OnStream MediaPlayer+ User Guide

VisualOn, Inc.

September, 2015

20150930

VisualOn

Copyright/Confidentiality Notice

Proprietary and Confidential Information Notice

Contents

VisualOn

Figures

Tables

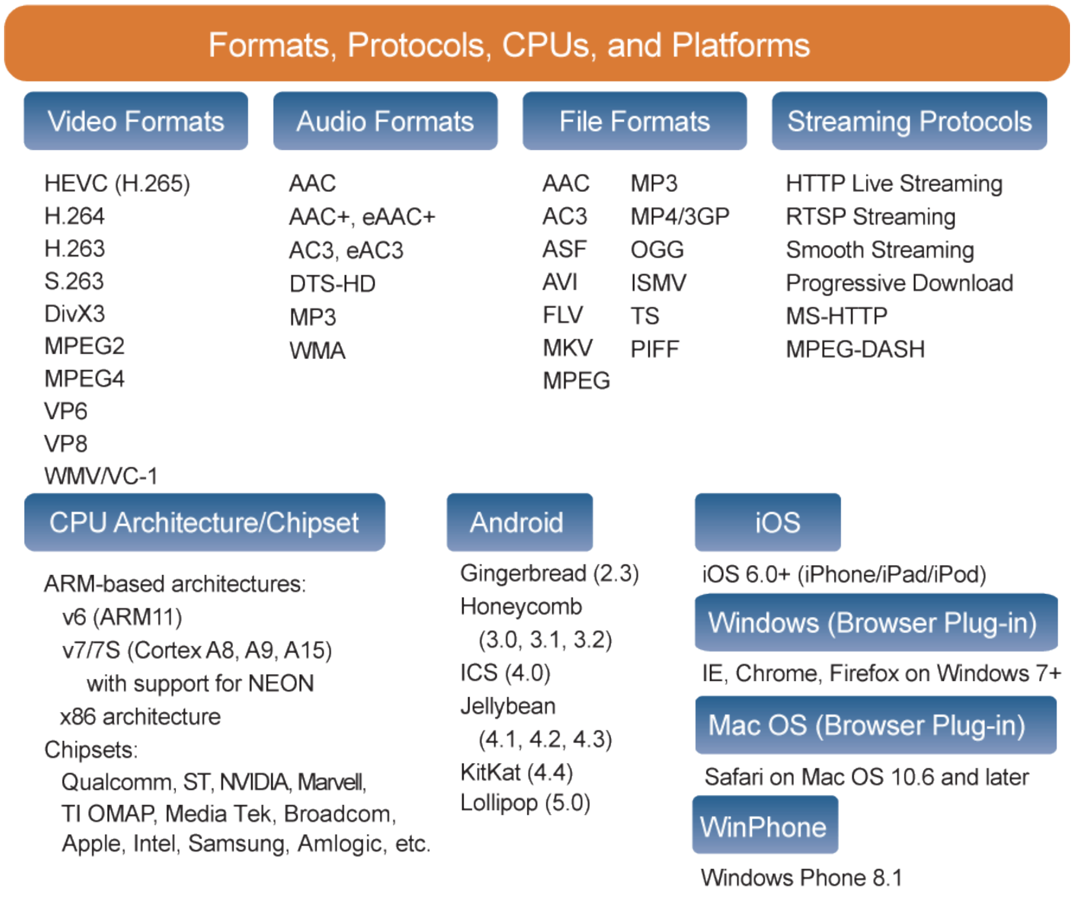
Chapter 1 Introduction

This chapter presents the following topics:

VisualOn

1.1 OSMP+ overview

OnStream MediaPlayer+ (referred hereafter to as OSMP+) is a multimedia player development kit enabling cross-platform content delivery and playback on connected devices including mobile handsets, tablets, desktops, set-top boxes, and smart TVs. OSMP+ enables the development of both enterprise and consumer grade applications to meet the latest trends in high-quality video/audio delivery and playback. Figure 1 demonstrates the video formats, audio formats, platforms, CPUs, and protocols supported by OSMP+.



Formats, Protocols, CPUs, and Platforms

Video Formats

HEVC (H.265)
H.264
H.263
S.263
DivX3
MPEG2
MPEG4
VP6
VP8
WMV/VC-1

Audio Formats

AAC
AAC+, eAAC+
AC3, eAC3
DTS-HD
MP3
WMA

File Formats

AAC
AC3
ASF
AVI
FLV
MKV
MPEG
MP3
MP4/3GP
OGG
ISMV
TS
PIFF

Streaming Protocols

HTTP Live Streaming
RTSP Streaming
Smooth Streaming
Progressive Download
MS-HTTP
MPEG-DASH

CPU Architecture/Chipset

ARM-based architectures:
v6 (ARM11)
v7/7S (Cortex A8, A9, A15)
with support for NEON
x86 architecture
Chipsets:
Qualcomm, ST, NVIDIA, Marvell,
TI OMAP, Media Tek, Broadcom,
Apple, Intel, Samsung, Amlogic, etc.

Android

Gingerbread (2.3)
Honeycomb
(3.0, 3.1, 3.2)
ICS (4.0)
Jellybean
(4.1, 4.2, 4.3)
KitKat (4.4)
Lollipop (5.0)

iOS

iOS 6.0+ (iPhone/iPad/iPod)

Windows (Browser Plug-in)

IE, Chrome, Firefox on Windows 7+

Mac OS (Browser Plug-in)

Safari on Mac OS 10.6 and later

WinPhone

Windows Phone 8.1

Figure 1. Supported protocols, formats, CPUs, and platforms

OSMP+ is based on a modular architecture that delivers maximum portability and scalability. Its common application programming interface (API) enables re-use and scalable deployment across multiple platforms. The modular architecture also accelerates feature-rich development. With OSMP+, integration into content delivery applications, with third party DRM modules, or with custom protocols is fast and easy. Video and audio post-processing support enables features such as closed captions and subtitles rendering, and advanced audio processing.

The OSMP+ toolkit supports a set of highly optimized software multimedia codecs for the most popular formats, as well as support for multiple streaming protocols across a wide range of media servers. OSMP+ builds upon market-proven technology that currently powers over 200 million mobile devices.

1.1.1 OSMP+ benefits

OSMP+ offers the following benefits when deploying a content delivery solution:

- Modular and flexible architecture enables easy integration into content delivery applications and addition of new or custom protocols.
- Enables development of multimedia players with no device/OS/protocol dependencies
- Enables delivery of uniform and compelling viewer experiences across multiple platforms
- Enables the efficient implementation, small memory footprint, and runtime memory usage
- Enables scalable deployment and single software codebase
- Enables unprecedented performance and reliability across all targeted devices
- Enables accelerated time to market and lower cost of deployment and support

1.1.2 OSMP+ SDK

The OSMP+ Software Development Kit (SDK) is a development kit to create a media player on supported devices and supports the following functions:

- Playback of multiple video formats and audio formats to devices with or without native support
- Playback of live or video on demand (VOD) streaming, progressive download, and local media sources
- Easy integration with Digital Rights Management (DRM) decryption engines
- Audio post-processing for third party audio effect plug-ins and equalizers
- Video post-processing for Closed Captions and subtitles rendering

1.2 Supported operating system and web browser

The following is the operating system (OS) supported by OSMP+:

- Windows 7, Windows 8, Windows 8.1 with (ActiveX, NPAPI, PPAPI) plug-in using Windowed mode, and Windows 10 with (NPAPI, PPAPI) plug-in using Windowed mode
- Mac OS 10.6, Mac OS 10.8, Mac OS 10.9, and Mac OS 10.10 with (NPAPI, PPAPI) plug-in using Windowless mode
- Android 2.3.x, Android 3.x, Android 4.0.x, Android 4.1.x, Android 4.2.x, Android 4.3.x, Android 4.4.x, Android 5.0, and Android M (beta)
- iOS6, iOS 7, iOS8, and iOS9
- Windows Phone 8.1 (beta)

Table 1 lists the web browser supported by OSMP+:

Table 1. Supported web browser

Platform	**OS**	**Browser**					
		Safari[1]	Chrome (NPAPI)	FireFox (NPAPI)	IE[1] (ActiveX)	Chrome (PPAPI)	Windows 10/Edge
32-bit	Mac	Yes	Yes[2]	Yes[2]	N/A	No (default is 64bit)	N/A
64-bit	Mac	32-bit mode: Yes 64-bit mode: No	No	No	N/A	Yes	N/A
32-bit	Windows	N/A	Yes	Yes	Yes	Yes	No
64-bit	Windows	N/A	No	No	Yes[3]	Yes	No

1.3 Terminology

Table 2 describes the terminology used in this guide.

[1] Safari 64-bit and IE 64-bit support 32-bit plug-ins, but Chrome 64-bit and FireFox 64-bit do not support 32-bit plug-ins.

[2] Chrome will terminate the support of NPAPI in Sep, 2015. FireFox will add a warning message when running NPAPI.

[3] IE8/9 64-bit has two modes, and 32-bit is the default mode. IE10/11 64-bit default mode is 64-bit. VisualOn 32-bit plug-in can run on IE10/11 64-bit.

Table 2. Terminology

Term	Definition
Adaptive Playback	Adaptive Playback or Adaptive Streaming refers to streaming technologies such as HTTP Live Streaming, Microsoft Smooth Streaming, and MPEG-DASH. These technologies enable seamless playback of video content over the air as client devices change their behavior according to external factors such as bandwidth. Refer to *Multimedia* section on Android 4.4 APIs for more information.
Bitrate Adaptation	Bitrate Adaptation is used in adaptive streaming standards, which includes HTTP Live Streaming (HLS) and Dynamic Adaptive Streaming over HTTP (DASH). Smooth Bitrate Adaptation requires two decoding pipelines, but using two SurfaceView has limitations when two hardware decoders are required. We recommend using one TextureView class and one SurfaceView class for two video pipelines.
Closed Caption (CC)	CC is the processes of displaying text on a television, video screen, or other visual display to provide additional or interpretive information.
Hypertext Transfer Protocol (HTTP)	HTTP is an application protocol for distributed, collaborative, hypermedia information systems, as well as the foundation of data communication for the World Wide Web.
IOMX	API for accessing hardware decoders on Android devices. IOMX is for Android devices with version 4.0 and 4.1.
MediaCodec	API for accessing hardware decoders on Android devices. MediaCodec is for Android devices with version 4.2 and later.
Progressive Download (PD)	Progressive Download is the transfer of digital media files from a server to a client, typically using the HTTP protocol when being initiated from a computer or other mobile devices.
Real-time Transport Protocol (RTP)	RTP, sometime referred to as RTTP, defines a standardized packet format for delivering audio and video over IP networks.
SurfaceView	Default View class for displaying video contents on Android devices. Refer to SurfaceView for more information.
Transmission Control Protocol (TCP)	TCP is one of the core protocols of the Internet protocol (IP) suite. TCP provides reliable, ordered, and error-checked delivery of a stream of octets between programs running on computers connected to a local area network, intranet, or the public Internet. TCP resides at the transport layer.
User Datagram Protocol (UDP)	UDP uses a simple connectionless transmission model with a minimum of protocol mechanism. It has no handshaking dialogues, thus exposes any unreliability of the underlying network protocol to the user's program. There is no guarantee of delivery, ordering, or duplicate protection. UDP provides checksums for data integrity, and port numbers for addressing different

Term	Definition
	functions at the source and destination of the datagram.
TextureView	General View class for displaying video contents on Android devices. Comparing with SurfaceView, TextureView offers better capabilities for animation, scaling, and transformation. Refer to TextureView for more information.

1.4 Features and codecs

Combined with VisualOn's implementation of standard streaming protocols, OSMP+ enables seamless content delivery and playback across multiple platforms, including iOS, Android, Windows, Mac OS, and Windows Phone. OSMP+ uses the most advanced video and audio decoding technologies.

This section describes the OSMP+ features, feature specifications, and codecs supported by OSMP+.

1.4.1 Adaptive streaming

Adaptive streaming is a technique used in streaming multimedia over computer networks, which works by detecting a user's bandwidth and CPU capacity in real time and adjusting the quality of a video stream accordingly. It requires the use of an encoder which can encode a single source video at multiple bitrates. The player switches between streaming the different encodings depending on available resources. Adaptive streaming includes the following three types:

- HTTP Live Streaming (HLS)
- Microsoft Smooth Streaming
- Dynamic Adaptive Streaming over HTTP (DASH)

1.4.1.1 HTTP Live Streaming

HTTP Live Streaming, namely HLS, is an HTTP-based media streaming communications protocol implemented by Apple Inc. as part of their QuickTime, Safari, OS X, and iOS software.

HLS works by breaking the overall stream into a sequence of small HTTP-based file downloads. Each download loads one short chunk of an overall potentially unbounded transport stream. As the stream is played, the client may select from a number of different alternative streams containing the same material encoded at a variety of data rates, allowing the streaming session to adapt to the available data rate.

The following is the tags supported by the OSMP+ SDK. Refer to HTTP Live Streaming for detailed definitions about HLS tags.

- EXT-X-TARGETDURATION

- EXT-X-MEDIA-SEQUENCE
- EXT-X-KEY(URI)
- EXT-X-ENDLIST
- EXT-X-STREAM-INF
- EXT-X-KEY(IV)
- EXT-X-DISCONTINUITY
- EXT-X-VERSION
- EXT-X-PLAYLIST-TYPE
- EXT-X-BYTERANGE
- EXT-X-MEDIA
- EXT-X-I-FRAME-STREAM-INF
- EXT-X-STREAM-INF(AUDIO/VIDEO)
- EXT-X-MEDIA(SUBTITLED/FORCED/CHARACTERISTICS)
- EXT-X-STREAM-INF(SUBTITLES)
- EXT-X-MAP

OSMP+ supports the following media formats:

- MPEG2-TS
- MPEG-ES/AAC
- MPEG-ES/MP3
- ID3(MPEG-ES)
- ID3(MPEG2-TS)
- WebVTT

1.4.1.2 Microsoft Smooth Streaming

The Microsoft Smooth Streaming protocol provides a means of delivering media from servers to clients in a way that can be cached by standard HTTP cache proxies in the communication chain. Allowing standard HTTP cache proxies to respond to requests on behalf of the server increases the number of clients that can be served by a single server.

The following is the codec supported by Microsoft Smooth Streaming on the OSMP+ player:

- Video codec
  - H.264
  - WMV/VC1
- Audio codec
  - AAC

  - WMA

Microsoft Smooth Streaming supported features include:

- Request and response through HTTP
- XML parser used for the manifest file
- Audio and video fragment process mechanism
- MP4 fragment parser callback
- Playback of multi-period VoD content
- Playback of indexed content
- Playback of live content
- Playback of VoD subtitle

1.4.1.3 Dynamic Adaptive Streaming over HTTP

Dynamic Adaptive Streaming over HTTP (DASH), also known as MPEG-DASH, is an adaptive bitrate streaming technique that enables high quality streaming of media content over the Internet delivered from conventional HTTP web servers. Similar to Apple's HTTP Live Streaming (HLS) solution, MPEG-DASH works by breaking the content into a sequence of small HTTP-based file segments, each segment containing a short interval of playback time of a content that is potentially many hours in duration, such as a movie or the live broadcast of a sports event.

The content is made available at a variety of bitrates, for example, alternative segments encoded at different bit rates covering aligned short intervals of play back time are made available.

As the content is played back by an MPEG-DASH client, the client automatically selects from the alternatives the next segment to download and play based on current network conditions. The client selects the segment with the highest bitrate possible that can be downloaded in time for playback without causing stalls or re-buffering events. Therefore an MPEG-DASH client can seamlessly adapt to changing network conditions, and provide high quality playback without stalls or re-buffering events.

DASH supports the following video and audio codecs:

- Video codec
  - H.264
  - WMV/VC1
  - H.265
- Audio codec
  - AAC
  - WMA

DASH supported features include:

- Request and response through HTTP
- XML parser used for Media Presentation Description (MPD) file
- Audio and video fragment process mechanism
- MP4 fragment parser callback
- TS fragment parser callback
- Playback of multi-period VoD content
- Playback of indexed content
- Playback of PD content (one file)
- Playback of live content
- Playback of VoD TTML or SMPTE subtitle

DASH uses the ISO, IEC, and 23009 tags. The Media Presentation Description (MPD) document contains metadata required by a DASH client to construct appropriate HTTP URLs to access segments and to provide the streaming service to the user.

A Media Presentation consists of one or more Periods. Multi_Period is used for advertisement insertion.

- **AdaptationSet:** Each Period consists of one or more AdaptationSets. In general the *AdaptionSet* is divided into three types of stream: **Video**, **Audio**, and **Subtitle**.
- **ContentComponent:** Each AdaptationSet contains one or more media ContentComponents. The properties of each media ContentComponent are described by a *ContentComponent* element or may be described directly on the *AdaptationSet* element if only one media ContentComponent is present in *Adaptation Set*.

A Representation is one of the alternative choices of the complete set or subset of media content components comprising the media content during the defined Period.

- The **Segment** template is defined by the *SegmentTemplate* element. In this case, specific identifiers that are substituted by dynamic values assigned to Segments, to create a list of Segments.
- The **SegmentTimeline** element expresses the earliest presentation time and presentation duration (in units based on the @timescale attribute) for each Segment in the Representation.

1.4.2 Streaming

Streaming is multimedia that is constantly received by and presented to an end user while being delivered by a provider. This section describes the following topics:

- Progressive Download

- Real Time Streaming Protocol (RTSP)
- Windows Media Hypertext Transfer Protocol (WMHTTP)

1.4.2.1 Progressive Download

Progressive Download is the transfer of digital media files from a server to a client, typically using the HTTP protocol when being initiated from a computer. End user may start playback of the media before the download is complete.

OSMP+ downloads the media data from http server to local buffer, like local memory or files, manages the limited buffer efficiently, and then offers the required data to file parser as needed, as shown in Figure 2.

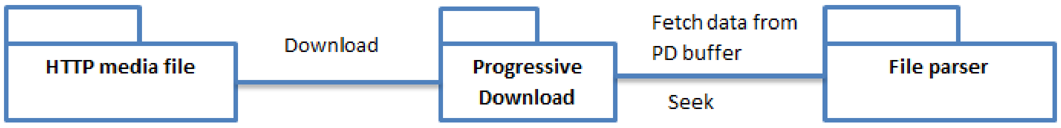


Figure 2. Progressive Download in OSMP+

To run Progressive Download, user needs to download at least 40 MB of the media file being played to the device's memory. If the required media data is not in the buffer, OSMP+ downloads data from the HTTP server. If the required media data is in the buffer, OSMP+ reads from local buffer and returns the data. Figure 3 demonstrates the roles of Progressive Download.

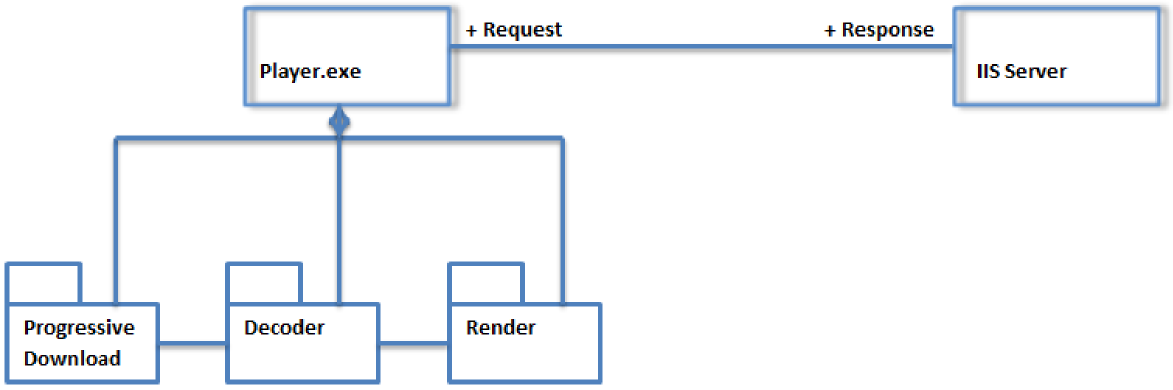


Figure 3. Progressive Download roles

1.4.2.2 Real Time Streaming Protocol

Real Time Streaming Protocol (RTSP) is a network control protocol designed for use in entertainment and communications systems to control streaming media servers. The protocol is used for establishing and controlling media sessions between end points. Clients of media servers issue VCR-style commands, such as play and pause, to facilitate realtime control of playback of media files from the server. The transmission of streaming data itself is not a task of the RTSP protocol. Most RTSP servers use the Real-time Transport Protocol (RTP) in conjunction with Real-time Control Protocol (RTCP) for media stream delivery. However some vendors implement proprietary transport protocols. The RTSP server software from RealNetworks, for example, also used RealNetworks' proprietary Real Data Transport (RDT).

The OSMP+ player supports the following audio and video codecs:

- Audio
  - MPEG4-GENERIC
  - MPEG4-LATM
  - AMR-WB+
  - AMR-WB
  - AMR-NB
  - QCELP
  - WMA
- Video
  - H.264
  - MPEG4
  - H.263
  - WMV

RTSP supports the following protocols:

- RFC 2326: Real Time Streaming Protocol (RTSP)
- RFC 2327: SDP: Session Description Protocol
- RFC 3550: RTP: A Transport Protocol for Real-Time Applications

The following is the supported RTP profile:

- RFC 3551: RTP Profile for Audio and Video Conferences with Minimal Control

The following is the supported RTP Payload types:

- RFC 3984: RTP Payload Format for H.264 Video
- RFC 3016: RTP Payload Format for MPEG-4 Audio/Visual Streams
- RFC 4629: RTP Payload Format for ITU-T Rec.H.263 Video
- RFC 3267: Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (ARM-WB) Audio Codecs
- RFC 4352: RTP Payload Format for the Extended Adaptive Multi-Rate Wideband (AMR-WB+) Audio Codec
- RFC 2658: RTP Payload Format for PureVoice (tm) Audio
- MS-RTSP: RTP Payload Format for ASF Streams

VisualOn

1.4.2.3 Windows Media HTTP

The Windows Media HTTP (WMHTTP) Streaming Protocol is a client/server–based protocol used to stream realtime data between a client (the receiver of streaming data) and a server (the sender of streaming data). OSMP+ supports the following WMHTTP features:

- Supports both Non-Pipelined mode and Pipelined mode.
  - **Non-Pipelined** mode: Mode of operation in which requests from the client must be sent on a TCP connection separate from the one being used by the server for streamingcontent to the client.
  - **Pipelined** mode: Mode of operation in which requests from the client can be sent on the same TCP connection being used by the server for streamingcontent to the client.
- Supports both Single File streaming and live streaming.
  - Single File streaming: See Single File Streaming for more information.
  - Live streaming: Content that is streamed while it is still being encoded by an encoder.
- Specifies a transmission rate.
- Specifies the amount of multimedia data in milliseconds, which the client requests the server to speed up
- KeepAlive request prevents the server from timing out.
- Submits statistics about the streamed content to the server.

See [MS-WMSP]: Windows Media HTTP Streaming Protocol for the detailed information about WMHTTP.

1.4.3 Local parsers

OSMP+ supports the following local parsers:

- Audio parser
- MP4 parser
- AVI parser
- FLV parser

1.4.3.1 Audio parser

Audio parser supports the following audio codecs:

- MP3
- AAC
- AC3
- AMR
- APE

- AU
- DTS-HD
- FLAC
- QCP
- WAV

1.4.3.2 MP4 parser

MP4 parser has no limitation on the maximum file size and supports the following specifications:

- ISO/IEC 14496-12
- ISO/IEC 14496-14
- ISO+IEC+23001-7-2012(CENC)

MP4 parser supports the following audio and video codecs:

- Audio
    - AAC
    - QCELP
    - MP3
    - AMR (NB, WB, and WB+)
    - EVRC
    - AC3
    - eAC3
    - ALAC
    - DTS-HD
    - MP4A
    - PCM
    - MULAW
    - ALAW
    - WMA
    - QCELP
- Video
    - H.264/AVC
    - H.263
    - MPEG4
    - H.265/HEVC

1.4.3.3 AVI parser

The maximum file size of AVI parser is 2G bytes. AVI parser supports OpenDML AVI File Format Extensions (version 1.02).

AVI parser supports the following audio and video codecs:

- Audio
  - AAC
  - QCELP
  - MP3
  - AMR (NB, WB, and WB+)
  - EVRC
  - AC3
  - eAC3
  - WMA
- Video
  - H.264/AVC
  - H.263
  - MPEG4
  - Motion JPEG
  - Xvid
  - DivX
  - WMV

1.4.3.4 FLV parser

FLV parser has no limitation on the maximum file size and supports the following audio and video codecs:

- Audio
  - AAC
  - FLV
- Video
  - AVC
  - Sorenson H.263
  - VP6

1.4.4 Codec

1.4.4.1 Video codec

OSMP+ supports the following video codecs:

- HEVC/H.265 (software codec)
- H.264 (software codec)

1.4.4.1.1 HEVC/H.265

The input format of this video codec is ISO/IEC23008-2 compliant raw video data, and the output format is YUV420 planar. The maximum supported video size for HEVC is 1080p.

HEVC/H.265 supports the following profiles:

- I,P,B frame
- CU size: 8x8 to 64x84
- PU partition: Symmetric
- TU partition: RQT
- TU size: DCT 4x4 to 32x32, DST 4x4
- Intra Prediction mode: all size of DC, Planner, 33 directions mode
- Quarter pixed motion compensation and weighted prediction
- Parallel processing tools deblocking, SAO
- Entropy coding: context-adaptive binary arithmetic coding (CABAC)

1.4.4.1.2 H.264

For H.264 codec, the output format is YUV420. The input formats include:

- 14496-10 AnnexB format
- 14496-15 single NALU format
- 14496-15 AVC format

The maximum supported video size for H.264 codec is 1080p. Table 3 lists the profiles supported by H.264 codec.

Table 3. Supported profiles

	Constrained Baseline	**Baseline**	**Extended**	**Main**	**High**	**Hi10P**	**Hi422P**	**Hi444PP**
B slices	No	No	Yes	Yes	Yes	Yes	Yes	Yes
SI and SP slices	No	No	Yes	No	No	No	No	No
Flexible macroblock ordering (FMO)	No	Yes	Yes	No	No	No	No	No

	Constrained Baseline	Baseline	Extended	Main	High	Hi10P	Hi422P	Hi444PP
Arbitrary slice ordering (ASO)	No	Yes	Yes	No	No	No	No	No
Redundant slices (RS)	No	Yes	Yes	No	No	No	No	No
Data partitioning	No	No	Yes	No	No	No	No	No
Interlaced coding (PicAFF, MBAFF)	No	No	Yes	Yes	Yes	Yes	Yes	Yes
CABAC entropy coding	No	No	No	Yes	Yes	Yes	Yes	Yes
8×8 vs. 4×4 transform adaptivity	No	No	No	No	Yes	Yes	Yes	Yes
Quantization scaling matrices	No	No	No	No	Yes	Yes	Yes	Yes
Separate C_b and C_r QP control	No	No	No	No	Yes	Yes	Yes	Yes
Monochrome (4:0:0)	No	No	No	No	Yes	Yes	Yes	Yes
Chroma formats	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0/4:2:2	4:2:0/4:2:2/4:4:4
Sample depths (bits)	8	8	8	8	8	8 to 10	8 to 10	8 to 14
Separate color plane coding	No	No	No	No	No	No	No	Yes
Predictive lossless coding	No	No	No	No	No	No	No	Yes

1.4.4.2 Audio codec

OSMP+ supports the following audio codecs:

- AAC

- AAC+
- eAAC+
- MP3

Note: Windows Phone supports AAC audio codec only.

1.4.4.2.1 AAC, AAC+, and eAAC+

This audio codec supports the following object types:

- AAC-LC
- AACLT
- HEAAC (SBR)
- EAAC+ (HE_PS)
- BSAC (ER_BSAC)

For this audio codec, the bitrate mode includes variable bitrate (VBR) and constant bitrate (CBR), and the output format is 16 bit PCM data. The input formats include:

- RAW data AAC
- ADTS AAC
- ADIF AAC
- LATM AAC

Table 4 lists the MPEG4 audio object details.

Table 4. MPEG4 audio object details

Object Type	**Object ID**	**Description**	**Support (Yes or No)**
NULL	0	NULL Object	Yes
AAC Main	1	AAC Main Object	No
AAC LC	2	AAC Low Complexity(LC) Object	No
AAC SSR	3	AAC Scalable Sampling Rate(SSR) Object	Yes
AAC LTP	4	AAC Long Term Predictor(LTP) Object	No
AAC SBR	5	AAC SBR object(HE_AAC)	No
AAC SCAL	6	AAC Scalable Object	Yes
TWIN VQ	7	TwinVQ Object	Yes

Object Type	Object ID	Description	Support (Yes or No)
CELP	8	CELP Object	Yes
HVXC	9	HVXC Object	Yes
RSVD 10	10	reserved	Yes
RSVD 11	11	reserved	Yes
TTSI	12	TTSI Object	Yes
MAIN SYNTH	13	Main Synthetic Object	Yes
WAV TAB SYNTH	14	Wavetable Synthesis	Yes
GEN MIDI	15	General MIDI Object	Yes
ALG SYNTH AUD_FX	16	Algorithmic Synthesis and Audio FX Object	Yes
ER AAC LC	17	Error Resilient(ER) AAC Low Complexity(LC) Object	Yes
RSVD 18	18	reserved	Yes
ER AAC LTP	19	Error Resilient(ER) AAC Long Term Predictor(LTP) Object	Yes
ER AAC SCAL	20	Error Resilient(ER) AAC Scalable Object	Yes
ER TWIN VQ	21	Error Resilient(ER) TwinVQ Object	Yes
ER BSAC	22	Error Resilient(ER) BSAC Object	No
ER AAC LD	23	Error Resilient(ER) AAC LD Object	Yes
ER CELP	24	Error Resilient(ER) CELP Object	Yes
ER HVXC	25	Error Resilient(ER) HVXC Object	Yes
ER HILN	26	Error Resilient(ER) HILN Object	Yes
ER PARA	27	Error Resilient(ER) Parametric Object	Yes
SSC	28	SSC Object	Yes
HE PS	29	AAC High Efficiency with Parametric Stereo coding (HE-	No

Object Type	Object ID	Description	Support (Yes or No)
		AAC v2, EAAC+)	
RSVD 30	30	reserved	Yes
RSVD 31	31	reserved	Yes
Layer 1	32	MPEG Layer 1 Object	No
Layer 2	33	MPEG Layer 2 Object	No
Layer 3	34	MPEG Layer 3 Object	No
DST	35	DST Object	Yes

1.4.4.2.2 MP3

Table 5 lists the supported MPEG audio layer.

Table 5. Supported MPEG audio layer

Decoder	Layer type
MPEG audio	• MPEG1 Layer1 • MPEG1 Layer2 • MPEG1 Layer3 • MPEG2 Layer1 • MPEG2 Layer2 • MPEG2 Layer3 • MPEG2.5 Layer1 • MPEG2.5 Layer2 • MPEG2.5 Layer3

1.4.4.2.3 DSAPlus decoder

OSMP+ uses DSAPlus decoder to decode and post-process the Dolby Digital Plus contents, as well as supporting the sample rates, bitrates, and output modes for each of the audio formats. For detailed information about DSAPlus decoder, visit Dolby's official website.

1.4.4.2.4 Audio speed decoder

The following is the output of Audio speed decoder:

- Support Mono and two channels output
- 16 bits, PCM
- -3x to +3x audio speed adjustment

1.4.5 Bitrate Adaptation

Bitrate Adaptation is a process to effectively deliver streaming contents to achieve the best possible playback quality under various environmental conditions. OSMP+ supports Bitrate Adaptation for HTTP-based adaptive streaming protocols, namely HLS, Smooth Streaming, and DASH. OSMP+ takes network congestion, device capability, device loading, and several other factors into consideration to determine an optimal video bitrate to be played. Adaption allows seamless transitions among different video qualities without interrupting the playback.

In additional to automatic Bitrate Adaptation, OSMP+ supports many configuration options to perfect adaption behaviors. These configuration options include:

- Initial (starting) bitrate
- Start buffering time
- Playback re-buffering time
- Maximum buffering time
- Minimum bitrate
- Maximum bitrate
- CPU based adaption
- Device capability configuration

Combining these options with the automatic Bitrate Adaptation technology provides a flexible solution to fit a wide range of requirements.

1.4.6 Multiple audio tracks

In adaptive streaming, if multiple audio tracks are advertised in the manifest file, OSMP+ extracts the information and provides a mechanism to select a specific audio track to be played. OSMP+ also provides APIs to query a number of properties related to audio, such as number of alternative audio available, codec type, sample rate, number of channels, language, and so on.

OSMP+ supports the feature to dynamically change audio tracks during playback. In case that audio tracks of multiple languages are available, to simplify the content preparation for different geographic regions, preferred languages can be specified so that the playback starts with the most favorable language.

1.4.7 Hardware acceleration

1.4.7.1 Hardware acceleration for Android

This section describes the hardware integration implemented in OSMP+ on the Android platform.

1.4.7.1.1 Hardware acceleration introduction

1.4.7.1.1.1 APIs for accessing hardware decoders

The Android operating system provides the following two types of API to access hardware decoders:

- **MediaCodec**: MediaCodec is the official Android API. This API was initially released for Android version 4.1.2 and is supported on most Android devices with version 4.2 and later.
- **IOMX**: IOMX is not an official Android API. This API was initially released for Android version 2.1. This API needs device tools to compile the supported libraries, and is commonly used to access the hardware decoders on Android devices with version 4.0 and version 4.1.

1.4.7.1.1.2 OSMP+ codec options

The following are the available video decoder types that can be set by the application by using the *setDecoderType()* method:

- Software
- IOMX
- MediaCodec
- AutoHW

For **AutoHW**, OSMP+ selects the API based on the following default rules. OSMP+ will automatically fall back on other options if the selected option is not working.

- **MediaCodec** for Android devices with version 4.2 and later.
- **IOMX** for Android devices with version 4.0 and 4.1.
- **Software** for Android devices with version older than 4.0.

OSMP+ provides a white list file that overrides the decoder type selection for specific devices when the IOMX, MediaCodec, or AutoHW types are set. See 1.4.7.1.2 White List for more information.

1.4.7.1.1.3 Smooth Bitrate Adaptation

A consequence of Bitrate Adaptation is to dynamically switch between streams having different bitrates depending on network conditions. Sometimes the streams also have different resolutions.

Many hardware video decoders do not handle bitrate switching or resolution switching smoothly. Artifacts such as black frames or pause may impact the viewer’s experience. Performance of Bitrate Adaptation switching varies depending on device configurations.

To enhance viewing experience, OSMP+ supports Smooth Bitrate Adaptation. OSMP+ Smooth Bitrate Adaptation minimizes switching artifacts by using several resources such as a second view, TextureView. Smooth Bitrate Adaptation is enabled by using *enableVOAdaptivePlayback(true)*.

The behavior of Smooth Bitrate Adaptation can be fine-tuned for specific devices configurations. See 1.4.7.1.2 White List for more information.

1.4.7.1.2 White List

OSMP+ offers the option to change the default audio and video hardware decoding settings for specific device configurations. The list of device configurations (also known as the white list) is saved in the *device.xml* file under the **assets** directory of your Android project. The *device.xml* file supports the following fields:

Note: Use the Android adb command, *adb shell getprop*, to get the information about **Model**, **Release**, and **Platform**.

- **Model**: product model information
- **Release**: Android version
- **Platform**: platform name
- **CodecSolution**: IOMX, MediaCodec, or None
- **TextureView**: *No* is the only interpreted value, other values are ignored.

Figure 4 shows an example of a white listed device configuration. For more information about the white list, refer to the comment section of the *device.xml* file shipped with the VisualOn OSMP+ SDK package. The *device.xml* file can be found under **Android > SamplePlayer > assets**. You can edit this xml file to add or remove device configurations.

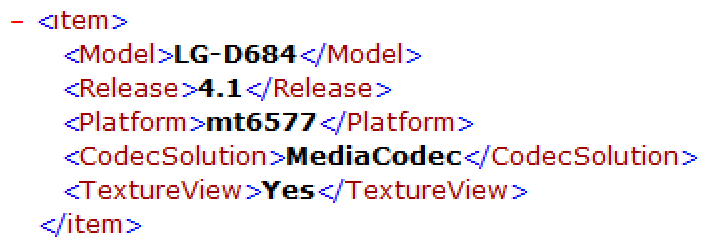

```
- <item>
    <Model>LG-D684</Model>
    <Release>4.1</Release>
    <Platform>mt6577</Platform>
    <CodecSolution>MediaCodec</CodecSolution>
    <TextureView>Yes</TextureView>
  </item>
```

Figure 4. Example of white list

OSMP+ SDK compares the values of the white list fields to the values of the device on which it is running.

- If the running device values have a match in the white list, OSMP+ changes the hardware decoding settings according to the white list settings.
- If the running device values do not match, OSMP+ uses the current hardware decoding settings.
- The interpretation of the white list is affected by the values of *setDecoderType()* and *enableVOAdaptivePlayback().* For more information about the white list mechanisms, refer to the *comment* section of *device.xml*.

1.4.7.1.3 List of tested devices

The following lists the devices tested for hardware decoding with OSMP+ as of the document publication date, which might not be updated at this time. Contact VisualOn sales for the latest list of tested devices for hardware integration with OSMP+.

- Amazon
  - Kindle HDX 7/4.3/800
  - Kindle FireTV/4.2.2/MSM8960
- ASUS
  - ASUS ME371/4.1/Intel Atom
- Samsung
  - Galaxy S5/4.4.2/MSM8974
  - Galaxy S3/4.0.4/MSM8960
  - Galaxy Note/4.0.3/8660
  - Galaxy S3/4.1.1/Exynos4
  - Galaxy Tab 10.1/4.1.2/Exynos4
  - Galaxy Tab3/4.1/ Exynos4
  - Galaxy Note II/4.1.2/Exynos4
  - Galaxy S4 (I9500)/4.2.2/Exynos5
  - Galaxy S4 (I9500)/4.4/ Exynos5
  - Galaxy Note 3/4.3/Exynos5
  - Samsung Tab4/4.4.2/PXA869
- Google
  - Nexus 4/4.4/APQ8064
  - Nexus 5/4.4.3/MSM8974
  - Nexus 5/Android L/MSM8974
  - Nexus 5/Android L/ MSM8974
  - Nexus 6/Android L/ MSM8974
  - Nexus 7/4.3/Tegra3
  - Nexus 7/4.4.2/Tegra3
  - Nexus 7 gen2/4.4.2/S4Pro
  - Nexus 7 gen2/Android L/MSM8960
  - Nexus 9/Android L/Tegra K1
  - Nexus 10/4.2.2/Exynos5
- HTC

    - HTC ONE/4.4.2/APQ8064T
    - HTC ONE X/4.0/Tegra3
- Huawei
    - Huawei P6/4.4.2/Hi6620
- Lenovo
    - Lenovo K900/4.2.2/Intel Atom
- LG
    - LG G3/4.4/MSM8975
    - LG G2/4.2.2/MSM8974
    - LG-V500/4.2.2/APQ8064
- Moto
    - Moto MB886/4.0.4/MSM8960
    - Moto Droid Razr/4.4.2/MSM8960
- Nubia
    - Nubia/4.2.2/MSM8974
- Nvidia
    - SHIELD P2450/4.3/Tegra4
    - SHIELD Android TV/5.0/Tegra X1
- Sony
    - Sony Z Ultra/4.2.2/MSM8974
    - Sony Z1/4.4.4/MSM8974
    - Sony Z1 compact/4.4.2/MSM8974
    - Sony Tablet Z/4.4.2/APQ8064
- Teclast
    - Teclast Tpad/4.1.1/Rockchip

1.4.7.1.4 UI limitation using TextureView

Dual decoders and dual surfaces are used to enable Smooth Bitrate Adaptation when selecting video hardware decoding. Dual surfaces include SurfaceView and TextureView.

When using TextureView surfaces, some UI designs might impact overall performance. OSMP+ optimizes Google’s default widget, for example TextView, to reduce the impact on UI performance and subtitle display.

See 1.4.7.1.5 UI optimization using TextureView for the detailed recommendations with respect to suggested UI optimizations.

1.4.7.1.5 UI optimization using TextureView

The following are recommendations to optimize UI performance when enabling Smooth Bitrate Adaptation through the use a TextureView.

- Avoid complicated layout background settings. For example, use *android:theme="@android:style/Theme.Black.NoTitleBar.Fullscreen"* for full screen playback.
- Reduce the number of overlay on top of the video.
- OSMP+ optimizes Google's default widget such as TextView to improve UI performance and subtitle display. Select these optimized UI widgets instead of the standard Android widgets. For example, use *com.visualon.widget.TextViewPlus* instead of the default TextView to show video duration and playback position.
- Use the *adb shell dumpsys gfxinfo* command to check video rendering time and monitor performance.

Refer to the following websites for general recommendation on how to optimize the UI layout:

- Analyzing Display and Performance
- Android Performance Case Study
- Improving Layout Performance
- Improving Your Code with lint
- Optimizing Your UI
- Using the Dev Tools App

1.4.7.1.6 Known limitations

The following are known issues identified by VisualOn when using video hardware decoding with OSMP+.

- 1080p video contents cannot be played simultaneously on two players for Nexus 7 and Nexus 7 Gen2 that have Android 4.4 or Android 5.0 installed.

1.4.7.2 Hardware acceleration for iOS

To perform the hardware acceleration on the iOS platform, the source must satisfy the following:

Note: OSMP+ does not support PCM, as PCM is not supported in HLS native player.

- Regular sources meeting the spec of all kinds of streaming
- HTTP Live Streaming
- Smooth Streaming (H.264 and AAC)
- Local MP4 (H.264 and AAC)

- Progressive Download (MP4, H.264, and AAC)

Hardware acceleration supports the following codec types:

- AAC, HE-AAC
- H.264 (Baseline, High Profile, and Main Profile)

Hardware acceleration supports iOS 4.0 and later, and is available on the following devices:

- iPhone 4, iPhone 4S, iPhone 5, and above
- iPad, iPad2, new iPad, iPad 4, and above
- iPod Touch 4th, iPod Touch 5th, and above

The following are the modules related to hardware acceleration:

- *TS muxer (libvoTsFW.a)*
- *Video parser (libvoVideoParser.a)*
- *AVFoundation.framework*
- *CoreMedia.framework*
- *AudioToolbox.framework*

1.4.8 Closed Captions

OSMP+ supports Closed Captions based on specification CEA-608 (also known as Line 21) and specification CEA-708-D (also known as EIA-708) per ATSC A/65.

CEA608 was developed in the 1970s for carrying 960bps captioning data and services in the VBI (Vertical Blanking Interval) of NTSC video, an analogue composite video format. In latter day digital component version of NTSC, this analogue waveform is digitally encoded and embedded in Line 21 of the Video.

CEA-708-D defines DTV Closed Captioning (DTVCC) and provides specifications and guidelines for caption service providers, distributors of television signals, decoder and encoder manufacturers, DTV receiver manufacturers, and DTV signal processing equipment manufacturers.

OSMP+ supports the following Closed Captions features:

- Text
- Horizontal Justification (Left, Right, and Center)
- Vertical Justification (Top, Bottom, and Center)
- Print Direction (Left-Right, Right-Left, Top-Bottom, and Bottom-Top)
- Text Wrap
- Scroll Direction (Left-Right, Right-Left, Top-Bottom, and Bottom-Top)

- Rectangle Border (None, Raised, Depressed, Uniform, Shadow_Left, and Shadow_Right)
- Rectangle Border Color
- Rectangle Border Fill Color
- Rectangle Display Effect (Snap, Fade, and Wipe)
- Rectangle Display Effect Direction (LR, RL, TB, and BT)
- Rectangle Display Effect Speed
- Font Size (Standard, Small, and Large)
- Font Color (RGBA, 32bit)

OSMP+ supports the following Closed Captions font styles:

- Default (undefined)
- Monospaced with serifs
- Proportionally spaced with serifs
- Monospaced without serifs
- Proportionally spaced without serifs
- Casual font type
- Cursive font type
- Small capitals

OSMP+ supports the following Closed Captions text tags:

- Dialog
- Source_speaker_ID
- Electronically_reproduced_voice
- Dialog_language_other_than_primary
- Dialog_Voiceover
- Dialog_Audible_Translation
- Dialog_Subtitle_Translation
- Dialog_Voice_quality_description
- Dialog_Song_Lyrics
- Dialog_Sound_effect_description
- Dialog_Musical_score_description
- Dialog_Expletive
- Dialog_Text_not_to_be_displayed

OSMP+ provides APIs for the application to control attributes of Closed Captions text, including:

- Text size
- Text color and opacity
- Text effects, for example, italic, underline, bold
- Background color and opacity
- Window color and opacity
- Font selection

1.4.9 Subtitles

Refer to Timed Text Markup Language 1 (TTML1) for a description of text-based subtitles.

