# P3Purpose

Our company has been selected to build an **MP3 Tracker** application to download (legally, of course), catalog, and play MP3 music files. It is to be developed in several stages. In the first stage (this project), we are to implement an **MP3** class and demonstrate its functionality with a **driver** class. Below, please find detailed specifications.

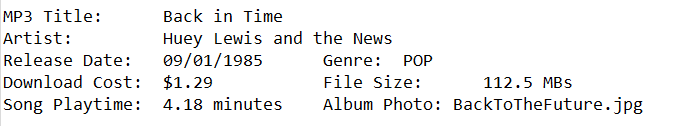
# Specifications

## The MP3 Class

The **MP3** class represents one music file. Each music file has the following attributes.

1. **Song Title** (String)
2. **Artist** (String)
3. **Song Release Date** (String)
4. **Playback time in minutes** (double)
5. **Genre** – an **enumerated type** (a separate Enum class) with possible values: **Rock, Pop, Jazz, Country, Classical, Other**. This type should be defined in its own **.cs** file.
6. **Download Cost** (decimal)
7. **File size in MBs** (double)
8. **Path** of a **.jpg file** containing the MP3’s **album cover photo** (such as **photos\FunkyTown.jpg**). *Note: This is just a string value. We’re not adding actual pictures for this project.*

The **MP3** class does no input or output. It should have standard functionality such as **constructors**, **getters**/**setters**, and a **ToString** method that formats an MP3 object for possible display by another class. An example of output a driver that uses an MP3’s **ToString** method follows

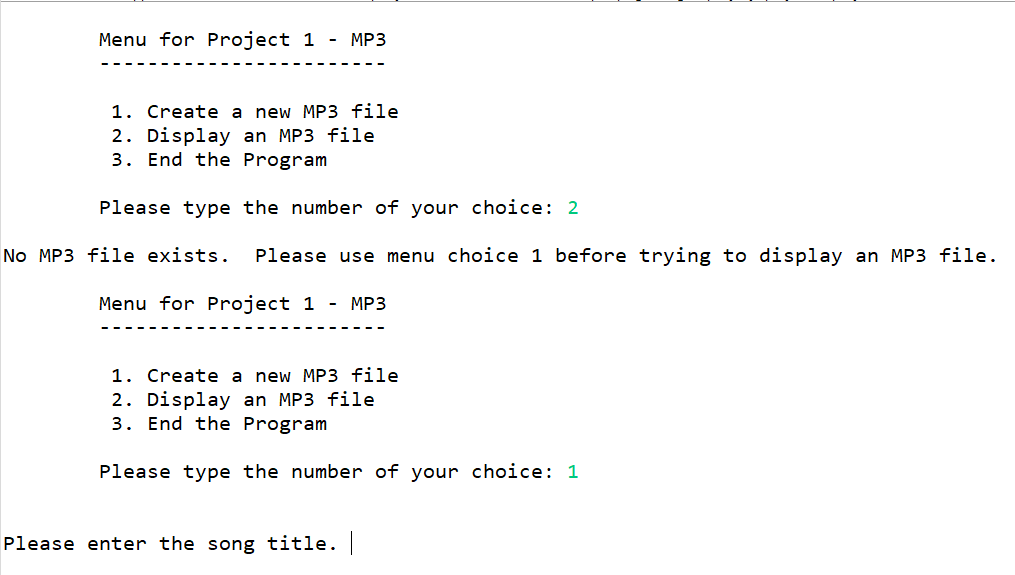


The **MP3** class may have additional methods as needed.

## The Driver Class

The **driver** class handles all interaction between the program and the user.

1. The **driver** should display a welcome message that includes a short description of what the program does and who wrote it.
2. The **driver** should prompt for the user’s name, and remember it.
3. The **driver** should be **menu**-**driven** and give the users the opportunity to
   1. Create a new **MP3** object and populate its attribute values from user input. Make this process as **user-friendly** as possible.
   2. Display an existing **MP3** object in a well-formatted manner that is easy for a user of the program to understand. If the user selects this menu choice, the program should verify that an MP3 object exists before trying to display it. See the following for an example of how the program should handle this situation.



* 1. Terminate the program.

1. You may use the instructor’s **Menu** class and the associated **Util** class or you may create your own.
2. The driver, in its current state, only needs to manage a single MP3 object. If additional MP3 objects are created, they can overwrite the previous MP3 object.
3. When the user indicates it is time to terminate the program, display a **goodbye/thank you message** that includes the **user’s name** (obtained in item 2 above).

# Other Requirements

Use good object-oriented principles and style.

* Create properties instead of setters and getters where possible.
* Use standard **C#** naming conventions for **classes**, **attributes**, **local** **variables**, **constants**, and **methods**.
* The **MP3** and **driver** classes and the **Genre** enumeration for this project must be in a **project** named **Project1MP3**. It is not necessary to move other classes to this project, but it will be necessary to add **using** statements for any files you use that is not part of the **Project1MP3** project. The finished project should look similar to the following in **Visual Studio**.

Graphical user interface, text, application

Description automatically generated

* Set up and use **Visual Studio** as described in the first lab session of this course.

# Project Documentation (via Comments)

**Project documentation is NOT optional in this course.** It is absolutely required in **every** programming assignment. See the **Documentation Standards** on D2L for details and for examples of proper documentation.

An otherwise perfect project may still receive a **failing grade** if the required documentation is missing, incomplete, or poorly done. Examples of proper documentation appear in the course **Documentation** **Standards** document

# Submission

Create a **design document** for the project. See the D2L for instructions and an example.

Your submission should be a single **zipped** file with a name in the format of ***1260-LastFirst-Project1*** containing **only** your **design document** and your **Project1MP3** code. Submit the **one** **zipped** **file** to your instructor via the D2L dropbox. Use Windows Zip or 7-Zip.