

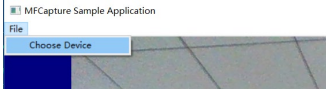
1. 阅读思路:

- a) 通过设定断点进行 debug 的方式确定哪些函数是在运行过程中被调用的、哪些函数是直接和程序有关需要修改的, 并且通过逐句调试的方式确定程序运行的流程, 如果函数跳变到了非项目文件中 (如微软已经给出的函数) 则默认这个函数的运行机理不需要掌握。
- b) 通过注释以及函数和变量的英文名字确定函数以及变量的意义以及功能, 若遇到了不明白的地方便上网搜索。
- c) 通过修改部分参数的方式弄明白某项变量的意义, 进而弄明白其所在函数的功能。

2. 原项目框架下重要函数的具体功能解析

- a) winmain.cpp 源文件的主要功能是控制图像窗口以及事件监听, 其中重要的函数有以下几个

- i. **OnCommand()**: 这个函数用来监听菜单栏命令。窗口的上方有一个菜单栏, 里面分为多个选项, 如“file”中“choose device”, 这

个选项实际上是一个按钮 , 它有独属于自己的唯一的一个属性“id”, 如在这个程序中该按钮的 id 为

ID_FILE_CHOOSSEDEVICE, 我们在 OnCommand 函数中通过 switch 监听这个按钮的 id 看这个按钮是否被点击, 然后再执行接下来的操作, 如在原始函数中的逻辑是如果 choose device 按钮被按下则执行 **OnChooseDevice(hwnd, TRUE);**。

- ii. **WindowProc()**: 这是一个全局事件监听函数, 其监听的对象包括程序运行中出现错误的时候的消息、设备更换时的消息以及键盘按下的消息, 如上一个函数一样, 每一个消息都有自己对应的“id”, 也是可以直接通过 switch 函数监听消息并通过接下来的操作完成回调。

- b) device.cpp 源文件主要用于从设备处获取图像信息, 将其格式转化成 RGB32 格式, 并将其呈现在指定的地方

- i. **DrawDevice::DrawFrame()**: 这个函数是 DrawDevice 类下的一个函数, 其功能是从设备中获取数据的指针, 并把数据通过微软定制的 m_convertFn 函数把视频变成 RGB32 并输出。通过查阅资料, 我们搞清楚了在 m_convertFn 的参数中指向图像数据的指针为

(BYTE*)lr.pBits, 可以看出它是一个指向 BYTE 型数据的指针, 而它指向的数据结构是这样的:

1. **BYTE1**: rgb[1].Blue
2. **BYTE2**: rgb[1].Green
3. **BYTE3**: rgb[1].Red
4. **BYTE4**: rgb[1].Reserved
5. **BYTE5**: rgb[2].Blue
6. **BYTE6**: rgb[2].Green

7. **BYTE7:**rgb[2].Red
8. **BYTE8:**rgb[2].Reserved
9. ...

因此是每四个字节保存一位像素的所有信息，包括蓝色、绿色、红色以及预留位置。