OSMP+ supports the following subtitle formats:

- TTML
- SMPTE
- SAMI
- SRT
- WEBVTT

OSMP+ supports the following encoding formats:

- EUC-KR
- windows-949
- GB2312
- UTF-8
- ISO-8859-1
- ISO-8859-2
- ISO-8859-3
- ISO-8859-4
- ISO-8859-5
- ISO-8859-6 (Android and iOS)
- ISO-8859-7 (Android and iOS)
- ISO-8859-8 (Android and iOS)
- ISO-8859-9
- ISO-8859-10 (Android, Windows, and Mac OS)
- ISO-8859-11 (Android)
- ISO-8859-13
- ISO-8859-14 (Windows and Mac OS)

- ISO-8859-15
- ISO-8859-16 (Windows and Mac OS)
- Shift_JIS
- Unicode

OSMP+ supports the following subtitles features:

- Text
- Horizontal Justification (Left, Right, and Center)
- Vertical Justification (Top, Bottom, and Center)
- Print Direction (Left-Right, Right-Left, Top-Bottom, and Bottom-Top)
- Text Wrap
- Scroll Direction (Left-Right, Right-Left, Top-Bottom, and Bottom-Top)
- Rectangle Border (None, Raised, Depressed, Uniform, Shadow_Left, and Shadow_Right)
- Rectangle Border Color
- Rectangle Border Fill Color
- Rectangle Display Effect (Snap, Fade, and Wipe)
- Rectangle Display Effect Direction (LR, RL, TB, and BT)
- Rectangle Display Effect Speed
- Font Size (Standard, Small, and Large)
- Font Color (RGBA, 32bit)

OSMP+ supports the following subtitle font styles:

- Default (undefined)
- Monospaced with serifs
- Proportionally spaced with serifs
- Monospaced without serifs
- Proportionally spaced without serifs
- Casual font type
- Cursive font type
- Small capitals

OSMP+ provides APIs for the application to control attributes of subtitles text, including:

- Text size
- Text color and opacity
- Text effects, for example, italic, underline, bold

- Background color and opacity
- Window color and opacity
- Font selection

1.4.10 FCC compliance

FCC compliance supports Timed Text Markup Language (TTML) and WebVtt. Refer to Notice for the FCC requirements.

1.4.10.1 Presentation

All apparatus must implement captioning so that the caption text may appear within one or separate caption windows, and supports the following modes:

- Text that appears all at once (pop-up)
- Text that scrolls up as new text appears (rollup)
- Text where each new letter or word appears as it arrives (paint-on).

OSMP+ has no API to support the presentation mode; instead OSMP+ gets this data from streaming based on specification CEA-608 and specification CEA-708.

1.4.10.2 Character Size

All apparatus must implement captioning so that users are provided with the ability to vary the size of captioned text and needs provide a range of sizes from 50% of the default character size to 200% of the default character size.

To set the FCC character size, use *setSubtitleFontSizeScale* API.

1.4.10.3 Character Color

All apparatus must implement captioning so that characters may appear in the 64 colors defined in CEA–708, and users are provided with the ability to override the authored color for characters and select from a palette of at least 8 colors, including white, black, red, green, blue, yellow, magenta, and cyan.

To set the FCC character color, use *setSubtitleFontColor* API.

1.4.10.4 Character Opacity

All apparatus must implement captioning so that users are provided with the ability to vary the opacity of captioned text and select between opaque and semitransparent opacities.

To set the FCC character opacity, use *setSubtitleFontOpacity* API.

1.4.10.5 Fonts

All apparatus must implement captioning so that fonts are available to implement the eight fonts required by CEA–708. Users must be provided with the ability to assign the fonts included on their apparatus as the default font for each of the following eight styles:

- Default (undefined)
- Monospaced with serifs (similar to Courier)
- Proportionally spaced with serifs (similar to Times New Roman)
- Monospaced without serifs (similar to Helvetica Monospaced)
- Proportionally spaced without serifs (similar to Arial and Swiss)
- Casual font type (similar to Dom and Impress)
- Cursive font type (similar to Coronet and Marigold)
- Small capitals (similar to Engravers Gothic)

To set the FCC font, use *setSubtitleFontName* API.

1.4.10.6 Caption Background Color and Opacity

All apparatus must implement captioning so that the caption background may appear in the 64 colors defined in CEA–708, and users are provided with the ability to override the authored color for the caption background and select from a palette of at least 8 colors, including white, black, red, green, blue, yellow, magenta, and cyan.

To set the FCC caption background, use *setSubtitleFontBackgroundColor* API.

All apparatus must implement captioning so that users are provided with the ability to vary the opacity of the caption background and select within opaque, semitransparent, and transparent background opacities.

To set the FCC caption opacity, use *setSubtitleFontBackgroundOpacity* API.

1.4.10.7 Character Edge Attributes

All apparatus must implement captioning so that character edge attributes may appear, and users are provided the ability to select character edge attributes that include:

- No edge attribute
- Raised edges
- Depressed edges
- Uniform edges
- Drop shadowed edges: left drop shadow/right drop shadow

To set the FCC character edge attributes, use *setSubtitleFontEdgeType* API.

1.4.10.8 Caption Window Color and Opacity

All apparatus must implement captioning so that the caption window color may appear in the 64 colors defined in CEA–708, and users are provided with the ability to override the authored color for the caption window and select from a palette of at least 8 colors, including white, black, red, green, blue, yellow, magenta, and cyan.

To set the FCC caption window color, use *setSubtitleWindowBackgroundColor* API.

All apparatus must implement captioning so that users are provided with the ability to vary the opacity of the caption window and select within opaque, semi-transparent, and transparent background opacities.

To set the FCC caption window opacity, use *setSubtitleWindowBackgroundOpacity* API.

1.4.10.9 Language

All apparatus must implement the ability to select between caption tracks in additional languages when such tracks are presented and are provided the ability for the user to select simplified or reduced captions when such captions are available and identify such a caption track as ‘‘easy reader.’’

To set the FCC language, use *setSubtitlePath* API.

1.4.10.10 Preview and Setting Retention

All apparatus must provide the ability for the user to preview default and user selection of the caption features required by this section, and must retain such settings as the default caption configuration until the caption configuration is changed by the user.

To set the FCC preview and retention, use *previewSubtitle* API.

1.4.11 Zoom modes

OSMP+ supports the following zoom modes to control rendering of video:

- Letterbox
- Pan & Scan
- Fit to Window
- Original Size
- Zoom In

1.4.11.1 Letterbox mode

The letterbox mode matches the height or width of the video to the display area while keeping its original aspect ratio depending on the orientation and size of the display area. See Letterboxing (filming) for more information about the letterbox mode.

Figure 5 shows an example where the height of a 4:3 video is changed to match the height of a 16:9 display area thus resulting in black bar on the left and right sides.

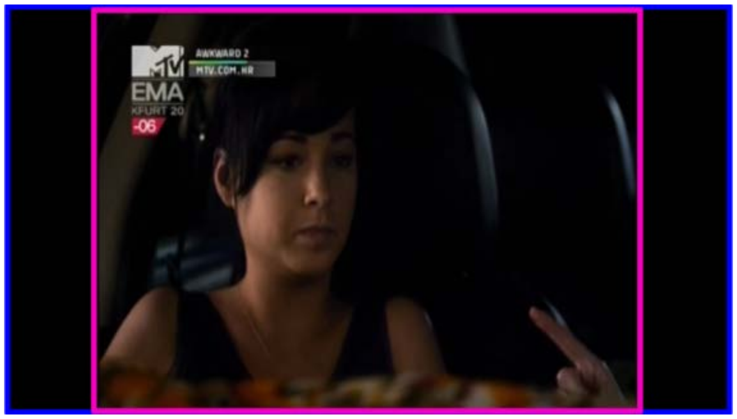


Figure 5. Sample 1: letterbox

Figure 6 shows an example where the width of the video is adjusted to match a rotated 16:9 display area resulting in black bars above and below the centered video content.

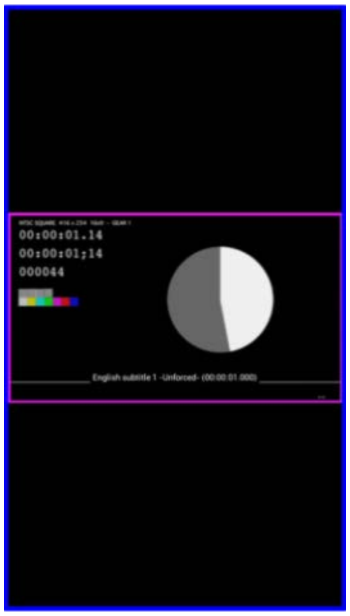


Figure 6. Sample 2: Letterbox

1.4.11.2 Pan & Scan mode

The Pan & Scan mode matches the height or width of the video to the display area while keeping its original aspect ratio depending on the orientation and size of the display area. As opposed to the letterbox mode, the Pan & Scan mode removes any black bar from the display area thus effectively cutting part of the video content. See Pan and scan for more information about the Pan & Scan mode.

A 4:3 video rendered on a 16:9 display area of matching orientation is scaled so that its width matches the width of the display area while keeping its aspect ratio. This leads to the video feeling the entire display area and part of the video being cut at the top and bottom, as shown in Figure 7.

Note: Pan & Scan mode is not supported on the Windows plug-in and Mac OS plug-in.

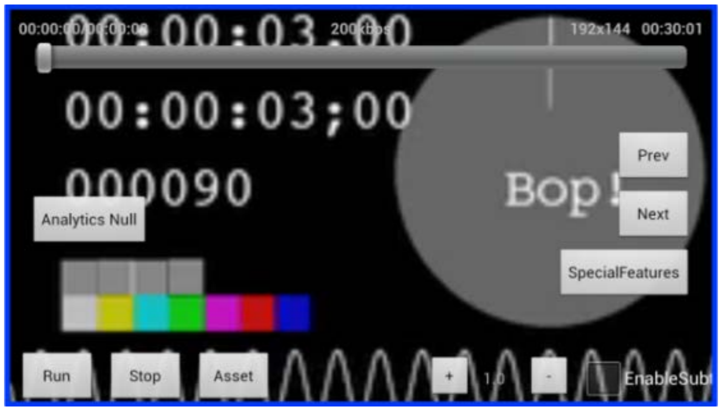


Figure 7. Sample 1: Pan & Scan

Figure 8 shows and example of 4:3 video rendered in Pan & Scan mode in a 16:9 display area of different orientation thus resulting in the height of the video matching the height of the display area and video being cut on the left and right sides.

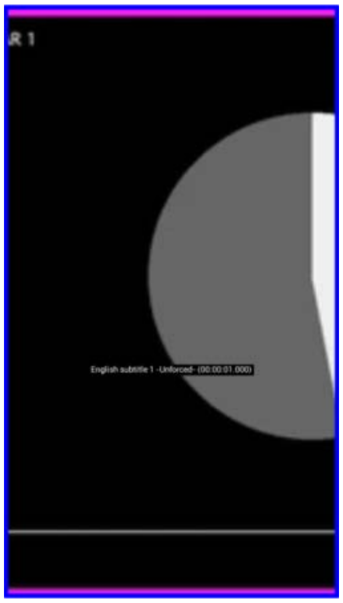


Figure 8. Sample 2: Pan & Scan

1.4.11.3 Fit to Window mode

Regardless of its aspect ratio and size, the video is resized to fill the entire display area, as shown in Figure 9.

Note: Fit to Window mode is not supported on the Windows plug-in and Mac OS plug-in.

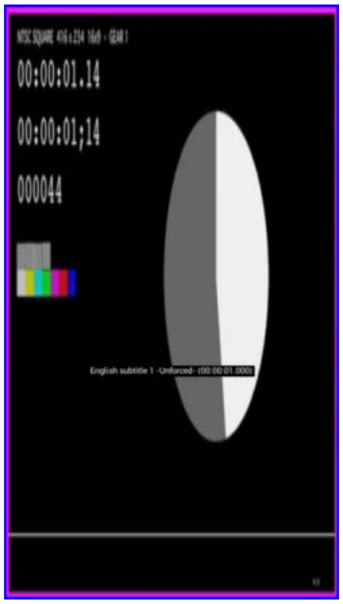

NTSC SQUARE 416 x 234 16x9 - GEAR 1
00:00:01.14
00:00:01;14
000044
English subtitle 1 -Unforced- (00:00:01.000)

Figure 9. Example of Fit to Window

1.4.11.4 Original Size mode

The Original Size mode renders the video with its original size and aspect ratio regardless of the display area size and aspect ratio, as shown in Figure 10.

Note: Original Size mode is not supported on the Windows plug-in and Mac OS plug-in.

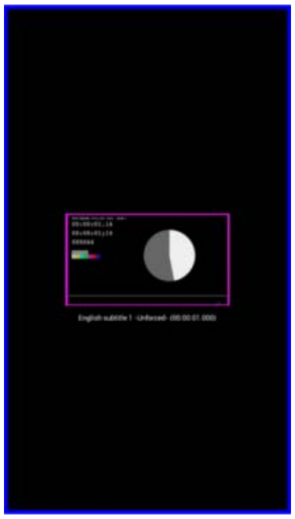


Figure 10. Example of Original Size

1.4.11.5 Zoom In mode

The Zoom In mode specifies a rectangle area within the video to fill the display area while preserving the original video aspect ratio, as shown in Figure 11. The specified rectangle boundaries cannot be outside the video actual size.

Note: Zoom In mode is not supported on the Windows plug-in and Mac OS plug-in.

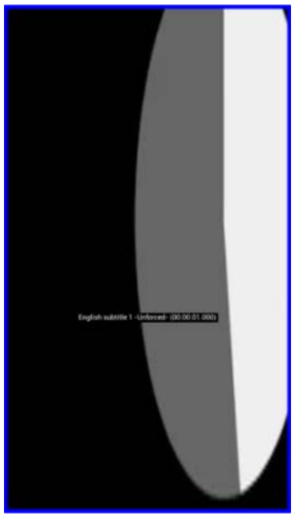


Figure 11. Example of Zoom In

1.4.12 Download Manager

Download Manager is an OSMP+ feature that is used to download content for offline viewing.

Note: Download Manager is only available on Android and iOS.

The Download Manager module supports the following features:

- Download content to local storage while playing with playback or download only. If Download Manager is turned on, the **Select Asset** dialog will prompt when starting the playback.
- Select video bitrate, audio track, and subtitle track to be downloaded. If the Download Manager feature is turned on, you will be prompted to select only one video bitrate, audio track, or subtitle track for downloading in the **Select Asset** dialog.
- Check the DRM key expiration.
- Report and display download progress.
- Resume download from the last break point.
- Report errors due to connection failure or full device storage.
- Start, stop, pause, resume, and delete download from the application.

The current OSMP+ Download Manager feature only supports HLS VoD streaming in combination with DRMs. In addition, only the Android and iOS platforms are currently supported. For more information about Download Manager, refer to *OnStream MediaPlayer+.Download Manager Integration Guide for Android.pdf* and *OnStream MediaPlayer+.Download Manager Integration Guide for iOS.pdf* in your installation package.

1.4.13 DRM

Digital Rights Management (DRM) is a class of technologies that are used by hardware manufacturers, publishers, copyright holders, and individuals with the intent to control the use of digital content (video, audio, and so on) and devices after sale.

VisualOn’s OSMP+ is agnostic to any DRM technology and can integrate with all commercial and proprietary DRM engines to enable playback of protected content on mobile devices. The DRM component allows or prevents playback of digital assets on mobile devices while OSMP+ handles the steps related to high-quality playback of those assets. DRM engine is placed underneath the application, and DRM Adaptor is used to translate the private DRM interface into a common interface to be used by other engines, as shown in Figure 12.

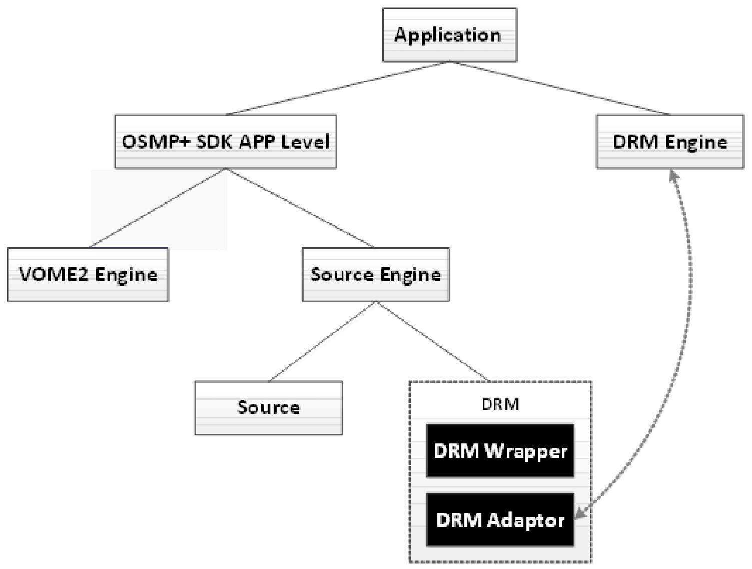


Figure 12. DRM workflow

1.4.14 AirPlay on iOS

AirPlay is a proprietary protocol stack/suite that allows wireless streaming between devices of audio, video, device screens, and photos, together with related metadata.

With AirPlay enabled on iOS, you can play/stop, pause, suspend/resume, and seek the content, as well as controlling the content volum through buttons. The OSMP+ SDK supports the following two AirPlay modes on iOS platform.

- AirPlay Mirroring: enables the output of the iOS device screen to an external display. If Mirroring is detected to be in use on the iOS device, the OSMP+ player shows the content on both iOS device screen and the external display.
- AirPlay Streaming Export: allows content to be re-directed from iOS device to Apple TV.

The OSMP+ AirPlay Streaming Export on iOS is available with limitations:

Note: AirPlay Streaming Export is only available when using AVPlayer hardware engine.

- Operating System (OS): iOS
- Platform: iOS 4.3 or later, iTunes 10.2 or later
- Streaming protocol: Live and VOD for HLS and Smooth Streaming
- DRM: PlayReady DRM
- Video decoder: H.264
- Audio decoder: AAC

Chapter 2 Release Package Structure

This chapter presents the following topics:

2.1 Android package

This section describes how the files are organized in the Android release package:

```
├── Android
    ├── Bin: sample player apks
    ├── Doc: Integration documentation, API reference, and lab projects
    ├── Jar: Jar libraries
        └── debug
    ├── Libs: ARM NDK and Intel x86 libraries
        └── debug
            ├── armeabi
            └── x86
        └── release
            ├── armeabi
            └── x86
    ├── License: temporary license file
    └── SamplePlayer: source codes for the sample player
```

2.2 iOS package

This section describes how the files are organized in the iOS release package:

```
├── iOS
    ├── Bin: sample player APIs
    ├── Doc: Integration documentation, API reference, and lab projects
    ├── Include: header files
    ├── Libs: libraries
        └──debug
    ├── License: temporary license file
    ├── SamplePlayer: source codes for the sample player
    └── Tool (optional for some DRM packages)
```

2.3 Plug-in package

This section describes how the files are organized in the Windows plug-in and Mac OS plug-in release packages:

```
└── Plugin
    ├── Bin: sample player installer
        ├── MacOS
        └── Win32
    ├── Doc: Integration documentations and API reference
    ├── Include: included .js files
    ├── Libs: .dll libraries
        ├── MacOS
            └── x86
              └── debug
        └── Win32
            ├── debug
            ├── Resource
            └── ThirdParty
    ├── License: temporary license file
        ├── MacOS
        └── Win32
    ├── Project: necessary projects to make the installer
        ├── MacOS
        └── Win32
    ├── Sample: source codes for sample scripts
        ├── guardian : DOM Tree test page
        └── SamplePlayer: sample player
    └── Tools: miscellaneous to make the installer
        ├── MacOS
        └── Win32
```

2.4 Chrome PPAPI package

This section describes how the files are organized in the Chrome PPAPI release packages:

```
└── PluginPNaCl
    ├── Bin: sample player installer
```

```
        ├── MacOS
        ├── PNaCl
        └── Win32
    ├── Doc: Integration documentations and API reference
    ├── Include: included .js files
    ├── Libs: .dll libraries
        ├── MacOS
            └── x86
                └── debug
        └── Win32
            └── debug
    ├── License: temporary license file
        ├── MacOS
        └── Win32
    ├── Project: necessary projects to make the installer
        ├── MacOS
        ├── Source
        └── Win32
    ├── Sample: source codes for sample scripts
        ├── HelperExtension
        └── SamplePlayer
    └── Tools: miscellaneous to make the installer
        ├── Installer
        └── SimpleHttpd
```

2.5 Windows Phone package

This section describes how the files are organized in the Windows Phone release package:

```
├── WinPhone
    ├── Bin: sample player APIs
    ├── Doc: Integration documentation, API reference, and lab projects
    ├── Include: header files
    ├── Libs: libraries
        └──debug
```

```
├── License: temporary license file
└── SamplePlayer: source codes for the sample player
```

Chapter 3 Sample Player

This chapter presents the following topics:

3.1 Android

3.1.1 Installation

This section describes the installation of the Sample Player application included with the SDK on Android.

We recommend that you install the Sample Player application with the latest Eclipse, which has the ADT extension and JRE 1.6.0 or higher versions. Refer to Installing the Eclipse Plugin for instructions about how to install ADT extension in Eclipse. See 1.2 Supported operating system and web browser for the Android versions supported by OSMP+.

3.1.1.1 Importing SamplePlayer project into Eclipse

To import the SamplePlayer project:

1. From **Eclipse IDE**, select **File > Import…**. The **Import** dialog appears, as shown in Figure 13.

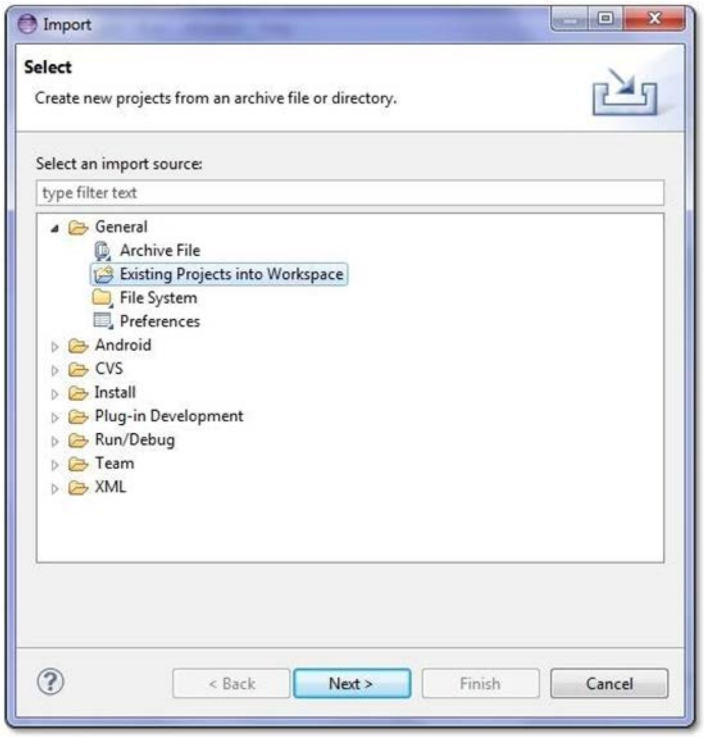


Figure 13. Eclipse IDE Import Dialog

2. In the **Import** dialog, select **General**, select **Existing Projects into Workspace**, and then click **Next**.

3. Select **Select root directory,** and then type or browse to *<SDK_INSTALL_DIR>\Android\SamplePlayer*. Under **Projects**, ensure that *<SamplePlayer project>* is selected, and then click **Finish**, as shown in Figure 14.

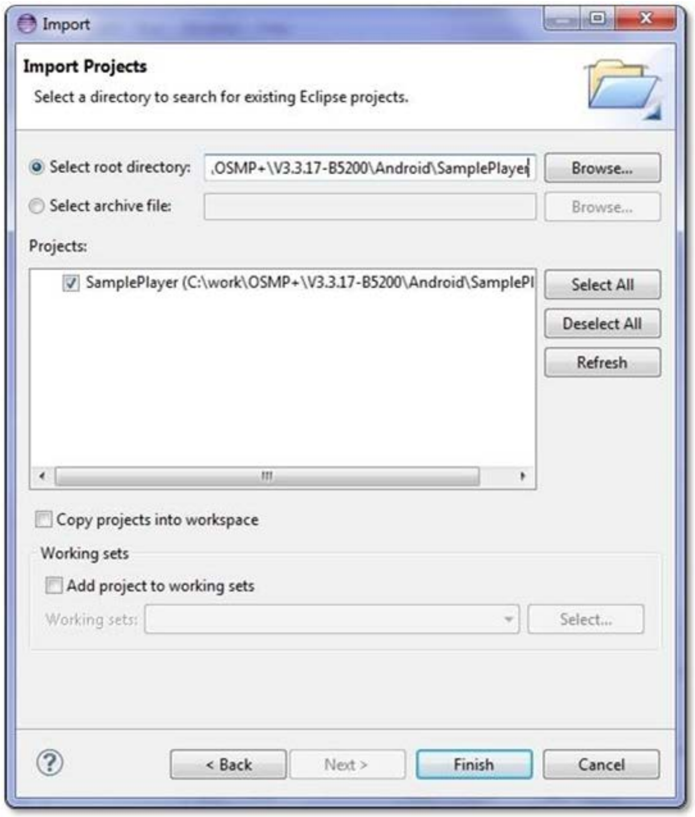

Import
Import Projects
Select a directory to search for existing Eclipse projects.
Select root directory: ,OSMP+\V3.3.17-B5200\Android\SamplePlayer
Browse...
Select archive file:
Browse...
Projects:
SamplePlayer (C:\work\OSMP+\V3.3.17-B5200\Android\SamplePl
Select All
Deselect All
Refresh
Copy projects into workspace
Working sets
Add project to working sets
Working sets:
Select...
< Back
Next >
Finish
Cancel

Figure 14. Select the SamplePlayer project

4. An Android project named **SamplePlayer** is imported into Eclipse, and appears in **Package Explorer**, as shown in Figure 15.

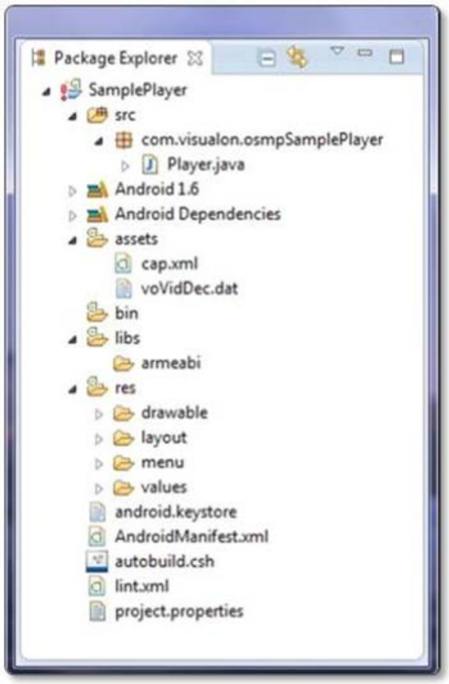


Figure 15. Eclipse IDE Package Explorer

If the imported project contains multiple errors and warnings marked in red, adding the SDK JAR and so libraries to the project removes these errors and warnings.

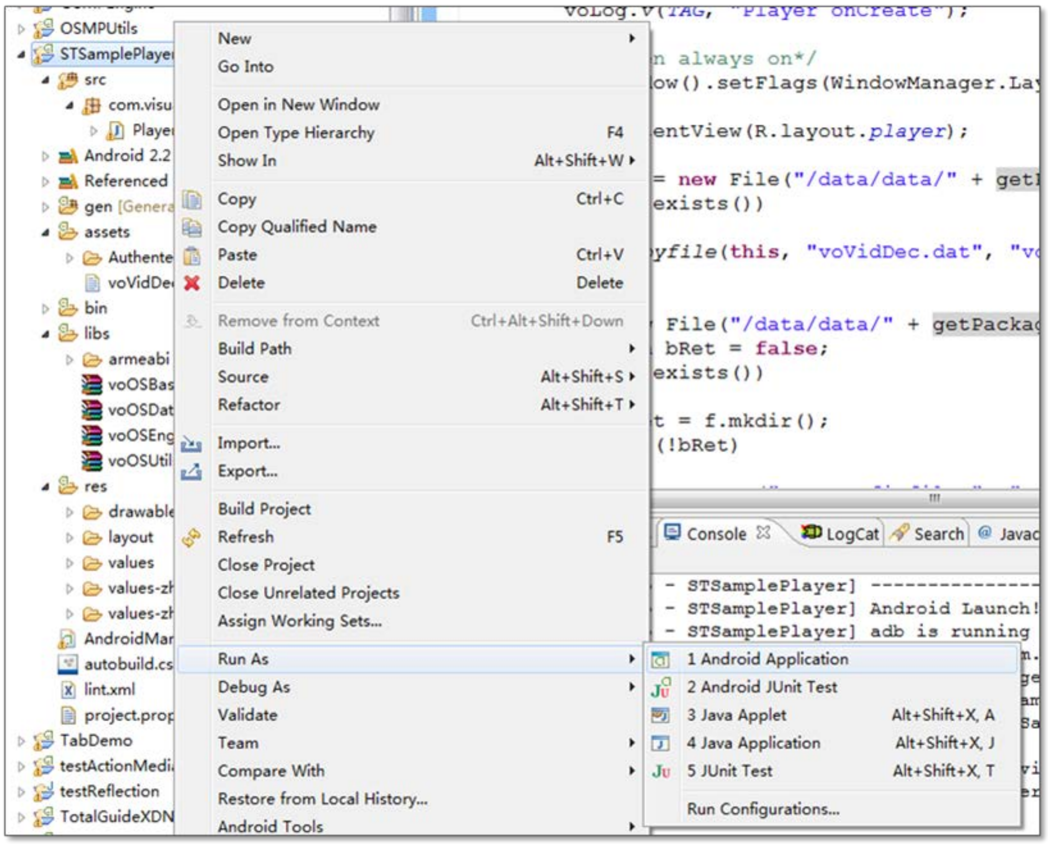
Note: Refer to the *Add the SDK Libraries section* of the *OnStream MediaPlayer+ SDK Project Setup for Android* document for details about how to add the SDK libraries to a project.

3.1.1.2 Running the Sample Player on a device

When the SDK is added to the SamplePlayer project in Eclipse, you are ready to run the application on an Android device.

To run the Sample Player on an Android device:

1. Connect the device to the Eclipse workstation with a USB cable.
2. In **Eclipse**, right click the **SamplePlayer** project in **Project Explorer**, and then select **Run as > Android Application**, as shown in Figure 16. Then the SamplePlayer application can be explored on the Android device.



```
voLog.v(TAG, "Player onCreate");
n always on*/
ow().setFlags(WindowManager.La
entView(R.layout.player);
= new File("/data/data/" + get
exists())
yfile(this, "voVidDec.dat", "v
 File("/data/data/" + getPacka
 bRet = false;
exists())
t = f.mkdir();
(!bRet)
```

```
- STSamplePlayer] ---------------
- STSamplePlayer] Android Launch!
- STSamplePlayer] adb is running
```

Figure 16. Context Menu for SamplePlayer Project in Eclipse IDE

3.1.1.3 Uninstalling the Sample Player on a device

To uninstall the Sample Player on an Android device:

1. Tap **Settings**.

2. Tap **Apps** under DEVICE.
3. Tap **Sample Player** in the Apps list.
4. Tap **Uninstall**.
5. Tap **OK** to confirm the uninstallation.

3.1.2 Sample player usage

This section describes the usage of the Sample Player application included with the SDK. The Sample Player application implements a fully integrated media player with the following features:

- Select the media source
- Select the subtitle source
- Select the option for opening the media source, including Synchronous and Asynchronous
- Set the preferred audio and subtitle languages
- Control the playback through buttons
- Control the sound volume and mute/unmute the sound
- Support the Closed Captions and subtitle rendering
- Configure the asset properties

3.1.2.1.1 Opening media source

Media source can be opened in either Sync or Async mode. The Sample Player uses Async mode by default. For sample code of opening media source, refer to 4.1.3 Opening media source.

- Sync mode: Opening the media source in sync mode causes the client application to block until the open method completes or returns.
- Async mode: In Async mode, the API call returns immediately, and the client application is not blocked waiting for the media source to be opened. The client application can then wait for an event to indicate that the media source was opened successfully before starting the media playback.

To activate the asynchronous option:

1. In the initial view tap **Options**.
2. Select the checkbox next to **Asynchronous/Synchronous Open**.

3.1.2.1.2 Selecting/Playing the Media Source

The initial view allows you to select the media source (for example, H.264 file, MP4 file, or HTTP Live Streaming link) for playback. You can type the media source path manually or select from a list of media source listed in a text file.

To manually type a media file or link:

1. In the initial view, type the full URL starting with *http://* or file path to the media source in the Input streaming URL address or file path text box.
2. Tap the play button.

Alternately, a list of media files and links can be supplied to the Sample Player through a text file. The text list must be saved to */sdcard/url.txt*, with each link URL and file path separated by line. To select a media file or link from *url.txt*:

1. In the initial view, tap the button next to the URL field to display the media file list.
2. Select the required media source, and then tap **OK**.
3. Tap the play button.

Refer to 4.1.5 Starting playback for the sample code of starting playback. For details about initializing the SDK player, opening the media source, and starting playback, refer to section 3 (Basic Integration) in the *OnStream MediaPlayer+ Player SDK Integration Guide for Android Platforms.*

3.1.2.2 Options

3.1.2.2.1 Selecting Video Render Type

Tap **Video Render Type** from **Options**, and then tap the required Video Render Type for the Sample Player, as shown in Figure 17. Refer to *setRenderType(VO_OSMP_RENDER_TYPE type)* for API related information.

- **Native Window**: The output format of rendering data is .rgb.
- **Native C**: The output format of rendering data is .yuv.

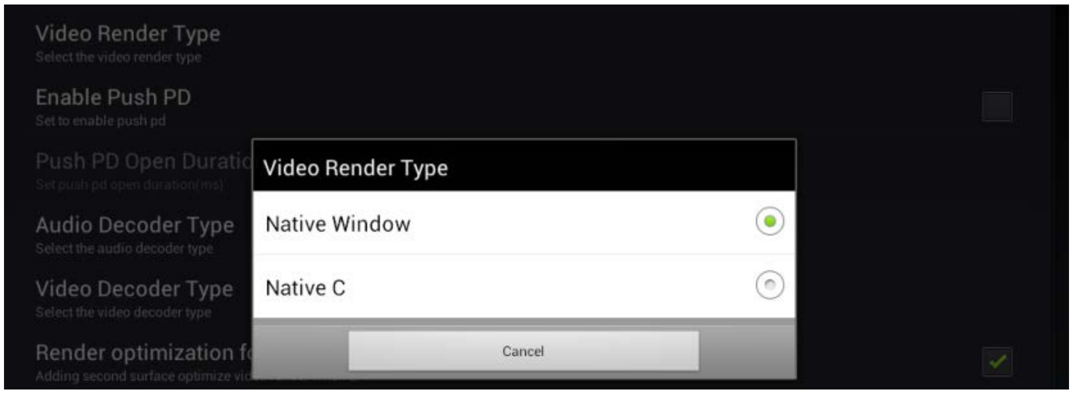


Figure 17. Select Video Render Type

3.1.2.2.2 Enabling Push PD

Push PD enables to download the video to the viewer's computer or device and stored in a temporary directory, when the video is played. The video will begin to play when enough of the file has been downloaded to the computer or device. Refer to *open(String url, VO_OSMP_SRC_FLAG flag, VO_OSMP_SRC_FORMAT type, VOOSMPOpenParam openParam)* for API related information. Modify the value of *VO_OSMP_SRC_FORMAT type* in the above API to set the type of opening source.

To enable Push PD, tap the checkbox next to **Enable Push PD** from **Options** to enable Push PD for the Sample Player.

To set the duration for Push PD:

1. Enable Push PD. Refer to 3.1.2.2.2 Enabling Push PD for details.
2. Tap **Push PD Open Duration**, and then type the number to set the duration for push PD for the Sample Player, as shown in Figure 18. The PD duration is measured in millisecond.
3. Tap **OK**.

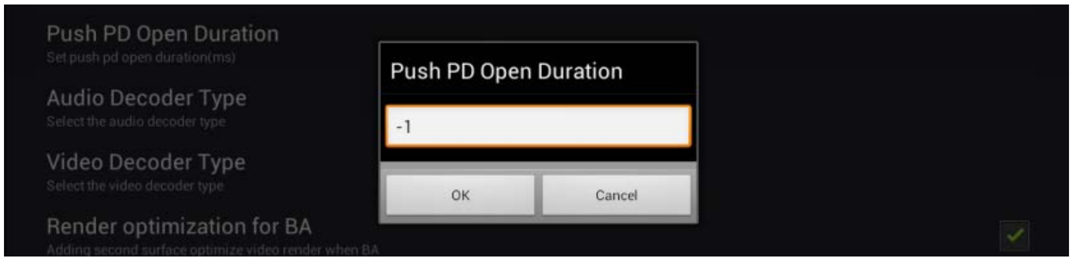

Push PD Open Duration
Set push pd open duration(ms)
Audio Decoder Type
Select the audio decoder type
Video Decoder Type
Select the video decoder type
Render optimization for BA
Adding second surface optimize video render when BA
Push PD Open Duration
-1
OK
Cancel

Figure 18. Set duration for push PD

Refer to *open(String url, VO_OSMP_SRC_FLAG flag, VO_OSMP_SRC_FORMAT type, VOOSMPOpenParam openParam)* and *openParam.setDuration(value)* for API related information.

3.1.2.2.3 Selecting audio decoder/video decoder

The Sample Player allows you to select the decoder type manually. Refer to 1.4.7.1 Hardware acceleration for Android for details on explanations and settings of decoder types.

3.1.2.2.4 Render optimization for BA

The Sample Player allows you to optimize rendering when bitrate adaptation is being applied by selecting **Render Optimization for BA** from **Options**.

3.1.2.2.5 Dolby Surround Module

The Sample Player has a Dolby Surround Module to provide the better audio effects. Tap **Dolby Surround Module** from **Options** to load the module.

Refer to *enableAudioEffect(boolean value)* for API related information. You can set *setAudioEffectEndpointType(VO_OSMP_AUDIO_EFFECT_ENDPOINT_TYPE.VO_OSMP_AUDIO_EFFECT_ENDPOINT_HEADPHONE)* to enhance the audio effect.

3.1.2.2.6Enabling/disabling deblock

Deblocking is a feature applied to decoded compressed video to improve visual quality and prediction performance by smoothing the sharp edges which can form between macroblocks when block coding techniques are used.

Enabling deblock helps improve the playback quality, especially when the video has mosaic. Tap **Deblock** from **Options** to enable the deblock feature. Refer to *enableDeblock(boolean enable)* for API related information.

3.1.2.2.7Selecting Subtitle source

The Sample Player allows you to optionally select an external subtitle source (for example, CEA 608, CEA 708, SRT, and SMI) for Closed Captions or subtitles rendering. A list of subtitle files and links can be supplied to the Sample Player through a text file. The text list must be saved to */sdcard/subtitle.txt*, with each link URL and file path separated by a line. Refer to *setSubtitlePath(String filePath)* for API related information.

To select an external subtitle file or link from *subtitle.txt*:

1. Tap **Subtitle URL**.
2. Select a URL from the */documents/subtitle.txt* list menu, as shown in Figure 19.
3. Select the required subtitle source.

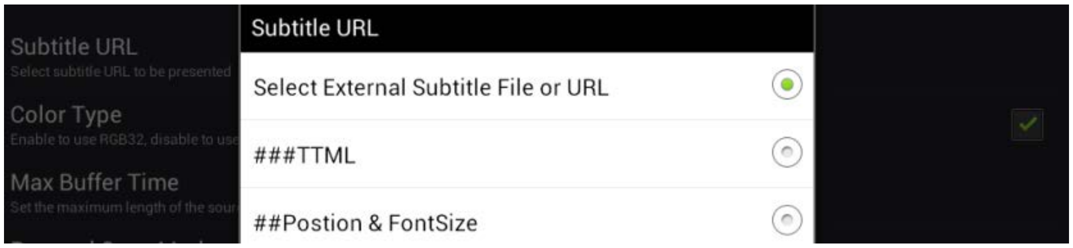

Subtitle URL
Select subtitle URL to be presented
Color Type
Enable to use RGB32, disable to us
Max Buffer Time
Set the maximum length of the sour
Subtitle URL
Select External Subtitle File or URL
###TTML
##Postion & FontSize

Figure 19. Select an external subtitle

3.1.2.2.8 Setting buffering time

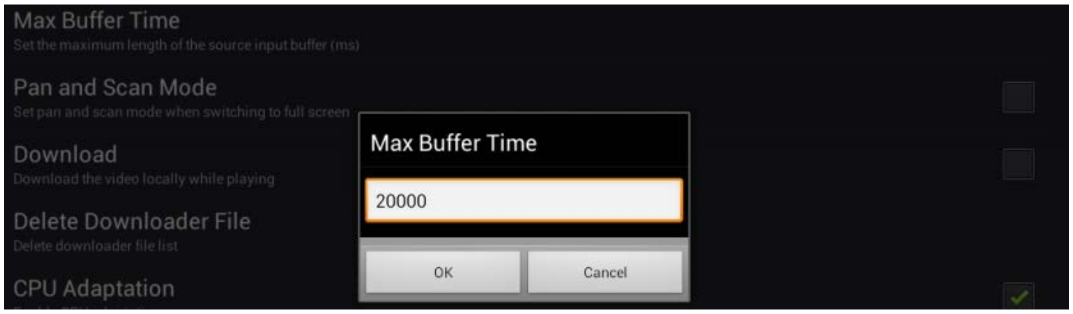
The OSMP+ player allows you to set the buffering time of the source to be played.

3.1.2.2.8.1 Max Buffering Time

Refer to *setMaxBufferingTime(int time)* for API related information.

To set the maximum time of source buffer:

1. Tap **Max Buffering Time** from **Options**.
2. Type the number to set the maximum buffering time in milliseconds, as shown in Figure 20.


Max Buffer Time
Set the maximum length of the source input buffer (ms)
Pan and Scan Mode
Set pan and scan mode when switching to full screen
Download
Download the video locally while playing
Delete Downloader File
Delete downloader file list
CPU Adaptation

Max Buffer Time
20000

Figure 20. Set maximum buffer time

3. Tap **OK**.

3.1.2.2.8.2 Initial Buffering Time

Turn on **Initial Buffering Time** option to set the buffering time of the source in milliseconds before the playback starts. Refer to *setInitialBufferingTime(int time)* for API related information.

To enable the initial buffering time of source buffer:

1. Tap **Initial Buffering Time** from **Options**.
2. Type the number to set the initial buffering time in milliseconds.

3.1.2.2.8.3 Playback Buffering Time

When re-buffering is needed during playback, the player enables you to set the buffering time with the **Playback Buffering Time** option. Refer to *setPlaybackBuferingTime(int time)* for API related information.

To set the re-buffering time:

1. Tap **Playback Buffering Time** from **Options**.
2. Type the number to set the time in milliseconds.

3.1.2.2.9 Enabling Download Manager

The video files can be downloaded locally while playing if the Download Manager is enabled. This feature is only for VoD and Smooth Streaming.

To enable Download Manager, tap the checkbox next to **Download** from **Options**. For sample code of Download Manager related features, refer to 4.1.10 Download Manager (Optional). For details about how to integrate Download Manager on Android, refer to the *OnStream MediaPlayer+.Download Manager Integration Guide for Android.pdf* document in your installation package.

If Download Manager is turned on, the **Select Asset** dialog prompts when starting the playback. Specify the Video, Audio, and Subtitle tracks to be downloaded, and then tap **OK** to download the source to be played. If you do not want to keep the downloaded files on your device, tap **Delete Downloader File** from **Options**, tap **Delete All** to remove all downloaded files, or select the file to be removed and then tap **Delete**.

