# Adding Music to Your Program with PLAYMIDI

The PLAYMIDI command is a bit more complicated than the PLAYWAVE command. Unlike the PLAYWAVE command, the programmer using PLAYMIDI must keep track of the playmidi progress.
(1) Whether a midi file is currently playing
(2) Whether the midi file has finished playing
To keep track, you must assign variables. Such variables are commonly referred to as flags. The programmer must also determine the length of the midi file. Fortunately, this value is extracted by Just BASIC as part of the PLAYMIDI command. The directions from the Help File

```
Description:
This plays a *.MIDI sound from a file on disk as specified in filename
.
The length variable will hold the length of the MIDI file (not in seco
nds).
You can only play one file at a time. Periodically, you will need to u
se
the MIDIPOS( ) function to see if you've reached the end of the music:

Finally, use the STOPMIDI command to close the music file before you c
an
play a different one.

Usage:

'the playmidi command returns the length of the midi in
' the variable howLong
playmidi "c:\somedir\mymusic.midi", howLong
timer 1000, [checkPlay]
wait

[checkPlay]
if howLong = midipos( ) then [musicEnded]
wait

[musicEnded]
stopmidi
timer 0
wait
```

Liberty BASIC includes the midi file theme.mid in its MEDIA subfolder. This article assumes you are running the demo codes from your Liberty BASIC directory.

First, the code to begin playing the midi

```
Playmidi DefaultDir$;"\MEDIA\theme.mid", midiLength
Print "It is important to press Enter BEFORE the music stops."
Input "Press Enter at any time to stop the midi: ";any$
Stopmidi
End
```

Unfortunately, there is no automatic way of knowing whether the midi file has finished playing. And, unlike the PLAYWAVE "" command, if a STOPMIDI command is issued when no midi was playing, your program will crash. Run this Code

```
Playwave ""
Print "This works fine even if no wav file was being played."
Print "But…."
STOPMIDI
End
```

A flag variable is required to keep track of whether a midi file is being played or not. In this demo, we'll use midiFlag as that flag variable.

```
midiFlag = 0 'No midi file being played
midiFlag = 1 'yes a midi file is being played
```

But, how do we keep track of whether the midi file is finished or not? The PLAYMIDI command gets the length of that midi file and stores it in the variable you assign. In this demo, that variable is midiLength.

```
Playmidi DefaultDir$;"\MEDIA\theme.mid", midiLength
Print "midiLength = ";midiLength
Stopmidi
End
```

If you're using the theme.mid file, you might find that midiLength = 627. Knowing the full length of the file is only useful if the current position of the midi file is known. That position is made known by the command MIDIPOS(). Run this code to see MIDIPOS() in action. Note also the use of the midiFlag variable to indicate whether a midi file is being played or not.

```
Playmidi DefaultDir$;"\MEDIA\theme.mid", midiLength
```

```
midiFlag = 1
For i = 1 to 300
     Print “midiLength = “;midiLength
     Print “midiPosition = “;Midipos()
Next i
Stopmidi
midiFlag = 0
End
```

The above code isn't very practical for use within a program. It would be better to check the position of the midi file being played and still continue on with the program. In this case, set up a timer. A timer is kind of an alarm clock that’s set not for a particular time, but for a particular time span. Everytime that time span has been reached, the timer executes the assigned block of code, then sits off to the side and waits for that time span to be reached again. In this next example, a timer will be used to check the position of the midi file every 1000 milliseconds (1 second). While that’s happening, the program goes on as usual.

```
Playmidi DefaultDir$;”\MEDIA\theme.mid”, midiLength
midiFlag = 1

Timer 1000, [checkPlay] ‘Every 1000 milliseconds, check the midi posit
ion

[doSomethingHere]
If midiFlag = 1 Then
     msg$ = “Do you want to stop the music yet?”
Else
     msg$ = “Press Enter to Quit”
End If
If midiFlag = 0 Then
     Confirm msg$;any$ ‘Confirm is used here only to prevent the BEEP
from Notice
     End
End If
Confirm msg$;yn$
If midiFlag = 1 Then
     If yn$ = “yes” Then
          Timer 0 ‘Turn off the timer
          Stopmidi ‘Stop the midi file
          midiFlag = 0 ‘Set the midiFlag to 0 to indicate midi file is
 no longer being played
          End If
     End If
Goto [doSomethingHere]
```

```
[checkPlay]
If Midipos() = midiLength Then ‘If the position is at the total length
 then
     Stopmidi ‘Stop playing the midi
     midiFlag = 0 ‘Set the midiFlag to 0 to indicate midi file is no l
onger being played
     Timer 0 ‘Stop the timer
End If
Wait
```

