API Collection

Music Player

Create and play a sequence of tracks, and manage aspects of playback in response to standard events.

Topics

Managing a Music Player

`func NewMusicPlayer(UnsafeMutablePointer<MusicPlayer?>) -> OSStatus`

Creates a new music player.

`func DisposeMusicPlayer(MusicPlayer) -> OSStatus`

Disposes of a music player.

`func MusicPlayerGetBeatsForHostTime(MusicPlayer, UInt64, UnsafeMutablePointer<MusicTimeStamp>) -> OSStatus`

Gets the beat number associated a specified host time.

`func MusicPlayerGetHostTimeForBeats(MusicPlayer, MusicTimeStamp, UnsafeMutablePointer<UInt64>) -> OSStatus`

Gets the host time associated with a specified beat.

`func MusicPlayerGetPlayRateScalar(MusicPlayer, UnsafeMutablePointer<Float64>) -> OSStatus`

Gets the playback rate multiplier for a music player.

`func MusicPlayerGetSequence(MusicPlayer, UnsafeMutablePointer<MusicSequence?>) -> OSStatus`

Gets the music sequence associated with a music player.

`func MusicPlayerGetTime(MusicPlayer, UnsafeMutablePointer<MusicTimeStamp>) -> OSStatus`

Gets the playback point for a music player, in beats.

`func MusicPlayerIsPlaying(MusicPlayer, UnsafeMutablePointer<DarwinBoolean>) -> OSStatus`

Indicates whether or not a music player is playing.

`func MusicPlayerPreroll(MusicPlayer) -> OSStatus`

Prepares a music player to play.

`func MusicPlayerSetPlayRateScalar(MusicPlayer, Float64) -> OSStatus`

Sets a playback rate multiplier for a music player.

`func MusicPlayerSetSequence(MusicPlayer, MusicSequence?) -> OSStatus`

Sets the music sequence for the music player to play.

`func MusicPlayerSetTime(MusicPlayer, MusicTimeStamp) -> OSStatus`

Sets the playback point for a music player, in beats.

`func MusicPlayerStart(MusicPlayer) -> OSStatus`

Starts playback of a music player.

`func MusicPlayerStop(MusicPlayer) -> OSStatus`

Stops playback of a music player.

`typealias MusicPlayer`

A music player plays a music sequence (of type `MusicSequence`).

`typealias MusicTimeStamp`

A timestamp for use by a music sequence.

`var kMusicTimeStamp_EndOfTrack: Double`

Indicates a time immediately beyond the last music event in a music track. Use this value when selecting all music events starting at a designated time and extending to, and including, the last event in a track. Also use this value to position an iterator for moving backward through a track, from the end to the start. See also `NewMusicEventIterator(_:_:)` and `MusicEventIteratorSeek(_:_:)`.

Iterating Over Music Events

```
func NewMusicEventIterator(MusicTrack, UnsafeMutablePointer<MusicEvent
Iterator?>) -> OSStatus
```

Creates a new music event iterator.

```
func DisposeMusicEventIterator(MusicEventIterator) -> OSStatus
```

Disposes of a music event iterator.

```
func MusicEventIteratorNextEvent(MusicEventIterator) -> OSStatus
```

Positions a music event iterator at the next event on a music track.

```
func MusicEventIteratorSeek(MusicEventIterator, MusicTimeStamp) ->
OSStatus
```

Positions a music event iterator at a specified timestamp, in beats.

```
func MusicEventIteratorDeleteEvent(MusicEventIterator) -> OSStatus
```

Deletes the event at a music event iterator’s current position.

```
func MusicEventIteratorGetEventInfo(MusicEventIterator, UnsafeMutable
Pointer<MusicTimeStamp>, UnsafeMutablePointer<MusicEventType>, Unsafe
MutablePointer<UnsafeRawPointer?>, UnsafeMutablePointer<UInt32>) ->
OSStatus
```

Gets information about the event at a music event iterator’s current position.

```
func MusicEventIteratorHasCurrentEvent(MusicEventIterator, Unsafe
MutablePointer<DarwinBoolean>) -> OSStatus
```

Indicates whether or not a music track contains an event at the music event iterator’s current position.

```
func MusicEventIteratorHasNextEvent(MusicEventIterator, UnsafeMutable
Pointer<DarwinBoolean>) -> OSStatus
```

Indicates whether or not a music track contains an event beyond the music event iterator’s current position.

```
func MusicEventIteratorHasPreviousEvent(MusicEventIterator, Unsafe
MutablePointer<DarwinBoolean>) -> OSStatus
```

Indicates whether or not a music track contains an event before the music event iterator’s current position.

```
func MusicEventIteratorPreviousEvent(MusicEventIterator) -> OSStatus
```

Positions a music event iterator at the previous event on a music track.

