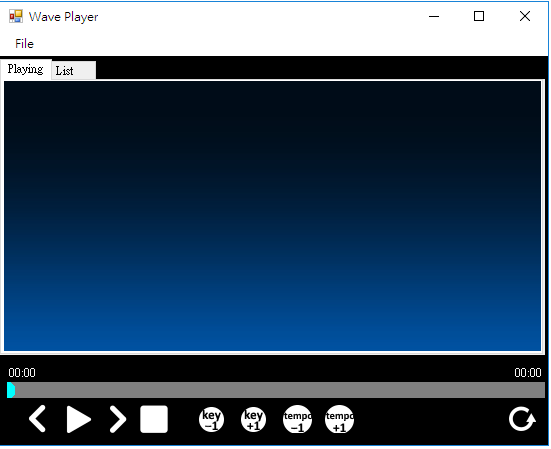
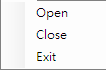
Wave Player

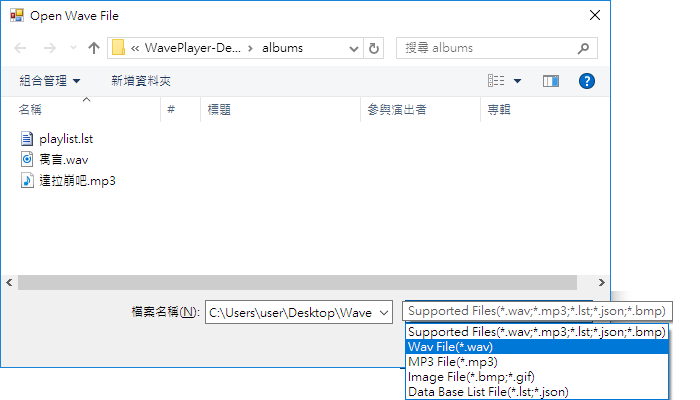
UI總覽



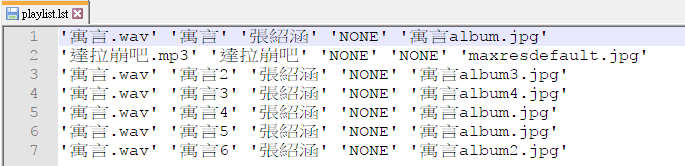
程式提供一個File Menu，裡面有Open、Close、Exit。



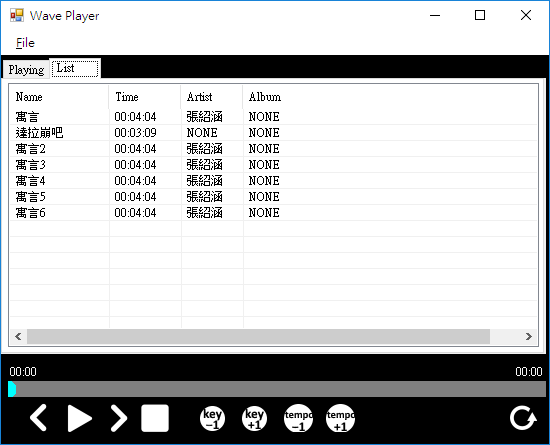
開啟檔案支援wav, mp3, lst, json, bmp以及gif



其中lst檔案為一個以單引號成對的檔案，如下圖所示



畫面呈現一個Tab Control，分別是Playing與List，Playing會顯示音樂資訊，當使用者開啟的是PlayList(\*.lst)，會顯示<歌手> 專輯 – 歌名在左上角。如果歌手名稱為NONE，會顯示<Unknown Artist>，若專輯名稱為NONE，則後方專輯名稱會以歌名顯示。若歌名為空白，會取檔名作為歌名。當使用者開啟lst檔後，程式會用MusicPlayList以Regular Expression解析每一行的資訊。由左到右個別為檔名、歌名、作者、專輯名稱、專輯封面。開啟後，主程式會將MusicPlayList的內容匯入到List頁面內。並以Name, Time, Artist, Album顯示。



Time欄位會分兩種檔案擷取方式，一種是Wav，另一種是MP3，當檔案為Wav時，會開啟WavFormat/Format/WaveFile.cs的WaveFile解析。一個Wave檔的時間長度為

Duration=DataSize/(Channels\*SampleRate\*(BitDepth/8))。當檔案為MP3時，會使用NAudio的MP3FileReader取得TotalSeconds來獲得Duration資訊。

