Review of implementation

Media Organiser – Harry Wigman – 29/04/19

# Software used

To create the application, I used Windows Presentation Foundation (WPF) in the C# language. The work was done in the Visual Studio 2019 IDE, it followed the MVVM (Models, Views, View Models) architectural pattern with implementation using Caliburn.Micro framework.

# Limitations of design/implementation

During my implementation I came across some limitations:

Playlists can’t be uniquely ordered – While you can search a playlist and order the list by the any of its values, you can’t order it specifically for the playlist you’re viewing. This could be implemented through adding numbering while adding a related playlist to media files.

Design windows – When creating the designs, I intended to have a different window per each view. However due to the design methodology of my MVVM framework, it became easier to have one window with multiple views. Overall, I feel this was a positive change and keeps the application together.

Can’t reload available playlists filter – Once you’ve added a new playlist you currently need to re-start the application for it to show as a filter. I was unable to find a way to subscribe the element to update on changes, this would require further investigation into the MVVM framework.

# Future improvements

The following is a list of improvements which were out of scope for the initial project spec but could be added in future.

* Media files could show a combo box of the different associated playlists/categories instead of the current text solution.
* When editing a list in one place you could set it so that it updates any and all occurrences elsewhere.
* You could add an options menu where the user could specify application settings. For example, which file extension the application would accept during a search.
* The home screen could have more information added to it to make it more useful. This could include number of playlists, number of media files recorded, last performed search etc.
* The application could inherit a UI design language like Google’s [Material Design](https://material.io/design/).
* The application could use default pictures for audio and video. When a media file is added for the first time, it could inherit either the default audio or video picture depending on the file extension.