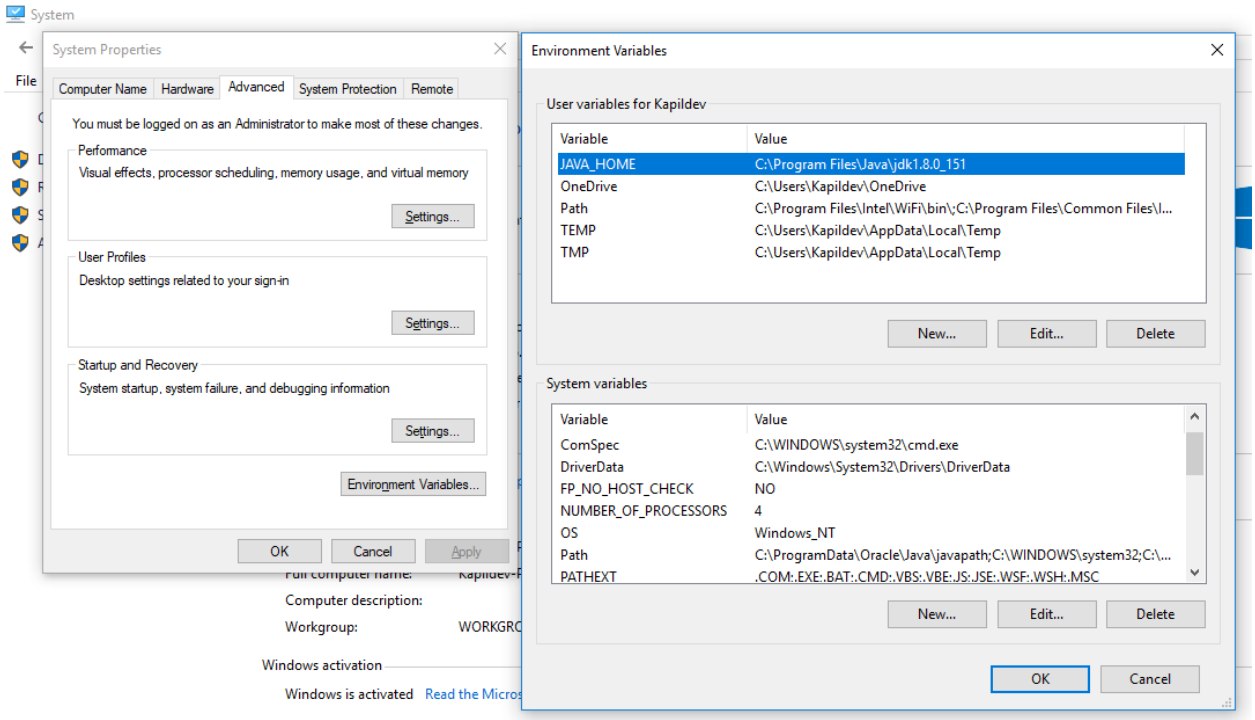

Fig 1 : To set environmental variables

Step 5: To verify environment variables and java is installed correctly

- Restart the system and open command prompt and check java is correctly installed or not by typing java and press enter.

1.2 Eclipse Installation

Step 1: Go to Internet browser and type

http://www.eclipse.org/downloads/packages/release/luna/r/eclipse-ide-java-developers

Step 2: Download eclipse according to the operating system type (Windows 32 bit or Windows 64 bit)

Step 3: Start the Eclipse installer executable file

Step 4: Select the package to install “Eclipse IDE for Java Developers”

Step 5: Select your installation folder and click on next-next

Step 6: Launch eclipse

2. Data Files and its Directory

Step 1: Create a folder in which music files are stored

Example-E:\MUSIC CITY\prgmMusic

This path will be used in the programming code to find the location of music files

Step 2: Music files should be in .wav extension because Java doesn't supports .mp3 file

Step 3: Convert the song in .wav file online by clicking following site:-

https://audio.online-convert.com/convert-to-wav

- Choose the .mp3 files that needs to be converted in .wav files and upload it online
- Click on start conversion and once it is done, download the files and store in the above folder you have already created.