如果PlayList有指定專輯封面，playing會將圖片顯示出來，當成專輯封面。



同樣的，JSON為標準的JSON格式，我們的播放器增加了Cover欄位，用以代表音樂播放時顯示的封面。

作為應用程式的資料庫，data.json會在WavePlayer啟動時自動載入，並且成為預設的音樂清單。WavePlayer會利用newtonsoft.json解開。

當音樂檔的資料夾內有.lrc(字幕)檔案時，播放器會利用LyricParser解析字幕，並將字幕放在一個<Time, String>的串列。當使用者播放進度改變時，播放器會利用當下的時間去LyricParser查詢符合播放時間區間的字幕並顯示。

在畫面下方的控制面板會顯示目前時間、總長度，一個可以點擊、拉動的進度條，倒退(Rollback)，播放/暫停、快進(Fastforward)、停止播放、降key、升key、降tempo、升tempo以及重複按鈕。



由於播放器以及相應的檔案串流只能提供基於檔案位置與檔案長度的位置資訊，要顯示當下的時間，就必須以Duration做為參考基準。利用串流總長度 / Duration獲得每一個單位位移的時間，再乘上進度就能獲得目前播放的總秒數。

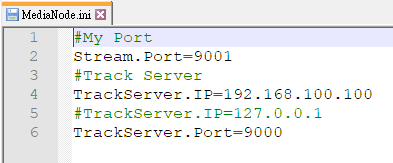
當播放器提示播放進度更新時，更新的時間會回饋到UI，用播放時間/Duration獲得比例後，再改變ProgressBar的數值就能呈現出播放進度。當使用者點下ProgressBar的一個位置時，播放器會計算相對ProgressBar的點擊位移，再利用位移除上總長度換算進度比例，再利用進度比例乘上Duration得到預計移動的時間位置。這時程式就會對播放器的實體要求更新播放位置來達到SeekTo Position的效果。

當使用者點下倒退鍵，播放器會在使用者放開之前，在每一次更新進度時要求播放器將播放位置-1，相對的當使用者按下快進鍵(Fastforward)，則播放器會在每次更新進度時要求將播放位置+1。因為更新進度的來源是來自DirectX的Buffer，當Buffer要求應用程式填入聲音時，播放器就會提出目前進度更新的訊息，播放器會利用這個快速的要求來達成Rollback以及Fastforward。相對的，由於更新畫面不需要這麼頻繁，播放器在時間更動達一秒的播放進度時再更新畫面，如此一來，就能順暢地播放且即時更新播放進度。

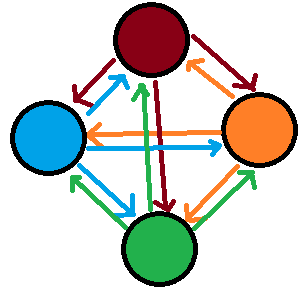
key-1以及key+1按鈕允許使用者改變播放器的音調，播放器會將key限制在-12到+12之間。當使用者按下key-1、key+1時，播放器會向SoundTouch要求改變Pitch，當SoundTouch的Buffer用光時，下一個Buffer就會收到調音的結果。同樣的，tempo -1以及tempo+1是用以讓使用者改變播放速度，每一個單位為0.1，播放器會將播放速度限制在0.1~2.0之間，當使用者點擊tempo -1、tempo +1，播放器就會向SoundTouch要求改變Tempo，不同的是，當使用者要求Tempo -1的時候，如果當下的tempo已經小於1，因為前一個Buffer的資料長遠超過1秒，為了提高使用者體驗，讓點下後的tempo可以立即反應到播放中的緩衝，播放器會要求SoundTouch清除掉Buffer，如此一來新的取樣聲音就會套用效果。

Repeat按鈕是一個Toggle Button，使用者按下後，如果播放的是wav或mp3檔案，當播放結束後，會重新播放。而如果使用者正在利用播放清單播放，播放器只會在最後一筆音樂播放完畢才才回頭播放第一個音樂。

當應用程式啟動後，會載入MediaNode.ini，裡面存放的是一個媒體節點的設定值，Stream.Port代表本地節點的Port；TrackServer.IP以及TrackerServer.Port個別代表Tracker節點的IP Address以及Port。



