Jetsons PlayBack

Overview :

There are four main classes for each job that is required for multi-threading song player. WaveFile Reader is responsible for loading wave file to buffer. WaveControler is the one that load data that is in the WaveFileReader and send to the twenty small buffers to play. WavePlayer tells which buffer to play and waveBuffer class send data to driver so it gives us sound.

WaveFileReader:

This class read wave data from file , set the flag so it does not load another file and lose previous one that is not copied to controllers buffer. After Controller copied all data from buffer in this class it set flag again and WaveFileReader load another file in to its buffer. This class is also sleep for 20ms after it checked the flag. There are four main functions.This class not wait for controller class signal to read. It always read before Contoler class request to load.

Run function is the one that is running on the thread.

ReadWaveFile is responsible for reading wave file. It is only read 22 wavefile its hardcoded after all files are copied to controller it sets file info as finished so it breaks to loop. After data is in the buffer it set isDataReady to true so it does not read again and lose it until buffer data copied to controller Buffer.

SwitchStatus is the one that changing condition of the Boolean isDataReady variable. Is Data ready protected by mutex .This function used by Controller and File Reader classes. One is or blocking another read other one is after all data copied to controller , it makes flag to false so it load another file.

IsAlive function check if this classes job is done . If all 22 file copied to controller then it returns false and run function break from while loop. It is dependent on the file info status variable. This variable set to More to Read and in constructer and its set to finished in the ReadWaveFile Function.

WaveFileControler:

This class has two responsibilities. One is load data into its Passive buffer and send Data to twenty 2K small buffers from its Active Buffer. These buffers are Object they contain information about file size, remaining size and 512K buffer that data loaded in. There is five main function in this class.

Run function is the one that is running on the thread.

First load is the load two buffers before sending in to buffers.

LoadBuffer copy all data in the FileReader buffer to Passive buffer.

SendData responsible for sending data to twenty 2K buffers. This function work until all data in the Active buffer copied over. It check queue is empty or not because all request to send data over small buffers are in the queue. If its empty that means there no request so no need to send data . Because its in while loop it wait until queue contains request. Queue contains address of small buffer object so using this information it calls load Buffer method of WaveBuffer class and send pointer of current active buffer and size of how much to read. Then it update remaining and pointer to where next data that is going to send. There is a possibility that remaining data in Active buffer smaller than 2K , if this is the case size of to read equal to remaining otherwise it lead some unwanted sound.

SwitchBuffer is just changing Active and Passive Buffer. After all data in the ActiveBuffer depleted other buffer that is ready replace role of Active buffer and the one finished copied all data wait for the fill again.

WavePlayer:

This class contains of LinkList of WaveBuffer object and its responsible for which block going to play. Working order in while loop play WaveBuffer ,wait for callback , after getting signal from callback go to next one to play and continue until song finishes. For this synchronization this class contains condition variable. This condition variable sleeps the thread until signal come from waveOutWrites callback.

Before start playing Play function send request to load all small buffers. Then it waits until all of them loaded with data. It calls play in the waveBuffers before going to loop so Waveout header , queue what is going to play . Aim of this get rid of clicks while song playing. This class uses another queue named busy. This is intended for synchronize order of pushing data to request queue. Because when I call play method of buffer object and not wait until signal , order of pushing data to request queue changes .Which buffer finishes first call callback and send request. This makes order not as intended. The reason of this problem is one of

the parameter of Callback function is this WavePlayer object. And I push current wavebuffer to request queue. Before callback happens usually current changes (it go to next one) and it may push not the address of which buffer finishes but another . It may lead to deadlock. Play three or four times needed to get rid of clicks so this can of approach is needed for my design. Lastly before exiting thread it waits all the thread of WaveBuffer released.

WaveBuffer :

This class plays sound. WaveOutWrite function that send data to driver call in this class. This class have four main method.

Run function is the one that is running on the thread. This function wait for signal and after that it call WaveOutPlay function to send data to driver. Condition variable needed because otherwise it always try to play sound even buffer is empty and this leads playing buffers in out of order.

Play method used by WavePlayer to signal condition variable in Run function to play sound.

WaveOutPlay is the method prepares WaveHeader , Writes data to driver using WaveOutWrite and waits until playing is finishes.

SendRequest sends address of WaveBufferObject to request Queue to refill. This function using in Callback.

LoadBuffer using by Controller to load WaveBuffer object.