

COMP103 Course Project

MP3 Player Project Specification

Objectives

The objectives of the project is to test your knowledge of C# .NET by creating an application which satisfies the specification given below. This project is a formal piece of assessment and involves combining together all of the programming elements you have covered within a reasonably sized application. The final application must conform to the given specification, and be well structured and documented.

Reading

To complete this project you are expected to have completed practicals for Weeks 2 - 10. You will need the skills described in those practicals to complete the project.

Assessment

This project can be completed as a pair or individually. You are not allowed to use any code which you did not write unless you have clearly referenced the code in your project. You must sign the student declaration of originality on each coversheet to receive a grade for each project deliverable.

The project is broken up into two deliverables. The first deliverable requires you to create a paper prototype of the user interface for your project application and hand it in for marking. Read the entire project specification before you begin creating your paper prototype and application. You are allowed to make changes to your prototype later when you are creating the application.

The second deliverable requires you to create a C# application which satisfies the specifications given below, get it verified by a demonstrator during a 103 lab session and hand in the coversheet. Your project implementation will be copied from your account after the due date and then your code will be marked by a tutor/lecturer if you have handed in your coversheet.

Specification

You will be creating a simple software MP3 player using C# .NET. The player will allow the user to create a playlist and then add songs (mp3 files) to the playlist. The user can then save the playlist and re-open the playlist at a later date. The user can create as many playlists as they want but the player will only have one playlist open at a time.

Also the user will be able to play songs from the playlist, stop the currently playing song, resume the current song from where it was stopped. It will also allow the user to create a report file about the songs in the playlist.

Song Management

When a song is added to the playlist the following information about the song needs to be stored in the playlist:

ID number	Unique integer ID number. The ID will already be automatically generated by the application when a new song is added to the playlist .
Filename	The full path and filename of the song.
Rating	The user rating of the song. A value from 1 to 5, with 5 being the highest rating.

The above information about a song must be stored in a C# struct. **You must use a List to store all of the songs in a playlist.**

The application will have appropriate controls to allow the user to enter information about a new song or change information about an existing song in the playlist.

The ID number for existing and new songs must be displayed in a read-only control. The full path and filename of the song the user has chosen to add, should also be displayed in a read only control. Next to that control should be a button to allow the user to select the path and filename of the song the user wants to add, using the OpenFileDialog control. The rating of the song should be displayed in a text box or other control to allow the user to change the rating if they wish.

The application will also have a listbox displaying the filename of all the songs currently in the playlist. It should just display the filename only, excluding the path. When the user clicks on the name of a song in the listbox, the song's details will appear in the appropriate controls.

Playing Songs

The application will need buttons on the form to play the song which is currently displayed in the song detail controls, stop the currently playing song and resume the current song from where it was stopped. It will also need a control to display the elapsed time and the duration of the song. If the song detail controls are blank then no song is played.

The Template Project

A template project has been created for you for this project. Go to the **COMP103** library folder and copy the folder **MP3 Player 2009** into your account or download it from Moodle.

The template project includes a Dynamic Link Library (.dll) file (created using Visual Basic) which will allow you to create a player object which has methods and properties to handle playing MP3 files. This library has already been added to your project.

It also contains a method defined as shown below:

```
private string GetSongName(string filename)
```

If you pass the method a complete path and filename then it will return back just the filename. I.E. if you passed the following string to the function "H:\My Music\Alchemy Live\Tunnel of Love.mp3", then the function will return back "Tunnel of Love.mp3".

Instructions on how to create the player object in the code and how to play the songs will be provided at a later date in the tutorials.

Sample MP3 Files and Playlists

Sample MP3 files and playlists will be provided in the COMP103 library folder in a folder called **Songs**. You need to copy the folder into your My Documents folder (H: drive) otherwise the sample playlists will not be able to find the songs.

Tasks

- a) Create the following menu structure:

File

New Playlist

Open Playlist...

Save Playlist As...

Playlist Report by Rating

Exit

Song

Add/Update Song

Clear Song Information

You should give each menu item appropriate access keys and shortcut keys.

- b) Create the user interface for your player based on your paper prototype. If you wish to make changes to your paper prototype then you can and you don't have to redo the prototype.
- c) Create the C# struct definition to store details about a song and then create a class scope List to store each song in the playlist.
- d) Create a method called `ClearControls` which sets the ID label to the next ID number to use, clears the filename label and clears the rating control.
- e) Create a method called `UpdateListbox` which goes through each song in the arraylist and adds just the filename to the listbox.
- f) Create a menu click event method for the **Exit** menu item which will exit from the application.
- g) Create a menu click event method for the **New Playlist** menu item. Write the code that will clear the song detail controls, clear all names from the listbox, and the titlebar of the form should be set to "New File — MP3 Player". Also a new List should be generated.

