UNIVERSITY OF LONDON

BSc Computer Science

CM2005 Object Oriented Programming

Endterm assignment

Introduction

In this course, we have developed a basic DJ application called Otodecks. For the end of term assessment, you are tasked with developing the application further by adding a custom deck control Component and a music library Component, then integrating them into a new GUI layout of your own design. The custom deck control Component should have custom graphics, implemented in the paint function and it should offer a means to control a deck in some interesting way of your own design. The music library component should allow the user to manage a library of music within the application. They should be able to search the music library and load music from it into the decks. It should also persist between application loads, so it will need to store its state in a data file.

Requirements

We will assess your work based on the following requirements and criteria:

R1: The application should contain all the basic functionality shown in class:

- R1A: can load audio files into audio players
- R1B: can play two or more tracks
- R1C: can mix the tracks by varying each of their volumes
- R1D: can speed up and slow down the tracks

R2: Customise the user interface (UI): i.e., change the colours, and change the layout.

- R2A: GUI layout is significantly different from the basic DeckGUI shown in class
- R2B: GUI code has at least one event listener that was not in the original codebase seen in class.

R3: Investigate and implement ONE new feature inspired by a real DJ program.

- To complete this requirement, you need to research DJ applications and identify a feature to implement. You do not need to run the other DJ applications - you can use YouTube and other resources to find out what features other DJ applications have.
- Implement a DJ-related feature that you have seen in another DJ program.
- In your report, you should present a screenshot and analysis of another DJ application and the feature you want to implement. Then you should explain how you have engineered this feature, talking about the classes and other OOP constructs you used to do the work.
- Finally, you should include the feature in the code and video that you submit.

Code style and technique

Your code should be written according to the following style and technique guidelines:

C1: Code is organised into header (.h or .hpp) files and implementation files (.cpp). Header files contain class interface definitions, cpp files contain implementations of class function members.

C2: Class interfaces in header files have comments for each public function describing purpose, inputs and outputs

C3: Code is laid out clearly with consistent indenting

C4: Code is organised into functions with clear inputs and outputs and a clear, limited purpose

C5: Code is stateless wherever possible – functions make use of data passing in preference to global or class scope data.

C6: Functions, classes and variables have meaningful names, with a consistent naming style

C7: Functions do not change the state of class or global scope variables unless that is the explicit purpose of a function (e.g., a setter)

What you should submit

You need to submit the following items:

- **Source code** in standard **ZIP** format.* Please take note of the code style and technique items above.
- **Code in PDF** format.* It should contain all the code formatted with clear comments showing which parts of the code you personally wrote without assistance.
- **Report in PDF** format.* Please include the following sections in your report:
  - Description of how the basic program works (R1)
  - Description of how you went about customising the user interface (R2)
  - Research and technical information covering how you identified, analysed and implemented a new feature that you found in another DJ application (R3)
- **Video demonstrating** your program in an **MP4** format. Maximum 5 minutes. The video should contain the following:
  - You viewing your code in your IDE, then launching the program from your IDE
  - You narrating (*audibly- not computer-generated voice or text*) what the code is doing as you click on buttons and other user interface elements
  - Demonstration of functionality pertaining to ALL the requirements.

* **Please note**: your CODE implementation will be marked based on your evidence provided in the CODE PDF file where you clearly indicate where the code is located for ALL TASKS that you have personally written without assistance. Remember to submit a REPORT as a PDF file providing details of how all the tasks are implemented, what logic/methods were adopted and why. Also, screenshots of the results/output with reference to code fragments (*i.e., copy/ paste code in the report or provide page numbers where relevant to the CODE PDF file)*.

If you do NOT submit CODE PDF AND REPORT PDF files along with the program files in a ZIP file, NO MARKS will be awarded for the respective tasks for the CODE implementation marking criteria.

Marking criteria

We will mark your work according to the set of criteria shown below, which consider the requirements, your programming technique and style and the documentation you have provided:

Tasks	Points
Submit correct files: code as text in PDF, report as PDF, a ZIP file for the code & a video*.	4
Code style: indentation, descriptive comments	10
R1: basic program	10
R2: custom user interface	10
R3: new feature	30
Report	30
Video	6
TOTAL	**100**