`func MusicEventIteratorSetEventInfo(MusicEventIterator, MusicEventType, UnsafeRawPointer) -> OSStatus`

Sets information for the event at a music event iterator’s current position.

`func MusicEventIteratorSetEventTime(MusicEventIterator, MusicTimeStamp) -> OSStatus`

Sets the timestamp for the event at a music event iterator’s current position.

`typealias MusicEventIterator`

A music event iterator sequentially handles events on a music track.

`typealias MusicEventType`

MIDI and other music event types, used by music event iterator functions.

`struct ExtendedNoteOnEvent`

Describes a note-on event with extended parameters.

`struct ExtendedTempoEvent`

Describes a music track tempo in beats-per-minute.

`struct MusicEventUserData`

Describes a user-defined event.

`struct ParameterEvent`

Describes an audio unit parameter automation event.

`struct MusicDeviceNoteParams`

`struct MusicDeviceStdNoteParams`

`struct NoteParamsControlValue`

Managing Music Sequences

`func NewMusicSequence(UnsafeMutablePointer<MusicSequence?>) -> OSStatus`

Creates a new empty music sequence.

`func DisposeMusicSequence(MusicSequence) -> OSStatus`

Disposes of a music sequence.

`func MusicSequenceBarBeatTimeToBeats(MusicSequence, UnsafePointer<CABarBeatTime>, UnsafeMutablePointer<MusicTimeStamp>) -> OSStatus`

Formats a music sequence’s bar-beat time to its beat time.

```
func MusicSequenceBeatsToBarBeatTime(MusicSequence, MusicTimeStamp,
UInt32, UnsafeMutablePointer<CABarBeatTime>) -> OSStatus
```

Formats a music sequence's beat time to its bar-beat time.

```
func MusicSequenceDisposeTrack(MusicSequence, MusicTrack) -> OSStatus
```

Removes a music track from a music sequence, and disposes of the track.

```
func MusicSequenceFileCreate(MusicSequence, CFURL, MusicSequenceFile
TypeID, MusicSequenceFileFlags, Int16) -> OSStatus
```

Creates a MIDI file from the events in a music sequence.

```
func MusicSequenceFileCreateData(MusicSequence, MusicSequenceFileTypeID
, MusicSequenceFileFlags, Int16, UnsafeMutablePointer<Unmanaged<CFData
>?>) -> OSStatus
```

Creates a data object containing the events from a music sequence.

```
func MusicSequenceFileLoad(MusicSequence, CFURL, MusicSequenceFileType
ID, MusicSequenceLoadFlags) -> OSStatus
```

Loads data into a music sequence from a URL reference.

```
func MusicSequenceFileLoadData(MusicSequence, CFData, MusicSequenceFile
TypeID, MusicSequenceLoadFlags) -> OSStatus
```

Load data into a music sequence from a data reference.

```
func MusicSequenceGetAUGraph(MusicSequence, UnsafeMutablePointer<
AUGraph?>) -> OSStatus
```

Gets the audio processing graph associated with a music sequence.

```
func MusicSequenceGetBeatsForSeconds(MusicSequence, Float64, Unsafe
MutablePointer<MusicTimeStamp>) -> OSStatus
```

Calculates the number of beats that correspond to a number of seconds.

```
func MusicSequenceGetIndTrack(MusicSequence, UInt32, UnsafeMutable
Pointer<MusicTrack?>) -> OSStatus
```

Gets the music track at the specified track index.

```
func MusicSequenceGetInfoDictionary(MusicSequence) -> CFDictionary
```

Returns a dictionary containing music sequence information.

```
func MusicSequenceGetSMPTEResolution(Int16, UnsafeMutablePointer<Int8>,
UnsafeMutablePointer<UInt8>)
```

```
func MusicSequenceGetSecondsForBeats(MusicSequence, MusicTimeStamp,
UnsafeMutablePointer<Float64>) -> OSStatus
```

Calculates the number of seconds that correspond to a number of beats.

```
func MusicSequenceGetSequenceType(MusicSequence, UnsafeMutablePointer<
MusicSequenceType>) -> OSStatus
```

Gets the sequence type for a music sequence.

```
func MusicSequenceGetTempoTrack(MusicSequence, UnsafeMutablePointer<
MusicTrack?>) -> OSStatus
```

Gets the tempo track for a music sequence.

```
func MusicSequenceGetTrackCount(MusicSequence, UnsafeMutablePointer<
UInt32>) -> OSStatus
```