3.1.2.2.10 Enabling CPU Adaptation

The Bitrate Adaptation of Sample Player subjects to setting that is hardcoded in the player if CPU Adaptation is enabled.

To enable CPU Adaptation, tap the checkbox next to **CPU Adaptation** from **Options**. Refer to *enableCPUAdaptation(boolean value)* for API related information.

3.1.2.2.11 RTSP Settings

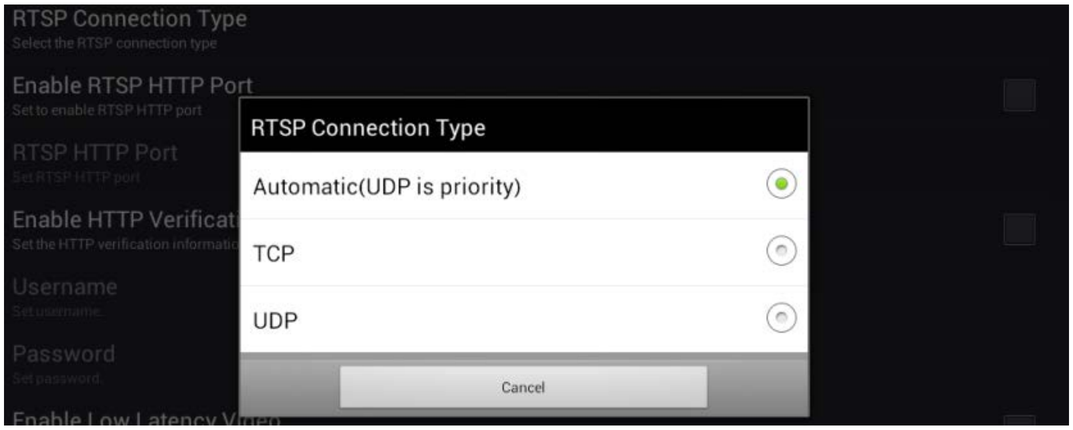
For the sample code regarding RTSP settings, refer to 4.1.14 RTSP settings (Optional).

3.1.2.2.11.1 Setting RTSP Connection Type

Refer to *setRTSPConnectionType(VO_OSMP_RTSP_CONNECTION_TYPE type)* for API related information.

To set RTSP Connection Type:

1. Tap **RTSP Connection Type** from **Options**.
2. Three connection types, **Automatic(UDP is priority), TCP**, and **UDP** are available for selection, as shown in Figure 21.
   - **Automatic (UDP is priority):** The player selects the connection types automatically.
   - **TCP**: refer to 1.3 Terminology for TCP definition.
   - **UDP**: refer to 1.3 Terminology for UDP definition.


RTSP Connection Type
Select the RTSP connection type
Enable RTSP HTTP Port
Set to enable RTSP HTTP port
RTSP HTTP Port
Set RTSP HTTP port
Enable HTTP Verificat
Set the HTTP verification informatio
Username
Set username
Password
Set password
Enable Low Latency Video

RTSP Connection Type
Automatic(UDP is priority) [✓]
TCP
UDP

Figure 21. Available RTSP connection types

3. Select the connection type as needed.

3.1.2.2.11.2 Enabling RTSP HTTP Port

Refer to *enableRTSPOverHTTP(boolean enable)* and *setRTSPOverHTTPConnectionPort(int portNum)* for API related information.

Tap **Options**, and then tap **Enable RTSP HTTP Port** to enable RTSP HTTP port function. Tap **RTSP HTTP Port**, and then type a value to define the Port to access the RTSP media source, and then tap **OK**, as shown in Figure 22.

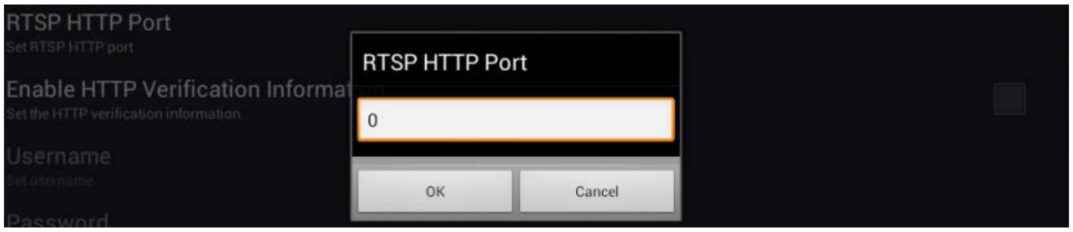


Figure 22. Define the port

3.1.2.2.11.3 Enabling Low Latency Video

The Low Latency Video feature allows you to easily build a high quality and low latency HD video transmission system.

Tap **Enable Low Latency Video** to set a value as the acceptable level for RTSP Camera video latency.

Refer to *enableLowLatencyVideo(boolean value)* for API related information.

3.1.2.2.11.4 Enabling RTSP Max Socket Error Count

Refer to *setRTSPMaxSocketErrorCount(int count)* for API related information.

The player allows setting the maximum number of socket error. To set the RTSP Max Socket Error Count:

1. Tap **Enable RTSP Max Socket Error Count** from **Options** to enable it.
2. Tap **RTSP Max Socket Error Count**.
3. Type a value to set the entry level for RTSP Socket error handling, and then tap **OK**, as shown in Figure 23.

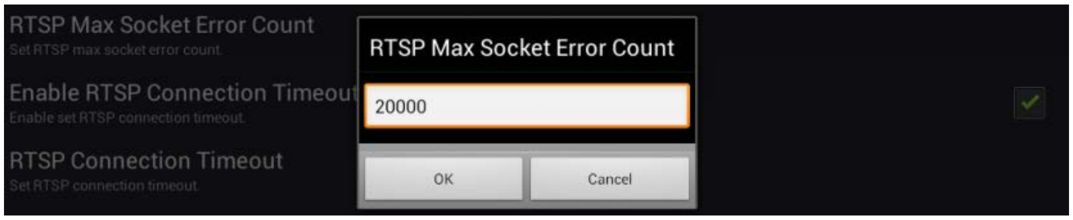

RTSP Max Socket Error Count
Set RTSP max socket error count
Enable RTSP Connection Timeout
Enable set RTSP connection timeout
RTSP Connection Timeout
Set RTSP connection timeout
RTSP Max Socket Error Count
20000
OK
Cancel

Figure 23. Set the entry level

3.1.2.2.11.5 Enabling RTSP Connection Timeout

Refer to *setRTSPConnectionTimeout (int count)* for API related information.

To set the RTSP Connection Timeout:

1. Tap **Enable RTSP Connection Timeout** from **Options** to enable it.
2. Tap **RTSP Connection Timeout**.

3. Type a number to set a value that is the players' retry times when server has no response, and then tap **OK**, as shown in Figure 24.

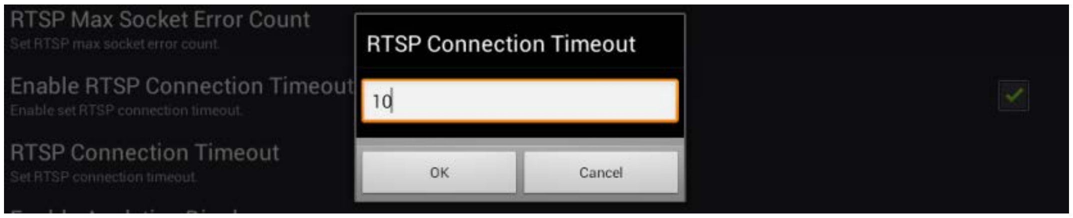

RTSP Max Socket Error Count
Set RTSP max socket error count.
Enable RTSP Connection Timeout
Enable set RTSP connection timeout.
RTSP Connection Timeout
Set RTSP connection timeout.
RTSP Connection Timeout
10
OK
Cancel

Figure 24. Set the retry times

3.1.2.2.12 HTTP Settings

3.1.2.2.12.1 Enabling HTTP Verification Information

To set the HTPP Verification Information:

1. Tap **Enable HTTP Verification Information** from **Options** to access the Username and Password, as shown in Figure 25.

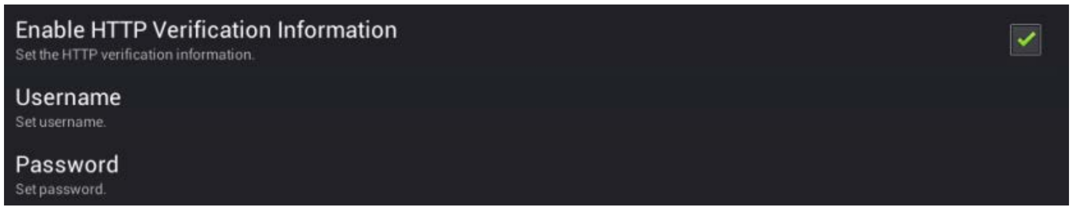

Enable HTTP Verification Information
Set the HTTP verification information.
Username
Set username.
Password
Set password.

Figure 25. Enable HTTP Verification Information

2. Tap **Username** or **Password**.
3. Type the user name and password to authorize the RTSP media source access, and then tap **OK**, as shown in Figure 26.

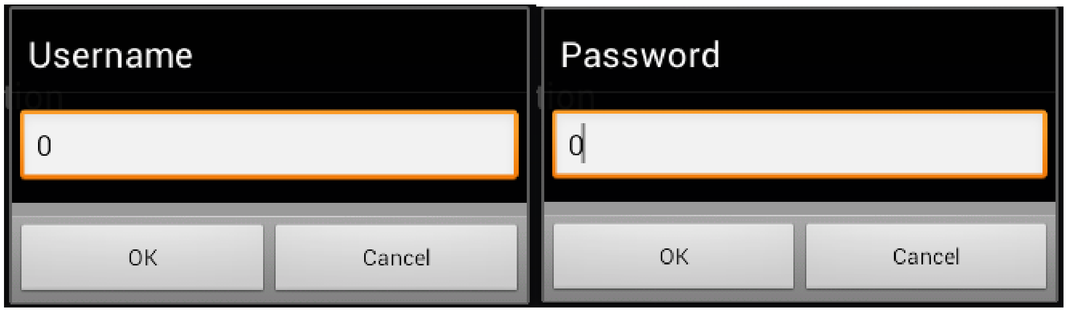


Figure 26. Set the user name and password

See the following sample code for reference.

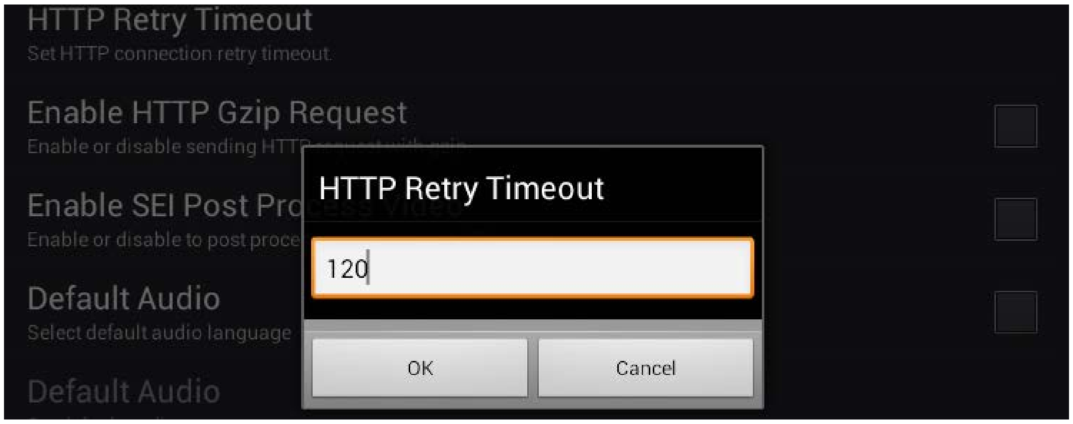
```
VOOSMPVerificationInfo verif = new VOOSMPVerificationInfo();   str
= userName+":"+password;
  verif.setVerificationString(str);
  verif.setDataFlag(1);
m_sdkPlayer.setHTTPVerificationInfo(verif);
```

3.1.2.2.12.2 Enabling HTTP Retry Timeout

Refer to *setHTTPRetryTimeout(int time)* for API related information.

To set the HTPP Retry Timeout:

1. Tap **Enable HTTP Retry Timeout** from **Options** to enable the retry timeout function.
2. Tap **HTTP Retry Timeout**.
3. Type a number to set the player retry times when server has no response, and then tap **OK**, as shown in Figure 27.


HTTP Retry Timeout
Set HTTP connection retry timeout
Enable HTTP Gzip Request
Enable or disable sending HTTP request with gzip
Enable SEI Post Process Video
Enable or disable to post proce
Default Audio
Select default audio language
Default Audio

HTTP Retry Timeout
120

Figure 27. Set retry times

3.1.2.2.12.3 Enabling HTTP Gzip Request

HTTP data is compressed before it is sent from the server: compliant browsers will announce what methods are supported to the server before downloading the correct format; browsers that do not support compliant compression method will download uncompressed data. Gzip is one of the most common compression schemes. Refer to *enableHTTPGzipRequest(boolean enable)* for API related information.

Tap **Enable HTTP Gzip Request** from **Options** to enable or disable the support of the GZIP format.

3.1.2.2.12.4 Setting token for HTTP Request

The sample player enables to set a token/string for HTTP requests. Refer to *setURLQueryString()* for API related information.

Tap **URL Query String** from **Options** to enable or disable the token support for HTTP requests. After enabling this option, tap **Enable URL Query String**, and then type the **Query String** in the prompted dialog.

Note: The query string should be in the format of '*key1=value&key2=value*'.

3.1.2.2.13 Playback Looping

If the playback looping feature is on and you're playing your timeline, the playhead cycles back to the start of the sequence when you reach the end. If it's off, your playhead simply stops at the end of the sequence.

Tap **Playback Looping** from **Options** to enable or disable the Playback looping.

This feature will be implemented when the player receives the *VO_OSMP_CB_PLAY_COMPLETE* event. Set *setPosition(0)* to playback again.

3.1.2.2.14 Subtitle Settings

Tap **Subtitle Settings** from **Options** to enable or disable the Subtitle Settings. For sample code of subtitle settings, refer to 4.1.9 Subtitle rendering (Optional).

Table 6 lists the available subtitle attributes that can be set in the OSMP+ player and their related API s.

Table 6. Subtitle attributes and related APIs

Attribute	**Definition**	**API**
Enable Subtitle	Enable/disable subtitles display. Disabled is set by default.	*enableSubtitle(Boolean value)*
Enable Default Settings	Use the subtitle settings from source stream	
Reset Subtitle Settings	Reset all parameters to the default values that are specified in source stream.	*resetSubtitleParameter()*
Enable Subtitle Bounding Box • Set Top Percent • Set Left Percent • Set Bottom Percent • Set Right Percent	Set the subtitle bounding box, which will overwrite the subtitle settings from parser.	*setSubtitleBoundingBox(int topPercent, int leftPercent, int bottomPercent, int rightPercent)*
Enable Subtitle Gravity • Set Horizontal Position • Set Vertical Position	Set the gravity of the bounding box.	*setSubtitleGravity(VO_OSMP_HORIZONTAL horizontal, VO_OSMP_VERTICAL vertical)*
Enable Subtitle Trim	Set the set of characters to be trimmed from the beginning or end of subtitle.	*setSubtitleTrim(String trimChars)*
Enable Subtitle Auto Adjustment	Enable/Disable the automatic adjustment for subtitle position/font size. Disabled is set by default.	*enableSubtitleAutoAdjustment(boolean value)*

Attribute	Definition	API
Enable Font size	Set subtitle font size scale.	*setSubtitleFontSizeScale(int scale)*
Enable Font Color Opacity	Set subtitle font color opacity rate.	*setSubtitleFontOpacity(int alpha)*
Enable Font Color List	Set subtitle font color.	*setSubtitleFontColor(int color)*
Enable Background Color Opacity	Set subtitle font background color opacity rate.	*setSubtitleFontBackgroundOpacity(int alpha)*
Enable Background Color List	Set subtitle font background color.	*setSubtitleFontBackgroundColor(int color)*
Enable Edge Color Opacity	Set subtitle font edge color opacity rate.	*setSubtitleFontEdgeOpacity(int alpha)*
Enable Edge Color List	Set subtitle font edge color.	*setSubtitleFontEdgeColor(int color)*
Enable Edge Type List	Set subtitle font edge type.	*setSubtitleFontEdgeType(int type)*
Enable Font List	Set subtitle font name.	*setSubtitleFontName(String name)*
Enable Window Background Color Opacity	Set window background color opacity rate.	*setSubtitleWindowBackgroundOpacity(int alpha)*
Enable Window Background Color List	Set window background color.	*setSubtitleWindowBackgroundColor(int color)*
Enable Underline Font	Enable or disable the underline font for subtitle.	*setSubtitleFontUnderline(Boolean enable)*
Enable Bold Font	Enable or disable the bold font for subtitle.	*setSubtitleFontBold(Boolean enable)*
Enable Italic Font	Enable or disable the italic font for subtitle.	*setSubtitleFontItalic(Boolean enable)*

3.1.2.2.15 Enabling Initial Bitrate

The OSMP+ player enables to select a suitable initial bitrate, which can utilize the TCP throughput to achieve a higher-than-lowest bitrate if possible. By providing a better picture quality, the quality of experience can be improved. Refer to *setInitialBitrate(int bitrate)* for API related information.

To set the initial bitrate:

1. Tap **Enable Initial Bitrate** from **Options** to enable or disable the Initial Bitrate setting.
2. Tap **Initial Bitrate**, type a value for the Initial Bitrate, and then Tap **OK**, as shown in Figure 28.

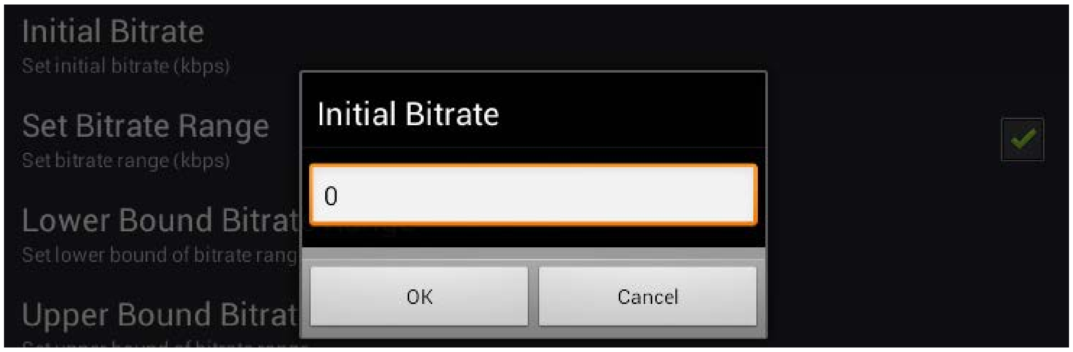


Figure 28. Set the value of Initial Bitrate

3.1.2.2.16 Setting Bitrate Range

Refer to *setBitrateThreshold(int upper, int lower)* for API related information.

To set the bitrate range:

1. Tap **Set Bitrate Range** from **Options** to enable or disable the Bitrate Range Settings.
2. Tap **Lower Bound Bitrate Range** or **Upper Bound Bitrate Range**, type the boundary value for lower and upper bound, and then Tap **OK**, as shown in Figure 29.

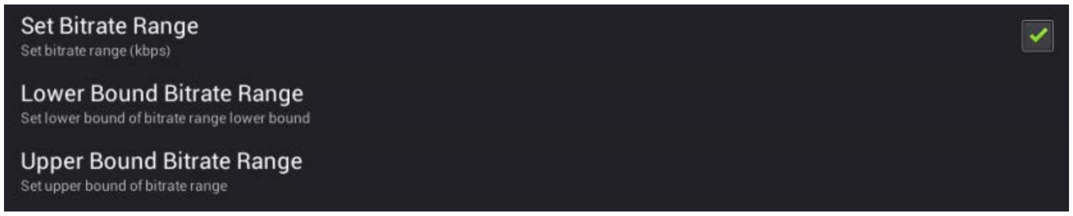

Set Bitrate Range
Set bitrate range (kbps)
Lower Bound Bitrate Range
Set lower bound of bitrate range lower bound
Upper Bound Bitrate Range
Set upper bound of bitrate range

Figure 29. Set the lower and upper bound value

3.1.2.2.17 Default Audio and Subtitle

Refer to *setPreferredAudioLanguage(String[] languageList)* and *setPreferredSubtitleLanguage(String[] languageList)* for API related information.

Tap **Default Audio** or **Default Subtitle** from **Options** to enable or disable the Default Audio and Subtitle settings, as shown in Figure 30.

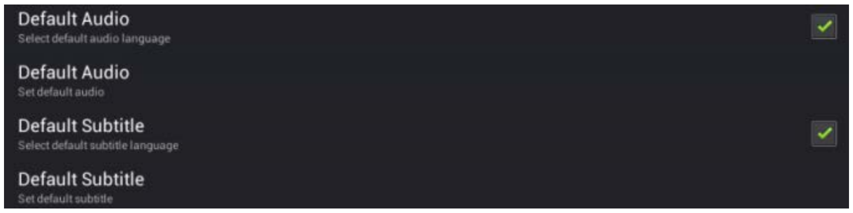

Default Audio
Select default audio language ✓
Default Audio
Set default audio
Default Subtitle
Select default subtitle language ✓
Default Subtitle
Set default subtitle

Figure 30. Enable or disable settings of audio and subtitle

3.1.2.2.18 Preference settings

The OSMP+ player allows user to set the preference of playback. Select **Set preferences** from **Options** to enable the following options. Refer to *setPreference()* for API related information.

3.1.2.2.18.1 Keeping Last Frame

During the transition from live channel to NTS media, the OSMP+ player keeps the last frame of the live channel being played until the first frame of NTS media is played, when the **Keeping Last Frame** option is turned on. In the meantime, the OSMP+ player keeps the screen black when stopping or closing any playback.

3.1.2.2.18.2 Seek Precise

The OSMP+ player provides a new option, **Seek Precise**, for smooth streaming to allow user setting the offset to the live point, so that user can seek to the play point based on this offset value precisely.

3.1.2.2.18.3 Select Audio Switch Immediately

When the **Select Audio Switch Immediately** options is turned off, the OSMP+ player continues playing the previous content and then switches to the selected audio track when the previous playback is complete. This option is turned on by default.

3.1.2.2.19 Set presentation delay time

The OSMP+ player provides the **Set presentation delay time** option for HLS and Smooth Streaming, for enabling user to download chunks with some seconds of delay that is defined by this option. Refer to *setPresentationDelay()* for API related information.

3.1.2.2.20 Enter NTS

The OSMP+ player provides the Enter Network Time Shifting (NTS) feature to test resuming playback from the pause point, rather than the live point.

The Enter NTS feature requires the collaboration within the streaming server, application, and OSMP+ SDK. Streaming server should create the NTS stream once the user pause live stream playback, and notify the application of the new URL of NTS stream.

The application should call OSMP+ SDK API to switch to NTS stream once the user resumes playback (after perform the pause operation for a while).

Also the application should call OSMP+ SDK API to switch back to Live stream once the user catches up with the Live content (typically after a seek operation).

The requirements to use this features are:

- The streams support #EXT-X-PROGRAM-DATE-TIME HLS
- The playlist of the live stream at the time of pause and the playlist of the NTS stream at the time of resume should have the same #EXT-X-PROGRAM-DATE-TIME HLS value

In other words, there are two streams, a live stream that is paused and a NTS stream that is resumed. Both streams share the same content.

The Enter NTS feature enables playback of the NTS stream instead of the live stream when resuming playback. To use this feature, enter the value of the *PROGRAM-DATE-TIME HLS* tag (when the live stream was paused) in the **Min pos** box. **Min Pos** needs to be converted into millisecond.

After clicking **EnterNTS**, the player starts playback at the HLS Live paused position using the NTS stream. To verify that the resume operation is successful, one can check that the UTC time display should not change more than one second when playback resumes.

To enable the Enter NTS playback:

1. Select **Enter NTS** from **Options**.
2. In the **NTS URL** box, type the URL address of the NTS stream that shares the same content with the live stream being played.
3. Convert the UTC time to unix timestamp, and then type this converted unix timestamp in the **NTS min position** box. The unit of Min pos is millisecond.

3.1.2.2.21 Enabling HW decoder max resolution

The OSMP+ player supports to specify both maximum height and maximum width to enable adaptive plahyback for a video decoder. To enable it, tap **Enable HW Decoder Max Resolution** from **Options**. For more information, refer to *setHWDecoderMaxResolution()* in the API Reference Manual.

3.1.2.2.22 Previous seek to

The OSMP+ player supports to play the content from a specified position. To enable the previous seek to feature, tap **Previous seek to** from **Options**, and then specify a position in the unit of ms in the **Seek To Value** dialog.

3.1.2.2.23 Enable Live Streaming DVR Position

The OSMP+ player supports defining position based on live streaming DVR window. DVR window is also referred to as the duration of live streaming.

To enable defining the position, tap **Live Streaming DVR Position** from **Options**.

When this option is enabled:

- Value of **Play**: this value is 00:00:00 when playing at the live head. If the play point is not the live point, this value will be a negative value to represent the offset between DVR window and live head.
- Value of **Max**: this value is always 00:00:00 that represents the live head.
- Value of **Min**: this value is (value of Max - DVR window size).

When this option is disabled:

- Value of **Play**: this value is the offset between the play head and the beginning of the playback.
- Value of **Max**: this value is the offset between the live head and the beginning of the playback.
- Value of **Min**: this value is (value of Max - DVR window size).

For more information, refer to *enableLiveStreamingDVRPosition()* in the API Reference Manual.

3.1.2.2.24Version

The Sample Player version information appears in **Version** of the **Options** menu.

3.1.2.3 Playback controls

During playback, the Sample Player provides a set of basic controls, including a shared Start/Pause button, Stop button, and a Seek bar.

Notes:

- Refer to Section 4 (Advanced Integration: Pause/Play Control) of the *OnStream MediaPlayer+ Player SDK Integration Guide for Android* for details about Pause/Play integration.
- Refer to Section 6 (Advanced Integration: Seekbar Control) of the *OnStream MediaPlayer+ Player SDK Integration Guide for Android* for details about Seekbar integration.

3.1.2.4 Channel switching

During playback, the SamplePlayer application supports channel switching from the list of added media sources saved in /sdcard/url.txt.

The API calling sequence should start with *stop()*, and then *close()*, *open()* should be last called. The player can set other configurations before *open()*.

To switch channels (sources):

1. Tap **Prev** or **Next** on the right side of the screen to switch to the previous or next media source respectively.

Note: Refer to Section 7 (Advanced Integration: Channel Switching) of the *OnStream MediaPlayer+ Player SDK Integration Guide for Android* for details about Channel switching.

3.1.2.5 Controlling audio speed

During playback, the Sample Player allows to control the audio speed.

Tap **+** to speed up the audio playback or **–** to slow down the audio playback.

Refer to *setAudioPlaybackSpeed(float speed)* for API related information.

3.1.2.6 Closed Captions

The Sample Player demonstrates the built-in SDK handling of CC and subtitles rendering. If CC data exists, it appears as shown in Figure 31.

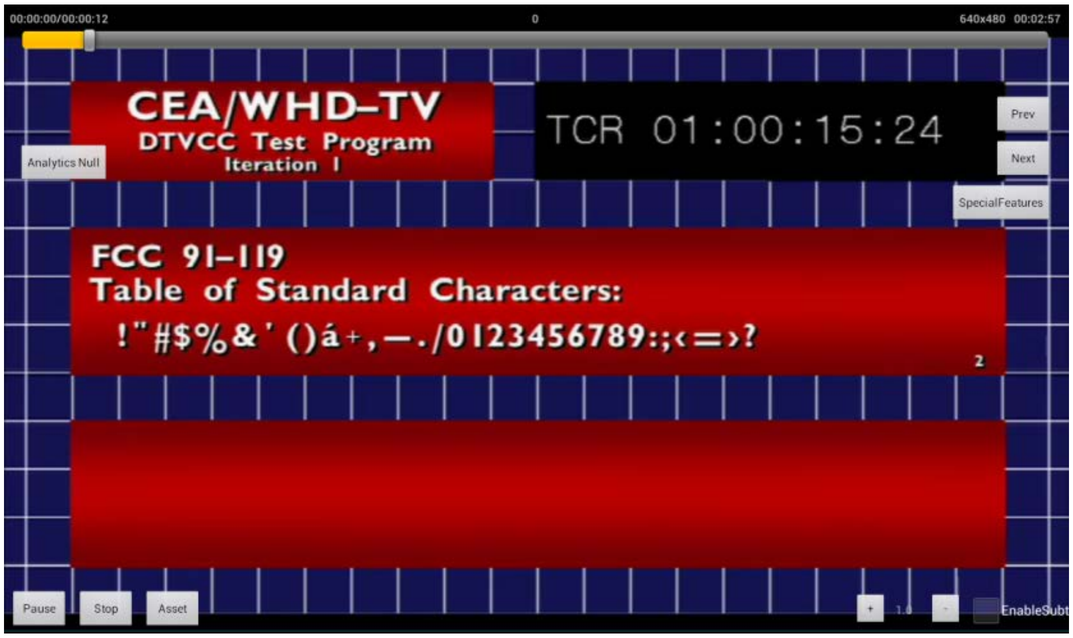


Figure 31. Sample Player with Closed Captions

When CC Show is selected (default), the SDK player automatically handles CC display when data arrives. When CC output is selected, the application manages CC data. Refer to *enableSubtitle(boolean value)* for API related information.

3.1.2.7 Switching Asset properties

If the media source includes multiple video tracks (also known as bitrate), audio tracks, or subtitles, the Sample Player application supports switching properties during the playback. Refer to 4.1.8 Selecting tracks for details about this feature.

To switch properties:

1. Tap **Asset** on the bottom of the video screen to display the **Asset** menu.
2. Tap the required Video, Audio, and/or Subtitle track.
3. Tap **Commit** to confirm your selection. Playback continues with your selection, as shown in Figure 32.

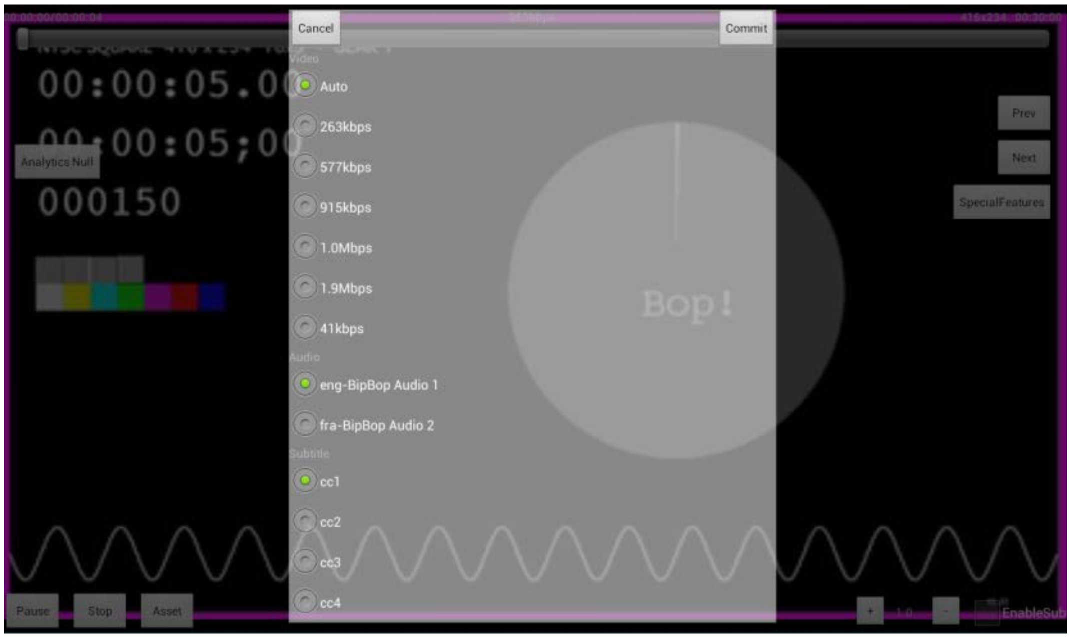


Figure 32. Sample Player Asset menu

3.1.2.8 Special features

3.1.2.8.1 Screen Brightness

You can adjust the screen Brightness during the playback by dragging the Screen Brightness bar in **SpecialFeatures**, as shown in Figure 33. Refer to *setScreenBrightness(int brightness)* for API related information.

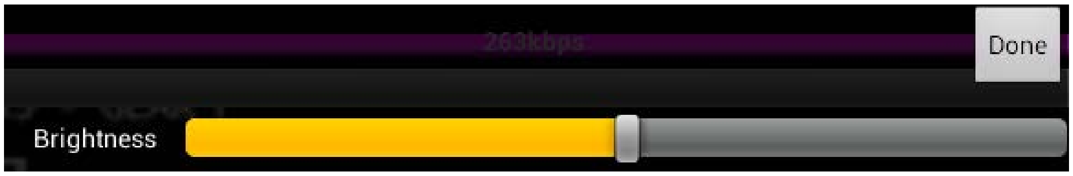


Figure 33. Screen brightness

3.1.2.8.2 Playback Volume

You have several options for controlling volume in the Sample Player, including adjusting the volume level, muting, or unmuting. Refer to *setVolume(float volume), mute(),* and *unmute()*for API related information.

To adjust the volume level, drag the playback volume bar and click **Mute** to mute the volume. To restore the audio, click **UnMute**.

3.1.2.8.3 Stereo Channel

The Sample Player application allows you to set the stereo channel as needed. Specify the need number for the left/right channel, and then tap **SetStereo**.

3.1.2.8.4 Updating the media source

During playback, use the update source URL function to balance the workload within media servers. This feature is specific for the optimization for content delivery network (CDN) delivery. Refer to *updateSourceURL(String url)* for API related information.

Note: The previous URL will be expired a few seconds after the new URL is activated. Master manifest, sub-manifest, and TS chunks of new URL must be exactly same with the previous media file.

---

1. Tap **SpecialFeatures**.
2. Type the required media URL address into the Update URL box, and then tap **Update**, as shown in Figure 34.

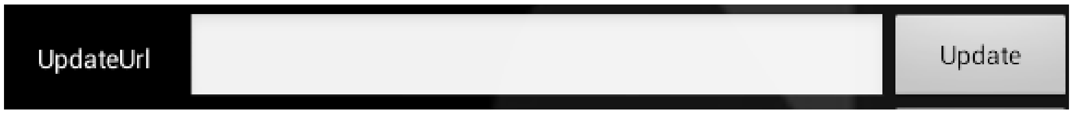


Figure 34. Update the media source

3.1.2.8.5 Setting Zoom mode

The Sample Player application allows you to select the zoom mode as needed. To set the zoom mode:

1. Tap **SpecialFeatures**.
2. Tap **Zoom**, select the required mode from the following list:
   a. FullWindow
   b. ZoomIn
   c. LetterBox
   d. Original
   e. PanScan

Refer to 1.4.11 Zoom modes for more information about zoom modes.

3.1.2.8.6 Setting the aspect ratio

The Sample Player application allows you to set the aspect ratio as needed. Refer to *setVideoAspectRatio(VO_OSMP_ASPECT_RATIO ar)*for API related information.

To set the aspect ratio:

1. Tap **SpecialFeatures**.
2. Tap **Auto** to select the required aspect ratio, as shown in Figure 35.

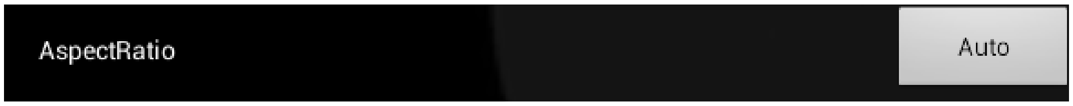


Figure 35. Set the aspect ratio

3. Tap **Done**.

3.1.2.8.7 SEI

During playback, tap the checkbox next to **SEI** to enable the SEI setting to collect the SEI data from the media source.

See the following for the sample code of starting SEI.

```
m_sdkPlayer.enableSEI(VO_OSMP_SEI_INFO_FLAG.VO_OSMP_SEI_INFO_PIC_T
IMING);
m_sdkPlayer.startSEINotification(5000);
```

See the following for the sample code of stopping SEI.

```
m_sdkPlayer.stopSEINotification();
```

3.1.2.8.8 RTSP Statistics

During playback, tap the checkbox next to **RTSP Statistics** to collect the RTSP information from the media source.

The player uses a timer to get RTSP statistics by using *getRTSPStatistics()*.

3.1.2.8.9 Download Status

During playback, tap the checkbox next to **Download Status** to track the HTML5 download information from the media source.

The player uses a timer to get the download status during playback. See the following for the sample code of getting download status.

```
int validBufferDuation = m_sdkPlayer.getValidBufferDuration();
 VO_OSMP_DOWNLOAD_STATUS audioDownloadStatus =
m_sdkPlayer.getDownloadStatus(VO_OSMP_SOURCE_STREAMTYPE.VO_OSMP_SS
_AUDIO);
 VO_OSMP_DOWNLOAD_STATUS videoDownloadStatus =
m_sdkPlayer.getDownloadStatus(VO_OSMP_SOURCE_STREAMTYPE.VO_OSMP_SS
_VIDEO);
 VO_OSMP_DOWNLOAD_STATUS subtitleDownloadStatus =
m_sdkPlayer.getDownloadStatus(VO_OSMP_SOURCE_STREAMTYPE.VO_OSMP_SS
_SUBTITLE);
```

3.1.2.8.10 Video/Audio decoding bitrate

During playback, you can get the video and audio decoding bitrate of the streaming being played in the **SpecialFeatures** page.

3.1.2.8.11 Multiple Instances

During playback, tap **Start** next o **Multi Instance** to display a second instance of the player on the top of the first player simultaneously (picture in picture), as shown in Figure 36.

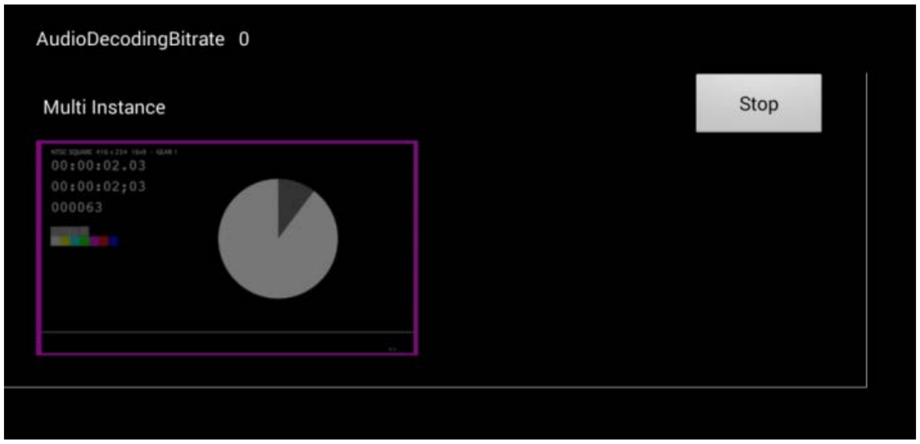


Figure 36. Enable multiple instances

3.1.2.9 Exiting the application

When the media playback is complete, the Sample Player returns to the initial view and you can type or select a new media source. The API calling sequence for exiting playback is *stop()*, *close()*, and then *destroy().*

If **Home** is used to leave the application during playback, the Sample Player pauses the playback and keeps the media source open. Playback resumes when user returns the application.

Refer to *suspend(false)* and *resume(SurfaceView view)*for API related information.

Note: Refer to Section 3 (Basic Integration) of the *OnStream MediaPlayer+ Player SDK Integration Guide for Android* for details about stopping playback.

3.2 iOS

3.2.1 Installation

This section describes the installation of the Sample Player application included with the SDK on iOS.

We recommend that you install the Sample Player application in the latest Xcode version. See 1.2 Supported operating system and web browser for the iOS versions supported by OSMP+.