- \

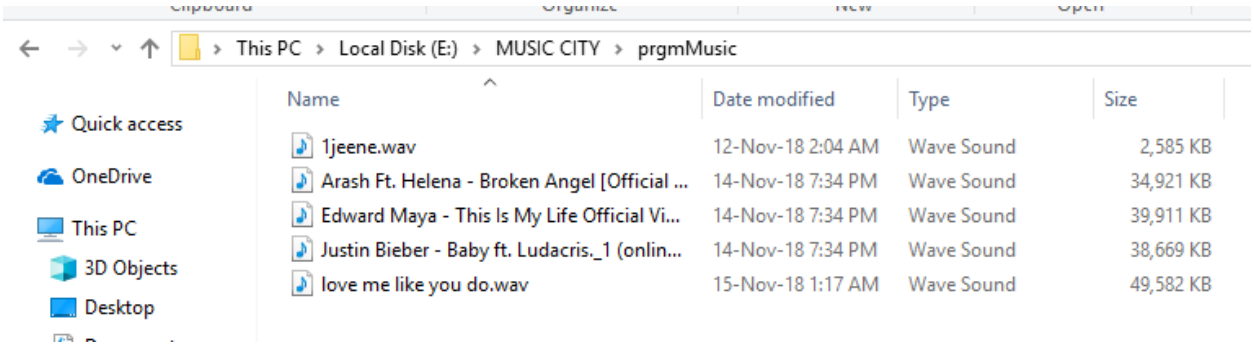

Fig 2 : Music files in .wav format

3. Steps for writing and executing java program in Eclipse

Step 1: Launch Eclipse

- Run "eclipse.exe" file which will be in eclipse installed directory
- Select workspace directory where you want to save the programming files
- If the "Welcome" screen appears close it by clicking on "cross" button which is next to the "Welcome" title

Step 2: Create a new Java Project

- Go to "File" menu → "New" →"Java project" (or "File" → "New" → "Project" → "Java project").

