

Network Media Player SDK for Android Programmer's Guide

V.E.G., LLC.

Aug 27, 2014

Content

1. Overview	- 2 -
2. How to Use.....	- 2 -
2.1 Android version	- 2 -
2.2 Folders and files	- 2 -
2.3 Development tools	- 3 -
2.4 Integration with an application	- 3 -
2.4.1 Integration using a resource file in 2 steps:.....	- 3 -
2.4.2 Integration dynamically (without modifying resources)	- 5 -
2.4.3 Integration with Activity	- 6 -
3. Media Player	- 8 -
3.1 API Reference	- 8 -
3.2 Notifications	- 8 -
3.2 Functions description	- 9 -
4. Thumbnailer.....	- 25 -
4.1 Functions description	- 25 -

1. Overview

Network Media Player SDK consists of a set of resources for fast and convenient development of mobile applications for viewing various media streams like RTMP, HLS, RTSP, RTP, MMS, WebM, FLV, MP4, TS ,other network video format and playback files with following format : AVI, MOV, MKV,FLV,AVI,3GP,3G2,ASF,WMV,MP4,M4V,TP,TS,MTP,M2T and other. The core of the SDK is a library for application development.

Key Features

Hardware acceleration – a new hardware accelerated decoder for HD video.

Multi-core decoding - support of the multiple processor cores for decoding.

Multi-channel support - simultaneous connection to multiple resources or multiple video channels and simultaneous video decoding.

Video integration with any Activity - based on SurfaceView and can be integrated with any Activity.

Hardware pre and post video processing – hardware de-interlacing and various pre and post video processing using OpenGL shaders.

Custom and standard notifications - notifies application about connection, disconnection and other events, possibility to add custom event.

Smart and online thumbnail – quick and simple API to get a thumbnail for local files and network streams.

Low latency for network stream – special API to control playback latency

2. How to Use

2.1 Android version

The SDK works with Android version 4.0 or newer . (Lower version 4.0 can be customized and provided by request as well).

2.2 Folders and files

The SDK package consists of the following folders.

bin Sample application package

MediaPlayerSDKTest.apk

libs Library files to be linked to the application

mediaplayersdk.jar

librtspplr-xx.so

librtstm-xx.so

libSDL2-xx.so

libxml2-xx.so

libyuv_shared-xx.so

where xx is one of supported platform : ARM general, x86, ARV V7 , ARM V7a.

src Sample project to test Media Player SDK

doc Documentation including this document

2.3 Development tools

Build environment is Eclipse. Please import the project to Eclipse for building the sample application.

2.4 Integration with an application

2.4.1 Integration using a resource file in 2 steps:

Step1: Add to layout xml for your activity as below:

```
<FrameLayout
    android:id="@+id/playerViewLayout"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" >

    <veg.mediaplayer.sdk.MediaPlayer
        android:id="@+id/playerView"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_gravity="center" />

</FrameLayout>
```

Step 2: Change main activity

(MainActivity.java)

```
public class MainActivity extends Activity implements
MediaPlayer.MediaPlayerCallback
{
...
    // callback handler
    #override
    public int Status(int arg) {return 0;}

    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        ...
        // Create Playr instance
        player = (MediaPlayer)findViewById(R.id.playerView);
// Get player instance
        ...
        // Connect or start playback
        player.Open(ConnectionUrl or File name,
decoderType,
rendererType,
synchroEnable,
synchroNeedDropVideoFrames,
                rendererEnableColorVideo,
                rendererEnableAspectRatio,
DataReceiveTimeout,
decoderNumberOfCpuCores,
this);
        ...
    }

    @Override
    protected void onDestroy()
    {
        // Destroy and close player
        if (player != null)
```

```
    {  
        // Close connection to server  
        player.close ();  
        // Destroy player  
        player.onDestroy();  
    }  
    super.onDestroy();  
}  
...  
}
```