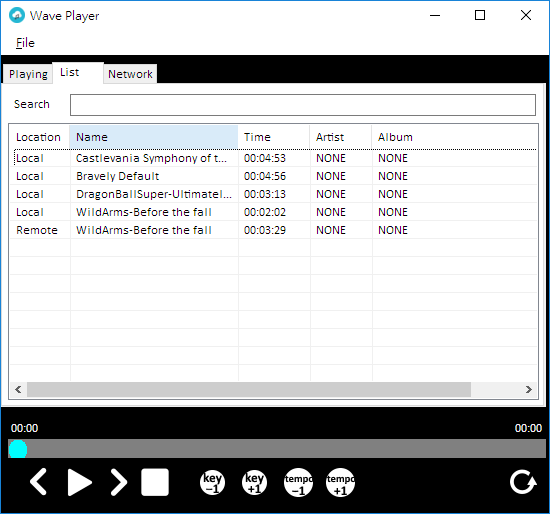
應用程式啟動後會啟動MediaNode(WavePlayer/MediaServer/MediaNode.cs)，MediaNode代表個別的多媒體節點。當MediaNode啟動後，會向TrackerServer註冊，傳送自己的可用Port，並且向TrackerServer擷取目前的MediaNode清單。MediaNode會維護與TrackerServer的連線，當TrackerServer不存在或者斷線時，MediaNode會週期性地嘗試連線。MediaNode會與TrackerServer建立兩條連線，一個做為主動溝通，另一個做為被動通知。當一個新的MediaNode加入到TrackerServer時，TrackerServer會對目前所有連接的MediaNode發送新的節點清單。當MediaNode收到節點清單更新時，會向其他尚未連接的MediaNode建立一條連線，如此一來就能做到類似Mesh的連線拓樸。



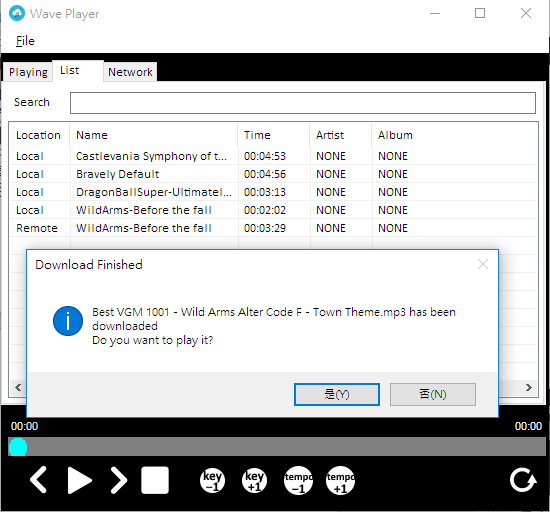
收到新的連線後，播放器會向其他節點要求取得音樂資料庫，並且合併成一個整合過的清單。在每個音樂來源都含有一個RemoteLocation欄位。合併過程中如果本地資料庫含有這份檔案，就會填入localhost，而當清單來自其他MediaNode，就會附上其他MediaNode的連結key。連結key以IP+ProcessID表示。當RemoteLocation不含有localhost時，FromRemote就會成為true，這時代表這份檔案不存在本地檔案。

|  |  |
| --- | --- |
|  | 不含有localhost的MusicListItem屬於remote |

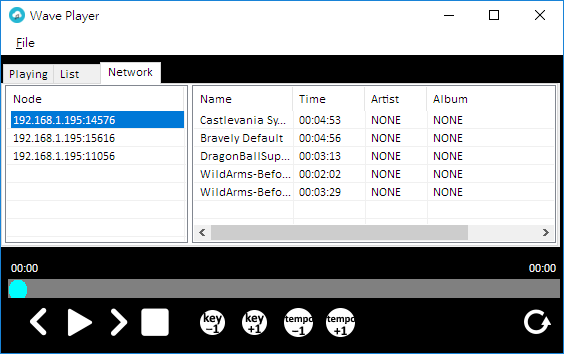
這時在音樂清單就會產生Location為Remote的紀錄。



對著Remote點兩下會向所有連接的MediaNode要求同樣檔名的檔案—亦即JSON描述中，檔名相同的紀錄。要求的MediaNode會以key查詢連線，向所有節點詢問檔案大小，取得最小的檔案大小後，會平均切割需要的檔案區塊，再利用直接的連線來傳輸檔案。檔案會被下載到播放器的Download資料夾，並且以xxx.p0,xxx.p1 ... 命名。當全部檔案下載完畢後，會將個別的檔案合併成一個檔案，並且更新data.json，再詢問使用者是否播放剛下載的檔案。