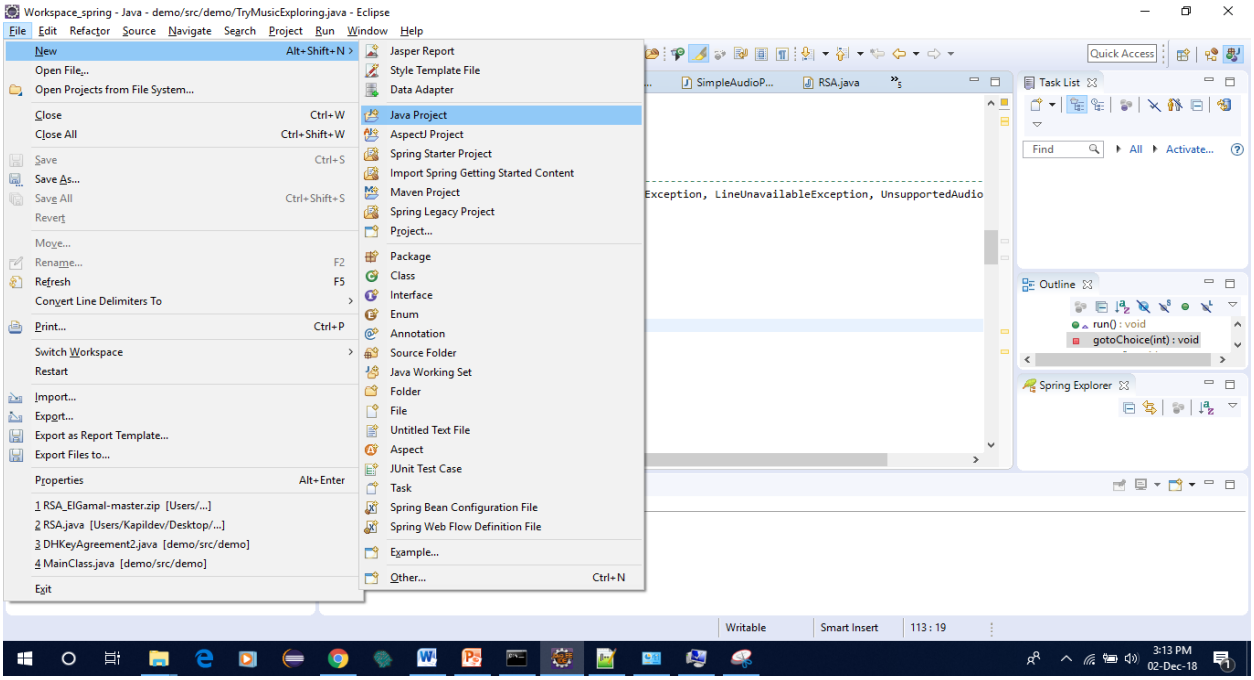

Fig 3: How to create Java project

- The "New Java Project" dialog pops up will appear.
    1. In "Project name", enter "`demo`".
    2. Check "Use default location".
    3. In "JRE", select "Use default JRE. Make sure that your JDK is 1.8 and above and then click on "Next" button.
    4. Finish.

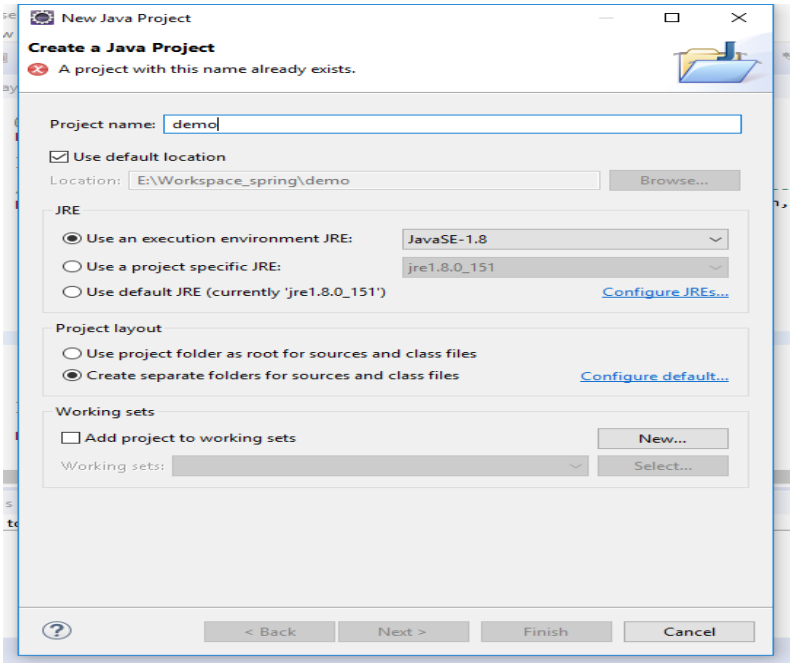

Fig 4: Writing first java program

Step 3: Writing Java Program

- In the “Package explorer” right click on “demo” folder →new→class
- New java class dialog box will appear

1. In "Source folder", write “demo".
2. In "Package", write “demo”.
3. In "Name", enter "`TryMusicExploring`". This is our main class.
4. Select "`public static void main(String[] args)`" to make this class main class.
5. Click on Finish button
6. Copy the source code and save it