3.2.1.1 Importing the SamplePlayer project into Xcode

To import the SamplePlayer project into Xcode:

1. In **Finder**, browse to the *<SDK_INSTALL_DIR>/iOS/SamplePlayer* directory, as shown in Figure 37.

Note:
Application Transport Security blocks the cleartext HTTP on iOS 9. To ensure that the content can be played back on iOS 9 devices, upgrade Xcode to version 7 and add the following code in the *plist.info* file.

```
<key>NSAppTransportSecurity</key>
<dict>
  <key>NSAllowsArbitraryLoads</key>
  <true/></dict>
```

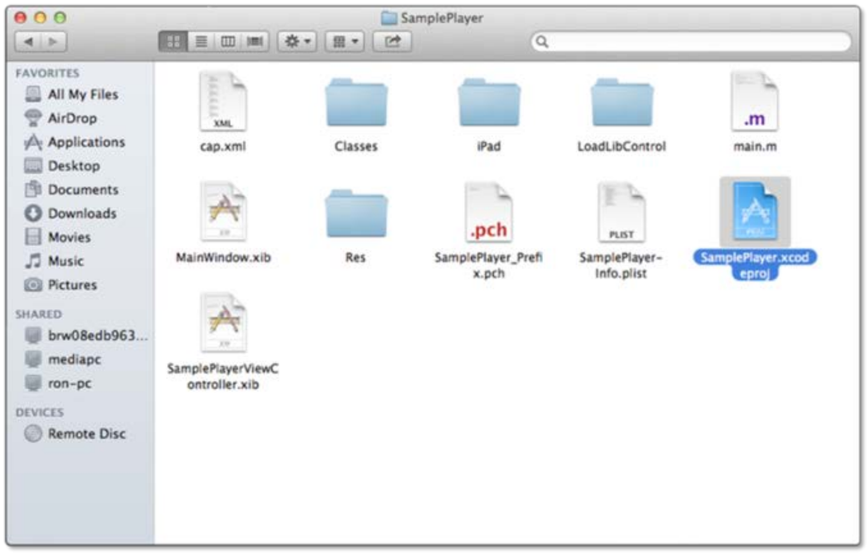
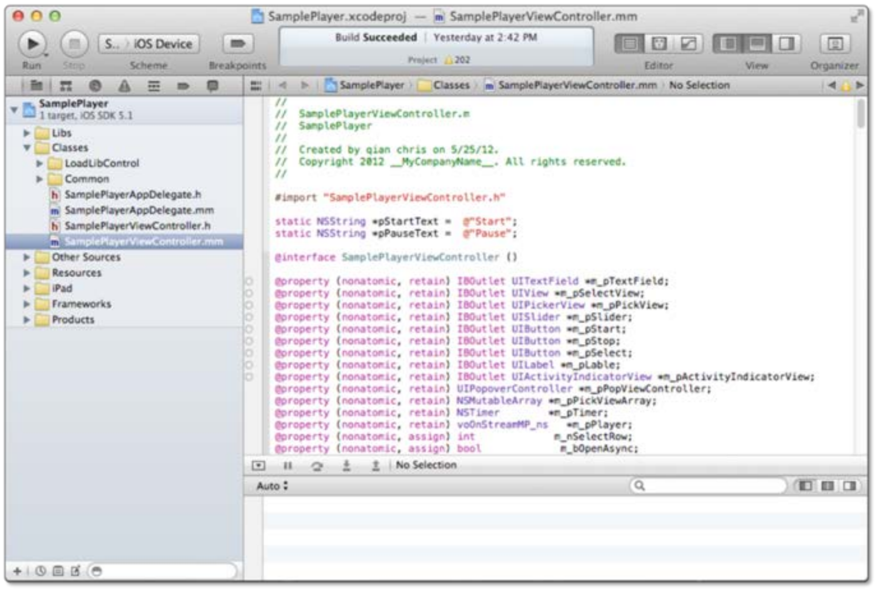


Figure 37. Sample Player Project in Finder

2. Double click **SamplePlayer.xcodeproj** to open the project in Xcode, as shown in Figure 38.


SamplePlayer.xcodeproj — SamplePlayerViewController.mm
Run
Stop
S.. iOS Device
Scheme
Breakpoints
Build Succeeded | Yesterday at 2:42 PM
Project 202
Editor
View
Organizer
SamplePlayer
1 target, iOS SDK 5.1
Libs
Classes
LoadLibControl
Common
SamplePlayerAppDelegate.h
SamplePlayerAppDelegate.mm
SamplePlayerViewController.h
SamplePlayerViewController.mm
Other Sources
Resources
iPad
Frameworks
Products
SamplePlayer
Classes
SamplePlayerViewController.mm
No Selection
No Selection
Auto

```
//
//  SamplePlayerViewController.m
//  SamplePlayer
//
//  Created by qian chris on 5/25/12.
//  Copyright 2012 __MyCompanyName__. All rights reserved.
//

#import "SamplePlayerViewController.h"

static NSString *pStartText =  @"Start";
static NSString *pPauseText =  @"Pause";

@interface SamplePlayerViewController ()

@property (nonatomic, retain) IBOutlet UITextField *m_pTextField;
@property (nonatomic, retain) IBOutlet UIView *m_pSelectView;
@property (nonatomic, retain) IBOutlet UIPickerView *m_pPickView;
@property (nonatomic, retain) IBOutlet UISlider *m_pSlider;
@property (nonatomic, retain) IBOutlet UIButton *m_pStart;
@property (nonatomic, retain) IBOutlet UIButton *m_pStop;
@property (nonatomic, retain) IBOutlet UIButton *m_pSelect;
@property (nonatomic, retain) IBOutlet UILabel *m_pLable;
@property (nonatomic, retain) IBOutlet UIActivityIndicatorView *m_pActivityIndicatorView;
@property (nonatomic, retain) UIPopoverController *m_pPopViewController;
@property (nonatomic, retain) NSMutableArray *m_pPickViewArray;
@property (nonatomic, retain) NSTimer        *m_pTimer;
@property (nonatomic, retain) voOnStreamMP_ns    *m_pPlayer;
@property (nonatomic, assign) int              m_nSelectRow;
@property (nonatomic, assign) bool              m_bOpenAsync;
```

Figure 38. Xcode IDE with Imported SamplePlayer Project

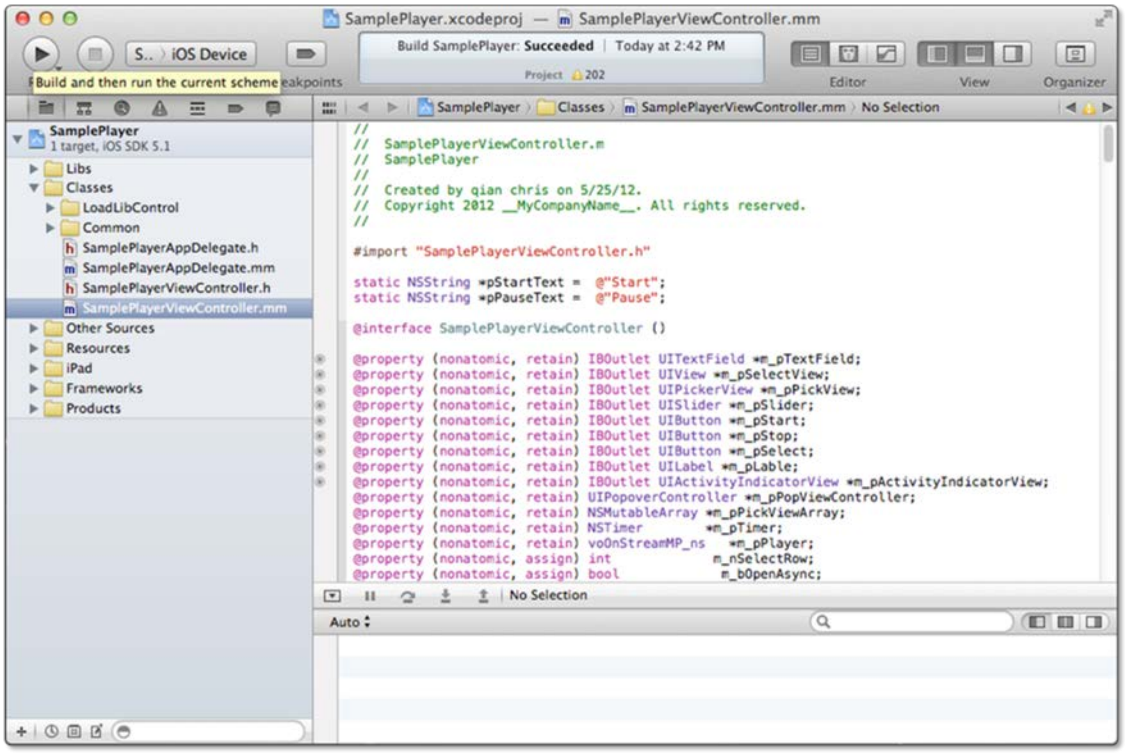
After importing the Sample Player project into Xcode, its files are visible in the **Navigator** area. The SDK header and library files have been added to the SamplePlayer project. You are ready to build the project and run the application on an iOS device.

3.2.1.2 Running the Sample Player on a device

To run the Sample Player on an iOS device:

Note: You must ignore Xcode's recommendation and select the target. Also, change the default *com.visualon.SamplePlayer* to the name of your developer certificate.

1. Connect the device to the Xcode workstation with a USB cable.
2. In Xcode, ensure that the **Scheme** is set to use the connected device.
3. Click **Run** to build and run it on the iOS device, as shown in Figure 39. The Sample Player application can be explored on the iOS device.


SamplePlayer.xcodeproj — SamplePlayerViewController.mm
iOS Device
Build and then run the current scheme
Build SamplePlayer: Succeeded | Today at 2:42 PM
Project 202
Editor
View
Organizer
SamplePlayer
1 target, iOS SDK 5.1
Libs
Classes
LoadLibControl
Common
SamplePlayerAppDelegate.h
SamplePlayerAppDelegate.mm
SamplePlayerViewController.h
SamplePlayerViewController.mm
Other Sources
Resources
iPad
Frameworks
Products
SamplePlayer
Classes
SamplePlayerViewController.mm
No Selection
Auto

```
//
//  SamplePlayerViewController.m
//  SamplePlayer
//
//  Created by qian chris on 5/25/12.
//  Copyright 2012 __MyCompanyName__. All rights reserved.
//

#import "SamplePlayerViewController.h"

static NSString *pStartText =  @"Start";
static NSString *pPauseText =  @"Pause";

@interface SamplePlayerViewController ()

@property (nonatomic, retain) IBOutlet UITextField *m_pTextField;
@property (nonatomic, retain) IBOutlet UIView *m_pSelectView;
@property (nonatomic, retain) IBOutlet UIPickerView *m_pPickView;
@property (nonatomic, retain) IBOutlet UISlider *m_pSlider;
@property (nonatomic, retain) IBOutlet UIButton *m_pStart;
@property (nonatomic, retain) IBOutlet UIButton *m_pStop;
@property (nonatomic, retain) IBOutlet UIButton *m_pSelect;
@property (nonatomic, retain) IBOutlet UILabel *m_pLable;
@property (nonatomic, retain) IBOutlet UIActivityIndicatorView *m_pActivityIndicatorView;
@property (nonatomic, retain) UIPopoverController *m_pPopViewController;
@property (nonatomic, retain) NSMutableArray *m_pPickViewArray;
@property (nonatomic, retain) NSTimer        *m_pTimer;
@property (nonatomic, retain) voOnStreamMP_ns   *m_pPlayer;
@property (nonatomic, assign) int              m_nSelectRow;
@property (nonatomic, assign) bool              m_bOpenAsync;
```

Figure 39. Xcode IDE with Imported Sample Player Project

3.2.1.3 Uninstalling the Sample Player on a device

To uninstall the Sample Player on an iOS device:

1. Press and hold the **Sample Player** icon for a few seconds.
2. The icon starts to wobble gently, and a small cross appears on the top right corner of the icon.
3. Tap the cross, and then tap **Delete** to confirm the uninstallation.

3.2.2 Sample Player usage

This section describes the usage of the Sample Player application included with the SDK. The Sample Player application implements a fully integrated media player with the following features:

- Select the media source
- Select the subtitle source
- Select the option for opening the media source, including Synchronous and Asynchronous
- Set the preferred audio and subtitle languages
- Control the playback through buttons
- Control the sound volume and mute/unmute the sound
- Support the Closed Captions and subtitle rendering
- Configure the asset properties

3.2.2.1 Media sources

The Sample Player application allows you to supply media files or text (*.txt) lists of media URLs for playback. The text file lists each link URL (including *http://*) separated by line.

To add media sources to the Sample Player application:

1. Connect the iOS device to the workstation with a USB cable.
2. Run iTunes and select the device under **DEVICES**.
3. Select **Apps**, and then select **SamplePlayer** to show a list of documents to be synchronized.
4. Drag the media files and/or *.txt media source list files into the SamplePlayer Documents list.
5. Click **Sync** to synchronize the file with the iOS device, as shown in Figure 40. The media sources are available for selection when you run the SamplePlayer application.

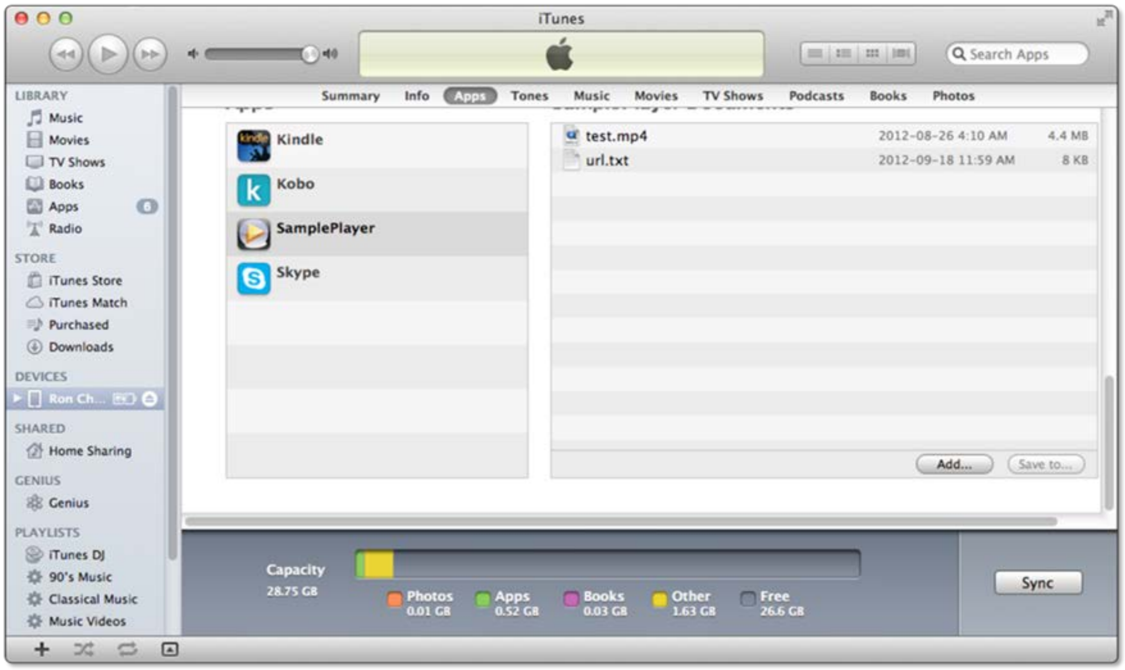


Figure 40. iTunes Apps View with SamplePlayer Documents

3.2.2.2 Initial view

The initial view of the Sample Player, as shown in Figure 41, provides the user interface (UI) to select the media source.

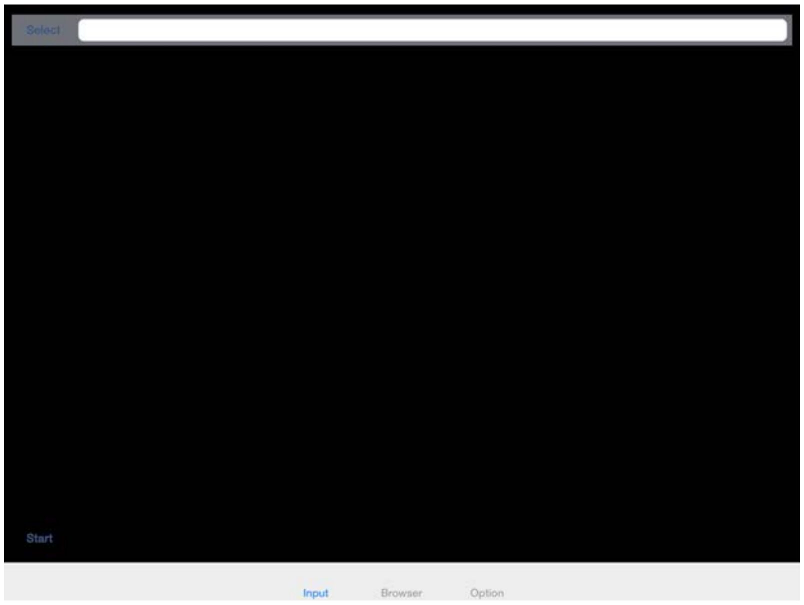


Figure 41. Initial View of the Sample Player

3.2.2.1.1 Opening media source

Media source can be opened in either Sync or Async mode. The Sample Player uses Async mode by default. For sample code of opening media source, refer to 4.2.3 Opening media source.

- Sync mode: Opening the media source in sync mode causes the client application to block until the open method completes or returns.
- Async mode: In Async mode, the API call returns immediately, and the client application is not blocked waiting for the media source to be opened. The client application can then wait for an event to indicate that the media source was opened successfully before starting the media playback.

To activate the asynchronous option:

1. In the initial view tap **Option**.
2. Slide the **Asynchronous/Synchronous Open** radio button to open source asynchronously.

3.2.2.2.2 Selecting the Media Source

The initial view allows you to select the media source (for example, H.264, MP4 file, or HTTP Live Streaming link and so on) for playback. The media source path can be inputted manually or be selected from the list of media source previously added to the application in the *documents* folder.

To manually type a media file or link:

1. Tap **Browser** to enter the full URL starting with *http://* in the text box.
2. Tap **Start**.

Alternately, a list of media files and links can be supplied to the Sample Player through a text file. The text list must be saved to */documents/url.txt*, with each link URL and file path separated by line. To select a media file or link from *url.txt*:

1. Tap **Select** to display the media source list and tap the required source file or URL. Tap anywhere outside the source list to close the list.
2. Tap **Start**.

3.2.2.2.3 Starting Playback

Tap **Start** to start playback. Refer to 4.2.5 Starting playback for the sample code of starting playback. Refer to Section 3 (Basic Integration) in *the OnStream MediaPlayer+ Player SDK Integration Guide for iOS*, for details about initializing the SDK player, opening the media source, and beginning playback.

3.2.2.2.4 Enabling AirPlay

Tap the AirPlay button ▲ in the initial view, select iPad/iPhone to display the content on the iOS device screen. If you want to stream the content to the external device screen, tap the AirPlay button, and then tap the name of the device to which you want to stream content.

Note: AirPlay is only applicable when AVPlayer engine is enabled. Refer to 3.2.2.3.1 Engine type for details about how to turn on AVPlayer engine.

3.2.2.3 Options

3.2.2.3.1 Engine type

The Sample Player application provides an option to initialize the SDK by using the AVPlayer engine or VisualOn VOME2 software engine (default).

Sample code of initializing the SDK is described in 4.2.2 Initializing SDK instance.

To use the AVPlayer engine:

1. Tap **Option**, and then tap **Engine Type**.
2. Select **AVPlayer** from the **Engine Type** list, and then tap **Done**.

3.2.2.3.2 Video Decoder type

The OSMP+ SDK uses Video Toolbox to resolve the performance issue caused by the AVPlayer decoder. To achieve the better rendering performance, Video Toolbox decoder is recommended being used on iOS devices. Visit Apple Developer website for more information about Video Toolbox.

To use the Video Toolbox decoder:

1. Tap **Option**, and then tap **Video Decoder Type**.
2. Select **AutoHW** from the **Engine Type** list, and then tap **Done**.

3.2.2.3.3 Dolby Surround Module

The Sample Player has a Dolby Surround Module to provide the better audio effects. Tap **Dolby Surround Module** from **Option** to load the module. Refer to *enableAudioEffect* for API related information.

3.2.2.3.4 Enabling/disabling deblock

Deblocking is a feature applied to decoded compressed video to improve visual quality and prediction performance by smoothing the sharp edges which can form between macroblocks when block coding techniques are used.

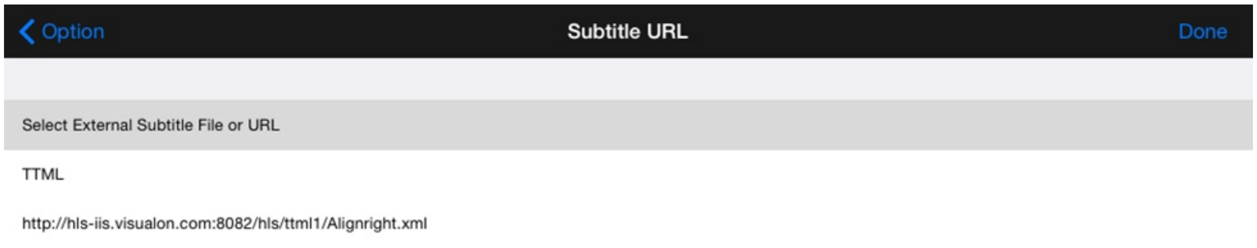
Enabling the deblock function helps improve the playback quality, especially when the video has mosaic. Tap **Deblock** from **Option** to enable the deblock feature. Refer to *enableDeblock* for API related information.

3.2.2.3.5 Selecting the Subtitle Source

The Sample Player allows you to optionally select an external subtitle source (for example, CEA 608, CEA 708, SRT, SMI, etc.) for Closed Captions (CC) or subtitles rendering. A list of subtitle files and links can be supplied to the Sample Player through a text file. The text list must be saved to */documents/subtitle.txt*, with each link URL and file path separated by a line. Refer to *setSubtitlePath* for API related information.

To select an external subtitle file or link from *subtitle.txt*:

1. Tap **Option**, and then tap **Subtitle URL**.
2. Select a URL on */documents/subtitle.txt* list menu, as shown in Figure 42.



Select External Subtitle File or URL

TTML

http://hls-iis.visualon.com:8082/hls/ttml1/Alignright.xml

Figure 42. Select an external subtitle

3. Select the required subtitle source, and then tap **Done**.

3.2.2.2.6 Setting buffering time

The OSMP+ player allows you to set the buffering time of the source to be played.

3.2.2.2.6.1 Initial Buffering Time

Turn on **Initial Buffering Time** option to set the buffering time of the source in milliseconds before the playback starts. Refer to *setInitialBufferingTime(int time)* for API related information.

To enable the initial buffering time of source buffer:

1. Tap **Initial Buffering Time** from **Option**.
2. Type the number to set the initial buffering time in milliseconds.

3.2.2.2.6.2 Setting Max Buffer Time

Refer to *setMaxBufferingTime* for API related information. To set the maximum time of source buffer:

1. Type the number in the **Max Buffer Time** box from **Option** to set the maximum buffer time in milliseconds.
2. Tap **Done**.

3.2.2.2.6.3 Playback Buffering Time

When re-buffering is needed during playback, the player enables you to set the buffering time with the **Playback Buffering Time** option. Refer to *setPlaybackBuferingTime(int time)* for API related information.

To set the re-buffering time:

1. Tap **Playback Buffering Time** from **Option**.
2. Type the number to set the time in milliseconds.

3.2.2.3.7 Enabling Download Manager

The video files can be downloaded locally while playing when the Download manager is enabled. This feature is only for adaptive streaming. Tap **Option**, and then tap **Download** to enable Download Manager. For details about how to integrate Download Manager on Android, refer to the *OnStream MediaPlayer+.Download Manager Integration Guide for Android.pdf* document in your installation package.

If Download Manager is turned on, the **Select Asset** dialog prompts when starting the playback. Specify the Video, Audio, and Subtitle tracks to be downloaded, and then tap **OK** to download the source to be played. If you do not want to keep the downloaded files on your device, tap **Delete Downloader File** from **Options**, tap **Delete All** to remove all downloaded files, or select the file to be removed and then tap **Delete**.

3.2.2.3.8 Enabling CPU Adaption

The Bitrate Adaption of Sample Player subjects to setting which is hardcoded in the player when enabling CPU Adaption. Refer to *enableCPUAdaptation* for API related information.

To enable CPU Adaption, tap **Option**, and then tap **CPU Adaption**.

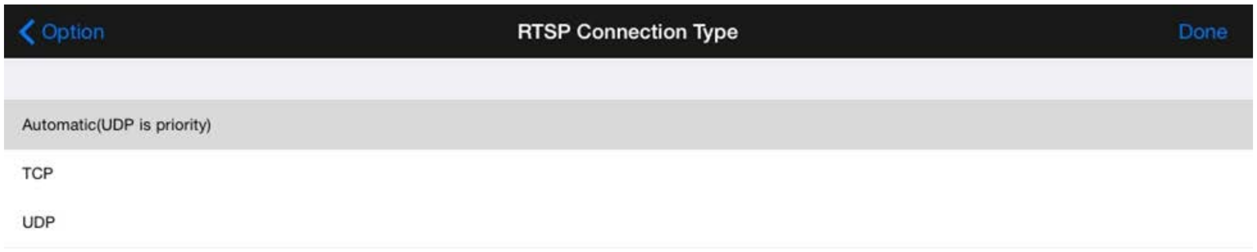
3.2.2.3.9 RTSP Setting

For the sample code regarding RTSP settings, refer to 4.2.14 RTSP settings (Optional).

3.2.2.3.9.1 Setting RTSP Connection Type

To set RTSP Connection Type:

1. Tap **RTSP Connection Type** from **Option**.
2. Three connection types, **Automatic(UDP is priority), TCP**, and **UDP** are available for selection, as shown in Figure 43.
   - **Automatic (UDP is priority):** The player selects the connection types automatically.
   - **TCP**: refer to 1.3 Terminology for TCP definition.
   - **UDP**: refer to 1.3 Terminology for UDP definition.



Automatic(UDP is priority)
TCP
UDP

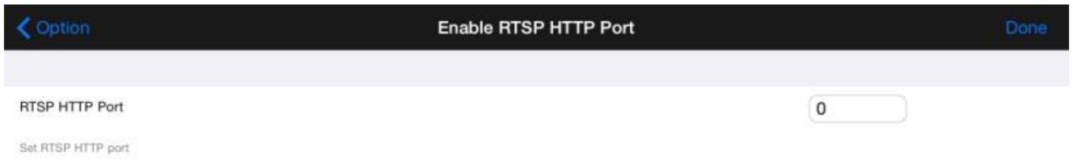
Figure 43. Available RTSP Connection Types

3. Select the required connection type, and then tap **Done**.

3.2.2.3.9.2 Enabling RTSP HTTP Port

Tap **Option**, and then tap **Enable RTSP HTTP Port** to enable it. Refer to *enableRTSPOverHTTP* for API related information.

Tap **Set to enable RTSP HTTP Port**, and then type a value to define the Port to access the RTSP media source, as shown in Figure 44. Refer to *setRTSPConnectionPort* for API related information.



RTSP HTTP Port 0
Set RTSP HTTP port

Figure 44. Define the port

3.2.2.3.9.3 Enabling Low Latency Video

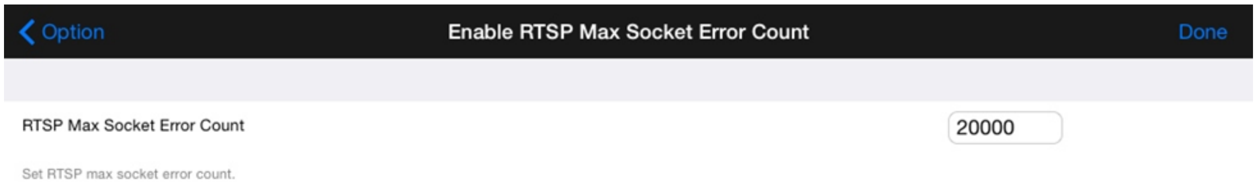
The Low Latency Video feature allows you to easily build a high quality and low latency HD video transmission system. Refer to *enableLowLatencyVideo* for API related information.

Tap **Option**, and then tap **Enable Low Latency Video** to enable RTSP Camera video latency.

3.2.2.3.9.4 Enabling RTSP Max Socket Error Count

Tap **Option**, and then tap **Enable RTSP Max Socket Error Count** to enable RTSP Socket Error handling. Refer to *setRTSPMaxSocketErrorCount* for API related information.

Tap **Set RTSP max socket error count.**, and then type a value to set the entry level for RTSP Socket error handling, as shown in Figure 45.



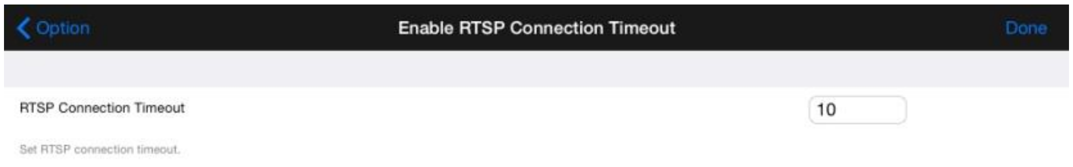
RTSP Max Socket Error Count 20000
Set RTSP max socket error count.

Figure 45. Set the entry level

3.2.2.3.9.5 Enable RTSP Connection Timeout

Tap **Option**, and then tap **Enable RTSP Connection Timeout** to enable the connection timeout function. Refer to *setRTSPConnectionTimeout* for API related information.

Tap **Set RTSTP connection timeout.**, and then type a number to set a value for the retry times when server has no response, as shown in Figure 46.



RTSP Connection Timeout 10

Set RTSP connection timeout.

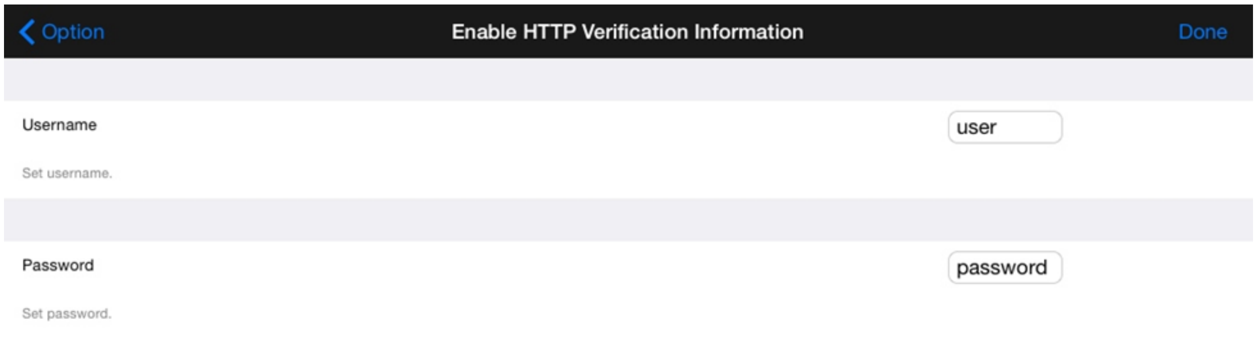
Figure 46. Set the retry times

3.2.2.3.10 HTTP Setting

3.2.3.10.1 Enabling HTTP Verification Information

Tap **Option**, and then tap **Enable HTTP Verification Information**. Refer to *setHTTPVerificationInfo* for API related information.

Tap **Set the HTTP verification information.**, and then type the user name and password to authorize the RTSP media source access, as shown in Figure 47.



Username user

Set username.

Password password

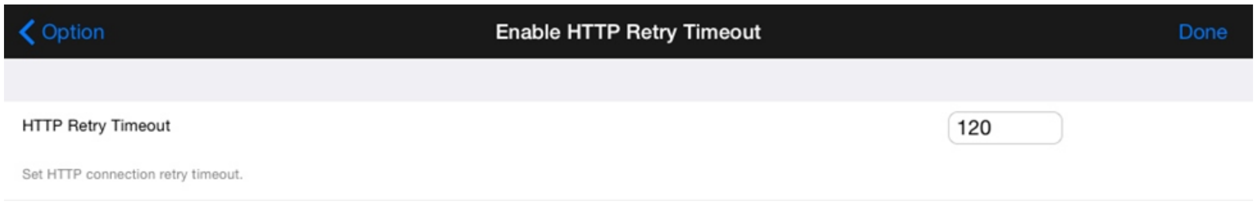
Set password.

Figure 47. Set the user name and password

3.2.3.10.2 Enabling HTTP Retry Timeout

Tap **Option**, and then tap **Enable HTTP Retry Timeout** to enable the retry timeout function. Refer to *setHTTPRetryTimeout* for API related information.

Tap **Enable HTTP connection retry timeout.**, and then type a number to set the player retry times when server has no response, as shown in Figure 48.



HTTP Retry Timeout: 120

Set HTTP connection retry timeout.

Figure 48. Set retry times

3.2.3.10.3 Enabling HTTP Gzip Request

HTTP data is compressed before it is sent from the server: compliant browsers will announce what methods are supported to the server before downloading the correct format; browsers that do not support compliant compression method will download uncompressed data. Gzip is one of the most common compression schemes. Refer to *enableHTTPGzipRequest* for API related information.

Tap **Option**, and then tap **Enable HTTP Gzip Request** to enable or disable the support of the GZIP format.

3.2.3.10.4 Setting token for HTTP Request

The sample player enables to set a token/string for HTTP requests. Refer to *setURLQueryString* for API related information.

Tap **URL Query String** from **Options** to enable or disable the token support for HTTP requests. After enabling this option, tap **Enable URL Query String**, and then type the **Query String** in the prompted dialog.

Note: The query string should be in the format of '*key1=value&key2=value*'.

3.2.2.3.11 Playback Looping

If the playback looping feature is on and you're playing your timeline, the playhead cycles back to the start of the sequence when you reach the end. If it's off, your playhead simply stops at the end of the sequence.

Tap **Option**, and then tap **Playback Looping** to enable or disable the Playback looping.

3.2.2.3.12 Subtitle Settings

Tap **Option**, and then tap **Subtitle Settings** to enable or disable the Subtitle Settings. For API related information, refer to 4.2.9 Subtitle rendering (Optional).

Table 7 lists the available subtitle attributes that can be set in the OSMP+ player and their related API s.

Table 7. Subtitle attributes and related APIs

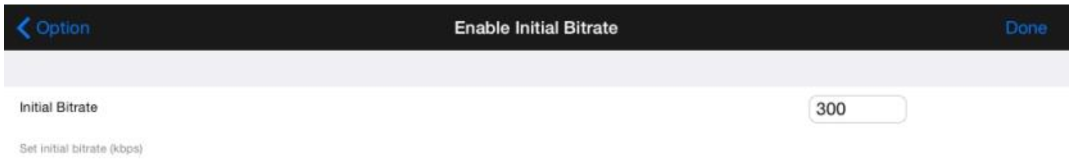
Attribute	**Definition**	**API**
Enable Subtitle	Enable/disable subtitles display. Disabled is set by default.	*enableSubtitle()*
Enable Default Settings	Use the subtitle settings from source stream	N/A
Reset Subtitle Settings	Reset all parameters to the default values that are specified in source stream.	*resetSubtitleParameter()*
Enable Subtitle Bounding Box • Set Top Percent • Set Left Percent • Set Bottom Percent • Set Right Percent	Set the subtitle bounding box, which will overwrite the subtitle settings from parser.	*setSubtitleBoundingBox()*
Enable Subtitle Gravity • Set Horizontal Position • Set Vertical Position	Set the gravity of the bounding box.	*setSubtitleGravity()*
Enable Subtitle Trim	Set the set of characters to be trimmed from the beginning or end of subtitle.	*setSubtitleTrim()*
Enable Subtitle Auto Adjustment	Enable/Disable the automatic adjustment for subtitle position/font size. Disabled is set by default.	*enableSubtitleAutoAdjustment()*
Enable Font size	Set subtitle font size scale.	*setSubtitleFontSizeScale()*
Enable Font Color Opacity	Set subtitle font color opacity rate.	*setSubtitleFontOpacity()*
Enable Font Color List	Set subtitle font color.	*setSubtitleFontColor()*
Enable Background Color Opacity	Set subtitle font background color opacity rate.	*setSubtitleFontBackgroundOpacity()*
Enable Background Color List	Set subtitle font background color.	*setSubtitleFontBackgroundColor()*
Enable Edge Color	Set subtitle font edge color	*setSubtitleFontEdgeOpacity()*

Attribute	Definition	API
Opacity	opacity rate.	
Enable Edge Color List	Set subtitle font edge color.	*setSubtitleFontEdgeColor()*
Enable Edge Type List	Set subtitle font edge type.	*setSubtitleFontEdgeType()*
Enable Font List	Set subtitle font name.	*setSubtitleFontName()*
Enable Window Background Color Opacity	Set window background color opacity rate.	*setSubtitleWindowBackgroundOpacity()*
Enable Window Background Color List	Set window background color.	*setSubtitleWindowBackgroundColor()*
Enable Underline Font	Enable or disable the underline font for subtitle.	*setSubtitleFontUnderline()*
Enable Bold Font	Enable or disable the bold font for subtitle.	*setSubtitleFontBold()*
Enable Italic Font	Enable or disable the italic font for subtitle.	*setSubtitleFontItalic()*

3.2.2.3.13 Enabling Initial Bitrate

The OSMP+ player enables to select a suitable initial bitrate, which can utilize the TCP throughput to achieve a higher-than-lowest bitrate if possible. By providing a better picture quality, the quality of experience can be improved. Tap **Option**, and then tap **Enable Initial Bitrate** to enable or disable the Initial Bitrate setting. Refer to *setInitialBitrate* for API related information.

Tap **Set to enable initial bitrate**, type a value for the Initial Bitrate, and then Tap **Done**, as shown in Figure 49.



Initial Bitrate 300

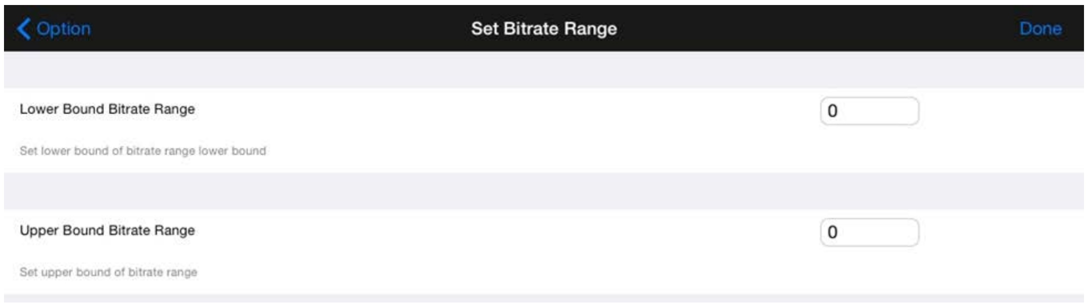
Set initial bitrate (kbps)

Figure 49. Set the value of Initial Bitrate

3.2.2.3.14 Setting Bitrate Range

Tap **Option**, and then tap **Set Bitrate Range** to enable or disable the Bitrate Range Settings. Refer to *setBitrateThreshold* for API related information.

Tap **Set bitrate range (kbps)**, type the number for lower and upper bound bitrate range, and then tap **Done**, as shown in Figure 50.

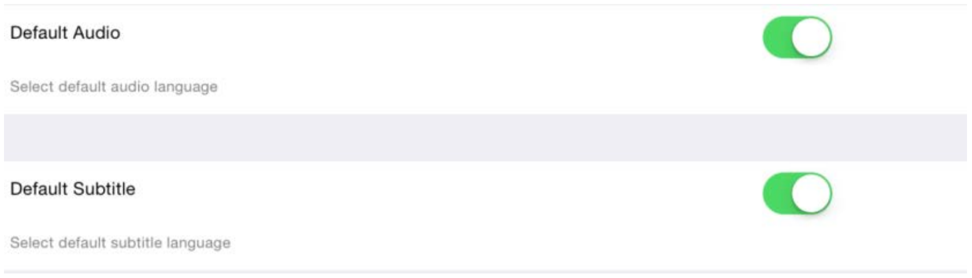


Lower Bound Bitrate Range 0
Set lower bound of bitrate range lower bound
Upper Bound Bitrate Range 0
Set upper bound of bitrate range

Figure 50. Set the lower and upper boundary value

3.2.2.3.15 Default Audio and Subtitle

Tap **Option**, and then tap **Default Audio/Default Subtitle** to enable or disable the Default Audio and Subtitle settings, as shown in Figure 51. Refer to *setPreferredAudioLanguage* or *setPreferredSubtitleLanguage* for API related information.



Default Audio
Select default audio language
Default Subtitle
Select default subtitle language

Figure 51. Enable or disable settings of audio and subtitle

3.2.2.2.16 Anti Mirror

The OSMP+ player allows user to enable or disable the anti-mirroring mode on iOS devices. Slide to enable **Anti Mirror** from **Option**. Refer to **enableAntiMirror()** for API related information.

3.2.2.2.17 Preference settings

The OSMP+ player allows user to set the preference of playback. Select **Set preferences** from **Options** to enable the following options. Refer to *setPreference()* for API related information.

3.2.2.2.17.1 Keeping Last Frame

During the transition from live channel to NTS media, the OSMP+ player keeps the last frame of the live channel being played until the first frame of NTS media is played, when the **Keeping Last Frame** option is turned on. In the meantime, the OSMP+ player keeps the screen black when stopping or closing any playback.

3.2.2.2.17.2 Seek Precise

The OSMP+ player provides a new option, **Seek Precise**, for smooth streaming to allow user setting the offset to the live point, so that user can seek to the play point based on this offset value precisely.

3.2.2.2.17.3 Select Audio Switch Immediately

The option, **Select Audio Switch Immediately**, is added into the OSMP+ player to continue play the previous content and then switch to the selected audio track when the previous playback is complete, when this option is turned off. This option is turned on by default.

3.2.2.2.18 Set presentation delay time

The OSMP+ player provides the **Set presentation delay time** option for HLS and Smooth Streaming, to allow user downloading chunks with some seconds of delay that is defined by this option. Refer to *setPresentationDelay()* for API related information.

3.2.2.2.19 Enter NTS

The OSMP+ player provides the Enter Network Time Shifting (NTS) feature to test resuming playback from the pause point, rather than the live point.

The Enter NTS feature requires the collaboration within the streaming server, application, and OSMP+ SDK. Streaming server should create the NTS stream once the user pause live stream playback, and notify the application of the new URL of NTS stream.

The application should call OSMP+ SDK API to switch to NTS stream once the user resumes playback (after perform the pause operation for a while). Also the application should call OSMP+ SDK API to switch back to Live stream once the user catches up with the Live content (typically after a seek operation).

The requirements to use this features are:

- The streams support #EXT-X-PROGRAM-DATE-TIME HLS
- The playlist of the live stream at the time of pause and the playlist of the NTS stream at the time of resume should have the same #EXT-X-PROGRAM-DATE-TIME HLS value

In other words, there are two streams, a live stream that is paused and a NTS stream that is resumed. Both streams share the same content.

The Enter NTS feature enables playback of the NTS stream instead of the live stream when resuming playback. To use this feature, enter the value of the *PROGRAM-DATE-TIME HLS* tag (when the live stream was paused) in the **Min pos** box. **Min Pos** needs to be converted into millisecond.

After clicking **EnterNTS**, the player starts playback at the HLS Live paused position using the NTS stream. To verify that the resume operation is successful, one can check that the UTC time display should not change more than one second when playback resumes.

To enable the Enter NTS playback:

1. Select **Enter NTS** from **Option**.
2. In the **NTS URL** box, type the URL address of the NTS stream that shares the same content with the live stream being played.
3. Convert the UTC time to unix timestamp, and then type this converted unix timestamp in the **NTS min position** box. The unit of Min pos is millisecond.

3.2.2.2.20 Previous seek to

The OSMP+ player supports to play the content from a specified position. To enable the previous seek to feature, tap **Previous seek to** from **Option**, and then specify a position in the unit of ms in the **Seek To Value** dialog.

3.2.2.2.21 Spherical Video

The sample player supports the playback of the content that is recorded by using the 360 degree camera. To enable the decoding of 360 degree video, tap **Spherical Video** from **Option**. For more information, refer to the *setSphericalVideoView* function in the API Reference Manual.

3.2.2.2.22 Enable Live Streaming DVR Position

The OSMP+ player supports defining position based on live streaming DVR window. DVR window is also referred to as the duration of live streaming. To enable defining the position, tap **Live Streaming DVR Position** from **Option**.

When this option is enabled:

- Value of **Play**: this value is 00:00:00 when playing at the live head. If the play point is not the live point, this value will be a negative value to represent the offset between DVR window and live head.
- Value of **Max**: this value is always 00:00:00 that represents the live head.
- Value of **Min**: this value is (value of Max - DVR window size).

When this option is disabled:

- Value of **Play**: this value is the offset between the play head and the beginning of the playback.
- Value of **Max**: this value is the offset between the live head and the beginning of the playback.
- Value of **Min**: this value is (value of Max - DVR window size).

For more information, refer to *enableLiveStreamingDVRPosition()* in the API Reference Manual.

3.2.2.3.23 Version

Tap **Version** from **Option** to shown the Sample Player version information.

3.2.3.4 Playback controls

During playback, the Sample Player provides a set of basic controls, including a shared Start/Pause button, Stop button, and a Seek bar, as shown in Figure 52.

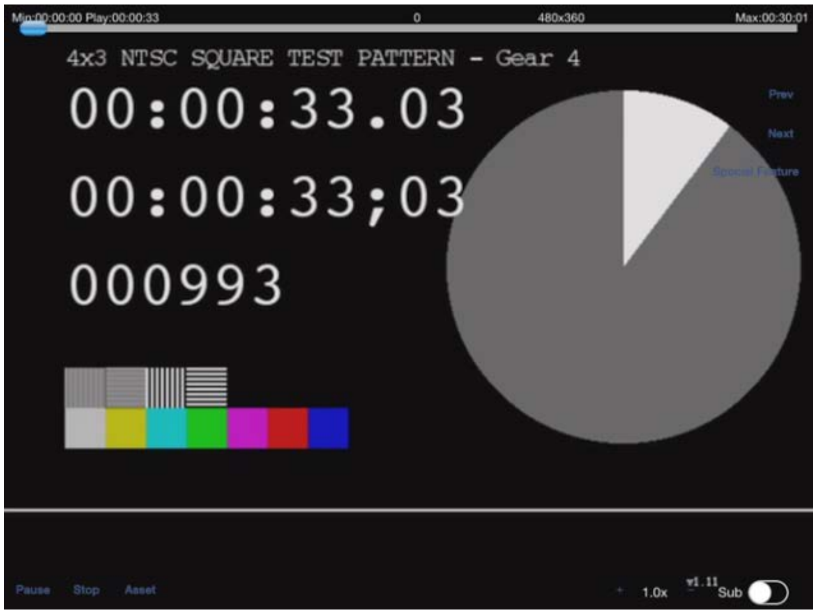


Figure 52. Playback Controls when Media is Playing

Notes:

- Refer to Section 4 (Advanced Integration: Pause/Play Control) of the *OnStream MediaPlayer+ Player SDK Integration Guide for iOS* for details about Pause/Play integration.
- Refer to Section 6 (Advanced Integration: Seekbar Control) of the *OnStream MediaPlayer+ Player SDK Integration Guide for iOS* for details about Seekbar integration.

3.2.3.5 Switching channel

During playback, the Sample Player supports channel switching from the list of added media sources.

To switch channels (sources):

1. Tap **Prev** or **Next** on the right side of the screen to switch to the previous or next media source respectively.

Note: Refer to Section 7 (Advanced Integration: Channel Switching) of the *OnStream MediaPlayer+ Player SDK Integration Guide for iOS* for details about Channel switching.

3.2.3.6 Controlling audio speed

During playback, the Sample Player can control the audio speed. Refer to *setAudioPlaybackSpeed* for API related information.

Tap **+** to speed up the audio playback or **–** to slow down the audio playback, as shown in Figure 53.

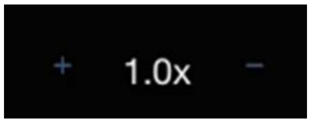


Figure 53. Control audio speed

3.2.3.7 Closed Captions

The Sample Player demonstrates the built-in SDK handling of CC and subtitles rendering. If CC data exists, it appears as shown in Figure 54.