2.4.2 Integration dynamically (without modifying resources)

Step 1: Change main activity

```
public class MainActivity extends Activity implements  
MediaPlayer.MediaPlayerCallback  
{  
...  
    // callback handler  
    #override  
    public int Status(int arg) {return 0;}  
  
    @Override  
    public void onCreate(Bundle savedInstanceState)  
    {  
        ...  
        // Create instance of Player  
        player = new MediaPlayer(this);  
        // Set size and position for layout  
        FrameLayout.LayoutParams params = new FrameLayout.LayoutParams(250,250,  
Gravity.CENTER);  
        player.setLayoutParams(params);  
  
        //  
        // Add Player Instance to layout
```

```
        FrameLayout lp = (FrameLayout)findViewById(R.id.playerView);
        lp.addView(player);
        ...
    // connect and start playback
        player.Open(  ConnectionUrl,
                      decoderType,
                      rendererType,
                      synchroEnable,
                      synchroNeedDropVideoFrames,
                      rendererEnableColorVideo,
                      rendererEnableAspectRatio,
                      DataReceiveTimeout,
                      decoderNumberOfCpuCores,
                      this);
        ...
    }

    @Override
    protected void onDestroy()
    {
        // Close network connection to server
        player.close ();
        // Destroy player
        player.onDestroy();
        super.onDestroy();
    }
```

2.4.3 Integration with Activity

The SDK is based on SurfaceView and can be integrated with any Activity using the code below:

```
<FrameLayout

        android:id="@+id/playerViewLayout"

        android:layout_width="fill_parent"
```

```
        android:layout_height="wrap_content" >

        <veg.mediaplayer.sdk.MediaPlayer

            android:id="@+id/playerView"

            android:layout_width="fill_parent"

            android:layout_height="fill_parent"

            android:layout_gravity="center" />

    </FrameLayout>
```


3. Media Player

3.1 API Reference

There are following API providers in SDK: content provider, decoder provider and render provider:

Provider name	Provider acronym	Description
Pipeline Provider	PLP_	Control pipeline and all components
Content Provider	CP_	Connect to server, download data and control connection
Video Decoder Provider	VDP_	s/w or h/w video decoding
Audio Decoder Provider	ADP_	s/w or h/w video decoding
Video renderer Provider	VRP_	Video renderer
Audio renderer Provider	ARP_	audio renderer

3.2 Notifications

Providers notifies about results, errors and notifications using “MediaPlayerCallback” callback. All messages are synchronous and provider wait until the application handles a message.

Value	Name	Type	Description
1	PLP_BUILD_STARTING	NOTIFICATION	PLP notifies that pipeline is started to build
2	PLP_BUILD_SUCCESSFUL	RESULT	Pipeline has been built successfully
3	PLP_BUILD_FAILED	RESULT	Pipeline can not be built
4	PLP_PLAY_STARTING	NOTIFICATION	Pipeline is going to starting
5	PLP_PLAY_SUCCESSFUL	RESULT	Pipeline has been ran successfully
6	PLP_PLAY_FAILED	RESULT	Error on pipeline starting
7	PLP_CLOSE_STARTING	NOTIFICATION	Pipeline is going to stopping
8	PLP_CLOSE_SUCCESSFUL	RESULT	Pipeline has been closed successfully
9	PLP_CLOSE_FAILED	RESULT	Error on pipeline closing
10	PLP_ERROR	ERROR	Pipeline is disconnected due inner error