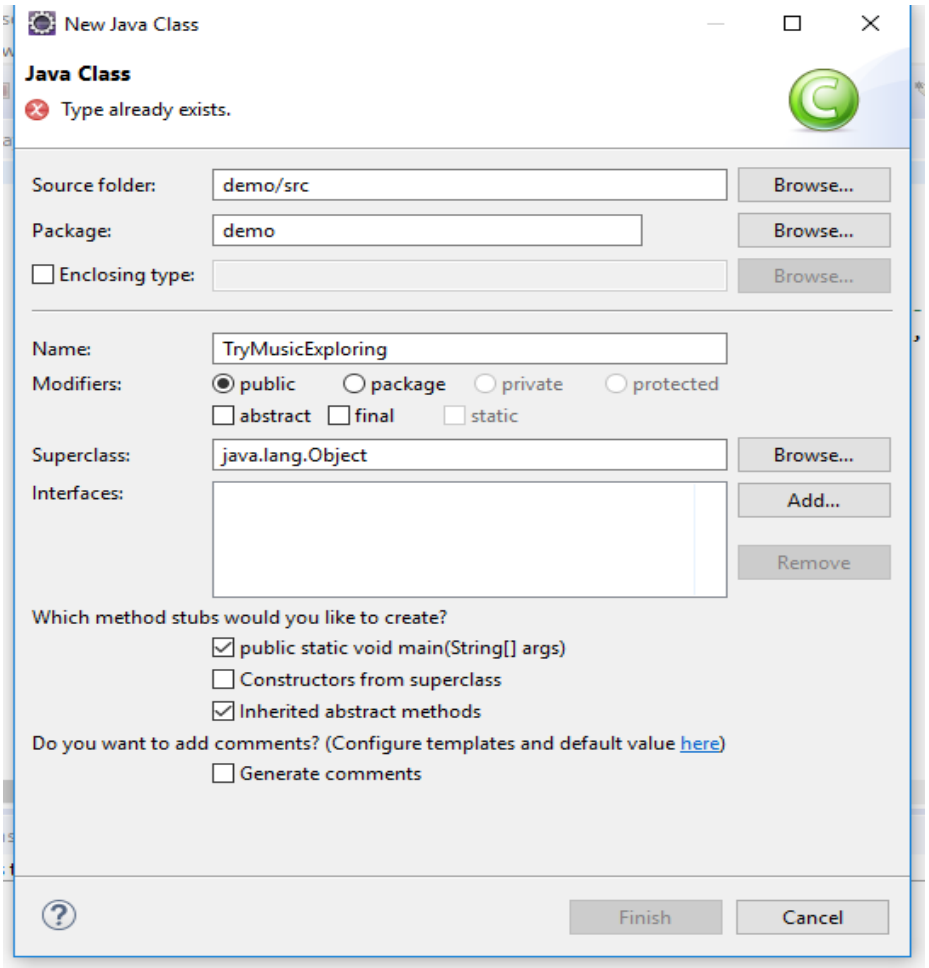

Fig 5: Giving Class name to java program

(Note: In screenshot, error means already file is created with the same name and two file cannot have same name in the same folder)

Step 4: Executing java program

1. Right click on your java main class file
2. Select “Run as”
3. Then click on Java Application
4. Program will start execution if there is no error or else it will throw exception or error