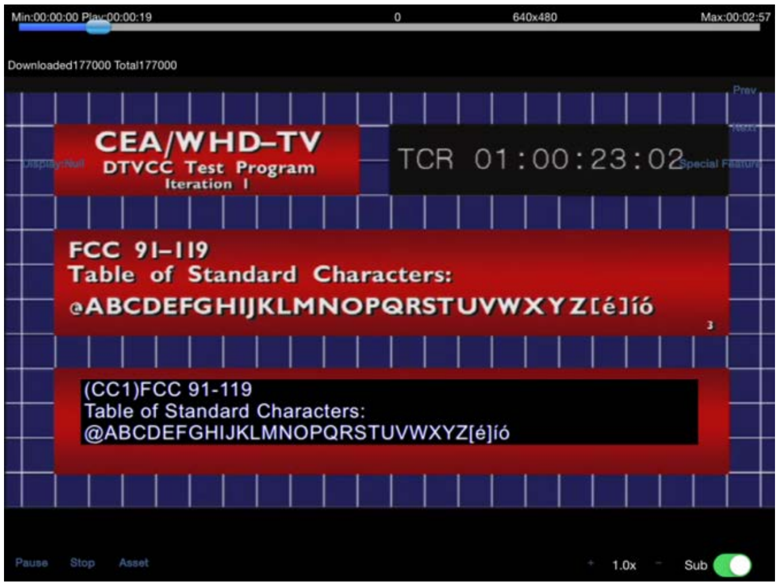

Min:00:00:00 Play:00:00:19
0
640x480
Max:00:02:57
Downloaded177000 Total177000
Prev
CEA/WHD-TV
DTVCC Test Program
Iteration 1
TCR 01:00:23:02
FCC 91-119
Table of Standard Characters:
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[é]íó
(CC1)FCC 91-119
Table of Standard Characters:
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[é]íó
Pause
Stop
Asset
+
1.0x
−
Sub

Figure 54. Sample Player with Closed Captions Display

When CC Show is selected (default), the SDK player automatically handles CC display when data arrives. When CC output is selected, the application manages CC data. Enable or disable the subtitle display by taping **Sub** on lower right corner.

3.2.3.8 Switching Asset properties

If the media source includes multiple video (bit rate/angle), audio, or subtitle (language) tracks, the Sample Player supports property switching during playback.

To switch properties:

1. Tap **Asset** on the bottom of the screen to display the **Asset Selection** dialog.

2. Tap the required Video, Audio, and/or Subtitle track.
3. Tap **Commit** to confirm your selection. Playback continues with your selection, as shown in Figure 55.

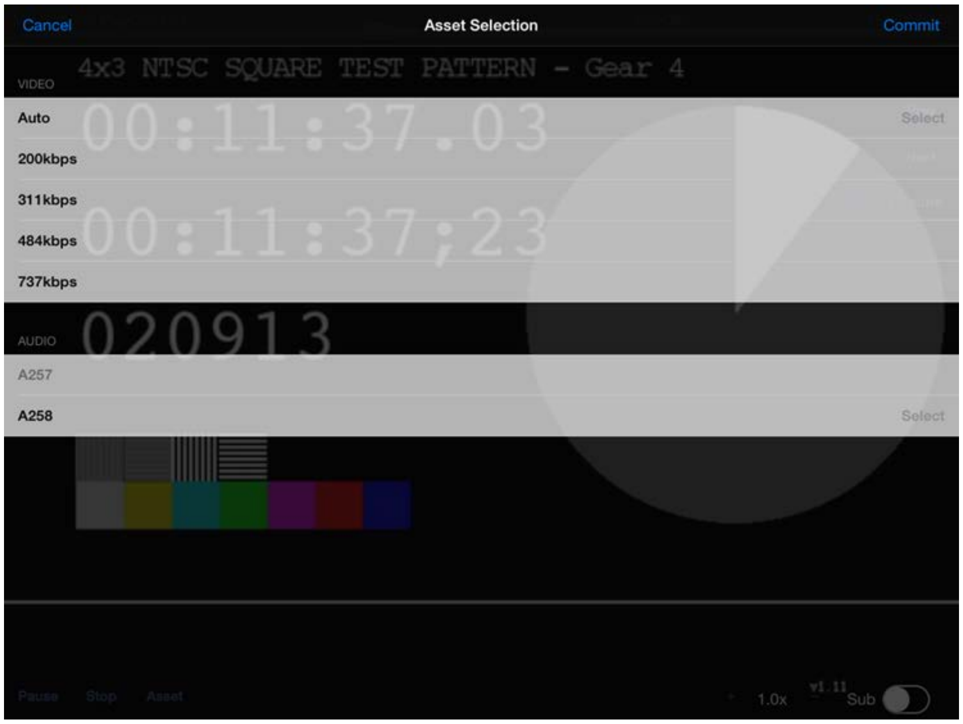


Figure 55. Sample Player Asset Selection Menu

3.2.3.9 Special features

3.2.3.9.1 Screen Brightness

You can adjust the screen Brightness during the playback by dragging the Screen Brightness bar in **Special Feature**, as shown in Figure 56. Refer to *getScreenBrightness* for API related information.

Figure 56. Screen brightness

3.2.3.9.2 Playback Volume

You have several options for controlling volume in the Sample Player, including adjusting the volume level, muting, or unmuting. Refer to *setVolume* for API related information.

To adjust the volume level, drag the playback volume bar and click **Mute** to mute the volume. To restore the audio, click **UnMute**. Refer to *mute* or *unmute* for API related information.

3.2.3.9.3 Setting zoom mode

The Sample Player application allows you to select the zoom mode as needed. To set the zoom mode:

1. Tap **Special Feature**.
2. Tap **Zoom Model**, select the required mode from the following list:
    a. Full
    b. ZoomIn
    c. LetterBox
    d. Original
    e. Pan&Scan

Refer to 1.4.11 Zoom modes for more information about zoom modes.

3.2.3.9.4 Setting the aspect ratio

The Sample Player application allows you to set the aspect ratio as needed. Refer to *setVideoAspectRatio* for API related information.

To set the aspect ratio:

1. Tap **Special Feature**.
2. Tap **Aspect Ratio**, select the required aspect ratio.

3.2.3.9.5 Updating the media source

During playback, use the update source URL function to balance the workload within media servers. This feature is specific for the optimization for content delivery network (CDN) delivery. Refer to *updateSourceURL* for API related information.

Note: The previous URL will be expired a few seconds after the new URL is activated. Master manifest, sub-manifest, and TS chunks of new URL must be exactly same with the previous media file.

1. Tap **Special Feature**.
2. Type the required media URL address into the **Update URL** box, and then tap **Update**, as shown in Figure 57.

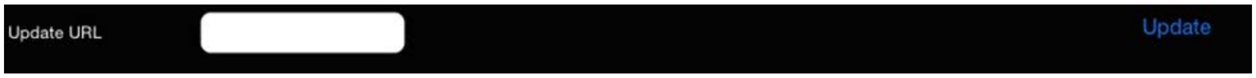


Figure 57. Update the media source

3.2.3.9.6 SEI

During playback, tap the checkbox next to **SEI** to collect the SEI data from the media source. Refer to *startSEINotification* or *enableSEI* for API related information.

3.2.3.9.7 RTSP Statistics

During playback, tap the checkbox next to **RTSP Statistics Setting** to collect the RTSP information from the media source. Refer to *getRTSPStatistics* for API related information.

3.2.3.9.8 Download Status

During playback, tap the checkbox next to **Download Status** to track the HTML5 download information from the media source. Refer to *getDownloadStatus* for API related information.

3.2.3.9.9 Multiple Instances

During playback, enable Multi Instance to display a second player instance on top of the first one simultaneously (picture in picture).

Disable the multiple instances by taping **Stop**, as shown in Figure 58.

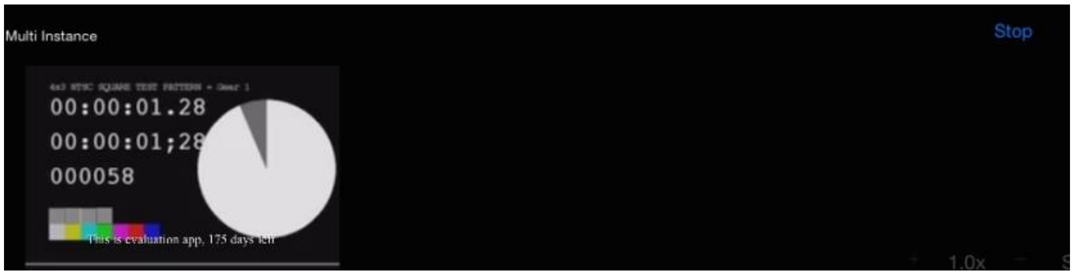


Figure 58. Enable multiple instances

3.2.3.10 Exiting the application

When the media playback is complete, the Sample Player returns the initial view and you can type or select a new media source.

If **Home** is used to leave the application during playback, the Sample Player pauses the playback and keeps the media source open. When the user returns to the application, playback resumes. Refer to *suspend* or *resume* for API related information.

Note: Refer to Section 3 (Basic Integration) of the *OnStream MediaPlayer+ Player SDK Integration Guide for iOS* for details about stopping playback.

3.3 Plug-ins

3.3.1 Installation and uninstallation

This section describes how to install the player plug-ins and how to run the associated SDK Sample Player applications.

The operations for hosting the plug-ins, downloading the plug-ins, and signing the plug-ins may vary depending on supported internet browser. We recommend that you refer to the internet browser developer’s guide or user manual or visit their official website for detailed information on these operations. See 1.2 Supported operating system and web browser for the Windows version and Mac OS version supported by OSMP+.

3.3.1.1 Installing the player plug-ins

The plug-in package provides plug-ins sample installation files for Windows and Mac OS.

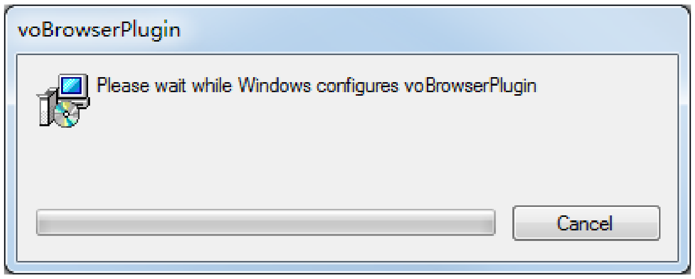
3.3.1.1.1 Installing the Windows plug-in

Note: The evaluation plug-in files are copied to *C:\ProgramData\VisualOn* folder during the installation.

1. Decompress the plug-in package.
2. In Windows Explorer, type or browse to *the <SDK_INSTALL_DIR>* directory where the plug-in package is decompressed.

Note: Ensure that all web browsers are closed and are killed in Task Manager before starting the installation.

3. Select **Bin > Win32**, and then double click the *VisualOnBrowserPlugin_release.msi* file. The installation wizard appears, as shown in Figure 59.


voBrowserPlugin

Please wait while Windows configures voBrowserPlugin

Figure 59. Installation wizard on Windows

4. Follow the onscreen prompts to complete the installation.

3.3.1.1.2 Installing the Mac OS plug-in

1. Decompress the plug-in package.
2. In Finder, browse to the *<SDK_INSTALL_DIR>* directory where the plug-in package is decompressed.

Note: Ensure that all web browsers are closed and are killed in Activity Monitor before starting the installation.

3. Select **Bin > MacOS**, decompress *VisualOnBrowserPluginInstaller_release.zip*.
4. Double click **VisualOnABrowserPluginInstaller.app**. The installation wizard appears, as shown in Figure 60.

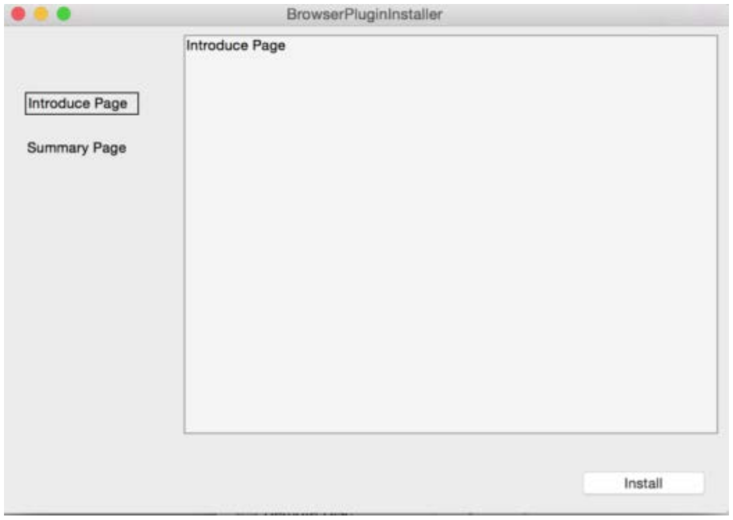


Figure 60. Installation wizard on Mac OS

5. Follow the onscreen prompts to complete the installation.

3.3.1.2 Running Sample Player

3.3.1.2.1 Running on Windows platform

OSMP+ provides the Sample Player with configurable UI for Windows plug-in. See 3.3.2.2 Configurable UI for more information.

Note: This configurable UI is only available for Windows plug-in.

To run the Sample Player application with the configurable UI on a Windows device:

1. In **Windows Explorer**, type or browse to *<SDK_INSTALL_DIR>> Sample > SamplePlayer*.
2. Double click the *SamplePlayer.html* file in Internet Explorer, Chrome, or Firefox.

3.3.1.2.2 Running on Mac OS platform

To run the Sample Player application on a Mac OS device:

1. In **Finder**, browse to the *<SDK_INSTALL_DIR> > Sample > SamplePlayer*.
2. Double click the *SamplePlayer.html* file in Safari.

When opening the Sample Player in the web browser, you may see a message from the browser requesting the permission to run active content or ActiveX content. The message may vary depending on your browser's configuration. Follow the prompts and accept to run ActiveX content.

To quickly verify that the Sample Player is functional, click the http://devimages.apple.com/iphone/samples/bipbop/bipbopall.m3u8 link underneath the video screen. The video should be started after a few seconds.

3.3.1.3 Uninstalling the plug-ins

3.3.1.3.1 Uninstalling the Windows plug-in

1. Open **Control Panel** from the **Start** menu.
2. Select **Programs and Features** to show the Uninstall or change a program page.

Note: Ensure that all web browsers are closed before starting the uninstallation.

3. Select **voBrowserPlugin** from the list, and then click **Uninstall**.

3.3.1.3.2 Uninstalling the Mac OS plug-in

1. Click **Go** on the Finder menu bar.
2. Press **Option** on the keyboard to display the **Library** option.
3. Click **Library** to display the folders on your system.
4. Click **Internet Plug-Ins**.
5. Select *voBrowserPlugin.plugin*, and then select **Move to Trash**.

Or

1. Click **Go** on the Finder menu bar.
2. Select **Terminal** from **Application > Utilities**.
3. In **Terminal**, type the following command to uninstall the plug-in.

```
rm -rf ~/Library/Internet\ Plug-Ins/voBrowserPlugin.plugin/
```

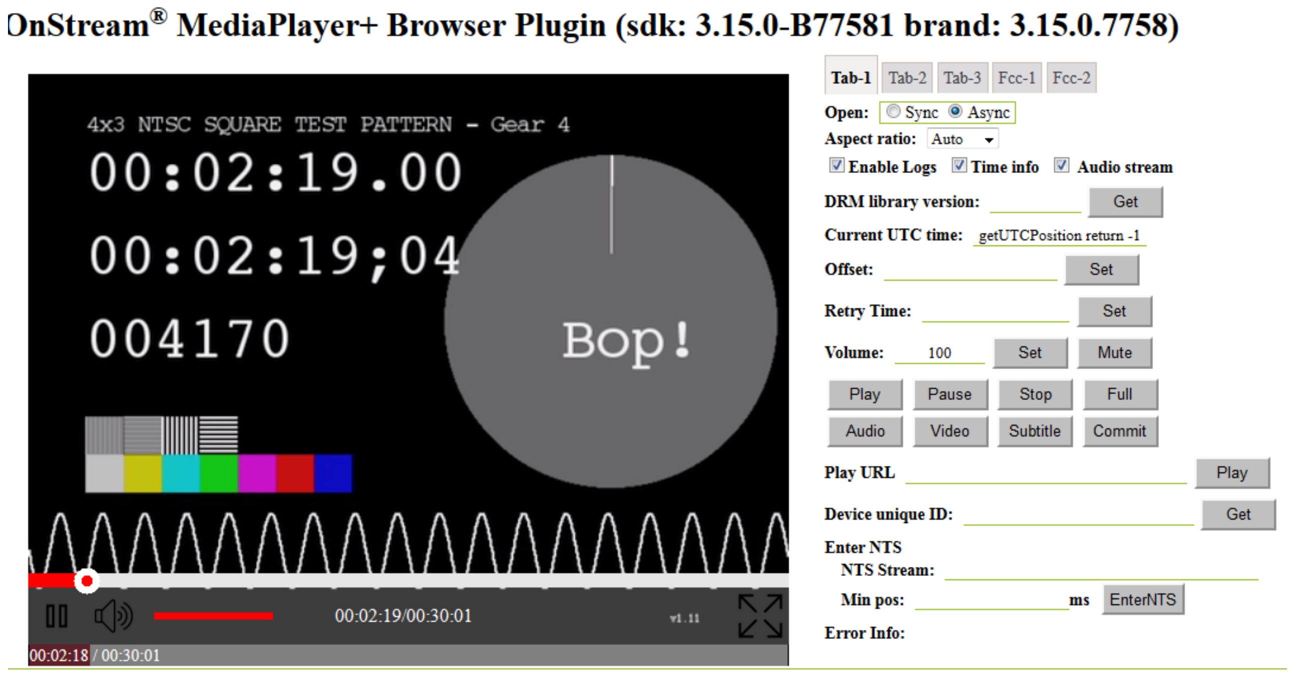
3.3.2 Sample player usage (Windows)

This section describes the use of the Sample Player application integrated with the SDK. The Sample Player application implements a fully integrated media player with the following features:

- Select the media source
- Support the configurable UI for Windows
- Select the subtitle source
- Select the option for opening the media source, including Synchronous and Asynchronous
- Set the preferred audio and subtitle languages
- Control the playback through buttons
- Control the sound volume and mute/unmute the sound
- Support the Closed Captions and subtitle rendering
- Configure the asset properties

3.3.2.1 Initial view

The initial view of the Sample Player application provides users with an interface to select media sources, subtitle sources, and asset properties, as shown in Figure 61.


OnStream® MediaPlayer+ Browser Plugin (sdk: 3.15.0-B77581 brand: 3.15.0.7758)

Open: [ ] Sync [✓ Async]
Aspect ratio: Auto
[✓] Enable Logs [✓] Time info [✓] Audio stream
DRM library version:
Current UTC time: getUTCPosition return -1
Offset:
Retry Time:
Volume: 100
Play URL
Device unique ID:
Enter NTS
NTS Stream:
Min pos: ms
Error Info:

Figure 61. Initial view of the Sample Player

3.3.2.1.1 Selecting the media source

The Sample Player application allows you to select the media source, including video sources and audio sources.

To select the media source:

1. In **Tab-1**, type the required URL address in the **Play URL** box.
   - Type the URL address starting with *http://* or *rtsp://* to play streaming resources on Internet
   - Type the local directory path like *C:\* to play local files
2. Click **Play** to start the playback.

Note: For details about how to initialize the SDK, open the media source, and start the playback, refer to section 4 and section 5 in the *OnStream MediaPlayer+ Player SDK Integration Guide for Windows*.

3.3.2.1.2 Selecting a license file

Setting a new license is optional. To set the license:

1. Click **Tab-2**.
2. Type the required license file in the **License file** box.

3. Click **Set**. The default license file path is *C:\ProgramData\VisualOn\BrowserPlugin\voVidDec.dat*. Renewing a license file or the use of additional features may require additional license files.
4. The new license file takes effect from the next playback.

3.3.2.1.3 Setting the user agent

The Sample Player application allows you to set the user agent according to your requirements.

To set the user agent:

1. Click **Tab-2**.
2. Select **User agent settings**.
3. Type the agent name in the **Name** box as needed. This value is set to **VisualOn OSMP+ Player(Windows)** by default.
4. Type the required agent value in the **Value** box.
5. Click **Set**. The new user agent takes effect from the next playback.

3.3.2.1.4 Setting the proxy

The Sample Player application allows you to set the proxy according to your requirements.

To set the proxy:

1. Click **Tab-2**.
2. Select **Proxy settings.**
3. Type the required agent name in the **Host** box.
4. Type the required agent value in the **Port** box.
5. Click **Set**. The new proxy settings take effect from the next playback.

3.3.2.1.5 HTTP Gzip support

The Sample Player application allows you to enable or disable the support of the GZIP format.

To enable or disable GZIP support:

1. Click **Tab-2**.
2. Select **HTTP gzip request** or not as needed.
   The HTTP gzip support takes effect from the next playback.

3.3.2.1.6 Set the aspect ratio

The Sample Player application allows you to set the aspect ratio as needed.

To set the aspect ratio:

In **Tab-1**, select the required ratio from the **Aspect ratio** list, as shown in Figure 62.

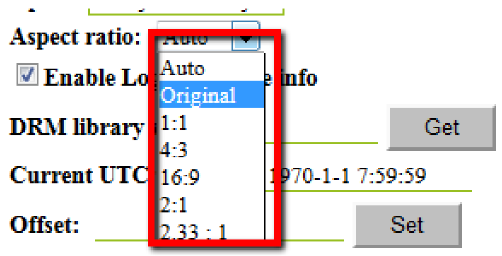


Figure 62. Set the aspect ratio

3.3.2.1.7 Opening media source

Media source can be opened in either Sync or Async mode.

- Sync mode: Opening the media source in sync mode causes the client application to block until the open method completes or returns.
- Async mode: In Async mode, the API call returns immediately, and the client application is not blocked waiting for the media source to be opened. The client application can then wait for an event to indicate that the media source was opened successfully before starting the media playback.

To activate the asynchronous option, select **Async** in **Tab-1**.

3.3.2.2 Configurable UI

When the mouse is over the video screen, the configurable UI appears. The configurable UI includes a shared Start/Pause button, a seek bar, the volume adjustment, and a FullScreen icon, as shown in Figure 63.

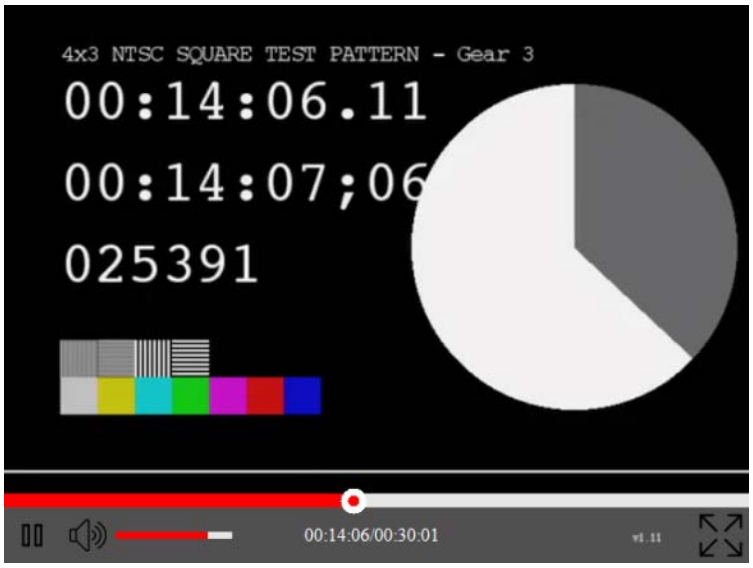


Figure 63. Example of configurable UI

For more information about configurable UI, refer to the *OnStream MediaPlayer+ Plugin UI-Control API Reference* document in your installation package.

3.3.2.3 Playback controls

During playback, the Sample Player application provides a set of basic controls, including a Play button, a Pause button, and a Stop button, as shown in Figure 64.

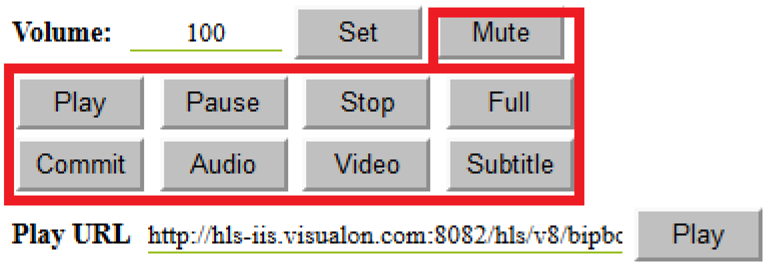


Figure 64. Playback control buttons

3.3.2.3.1 Starting the playback

After initializing the SDK and selecting the media source, click **Play** to start the playback. See 3.3.2.1.1 Selecting the media source for details.

3.3.2.3.2 Pausing the playback

During playback, click **Pause** to stop playing the media temporarily. Click **Play** to resume the playback.

3.3.2.3.3 Stopping the playback

During playback, click **Stop** to stop playing the media.

3.3.2.4 Display mode controls

Full-screen mode makes videos fill the entire screen when you play them.

To play the video full screen:

1. In **Tab-1**, click **Full** or the **View full screen** icon in the lower right corner of the player, as shown in Figure 65.

Note: You might need to resize the window to expose the FullScreen button.

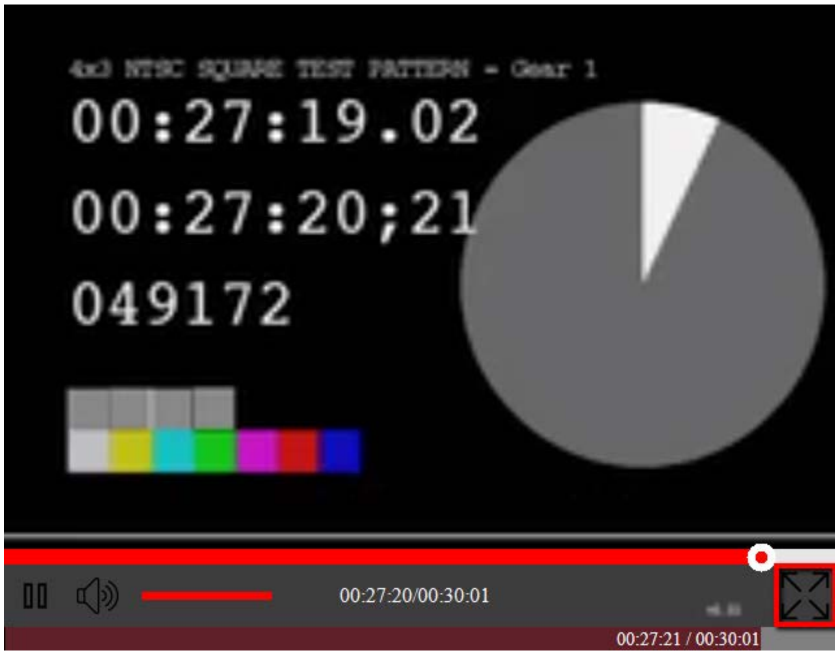


Figure 65. View full screen button

2. To quit the full-screen mode, press **Esc** on the keyboard or the **Exit full screen** icon in the lower right corner.

3.3.2.5 Sound control

You have several options for controlling volume in the Sample Player, including adjusting the volume level, muting, or unmuting.

To adjust the volume level, in **Tab-1**, type the percent number in the **Volume** box, and then click **Set**. The value should be any number from 0 to 100.

In **Tab-1**, click **Mute** to mute the volume. To restore the audio, click **Mute** again.

3.3.2.6 Asset property selections

If the media source includes multiple video tracks (also known as bitrate), audio tracks, or subtitles, the Sample Player application supports switching within various tracks during the playback.

The Sample Player provides a set of buttons to support the property switches, as shown in Figure 66.

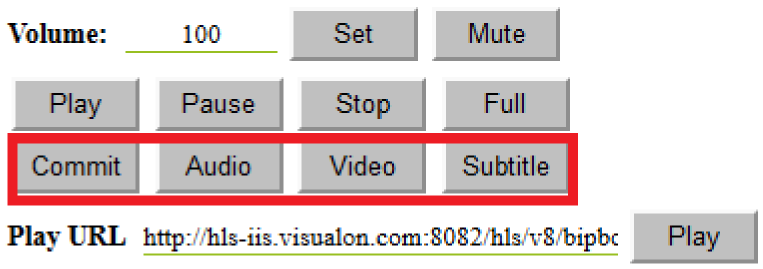


Figure 66. Property switches buttons

3.3.2.6.1 Selecting audios

To select the audio,

1. In **Tab-1**, click **Audio**, and then select the required audio track from the audio list menu.
2. Click **Commit**.

3.3.2.6.2 Setting the preferred audio language

Set your preferred language for the audio track.

1. In **Tab-2**, type your preference in the **Preferred audio language** box.

Note: Type the abbreviated language code that complies with the iso639-2 standard.

---

2. Click **Set**.

3.3.2.6.3 Selecting videos

To select the video,

1. In **Tab-1**, click **Video**, and then select the required video track from the video list menu, as shown in Figure 67.

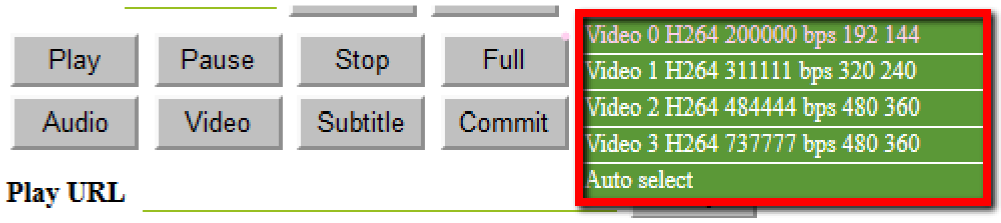

Play
Pause
Stop
Full
Audio
Video
Subtitle
Commit
Play URL
Video 0 H264 200000 bps 192 144
Video 1 H264 311111 bps 320 240
Video 2 H264 484444 bps 480 360
Video 3 H264 737777 bps 480 360
Auto select

Figure 67. Switch the video

2. Click **Commit**.

3.3.2.6.4 Selecting subtitle tracks

To select the subtitle,

1. In **Tab-1**, click **Subtitle**, and then select the required subtitle from the subtitle list menu.
2. Click **Commit**.

3.3.2.6.5 Setting external subtitle path

The Sample Player allows you to optionally select an external subtitle source (for example, CEA 608, CEA 708, SRT, and SMI) for Closed Captions or subtitles rendering. You can set the directory for storing external subtitles. Refer to *setSubtitlePath(String filePath)* for API related information.

To specify the directory for an external subtitle file:

1. In **Tab-2**, input the directory path in the **Subtitle path:** textbox.
2. Click **Set**.

3.3.2.6.6 Displaying subtitles

The OSMP+ player enables you to turn on or turn off the subtitle display during the playback. To enable or disable the subtitle display, in **FCC-1**, click **Subtitle(Enable subtitle presentation)**.

The subtitle settings support the following attributes. Refer to Table 6 for the attribute explanations and the corresponding API information.

Note: Some of subtitle attributes need to be configured in **FCC-2** tab.

- Enable subtitle auto adjustment
- Enable default settings
- Reset subtitle settings
- Enable subtitle bounding box
    - Set font top percent
    - Set font left percent
    - Set font bottom percent
    - Set font right percent
- Enable subtitle gravity
- Enable font size
- Enable font color opacity
- Enable font color list
- Enable background color opacity
- Enable background color list
- Enable edge color opacity
- Enable edge color list
- Enable edge type list
- Enable font list
- Enable window background color opacity

- Enable window background color list
- Enable underline font
- Enable bold font
- Enable italic font

3.3.2.6.7 Setting the preferred subtitle language

Set your preferred language for the subtitle.

1. In **Tab-2**, type your preference in the **Preferred subtitle language** box.

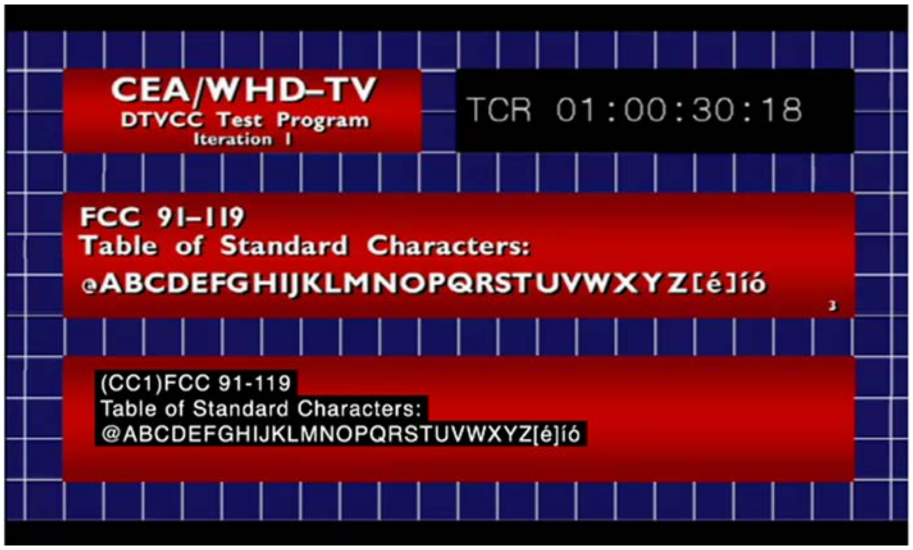
Note: Type the abbreviated language code that complies with the iso639-2 standard.

---

2. Click **Set**.

3.3.2.7 Displaying Closed Captions

The Sample Player application demonstrates the built-in SDK handling of Closed Captions and subtitles rendering. CC data appears on the video screen if it exists, as shown in Figure 68.



(CC1)FCC 91-119
Table of Standard Characters:
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[é]íó

Figure 68. Example of Closed Captions

3.3.2.8 Getting DRM library version

The Sample Player allows you to get the DRM library version number.

1. In **Tab-1**, click **Get** next to the **DRM library version** box.

3.3.2.9 Getting the Device unique ID

The Sample Player allows you to get the unique ID.

1. In **Tab-1**, click **Get** next to the **Device unique ID** box.

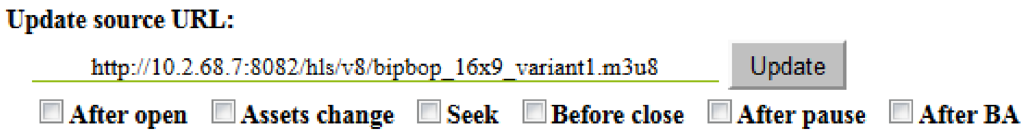
3.3.2.10 updating the media source

During playback, you can use the update source URL function to balance the workload within media servers. This feature is specific for the optimization for content delivery network (CDN) delivery.

Notes: The previous URL will be expired a few seconds after the new URL is activated. Master manifest, sub-manifest, and TS chunks of new URL must be exactly same with the previous media file.

To update the source URL address:

1. Click **Tab-2**.
2. Type the required media URL address into the **Update source URL** box, as shown in Figure 69.


Update source URL:

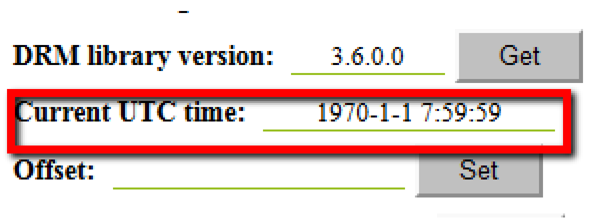
http://10.2.68.7:8082/hls/v8/bipbop_16x9_variant1.m3u8

Figure 69. Update the media source

3. Select the required option.
   - **After open**: Update the media source path after opening the playback.
   - **Assets change**: Update the media source path after changing any of the asset proprieties.
   - **Seek**: Update the media source path after performing the seek operation.
   - **Before close**: Update the media source path before closing the playback.
   - **After pause**: Update the media source path after performing the pause operation.
   - **After BA**: Update the media source path after the bitrate of the current playback is changed automatically.
4. Click **Update**.

3.3.2.11 Current UTC time

This window displays the current UTC time as defined by the *#EXT-X-PROGRAM-DATE-TIME HLS* tag, as shown in Figure 70.


DRM library version: 3.6.0.0 Get

Current UTC time: 1970-1-1 7:59:59

Offset: Set

Figure 70. UTC time

If *#EXT-X-PROGRAM-DATE-TIME* is not supported, the message *"getUTCPosition return -1"* appears or the UTC time remains frozen in 1970.

3.3.2.12 Download status & valid buffer duration

The OSMP+ player enables to display the download status and valid buffer in the **Logs** area. To display the status, click **Tab-2**, and then select **Download status & valid buffer duration**.

3.3.2.13 Offset

The offset value defines the starting playback position for the next stream playback. The OSMP+ player allows you to set the offset in milliseconds. For example, if the offset is set to 10 seconds, the player plays the next stream from the scene where the timestamp is 10 seconds.

Note: The offset feature only applies to the VoD links, other than the Live links.

To set the offset:

1. Click **Tab-1**.
2. Type the required offset value into the **Offset** box.
3. Click **Set**.

3.3.2.14 Enter NTS

The OSMP+ player provides the Enter Network Time Shifting (NTS) feature to test resuming playback from the pause point, rather than the live point.

The Enter NTS feature requires the collaboration within the streaming server, application, and OSMP+ SDK. Streaming server should create the NTS stream once the user pause live stream playback, and notify the application of the new URL of NTS stream.

The application should call OSMP+ SDK API to switch to NTS stream once the user resumes playback (after perform the pause operation for a while). Also the application should call OSMP+ SDK API to switch back to Live stream once the user catches up with the Live content (typically after a seek operation).

The requirements to use this features are:

- The streams support #EXT-X-PROGRAM-DATE-TIME HLS

- The playlist of the live stream at the time of pause and the playlist of the NTS stream at the time of resume should have the same #EXT-X-PROGRAM-DATE-TIME HLS value

In other words, there are two streams, a live stream that is paused and a NTS stream that is resumed. Both streams share the same content.

The Enter NTS feature enables playback of the NTS stream instead of the live stream when resuming playback. To use this feature, enter the value of the *PROGRAM-DATE-TIME HLS* tag (when the live stream was paused) in the **Min pos** box. **Min Pos** needs to be converted into millisecond.

After clicking **EnterNTS**, the player starts playback at the HLS Live paused position using the NTS stream. To verify that the resume operation is successful, one can check that the UTC time display should not change more than one second when playback resumes.

To enable the Enter NTS playback:

4. Click **Tab-1**.
5. In the **NTS Stream** box, type the URL address of the NTS stream that shares the same content with the live stream being played.
6. Convert the UTC time to unix timestamp, and then type this converted unix timestamp in the **Min pos** box, as shown in Figure 71. The unit of Min pos is millisecond.

   Refer to 3.3.2.11 Current UTC time for details about how to get the UTC time.

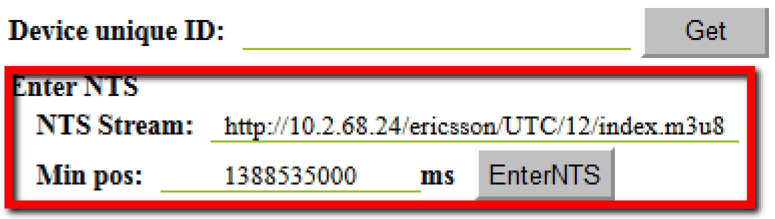

Device unique ID:
Enter NTS
NTS Stream: http://10.2.68.24/ericsson/UTC/12/index.m3u8
Min pos: 1388535000 ms

Figure 71. Enable Enter NTS

7. Click **EnterNTS**.

3.3.2.15 Enabling log

During the playback, the OSMP+ player allows to you to enable the log.

Note: This feature only applies to the debug version, other than the release version.

To enable the log:

1. Click **Tab-1**.
2. Select **Enable logs**.

3.3.2.16 Enabling time info

After the log display is enabled, the OSMP+ player allows you to display the time information in the show log area.

To enable the time info:

1. Click **Tab-1**.
2. Select **Enable logs** to show the **Time info** option.
3. Select **Time info**.

3.3.2.17 Enabling audio stream

The OSMP+ player allows you to enable or disable the audio during the playback, if the stream being played has both the audio and video streams.

To enable the playback of audio stream:

1. Click **Tab-2**.
2. Select **Enable audio stream**.

3.3.2.18 Set preference

The OSMP+ player allows user to set the preference of playback. Select **Set preferences** in **Tab-3** to enable the following options. Refer to *setPreference()* for API related information.

3.3.2.18.1 Stop keep last frame

During the transition from live channel to NTS media, the OSMP+ player keeps the last frame of the live channel being played until the first frame of NTS media is played, when the **Stop keep last frame** option is turned off. In the meantime, the OSMP+ player keeps the screen black when stopping or closing any playback.

3.3.2.18.2 Select Audio Switch Immediately

When the **Select Audio Switch Immediately** options is turned off, the OSMP+ player continues playing the previous content and then switches to the selected audio track when the previous playback is complete. This option is turned on by default.

3.3.3.19 Cookies management

You can view or clear cookies in your sample player by modifying the *basePlayer.js* file that is under *\Sample\SamplePlayer\player-js\* of your plug-ins package.

To set cookies, change the value of **headerName** in *basePlayer.js* to a string. For example:

- For http links: the value should be *`cookie name=Test; domain=visualon.com; path=/; expires=2147483647;`*

- For https links: the value should be `cookie name=Test; domain=visualon.com; path=/; expires=2147483647; secure=false;`

To clear cookies, set the value of **headerName** in *basePlayer.js* to void.

3.3.2.20 Retry times

In **Tab-1**, input a number to set the player retries times when server has no response, and then clicks **Set**.

3.3.2.21 Exiting the application

When the media playback is complete, the Sample Player application automatically stops the media pipeline. Use the **Close** icon of your Internet Explorer to exit the Sample Player.

3.3.3 Sample player usage (Mac OS)

This section describes the use of the Sample Player application integrated with the SDK. The Sample Player application implements a fully integrated media player with the following features:

- Select the media source
- Select the subtitle source
- Select the option of opening the media source, including Synchronous and Asynchronous
- Set the preferred audio and subtitle languages
- Control the playback through buttons
- Control the sound volume and mute the sound
- Support the Closed Captions and subtitle rendering
- Configure the asset properties

3.3.3.1 Initial view

The initial view of the Sample Player application provides users with an interface to select media sources, subtitle sources, and asset properties.

3.3.3.1.1 Selecting the media source

The Sample Player application allows you to select the media source, including video sources and audio sources. For information about the supported media formats, refer to 1.1 OSMP+ overview1.1 OSMP+ overview.

To select the media source:

1. In **Tab-1**, type the required URL address in the **Play URL** box.
   - Type the URL address starting with *http://* or *rtsp://* to play streaming resources on Internet
   - Type the local directory path like *C:\* to play local files

2. Click **Play** to start the playback.

Note: For details about how to initialize the SDK, open the media source, and start the playback, refer to section 4 and section 5 in the OnStream MediaPlayer+ Player SDK Integration Guide for iOS.

3.3.3.1.2 Selecting a license file

Setting a license is optional. To set the license:

1. Click **Tab-2**.
2. Type the required file path in the **License file** box.
3. Click **Set**. The default license file path is *C:\ProgramData\VisualOn\BrowserPlugin\voVidDec.dat*.

   Renewing a license file or the use of additional features may require additional license files.

4. The new license file takes effect from the next playback.

3.3.3.1.3 Setting the user agent

The Sample Player application allows you to set the user agent according to your requirements.

To set the user agent:

1. Click **Tab-2**.
2. Select **User agent settings**.
3. Type the agent name in the **Name** box as needed. This value is set to **VisualOn OSMP+ Player(MacOS)** by default.
4. Type the required agent value in the **Value** box.
5. Click **Set**. The new user agent takes effect from the next playback.

3.3.3.1.4 Setting the proxy

The Sample Player application allows you to set the proxy according to your requirements.

To set the proxy:

1. Click **Tab-2**.
2. Select **Proxy settings**.
3. Type the required host name in the **Host** box.
4. Type the required port number in the **Port** box.
5. Click **Set**. The new proxy settings take effect from the next playback.

3.3.3.1.5 HTTP gzip support

The Sample Player application allows you to enable or disable the support of the GZIP format.

To enable or disable GZIP support:

1. Click **Tab-2**.
2. Select **HTTP gip Request** or not as needed.
   The HTTP gzip support takes effect from the next playback.

3.3.3.1.6 Setting the aspect ratio

The Sample Player application allows you to set the aspect ratio as needed.

To set the aspect ratio, in **Tab-1**, select the required ratio from the **Aspect ratio** list menu.