- h) Create a menu click event method for the **Open Playlist...** menu item. Write the code that will allow the user to select the playlist they want to open using the **OpenFileDialog** control. All playlist information is stored in a CSV file. So the file will contain one CSV line for each song. Once the playlist is opened, the application should read each song's information from the file and store it in the List until there is no more information in the file.

Then the names of all the songs in the playlist should be displayed to the listbox. The titlebar of the form should be set to "<path and filename of playlist> - MP3 Player". The song information should be cleared and the ID label should be set to the next ID to be used for a new song.

You can test this by opening the sample playlists provided with the sample MP3 files.

- i) Create a menu click event method for the **Save Playlist As...** menu item. Write the code that will ask the user for the file they want to save the information to, using the **SaveFileDialog** control. The information about each song should be saved to the file on one CSV line, therefore it should have exactly the same structure as the original file. Each song's information should be written to the file until every song has been processed in the List. The titlebar of the form should be set to "<path and filename of playlist> - MP3 Player".
- l) Create a **SelectedIndexChanged** event method for the listbox control. Write the code that will allow the user to click on a song name in the list box and then the song's information is displayed in the appropriate controls.
- m) Create a menu click event method for the **Clear Song Information** menu item. Write the code that will clear all information displayed about the currently selected song and the ID label is set to the value to be used for the next new song.

Note: **DO NOT** clear all of the song filenames from the list box, that is only done when a new playlist is created.

- n) Create a menu click event method for the **Add/Update Song** menu item. Write the code so that if the information to be stored is for an existing song, then the song information is stored back into the correct place in the List. If the information is for a new song then the information is added to the end of the List and the listbox is updated.

- o) Create a click event method for the **Play** button which will play the song that is currently displayed in the song detail controls. When the song is playing the elapsed time will be updated every second.

Hint: You may need to use a timer control to update the elapsed time.

- p) Create the click event method for the **Stop** button which will stop the song which is currently playing.

- q) Create the click event method for the **Resume** button which will resume playing a stopped song from where it was stopped.

- r) Create a menu click event procedure for the **Playlist Report by Rating** menu item. The report should display the song ID and path and filename grouped by rating, **with the information written to a file**. Allow the user to select the location and name of the report file. The report should be aligned neatly so that the information is in columns and should have appropriate headers for each column. I.E. the report should have the following structure:

```
Rating 1
<name>                <path and filename>
<name>                <path and filename>
...
Rating 2
<name>                <path and filename>
...
etc
```

Hint: Use the PadRight or PadLeft methods to align the information.

Project Deliverable 1

Create a paper prototype of your application for the specification given previously. Your paper prototype should include your form and the menu structure. Your paper prototype must be hand drawn and hand written and has to be neat and readable. On your sketches you must state the type of each control in your paper prototype and give a brief description for menu events and button events.

Note: You are allowed to change your prototype when you actually start building the application for the project.

What to Hand In

You should attach your paper prototype to the Project Deliverable 1 Cover Sheet. It will be assessed on the completeness of the prototype, layout of the form, use of appropriate controls, description of events which will occur and the readability of the paper prototype.

Project Deliverable 2

Create an application that satisfies the specification given previously. The user interface of your application should match your paper prototype you created previously but you are allowed to make changes to your paper prototype during the application development if you want to.

What to Hand In

The second project deliverable requires you to get it verified by a demonstrator during a lab session.

The project will be graded half on how the application runs and half on the structure and quality of your code, appropriate use of the C# language, use of methods and documentation of your code.

When your project is verified by the demonstrator they will make sure that the project is on your desktop. **Do not rename the folder from the default name.** This folder will be copied later from your account so that the tutor/lecturer can mark the quality of your code. Therefore you only need to hand in the Project Deliverable 2 coversheet with the demonstrator mark and verification stamp on it. **You DO NOT need to hand in a printout of the code.**

Verification Tasks

The following is a list of the tasks the application should be able to perform. The demonstrator will give you 1 mark for each task that works as specified in the project.

1. Create a new playlist
 2. Open an existing playlist
 3. Path and name of playlist file shown in titlebar of window
 4. Filenames of songs are displayed in a listbox
 5. Song information displayed when name clicked on in listbox
 6. ID is automatically generated when adding a new song
 7. New song can be added to playlist
 8. Existing song information can be changed and updated
 9. All song information can be saved to new playlist
 10. Displayed song information can be cleared
 11. Plays currently displayed song
 12. Stops currently playing song
 13. Resumes song from where it was stopped
 14. Creates the Playlist by Rating report file
 15. Have created the menu structure specified.
-