Qualcomm Technologies, Inc.

Application to Qualcomm® DASH Player 2

Interface Specification for Release 5.4

80-72114-1 Rev. A

October 26, 2023

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision History

Revision	Date	Description
A	Oct 2023	New version with CEA 608 feature (captions and timed text)

Contents

1 Introduction

1.1 Purpose

This document describes how to use the Qualcomm Dynamic Adaptive Streaming over HTTP (DASH) Player API in a UI application.

1.2 Conventions

Function declarations, function names, type declarations, and code samples appear in a different font, e.g., `#include`.

Code variables appear in angle brackets, e.g., `<number>`.

Commands and command variables appear in a different font, e.g., **`copy a:*.* b:`**.

Button and key names appear in bold font, e.g., click **Save** or press **Enter**.

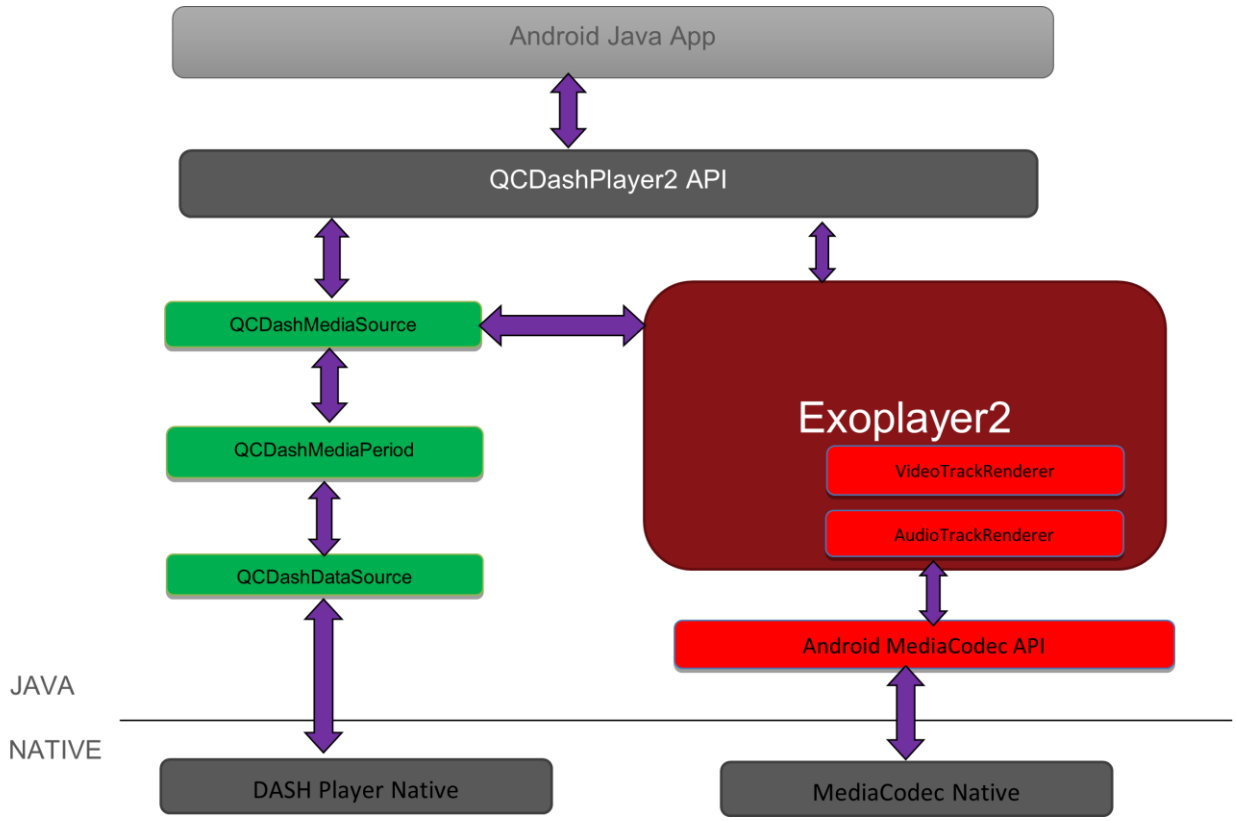
1.3 Support

For support, visit the LTE Broadcast SDK web page on the Qualcomm Developer Network (QDN): https://developer.qualcomm.com/ltebroadcast.

2 Overview

The Qualcomm DASH Player API enables the application to play DASH encoded contents on Android devices. These APIs are based on the Google ExoPlayer API and uses Android's low-level media APIs, such as MediaCodec and AudioTrack.

The following diagram shows the Qualcomm DASH Player architecture.



The Qualcomm DASH Player consists of the following parts:

- **JAVA**
  - Qualcomm DASH player API – Provides an easy-to-use API to enable the UI app for consuming the DASH encoded content.
  - ExoPlayer2 – Google ExoPlayer2 implementation that consists of Video track renderer, Audio track renderer, and other player components.
  - QCDashMediaSource – Qualcomm extension of MediaSource component of ExoPlayer2 for injecting media content into ExoPlayer2.
  - QCDashMediaPeriod – Qualcomm extension of MediaPeriod component of ExoPlayer2, for defining period boundaries for DASH content.
  - QCDashDataSource – Component that interfaces with the Native part of the Qualcomm DASH player for getting Multimedia data.
- **Native**
  - DASH Player Native – Responsible for getting DASH segments from DASH server (local server or remote server), per MPD and Segment timelines. Also responsible for providing parsed video and audio frames to be used in the JAVA part of the DASH Player for playback.
  - MediaCodec Native – Responsible for decoding video and audio frames to be rendered by ExoPlayer2.