3.3.3.1.7 Opening media source

Media source can be opened in either Sync or Async mode. Opening the media source in sync mode causes the client application to block until the open method completes or returns. In Async mode however, the API call returns immediately, and the client application is not blocked waiting for the media source to be opened. The client application can then wait for an event to indicate that the media source was opened successfully before starting the media playback.

To activate the asynchronous option, select **Async** in **Tab-1**.

3.3.3.2 Playback controls

During playback, the Sample Player application provides a set of basic controls, including a Play button, a Pause button, and a Stop button, as shown in Figure 72.

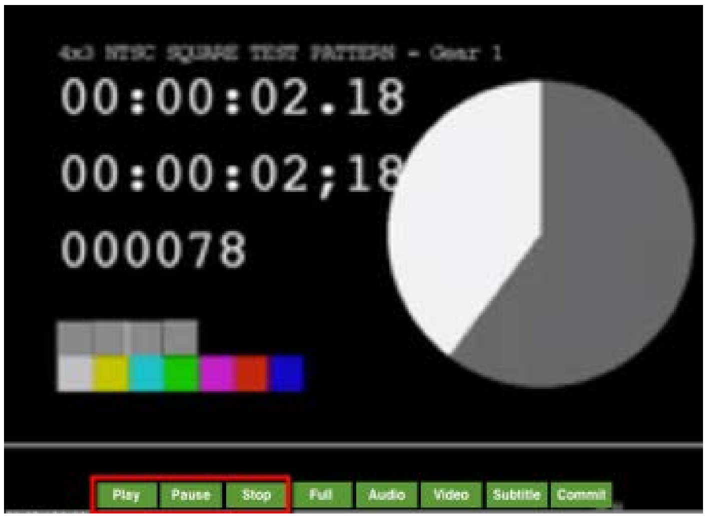


Figure 72. Playback control buttons

3.3.3.2.1 Starting the playback

After initializing the SDK and selecting the media source, click **Play** to start the playback. See 3.3.3.1.1 Selecting the media source for details.

3.3.3.2.2 Pausing the playback

During playback, click **Pause** to stop playing the media temporarily. Click **Play** to resume playback.

3.3.3.2.3 Stopping the playback

During playback, click **Stop** to stop playing the media.

3.3.3.3 Display mode controls

Full-screen mode makes videos fill the entire screen when you play them. To play the video full screen:

1. Click **Full** on the video screen, as shown in Figure 73.

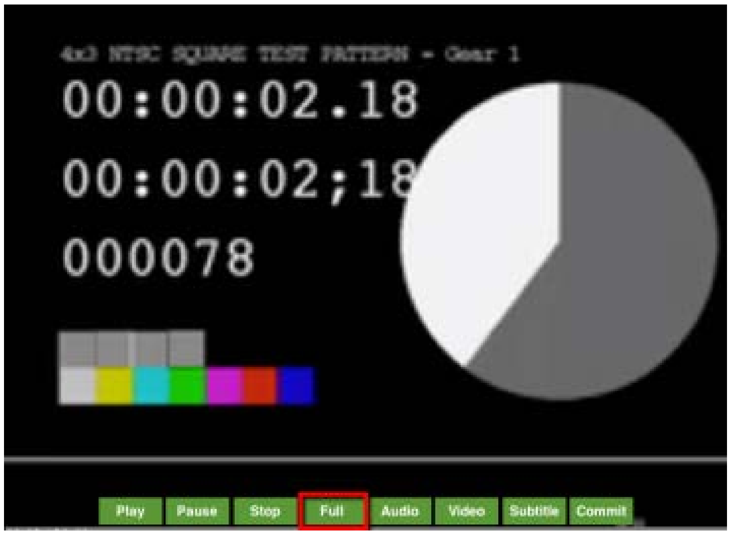


Figure 73. View full screen button

2. To quit the full-screen mode, press **Esc** on the keyboard or click **Normal**.

3.3.3.4 Sound control

You have several options for controlling the volume in the Sample Player, including adjusting the volume level and muting the sound.

To adjust the volume level, type the percent number in the **Volume** box, and then click **Set**. The value should be any number from 0 to 100.

Click **Mute** to mute the volume, as shown in Figure 74.

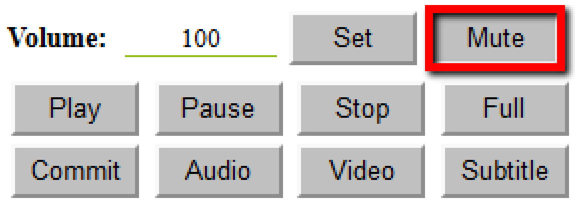


Figure 74. Sound control buttons

3.3.3.5 Asset property selections

If the media source includes multiple video tracks (also known as bitrate), audio tracks, or subtitles, the Sample Player application supports switching within various tracks during the playback.

The Sample Player provides a set of buttons to support the property switches, as shown in Figure 75.

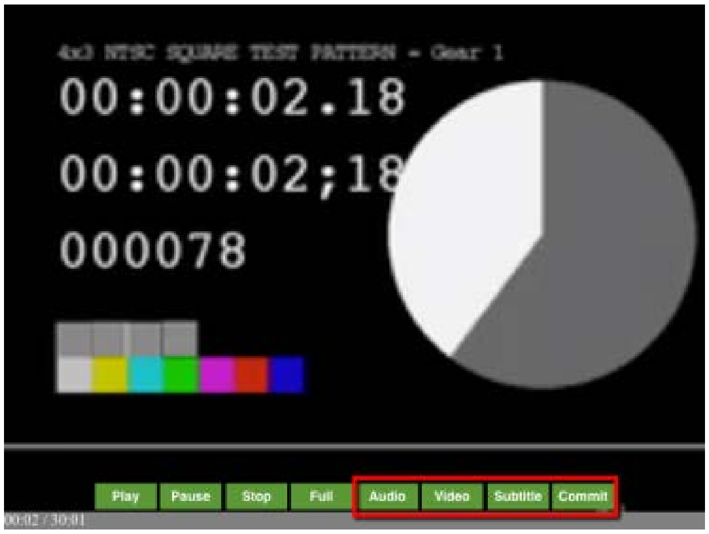


Figure 75. Property switches buttons

3.3.3.5.1 Selecting the audio

To select the audio,

1. In **Tab-1,** click **Audio**, and then select the required audio track from the audio list menu.
2. Click **Commit**.

3.3.3.5.2 Setting the preferred audio language

Set your preferred language for the audio track.

1. In **Tab-2**, type your preference in the **Preferred audio language** box.

Note: Type the abbreviated language code that complies with the iso639-2 standard.

---

2. Click **Set**.

3.3.3.5.3 Selecting the video

To select the video,

1. In **Tab-1**, click **Video**, and then select the required video track from the video list menu.
2. Click **Commit**.

3.3.3.5.4 Selecting the subtitle

To select the subtitle,

1. In **Tab-1**, click **Subtitle**, and then select the required subtitle from the subtitle list menu.
2. Click **Commit**.

3.3.3.5.5 Setting external subtitle path

The Sample Player allows you to optionally select an external subtitle source (for example, CEA 608, CEA 708, SRT, and SMI) for Closed Captions or subtitles rendering. You can set the directory for storing external subtitles. Refer to *setSubtitlePath(String filePath)* for API related information.

To specify the directory for an external subtitle file:

1. In **Tab-2**, input the directory path in the **Subtitle path:** textbox.
2. Click **Set**.

3.3.3.5.6 Displaying the subtitle

The OSMP+ player enables to you turn on or turn off the subtitle display during the playback. To enable the subtitle display, in **FCC-1**, click **Subtitle(Enable subtitle presentation)**.

The subtitle settings support the following attributes. Refer to Table 6 for the attribute explanations and the corresponding API information.

Note: Some of subtitle attributes needs to be configured in **FCC-2** tab.

- Enable subtitle auto adjustment
- Enable default settings
- Reset subtitle settings
- Enable subtitle bounding box
  - Set font top percent
  - Set font left percent
  - Set font bottom percent
  - Set font right percent
- Enable subtitle gravity

- Enable font size
- Enable font color opacity
- Enable font color list
- Enable background color opacity
- Enable background color list
- Enable edge color opacity
- Enable edge color list
- Enable edge type list
- Enable font list
- Enable window background color opacity
- Enable window background color list
- Enable underline font
- Enable bold font
- Enable italic font

3.3.3.5.6 Setting the preferred subtitle language

Set your preferred language for the subtitle.

1. In **Tab-2**, type your preference in the **Preferred subtitle language** box.

Note: Type the abbreviated language code that complies with the iso639-2 standard.

---

2. Click **Set**.

3.3.3.6 Displaying Closed Captions

The Sample Player application demonstrates the built-in SDK handling of Closed Captions and subtitles rendering. CC data appears on the video screen if it exists.

3.3.3.7 Getting DRM library version

The Sample Player allows you to get the DRM library version number.

1. In **Tab-1**, click **Get** next to the **DRM library version:** box.

3.3.3.8 Getting the Device unique ID

The Sample Player allows you to get the Device unique ID.

1. In **Tab-1**, click **Get** next to the **Device unique ID:** box.

3.3.3.9 Updating the media source

During the playback, you can select the same media source from the backup location by using the update source URL function. This feature is specific for the optimization for content delivery network (CDN) delivery.

Note: The previous URL will be expired a few seconds after the new URL is activated. Master manifest, sub-manifest, and TS chunks of new URL must be exactly same with the previous media file.

To update the source URL address:

1. Click **Tab-2**.
2. Type the required media URL address into the **Update source URL** box, as shown in Figure 76.

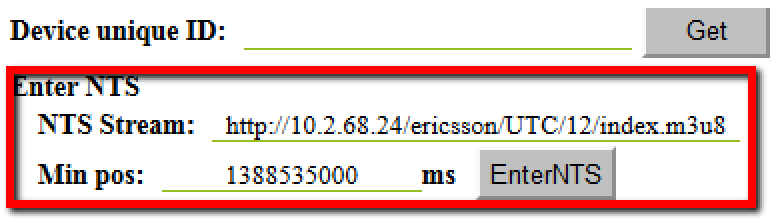

Device unique ID:
Enter NTS
NTS Stream: http://10.2.68.24/ericsson/UTC/12/index.m3u8
Min pos: 1388535000 ms

Figure 76. Update the media source

3. Select the required option.
   - **After open**: Update the media source path after opening the playback.
   - **Assets change**: Update the media source path after changing any of the asset proprieties.
   - **Seek**: Update the media source path after performing the seek operation.
   - **Before close**: Update the media source path before closing the playback.
   - **After pause**: Update the media source path after performing the pause operation.
   - **After BA**: Update the media source path after the bitrate of the current playback is changed automatically.
4. Click **Update**.

3.3.3.10 Current UTC time

This window displays the Current UTC time as defined by the *#EXT-X-PROGRAM-DATE-TIME HLS* tag, as shown in Figure 77.

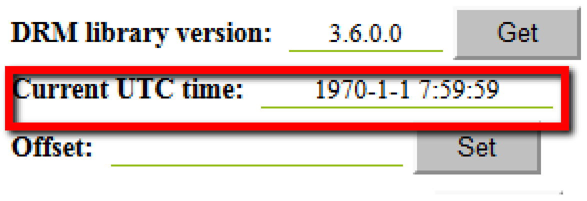

DRM library version: 3.6.0.0
Current UTC time: 1970-1-1 7:59:59
Offset:

Figure 77. UTC time

If *#EXT-X-PROGRAM-DATE-TIME* is not supported, the message *"getUTCPosition return -1"* appears or the UTC time remains frozen in 1970.

3.3.3.11 Download status & valid buffer duration

The OSMP+ player enables to display the download status and valid buffer in the **show log** area. To display the status, click **Tab-2**, and then select **Download status & valid buffer duration**.

3.3.3.12 Offset

The offset value defines the starting playback position for the next stream playback. The OSMP+ player allows you to set the offset in milliseconds. For example, if the offset is set to 10 seconds, the player plays the next stream from the scene where the timestamp is 10 seconds.

Note: The offset feature only applies to the VoD links, other than the Live links.

To set the offset:

1. Click **Tab-1**.
2. Type the required offset value into the **Offset** box.
3. Click **Set**.

3.3.3.13 Enter NTS

The OSMP+ player provides the Enter Network Time Shifting (NTS) feature to test resuming playback from the pause point, rather than the live point.

The Enter NTS feature requires the collaboration within the streaming server, application, and OSMP+ SDK. Streaming server should create the NTS stream once the user pause live stream playback, and notify the application of the new URL of NTS stream.

The application should call OSMP+ SDK API to switch to NTS stream once the user resumes playback (after perform the pause operation for a while). Also the application should call OSMP+ SDK API to switch back to Live stream once the user catches up with the Live content (typically after a seek operation).

The requirements to use this features are:

- The streams support #EXT-X-PROGRAM-DATE-TIME HLS
- The playlist of the live stream at the time of pause and the playlist of the NTS stream at the time of resume should have the same #EXT-X-PROGRAM-DATE-TIME HLS value

In other words, there are two streams, a live stream that is paused and a NTS stream that is resumed. Both streams share the same content.

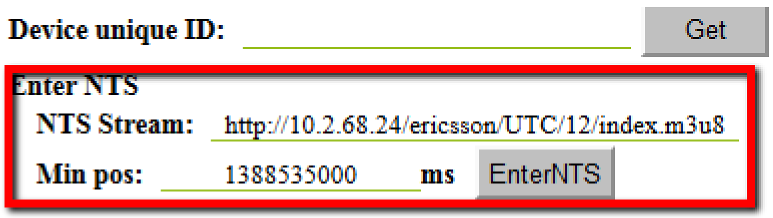
The Enter NTS feature enables playback of the NTS stream instead of the live stream when resuming playback. To use this feature, enter the value of the *PROGRAM-DATE-TIME HLS* tag (when the live stream was paused) in the **Min pos** box. **Min Pos** needs to be converted into millisecond.

After clicking **EnterNTS**, the player starts playback at the HLS Live paused position using the NTS stream. To verify that the resume operation is successful, one can check that the UTC time display should not change more than one second when playback resumes.

To enable the Enter NTS playback:

1. Click **Tab-1**.
2. In the **NTS Stream** box, type the URL address of the NTS stream that shares the same content with the live stream being played.
3. Convert the UTC time to unix timestamp, and then type this converted unix timestamp in the **Min pos** box, as shown in Figure 78. The unit of Min pos is millisecond.

   Refer to 3.3.3.10 Current UTC time for details about how to get the UTC time.



Device unique ID:
Get
Enter NTS
NTS Stream: http://10.2.68.24/ericsson/UTC/12/index.m3u8
Min pos: 1388535000 ms
EnterNTS

Figure 78. Enable Enter NTS

4. Click **EnterNTS**.

3.3.3.14 Enabling log

During the playback, the OSMP+ player allows to you to enable the log.

Note: This feature only applies to the debug version, other than the release version.

To enable the log:

1. Click **Tab-1**.
2. Select **Enable Logs**.

3.3.3.15 Enabling time info

After the log display is enabled, the OSMP+ player allows you to display the time information in the Logs area.

To enable the time info:

1. Click **Tab-1**.
2. Select **Enable Logs** to show the **Time info** option.
3. Select **Time info**.

3.3.3.16 Enabling audio stream

The OSMP+ player allows you to enable or disable the audio during the playback, if the stream being played has both the audio and video streams.

To enable the playback of audio stream:

1. Click **Tab-2**.
2. Select **Enable audio stream**.

3.3.3.17 Cookies management

You can view or clear cookies in your sample player by modifying the *basePlayer.js* file that is under *\Sample\SamplePlayer\player-js\* of your plug-ins package.

To set cookies, change the value of **headerName** in *basePlayer.js* to a string. For example:

- For http links: the value should be *`cookie name=Test; domain=visualon.com; path=/; expires=2147483647;`*
- For https links: the value should be `cookie name=Test; domain=visualon.com; path=/; expires=2147483647; secure=false;`

To clear cookies, set the value of **headerName** in *basePlayer.js* to void.

3.3.2.18 Set preference

The OSMP+ player allows user to set the preference of playback. Select **Set preferences** in **Tab-3** to enable the following options. Refer to *setPreference()* for API related information.

3.3.2.18.1 Stop keep last frame

During the transition from live channel to NTS media, the OSMP+ player keeps the last frame of the live channel being played until the first frame of NTS media is played, when the **Stop keep last frame** option is turned off. In the meantime, the OSMP+ player keeps the screen black when stopping or closing any playback.

3.3.2.18.2 Select Audio Switch Immediately

When the **Select Audio Switch Immediately** options is turned off, the OSMP+ player continues playing the previous content and then switches to the selected audio track when the previous playback is complete. This option is turned on by default.

3.3.2.19 Retry times

In **Tab-1**, input a number to set the player retries times when server has no response, and then clicks **Set**.

3.3.3.20 Exiting the application

When the media playback is complete, the Sample Player application automatically stops the media pipeline. Use the **Close** icon of your Internet Explorer to exit the Sample Player.

3.4 Windows Phone

3.4.1 Installation and uninstallation

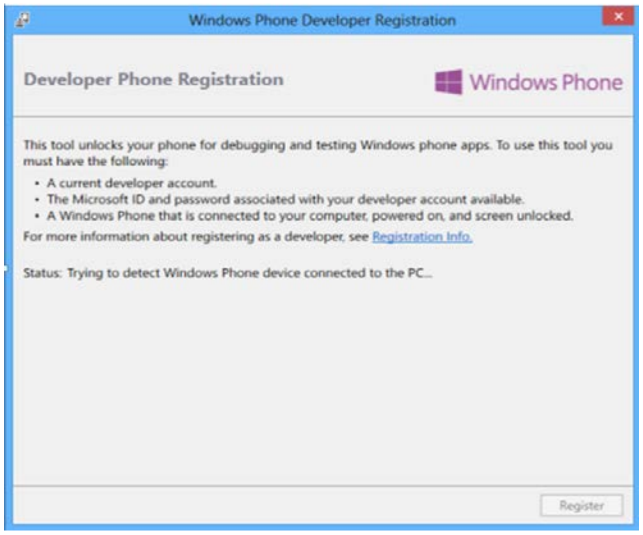
To install the Sample Player on a Windows phone, ensure that you have registered the Windows phone by using the Windows Phone Developer Registration Tool. To register your Windows phone:

1. Turn on your phone and unlock the phone screen.

   Note: Ensure that the date and time on your phone are correct.

2. Connect your phone to your computer with a USB cable that comes with your phone.
3. In **Microsoft Visual Studio 2013**, select **Tools > Windows Phone 8.1 > Developer Unlock**.
4. The **Developer Phone Registration** dialog appears, as shown in Figure 79. Ensure that the **Status** message displays **Identified Windows Phone 8 device**.

   If your phone is already registered, the **Status** message indicates this and you see a **Unregister** button.


Windows Phone Developer Registration
Developer Phone Registration
Windows Phone
Register

This tool unlocks your phone for debugging and testing Windows phone apps. To use this tool you must have the following:

- A current developer account.
- The Microsoft ID and password associated with your developer account available.
- A Windows Phone that is connected to your computer, powered on, and screen unlocked.

For more information about registering as a developer, see Registration Info.

Status: Trying to detect Windows Phone device connected to the PC...

Figure 79. Windows Phone Developer Registration

5. Click **Register**.
6. Type your Microsoft developer account and password in the **Sign In** dialog, and then click **Sign In**.

After your phone is successfully registered, the **Status** message displays **Congratulations! You have successfully unlocked your Windows Phone**.

After registering the Windows phone, install the Sample Player on the Windows phone:

1. Connect the device to the computer with a USB cable.
2. Open **Microsoft Visual Studio 2013**, select **Tools > Windows Phone 8.1 > Application Deployment**.
3. Browse to select *VisualOnASamplePlayer_release.appx*, and then click **Deploy**.
4. The installation is automatically running in the background. Then the Sample Player application can be explored under **Applications** on the device.

To uninstall the Sample Player on a Windows phone, press and hold the Sample Player application, and then select **Uninstall** from the context menu.

3.4.2 Opening media source

Media source can be opened in either Sync or Async mode. The Sample Player uses Async mode by default. For sample code of opening media source, refer to 4.4.3 Opening media source.

- Sync mode: Opening the media source in sync mode causes the client application to block until the open method completes or returns.
- Async mode: In Async mode, the API call returns immediately, and the client application is not blocked waiting for the media source to be opened. The client application can then wait for an event to indicate that the media source was opened successfully before starting the media playback.

3.4.3 Selecting/Playing media source

The initial view allows you to select the media source for playback. You can type the media source path manually or select from a list of media source listed in a text file.

To manually type a media file or link:

1. In the initial view, type the full URL starting with *http://* or file path to the media source in the Input streaming URL address or file path text box.
2. Tap **Play**.

Alternately, a list of media files and links can be supplied to the Sample Player through a text file. The text list must be saved to */sdcard/url.txt*, with each link URL and file path separated by line. To supply the media text file into a Windows phone:

1. Download and install IsoStoreSpy.
2. Open **IsoStoreSpy**, and then click **WP Application**.
3. In the **Choose an application** dialog, select **Device** from the **<select a device>** list box.
4. In the **Files** section, click the upload button ⤒.
5. Browse to select the *url.txt* file, and then click **OK**. You can add the playback url addresses in the *url.txt* file if needed.

To select a media file or link from *url.txt* in the Sample Player:

1. In the initial view, tap **Select** to display the media file list.
2. Select the required media source.
3. Tap **Play**.

3.4.4 Playback options

3.4.4.1 Enabling/disabling deblock

Deblocking is a feature applied to decoded compressed video to improve visual quality and prediction performance by smoothing the sharp edges which can form between macroblocks when block coding techniques are used.

Enabling deblock helps improve the playback quality, especially when the video has mosaic. Tap **Set to enable deblock function** from **Options** to enable the deblock feature. Refer to *enableDeblock()* for API related information.

3.4.4.2 Setting buffering time

The OSMP+ player allows you to set the buffering time of the source to be played.

3.4.4.2.1 Max Buffering Time

Refer to *setMaxBufferingTime()* for API related information.

To set the maximum time of source buffer:

1. Tap **Max Buffering Time** from **Options**.
2. Type the number to set the maximum buffering time in milliseconds.
3. Tap **OK**.

3.4.4.2.2 Initial Buffering Time

Turn on **Initial Buffering Time** option to set the buffering time of the source in milliseconds before the playback starts. Refer to *setInitialBufferingTime()* for API related information.

To enable the initial buffering time of source buffer:

1. Tap **Initial Buffering Time** from **Options**.
2. Type the number to set the initial buffering time in milliseconds.
3. Tap **OK**.

3.4.4.2.3 Playback Buffering Time

When re-buffering is needed during playback, the player enables you to set the buffering time with the **Playback Buffering Time** option. Refer to *setPlaybackBuferingTime()* for API related information.

To set the re-buffering time:

1. Tap **Playback Buffering Time** from **Options**.
2. Type the number to set the time in milliseconds.
3. Tap **OK**.

3.4.4.3 Enabling CPU Adaptation

The Bitrate Adaptation of Sample Player subjects to setting that is hardcoded in the player if CPU Adaptation is enabled.

To enable CPU Adaptation, tap **Enable CPU adaptation** from **Options**. Refer to *enableCPUAdaptation()* for API related information.

3.4.4.4 Subtitle settings

Tap **Options**, and then tap **Set to enable subtitle settings** to enable or disable subtitle settings.

Table 8 lists the available subtitle attributes that can be set in the OSMP+ player and their related API s.

Table 8. Subtitle attributes and related APIs

Attribute	**Definition**	**API**
Enable Subtitle	Enable/disable subtitles display. Disabled is set by default.	*enableSubtitle()*

Attribute	Definition	API
Enable Default Settings	Use the subtitle settings from source stream	N/A
Reset Subtitle Settings	Reset all parameters to the default values that are specified in source stream.	*resetSubtitleParameter()*
Enable Font size	Set subtitle font size scale.	*setSubtitleFontSizeScale()*
Enable Font Color Opacity	Set subtitle font color opacity rate.	*setSubtitleFontOpacity()*
Enable Font Color List	Set subtitle font color.	*setSubtitleFontColor()*
Enable Background Color Opacity	Set subtitle font background color opacity rate.	*setSubtitleFontBackgroundOpacity()*
Enable Background Color List	Set subtitle font background color.	*setSubtitleFontBackgroundColor()*
Enable Edge Color Opacity	Set subtitle font edge color opacity rate.	*setSubtitleFontEdgeOpacity()*
Enable Edge Color List	Set subtitle font edge color.	*setSubtitleFontEdgeColor()*
Enable Edge Type List	Set subtitle font edge type.	*setSubtitleFontEdgeType()*
Enable Font List	Set subtitle font name.	*setSubtitleFontName()*
Enable Window Background Color Opacity	Set window background color opacity rate.	*setSubtitleWindowBackgroundOpacity()*
Enable Window Background Color List	Set window background color.	*setSubtitleWindowBackgroundColor()*
Enable Underline Font	Enable or disable the underline font for subtitle.	*setSubtitleFontUnderline()*
Enable Bold Font	Enable or disable the bold font for subtitle.	*setSubtitleFontBold()*
Enable Italic Font	Enable or disable the italic font for subtitle.	*setSubtitleFontItalic()*

3.4.4.5 Setting bitrate range

Tap **Options**, and then tap **Set bitrate range (kbps)** to enable or disable the bitrate range settings. Refer to *setBitrateThreshold* for API related information. Refer to *setBitrateThreshold()* for API related information.

Tap **Lower Bound Bitrate Range/Upper Bound Bitrate Range**, type the number for lower and upper bound bitrate range, and then tap **OK**.

3.4.4.6 Setting presentation delay time

The OSMP+ player provides the **Presentation delay time** option for HLS and Smooth Streaming, to allow user downloading chunks with some seconds of delay that is defined by this option. Refer to *setPresentationDelay()* for API related information.

3.4.4.7 HTTP settings

3.4.4.7.1 Enabling HTTP Retry Timeout

Tap **Options**, and then tap **Enable set HTTP connection retry timeout.** to enable the retry timeout function. Refer to *setHTTPRetryTimeout()* for API related information.

Tap **HTTP Retry Timeout**, and then type a number to set the player retry times when server has no response.

3.4.4.7.2 Enabling HTTP Gzip Request

HTTP data is compressed before it is sent from the server: compliant browsers will announce what methods are supported to the server before downloading the correct format; browsers that do not support compliant compression method will download uncompressed data. Gzip is one of the most common compression schemes. Refer to *enableHTTPGzipRequest()* for API related information.

Tap **Options**, and then tap **Enable or disable sending HTTP request with gzip** to enable or disable the support of the GZIP format.

3.4.4.7.3 Setting HTTP proxy

To set the HTTP proxy server:

1. Tap **Options**, and then tap **Set the HTTP proxy.**
2. Top **HTTP Proxy Host**, type the host sever, and then tap **OK**.
3. Tap **HTTP Proxy Port**, type the host sever, and then tap **OK**.

3.4.4.8 Setting default audio language

Tap **Options**, and then tap **Default Audio** to enable or disable the Default Audio settings. Refer to *setPreferredAudioLanguage* for API related information.

3.4.5 Switching Asset properties

If the media source includes multiple video tracks (also known as bitrate), audio tracks, or subtitles, the Sample Player application supports switching properties during the playback.

To switch properties:

1. Tap **Asset** on the bottom of the video screen to display the **Asset** menu.
2. Tap the required video, audio, and/or subtitle track.
3. Tap **Commit** to confirm your selection. Playback continues with your selection.

3.4.6 Capturing log

Follow the instructions to obtain your log file on Windows Phone 8.1:

Note: The logging system is only enabled in the debug version. The difference between debug and release versions, is that the debug version can output the log information in different levels with various detailed information.

1. Open **IsoStoreSpy**.
2. Click the upload button , and then browse to select *volog.cfg*.
3. Logs are generated and save in the *volog.log* file located under the same directory with *volog.cfg*.

Chapter 4 OSMP+ Integration

This chapter presents the following topics:

4.1 Integration on Android

Follow the instructions below to complete the OSMP+ integration on the Android platform.

The integration Labs, which are compressed into the *labs.zip* file under **Doc** of your installation package, provide working examples to accompany the *OSMP+ Player SDK Integration Guide for Android Platforms*. The integration Labs focus on individual integration topics and require installation of the SDK and license file. Start with *README.txt* compressed in the *labs.zip* file to learn how to use Labs.

4.1.1 Preparing SDK files

Decompress the Android package, and you see the following directory structure under **Android** of your installation directory. Refer to 2.1 Android package for details.

- Bin
- Doc
- Jar
- Libs
- Libs_x86
- License
- SamplePlayer

Follow these steps to complete the prerequisites for importing the SDK libraries:

1. Create the **libs** directory under the **SamplePlayer** directory, if the **SamplePlayer** directory does not include the **libs** directory already.
2. Copy all .jar files (Java SDK libraries) to **libs**.

   If you want to use the debug version to import logs, copy all .jar files under **Jar\debug** to **libs** and copy all .so files under **Libs\debug** to the folder created in step 3.
3. Create a directory named **armeabi** under **libs**.

   Or

   If you use x86, create a directory named **x86** under **libs**.
4. Copy all .so files (C/C++ shared libraries) to **armeabi**.

   Or

   Copy all .so files (C/C++ shared libraries) to **x86**.
5. Create a directory named **assets** under **SamplePlayer**, if it does not exist already, and then copy the license file, **voVidDec.dat** to **assets**.

To use the OSMP+ SDK, add the following sample codes to import Java SDK libraries.

```
import com.visualon.OSMPPlayer.*;
import com.visualon.OSMPPlayerImpl.*;
```

Table 9 describes the Java libraries used in OSMP+.

Table 9. Java libraries

Package	Commonly used Interfaces and classes
com.visualon.OSMPPlayer	• VOCommonPlayer • VOCommonPlayerAssetSelection • VOCommonPlayerConfiguration • VOCommonPlayerControl • VOCommonPlayerDeviceInfo • VOCommonPlayerHDMI • VOCommonPlayerHTTPConfiguration • VOCommonPlayerListener • VOCommonPlayerRTSPConfiguration • VOCommonPlayerStatus • VOCommonPlayerSubtitle • VOOSMPChunkInfo • VOOSMPInitParam • VOOSMPType • VOOSMPVerificationInfo
com.visualon.OSMPPlayerImpl	• VOCommonPlayerHDMIImpl • VOCommonPlayerImpl • VOOSMPAssetIndexImpl • VOOSMPAssetPropertyImpl • VOOSMPEnumUtils • VOOSMPInitParamImpl

4.1.2 Initializing SDK instance

The SDK instance initialization includes the following:

- Create an instance
- Configure a view
- Define the view size
- Set the license
- Set the event callback

The following is the sample code for initializing an SDK instance.

```
m_sdkPlayer = new VOCommonPlayerImpl();
// Retrieve location of libraries
String apkPath = getUserPath(this) + "/lib/";
String cfgPath = getUserPath(this) + "/";
// SDK player engine type
VO_OSMP_PLAYER_ENGINE eEngineType =
VO_OSMP_PLAYER_ENGINE.VO_OSMP_VOME2_PLAYER;
// Initialize SDK player
 m_sdkPlayer.init(eEngineType, init);
// Set view
m_sdkPlayer.setView(m_svMain);
// Set surface view size
DisplayMetrics dm = new DisplayMetrics();
getWindowManager().getDefaultDisplay().getMetrics(dm);
m_sdkPlayer.setViewSize(dm.widthPixels, dm.heightPixels);
// Register SDK event listener
m_sdkPlayer.setOnEventListener(m_listenerEvent);
// Set device capability file location
String capFile = cfgPath + "cap.xml";
m_sdkPlayer.setDeviceCapabilityByFile(capFile);
// Set license content
InputStream is = null;
byte[] b = new byte[32*1024];
try {
     is = m_context.getAssets().open("voVidDec.dat");
     is.read(b);
     is.close();
     } catch (IOException e) {
        e.printStackTrace();
     }
m_sdkPlayer.setLicenseContent(b);
```

4.1.3 Opening media source

Use *VOCommonPlayer.open()* to open the media source. See the following for sample codes.

```
// Auto-detect source format
VO_OSMP_SRC_FORMAT format =
VO_OSMP_SRC_FORMAT.VO_OSMP_SRC_AUTO_DETECT;
// Set source flag to asynchronous
VO_OSMP_SRC_FLAG eSourceFlag;
eSourceFlag = VO_OSMP_SRC_FLAG.VO_OSMP_FLAG_SRC_OPEN_ASYNC;
VOOSMPOpenParam openParam = new VOOSMPOpenParam();
openParam.setDecoderType(VO_OSMP_DECODER_TYPE.VP_OSMP_DEC_VIDEO_SW
.getValue() |
VO_OSMP_DECODER_TYPE.VO_OSMP_DEC_AUDIO_SW.getValue());
//Open media source
 m_sdkPlayer.open(m_strVideoPath, eSourceFlag, format, openParam);
```

4.1.4 Handling SDK events

Use *VOCommonPlayerListener()* to handle SDK events. See the following for sample codes.

```
private VOCommonPlayerListener m_ListenerEvent = new
VOCommonPlayerListener(){
// SDK Event Handling
...
```

```
public VO_OSMP_RETURN_CODE onVOEvent(VO_OSMP_CB_EVENT_ID nID, int
nParam1, int nParam2, Object obj) {
switch(nID) {
...
Case VO_OSMP_CB_OPEN_FINISHED: {
if (nParam1 == VO_OSMP_RETURN_CODE.VO_OSMP_ERR_NONE.getValue())
{ // MediaPlayer is opened
VO_OSMP_RETURN_CODE nRet;
// Start (play) media pipeline
mRet = m_sdkPlayer.start();
...
} else {
onError(m_sdkPlayer, nParam1, 0);
}
break;
}
...
}
return VO_OSMP_RETURN_CODE.VO_OSMP_ERR_NONE; }
```

4.1.5 Starting playback

The OSMP+ player can start the playback either in the Sync mode or in the Async mode. See the following for the source code of opening the source in Sync mode or in Async mode.

4.1.5.1 Sync

```
eSourceFlag = VO_OSMP_SRC_FLAG.VO_OSMP_FLAG_SRC_OPEN_SYNC;
m_sdkPlayer.open(...)
m_sdkPlayer.start();
```

4.1.5.2 Async

```
eSourceFlag = VO_OSMP_SRC_FLAG.VO_OSMP_FLAG_SRC_OPEN_ASYNC;

m_sdkPlayer.open(...);
```

The player uses *VOCommonPlayer.start()* to start playback after opening the media source. See the following for sample codes.

```
Case VO_OSMP_CB_OPEN_FINISHED: {
if (nParam1 == VO_OSMP_RETURN_CODE.VO_OSMP_ERR_NONE.getValue())
{ // MediaPlayer is opened
VO_OSMP_RETURN_CODE nRet;
// Start (play) media pipeline
mRet = m_sdkPlayer.start();
…
```

4.1.6 Stopping playback

Use *VOCommonPlayer.stop()* to stop playback. See the following for the sample code for the whole stop function.

```
m_sdkPlayer.stop();
m_sdkPlayer.close();
m_sdkPlayer.destroy();
m_sdkPlayer = null;
```

4.1.7 Seeking

Use *VOCommonPlayer.setPosition()* to perform the seeking operation. For example, `m_sdkPlayer.setPosition(60000)`, in this sample, the unit of time is millisecond, and the player seeks the position which time is 00:01:00.

See the following for the sample code of the seeking operation.

```
public void onStopTrackingTouch(SeekBar arg0)
{
// Calculate new position as percentage of total duration
int iCurrent = arg0.getProgress();
int iMax = arg0.getMax();
long m_nDuration = m_sdkPlayer.getDuration();
long lNewPosition;
if (m_nDuration > 0) {
lNewPosition = iCurrent * m_nDuration / iMax;
}
else {
long nMinPos = m_sdkPlayer.getMinPosition();
long nMaxPos = m_sdkPlayer.getMaxPosition();
int nDuration = (int) (nMaxPos - nMinPos);
lNewPosition = iCurrent * nDuration / iMax;
lNewPosition = lNewPosition + nMinPos;
}
if (m_sdkPlayer != null)
{
Log.v(TAG,"Seek To " + lNewPosition);
m_sdkPlayer.setPosition(lNewPosition); // Set new position
}
}
```

4.1.8 Selecting tracks

If the clip includes multiple tracks, use the *VOCommonPlayer.selectVideo(), VOCommonPlayer.selectAudio(),* or *VOCommonPlayer.selectSubtitle()*function to select the required track.

Note: Call the *VOCommonPlayer.commit()* function after calling the selection functions.

To select a track:

1. Get the number of video tracks by using *int count = m_sdkPlayer.getVideoCount()*;
2. Use *m_sdkPlayer.selectVideo(index)* and *m_sdkPlayer.commit()* to select the corresponding index of track for playback.
   - If the first track is selected, the index value is 0. Increase the index value in the ascending order. For example, if the second track is selected, the index value is 1.
   - Set the index value to -1, which means to adjust the video track automatically according to the network bandwidth.

Note: For audio and subtitle selection, the index value cannot be set to -1. Refer to *VOCommonPlayerAssetSelection* for the all related functions.

4.1.9 Subtitle rendering (Optional)

Use *VOCommonPlayer.enableSubtitle()* to enable the subtitle rendering, and use *VOCommonPlayer.setSubtitlePath()* to indicate an external file, for example:

```
m_sdkPlayer.enableSubtitle(true);
m_sdkPlayer.setSubtitlePath(strSubtitlePath);
```

Refer to the *VOCommonPlayerSubtitle* interface for the other subtitle related functions.

4.1.10 Download Manager (Optional)

OSMP+ supports to download streaming to local disks. Refer to the *VOOSMPStreamingDownloader* interface for details.

4.1.10.1 Initializing Download Manager

See the following for the sample code of initializing Download Manager.

```
m_downloader = new VOOSMPStreamingDownloaderImpl();
String apkPath = CommonFunc.getApkPath(m_context);
VOOSMPStreamingDownloaderInitParam init = new
VOOSMPStreamingDownloaderInitParam();
init.setContext(m_context);
init.setLibraryPath(apkPath);
m_downloader.init(this, init);
```

4.1.10.2 Download Manager (open media source)

See the following for the sample code for opening the media source in Download Manager.

```
m_cDownloader.open( strVideoPath, 0, strLocalDownloadPath);
```

In the above sample, the *strVideoPath* is the link address that is to be downloaded and *strLocalDownloadPath* is the path that is added into the local files.

4.1.10.3 Download Manager event

See the following for the sample code of Downloader Manager event.

```
public VO_OSMP_RETURN_CODE onVOStreamingDownloaderEvent(
            VO_OSMP_CB_STREAMING_DOWNLOADER_EVENT_ID event, int
arg1, int arg2,Object obj) {

switch(event){
        case
VO_OSMP_CB_STREAMING_DOWNLOADER_SYNC_AUTHENTICATION_DRM_SERVER_INF
O:
// DRM download support
        {
            VOOSMPVerificationInfo info = new
VOOSMPVerificationInfo();
            info.setDataFlag(1);
            if(m_verificationString != null)
                info.setVerificationString(m_verificationString);
            m_downloader.setDRMVerificationInfo(info);
```

```
          break;
        }
case VO_OSMP_CB_STREAMING_DOWNLOADER_OPEN_COMPLETE:
//Select video/audio/subtitle track to download;
m_downloader.selectVideo(index);
m_downloader.commitSelection();
…
m_downloader.start();
break;
case VO_OSMP_CB_STREAMING_DOWNLOADER_MANIFEST_OK:
//This is the download local path,it can be play throuth OSMP SDK
now
String localURL = String(obj);
m_sdkPlayer.open(localURL……);
         break;
…
}
}
```

4.1.10.4 Stopping Download

Use the following functions to stop the download.

```
m_cDownloader.stop();
m_cDownloader.close();
```

4.1.11 Hardware acceleration (Optional)

Use *VOCommonPlayer.open()* to specify the hardware decoder type on Android. Change the value of **param openParam** in *api open()* to use the hardware decoder.

See the following for the sample code for specifying the hardware decoder type.

```
VOOSMPOpenParam openParam = new VOOSMPOpenParam();
//use SW
openParam.setDecoderType(VO_OSMP_DECODER_TYPE.VP_OSMP_DEC_VIDEO_SW
.getValue() |
VO_OSMP_DECODER_TYPE.VO_OSMP_DEC_AUDIO_SW.getValue());
//use IOMX
openParam.setDecoderType(VO_OSMP_DECODER_TYPE.VP_OSMP_DEC_VIDEO_IO
MX.getValue() |
VO_OSMP_DECODER_TYPE.VO_OSMP_DEC_AUDIO_IOMX.getValue());
//use MediaCodec
openParam.setDecoderType(VO_OSMP_DECODER_TYPE.VP_OSMP_DEC_VIDEO_ME
DIACODEC.getValue() |
VO_OSMP_DECODER_TYPE.VO_OSMP_DEC_AUDIO_MEDIACODEC.getValue());
//use AutoHW
openParam.setDecoderType(VO_OSMP_DECODER_TYPE.VP_OSMP_DEC_VIDEO_
HARDWARE_AUTO_SELECTED.getValue() |
VO_OSMP_DECODER_TYPE.VO_OSMP_DEC_AUDIO_SW.getValue());
```

4.1.12 Playback speed (Optional)

Use *VOCommonPlayer.setAudioPlaybackSpeed()* to set the speed of audio playback. For example:

```
m_sdkPlayer.setAudioPlaybackSpeed(2.0);
```

The value is set to 1.0 by default.

4.1.13 Zoom

OSMP+ provides *VOCommonPlayer.setZoomMode()* to set the zoom mode, which includes **PanScan**, **FitWindow**, **Original**, **LetterBox**, and **ZoomIn**. Use *voSurfaceView* instead of *surfaceView* to support the hardware zoom feature in the application level. Refer to 1.4.11 Zoom for details about the zoom feature.

See the following for the sample code for setting the zoom mode.

- PanScan:
  m_sdkPlayer.setZoomMode(VO_OSMP_ZOOM_MODE.VO_OSMP_ZOOM_PANSCAN, null);
- FitWindow:
  m_sdkPlayer.setZoomMode(VO_OSMP_ZOOM_MODE.VO_OSMP_ZOOM_FITWINDOW, null);
- Original:
  m_sdkPlayer.setZoomMode(VO_OSMP_ZOOM_MODE.VO_OSMP_ZOOM_ORIGINAL, null);
- LetterBox:
  m_sdkPlayer.setZoomMode(VO_OSMP_ZOOM_MODE.VO_OSMP_ZOOM_LETTERBOX,null);
- ZoomIn:

  *Rect rect = new Rect(videoWidth / 4, videoHeight / 4, videoWidth * 3 / 4, videoHeight * 3 / 4);*

  m_sdkPlayer.setZoomMode(VO_OSMP_ZOOM_MODE.VO_OSMP_ZOOM_ZOOMIN, rect);

4.1.14 RTSP settings (Optional)

Use *VOCommonPlayer.RTSPConfiguration()*to change RTSP settings.

4.1.14.1 RTSPOverHTTP

```
// Enable RTSP over HTTP tunneling.
m_sdkPlayer. enableRTSPOverHTTP(true);
// Set port number for RTSP over HTTP tunneling.
m_sdkPlayer. setRTSPOverHTTPConnectionPort(num);
```

4.1.14.2 RTSP HTTP verification information

```
VOOSMPVerificationInfo verif = new VOOSMPVerificationInfo();
verif.setVerificationString(str);
verif.setDataFlag(1);
// Set the verification information to start HTTP verification
m_sdkPlayer .setHTTPVerificationInfo(verif);
```

4.1.14.3 RTSP statistics

```
// Get the RTSP module status value
VOOSMPRTSPStatistics RTSPStatistics =
m_cSpecialPlayer.getRTSPStatistics();
```

```
String log ="RTSP Statistics - PacketReceived: %d,
PacketDuplicated: %d, PacketLost: %d, PacketSent: %d,
AverageJitter: %d, AverageLatency: %d.";
log = String.format(log, RTSPStatistics.getPacketReceived(),
RTSPStatistics.getPacketDuplicated(),
RTSPStatistics.getPacketLost(),
RTSPStatistics.getPacketSent(),RTSPStatistics.getAverageJitter(),
RTSPStatistics.getAverageLatency());
```

4.1.14.4 Other RTSP interfaces

```
// Enable/Disable immediate video rendering with low latency.
enableLowLatencyVideo(boolean value)
//Set RTSP connection type,it can be auto/TCP/UDP
setRTSPConnectionType(VO_OSMP_RTSP_CONNECTION_TYPE type);
```

4.1.15 Enabling DRM

Use *VOCommonPlayer.setDRMLibrary()* to enable the DRM engine.

```
m_sdkPlayer.setDRMLibrary(“voDRM” “voGetDRMAPI”);
```

4.2 Integration on iOS

Follow the instructions below to complete the OSMP+ integration on the iOS platform.

