基于 MFC 的 MCI 音乐播放器的设计与实现

王婷 白俊鸽
四川大学锦城学院计算机与软件学院 四川 成都 610000

【摘 要】设计并实现了一个基于 MFC 的 MCI 音乐播放器，该音乐播放器由 MFC 开发，使用系统的 MCI 命令接口进行编程，能够播放本地音频文件。

【关键词】音乐播放器；MCI；MFC

引言

目前市面上的音乐播放器功能都很强大，但是在这些软件中经常会出现广告影响用户的体验。本文讨论了基于 MFC 的 MCI 音乐播放器，其操作简单，界面简洁，占用空间小，实现了播放本地音乐的功能。

1 MCI 接口的介绍

MCI 是一种用于控制多媒体设备的高级命令接口，包括 CD 音频、视频动画、Wave 格式数字声音和 MIDI 音序器等 [1]。

应用程序通过向 MCI 发送命令来控制媒体设备。MCI 命令接口有两种方式实现，分为命令字符串和命令消息，这两个命令都具有相同的功能。前者使用起来很简单，但它的执行效率低于后者。本文中采用的是命令消息的方法。

2 主要功能描述

音乐播放器实现的是播放本地的音频，主要分为三个部分的功能，包括播放列表的显示功能，播放音乐的控制功能以及当前播放音乐的状态显示、控制功能。

用户可以通过播放列表的显示，能够看到歌曲序号、已经添加的歌曲和高亮展示的当前正在播放的歌曲，方便用户在播放的时候进行歌曲的切换。播放音乐的控制功能实现了一些基本的音乐操作，包括打开文件、播放、暂停、切换播放模式、上一首、下一首、删除歌曲、清空播放列表，用户可以操作这些功能来播放音乐。当前播放音乐的状态显示，实现了当前播放歌曲的歌名的显示，播放进度的显示和当前歌词的显示，使用户清楚地知道当前播放歌曲信息。控制功能则实现了音量的控制，播放进度的控制，方便用户对当前歌曲进行控制。

3 界面设计

3.1 播放列表

播放列表主要显示的是已经添加的歌曲的信息，包括歌曲的序号，歌曲名 (包括歌手和歌名)。

第一列显示歌曲的序号，从零开始编号，第二列显示的是歌曲名，第三列显示的是歌曲的路径，但在界面初始化时就被隐藏了。用户可以通过双击界面上的歌曲来播放音乐，也能够通过看到高亮显示的行，从而知道当前播放的是哪一首歌曲。

3.2 播放控制台

播放音乐控制台显示了一些基本的音乐操作，包括打开文件、播放、暂停、切换播放模式、上一首、下一首、删除歌曲、清空播放列表，用户可以通过单击功能所在的按钮来执行此操作。

3.3 播放状态

播放状态主要显示了当前播放歌曲的歌名，当前播放歌曲的进度，当前播放歌曲的歌词以及当前播放歌曲的音量。用户可以通过播放状态了解当前播放歌曲的信息，也可以通过拖动或单击音量进度条和播放进度条来控制当前播放歌曲的音量和当前播放的进度。

4 功能实现

4.1 播放列表

播放列表是通过 MFC 的 list control 控件实现。在打开文件或将文件拖拽到列表控件中之后，通过调用自定义函数 AddMusicToList() 实现向列表控件中添加歌曲序号、歌曲名、歌曲路径。歌曲路径所在的列在界面初始化时被隐藏，为的是不让用户看见，但是在编写软件时，为了能够通过得到当前播放歌曲的位置从而获取歌曲路径来播放音乐，所以添加了歌曲路径这一列。

4.2 播放控制台

播放控制台是通过 MFC 的 Button 控件实现。实现了打开文件、暂停、上一首、下一首、切换播放模式、删除歌曲、清空列表的功能。

打开文件的实现，创建了一个 CFileDialog 的对象来弹出一个文件选择框，在打开文件之前，先对文件的格式进行过滤，过滤出所支持的音频文件，设置最多允许同时打开 200 个文件。再通过 while 循环来对打开的多个文件依次进行信息的获取，获取了选择的文件的路径和文件名后，通过调用自定义函数 AddMusicToList() 向列表控件中添加歌曲序号、歌曲名和歌曲路径，成功添加到播放列表过后，用户通过双击即可播放音乐。

暂停按钮的实现，通过获取按钮上面的文字，判断按钮上面的文字是暂停还是继续，是暂停则调用 mciSendCommand 函数发送 MCI_PAUSE 命令，暂停当前歌曲的播放，然后将按钮上面的文字设置成继续；是继续则调用 mciSendCommand 函数发送 MCI_RESUME 命令，继续当前歌曲的播放，然后将按钮上面的文字设置成暂停。

上一首按钮的实现，调用自定义函数 pre()。pre() 函数的实现，首先是判断播放列表中的行数是否等于 0，若等于 0，则调用 mciSendCommand 函数发送 MCI_CLOSE 命令，关闭 MCI 设备；若不为 0，则调用自定义函数 GetCurrentPos() 来获取播放列表中当前播放歌曲的位置，若当前播放歌曲的位置是在第 0 行，则将选中项即当前播放歌曲的位置倒置到最后，若不在第 0 行，则当前播放歌曲的位置减 1，设置选中行高亮，获取上一首歌曲的路径存入变量 strMusicPath 中，调用自定义函数 Stop()，关闭正在播放的音乐，然后调用自定义函数 Open(strMusicPath)，打开上一首歌曲，最后调用自定义函数 Play() 播放歌曲。

