基于 MFC 的本地音乐播放器系统的设计与实现

张洋 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】在以前的学习中，并没有过多的接触过 MFC。但在这次的课程项目中我选择了利用 MFC 实现一个功能简单易于操作的本地音乐播放器，主要是利用媒体控制接口 MCI 的基本知识，具有体积小、系统资源占用少的特点。该音乐播放器有播放、暂停、上一首、下一首、随机播放、歌词显示、音量调节、进度控制、双击播放等功能，本文详细地介绍了该系统是如何实现的。

【关键词】MFC；MCI；MP3；VC++

引言

随着时代的进步，社会的发展，笔者相信每个人在业余之余都有听音乐这个爱好。听音乐不仅仅能够让我们自己放松心情，还能够让我们找到情感的共鸣。得益于互联网的快速发展，市场上已经很成熟的音乐播放软件，它们不仅仅使用方便，而且用起来感到十分得心应手。但是，它们都有一个共同的瑕疵就是代码是不开源的，我们不能根据自己意愿来设计播放器的样式和功能。该本地音乐播放器结合了用户对音乐播放器的基本需求，设计出了一款界面简洁、操作简单，可以实现播放、暂停、上一首、下一首、随机播放、歌词显示、音量调节、进度控制等功能的本地音乐播放器，因为开放源代码，有能力的用户还可以根据自己的需求来改变按钮样式和背景图。

1 相关技术简介

1.1MFC 简介

MFC 是微软的一个基础类库的简称，是微软公司实现的一个 c++ 类库，主要封装了大部分的 windows API 函数，每个 MFC 类都包括了一些函数，函数放到类中，符合 c++ 编程方法。这些函数，必须通过类定义对象才能使用 [1]。MFC 除了是一个类库以外，还是一个通用的框架，帮你封装好了一些东西，这样你就不用考虑一些底层的东西，专注于你自己代码的逻辑性就好。当然，由于是通用的框架，就有了一定的束缚，所以丧失了一定的灵活性，效率相对来说也要低一些。

1.2 MCI 简介

MCI—媒体控制接口，是由微软和 IBM 开发的一个高级应用程序接口（API），它的好处是可以方便地控制绝大多数媒体设备包括，音频、视频等多媒体设备，而不需要知道它们的内部工作状况 [2]。使用 MCI（媒体控制接口）播放音频文件，MCI 为程序员提供了两种方式访问 MCI 设备：一种是基于消息的命令接口函数；另一种是使用字符串接口函数 [3]。两者的区别在于基本命令结构和发送信息到设备的原理不同，我们在该系统中使用的就是基于消息的命令接口函数。

2 系统界面实现

该音乐播放器是基于 MFC 实现的，恰好 MFC 为我们提供了设计界面的 DIALOG，我们在设计系统界面的时候就要方便许多。首先我们在 visualstudio2017 上创建一个 MFC 对话框项目 musci_play，打开解决方案资源管理器找到 music_play.rc 文件，然后打开 music_play_DIALOG 设计我们的系统界面。首先向对话框中添加 Button 控件，并重新命名其 ID，该控件主要是实现打开文件、播放、暂停、随机播放等功能；其次我们向对话框中添加 list control 控件，并重新命名其 ID，该控件主要是为了实现显示歌曲列表的功能；再然后向对话框中添加 static text 控件、slider control 控件、edit control 控件，并重新命名它们的 ID，这些控件主要是为了实现音量控件、进度控制、歌词显示等功能。我们可以根据自己的想法，随意摆放这些控件的位置，做一个自己喜欢的系统界面出来。

3 功能实现

3.1 打开音乐文件

音乐文件打开功能是该音乐播放系统的重中之重，灵魂所在。我们首先在 music_play_DIALOG 中双击打开按钮，进入到 music_playDlg.cpp 中编写有关打开函数的代码。我们先通过 CFileDialog 类打开一个新的文