One last word of caution. When you end your program, don’t expect the midi that’s playing to be stopped automatically (as is the case with PLAYWAVE). If your midi file hasn’t reached the end of the file and you haven’t issued a STOPMIDI command, your music will continue even after the program has ENDed. Running the same or another program will encounter the Midi File already playing error message when you try to issue the PLAYMIDI command again. This will happen even if the music has finished playing and you no longer hear the music. So, you must use STOPMIDI before ENDING your program, but ONLY IF the STOPMIDI command wasn’t issued prior. This is another place the midiFlag variable becomes cruicial. It is always prudent to issue a Timer 0 command at the end of any program using a timer, whether the program has already shut the timer off or not.

```
[quit]
Timer 0
If midiFlag = 1 Then
     Stopmidi
End If
Close #main
End
```

## Playmidi Demo

```
'Define the Menus
    Menu #w, "&File","&Load Midi",[loadMusic], _
        "&Play Midi",[playMusic],"&Stop Midi",[stopMusic],|, _
        "E&xit",[endProgram]

'Open the Program's Main Window
    WindowWidth = 320 'Sets Width of Main Window
    WindowHeight = 220 'Sets Height of Main Window
    UpperLeftX = Int((DisplayWidth-WindowWidth)/2)
'Centers Horizontally
    UpperLeftY = Int((DisplayHeight-WindowHeight)/2)
'Centers Vertically

    Nomainwin
    Graphicbox #w.g, 5, 15, 300, 98
    Button #w.b1, "LOAD", [loadMusic], UL, 35, 130, 60, 30
    Button #w.b2, "PLAY", [playMusic], UL, 127,130, 60, 30
    Button #w.b3, "STOP",[stopMusic], UL, 216, 130, 60, 30

    fileFlag = 0 '0 = No File Loaded, 1 = File Loaded
    midiFlag = 0 '0 = No Midi Playing, 1 = Midi Playing
    shortName$ = "No Midi Loaded" 'Name of Current Loaded File
    midiLength = 0 'Midi Length of Current Loaded File

    Open "Liberty BASIC Midi Player" for Window as #w
    #w "Trapclose [endProgram]"
    #w "Font Times_New_Roman 12 Bold Italic"

    #w.g "Down; Fill 0 211 155"
    #w.g "BackColor 0 211 155"
    #w.g "Font Times_New_Roman 12 Bold"
    Call grayBorder 298, 96
    #w.g "Flush"
    Call displayMidiLength shortName$, midiLength, midiFlag
    Wait

[endProgram]
    If midiFlag = 1 Then
        StopMidi
    End If
    Timer 0
    Close #w
    End

[loadMusic]
'File Dialog to Load New File
    FileDialog "Open Midi File", "*.mid", fileName$

'If No File Selected, Then Go Back to [awaitUserInput]
    If fileName$ = "" Then
        Call displayMidiLength shortName$, midiLength, midiFlag
        Wait
    End If

'Get Extension Type
    fileExtension$ = fileExtension$(fileName$)
    shortName$ = shortName$(fileName$)

'Is File a Valid Midi File?  valid extensions = .mid, .midi
```

```
    If validExt(fileExtension$) = 0 Then
        msg$ = msgInvalidFile$(shortName$)
        Notice msg$
    End If
    If validExt(fileExtension$) = 0 Then
        Call displayMidiLength shortName$, midiLength, midiFlag
        Wait
    End If

'If Valid File, then Check to See if Midi Already Playing
'End Midi if Playing
    Timer 0
    If midiFlag = 1 Then
        StopMidi
        midiFlag = 0
    End If
'If Valid File, Display Name of New File
    fileFlag = 1 'Midi File has been Loaded
    PlayMidi, fileName$, midiLength
    StopMidi
    Call displayMidiLength shortName$, midiLength, midiFlag
    Wait

[playMusic]
    If midiFlag = 1 Then
        Timer 0
        StopMidi
        midiFlag = 0
    End If
    If fileFlag = 0 Then
        msg$ = msgNoMidiLoaded$()
        Notice msg$
    Else
        Playmidi fileName$, midiLength
        midiFlag = 1
        Timer 500, [checkPlay]
    End If
    Call displayMidiLength shortName$, midiLength, midiFlag
    Wait

[stopMusic]
    If midiFlag = 0 Then