The integration Labs, which are compressed into the *labs.zip* file under **Doc** of your installation package, provide working examples to accompany the *OSMP+ Player SDK Integration Guide for iOS Platforms*. The integration Labs focus on individual integration topics and require installation of the SDK and license file. Start with *README.txt* compressed in the *labs.zip* file to learn how to use Labs.

4.2.1 Preparing SDK files

Import header files, libraries, and license file into the player project.

4.2.1.1 Importing header files

Import the OSMP+ SDK header files that are relevant for your player project. A basic SDK client requires the *VOCommonPlayerImpl.h* header file. Table 10 lists the header files dependency.

Table 10. Header files dependency

Header file	**Header files included in header file**	**Definition**
VOCommonPlayerImpl.h	• VOCommonPlayer.h • VOCommonPlayerAssetSelection.h • VOCommonPlayerConfiguration.h • VOCommonPlayerControl.h • VOCommonPlayerDelegate.h • VOCommonPlayerDeviceInfo.h (optional)	Common Player implementation, including configuration, control, subtitles, asset selection, and device information

Header file	Header files included in header file	Definition
	• VOCommonPlayerHDMI.h (optional) • VOCommonPlayerHTTPConfiguration.h (optional) • VOCommonPlayerImpl.h • VOCommonPlayerRTSPConfiguration.h (optional) • VOOSMPRTSPPort.h (optional) • VOOSMPRTSPStatistics.h (optional) • VOCommonPlayerStatus.h • VOCommonPlayerSubtitle.h (optional) • VOOSMPHTTPDownloadFailure.h (optional) • VOOSMPHTTPProxy.h (optional) • VOOSMPImageData.h (optional) • VOOSMPInitParam.h • VOOSMPOpenParam.h • VOOSMPType.h • VOOSMPVerificationInfo.h (optional)	
VOCommonPlayerHDMIImpl.h	VOCommonPlayerHDMI.h	HDMI connection
VOOSMPChunkInfo.h	N/A	Chunk information
VOOSMPBuffer.h	N/A	
VOOSMPPCMBuffer.h	N/A	
VOOSMPStreamingDownloaderImpl.h	VOOSMPStreamingDownloader.h VOOSMPStreamingDownloaderType.h	Download Manager

To import the *VOCommonPlayer implementation* header file, add the following lines:

```
#import <VOCommonPlayerImpl.h>
```

4.2.1.2 Importing libraries

voLoadLibControl is used to import the libraries and get the module API.

See the following for sample code of importing libraries:

```
void* voGetModuleAdapterFunc(char *pszApiName)
{if (0 == strcmp(pszApiName, "voGetAudioReadAPI")) {
```

```
                              return (void
*)voGetAudioReadAPIAdapter;
}
if (0 == strcmp(pszApiName, "voGetAudio2ReadAPI")) {
                              return (void
*)voGetAudio2ReadAPIAdapter;
                        }
}
```

4.2.1.3 Importing license file

Copy the **voVidDec.dat** file to the player project file.

4.2.1.4 Importing framework files

Ensure that the following .framework files and lib files are included in your sample player project for iOS, to guarantee the successful compilation environment.

- AudioToolbox.framework
- AVFoundation.framework
- CFNetwork.framework
- CoreAudio.framework
- CoreGraphics.framework
- CoreLocation.framework
- CoreMedia.framework
- CoreMotion.framework
- CoreText.framework
- CoreVideo.framework
- ExternalAccessory.framework
- Foundation.framework
- libresolv.dylib
- MediaPlayer.framework
- OpenGLES.framework
- QuartzCore.framework
- Security.framework
- SystemConfiguration.framework
- UIKit.framework
- VideoToolbox.framework

4.2.2 Initializing SDK instance

The SDK instance initialization includes the following:

- Creating an instance
- Configuring a view

- Defining the view size
- Setting the license
- Setting the event callback

See the following for the sample code for initializing an SDK instance.

4.2.2.1 Initializing a player instance

```
playEngineType  [in] Refer to {@link VO_OSMP_PLAYER_ENGINE}.
initParam       [in] Currently unused, should be set to nil.
VO_OSMP_PLAYER_ENGINE playEngineType = VO_OSMP_VOME2_PLAYER;
VOOSMPInitParam *parma = nil;
self.player = [[[VOCommonPlayerImpl alloc] init:
VO_OSMP_VOME2_PLAYER initParam:parma] autorelease];
```

4.2.2.2 Setting surface view for video playback

```
view  [in] The UIView(iOS)
[self.player setView:view];
```

4.2.2.3 Notifying the player on view size change

```
[self.player notifyViewSizeChanged];
```

4.2.2.4 Setting license file path

```
path  [in] Location of the license file.
[self.player setLicenseFilePath:path];
[self.player setPreAgreedLicense:agreedLicenseString];
```

4.2.2.5 Setting event callback

```
delegate [in] Event handler instance.
[self.player setOnEventDelegate:self];
```

4.2.3 Opening media source

Use *VOCommonPlayer.open()* to open the media source.

```
VOOSMPOpenParam * param = nil;
int sourceType = VO_OSMP_SRC_AUTO_DETECT;
[self.player open:url flag:VO_OSMP_FLAG_SRC_OPEN_ASYNC
sourceType:sourceType openParam:param];
```

4.2.4 Handling SDK event

See the following for the sample code for handling the basic event.

```
(VO_OSMP_RETURN_CODE) onVOEvent:(VO_OSMP_CB_EVENT_ID)nID
param1:(int)param1 param2:(int)param2 pObj:(void *)pObj
{
   if (nID == VO_OSMP_CB_PLAY_COMPLETE) {
        }
    else if (nID == VO_OSMP_SRC_CB_OPEN_FINISHED)
    {
}
return VO_OSMP_ERR_NONE;
}
```

4.2.5 Starting playback

Use *VOCommonPlayer.start()* to start playback after opening the media source.

```
[self.player start];
```

4.2.6 Stopping playback

Use *VOCommonPlayer.stop()* to stop playback.

```
[self.player stop];
[self.player close];
```

4.2.7 Seeking

Use *VOCommonPlayer.setPosition()* to perform the seeking operation.

```
[self.player setPosition:msec];
```

4.2.8 Selecting tracks

If the clip includes multiple tracks, use the *VOCommonPlayer.selectVideo()*, *VOCommonPlayer.selectAudio()*, or *VOCommonPlayer.selectSubtitle()* function to select the required track.

4.2.8.1 Getting asset count

```
[self.player getVideoCount];
[self.player getAudioCount];
[self.player getSubtitleCount];
```

4.2.8.2 Getting asset property

```
id<VOOSMPAssetProperty> videoItem = [self.player
getVideoProperty:index];
id<VOOSMPAssetProperty> audioItem = [self.player
getAudioProperty:index];
id<VOOSMPAssetProperty> subtitleItem = [self.player
getSubtitleProperty:index];
```

4.2.8.3 Checking availability of the specified track

```
BOOL *videoAvailable = [self.player isVideoAvailable:index];
BOOL *audioAvailable = [self.player isAudioAvailable:index];
BOOL *subtitleAvailable = [self.player isSubtitleAvailable:index];
```

4.2.8.4 Getting the selected track

```
id<VOOSMPAssetIndex> index = [self.player getCurrentSelection];
```

4.2.8.5 Getting the actual index of track being played

```
id<VOOSMPAssetIndex> index = [self.player getPlayingAsset];
```

4.2.8.6 Selecting track by index

The value of index is valid from 0.

```
[self.player selectVideo:index];
[self.player selectAudio:index];
[self.player selectSubtitle:index];
```

4.2.8.7 Committing all asset selections

```
[self.player commitSelection];
```

4.2.8.8 Removing uncommitted selections

```
[self.player clearSelection];
```

4.2.9 Subtitle rendering (Optional)

Use *VOCommonPlayer.enableSubtitle()* to enable the subtitle rendering, and use *VOCommonPlayer.setSubtitlePath()* to indicate an external file.

4.2.9.1 Setting file/URL path for external subtitles

```
[self.player setSubtitlePath:filePath];
```

4.2.9.2 Enabling/disabling subtitle display

This value is set to disabled by default.

```
[self.player enableSubtitle:value];
```

4.2.9.3 Setting subtitle font color

```
[self.player setSubtitleFontColor:color]; // color [in] Font color
(0x00RRGGBB) of subtitle text.
```

4.2.9.4 Setting subtitle font size

```
[self.player setSubtitleFontSizeScale:scale]; // scale [in] Font
size scale for subtitle text (percent).
```

4.2.9.5 Setting subtitle font background color

```
[self.player setSubtitleFontBackgroundColor:color]; // color [in]
Subtitle font background color (0x00RRGGBB).
```

4.2.9.6 Setting opacity of subtitle font background color

```
[self.player setSubtitleFontBackgroundOpacity:alpha]; // alpha
[in] Subtitle font background color opacity rate.
```

4.2.9.7 Setting window background color

```
[self.player setSubtitleWindowBackgroundColor:color]; // color
[in] Subtitle window background color (0x00RRGGBB).
```

4.2.9.8 Setting opacity of window background color

```
[self.player setSubtitleWindowBackgroundOpacity:alpha]; // alpha
[in] Subtitle window background color opacity rate.
```

4.2.9.9 Enabling/disabling subtitle font italics

```
[self.player setSubtitleFontItalic:enable]; // enable [in]
Enable/Disable
```

4.2.9.10 Enabling/disabling subtitle font bold

```
[self.player setSubtitleFontBold:enable]; // enable [in]
Enable/Disable
```

4.2.9.11 Enabling/disabling subtitle font underlined

```
[self.player setSubtitleFontUnderline:enable]; // enable [in]
Enable/Disable
```

4.2.9.12 Setting subtitle font name

```
[self.player setSubtitleFontName:name]; // name [in] Font name for
subtitle text
```

4.2.9.13 Setting subtitle font edge type

```
[self.player setSubtitleFontEdgeType:type]; // type [in] Edge type
of subtitle font.
```

4.2.9.14 Setting subtitle font edge color

```
[self.player setSubtitleFontEdgeColor:color]; // color [in] Font
edge color (0x00RRGGBB) of subtitle text.
```

4.2.9.15 Setting opacity rate of subtitle font edge color

```
[self.player setSubtitleFontEdgeOpacity:alpha]; // alpha [in] Edge
color opacity rate of subtitle font.
```

4.2.9.16 Previewing subtitle by sending some sample text to be rendered

```
[self.player previewSubtitle:@"Sample" view:view]; // sampleText
[in] text for previewing current subtitle settings; view [in] view
to which the sample text is being rendered.
```

Use *[self.player resetSubtitleParameter];* to reset all parameters to their default values. Subtitles are presented as specified in the subtitle stream.

4.2.10 Download Manager (Optional)

OSMP+ allows you to download streams to local disks.

4.2.10.1 Initializing a Download Manager instance

See the following for the sample code for initializing a Download Manager instance.

```
@param   self [in] Refer to {@link
VOOSMPStreamingDownloaderDelegate}
@param   initParam [in] Refer to {@link
VOOSMPStreamingDownloaderInitParam}
VOOSMPStreamingDownloaderInitParam *initParam = NULL;
id<VOOSMPStreamingDownloader> downloader =
[[VOOSMPStreamingDownloaderImpl alloc] init:self
initParam:initParam];
```

4.2.10.2 Download Manager (Open media source)

See the following for the sample code for opening downloader media source.

```
NSString *filePath =
[[NSSearchPathForDirectoriesInDomains(NSDocumentDirectory,
NSUserDomainMask, YES) objectAtIndex:0]
stringByAppendingPathComponent:@"Downloader"];
```

```
VO_OSMP_RETURN_CODE  ret = [self.downloader open:url flag:flag
localDir:filePath];
```

4.2.10.3 Creating the player and playback

After receiving the *APP_DOWNLOADER_DOWNLOADER_MANIFEST_OK* event, set *pObj* for the URL of playback. See the following for the sample code for reference.

```
-(VO_OSMP_RETURN_CODE)
handleDownloaderEvent:(APP_DOWNLOADER_EVENT_ID)nID
param1:(int)param1 param2:(int)param2 pObj:(void *)pObj
{
[self performSelectorOnMainThread:@selector(startPlayDownloader:)
withObject:(NSString *)pObj waitUntilDone:NO];
}
- (void)startPlayDownloader:(NSString *)pURL
{
    // Create player and open with pURL.
 }
```

4.2.10.4 Selecting a downloader track and starting playback

After receiving the *APP_DOWNLOADER_OPEN_COMPLETE* event, select a track, and then start playback. See the following for the sample code for reference.

```
- (VO_OSMP_RETURN_CODE)
handleDownloaderEvent:(APP_DOWNLOADER_EVENT_ID)nID
param1:(int)param1 param2:(int)param2 pObj:(void *)pObj
{
   if (nID == APP_DOWNLOADER_OPEN_COMPLETE)
{
[self performSelectorOnMainThread:@selector(startDownloader)
withObject:nil waitUntilDone:NO];
}
}

- (void)startDownloader
{
    // Select downloader track and start playback.
}
```

4.2.10.5 Starting download

See the following for the sample code for starting download.

```
[self.downloader startDownloader];
```

4.2.10.6 Stopping download

See the following for the sample code for stopping download.

```
[self.downloader stop];
```

4.2.10.7 Closing download

See the following for the sample code for closing download.

```
[self.downloader close];
```

4.2.11 Hardware acceleration (Optional)

Use *VOCommonPlayer.init()* to set the engine type. VO_OSMP_AV_PLAYER is used for iOS.

4.2.11.1 Initializing the player with VisualOn media framework engine

See the following for the sample code for initializing the player with VisualOn media framework engine.

```
VOOSMPInitParam *parma = nil;
id<VOCommonPlayer> player = [[VOCommonPlayerImpl alloc] init:
VO_OSMP_VOME2_PLAYER initParam:parma];
```

4.2.11.2 Initializing the player with AVPlayer engine

This is for iOS platform only. AVPlayer supports H.264 video decoder and AAC audio decoder. See the following for the sample code for initializing the player with AVPlayer engine.

```
VOOSMPInitParam *parma = nil;
id<VOCommonPlayer> player = [[VOCommonPlayerImpl alloc] init:
VO_OSMP_AV_PLAYER initParam:parma];
```

4.2.12 Playback speed (Optional)

Use *VOCommonPlayer.setAudioPlaybackSpeed()* to set the audio playback speed. See the following for the sample code for setting the playback speed.

```
speed  [in] speed multiplier with respect to realtime playback.
Default is 1.0
[self.player setAudioPlaybackSpeed:speed];
```

4.2.13 Zoom

Use *VOCommonPlayer.setZoomMode()* to set the zoom mode. See the following for the sample code of the zoom mode.

```
mode [in] Zoom mode. Refer to {@link VO_OSMP_ZOOM_MODE}.

rect [in] The rectangular area of the video to be displayed. This
argument is only used when the zoom mode is {@link
VO_OSMP_ZOOM_ZOOMIN}.

VO_OSMP_ZOOM_MODE mode = VO_OSMP_ZOOM_LETTERBOX;

Rect rt;
memset(&rt, 0, sizeof(Rect));
[self.player setZoomMode:mode rect:rt];
```

4.2.14 RTSP settings (Optional)

Use the *VOCommonPlayerConfiguration* interface to change RTSP settings.

4.2.14.1 Setting RTSP Connection Type

See the following for the sample code for setting the RTSP Connection Type.

```
Type [in] Connection type. {@link VO_OSMP_RTSP_CONNECTION_TYPE}
[self.player setRTSPConnectionType:type];
```

4.2.14.2 Setting port number for RTSP connection

See the following for the sample code for setting port number for RTSP connection.

```
port [in] port number
VOOSMPRTSPPort *port = [[VOOSMPRTSPPort alloc] init:6666
videoConnectionPort:8888];
VO_OSMP_RETURN_CODE ret = [self.player
setRTSPConnectionPort:port];
[port release];
port = nil;
```

4.2.14.3 Setting port number for RTSP over HTTP tunneling

See the following for the sample code for setting port number for RTSP over HTTP tunneling.

```
portNum   [in] port number
[self.player setRTSPOverHTTPConnectionPort:portNum];
```

4.2.14.4 Enabling RTSP over HTTP tunneling

See the following for the sample code for enabling RTSP over HTTP tunneling.

```
enable  [in] Enable/Disable;
[self.player enableRTSPOverHTTP:enable];
```

4.2.14.5 Setting maximum socket errors

See the following for the sample code for setting the maximum socket errors.

```
[self.player setRTSPMaxSocketErrorCount:count];
```

4.2.14.6 Setting RTSP connection timeout

See the following for the sample code for setting RTSP connection timeout.

```
time [in] RTSP connection timeout (in seconds), default is 10
seconds
[self.player setRTSPConnectionTimeout:time];
```

4.2.14.7 Getting the RTSP module statistics

See the following for the sample code for getting the RTSP module statistics.

```
id<VOOSMPRTSPStatistics> info = [self.player getRTSPStatistics];
```

4.2.15 Enabling DRM

The OSMP+ SDK uses the AES-128 DRM by default. See the following for the sample code for enabling the AES-128 DRM.

```
[self.player setDRMLibrary:@"voDRM_VisualOn_AES128" libApiName:@"v
oGetDRMAPI”];
```

4.3 Integration on Windows and Mac OS

Follow the instructions below to construct a VisualOn OSMP+ plug-in instance on Windows or Mac OS. For the detailed API information, refer to the *API Reference Manual.zip* file in your installation package.

4.3.1 Creating a plug-in object

To create a VisualOn OSMP+ plug-in object:

1. Use the following code to create an object element.

   ```
   var pluginObj = document.createElement("object");
   ```

2. Specify the newly created object as a VisualOn OSMP+ plug-in object. The method for specifying this object varies from the browser and platform.

   - For Windows IE, use classid to specify the plug-in object:

     ```
     pluginObj.setAttribute("classid","clsid:********-****-****-
     ****-************");
     ```

     Fill the * characters with the register key of VisualOn OSMP+ plug-in object.

   - For other browsers on Windows, use the object type to specify the plug-in object:

     ```
     pluginObj.setAttribute("type", "application/x-visualon-
     osmp");
     ```

   - For Safari on Mac OS, the video is plotted directly on the Safari window.

     ```
     pluginObj.setAttribute("wmode", "transparent");
     ```

4.3.2 Initializing OSMP+

To create an OSMP+ instance, and then bind this instance to the plug-in object created in 4.3.1 Creating a plug-in object:

1. Use the following code to create an OSMP+ instance, and then bind it to the plug-in object:

   ```
   voplayer = new VOCommonPlayer(OSMPPlayerPlugin);
   voplayer.init(voOSMPType.VO_OSMP_PLAYER_ENGINE.VO_OSMP_VOME2
   _PLAYER, '');
   ```

2. Use the following code to register a callback that is used to handle the player event:

   ```
   var callbackFunction = foo (nID, nParam1, nParam2, object)
   {…;}*
   voplayer.registerEventHandler(callbackFunction);
   ```

For more event handling information, refer to the *OnStream MediaPlayer+ Plugin SDK Integration Guide for Windows. pdf* or the *OnStream MediaPlayer+ Plugin SDK Integration Guide for Mac OS.pdf*.

4.3.3 Enabling DRM

The OSMP+ SDK uses the AES-128 DRM by default. Refer to **DRM information (optional)** in *Table 4: Examples of Registry location* the following documents to find the information to change the DRM server.

- *OnStream MediaPlayer+ Plugin SDK Integration Guide for Windows. pdf*
- *OnStream MediaPlayer+ Plugin SDK Integration Guide for Mac OS.pdf*.

4.4 Integration on Windows Phone

Follow the instructions below to complete the OSMP+ integration on the Windows Phone 8.1 platform.

The following is the required software for interfacing with the OSMP+ SDK on Windows Phone 8.1:

- 64-bit Windows 8 Pro or higher
- Windows Phone 8.x SDK
- Microsoft Visual Studio Ultimate 2013 or higher
- Register a Windows Phone 8.x device by signing up for a developer account
- Enable Hyper-V for Windows Phone 8.x emulator
- IsoStoreSpy for file exploration

Note: Refer to Register your Windows Phone for instructions about how to register a Windows Phone device.

4.4.1 Preparing SDK files

Import header files, libraries, and license file into the player project.

Import the OSMP+ SDK header files that are relevant for your player project. A basic SDK client requires the *VOCommonplayer.h* header file. Table 11 lists the header files dependency.

Table 11. Header files dependency

Header file	Header files included in header file	Definition
VOCommonPlayer.h	• VOCommonPlayer.h • VOCommonPlayerAnalytics.h • VOCommonPlayerAssetSelection.h	Common Player implementation, including configuration, control, subtitles, asset

Header file	Header files included in header file	Definition
	• VOCommonPlayerConfiguration.h • VOCommonPlayerControl.h • VOCommonPlayerDeviceInfo.h • VOCommonPlayerHTTPConfiguration.h • VOCommonPlayerListener.h • VOCommonPlayerRTSPConfiguration.h • VOCommonPlayerStatus.h • VOCommonPlayerSubtitle.h	selection, and device information

To use VisualOn OSMP+ SDK on Windows Phone, include *VOCommonplayer.h* in VS2013 project, and then set a proper additional library path to include *voOSMP.lib* in the link phase. Also, import license file in VS2013 project.

4.4.2 Initializing SDK instance

The SDK instance initialization includes the following:

- Creating an instance
- Configuring a view
- Setting the event callback

4.4.2.1 Initializing a player instance

See the following for the sample code for initializing a player instance.

```
m_commonPlayer = std::unique_ptr<VOCommonPlayer>(new
VOCommonPlayer());
VO_OSMP_PLAYER_ENGINE playEngineType = VO_OSMP_VOME2_PLAYER;
VOOSMPInitParam initParam;
char stra[2048];
{
   memset(stra, 0, sizeof(stra));
   WideCharToMultiByte(CP_ACP, 0, (TCHAR*)szPath, -1, stra, 1024,
NULL, NULL);
   VOOSMPString pth(stra);
   initParam.setLibraryPath(pth);
}

int nRC = m_commonPlayer->init(playEngineType, initParam);
```

4.4.2.2 Setting where the video is plotted by setView

On Windows Phone 8.1, OSMP+ SDK takes two arguments, including the current window and a SwapChainPanel instance. For more information about these two classes, see documents and examples of Windows Phone DirectX with XAML project.

```
Windows::UI::Core::CoreWindow^ window =
CoreWindow::GetForCurrentThread();

void* viewParam[2] = {
        reinterpret_cast<void *>(window),

        reinterpret_cast<void *>(swapChainPanel)

};

m_commonPlayer->setView(reinterpret_cast<void *>(viewParam));
```

4.4.2.3 Setting event listener to handle OSMP+ events with callback

The following sample code demonstrates how to register a listener `VOEventListenerWP` that is derived from `VOCommonPlayerListener`. In the definition of `VOEventListenerWP`, user customizes the event-handling logic.

```
m_commonPlayerListener =
std::unique_ptr<VOCommonPlayerListener>((VOCommonPlayerListener*)(
new VOEventListenerWP(this)));
if (m_commonPlayer != nullptr) {
        m_commonPlayer>setOnEventListener(*m_commonPlayerListener);
    }
}
```

4.4.3 Opening media source

Use *VOCommonPlayer.openSource()* to open the media source. The following sample code demonstrates how to register a callback for external source in push PD mode.

```
VOOSMPSetPushBufferPrototype source = m_simulateDataSource-
>GetCallback();
if (source != NULL)
{
      ret = m_commonPlayer->openSource(source,
VO_OSMP_FLAG_SRC_PUSH_BUFFER_FUNCTION);
      if (ret != VO_OSMP_ERR_NONE)
      {
            auto msgDlg = ref new MessageDialog("Open Source
Failed!", "Error");
            msgDlg->ShowAsync();
            return ret;
      }
}
```

4.4.4 Handling SDK event

We[4] implement a simple listener wrapper to demonstrate how to bind callbacks to handle OSMP+ events. The wrapper is implemented in the files *VOEventListenerWP.h* and *VOEventListenerWP.cpp*.

With this wrapper, user can put the event handling logic in DirectXPage and achieve the interaction between XAML UI and OSMP+ SDK.
For more information about DirectXPage, refer to DirectX and XAML project template of VS2013. For more information about event handling

[4] In this guide, “we” refer to the VisualOn development engineering team that develops the OSMP+ SDK.

logic, refer to *DirectXPage::initVOEventListener()* function in the push-mode OSMP+ sample.

4.4.5 Supported APIs and push buffer usage

OSMP+ SDK for Windows Phone provides the following subset of APIs to retrieve the streaming data with push-mode. For more information about the following APIs, refer to *API Reference Manual.zip* under your installation package.

- init
- destroy
- setView
- openSource
- close
- start
- pause
- stop
- setPosition
- setOnEventListener
- getPlayerStatus
- getPosition
- getMinPosition
- getMaxPosition
- getDuration
- getParameter
- setLicenseContent
- setLicenseFilePath

In addition to the APIs listed above, source with push-mode is transferred by registering the following callbacks:

- VOOSMPGetInfoPrototype
- VOOSMPPushBufferPrototype
- VOOSMPSetPushBufferPrototype

For the definitions of the prototypes above, refer to *VOOSMPPushBuffer.h*. For the example of setting callbacks to push-mode source, refer to 4.4.3 Opening media source and the sample player.

4.5 Best practices

VisualOn considered the following key best practices in the OSMP+ integration to ensure the optimal performance:

- When using **Async** mode to open the media source, call the *start()* API after receiving a completed open event. The **Sync** mode, by contrast, calls the *start()* API directly after calling the open event, because the open event blocks the current thread until the open event completes.
- Call two APIs, *suspend()* and *resume(),* on the main thread.
- Keep the video view active during the playback.
- Keep the *VOCommonPlayerListener* object or *VOCommonPlayerDelegate* object on iOS active during the playback.
- Ensure that no function is called when deleting the player object.
- Call *setLicenseContent()*, *setLicenseFilePath(),* and *setPreAgreedLicense()* after completing the *init()* API.
- Call the following control functions in one thread:
  - open()
  - start()
  - pause()
  - setPosition()
  - stop()
  - close()
- Do not support cloning the player object, which may result in the incorrect player status.
- We recommend NOT changing the voSurfaceView or its Activity during playback. If you have to change the voSurfaceView or its Activity during playback, we suggest performing the following:
  a. Call suspend() at first.
  b. Change the voSurfaceView or its Activity. Ensure that the change operation is completed before proceeding with step c.
  c. Call resume() with the new voSurfaceView.

  Call setDRMLibrary() to enable DRM engine. If it is required to set the unique identifier or verification information, ensure that you call setDRMUniqueIdentifer()/setDRMVerificationInfo() after calling setDRMLibrary(). Meanwhile, ensure that you call setDRMverificationInfo() after calling setDRMUniqueIndentifer() if you need to set both functions.

Chapter 5 Logging System

This chapter presents the following topics:

5.1 Overview

An effective logging strategy has long been recognized by system administrators as a way to keep track of problems with components and applications, provide quantifiable statistics for an application's history, help in troubleshooting issues, as well as help in monitoring the overall health of the systems that administrators are responsible for.

This section discusses how you, as a developer or support engineer, can use the logging mechanism provided to make it easier to understand the OSMP+ SDK, provide more in-depth information for debugging, and communicate with field support engineers when a problem is encountered or a threshold is reached.

5.2 Log File Location

Follow the instructions to obtain your log file on separate platforms:

Note: The logging system is only enabled in the debug version. The difference between debug and release versions, is that the debug version can output the log information in different levels with various detailed information. The release version is obfuscated on Android to improve the security.

- **Android**: Logs are generated and saved in the directory by specifying *$dumpfilePath$* value in *volog.cfg*.
- **iOS**: Logs are generated and save at */application/Documents*
- **Windows**: Logs are generated and saved at *C:\Users\<user name>\AppData\LocalLow*
- **Mac OS**: Logs are generated and saved at *~/Library/Internet/Plug-ins/voBroserPlugin.plugin*
- **Windows Phone**: Logs are generated and saved in your app’s local folder.

5.3 Log File Configurations on Android

In addition to Android platform, the other platforms, including iOS, Windows, Mac OS, and Windows Phone, dump the log file to the predefined local directory that is not allowed being changed. For the information about the predefined log directory, refer to 5.2 Log File Location.

To dump a log into the local storage on Android:

3. Open *volog.cfg* under the *sdcard* directory with a text editor.
4. Specify the following values in *volog.cfg*:

VisualOn

- *Local log enabling flag*: 0xFFFFFFFF
- *Path to save log file*: /AA/BB/CC
- *Max. log file size in Mbyte*

Take *0xFFFFFFFF,/sdcard/VR,128* in *volog.cfg* for example:

- **0xFFFFFFFF**: Used to enable this local logging capability
- **/sdcard/VR**: Log file '*volog.log'* will be saved under */sdcard/VR*.
- **128**: Maximum file size is 128MByte. Player will overwrite the file if the message size captured is greater than 128 Mbytes.

In general, the log file is flushed to the local storage from cache after the workflow completes. However the player might crash during playback and the log will be lost as log information is temporarily saved in the cache and is not flushed to the local storage. If you want to capture this type of log, perform the following in *volog.cfg*:

1. Open *volog.cfg* in the debug version with text editor.
2. Specify the following values in *volog.cfg*.

    - *Local log enabling flag*: 0xFFFFFFFE
    - (only available for Android) *Path to save log file*: /AA/BB/CC
    - *Max. log file size in Mbyte*
    - *Flag for enabling this approach (0/1): 0* is to enable this special logging approach*, and 1* is to disable this approach. 0 is set by default.

5.4 Log File Categories

To better understand the logging mechanism and locate the required information with ease, the OSMP+ SDK categorizes the logging definitions as follows:

- Log messages
    - Bitrate Adaptation logs
    - Buffer logs
    - Content logs
    - Controller logs
    - DRM logs
    - Hardware codec logs
    - License logs
    - Local file logs
    - KPI logs

    - OS source logs
    - Output control logs
    - RTSP logs
    - Streaming logs
    - SourceIO logs
    - Subtitle logs
    - VOME2 engine logs
- Error messages

See 5.5 Logging Messages for the detailed information about the log categories and log definitions. For the detailed information about error messages, refer to Appendix A: Error codes.

5.5 Logging Messages

Before explaining the logging keywords, take the following for example to understand the variables' meaning in your log.

```
$time$
@@@VOLOG,$LevelName$,$ModuleID$,$ThreadID$,$FileName$,$FunctionNam
e$,$LineNumber$,$YOURSLOG$
```

- *$time$:* The current time, for example "12:23:56.789"
- *$LevelName$:* can be either of Error, Warning, Info, Status, Run, or Function
- *$ModuleID$:* The value of module's compiling tag (*_VOMODULEID)*
- *$ThreadID$***:** The thread id that calls the outputted log
- *$FileName$:* The file name that calls the outputted log
- *$FunctionName$:* The function name that calls the outputted log
- *$LineNumber$:* The line's number that calls the outputted log
- *$YOURSLOG$:* The log message that you want to output

See the following for an example of log entry.

```
10:43:00.721 @@@VOLOG,Info,09020000,40047FFC,COSSourceBase.cpp,
Open,825,openned over!
```

The following sections describe the logging keywords that are defined in the *voLog.h* header file. The sample output message through Table 12 to Table 27 is *$YOURSLOG$* of the above sample.

Note: Contents in *Sample output message* column demonstrate an example of keyword's output message. Characters/Strings/Numbers included in angle brackets <> vary from your actual log message.

5.5.1 Bitrate Adaptation logs

Table 12 lists the keywords, keywords' definitions, parameter types, and sample output messages for the Bitrate Adaptation related logs labeled with LOGLEVEL=0.

Table 12. Bitrate Adaptation related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_BA_CAP(x)	Maximum bitrate that can be used	int	BA CAP bandwidth: <200000 b>
VOLOG_KEYWORD_BA_INIT_BITRATE(x)	Initial bitrate that should be selected	int	BA initial bitrate: <200000 bps>
VOLOG_KEYWORD_BA_CPU_LOADING(x)	CPU loading value when performing bitrate adaptation	int	BA CPU-loading: <57%>
VOLOG_KEYWORD_BA_SWITCHING(x,y)	Stream bitrate is changed	Int, int	BA stream bitrate changes from bitrate: <200000 bps> to bitrate: <737777 bps>

5.5.2 Buffer logs

Table 13 lists the keywords, keywords' definitions, parameter types, and sample output messages for the buffer related logs labeled with LOGLEVEL=0. All keywords are the information about player's buffer pool or the buffering operation.

Table 13. Buffer related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_BUFFER_MAXLEN(x)	Maximum duration of buffer pool	int	Buffer maximum length: <20000 ms>
VOLOG_KEYWORD_BUFFER_BA_POS(x,y)	Position where bitrate adaptation occurs	int	Buffer BA position: <10032 ms>
VOLOG_KEYWORD_BUFFER_BUFFERING	Indicates that buffering is progress	N/A	Buffer buffering in progress

5.5.3 Content logs

Table 14 lists the keywords, keywords' definitions, parameter types, and sample output messages for the content related logs labeled with LOGLEVEL=0.

Table 14. Content related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_CONTENT_URL(x)	Information of content URL string	char*	Content URL: <http://10.2.68.105/nts/apple/gear4/fileSequence1.ts>
VOLOG_KEYWORD_CONTENT_VIDEO_CODEC(x)	Video codec type used for the content being played	char*	Content video codec: <AVPlayer>
VOLOG_KEYWORD_CONTENT_AUDIO_CODEC(x)	Audio codec type used for the content being played	char*	Content audio codec: <AAC>

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_CONTENT_RESOLUTION(x,y)	Video resolution in width <x> height <y>	int, int	Content resolution: <1024 x 768>
VOLOG_KEYWORD_CONTENT_STREAMING_MEDIA_TYPE(x)	Streaming type that is either Live or VOD	char*	Content streaming type: <Live>

5.5.4 Controller logs

Table 15 lists the keywords, keywords' definitions, parameter types, and sample output messages for the controller related logs labeled with LOGLEVEL=0.

Table 15. Controller related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_CONTROLLER_DOWNLOAD_CHUNK(a,b,c,d,e,f)	Information of chunk being downloaded	char*, int, long, int, int, char*	Controller downloads a chunk [result<processing OK>, size<942444 B>, downloadtime<443 ms>, chunk duration<17019302 b>, download speed(123022 bps),url<http://10.2.68.105/nts/apple/gear4/fileSequence1.ts>]
VOLOG_KEYWORD_CONTROLLER_STREAMING_INFO(x,y,z)	Streaming information (id, bitrate, selected(Y/N) from streaming parser.	int, int,int	Controller streaming info [id<0>, bitrate<200000 bps>, selected<N>]

5.5.5 DRM logs

Table 16 lists the keywords, keywords' definitions, parameter types, and sample output messages for the DRM related logs labeled with LOGLEVEL=0.

Table 16. DRM related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_DRM_START	DRM engine initialization starts	N/A	DRM engine initialization starts
VOLOG_KEYWORD_DRM_STOP	DRM engine stops	N/A	DRM engine stops
VOLOG_KEYWORD_DRM_LICENSE_SERVER(x)	Information of DRM license server URL	char*	DRM license server URL: <http://10.2.68.105 >
VOLOG_KEYWORD_DRM_SOURCE_URL(x)	Information of DRM source URL	char*	DRM source URL: <http://10.2.68.105/nts/apple/gear1/fileSequence0.ts>
VOLOG_KEYWORD_DRM_DESTINATION_URL(x)	Information of DRM destination URL that is converted from the source URL	char*	DRM destination URL: <http://10.2.68.105/nts/apple/gear1/fileSequence0.ts>

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_DRM_DRM_TYPE(x)	Information of DRM vendor, for example, playready.	char*	DRM type: <AES128_CommonAPI>
VOLOG_KEYWORD_DRM_KEY_STRING(x)	Information of the DRM key string	char*	DRM key string: <>
VOLOG_KEYWORD_DRM_KEY_VRAPI(x)	Verimatrix API name	char*	DRM Verimatrix API: < VR VerifyHandshake- 0>

5.5.6 Hardware Codec logs

Table 17 lists the keywords, keywords' definitions, parameter types, and sample output messages for the hardware codec related logs labeled with LOGLEVEL=0.

Table 17. Hardware codec related logs

Keyword	Definition	Parameter type	Sample Output message
VOLOG_KEYWORD_MEDIACODEC_OPEN	Indicates that source is opened using MediaCodec	N/A	Hardware codec MediaCodec is created
VOLOG_KEYWORD_MEDIACODEC_FINISHED	Indicates that source opened with MediaCodec is completed	N/A	Hardware codec MediaCodec is destroyed
VOLOG_KEYWORD_MEDIACODEC_VIDEO_HEAD_DATA(x)	Shows the video head data of MediaCodec	char*	Hardware codec MediaCodec video head data: <0x00>
VOLOG_KEYWORD_IOMX_START	MediaCodec starts	N/A	Hardware codec IOMX is created
VOLOG_KEYWORD_IOMX_STOP	MediaCodec completes	N/A	Hardware codec IOMX is destroyed
VOLOG_KEYWORD_IOMX_DEVICE_NAME(x)	Shows the device name	char*	Hardware codec IOMX device name: <Galaxy S3>
VOLOG_KEYWORD_IOMX_COMPONENT(x)	Shows the IOMX component information	char*	Hardware codec IOMX component: <OMX.Nvidia,h264,decoder>
VOLOG_KEYWORD_IOMX_VIDEO_RESOLUTION(x,y)	Shows the video resolution	int, int	Hardware codec IOMX video resolution: <1024 x 768>
VOLOG_KEYWORD_IOMX_SURFACE_POINTER(x)	Shows the surface pointer information	int	Hardware codec IOMX surface pointer: <0x12345>

5.5.7 License logs

Table 18 lists the keywords, keywords' definitions, parameter types, and sample output messages for the license related logs labeled with LOGLEVEL=0.

Table 18. License related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_LICENSE_START	Indicates that license check starts	N/A	License check starts
VOLOG_KEYWORD_LICENSE_STOP	Indicates that license check completes	N/A	License check completes
VOLOG_KEYWORD_LICENSE_CHECK_RESULT(x)	Result of module being performed the license check	int	License check result: <successful>

5.5.8 Local File logs

Table 19 lists the keywords, keywords' definitions, parameter types, and sample output messages for the local file related logs labeled with LOGLEVEL=0.

Table 19. Local file related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_LOCAL_FILE_FORMAT(x)	Shows the file format information	char*	Local file format: <TS>
VOLOG_KEYWORD_LOCAL_CLIP_DURATION(x)	Shows the duration of local clip	int	Local file clip duration: <87670123 ms>

5.5.9 KPI logs

Table 20 lists the keywords, keywords' definitions, parameter types, and output messages for the KPI related logs labeled with LOGLEVEL=0.

Table 20. KPI related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_KPI_FPS(x)	Information of video rendering performance in the unit of FPS (frame per second)	int	KPI Video rendering performance: <247 FPS>
VOLOG_KEYWORD_KPI_FRAME_DROP(x)	Quantity of video frames being dropped after playback	int	KPI <241> Video frame(s) dropped

5.5.10 OS Source logs

Table 21 lists the keywords, keywords' definitions, parameter types, and sample output messages for the OS source related logs labeled with LOGLEVEL=0.

Note: If ID number is -1 in output message, which means stream/audio/subtitle is the track selected by default when opening the player.

Table 21. OS source related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_OSSOURCE _SELECT_STREAM(x,y)	Indicates that a stream is selected	int, int	OS source switches stream from ID <-1> to ID <1>
VOLOG_KEYWORD_OSSOURCE _SELECT_VIDEO(x,y)	Indicates video IDs selected by OS source	int, int	OS source switches video from ID <-1> to ID <1>
VOLOG_KEYWORD_OSSOURCE _SELECT_AUDIO(x,y)	Indicates audio IDs selected by OS source	int, int	OS source switches audio from ID <-1> to ID <1>
VOLOG_KEYWORD_OSSOURCE _SELECT_SUBTITLE(x,y)	Indicates subtitle IDs selected by OS source	int, int	OS source switches subtitle from ID <-1> to ID <1>
VOLOG_KEYWORD_OSSOURCE _PROGRAMINFO_RESET	Indicates that program information is reset.	N/A	OS source program info gets reset
VOLOG_KEYWORD_OSSOURCE _PROGRAMINFO_CHANGED	Indicates that program information is changed.	N/A	OS source program info gets changed

5.5.11 Output Control logs

Table 22 lists the keywords, keywords' definitions, parameter types, and sample output messages for the OS source related logs labeled with LOGLEVEL=0.

Table 22. Output control related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_OUTPUT_ CONTROL_BEST_EFFORT(x)	Best effort setting is enabled or not (0/1)	int	Output control best effort setting enabled: <1>
VOLOG_KEYWORD_OUTPUT_ CONTROL_HDCP(x)	HDCP is enforced or not(0/1) when using HDMI output	int	Output control HDCP enforced: <0>
VOLOG_KEYWORD_OUTPUT_ CONTROL_ANTI_MIRROR(x)	Airplay anti-mirroring is enabled or not(0/1)	int	Output control anti-mirroring enabled: <0>
VOLOG_KEYWORD_OUTPUT_ CONTROL_CIT_DIGITAL(x)	Resolution is downgraded or not(0/1) when HDCP rule cannot be enforced	int	Output control CIT digital setting enabled: <1>
VOLOG_KEYWORD_OUTPUT_ CONTROL_HDMI_CONNECTE D(x)	HDMI cable status plugged/unplugged(0/1)	int	Output control HDMI connected: <1>
VOLOG_KEYWORD_OUTPUT_ CONTROL_MIRROR_CONNEC TED(x)	Airplay mirroring status connected/disconnected(0/1)	int	Output control Airplay mirroring connected: <0>
VOLOG_KEYWORD_OUTPUT_ CONTROL_BLOCK_OUTPUT(x)	Output is blocked/unblocked(0/1)	int	Output control output blocked: <0>

5.5.12 RTSP logs

Table 23 lists the keywords, keywords' definitions, parameter types, and sample output messages for the RTSP related logs labeled with LOGLEVEL=0.