件窗口，我们可以指定打开的文件类型，在这儿我们就是要找到在本地的 mp3 文件。可以通过一个 while 循环将本地文件夹中的多首歌曲一次性显示到列表中，在 while 循环中调用预先写好的添加歌曲到列表的函数。通过 dlg.GetNextPathName(pose) 函数获取选择播放音乐文件的地址，然后通过 mciSendCommand(NULL, MCI_OPEN, MCI_OPEN_ELEMENT | MCI_WAIT, (DWORD)&mciopenParams) 函数发送消息打开该音乐文件。MCI_WAIT 的意思是通知 MCI 设备，MCI 命令执行完后才能将控制权还给应用程序。当用 MCI_OPEN 命令消息打开一个设备时，将自动创建一个设备标识符。如果 mciSendCommand 调用成功，则有返回值；否则表示设备驱动出错，可以用 MCIERROR 来获取出错信息。打开音乐成功后，就将该 id 号赋给先定义好的 m_DeviceID。

3.2 播放、暂停功能

当我们成功获取设备 ID 后，就可以在 music_paly_DIALOG 中双击播放按钮，然后编写有关播放函数的代码。我们可以通过 mciSendCommand(m_DeviceID, MCI_PLAY, MCI_FROM | MCI_NOTIFY, (DWORD)&mciplayparms) 来播放我们选中的歌曲，MCI_PLAY 表示开始播放数据，MCI_NOTIFY 是表示通知 MCI 设备立刻将控制权交给应用程序，但当命令执行完成之后，向应用程序发送 MM_MCINOTIFY 消息。当我们想要暂停播放时，就可以使用 mciSendCommand(m_DeviceID, MCI_PAUSE, 0, 0) 发送消息暂停播放。又想重新播放，就向设备发送 mciSendCommand(m_DeviceID, MCI_RESUME, 0, 0)。在这儿有个地方需要注意，每次播放新歌曲之前要发送消息 mciSendCommand(m_DeviceID, MCI_CLOSE, 0, 0)，清除上次设置。

3.3 上一首、下一首、随机播放功能

我们先获取当前播放歌曲在列表中的行号，然后获取当前行，上一行的内容保存在先定义好的 str 中，然后发送消息 mciSendCommand(NULL, MCI_OPEN, MCI_OPEN_ELEMENT | MCI_WAIT, (DWORD)&mciopenParams) 打开该文件，调用播放函数播放歌曲；下一首同理，获取当前行，下一行的内容保存在 str 中；随机播放要先定义一个随机数种子，产生一个随机数，获取该行的内容保存到 str 中，然后 mciSendcommand 发送消息，调用播放函数播放音乐。在这儿我们要注意的是，如果该音乐已经是第一首歌或者最后一首歌就提示用户，而且随机数的大小也应该控制在当前列表总行数的范围内。

3.4 双击播放功能

在 music_play_DIALOG 中找到 list control 控件，右击选择添加类向导，为控件添加 NM_DBLCLK 消息。然后通过 POSITION pos = list.GetFirstSelectedItemPosition() 获取当前行号，将当前行号的内容保存到 str 中。然后 mciSendcommand 发送消息，调用播放函数播放音乐。

3.5 音量控制功能

在初始化函数里面通过 m_slider.SetRange(0, 1000) 设置音量范围大小，通过 m_slider.SetPos(1000) 设置它的初始值位置。然后通过 m_slider.GetPos() 获取滑动控件的位置既是音量的大小，通过 mciSendCommand(m_DeviceID, MCI_SETAUDIO, MCI_DGV_SETAUDIO_VALUE | MCI_DGV_SETAUDIO_ITEM, DWORD(&msisetvolume)) 发送消息改变音量的大小。

