**Media Library for iOS User Guide**

**version 2.23**

**Media Library for iOS** provides the classes and protocols for media live stream publishing and playback, or VOD recording and playback, via RTMP. It allows the iOS devices to interact with the media servers: Adobe FMS, Wowza MS, Red5, crtmp, rtmpd.

**Media Library for iOS** uses the classes and protocols of **Communication Library for iOS** (see *Communication Library for iOS User Guide*).

**Requirements**

The library imposes the following requirements on the iOS applications utilizing it:

1. iOS Deployment Target - 6.0 or above;
2. the following frameworks and libraries must be added to the list of libraries linked to the binary:

* *AudioToolbox.framework, AVFoundation.framework, CFNetwork.framework, CoreData.framework, CoreFoundation.framework, CoreGraphics.framework, CoreMedia.framework, CoreVideo.framework, Foundation.framework, Security.framework, libz.dylib*;
* *The Midnight Coders static library: CommLibiOS.a*;
* the FFmpeg/Libav static libraries: libavcodec.a, libavdevice.a, libavfilter.a, libavformat.a, libavutil.a, libswscale.a

**Constants**

typedef enum media\_player\_state MediaPlayerState;

enum media\_player\_state

{

CONN\_DISCONNECTED,

CONN\_CONNECTED,

STREAM\_CREATED,

STREAM\_PLAYING,

STREAM\_PAUSED

};

typedef enum video\_encoder\_resolution MPVideoResolution;

enum video\_encoder\_resolution

{

RESOLUTION\_LOW, // 144x192px (landscape) & 192x144px (portrait)

RESOLUTION\_CIF, // 288x352px (landscape) & 352x288px (portrait)

RESOLUTION\_MEDIUM, // 360x480px (landscape) & 480x368px (portrait)

RESOLUTION\_VGA, // 480x640px (landscape) & 640x480px (portrait)

};

typedef enum publish\_type PublishType;

enum publish\_type

{

PUBLISH\_RECORD,

PUBLISH\_APPEND,

PUBLISH\_LIVE,

};

**IMediaStreamEvent Protocol Reference**

Delegate instance of *BroadcastStreamClient* and *MediaStreamPlayer* classes must conform this protocol.

**Instance Methods**

**stateChanged:description:**

The delegate receive this method as notification about state change (state) and description of current state.

-(void)stateChanged:(MediaStreamState)state description:(NSString \*)description;

**Arguments**

*state* Current state of *MediaPlayerState* type

*description* Description of current state

**connectFaield:description:**

The delegate receive this method as notification about error situation (code) and description of this error.

-(void)connectFailed:(int)code description:(NSString \*)description;

**Arguments**

*code* Error code

*description* Error description

@optional

-(void)metadataReceived:(id)sender event:(NSString \*)event metadata:(NSDictionary \*)metadata;

-(void)pixelBufferShouldBePublished:(CVPixelBufferRef)pixelBuffer timestamp:(int)timestamp;

**IVideoPlayer Protocol Reference**

Delegate instance of *BroadcastStreamClient* and *MediaStreamPlayer* classes can conform this protocol for showing the video publishing or playback.

**Instance Methods**

**playImageFrame:**

The delegate receive this method with *VideoFrameData* instance for showing.

-(void)playVideoFrame:(VideoFrameData \*)data;

**Argument**

*data* Instance contains the video frame data

**playImageBuffer:**

(Optional). The delegate receive this method with *CVImageBufferRef*  reference to video frame structure in style of Core Video framework.

-(void)playImageBuffer:(CVImageBufferRef)frameBuffer;

**Argument**

*frameBuffer* Video frame data structure reference

**BroadcastStreamClient Class Reference**

**Properties**

**delegate**

A property with a reference to an instance of the *IMediaStreamEvent* protocol. Provides access to the connection-related events.

@property (nonatomic, assign) id <IMediaStreamEvent> delegate;

**player**

A property with a reference to an instance of the *IVideoPlayer*  protocol. Provides access to the video displaying events.

@property (nonatomic, retain) id <IVideoPlayer> player;

**state**

A property contains a current state of *MediaPlayerState* type.

@property MediaPlayerState state;

**Instance Methods**

**init:**

This is an instance constructor.

-(id)init:(NSString \*)url;

**Argument**

*url* URL of the server where the RTMP server is running

**initWithClient:**

This is an instance constructor.

-(id)initWithClient:(RTMPClient \*)client;

**Argument**

*client* Existing RTMP connection

**connect:name:publishType:**

Establishes an RTMP connection to the application on the specified RTMP server running on the given URL with name of file/stream and publishType (record, append or live).

-(BOOL)connect:(NSString \*)url name:(NSString \*)name publishType:(PublishType)type;

**Arguments**

*url* URL of the server where the RTMP server is running

*name* The name of recorded file or live stream

*publishType* The publish action (record, append or live) of PublishType constants

**attach:name:publishType:**

Uses the given RTMP connection to the application for publish of file/stream with name and publishType (record, append or live).