Gets the number of music tracks owned by a music sequence.

```
func MusicSequenceGetTrackIndex(MusicSequence, MusicTrack, Unsafe
MutablePointer<UInt32>) -> OSStatus
```

Gets the index number for a specified music track.

```
func MusicSequenceNewTrack(MusicSequence, UnsafeMutablePointer<Music
Track?>) -> OSStatus
```

Add a new, empty music track to a music sequence.

```
func MusicSequenceReverse(MusicSequence) -> OSStatus
```

Reverses the MIDI and tempo events in a music sequence, so the start becomes the end.

```
func MusicSequenceSetAUGraph(MusicSequence, AUGraph?) -> OSStatus
```

Associates an audio processing graph with a music sequence.

```
func MusicSequenceSetMIDIEndpoint(MusicSequence, MIDIEndpointRef) ->
OSStatus
```

Associates a specified MIDI endpoint with all music tracks in a music sequence.

```
func MusicSequenceSetSMPTEResolution(Int8, UInt8) -> Int16
```

```
func MusicSequenceSetSequenceType(MusicSequence, MusicSequenceType) ->
OSStatus
```

Sets the sequence type for a music sequence.

```
func MusicSequenceSetUserCallback(MusicSequence, MusicSequenceUser
Callback?, UnsafeMutableRawPointer?) -> OSStatus
```

Registers a user callback function with a music sequence.

`typealias MusicSequence`

A music sequence.

`typealias MusicSequenceUserCallback`

`struct MusicSequenceFileFlags`

Flags that configure the behavior of the `MusicSequenceFileCreate(_:_:_:_:_:)` and `MusicSequenceFileCreateData(_:_:_:_:_:)` functions.

`struct MusicSequenceLoadFlags`

Flags used to configure the behavior of the `MusicSequenceFileLoad(_:_:_:_:)` and `MusicSequenceFileLoadData(_:_:_:_:)` functions.

Managing Music Tracks

`func MusicTrackClear(MusicTrack, MusicTimeStamp, MusicTimeStamp) -> OSStatus`

Removes a specified range of music track events.

`func MusicTrackCopyInsert(MusicTrack, MusicTimeStamp, MusicTimeStamp, MusicTrack, MusicTimeStamp) -> OSStatus`

Copies a range of events from one music track and inserts them into another music track.

`func MusicTrackCut(MusicTrack, MusicTimeStamp, MusicTimeStamp) -> OSStatus`

Removes a specified range of music track events, and shifts later events toward the start of the track to fill in the gap.

`func MusicTrackGetDestMIDIEndpoint(MusicTrack, UnsafeMutablePointer<MIDIEndpointRef>) -> OSStatus`

Gets the MIDI endpoint that is the event target for a music track.

`func MusicTrackGetDestNode(MusicTrack, UnsafeMutablePointer<AUNode>) -> OSStatus`

Gets the audio unit node that is the event target for a music track.

`func MusicTrackGetProperty(MusicTrack, UInt32, UnsafeMutableRawPointer, UnsafeMutablePointer<UInt32>) -> OSStatus`

Gets a music track property value.

`func MusicTrackGetSequence(MusicTrack, UnsafeMutablePointer<MusicSequence?>) -> OSStatus`

Gets the music sequence that the music track is a member of.

`func MusicTrackMerge(MusicTrack, MusicTimeStamp, MusicTimeStamp, MusicTrack, MusicTimeStamp) -> OSStatus`

Copies a range of events from one music track and merges them into another music track.

`func MusicTrackMoveEvents(MusicTrack, MusicTimeStamp, MusicTimeStamp, MusicTimeStamp) -> OSStatus`

Shifts music track events forward or backward in time, in terms of beats.

`func MusicTrackNewAUPresetEvent(MusicTrack, MusicTimeStamp, UnsafePointer<AUPresetEvent>) -> OSStatus`

Adds an event of type `AUPresetEvent` to a music track.

`func MusicTrackNewExtendedNoteEvent(MusicTrack, MusicTimeStamp, UnsafePointer<ExtendedNoteOnEvent>) -> OSStatus`

Adds an event of type `ExtendedNoteOnEvent` to a music track.

`func MusicTrackNewExtendedTempoEvent(MusicTrack, MusicTimeStamp, Float64) -> OSStatus`

Adds a tempo to a music track.

`func MusicTrackNewMIDIChannelEvent(MusicTrack, MusicTimeStamp, UnsafePointer<MIDIChannelMessage>) -> OSStatus`

Adds an event of type `MIDIChannelMessage` to a music track.