        msg$ = msgNoMidiPlaying$()
        Notice msg$
    Else
        Timer 0
        StopMidi
        midiFlag = 0
    End If
    Call displayMidiLength shortName$, midiLength, midiFlag
    Wait

[checkPlay]
    If midiFlag = 1 Then
        If midiLength = MidiPos() Then
            StopMidi
            Timer 0
            midiFlag = 0
        End If
    End If
    Call displayMidiLength shortName$, midiLength, midiFlag
Wait

Function placeZero$(n, z)
    placeZero$ = Str$(n)
    If Len(placeZero$) = 0 Then
        placeZero$ = "0"
    End If
    z = Abs(z)
    If z < 0 Then
        z = 0
    End If
    If Len(placeZero$) > 8 Then
        placeZero$ = Right$(placeZero$,8)
    End If
    While Len(placeZero$) <> z
        placeZero$ = "0";placeZero$
    Wend
End Function

Function validExt(fileExtension$)
    validExt = 0
    fileExtension$ = UPPER$(fileExtension$)
    If fileExtension$ = "MID" Then
        validExt = 1
    End If
    If fileExtension$ = "MIDI" Then
        validExt = 1
    End If
End Function

Function shortName$(fileName$)
```

```
    c$ = ""
    c = Len(fileName$)
    While c$ <> "\"
        c = c - 1
        c$ = Mid$(fileName$, c, 1)
    Wend
    shortName$ = Right$(fileName$, Len(fileName$)-c)
End Function

Function fileExtension$(fileName$)
    c$ = ""
    c = Len(fileName$)
    While c$ <> "."
        c = c - 1
        c$ = Mid$(fileName$, c, 1)
    Wend
    fileExtension$ = Right$(fileName$, Len(fileName$)-c)
End Function

Sub displayMidiLength shortName$, midiLength, midiFlag
    midiLength$ = placeZero$(midiLength, 8)
    If midiFlag = 1 Then
        musicPosition$ = placeZero$(MidiPos(), 8)
    Else
        musicPosition$ = placeZero$(0, 8)
    End If
    nSpaces = 32 - Len(shortName$)
    If nSpaces < 0 Then
        nSpaces = 0
    End If
    #w.g "Color DarkPink"
    #w.g "Place 100 30"
    #w.g "\";shortName$;Space$(nSpaces)
    #w.g "Place 100 54"
    #w.g "\";midiLength$;Space$(4)
    #w.g "Place 100 78"
    #w.g "\";musicPosition$;Space$(4)
    #w.g "Discard"
End Sub

Sub grayBorder w, h
    hue = 0: hueInc = 32: start = 0
    For box = 0 to 10
        #w.g "Color ";hue;" ";hue;" ";hue
        #w.g "Place ";start;" ";start
        #w.g "Box ";start + w;" ";start + h
        start = start + 1
        w = w - 2: h = h - 2
        If hue = 160 Then
            hueInc = (-32)
        End If
        hue = hue + hueInc
    Next box
    #w.g "Color DarkCyan"
    #w.g "Place 40 30"
    #w.g "\Midi:"
    #w.g "Place 40 54"
    #w.g "\Length:"
    #w.g "Place 40 78"
    #w.g "\Position:"
    #w.g "Flush; Discard"
    #w.g "Getbmp midiBox 0 0 w h"
    #w.g "Drawbmp midiBox 0 0"
    #w.g "Flush; Discard"
End Sub

Function msgInvalidFile$(shortName$)
    msgInvalidFile$ = "No Midi File Loaded . . . ";Chr$(13)
    msgInvalidFile$ = msgInvalidFile$;shortName$;Chr$(13)
    msgInvalidFile$ = msgInvalidFile$;
"does not appear to be a";Chr$(13)
    msgInvalidFile$ = msgInvalidFile$;"valid midi file."
End Function

Function msgNoMidiLoaded$()
    msgNoMidiLoaded$ = "No Midi File To Play . . . ";Chr$(13)
    msgNoMidiLoaded$ = msgNoMidiLoaded$;
"There does not appear to be";Chr$(13)
    msgNoMidiLoaded$ = msgNoMidiLoaded$;"a valid midi file loaded."
End Function

Function msgNoMidiPlaying$()
    msgNoMidiPlaying$ = "No Midi File To Stop . . . ";Chr$(13)
    msgNoMidiPlaying$ = msgNoMidiPlaying$;
"There does not appear to be";Chr$(13)
    msgNoMidiPlaying$ = msgNoMidiPlaying$;"a valid midi file playing."
End Function
```

By the way, although you can’t play multiple wav files or multiple midi files simultaneously, you can PLAYWAVE a wav file even while a midi file is playing. In this way, you don’t have to interrupt background music to hear those blood curdling sound effects.