-(BOOL)attach:(RTMPClient \*)client name:(NSString \*)name publishType:(PublishType)type;

**Arguments**

*client* Existing RTMP connection

*name* The name of recorded file or live stream

*publishType* The publish action (record, append or live) of PublishType constants

**stream:publishType:**

Uses the existing RTMP connection to the application for publish of file/stream with name and publishType (record, append or live).

-(BOOL)stream:(NSString \*)name publishType:(PublishType)type;

**Arguments**

*name* The name of recorded file or live stream

*publishType* The publish action (record, append or live) of PublishType constants

**start**

Begins a media capturing and publishing.

-(void)start;

**pause**

Sets a pause in capturing and publishing the named file or live stream.

-(void)pause;

**resume**

Resets the pause in capturing and publishing the named file or live stream.

-(void)resume;

**stop**

Finishes a media capturing and publishing.

-(void)stop;

**disconnect**

Releases earlier established RTMP connection.

-(void)disconnect;

**MediaStreamPlayer Class Reference**

**Properties**

**delegate**

A property with a reference to an instance of the *IMediaStreamEvent* protocol. Provides access to the connection-related events.

@property (nonatomic, assign) id <IMediaStreamEvent> delegate;

**player**

A property with a reference to an instance of the *IVideoPlayer*  protocol. Provides access to the video displaying events.

@property (nonatomic, retain) id <IVideoPlayer> player;

**state**

A property contains a current state of *MediaPlayerState* type.

@property MediaPlayerState state;

**Instance Methods**

**init:**

This is an instance constructor.

-(id)init:(NSString \*)url;

**Argument**

*url* URL of the server where the RTMP server is running

**initWithClient:**

This is an instance constructor.

-(id)initWithClient:(RTMPClient \*)client;

**Argument**

*client* Existing RTMP connection

**connect:name:**

Establishes an RTMP connection to the application on the specified RTMP server running on the given URL with name of file/stream for playing

-(BOOL)connect:(NSString \*)url name:(NSString \*)name;

**Arguments**

*url* URL of the server where the RTMP server is running

*name* The name of played file or live stream

**attach:name:**

Uses the given RTMP connection to the application for playing of file/stream with given name.

-(BOOL)attach:(RTMPClient \*)client name:(NSString \*)name;

**Arguments**

*client* Existing RTMP connection

*name* The name of recorded file or live stream

**stream:**

Uses the existing RTMP connection to the application for playing of file/stream with given name.

-(BOOL)stream:(NSString \*)name;

**Argument**

*name* The name of recorded file or live stream

**start**

Begins a playing the named file or live stream.

-(void)start;

**isPlaying**

Returns YES if file/stream is playing, otherwise - NO.

-(BOOL)isPlaying;

**pause**

Sets a pause in playing the named file or live stream.

-(void)pause;

**resume**

Resets the pause in playing the named file or live stream.

-(void)resume;

**stop**

Finishes a media playing.

-(void)stop;

**disconnect**

Releases earlier established RTMP connection.

-(void)disconnect;

**VideoFrameData Class Reference**

**Properties**

**data**

A property contains a frame buffer (byte array).

@property (readonly) uint8\_t \*data;

**size**

A property contains a size of frame buffer.

@property size\_t size;

**bytesPerRow**

A property contains a value of frame bytes per row parameter.

@property size\_t bytesPerRow;

**width**

A property contains a value of frame width parameter.

@property size\_t width;

**height**

A property contains a value of frame height parameter.

@property size\_t height;

**timestamp**

A property contains a value of frame timestamp parameter.

@property int timestamp;

**baseAddress**

A property contains a reference to external frame buffer.

@property uint8\_t \*baseAddress;

**Instance Constructors**

**initWithData:**

This is an instance constructor.

-(id)initWithData:(uint8\_t \*)\_data size:(size\_t)\_size;

**Arguments**

*data* frame buffer

*size* size of frame buffer

**initWithFrame:**

This is an instance constructor.

-(id)initWithFrame:(uint8\_t \*)\_data size:(size\_t)\_size width:(size\_t)\_width height:(size\_t)\_height timestamp:(int)\_timestamp;

**Arguments**

*data* frame buffer

*size* size of frame buffer

*width* frame width

*height* frame height

*timestamp* frame timestamp

**videoFrame:**

This is an autoreleased instance constructor.

+(id)videoFrame:(uint8\_t \*)\_data size:(size\_t)\_size width:(size\_t)\_width height:(size\_t)\_height timestamp:(int)\_timestamp;

**Arguments**

*data* frame buffer

*size* size of frame buffer

*width* frame width

*height* frame height

*timestamp* frame timestamp