101	CP_CONNECT_STARTING	NOTIFICATION	CP is initialized and is going to start connection
102	CP_CONNECT_SUCCESSFUL	RESULT	CP has been connected successfully
103	CP_CONNECT_FAILED	RESULT	CP notifies that connection is failed
104	CP_INTERRUPTED	RESULT	CP notifies that connection with server is interrupted by close function
105	CP_ERROR_DISCONNECTED	NOTIFICATION	CP notifies that connection with server is lost
106	CP_STOPPED	NOTIFICATION	CP has been stopped
107	CP_INIT_FAILED	RESULT	CP notifies that there is error on initialization
201	VDP_STOPPED	NOTIFICATION	VDP has been stopped
202	VDP_INIT_FAILED	RESULT	VDP notifies that there is error on initialization
300	VRP_STOPPED	NOTIFICATION	VRP has been stopped
301	VRP_INIT_FAILED	RESULT	VRP notifies that there is error on initialization
302	VRP_NEED_SURFACE	NOTIFICATION	VRP notifies that it is going to allocate surface
400	ADP_STOPPED	RESULT	ADP has been stopped
401	ADP_INIT_FAILED	RESULT	ADP notifies that there is error on initialization
500	ARP_STOPPED	NOTIFICATION	ARP has been stopped
501	ARP_INIT_FAILED	NOTIFICATION	ARP notifies that there is error on initialization

3.2 Functions description

Following functions are member of MediaPlayer class . These functions should be used to playback network content and media files.

Open

Connect to network server or open media file, create pipeline and playback media data.

Definition

```
public void Open(
    final String ConnectionUrl,
    final int DataReceiveTimeout,
```

final MediaPlayerCallback callback)

```
public void Open(    final String  ConnectionUrl,
final  int ConnectionNetworkProtocol,
final int ConnectionDetectionTime,
final int ConnectionBufferingTime,
final int DecoderType,
final int RendererType,
final int SynchroEnable,
final int SynchroNeedDropVideoFrames,
final int EnableColorVideo,
final int EnableAspectRatio,
final int DataReceiveTimeout,
final int NumberOfCPUCores,
final MediaPlayerCallback callback)
```

```
public void Open(final MediaPlayerConfig config, final MediaPlayerCallback callback)
```

Parameters

ConnectionUrl	URL to network resource (RTSP, HTTP ,RTMP, HLS, UPD and so on) or full path for media file
ConnectionNetworkProtocol	network protocol or RTP or RTSP tunneling (0 – RTP by UDP, 1 – RTP by TCP, 2 – RTSP over http, 3 – RTSP over https, -1 - AUTO)
ConnectionDetectionTime	Probing time to detect video and audio format of network stream (in milliseconds)
ConnectionBufferingTime	Buffering on playback start to avoid network jitter (in milliseconds)
DecoderType	select s/w or h/w video decoder
RendererType	select SDL or openGL render
SynchroEnable	enable A/V synchronization 1- synchronization is on, 0 is off
SynchroNeedDropVideoFrames	drop video framer if frame is later 1 is on , 0 is off
EnableColorVideo	Enable grayscale video
EnableAspectRatio	Set video output mode (0 - stretch, 1 – fit to screen with aspect ratio, 2 - crop, 3 - 100% size, 4 - zoom mode, 5 - move mode)
DataReceiveTimeout	reconnect timeout, SDK does reconnect if there is not

<p>MediaPlayerCallback</p> <p>NumberOfCpuCores</p>	<p>received data during some time (milliseconds).</p> <p>notification callback, event is provided over this callback</p> <p>Number of CPU core to decode video, 0 – autodetect and set the number according device capability, positive number sets number according application needs</p>
--	--

Return Value

Upon successful completion **Open()** returns 0. Otherwise -1 is returned . All errors are provided in callback status.

Remarks

Connect to network resource or open local media file, create pipeline , allocate resource and start video playback.

Examples

Example N1

```

player.Open(
    "http://example", // correct URL or full path for media file
    2,                // RTSP over http tunneling
    500,              // 500 ms on probing
    500,              // 500 ms buffer on start
    0,                // Decoder type : 0- S/W 1, - H/W
    1,                // Renderer Type : 0 - SDL, 1 - pure OpenGL
    1,                // A/V synchronization: 1- Enable , 0 - Disable
    0,                // Drop Video frame if it is late : 1- Enable , 0 – Disable
    1,                // Color / Grayscale video output : 0 - grayscale, 1 – color
    1,                // Aspect ratio / Full size : 1 – aspect rate
    30000,            // Reconnection timeout (milliseconds),
    0,                // Number Of Cpu Cores for decoding (1-6), 0-autodetect
    This);
    
```