Table 23. RTSP related logs

Keyword	**Definition**	**Parameter type**	**Sample output message**
VOLOG_KEYWORD_RTSP_SERVER(x,y)	Information of RTSP server address and port	char*, int	RTSP server connection address: <102.110.0.8> and port: <80>
VOLOG_KEYWORD_RTSP_DESCRIBE_REQUEST(x)	Information of RTSP DESCRIBE request string	char*	RTSP DESCRIBE request message: <(DESCRIBE rtsp://rtsp-helix.visualon.com/source/mp4/mpeg4aac/mpeg4_d1_1000kbps_30fps_aac_ste_48khz_128kbps.mp4 RTSP/1.0>
VOLOG_KEYWORD_RTSP_DESCRIBE_RESPONSE(x)	Information of RTSP DESCRIBE response string	char*	RTSP DESCRIBE response message: <(RTSP/1.0 200 OK>
VOLOG_KEYWORD_RTSP_SETUP_REQUEST(x)	Information of RTSP SETUP request string	char*	RTSP SETUP request message: <(SETUP rtsp://rtsp-helix.visualon.com/source/mp4/mpeg4aac/mpeg4_d1_1000kbps_30fps_aac_ste_48khz_128kbps.mp4/streamid=65736 RTSP/1.0>
VOLOG_KEYWORD_RTSP_SETUP_RESPONSE(x)	Information of RTSP SETUP response string	char*	RTSP SETUP response message: <(RTSP/1.0 200 OK>
VOLOG_KEYWORD_RTSP_PLAY_REQUEST(x)	Information of RTSP PLAY request string	char*	RTSP PLAY request message: <(PLAY rtsp://rtsp-helix.visualon.com/source/mp4/mpeg4aac/mpeg4_d1_1000kbps_30fps_aac_ste_48khz_128kbps.mp4/streamid=65736 RTSP/1.0>
VOLOG_KEYWORD_RTSP_PLAY_RESPONSE(x)	Information of RTSP PLAY response string	char*	RTSP PLAY response message: <(RTSP/1.0 200 OK>
VOLOG_KEYWORD_RTSP_TEARDOWN_REQUEST(x)	Information of RTSP PLAY request string	char*	RTSP TEARDOWN request message: <(TEARDOWN rtsp://rtsp-helix.visualon.com/source/mp4/mpeg4aac/mpeg4_d1_1000kbps_30fps_aac_

Keyword	Definition	Parameter type	Sample output message
			ste_48khz_128kbps.mp4 RTSP/1.0>
VOLOG_KEYWORD_RTSP_TEARDOWN_RESPONSE(x)	Information of RTSP PLAY response string	char*	RTSP TEARDOWN response message: <(RTSP/1.0 200 OK>
VOLOG_KEYWORD_RTSP_OPTIONS_REQUEST(x)	Information of RTSP OPTIONS request string	char*	RTSP OPTIONS request message: <(OPTIONS rtsp://rtsp-helix.visualon.com/source/mp4/mpeg4aac/mpeg4_d1_1000kbps_30fps_aac_ste_48khz_128kbps.mp4 RTSP/1.0>
VOLOG_KEYWORD_RTSP_OPTIONS_RESPONSE(x)	Information of RTSP OPTIONS response string	char*	RTSP OPTIONS response message: <(RTSP/1.0 200 OK>
VOLOG_KEYWORD_RTSP_GETPARAM_REQUEST(x)	Information of RTSP GET_PARAMETER request string	char*	RTSP GET_PARAMETER request message: <(GET_PARAMETER rtsp://rtsp-helix.visualon.com/source/mp4/mpeg4aac/mpeg4_d1_1000kbps_30fps_aac_ste_48khz_128kbps.mp4 RTSP/1.0>
VOLOG_KEYWORD_RTSP_GETPARAM_RESPONSE(x)	Information of RTSP GET_PARAMETER response string	char*	RTSP GET_PARAMETER response message: <(RTSP/1.0 200 OK>
VOLOG_KEYWORD_RTSP_GET_REQUEST(x)	Information of RTSP GET request string	char*	RTSP GET request message: <(GET rtsp://rtsp-helix.visualon.com/source/mp4/mpeg4aac/mpeg4_d1_1000kbps_30fps_aac_ste_48khz_128kbps.mp4 RTSP/1.0>
VOLOG_KEYWORD_RTSP_POST_REQUEST(x)	Information of RTSP POST request string	char*	RTSP POST request message: <(POST rtsp://rtsp-helix.visualon.com/source/mp4/mpeg4aac/mpeg4_d1_1000kbps_30fps_aac_ste_48khz_128kbps.mp4 RTSP/1.0>

5.5.13 Streaming logs

Table 24 lists the keywords, keywords' definitions, parameter types, and sample output messages for the streaming related logs labeled with LOGLEVEL=0.

Table 24. Streaming related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_STREAMING_PROTOCOL(x)	Streaming protocol type(HLS/DASH/MS-SSTR)	char*	Streaming protocol: <HLS>
VOLOG_KEYWORD_STREAMING_VERSION(x)	Streaming spec version	char*	Streaming version: <CE708>
VOLOG_KEYWORD_STREAMING_BITRATE(x)	All bitrates that streaming contains	char*	Streaming bitrate: <200000 bps>
VOLOG_KEYWORD_STREAMING_TYPE(x)	Streaming type(VoD/Live/DVR)	char*	Streaming type: <Live>
VOLOG_KEYWORD_STREAMING_DURATION(x)	Streaming duration information	int	Streaming duration: <879932112 ms>

5.5.14 SourceIO logs

Table 25 lists the keywords, keywords' definitions, parameter types, and sample output messages for the sourceIO related logs labeled with LOGLEVEL=0.

Table 25. SourceIO related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_SOURCEIO_PROXY(x,y)	Proxy address and port	char*, int	SourceIO proxy address: <108.110.0.8> and port: <8088>
VOLOG_KEYWORD_SOURCEIO_REDIRECT(x,y)	Redirection operation occurs when requesting a normal url	char*, char*	SourceIO redirects from url: <http://public.infozen.cshls.lldns.net/infozen/public/public.m3u8> to url: <http://public.infozen.cshls.lldns.net/infozen/public/public/public_1000.m3u8>

5.5.15 Subtitle logs

Table 26 lists the keywords, keywords' definitions, parameter types, and sample output messages for the sourceIO related logs labeled with LOGLEVEL=0.

Table 26. Subtitle related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_SUBTITLE_EXTERNAL(x)	Indicates external subtitle is played, and shows the subtitle URI	char*	Subtitle external subtitle URI: <http://108.110.0.2>
VOLOG_KEYWORD_SUBTITLE_INTERNAL	Indicates subtitle from streaming source is played	N/A	Subtitle internal subtitle
VOLOG_KEYWORD_SUBTITLE_SEPC(x)	Indicates the subtitle sepc, like TTML, SMPTE-TT, CC608, CC708, WEBVTT, DVB-Sub, and so on	char*	Subtitle spec: <WEBVTT>
VOLOG_KEYWORD_SUBTITLE_CONTAINER(x)	Indicates the subtitle container, like TS, MP4, Manifest, ID3	char*	Subtitle container: <MP4>

5.5.16 VOME2 engine logs

Table 27 lists the keywords, keywords' definitions, parameter types, and sample output messages for the VOME2 engine related logs labeled with LOGLEVEL=0.

Table 27. VOME2 engine related logs

Keyword	Definition	Parameter type	Sample output message
VOLOG_KEYWORD_VOME2ENGINE_BUFFER_START(x)	Indicates the buffering starts	char*	VOME2 engine <video> buffering starts
VOLOG_KEYWORD_VOME2ENGINE_BUFFER_END(x)	Indicates the buffering completes	char*	VOME2 engine <video> buffering completes
VOLOG_KEYWORD_VOME2ENGINE_AV_NSYNC(x)	Indicates the difference between render time and playback time	int	VOME2 engine Warning: AV render time diff(render time - playback time): <270048 ms>. AV may not be in sync!
VOLOG_KEYWORD_VOME2ENGINE_FRAME_DROP	Indicates frame dropping may cause video play in fast forward	N/A	VOME2 engine drops frame may cause video play in fast forward
VOLOG_KEYWORD_VOME2ENGINE_MODULE_FRAME_DROP(x)	Indicates dropping of video frame may cause the playback in fast forward	char*	VOME2 engine <video> drops frame may cause video play in fast forward
VOLOG_KEYWORD_VOME2ENGINE_READ_SOURCE(x)	Indicate VOME2 engine completes	int	VOME2 engine succeeds, <11 ms> used
VOLOG_KEYWORD_VOME2E	Indicates the information of	char*	VOME2 engine <video>

Keyword	Definition	Parameter type	Sample output message
NGINE_NEW_FORMAT(x)	new format		new format
VOLOG_KEYWORD_VOME2ENGINE_AV_ONLY(x)	Indicate the content type	char*	VOME2 engine <audio> only
VOLOG_KEYWORD_VOME2ENGINE_EOS(x)	Indicates the content reaches the end of the source	char*	VOME2 engine <video> reaches the end of the source
VOLOG_KEYWORD_VOME2ENGINE_INPUT_TIME_DIFF(x,y)	Indicates the time that input timestamp jumps	char*, int	VOME2 engine <audio> input timestamp jumps: <304600> ms
VOLOG_KEYWORD_VOME2ENGINE_DEC_OUTPUT_DIFF(x,y)	Indicates the time that output timestamp jumps	char*, int	VOME2 engine <audio> decoder output timestamp jumps: <10519> ms
VOLOG_KEYWORD_VOME2ENGINE_LATENCY_ABNM(x)	Indicates the audio latency is too large	int	VOME2 engine Warning: audio latency is too big: <304600> ms
VOLOG_KEYWORD_VOME2ENGINE_RENDER_DIFF(x,y)	Indicates the time that render timestamp jumps	char*, int	VOME2 engine <video> render timestamp jumps: <304582> ms

Chapter 6 License

This chapter presents the following topics:

6.1 Evaluation license

The evaluation license has an expiration time from the date when the build is created. You do not need the additional key to activate OSMP+.

6.2 Production license

The production license varies depending on the platform what you are using. The production license file needs to match the package name defined by customers on the Android platform. For Windows, Mac OS, iOS, and Windows Phone platforms, a key is needed to activate OSMP+.

6.2.1 Android

See the following for the required code to activate production license on Android.

```
        InputStream is = null;
        byte[] b = new byte[32*1024];
        try {
            is = context.getAssets().open("voVidDec.dat");
            is.read(b);
            is.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        m_sdkPlayer.setLicenseContent(b);
```

6.2.2 iOS

See the following for the required code to activate production license on iOS.

```
//Set license text
        id<VOCommonPlayerConfiguration> pPlayerConfiguration = 
self.player;
        NSString *strLicense = [[NSBundle mainBundle] 
pathForResource:@"voVidDec.dat" ofType:nil];
        [pPlayerConfiguration setLicenseFilePath:strLicense];
        [pPlayerConfiguration setPreAgreedLicense:@"License key 
here"];
```

6.2.3 Windows plug-in

The following is the needed code to activate production license on Windows plug-in.

```
// Write license in the Registry when install the plugin package
        // Set the Registry place in "voPluginConfig.cpp" in 
plugin source code file
        /*
        License key info,for example:
        HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\VisualOn\BrowserPl
ugin\PreAgreedLicense\key = PreAgreedLicenseString
        */
        const char* VOPLUGIN_REGISTER_PATH_ROOT = 
"SOFTWARE\VisualOn\BrowserPlugin\";
        const char* VOPLGUIN_ERGISTER_PRE_LICENSE_KEY = 
"PreAgreedLicense";
        const char* VOPLGUIN_ERGISTER_PRE_LICENSE_VALUE = "key";
```

6.2.4 Mac OS plug-in

See the following for the required code to activate production license on Mac OS plug-in.

```
// Put license in package folder /License/MacOS/
        // Set license text in VOPlugin-Info.plist
        <key>PreAgreedLicense</key>
        <string>License key here</string>
```

6.2.5 Windows Phone

See the following for the required code to activate production license on Windows Phone.

```
// read license file content.
         Platform::String^ fullPath = 
Windows::ApplicationModel::Package::Current->InstalledLocation-
>Path;
        Platform::String^ lisenceFileName = fullPath + 
"\\voVidDec.dat";
        char stra[2048];
        memset(stra, 0, sizeof(stra));
        WideCharToMultiByte(CP_ACP, 0, (WCHAR*)lisenceFileName-
>Data(), -1, stra, 1024, NULL, NULL);
        std::string strFilePath = ((char*)stra);

        FILE *f;
        fopen_s(&f, strFilePath.c_str(), "rb");
        fseek(f, 0, SEEK_END);
        long fsize = ftell(f);
        fseek(f, 0, SEEK_SET);
        unsigned char *licenseContent = (unsigned char 
*)malloc(fsize);
        fread(licenseContent, 1, fsize, f);
        fclose(f);
        int nRC = m_commonPlayer-
>setLicenseContent(licenseContent, fsize);
        nRC = m_commonPlayer->setPreAgreedLicense("License key 
here");
        free(licenseContent);
```

Chapter 7 Frequently Asked Questions

This chapter presents the following topics:

7.1 Frequently Asked Questions

The list that follows is designed to answer your questions about OSMP+.

Question 1: Why my device cannot play the video with the highest bitrate?

Answer 1: This problem may be caused by the configuration in *cap.xml* and your network bandwidth. Check the following to figure out the solution depending on your case:

- Record your device model, and then check if your device model is listed in the *cap.xml* file located under your installation package. Configurations in *cap.xml* may result in the limited bitrate on your device.
- Check your network bandwidth to see if the network bandwidth is enough to play the content with the highest bitrate.

Question 2: Why I see the expiration message on the player?

Answer 2: The evaluation message appears on the video screen to indicate that you are using the evaluation license. If your product is target to release to market, check if you have the production license. If not, contact VisualOn's sales.

Question 3: How can I create the player for the debug purpose?

Answer 3: The solution may vary depending on the operating system that you are using.

- Mac OS plug-in: Use the libraries under the *Macintosh /Libs/MacOS/x86/debug* directory
- Android: Use the .so files under the *\Libs\debug* directory and the .jar files under the *\Jar\debug* directory in your installation package.
- iOS: Use the .a files under the */Libs/debug* directory.
- Windows plug-in: Use the .dll files and configuration files under *\Libs\Win32\debug* directory.

Question 4: How can I quickly get started with the features provided by the OSMP+ product?

Answer 4: The integration Labs, which are compressed into the *labs.zip* file under **Doc** of your installation package, provide working examples to accompany the *OSMP+ Player SDK Integration Guide*. The integration Labs focus on individual integration topics and require installation of the SDK and license file. Start with *README.txt* compressed in the *labs.zip* file to learn how to use Labs.

Question 5: When an HLS stream is paused, what is the default time that OSMP+ will buffer?

Answer 5: The buffer time is the sum of maximum buffering time and one segment. The default maximum buffering time is 20 seconds.

Question 6: When a Live stream is paused, what is the default time that OSMP+ will buffer?

Answer 6: The buffer time is the sum of maximum buffering time and one segment. The default maximum buffering time is 20 seconds.

Question 7: Is there a limit to the value of *.setMaxBufferingTime*?

Answer 7: This value varies from the device. We do not have a limitation on the maximum buffering time.

Appendix A Programming reference

This appendix presents the following topics:

API reference

For more information about the OSMP+ API, refer to the *API Reference Manual.zip* file under the **Doc** directory in the installation package.

Events

This section briefly introduces error codes and event IDs used in OSMP+.

Error codes

VO_OSMP_ERR_NONE

Value: 0X00000000

Description: No error

Suggestion: The operation is successful. This error code can be ignored.

VO_OSMP_ERR_EOS

Value: 0X00000001

Suggestion: Reserved. Do not use.

VO_OSMP_ERR_RETRY

Value: 0X00000002

Suggestion: Reserved. Do not use.

VO_OSMP_ERR_FORMAT_CHANGE

Value: 0X00000003

Suggestion: Reserved. Do not use.

VO_OSMP_ERR_AUDIO_NO_NOW

Value: 0X00000010

Description: Audio is not available.

Suggestion: Status. The audio is not available, and user can choose to play the video or stop playback.

VO_OSMP_ERR_VIDEO_NO_NOW

Value: 0X00000011

Description: Video is not available

Suggestion: Status. Video is not available, and user can choose to play the video or stop playback.

VO_OSMP_ERR_FLUSH_BUFFER

Value: 0X00000012

Description: Buffer needs to be flushed

Suggestion: Status. This error code can be ignored.

VO_OSMP_ERR_VIDEO

Value: 0X80000004

Suggestion: Reserved. Do not use.

VO_OSMP_ERR_AUDIO

Value: 0X80000005

Suggestion: Reserved. Do not use.

VO_OSMP_ERR_OUTMEMORY

Value: 0X80000006

Description: Out of memory occurs. The player crashes in a moment.

Suggestion: Fatal error. It stops playback.

VO_OSMP_ERR_POINTER

Value: 0X80000007

Description: API calling sequence is incorrect or API parameters are invalid.

Suggestion: Fatal error. It stops playback.

VO_OSMP_ERR_PARAMID

Value: 0X80000008

Description: The current function is not implemented.

Suggestion: This is not a fatal error. Stop using the invalid parameter.

VO_OSMP_ERR_STATUS

Value: 0X80000009

Description: Cannot call the requested API under the current status.

Suggestion: Status. This error code can be ignored.

VO_OSMP_ERR_IMPLEMENT

Value: 0X8000000A

Description: The requested function is not implemented.

Suggestion: Function or parameter is not implemented. This error code can be ignored.

VO_OSMP_ERR_SMALLSIZE

Value: 0X8000000B

Suggestion: Reserved. Do not use.

VO_OSMP_ERR_OUT_OF_TIME

Value: 0X8000000C

Suggestion: Reserved. Do not use.

VO_OSMP_ERR_WAIT_TIME

Value: 0X8000000D

Suggestion: Reserved. Do not use.

VO_OSMP_ERR_UNKNOWN

Value: 0X8000000E

Description: Undefined operation or behavior

Suggestion: Fatal error. It stops playback

VO_OSMP_ERR_JNI

Value: 0X8000000F

Description: JNI library cannot be used on iOS.

Suggestion: Fatal error. It stops playback.

VO_OSMP_ERR_LICENSE_FAIL

Value: 0X80000011

Description: VisualOn product license is not valid.

Suggestion: Fatal error. It stops playback.

VO_OSMP_ERR_HTTPS_CA_FAIL

Value: 0X80000012

Description: HTTPS certification authority failed.

Suggestion: Fatal error. It stops playback.

VO_OSMP_ERR_ARGUMENT

Value: 0X80000013

Description: Passed the wrong parameter.

Suggestion: Fatal error. It stops playback.

VO_OSMP_ERR_MULTIPLE_INSTANCES_NOT_SUPPORTED

Value: 0X80000014

Description: Multiple instances are forbidden (in some cases).

Suggestion: Fatal error. It stops playback.

VO_OSMP_ERR_UNINITIALIZE

Value: 0X80000101

Description: Requested object cannot be initialized.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ERR_OPEN_SRC_FAIL

Value: 0X81000001

Description: Source cannot be opened.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ERR_CONTENT_ENCRYPT

Value: 0X81000002

Description: Content is encrypted, further operations are required.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ERR_PLAYMODE_UNSUPPORT

Value: 0X81000003

Description: Playmode is not supported on the current platform.

Suggestion: Status. Do not use this playmode.

VO_OSMP_SRC_ERR_ERROR_DATA

Value: 0X81000004

Description: Error in the source file.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ERR_SEEK_FAIL

Value: 0X81000005

Description: Requested seek operation failed. Retry or skip the seek operation.

Suggestion: Status. Seek failed. Seek again or skip the seek operation.

VO_OSMP_SRC_ERR_FORMAT_UNSUPPORT

Value: 0X81000006

Description: OSMP+ cannot support this media format.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ERR_TRACK_NOTFOUND

Value: 0X81000007

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_ERR_NO_DOWNLOAD_OP

Value: 0X81000008

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_ERR_NO_LIB_OP

Value: 0X81000009

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_ERR_OUTPUT_NOTFOUND

Value: 0X8100000A

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_ERR_CHUNK_SKIP

Value: 0X8100000B

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_ERR_SRC_UNINITIALIZE

Value: 0X80001001

Description: Source object is not initialized

Suggestion: Fatal error. It stops playback.

VO_OSMP_HTTP_CONNECT_FAILED

Value: 0x00000000

Description: Connection to HTTP server fails.

Suggestion: Status.

VO_OSMP_HTTP_INVALID_RESPONSE

Value: 0x00000001

Description: HTTP failed to get response or response cannot be parsed or is too large.

Suggestion: Status.

VO_OSMP_HTTP_CLIENT_ERROR

Value: 0x00000002

Description: HTTP 4xx error occurs.

Suggestion: Status.

VO_OSMP_HTTP_SERVER_ERROR

Value: 0x00000003

Description: HTTP 5xx error occurs.

Suggestion: Status.

VO_OSMP_SRC_ERR_DIVXUNSUPPORTED

Value: 0X8100000D

Description: Indicate that video does not support DIVX

Suggestion: Fatal error. It stops playback.

VO_OSMP_CB_EVENT_ID

This is the enumeration for the callback event IDs.

VO_OSMP_CB_ERROR

Value: 0X8000000C

Description: Undefined operation or behavior

Suggestion: Fatal error. It stops playback.

VO_OSMP_CB_MULTIPLE_INSTANCES_NOT_SUPPORTED

Value: 0X80000029

Description: Multiple instances are forbidden (in some cases).

Suggestion: Fatal error. It stops playback.

VO_OSMP_CB_PLAY_COMPLETE

Value: 0X00000001

Description: Source playback completed

Suggestion: Status. It can be ignored or stop playback.

VO_OSMP_CB_VIDEO_START_BUFFER

Value: 0X00000003

Description: Video stream started buffering.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_VIDEO_STOP_BUFFER

Value: 0X00000004

Description: Video stream stopped buffering.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_AUDIO_START_BUFFER

Value: 0X00000005

Description: Audio stream started buffering.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_AUDIO_STOP_BUFFER

Value: 0X00000006

Description: Audio stream stopped buffering.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_SRC_BUFFERING_TIME

Value: 0X00000007

Description: Buffering time in the source.

Suggestion: Information. This code can be ignored.

VO_OSMP_CB_SEEK_COMPLETE

Value: 0X0000000D

Description: The engine starts rendering the data.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_VIDEO_ASPECT_RATIO

Value: 0X0000000E

Description: Video aspect ratio is changed.

Suggestion: Information. This code can be ignored.

VisualOn

VO_OSMP_CB_VIDEO_SIZE_CHANGED

Value: 0X0000000F

Description: Video size is changed.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_CODEC_NOT_SUPPORT

Value: 0X80000010

Description: The requested decoder is not supported.

Suggestion: Fatal error. It stops playback.

VO_OSMP_CB_DEBLOCK

Value: 0X00000011

Description: Enable/disable deblock event of video codec

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_HW_DECODER_STATUS

Value: 0X00000013

Description: Hardware decoder status is available.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_AUTHENTICATION_RESPONSE

Value: 0X00000014

Description: Authentication response information, for example, report information from server;

Suggestion: Information. This code can be ignored.

VO_OSMP_CB_LANGUAGE_INFO_AVAILABLE

Value: 0X00000015

Description: Subtitle language info is parsed and is available

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_VIDEO_RENDER_START

Value: 0X00000016

Description: video render started

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_OPEN_SRC_COMPLETE

Value: 0X00000017

Description: Open Source completed

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_SEI_INFO

Value: 0X00000019

Description: Notify SEI info.

Suggestion: Information. This code can be ignored.

VO_OSMP_CB_AUDIO_RENDER_FAIL

Value: 0X8000001A

Description: The initialization of the audio track failed.

Suggestion: Fatal error. It stops playback.

VO_OSMP_CB_PCM_OUTPUT

Value: 0X0000001C

Description: This event is issued only when PCM data output is enabled.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_LICENSE_FAIL

Value: 0X8000001D

Description: The player cannot find a license or license is expired.

Suggestion: Fatal error. It stops playback.

VO_OSMP_CB_BLUETOOTHHANDSET_CONNECTION

Value: 0X00001011

Description: Bluetooth handset status

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_OUTPUT_CONTROL_BLOCK_PLAYBACK

Value: 0x00000024

Description: Playback should be stopped on both local and external devices

Suggestion: Fatal error. It stops playback

VO_OSMP_CB_OUTPUT_CONTROL_BLOCK_OUTPUT

Value: 0x00000025

Description: Sending output to external devices is blocked

Suggestion: Status. User can choose to play the video or stop playback.

VO_OSMP_CB_OUTPUT_CONTROL_CHANGE_RESOLUTION

Value: 0x00000026

Description: Resolution is changed by output control.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_OUTPUT_CONTROL_CAPTURE_SOFTWARE_RUNNING

Value: 0X0000002B

Description: Notify that the screen capturing software is running in the system.

Suggestion: Status. The user can choose to play the video or stop playback.

VO_OSMP_AD_CB_PLAYLIST_START

Value: 0X03000001

Description: Notify when the player starts displaying the first advertisement video or content video.

Suggestion: N/A. Not used or not available.

VO_OSMP_AD_CB_PLAYLIST_END

Value: 0X03000002

Description: Notify when the player completes displaying the last advertisement video or content video.

Suggestion: N/A. Not used or not available.

VO_OSMP_AD_CB_CONTENT_START

Value: 0X03000003

Description: Notify each time when a content is started.

Suggestion: The operation succeeded

VO_OSMP_AD_CB_CONTENT_END

Value: 0X03000004

Description: Notify when a content segment is finished.

Suggestion: The operation succeeded.

VO_OSMP_AD_CB_AD_START

Value: 0X03000005

Description: Notify when a video advertisement is started.

Suggestion: The operation succeeded.

VO_OSMP_AD_CB_AD_END

Value: 0X03000006

Description: Notify when a video advertisement is finished.

Suggestion: The operation succeeded.

VO_OSMP_AD_CB_VIDEO_PROGRESS

Value: 0X03000007

Description: Notify the current playing time.

Suggestion: The operation succeeded.

VO_OSMP_AD_CB_VIDEO_DONE

Value: 0X03000008

Description: Notify when the advertisement or content video is finished.

Suggestion: The operation succeeded.

VO_OSMP_AD_CB_STATE_CHANGE

Value: 0X03000009

Description: Notify when the status of player playback changes.

Suggestion: The operation succeeded.

VO_OSMP_AD_CB_PLAYBACKINFO

Value: 0X0300000A

Description: Get playback info.

Suggestion: The operation succeeded.

VO_OSMP_AD_CB_NO_AD_CONTENT

Value: 0X8300000B

Description: Notify when an advertisement has no content or does not be loaded as expected.

Suggestion: N/A. Not used or not available.

VO_OSMP_AD_CB_AD_LOAD_ERROR

Value: 0X8300000C

Suggestion: Reserved. Do not use.

VO_OSMP_AD_CB_VIDEO_NOT_AVAILABLE

Value: 0X8300000D

Description: Notify when the content video is expired or is unavailable.

Suggestion: Fatal error. It stops playback.

VO_OSMP_AD_CB_DATA_LOAD_ERROR

Value: 0X8300000E

Description: Notify when the content video experiences a fatal error while loading or playback

Suggestion: Fatal error. It stops playback.

VO_OSMP_AD_CB_VIDEO_GEO_BLOCKED

Value: 0X8300000F

Description: Notify when the GEO blocked

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_CB_CONNECTING

Value: 0X02000001

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_CB_CONNECTION_FINISHED

Value: 0X02000002

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_CB_CONNECTION_TIMEOUT

Value: 0X82000003

Description: Source connection times out

Suggestion: This is a fatal error for PD module only.

VO_OSMP_SRC_CB_CONNECTION_LOSS

Value: 0X82000004

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_CB_DOWNLOAD_STATUS

Value: 0X02000005

Description: HTTP download status.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_CONNECTION_FAIL

Value: 0X82000006

Description: Source connection failed

Suggestion: This is a fatal error for PD module. For Adaptive Streaming, this is not a fatal error.

VO_OSMP_SRC_CB_DOWNLOAD_FAIL

Value: 0X82000007

Description: Source download failed

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_DRM_FAIL

Value: 0X82000008

Description: Device provisioning failed or data decryption failed.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_CB_PLAYLIST_PARSE_ERR

Value: 0X82000009

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_CB_CONNECTION_REJECTED

Value: 0X8200000A

Description: Reaches the maximum number of connections. This event is only applicable for RTSP.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_CB_BA_HAPPENED

Value: 0X0200000B

Description: Bitrate is changed; param1 is new bitrate value (bps integer type)

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_DRM_NOT_SECURE

Value: 0X0200000C

Description: Device is rooted or is jail broken. This error code is only applicable for showtime DRM.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_CB_DRM_AV_OUT_FAIL

Value: 0X8200000D

Description: Initialization of an AV (Audio/Video) file failed. This error code is only applicable for showtime DRM.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_CB_DOWNLOAD_FAIL_WAITING_RECOVER

Value: 0X8200000E

Description: Download failed, waiting for recovery

Suggestion: Status. It can be ignored or user can stop playback.

VO_OSMP_SRC_CB_DOWNLOAD_FAIL_RECOVER_SUCCESS

Value: 0X0200000F

Description: Download recovery is successful.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_OPEN_FINISHED

Value: 0X02000010

Description: Source open completed.

Suggestion: Status. Send a notification when the source open is completed.

VO_OSMP_SRC_CB_CUSTOMER_TAG

Value: 0X02000020

Description: Customer tag information is available inside the source.

Suggestion: Status. Handle this information or ignore it.

VO_OSMP_SRC_CB_ADAPTIVE_STREAMING_INFO

Value: 0X02000030

Description: Streaming information

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_ADAPTIVE_STREAMING_ERROR

Value: 0X02000040

Description: Adaptive streaming error

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_CB_ADAPTIVE_STREAM_WARNING

Value: 0X02000050

Description: Adaptive streaming error warning.

Suggestion: The App can ignore it or collect warning information; *VOOSMP_SRC_CB_Adaptive_Streaming_Error* prompts after collecting several warning information.

VO_OSMP_SRC_CB_RTSP_ERROR

Value: 0X02000060

Description: Notify RTSP error.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_CB_SEEK_COMPLETE

Value: 0X02000070

Description: Reserved. Do not use.

VO_OSMP_SRC_CB_PROGRAM_CHANGED

Value: 0X02000071

Description: Notify when the program information is changed in source

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_PROGRAM_RESET

Value: 0X02000072

Description: Notify when the program information is reset in source

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_ADAPTIVE_STREAMING_SEEK2LASTCHUNK

Value: 0X02000073

Description: Notify when seeking to the last chunk of playlist (NTS link without END tag needs change to live mode)

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_NOT_APPLICABLE_MEDIA

Value: 0X02000074

Description: Network is not available now, but the player still has some buffer data that can be played.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_PD_DOWNLOAD_POSITION

Value: 0X02000075

Description: Notify the current media position downloaded by the progressive download module.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_PD_BUFFERING_PERCENT

Value: 0X02000076

Description: Notify the current buffering percent of the progressive download module.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_UPDATE_URL_COMPLETE

Value: 0X02000077

Description: Update source URL completed.

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_SYNC_EVENT_ID

This is the enumeration for the callback sync event IDs. These events are sent from the sub-thread. Do not update UI or call other VisualOn's API in callback.

VO_OSMP_SRC_CB_ IO_HTTP_START_DOWNLOAD

Value: 0X04000001

Description: Start HTTP download.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_ IO_HTTP_DOWNLOAD_FAIL

Value: 0X04000002

Description: HTTP download failed.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CB_DRM_INIT_DATA

Value: 0X03000004

Description: Event of DRM initialization data.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_CUSTOMERTAGID

This is the enumeration of HLS customer tag IDs, and is used in *VO_OSMP_SRC_CB_Customer_Tag* callback.

VO_OSMP_SRC_CUSTOMERTAGID_TIMEDTAG

Value: 0X00000001

Description: The player calls back the time tag of HLS streaming with this tag ID

Suggestion: Information. This code can be ignored.

VO_OSMP_SRC_CB_IO_HTTP_DOWNLOAD_FAIL

Value: 0X04000002

Description: The player allows to retry the download.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT

This is the enumeration of available stream info event, and is used in callback.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_BITRATE_CHANGE

Value: 0X00000001

Description: Bitrate is changed.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_MEDIATYPE_CHANGE

Value: 0X00000002

Description: Media type is changed.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_PROGRAM_TYPE

Value: 0X00000003

Description: Program type.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_CHUNK_BEGINDOWNL
OAD

Value: 0X00000004

Description: Chunk download starts. If the returned value of this callback is *VO_OSMP_RETURN_CODE* or *VO_OSMP_SRC_ERR_CHUNK_SKIP*, source must drop this chunk. This event is synced with callback by *onRequestListener*, a value should be returned.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_CHUNK_DROPPED

Value: 0X00000005

Description: Downloaded chunk is dropped.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_CHUNK_DOWNLOAD
OK

Value: 0X00000006

Description: Chunk download completed.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_PLAYLIST_DOWNLOAD
OK

Value: 0X00000007

Description: Indicate that playlist download completed.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_PROGRAM_CHANGE

Value: 0X00000008

Description: Indicates that program is changed in the source. When this event is received, you should get program information again. Ignore Parameter 2.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_FILE_FORMATSUPPORT
ED

Value: 0X00000009

Description: Indicates that chunk is supported

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_LIVESEEKABLE

Value: 0X0000000A

Description: Indicates that the live clip can be sought now.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_DISCONTINUE_SAMPLE

Value: 0X0000000B

Description: Indicates that this is the first sample from the discontinue chunk.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_ADAPTIVE_STREAMING_ERROR_EVENT

This is the enumeration of available streaming error codes, and is used in *VO_OSMP_SRC_CB_Adaptive_Streaming_Error* callback.

VO_OSMP_SRC_ADAPTIVE_STREAMING_ERROR_EVENT_PLAYLIST_PARSEFAIL

Value: 0X00000001

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_ADAPTIVE_STREAMING_ERROR_EVENT_PLAYLIST_UNSUPPORTED

Value: 0X00000002

Suggestion: Reserved. Do not use.

VO_OSMP_SRC_ADAPTIVE_STREAMING_ERROR_EVENT_STREAMING_UNSUPPORTED

Value: 0X00000003

Description: Stream is not supported.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ADAPTIVE_STREAMING_ERROR_EVENT_STREAMING_DOWNLOADFAIL

Value: 0X00000004

Description:

- If the download of master playlist is failed, this event is emitted immediately.
- If the download of media playlist is failed:

    - Live: this event will be emitted after two minutes.
    - VOD: this event will be emitted after HLS parser attempts to retry the download of total playsession (multiple playlists) five times. For every single playlist, the HLS parser attempts to retry the download four times.
- If the download of media segment is failed:
    - Live: this event will be emitted after two minutes.
    - VOD: every segment is retried for 3 times. This event will be emitted if continuous ten segments have been retried.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ADAPTIVE_STREAMING_ERROR_EVENT_STREAMING_DRMLICENSEERROR

Value: 0X00000005

Description: DRM error occurs.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ADAPTIVE_STREAMING_ERROR_EVENT_STREAMING_VOLIBLICENSEERROR

Value: 0X00000006

Description: Notify License error

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_ADAPTIVE_STREAMING_WARNING_EVENT

This is the enumeration of available streaming warning events, and is used in *VO_OSMP_SRC_CB_Adaptive_Stream_Warning* callback.

VO_OSMP_SRC_ADAPTIVE_STREAMING_WARNING_EVENT_CHUNK_DOWNLOADERROR

Value: 0X00000001

Description: Chunk download is failed.

Suggestion: Status. An adaptive streaming chunk fails to download.

VO_OSMP_SRC_ADAPTIVE_STREAMING_WARNING_EVENT_CHUNK_FILEFORMATUNSUPPORTED

Value: 0X00000002

Description: Chunk format is not supported.

Suggestion: Status. An adaptive streaming chunk's media format is not supported.

VO_OSMP_SRC_ADAPTIVE_STREAMING_WARNING_EVENT_CHUNK_DRMERROR

Value: 0X00000003

Description: DRM error occurs.

Suggestion: Status. The chunk decryption operation failed.

VO_OSMP_SRC_ADAPTIVE_STREAMING_WARNING_EVENT_PLAYLIST_DOWNLOADERROR

Value: 0X00000004

Description: Playlist download failed.

Suggestion: Status. Download of adaptive streaming's playlist failed.

VO_OSMP_AVAILABLE_TRACK_TYPE

This is the enumeration of available track types, and is used in *VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT* or *VO_OSMP_SRC_ADAPTIVE_STREAMING_INFO_EVENT_MEDIATYPE_CHANGE* callback.

VO_OSMP_AVAILABLE_PUREAUDIO

Value: 0X00000000

Description: Only audio is available.

Suggestion: Status. User can choose to play the audio only or stop playback.

VO_OSMP_AVAILABLE_PUREVIDEO

Value: 0X00000001

Description: Only video is available.

Suggestion: Status. User can choose to play video only or stop playback.

VO_OSMP_AVAILABLE_AUDIOVIDEO

Value: 0X00000002

Description: Both audio and video are available.

Suggestion: Status.

VO_OSMP_SRC_RTSP_ERROR

This is the enumeration of RTSP error.

VO_OSMP_SRC_RTSP_ERROR_CONNECT_FAIL

Value: 0X00000001

Description: Connection to the RSTP server failed.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_RTSP_ERROR_DESCRIBE_FAIL

Value: 0X00000002

Description:

- The player failed to send DESCRIBE command to the RTSP server.
- The player cannot get DESCRIBE response from the RTSP server.
- The RTSP server cannot reply using the response of 'RTSP/1.0 200 OK'.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_RTSP_ERROR_SETUP_FAIL

Value: 0X00000003

Description:

- The player failed to send SETUP command to the RTSP server.
- The player cannot get SETUP response from the RTSP server.
- The RTSP server cannot reply using the response of 'RTSP/1.0 200 OK'.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_RTSP_ERROR_PLAY_FAIL

Value: 0X00000004

Description:

- The player failed to send PLAY command to the RTSP server.
- The player cannot get PLAY response from the RTSP server.
- The RTSP server cannot reply using the response of 'RTSP/1.0 200 OK'.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_RTSP_ERROR_PAUSE_FAIL

Value: 0X00000005

Description:

- The player failed to send PAUSE command to the RTSP server.
- The player cannot get PAUSE response from the RTSP server.
- The RTSP server cannot reply using the response of 'RTSP/1.0 200 OK'.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_RTSP_ERROR_OPTION_FAIL

Value: 0X00000006

Description:

- The player failed to send OPTION command to the RTSP server.
- The player cannot get OPTION response from the RTSP server.
- The RTSP server cannot reply using the response of 'RTSP/1.0 200 OK'.

Suggestion: Status. This code can be ignored.

VO_OSMP_SRC_RTSP_ERROR_SOCKET_ERROR

Value: 0X00000007

Description:

- Socket error occurs when receiving the data.
- The player cannot receive the data after timeout.

Suggestion: Fatal error. It stops playback.

VO_OSMP_SRC_RTSP_ERROR_MAX

Value: 0XFFFFFFFF

Description: Maximum value definition

Suggestion: Fatal error. It stops playback.

VO_OSMP_CB_STREAMING_DOWNLOADER_EVENT_ID

This is the enumeration for the callback sync event IDs. This event is sent from the sub thread. Do not update UI or call other VisualOn's API in callback.

VO_OSMP_CB_STREAMING_DOWNLOADER_OPEN_COMPLETE

Value: 0X10000001

Description: Open operation completed.

Suggestion: Status

VO_OSMP_CB_STREAMING_DOWNLOADER_MANIFEST_OK

Value: 0X10000002

Description: Manifest file download is successful.

Suggestion: The Player can play back the URL.

VO_OSMP_CB_STREAMING_DOWNLOADER_END

Value: 0X10000004

Description: Entire content download completed.

Suggestion: This code can be ignored.

VO_OSMP_CB_STREAMING_DOWNLOADER_PROGRAM_INFO_CHANGE

Value: 0X10000005

Description: Program information is changed.

Suggestion: Status

VO_OSMP_CB_STREAMING_DOWNLOADER_MANIFEST_UPDATE

Value: 0X10000006

Description: Manifest file is updated.

Suggestion: Status

VO_OSMP_CB_STREAMING_DOWNLOADER_DOWNLOAD_MANIFEST_FAIL

Value: 0X90000001

Description: Download of manifest file failed because of network issues.

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_WRITE_MANIFEST_FAIL

Value: 0X90000002

Description: Write operation to manifest file failed.

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_DOWNLOAD_CHUNK_FAIL

Value: 0X90000003

Description: Three minutes after chunk download failed.

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_WRITE_CHUNK_FAIL

Value: 0X90000004

Description: Write operation to chunk file failed.

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_DISK_FULL

Value: 0X90000005

Description: Disk is full.

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_GENERATE_MANIFEST_FAIL

Value: 0X90000006

Description: Generation of manifest fails

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_LIVE_STREAM_NOT_SUPPORT

Value: 0X90000007

Description: Live stream is not supported

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_LOCAL_STREAM_NOT_SUPPORT

Value: 0X90000008

Description: Stream on local disk is not supported.

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_LOCAL_PLAYLIST_INIT_FAIL

Value: 0X90000009

Description: Playlist initialization failed

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_LISENCE_CHECK_FAIL

Value: 0X9000000A

Description: License check failed

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_INIT_IO_FAIL

Value: 0X9000000B

Description: Initialization of IO failed

Suggestion: Fatal error

VO_OSMP_CB_STREAMING_DOWNLOADER_SWITCH_STATUS_FAIL

Value: 0X9000000C

Description: Switching status failed

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_STREAMING_DOWNLOADER_CHECK_STATUS_FAIL

Value: 0X9000000D

Description: Checking status failed

Suggestion: Status. This code can be ignored.

VO_OSMP_CB_STREAMING_DOWNLOADER_CREATE_FOLDER_FAIL

Value: 0X90000010

Description: Download manager fails to create the folder.

Suggestion: Fatal error

Appendix B Configuration Files

This appendix presents the following topics:

Cap file

The device capability file (*cap.xml*) is an Extensible Markup Language (XML) file that specifies the maximum bitrate that the SDK tries to play when targeting a variant playlist (multiple tracks of the same content) on a specific type of device. The maximum bitrate defined in the device capability file overrides the default initial bitrate selection. The maximum bitrate is also respected by the SDK bitrate adaptation algorithm.

Note: The bitrate cap defined in the *cap.xml* file is only applied when targeting a variant playlist. If there is only a single audio or video track, the bitrate cap does not apply.

Clusters or groups of devices can be identified in the device capability file by their platform characteristics. Platforms characteristics include the following:

- Number of cores
- Availability of optimizations
- CPU frequency

The device capability file can be packaged with the application or be downloaded at runtime, and is supported across all platforms including Android, iOS, Windows browser plug-in, and Mac OS browser plug-in.

The *cap.xml* file is located:

Note: Do not change the file name of *cap.xml* or *hw_cap.xml*.

- Android: under the *\SamplePlayer\assets* directory of your package.
- iOS: under the */SamplePlayer/AppUI/config* directory of your package.
    - *cap.xml*: for OSMP+ usage
    - *hw_cap.xml*: for AV player usage
- Windows plug-in: under the \Libs\Win32 directory of your package.
- Mac OS plug-in: under the /Project/MacOS directory of your package.

For more information about the *cap.xml* file, refer to the *OnStream MediaPlayer+.Device Capability File AG.pdf* under the **Doc** directory of your installation package.

White List

OSMP+ offers the option to change the default audio and video hardware decoding settings for specific devices configurations. The list of devices configurations (also known as the white list) is saved in the *device.xml* file under the *assets* directory of the Android package.

Refer to 1.4.7.1.2 White List for details about the *device.xml* file.

Appendix C Reference Documentation

This appendix presents the following topics:

VisualOn Documentation

The following documents, available from your installation package, provide additional and relevant information. If you cannot find any document, contact your VisualOn representative.

- *API Reference Manual.zip*
- *Labs.zip*
- *OnStream MediaPlayer+.Device Capability File AG.pdf*
- *OnStream MediaPlayer+.Player SDK Integration Guide for Android.pdf*
- *OnStream MediaPlayer+.SDK Project Setup for Android.pdf*
- *OnStream MediaPlayer+.Player SDK Integration Guide for iOS.pdf*
- *OnStream MediaPlayer+.SDK Project Setup for iOS.pdf*
- *OnStream MediaPlayer+ Plugin SDK Integration Guide for Mac OS.pdf*
- *OnStream MediaPlayer+ Plugin SDK Integration Guide for Windows.pdf*
- *OnStream MediaPlayer+ Plugin UI-Control API Reference.pdf*

Links

Refer to the following topics for the references related to separate platforms when integrating or interfacing with the OSMP+ product.

Note: The links provided were working correctly at the time of publication.

- Android 4.4 APIs
- SurfaceView on Android
- TextureView on Android
- Tags for HTTP Live Streaming
- RFC 2326: Real Time Streaming Protocol (RTSP)
- RFC 2327: SDP: Session Description Protocol
- RFC 3550: RTP: A Transport Protocol for Real-Time Applications
- RFC 3551: RTP Profile for Audio and Video Conferences with Minimal Control
- RFC 3984: RTP Payload Format for H.264 Video
- RFC 3016: RTP Payload Format for MPEG-4 Audio/Visual Streams
- RFC 4629: RTP Payload Format for ITU-T Rec.H.263 Video

- RFC 3267: Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (ARM-WB) Audio Codecs
- RFC 4352: RTP Payload Format for the Extended Adaptive Multi-Rate Wideband (AMR-WB+) Audio Codec
- RFC 2658: RTP Payload Format for PureVoice (tm) Audio
- MS-RTSP: RTP Payload Format for ASF Streams
- Single File Streaming
- [MS-WMSP]: Windows Media HTTP Streaming Protocol
- Dolby Digital Plus
- Analyzing Display and Performance
- Android Performance Case Study
- Improving Layout Performance
- Improving Your Code with lint
- Optimizing Your UI
- Using the Dev Tools App
- Timed Text Markup Language 1 (TTML)
- WebVtt: The Web Video Text Tracks Format
- FCC Notice