在Network頁面，可以看到所有已經建立連線的節點，單點可以看到目標節點的資料庫，點兩下Node會開啟串流。

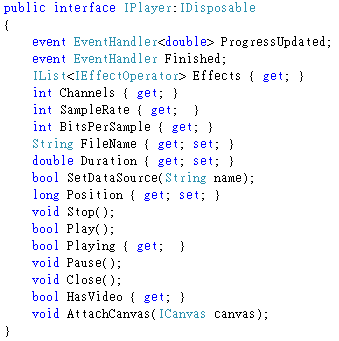


啟動串流時，MediaNode會向目標MediaNode發出MEDIA\_NODE\_SUBSCRIBE\_STREAM\_REQUEST來表示要Subscribe接下來的聲音串流。被Subscribe的MediaNode收到訊息後，會為Player啟動StreamBufferSharingEffectOperator，StreamBufferSharingEffectOperator是在IEffect的架構之下，會將經手的聲音取樣暫存到一個佇列，並且以一個額外的執行緒用MEDIA\_NODE\_REALTIME\_WAVE\_STREAM\_PUSH訊息推向所有Subscribe的MediaNode。

主動訂閱端的MediaNode在發出Subscribe後會以RemoteRealtimeStreamPlayer播放，當開始播放時，RemoteRealtimeStreamPlayer會向目標MediaNode要求MEDIA\_NODE\_GET\_STREAM\_WAVE\_FORMAT\_REQUEST，取得播放中的WAVE\_FORMAT，用來同步播放的WAVE格式。接著RemoteRealtimeStreamPlayer會向本地的MediaNode註冊OnStreamBytePushed事件，當收到目標的MEDIA\_NODE\_REALTIME\_WAVE\_STREAM\_PUSH，就會收到OnStreamBytePushed，這時RemoteRealtimeStreamPlayer會將收到的取樣填入一個串流緩衝MediaNodeWaveStreamWrapper，當播放期間DirectSound裝置會向MediaNodeWaveStreamWrapper取出填入的取樣並且播放。

UI下方的key+1, key-1, tempo+1, tempo-1用以調整音調以及節奏，當key以及tempo有所變動時，title bar會顯示目前的key以及tempo。

播放器以一個Interface IPlayer作為最上層，提供播放器基礎功能



IPlayer介面繼承IDisposable，當Dispose時，播放器要主動釋放掉目標資源，例如關閉DirectX的聲音裝置。

|  |  |
| --- | --- |
| event ProgressUpdated | 用以通知播放進度變更的callback |
| event Finished | 通知目前的音樂已經播放完畢的callback |
| Effects | 一組用以作為特效的結果，每個IEffectOperator提供Handle方法，接收EffectEventArgs參數。可以將收到的聲音片段轉換或作為視覺顯示。 |
| Channels | Channel數量 |
| SampleRate | 取樣率 |
| BitsPerSample | 又稱BitDepth，每一個取樣的Bit寬度 |
| FileName | 檔名 |
| Duration | 音樂檔案的時間長度，單位是秒 |
| SetDataSource | 設定目標音樂檔案 |
| Position | 目前的播放位置，單位是秒，可以用來改變播放的位置或者快進/倒退 |
| Stop | 停止播放，回到開頭位置 |
| Play | 開始播放 |
| Playing | 是否播放中(狀態) |
| Pause | 暫停 |
| Close | 關閉檔案 |
| HasVideo | 是否有Video(用於BMPPlayer, GIFPlayer) |
| AttachCanvas | 附加到ICanvas介面，用來將畫面更新到Canvas |

Player/WavPlayer.cs以及Enhance/MP3Player.cs實現IPlayer，會個別解開支援的檔案格式，並將串流交給StreamedSound，驅動DirectSound播放聲音。

使用third party library:

|  |  |  |
| --- | --- | --- |
| 名稱 | 用途 | 直接類別包裝 |
| Managed DirectSound | 讓.NET可以播放聲音取樣 | StreamedSound.cs |
| SoundTouch.Net | 調整音高、Tempo | Enhance/SoundTouchEffect.c |
| NAudio | 支援MP3 | Enhance/MP3Player.cs |
| LyricParser | 解析歌詞、以時間尋找歌詞 | WavPlayer/LyricParser.cs |
| Newtonsoftware.Json | 用以解開json檔 |  |