3.6 进度控制功能

进度控制功能主要体现在两个方面，一是可以实时显示当前播放的时间，二是 slider 控件 2 可以跟着歌曲播放走动，也可以通过它改变歌曲播放的进度。进度控制需要用到定时器，所以我们应该先创建一个 ontimer 函数。获取当前歌曲播放的总毫秒数主要是通过 mciSendCommand(m_DeviceID, MCI_STATUS, MCI_STATUS_ITEM, (DWORD)(LPVOID)&StatusParms)，然后 NOW = StatusParms.dwReturn 获取总毫秒数，将总毫秒数转换为分秒的形式显示。然后通过强制转换将 DWORD 类型转换为整型，m_step.SetPos(int(NOW)) 改变当前滑块的位置，最后在播放函数里面调用 SetTimer(2, 1000, NULL) 进度条就可以跟着歌曲的播放而动起来了。获取当前歌曲的总长度是通过 mciSendCommand(m_DeviceID, MCI_STATUS, MCI_STATUS_ITEM, (DWORD)(LPVOID)&StatusParms),StatusParms.dwItem= MCI_STATUS_LENGTH，LENGTH = StatusParms.dwReturn 而实现的。当然要想通过 slider 控件改变播放进度这还是不行的，我们还需要添加 OnHScroll 函数。在函数体内首先判断是不是点击了 slider 控件，如果是就获取当前点击的位置，将 DWORD 类型强制转换为整型，然后通过 mciSendCommand(m_DeviceID, MCI_PLAY, MCI_FROM | MCI_NOTIFY, (DWORD)&mciplayparms) 发送消息从当前位置开始播放。

3.7 歌词显示功能

歌词显示功能是所有功能中最复杂的一个，也是最难实现的一个。首先我们需要下载一个歌词文件，然后通过分析发现歌词文件是很有规律的。我们创建了一个结构体数据用来存放歌词的数据，其次对文件的操作需要加入头文件 #include<fstream>。我们通过 getline(fin, str) 一行一行的读取数据，然后分析一行就把数据保存到结构体数组中，直到数据全部读取完毕。我们在定时器中写歌词输出，我们需要获取当前歌曲的总毫秒数，

如果结构体数组里面的开始时间不为空就转换为毫秒数，与当前毫秒数进行比较，如果相等就输出当前行的歌词内容。在这儿值得注意的是，返回的总毫秒数不是那么地精确，所以需要设定的是一个范围。最后我们在播放函数里面调用 SetTimer(3, 1000, NULL) 就可以显示歌词了。

3.8 设置背景图和 Button 按钮重绘

我们需要先将自己喜欢的背景图片放入到资源文件中，然后在资源视图里面将该图片添加为位图。接下来在 OnPaint() 函数里编写代码，首先通过 bmp.LoadBitmap(IDB_BITMAP3) 加载背景图，然后 bmp.GetBitmap(&LOGBMP) 获取图片的宽和高，memDC.CreateCompatibleDC(&dc) 创建内存 DC，最后通过 dc.StretchBlt(0, 30, rect.Width(), rect.Height(), &memDC, 0, 0, LOGBMP.bmWidth, LOGBMP.bmHeight, SRCCOPY) 语句进行贴图，背景图就设置成功了。在这儿要注意的是，可能背景图的大小比不上界面的大小，就需要通过 dc.SetStretchBltMode(HALFTONE) 语句设置为清晰的拉伸。Button 按钮的重绘需要重新添加一个 CButtonUI 类，继承于 CButton 类。然后在 CButtonUI 类中按照设置背景图的方法，对 Button 按钮进行重绘，都设置为我们喜欢的图片。

4 结束语

通过自己独立实现了本地音乐播放器系统，增加了我对编程的浓厚兴趣。虽然这只是一个很小的项目，但是他对我了解整个开发流程起到了一个很好的学习作用。通过这个项目，我对 MFC 和 MCI 的了解又进了一步，虽然该系统不是很完善，但其基本功能都实现了，我相信按照我的步骤与思路，一定可以设计出一个属于自己的本地音乐播放器。

【参考文献】

[1] 王育坚 . Visual C++ 面向对象编程教程 [M]. 北京 : 清华大学出版社 , 2003.
[2] 侯俊杰 . 深入浅出 MFC. 第 2 版 [M]. 武汉 : 华中科技大学出版社 , 2001.
[3] 钟宁 . 利用 MCI 实现多媒体程序设计 [J]. 沿海企业与科技 , 2001(004):30-31.