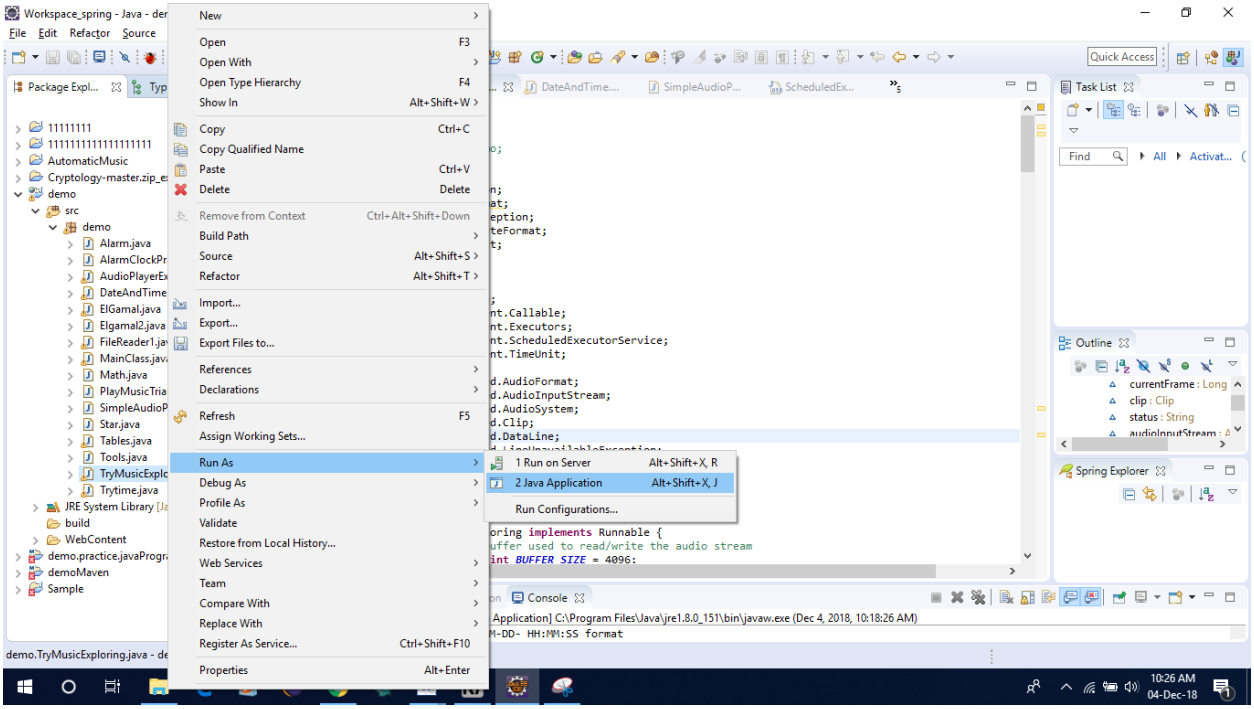

Fig 6: How to execute Java program

Code Explanation

- First we need to import the java libraries or packages which are required for the project. These packages provide a number of functionalities like input output file, exception, date format, arraylist, multithreading, music clip method, scanner for user input and output, unsupported audio file exception etc.

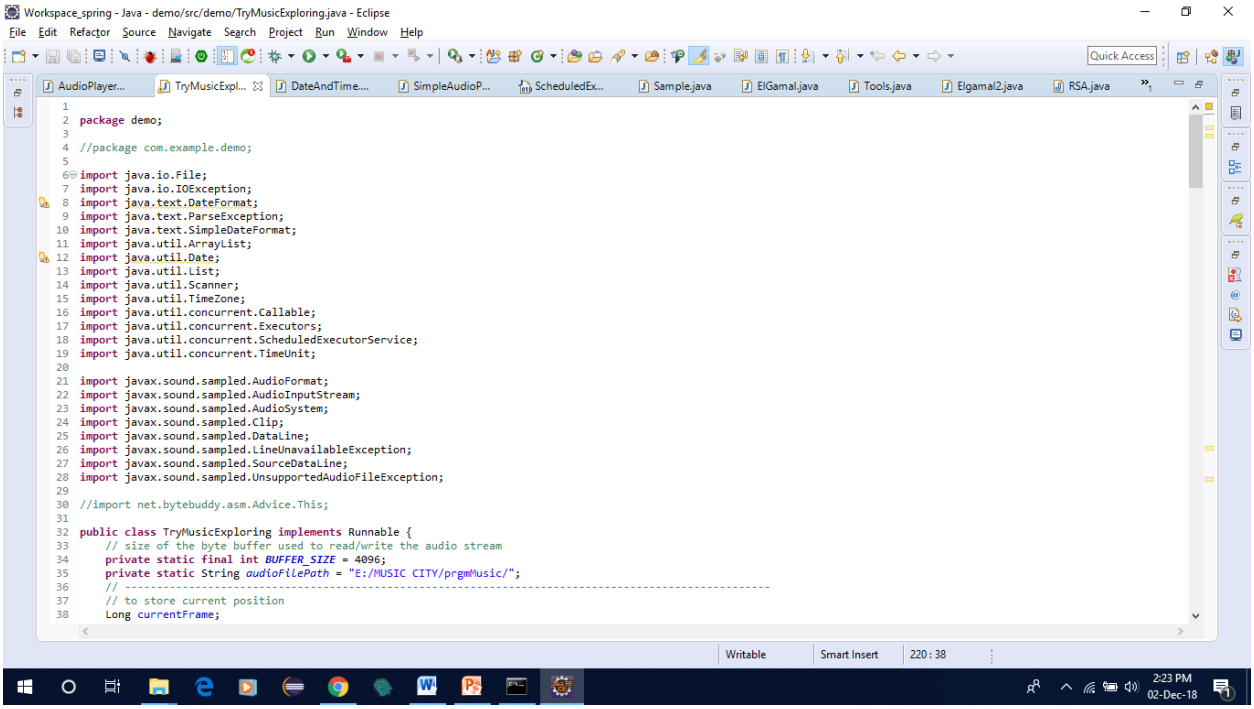

Fig 7: Java libraries and package

- Here we have implemented multithreading concept in our main class i.e TryMusicExploring class
- We have provided the path of the music folder in which songs are stored in .wav format
- Play method has been created in which three parameter is passed mainly file path, user starting time and user ending time respectively.
- Array-list has been created to play all songs one by one in a sequence