下一首按钮的实现则比上一首歌曲按钮的实现要复杂一些，因为下一首所要播放的歌曲是要根据用户选择的播放模式来进行播放的。首先获取播放模式按钮上面的文字进行判断，若是随机播放，则调用自定义函数 RandomPlay() 进行随机播放，RandomPlay() 函数的实现是先初始化随机种子，定义一个变量 randNum，然后调用自定义函数 Stop()，关闭正在播放的音乐，按照播放列表中歌曲的总数来产生随机数存入到变量 randNum 中，将变量 randNum 这一行设置选中行高亮，获取歌曲名设置到播放状态的当前播放歌曲控件中，再获取歌曲的路径存入变量 strMusicPath 中，调用自定义函数 Open(strMusicPath)，打开歌曲，然后调用自定义函数 Play() 播放歌曲；若是单曲循环，先判断选中项的歌曲和当前播放歌曲是否为同一首，不是则调用自定义函数 Next() 换歌；是同一首歌，则设置选中行即当前播放歌曲所在行高亮，获取歌曲的路径存入变量 strMusicPath 中，调用自定义函数 Stop()，关闭正在播放的音乐，再调用自定义函数 Open(strMusicPath)，打开歌曲，最后调用自定义函数 Play() 播放歌曲；若是列表循环，则直接调用自定义函数 Next()，进行播放。

切换播放模式按钮的实现，通过点击按钮调用按钮函数，将按钮的文字进行更改，从而达到设置播放模式的目的，方便在下一首歌曲按钮的函数中对播放模式进行识别。

删除歌曲按钮的实现，通过单击选择要删除的歌曲，然后点击删除歌曲的按钮进行删除。按钮函数的实现是，首先获取选中项的歌曲名和当前播放歌曲名进行比较，判断是否删除的是正在播放的歌曲，若是，则删除该行，然后调用自定义函数 Stop()，关闭正在播放的音乐，判断删除歌曲后播放列表是否为空，为空，则调用自定义函数 OnClose()，关闭定时器，调用自定义函数 InitControl()，重新初始化控件的值，不为空，判断删除的歌曲是否在第 0 行，在第 0 行则直接调用下一曲按钮的函数，进行播放，不在第 0 行，则将当前选中项的位置减 1，设置选中行高亮，再调用下一首按钮的函数，进行播放；若删除的不是正在播放的歌曲，则删除该行，删除之后，重新选中当前播放歌曲并高亮。最后在执行完删除操作之后，调用自定义函数 UpdateSerialNumber()，更新歌曲的序号。

清空列表按钮的实现，直接调用列表控件的方法 DeleteAllItems() 将播放列表清空，然后调用自定义函数 Stop()，关闭正在播放的音乐，调用自定义函数 InitControl()，重新初始化控件的值，调用自定义函数 OnClose()，关闭定时器。

4.3 播放状态

当前播放歌曲的显示是通过 MFC 的控件 Edit Control 实现，在每次调用自定义播放函数 Play() 之前，通过设置歌名控件的值，来显示正在播放的歌曲名。

当前歌词的显示是通过 MFC 的控件 Edit Control 实现，在每次调用自定义播放函数 Play() 时会调用自定义函数 GetLrc() 来获取当前歌曲的歌词，GetLrc() 函数的实现是先清空上一首的歌词，然后获取当前播放歌曲的位置，得到当前的歌曲路径，通过字符串替换获取歌词文件的路径，打开文件并一行一行的读取歌词，存入变量 strTmprLrc 中。设置显示歌词的定时器，每秒调用一次。在定时器的任务中，获取当前的播放进度，调用自定义函数 ShowCurrentLrc() 显示当前播放歌曲的歌词，ShowCurrentLrc() 函数的实现是获得当前播放的时间然后在 strTmprLrc 中进行查找，找到相同的时间就输出时间戳后面的歌词。

音量调节是通过 MFC 的控件 Slider Control 实现，在拖动滑块函数中调用自定义函数 SetVolume() 来调节音量，SetVolume() 函数的实现是定义变量 m_nvolume 获取

当前滑块的位置，然后调用 mciSendCommand 函数发送 MCI_SETAUDIO 命令设置音量。

播放进度显示是通过 MFC 的控件 Slider Control 和两个 Edit Control 控件实现，设置定时器来更新滑块的位置和当前播放时间，每秒调用一次。在默认调用定时器的函数中，调用函数 mciSendCommand 发送 MCI_STATUS 命令获得当前的播放进度，然后设置滑块的位置，调用自定义函数 UpdatePlaytime()，更新当前歌曲播放时间的文本，UpdatePlaytime() 函数的实现是调用函数 mciSendCommand 发送 MCI_STATUS 命令获得当前的播放进度存入变量 dCurrentPos 中，再调用自定义转化时间函数 TimeConversion(dCurrentPos)，将 dCurrentPos 转化为分秒的形式，设置控件的值，显示当前歌曲播放的时间。歌曲的总时间在自定义打开文件函数 Open() 中获取并设置控件的值进行显示。

拖动播放进度条是通过 MFC 的控件 Slider Control 实现，添加了处理滑块拖动的消息函数来进行处理，即 OnHScroll() 函数。OnHScroll() 函数的实现是先判断拖动的滑块是播放进度条的滑块，为的是防止在拖动音量进度条时，影响播放进度条，在拖动播放进度条时，先清空歌词控件中的歌词，再获取播放进度控件滑块的位置，然后调用 mciSendCommand 函数发送 MCI_SEEK 和 MCI_PLAY 命令跳到指定位置进行播放。

5 结束语

文中基于 MFC 所开发的音乐播放器，目前已经实现，能够在 Windows 操作平台上正常运行，实现了基本的音乐播放的操作，但是用户只能播放本地下载的音频。

【参考文献】

[1] 多媒体计算机软件开发接口 MCI 简介 [J]. 电脑爱好者，1996(04):60.