Example N2

```

// Create config
MediaPlayerConfig conf = new MediaPlayerConfig();
conf.setConnectionUrl(http://example); // correct URL or full path for media file
conf.setConnectionNetworkProtocol(2); // RTSP over http tunneling
conf.setConnectionDetectionTime(500); // Probing time – 500 ms
    
```

```
conf.setConnectionBufferingTime(500); // Buffering on start – 500 ms
conf.setDecodingType(1); // H/W decoder
conf.setRendererType(1); // pure OpenGL
conf.setSynchroEnable(1); // Audio and Video synchronization is ON

conf.setSynchroNeedDropVideoFrames(0); // Do not drop video if pts is later
conf.setEnableColorVideo(1); // Set color video
conf.setEnableAspectRatio(1); // Set aspect ratio
conf.setDataReceiveTimeout(30000); // Set timeout of connection , Disconnect event is
sent after(in milliseconds)
conf.setNumberOfCPUCores(0); // Number Of Cpu Cores for decoding (1-6), 0-
autodetect
player.Open(conf, This);
```

OpenAsPreview

Connect to network server or open media file, create pipeline and playback media data in Preview mode. Preview mode differs from normal : s/w decoding only key frames , real time render, no audio stream (only video).

Definition

```
public void OpenAsPreview(
    final String ConnectionUrl,
    final int DataReceiveTimeout,
    final MediaPlayerCallback callback)
```

Parameters

ConnectionUrl	URL to network resource (RTSP,HTTP,RTMP,UPD) or full path for media file
DataReceiveTimeout	reconnect timeout, SDK does reconnect if there is not received data during some time (milliseconds)
MediaPlayerCallback	notification callback, event is provided over this callback

Return Value

Upon successful completion **OpenAsPreview()** returns 0. Otherwise -1 is

returned . All errors are provided in callback status.

Remarks

Connect to network resource or open local media file, create pipeline , allocate resource and start playback in Preview mode.

Examples

```
player.OpenAsPreview(  
    "http://example", // correct URL or full path for media file  
    30000, // Connection timeout (milliseconds),  
    This);
```

Play

Resume play if player is in Pause state.

Definition

```
public void Play()
```

Parameters

There are no parameters for this call

Return Value

Upon successful completion, **Play()** returns 0. Otherwise -1 is returned . All errors are provided in callback status.

Remarks

Resume play if player is in Pause state. This function can be used with playback from file only.

Examples

```
player.Play ();
```

Pause

Change playback state from Play to Pause.

Definition

public void Pause()

Parameters

There are no parameters for this call

Return Value

Upon successful completion, **Pause()** returns 0. Otherwise -1 is returned . All errors are provided in callback status.

Remarks

Pause playback if player is in Play state. This function can be used with playback from file only.

Examples

```
player.Pause ();
```

GetState

Return player state.

Definition

public PlayerState getState()

Parameters

There are no parameters for this call

Return Value

Following states are provided :

- 0 - Opening
- 1 - Opened
- 2 - Started
- 3 - Paused
- 4 - Stopped
- 5 - Closing
- 6 - Closed

Remarks

Provide the current state of player.

Examples

```
if (player.getState() == PlayerState.Closing) ;
```

getStreamDuration

Return duration of media file in seconds. This function works only in case file playback.

Definition

```
public long getStreamDuration()
```

Parameters

There are no parameters for this call.

Return Value

Upon successful completion, `getStreamDuration()` returns file duration in seconds . Otherwise -1 is returned . All errors are provided in callback status.

Remarks

Provide the file duration that is played by player.

Examples

```
int duration = getStreamDuration() ;
```

getStreamPosition

Get position in played media file. This function works only in case of file playback.