`func MusicTrackNewMIDINoteEvent(MusicTrack, MusicTimeStamp, UnsafePointer<MIDINoteMessage>) -> OSStatus`

Adds an event of type `MIDINoteMessage` to a music track.

`func MusicTrackNewMIDIRawDataEvent(MusicTrack, MusicTimeStamp, UnsafePointer<MIDIRawData>) -> OSStatus`

Adds an event of type `MIDIRawData` to a music track.

`func MusicTrackNewMetaEvent(MusicTrack, MusicTimeStamp, UnsafePointer<MIDIMetaEvent>) -> OSStatus`

Adds an event of type `MIDIMetaEvent` to a music track.

`func MusicTrackNewParameterEvent(MusicTrack, MusicTimeStamp, UnsafePointer<ParameterEvent>) -> OSStatus`

Adds an event of type `ParameterEvent` to a music track.

`func MusicTrackNewUserEvent(MusicTrack, MusicTimeStamp, UnsafePointer<MusicEventUserData>) -> OSStatus`

Adds an event of type `MusicEventUserData` to a music track.

`func MusicTrackSetDestMIDIEndpoint(MusicTrack, MIDIEndpointRef) -> OSStatus`

Sets the music track's event target to a MIDI endpoint.

`func MusicTrackSetDestNode(MusicTrack, AUNode) -> OSStatus`

Sets the music track's event target to an audio unit node.

`func MusicTrackSetProperty(MusicTrack, UInt32, UnsafeMutableRawPointer, UInt32) -> OSStatus`

Sets a music track property value.

`typealias MusicTrack`

A music track consists of a series of music events, each timestamped using units of beats.

`struct MusicTrackLoopInfo`

Supports control of the looping behavior of a music track.

`struct MIDIChannelMessage`

Describes a MIDI channel message.

`struct MIDIMetaEvent`

Describes a MIDI metaevent such as lyric text, time signature, and so on.

`struct MIDINoteMessage`

Describes a MIDI note.

`struct MIDIRawData`

Describes a MIDI system-exclusive (SysEx) message.

Interacting with Music Devices

`func MusicDeviceMIDIEvent(MusicDeviceComponent, UInt32, UInt32, UInt32, UInt32) -> OSStatus`

`func MusicDeviceMIDIEventList(MusicDeviceComponent, UInt32, UnsafePointer<MIDIEventList>) -> OSStatus`

`func MusicDeviceStartNote(MusicDeviceComponent, MusicDeviceInstrumentID, MusicDeviceGroupID, UnsafeMutablePointer<NoteInstanceID>, UInt32,` `UnsafePointer<MusicDeviceNoteParams>) -> OSStatus`

`func MusicDeviceStopNote(MusicDeviceComponent, MusicDeviceGroupID, NoteInstanceID, UInt32) -> OSStatus`

`func MusicDeviceSysEx(MusicDeviceComponent, UnsafePointer<UInt8>, UInt32) -> OSStatus`

`typealias MusicDeviceComponent`

`typealias MusicDeviceGroupID`

`typealias MusicDeviceInstrumentID`

`typealias MusicDeviceMIDIEventProc`

`typealias MusicDeviceStartNoteProc`

`typealias MusicDeviceStopNoteProc`

`typealias MusicDeviceSysExProc`

Enumerations

Music Instrument Audio Unit Subtypes

Music Track Properties

Properties for music tracks.

`struct MusicSequenceFileFlags`

Flags that configure the behavior of the `MusicSequenceFileCreate(_:_:_:_:_:)` and `MusicSequenceFileCreateData(_:_:_:_:_:)` functions.

`enum MusicSequenceFileTypeID`

The various types of files that can be parsed by a music sequence.

`struct MusicSequenceLoadFlags`

Flags used to configure the behavior of the `MusicSequenceFileLoad(_:_:_:_:)` and `MusicSequenceFileLoadData(_:_:_:_:)` functions.

`enum MusicSequenceType`

The various types of music sequences.

Music Extended Control Event Type

Music Player Errors

- Music Event Types
- Music Note Events
- Music Device Selectors
- Music Device Properties
- Music Device Sample Frame Mask
- Music Device Unit Properties
- Instrument Types
- Music Device Generic Properties
- Music Effect and Instrument Unit Properties
- DLS Music Device Properties
- DLS Music Device Parameters

See Also

Playback and Recording

- Audio Queue Services

  Connect to audio hardware and manage the recording or playback process.

- Audio Services

  Play short sounds or trigger a vibration effect on iOS devices with the appropriate hardware.

- Anchoring sound to a window or volume

  Provide unique app experiences by attaching sounds to windows and volumes in 3D space.