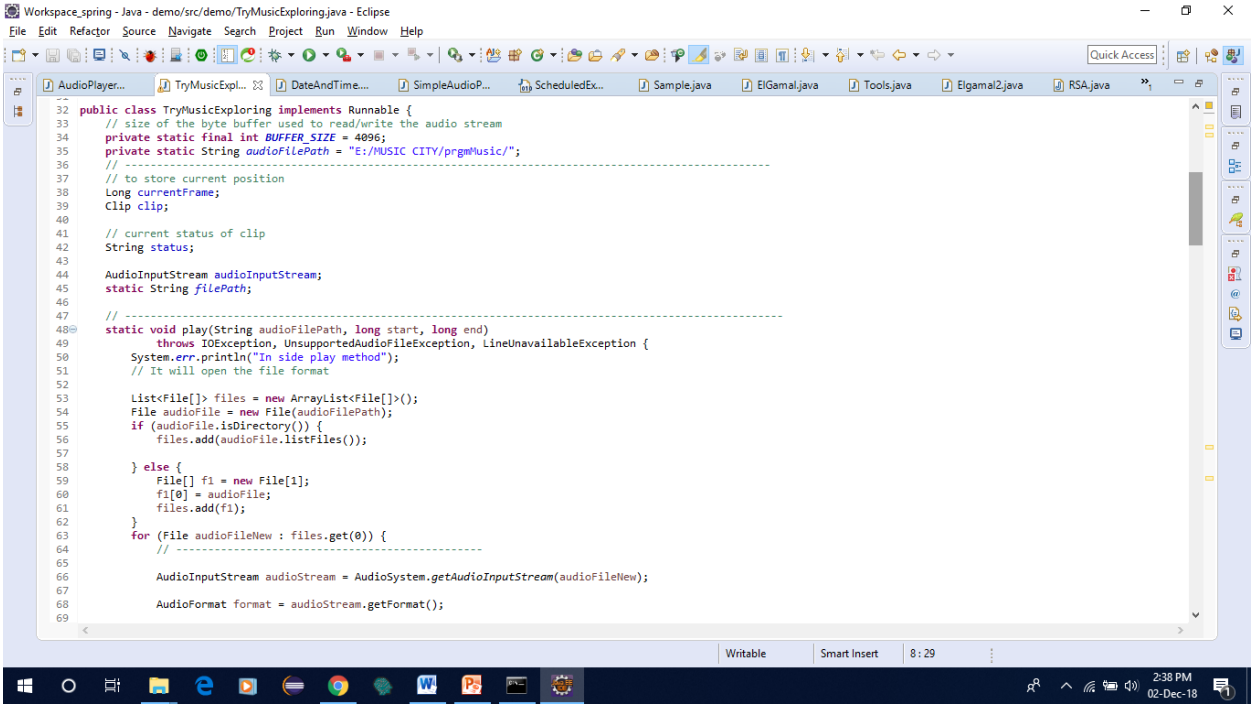

Fig 8: Play method and array list for files

- Here resetAudioStream method checks whether the music file is in .wav extension or not. If it is not in the proper format then it will throw exception
- In main method, we take starting time, ending time and date from user in the following format "YYYY-MM-DD- HH:MM:SS"
- System will calculate delay time which is
  Delay=User start time – current system time
  Which will be calculated in millisecond then we will convert it into second and delay time is shown to the user.
- This delay time is passed to the scheduler. System will wait for the delay time and it will automatically start playing song at scheduled time