Definition

```
public long getStreamPosition()
```

Parameters

There are no parameters for this call.

Return Value

Upon successful completion, `getPosition()` returns current position of played file (in seconds).

Remarks

Provide the file position that is played by player.

Examples

```
int position = getPosition() ;
```

setPosition

Set position of played media file. This function works only in case of file playback.

Definition

```
public void setPosition(final long lTime)
```

Parameters

`lTime` - new position in file (in seconds)

Return Value

No value is returned by function `setPosition`.

Remarks

Provide the file position of file that is played by player.

Examples

```
long position;  
setPosition(position) ;
```

getLiveStreamPosition

Function provides position, first and last position for live stream. This function works only in case of live stream playback (HLS).

Definition

```
Position getLiveStreamPosition()
```

Parameters

There are no parameters for this call.

Return Value

Upon successful completion, `getLiveStreamPosition` returns `Position` object.

```
public class Position
{
    private long first = 0;
    private long current = 0;
    private long last = 0;
    private long duration = 0;
}
```

`first` - dts of first segment in m3u8 list.

`last` - dts of last segment in m3u8 list.

`current` - dts of last downloaded packet in HLS stream .

Time base is milliseconds.

Remarks

Provide the `current` , `first` , `last` positions in stream that is played by player.

Examples

```
Position pos = player.getLiveStreamPosition();
if (pos == null)
{
    long duration = pos.getDuration();
    long first = pos.getFirst();
    long current = pos.getCurrent();
    long last = pos.getLast();
}
```

setLiveStreamPosition

Change position of played live stream. This function works only in case of live stream.

Definition

`public void setLiveStreamPosition(final long lTime)`

Parameters

`lTime` - new position in live stream (milliseconds)

Return Value

No value is returned by function `setStreamPosition`.

Remarks

Change the position of live stream that is played by player.

Examples

```
int position = getStreamPosition() ;
```

getRenderPosition

Function provides last position in played media file. This function works only in case of file playback.

Definition

`public long getRenderPosition()`

Parameters

There are no parameters for this call.

Return Value

Upon successful completion, `getStreamPosition()` returns PTS of last video frame or audio sample (milliseconds).

Remarks

Provide the PTS of last played video frame or audio sample.

Examples

```
long position = getRenderPosition() ;
```

Close

Disconnect from server and destroy pipeline.

Definition

```
public void Close()
```

Parameters

There are no parameters for this call

Return Value

Upon successful completion, **Close()** returns 0. Otherwise -1 is returned. All errors are provided in callback status.

Remarks

Disconnect from network server, destroy pipeline, free all resources that were allocated on Open() call.

Examples

```
player.Close ();
```

UpdateView

Set video output mode for current player instance.

Definition

```
public int UpdateView(final boolean isAspectRatioEnabled)  
public int UpdateView()
```

Parameters

isAspectRatioEnabled – set aspect ratio that is set in network stream, 1 – set aspect ratio that is set in network stream, 0 – resize picture on full screen.

Return Value

Upon successful completion, isAspectRatioEnabled 0 returns 0, otherwise -1 is returned. All errors are provided in callback status.

Remarks

UpdateView(1) sets aspect ratio or full screen mode. This function can be used during playback. UpdateView() function uses settings that are set in player config structure.

Video output mode of output picture

```
player.getConfig().setAspectRatioMode(VideoOutputMode);
```

VideoOutputMode can be :

0 – stretch

1 – fit to screen with aspect ratio

2 – crop video

3 - 100% size of picture

4 - zoom mode

Zoom multiplier of output picture (in percent,25-400%) is set in player config :

```
player.getConfig().setAspectRatioZoomModePercent(ZoomMultiplier);
```

5 - move mode

X and Y position is set in player config:

X position of output picture (in percent, 0-100%)

```
player.getConfig().setAspectRatioMoveModeX(X);
```

Y position of output picture (in percent, 0-100%)

```
player.getConfig().setAspectRatioMoveModeY(Y);
```

// zoom and move modes are experimental function, There can be issue in these modes.

