INTRODUCTION

In your role as a software developer, you have been contracted by QUB's Music Department to create a Java application capable of managing digital music (or sound) files for a Jukebox system. Specifically, you are required to build and test a prototype system that meets the deliverables outlined in the sections below.

GENERAL INSTRUCTIONS - CODE IMPLEMENTATION

1. Create a new Java project named 'Assessment2'.
2. Add three packages to 'Assessment2' named part01, part02 and part03.
3. Implement your code as described below:
    - 'part01' should contain all code associated with the core requirements.
    - 'part02' should contain the code related to implementation of additional features only, the only exception being the console application which you should copy from part01 to part02. Use import statements as required to access the definitions from part01.
    - 'part03' should contain any code relating to testing

SUBMISSION DATE/TIME – FRIDAY 26TH MARCH 2021 BY 5 PM.

PART 1: CORE FUNCTIONALITY (50 MARKS AVAILABLE)

To demonstrate that the proposed Java application is feasible, the following features must be implemented.

1.1 Implementation of the object classes, enumerator and interface shown in Figure 1 below.

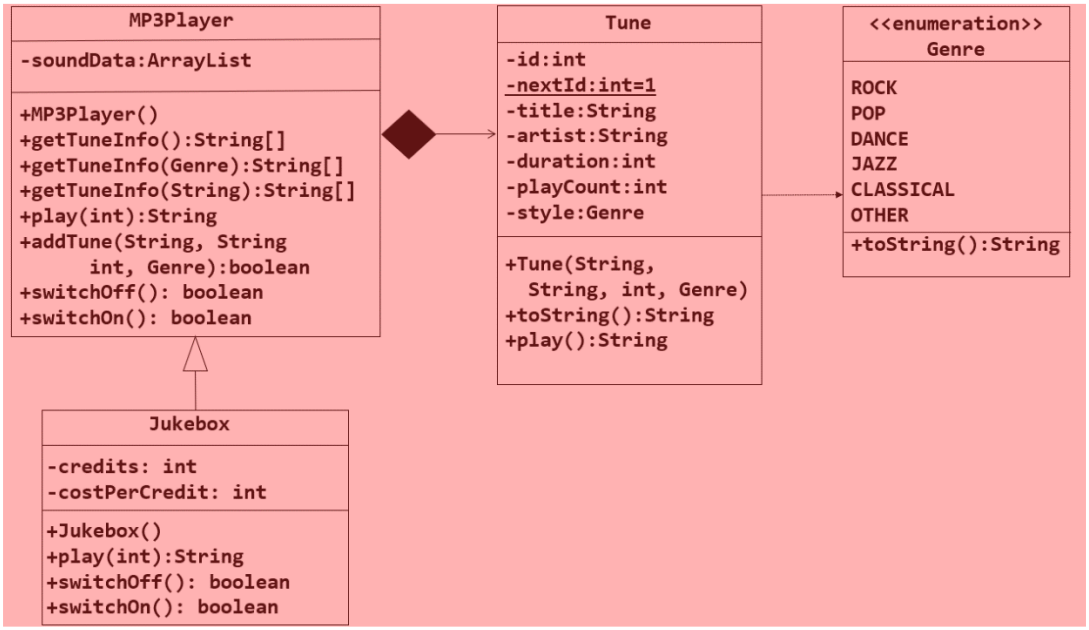

Figure 1: A (partial) UML Class Diagram for QUB Music Management Software.

Further details on the role and behaviour of the above classes follows – please read the entire document before starting. Canvas will also contain a list of frequently asked questions (FAQ) – a useful resource for clarification purposes.

Interfaces for ***MP3Player*** and ***Tune*** are available on Canvas to help you structure these classes. These will be discussed in class and must be used.

发生在MP3Player　　同时也和Tune有关联

1.2 A console (menu-based) application to manage (through an ***MP3Player*** instance) the interaction with ***Tune*** instances within the system. At this stage of development, there is no need to consider the ***Jukebox*** class. The following menu options should be provided:

- ~~***Select from Full List***: the user should be able to view summary details of ***all Tune instances*** in the system (listed one after another and ordered by title) and select one to be played (by id). If a selected ***Tune*** can be played, the application should report the details of the Tune with a message to say that it has been played (or an error)~~
- ~~***Select Tune by Artist***: the user should be able to view summary details of Tune instances (ordered by title) for a ***specified artist*** and select one (by id) to be played. If a selected ***Tune*** can be played, the application should report the details of the Tune with a message to say that it has been played (or an error).~~
- ~~***Select Tune by Genre***: the user should be able to view summary details of Tune instances (ordered by title) for a ***specified genre*** and select one (by id) to be played. If a selected ***Tune*** can be played, the application should report the details of the Tune with a message to say that it has been played (or an error).~~
- ~~***Add New Tune***: the user should be able to add a new tune to the ***MP3Player*** instance. The same tune (as specified by title, artist, genre and duration) can't be added more than once.~~ 在MP3Player上执行添加Tune的命令 同样的Tune只可以添加一次
- ***Display the Top 10***: user should be able to view a list of the top 10 tunes (names and number of plays) in order of number of plays.
- ***Switch Off***: the user should be able to turn the MP3Player off.
- ***Switch On***: the user should be able to turn the MP3Player on.
- ***Exit***: the user should be able to exit the system.

在开的情况下才可以关，在关的情况下才可以开。

Tune 要可以展现Tune的全部detail，还要可以play的method　　展示前十个Tune

选择了通过特殊的attribute来查找出来的Tunes要全部显示出来然后让我们选择，同时要显示他的id，因为我们是要通过id来选择。

要按照title来排序

通过id来选择Tune去play　　开关、离开按键

查询特殊的Tune

Notes:

添加Tune，并且不可以有重复的

A. The console application should create a ***MP3Player*** instance.

B. The console application should create and add a number (at least 5) of tunes to the ***MP3Player*** instance for testing purposes.

C. When adding a new tune, the user should be prompted for the title, artist, duration, style (Genre). 添加新的Tune时我们要提供title 之类的东西

D. When selecting from a full list of tunes, the user should be able to view the details of all available tunes within the ***MP3Player*** and be prompted to pick one for play using its (displayed) tune id.

E. When selecting a tune by artist, the user should be able to select an artist from a list and then able to view the details of all available tunes for that selection and be prompted to pick one for play using its (displayed) tune id.

F. When selecting a tune by genre, the user should be able to select a genre from a list and then able to view the details of all available tunes for that selection and be prompted to pick one for play using its (displayed) tune id.

G. When switching an ***MP3Player*** off, this can only be done if it is already on. When switching an ***MP3Player*** on, this can only be done if it is already off.

H. For enumeration ***Genre***, the ***toString***() method should return a String corresponding to the current value:

Value	***String***
ROCK	"Rock and Roll"
POP	"Easy Listening Pop"
DANCE	"Techno Dance"
JAZZ	"Smooth Jazz"
CLASSICAL	"Classical"
OTHER	"Unknown Genre"

I. You should make use of the ***Menu*** class defined in Menu.java (from the assessment specification on Canvas).

J. While implementing, you will find opportunities to use ***searching and sorting algorithms***. You must base any solution for these algorithms on the ***material presented in the lectures***. (i.e. you must implement yourself)

设定或者返回的method

K. You ***may not*** add additional public methods to ***MP3Player***. For class ***Tune***, you may add getters/setters as appropriate. You may add private attributes and methods to all classes, however, these must be justified (use comments).

我们不需要再添加public method 到MP3Player里面，但是可能会需要添加private的attribute或者method