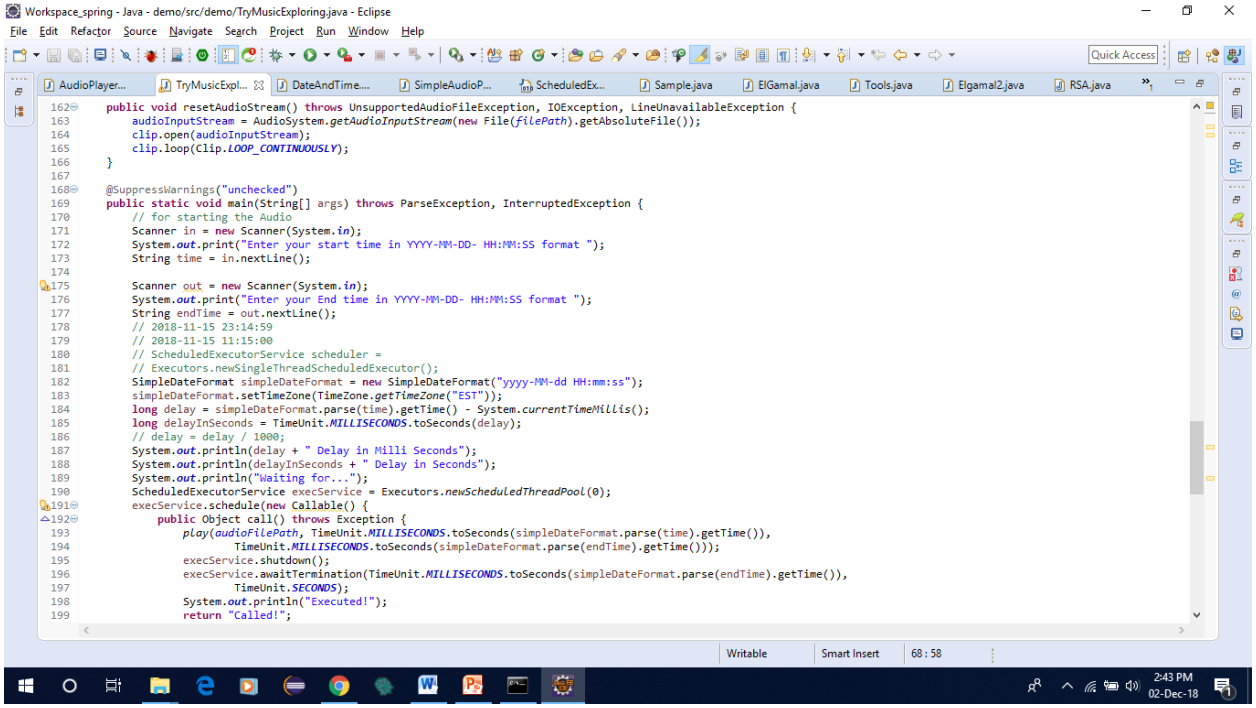

Fig 9: Showing main method and how input is taken from the user

Here three methods has been created pause, resumeAudio and restart. Pause method is used for pausing the video by the user when he presses specified key. Song will be resumed if the user play specified key for the resume. Restart method is used to restart the song id user presses specified key. In all three method Clip class method has been imported.

```java
    public void pause() {

        System.out.println("music paused");


          if (status.equals("paused"))
        {
          System.out.println("audio is already paused"); return;
        }
          this.currentFrame = this.clip.getMicrosecondPosition(); clip.stop();
          status = "paused";
          System.out.println("music 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.run();
          System.out.println("music is resuming.");

    }

    // Method to restart the audio
    public void restart() throws IOException, LineUnavailableException, UnsupportedAudioFileException {

          clip.stop();
          clip.close();
          resetAudioStream();
          currentFrame = 0L;
          clip.setMicrosecondPosition(0);
```

Fig 10: Method for pause, reset and restart music song

Below screenshot shows the method for taking input from the user when he presses specified key. While loop has been created so that it repeatedly asks questions to the user. For selecting specified key switch and case statement is used.

```java
    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;
        }
    }
```

Fig 11: User pressing specified key

```
            while (true) {
                System.out.println("1. pause");
                System.out.println("2. resume");
                System.out.println("3. restart");
                //System.out.println("4. stop");

                int c = in.nextInt();
                player.gotoChoice(c);
                if (c == 4)
                    break;
            }
            in.close();
        }

        catch (Exception ex) {
            System.out.println("Error with playing sound.");
            // ex.printStackTrace();

        }
```

Fig 12: For displaying options in the screen