Examples

Example N1

```
player.UpdateView (0);
```

Example N2

// Present video : picture size is 100% in center of screen

```
player.getConfig().setAspectRatioMoveModeX(50); // 50% center of screen
```

```
player.getConfig().setAspectRatioMoveModeY(50); // 50% center of screen
```

```
player.getConfig().setAspectRatioZoomModePercent(100); //size is 100%
```

```
player.getConfig().setAspectRatioMode(5); // Zoom and move mode
```

```
player.UpdateView();
```

backgroundColor

Set background color of player.

Definition

```
public void backgroundColor(final int clr)
```

Parameters

clr – color in RGB format (ARGB is not supported).

Return Value

Upon successful completion, `backgroundColor()` returns 0, otherwise -1 is returned. All errors are provided in callback status.

Remarks

Set background color of player.

Examples

```
backgroundColor(Color.BLACK);
```

setVisibility

Set the enabled state of this view

Definition

```
public void setVisibility(int visibility)
```

Parameters

visibility – Controls the initial visibility of the view. Value of parameters are described on android documentation.

http://developer.android.com/reference/android/view/View.html#attr_android:visibility

Return Value

No value is returned by function `setVisibility`.

Examples

```
player.setVisibility (1);
```

getVideoShot

Capture video picture from video stream.

Definition

```
public VideoShot getVideoShot(  
    final int desiredWidth,  
    final int desiredHeight  
)
```

Parameters

desiredWidth - width picture of returned picture

desiredHeight - height picture of returned picture

Return Value

Upon successful completion, `getVideoShot ()` returns `VideoShot` object.

```
public class VideoShot  
{  
    public int getWidth();  
    public int getHeight();  
    public ByteBuffer getData();  
}
```

Remarks

Provide the video shot of last render frame in format `ARGB_8888`. This function works in Preview mode only.

Example

```
VideoShot vs = player.getVideoShot(width, height);  
Bitmap bm = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888);  
bm.copyPixelsFromBuffer(vs.getData());
```

GetStatFPS

Return frame rate of downloaded stream so application can estimate possibility of network for defined stream.

Definition

```
public int GetStatFPS ()
```

Parameters

There are no parameters for this call.

Return Value

Upon successful completion, **GetStatFPS()** returns fps of network stream . It is frame rate of stream that is downloaded from network, otherwise -1 is returned . All errors are provided in callback status

Remarks

Provide the frame rate of captured stream (download speed) to estimate if network speed is enough to playback stream in real time.

Example

```
Int fps = player.GetStatFPS();
```

GetStatPercFree

Return fullness of inner buffers in pipeline so application can estimate if device can playback data in real time or latency.

Definition

```
public int GetStatPercFree ()
```

Parameters

There are no parameters for this call

Return Value

Upon successful completion, **GetStatPerFree()** returns level of capacity for inner buffers , otherwise -1 is returned . All errors are provided in callback status

Remarks

Return fullness of inner buffers in pipeline so application can estimate if device can playback data in real time or latency.

Example

```
Int buf_level = GetStatPerFree ();
```

IsHardwareDecoding

Return what decoder(s/w or h/w) is used by player.

Definition

```
public boolean IsHardwareDecoding ()
```

Parameters

There are no parameters for this call.

Return Value

Upon successful completion, IsHardwareDecoding returns true if h/w decoder is used and false in case s/w decoder

Remarks

Provide h/w or s/w video decode is used in player.

Example

```
Boolean hw_decoder = IsHardwareDecoding ();
```

4. Thumbnailer

Thumbnailer is Class that provides the functionality to make thumbnails and stream information for local files and network streams. Smart searching is used to make Thumbnail with maximum informativity.

4.1 Functions description

Following functions are member of Thumbnailer class . These functions should be used to get a thumbnail for file or network stream.

Open

Connect to network server or open local media file.

Definition

```
public Object Open(final String ConnectionUrl)
public Object Open(final ThumbnailerConfig config)
```

Parameters

ConnectionUrl	URL of network resource (RTSP, HLS, RTMP, MMS, UDP and so on) or full path of local media file.
---------------	---

Class ThumbnailerConfig provides additional setting to open Thumbnailer.

```
public ThumbnailerConfig( String connectionUrl,
                           int connectionNetworkProtocol,
                           int dataReceiveTimeout,
                           int numberOfCPUCores,
                           float bogomIPS )
```

connectionUrl	URL of network resource (RTSP, HLS, RTMP, MMS, UDP and so on) or full path of local media file
connectionNetworkProtocol	Protocol for RTP or RTSP tunneling, 0 – RTP by UDP, 1 – RTP by TCP, 2 – RTSP and HTTP tunneling , -1 – AUTO mode
dataReceiveTimeout	reconnect timeout, SDK does reconnect if there is not received data during some time (milliseconds)

numberOfCPUCores	Number of CPU core to decode video, 0 – autodetect and set the number according device capability, positive number sets number according application needs
------------------	--

Return Value

Upon successful completion **Open()** returns 0. Otherwise -ERROR is returned .

Remarks

Connect to network resource or open local media file, create pipeline , allocate resource. This function should be called before get Frame.

Example N1

```
thumbnailer.Open("http://example.com");
```

Example N2

```
thumbnailer.Open(ThumbnailerConfig);
```

getFrame

Capture thumbnail frame.

Definition

```
public ThumbnailFrame getFrame()
```

Parameters

There are no parameters for this call.

Return Value

Upon successful completion, getFrame returns ThumbnailFrame object.

```
public class ThumbnailFrame
{
    public int getWidth() ;
    public int getHeight() ;
    public ByteBuffer getData();
}
```

Remarks

Provide the thumbnail for local file or stream in format ARGB_8888.

Example

```
ThumbnailFrame frame = thumbnailer.getFrame();
shot.getData().rewind();
Bitmap bmp = Bitmap.createBitmap(shot.getWidth(), shot.getHeight(),
Bitmap.Config.ARGB_8888);
bmp.copyPixelsFromBuffer(shot.getData());
```

getInfo

Function returns the information about media file or network stream.

Definition

```
public String getInfo()
```

Parameters

There are no parameters for this call.

Return Value

Type – String, Format - xml.

Remarks

Provide the information about media file or network stream.

String is xml format like below :

```
<?xml version=1.0?>
<StreamInfo name="AVFileFormat" version="1.0">
<name      value="test.mp4" />
<duration  value="100" />
  <VideoStreams>
    <VideoStream id=0 >
      <format      value="h264" />
      <duration    value="100"  />
      <width       value="1920" />
      <height      value="1080" />
      <fps         value="30"  />
```

```
        </VideoStream>
    </VideoStreams>
    <AudioStreams>
        "<AudioStream id=1 >
            <format      value="aac"  />
            <duration    value="100"  />
            <samplerate   value="48000" />
            <channels     value="2"    />
        </AudioStream>
    </AudioStreams>
</StreamInfo>
```

Examples

```
String info = thumbnailer.getInfo();
```

GetState

Return thumbnailer state : .

Definition

```
public ThumbnailerState getState()
```

Parameters

There are no parameters for this call

Return Value

Following states are provided :

- 0 - Opening,
- 1 - Opened,
- 2 - Closing,
- 3 - Closed;

Remarks

Provide the current state of Thumbnailer.

Example

```
if (thumbnailer.getState() == ThumbnailerState.Opened) ;
```

Close

Disconnect from server or close file and destroy all resources.

Definition

```
public void Close()
```

Parameters

There are no parameters for this call

Return Value

Upon successful completion, **Close()** returns 0. Otherwise -ERROR is returned .

Remarks

Disconnect from network server, destroy pipeline, free all resources that were allocated on Open() call.

Example

```
thumbnailer.Close ();
```