3 Qualcomm DASH Player Call Flows

The Qualcomm DASH Player provides an easy-to-use API to enable the UI application for consuming the DASH encoded content. This chapter describes the call flows of various scenarios using these APIs.

3.1 Playback

3.1.1 Create player

The application creates the QC DASH Player using `QCDashPlayerFactory.createPlayer()`. `createPlayer()` takes application context as input.

The resources allocated during `QCDashPlayerFactory.createPlayer()` are released when the application calls the `QCDashPlayer.release()`.

3.1.2 Add listener

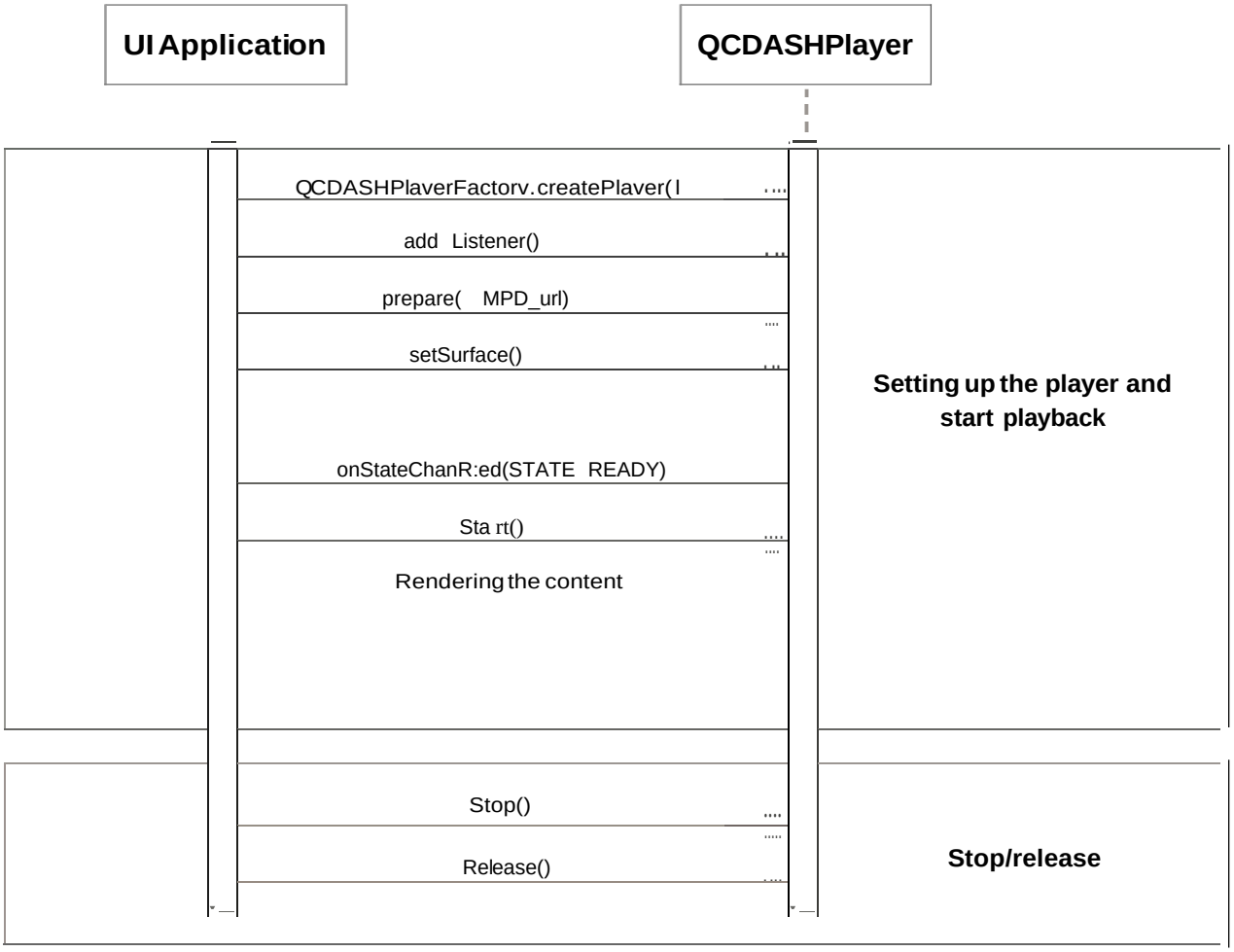
The application adds listener using `QCDashPlayer.addListener()` to receive the Qualcomm DASH Player notifications. The application implements the `QCDashPlayerListener` callback.

3.1.3 Start and stop player

`QCDashPlayer.prepare()` should be called to set the MPD URL of the content. In addition, the application should set the surface using `QCDashPlayer.setSurface()`. When the application gets the `onStateChanged.STATE_READY` notification, it starts rendering by calling `QCDashPlayer.start()`.

To stop the player, call `QCDashPlayer.stop()` and then `QCDashPlayer.release()` to free resources.

3.1.4 Call flow



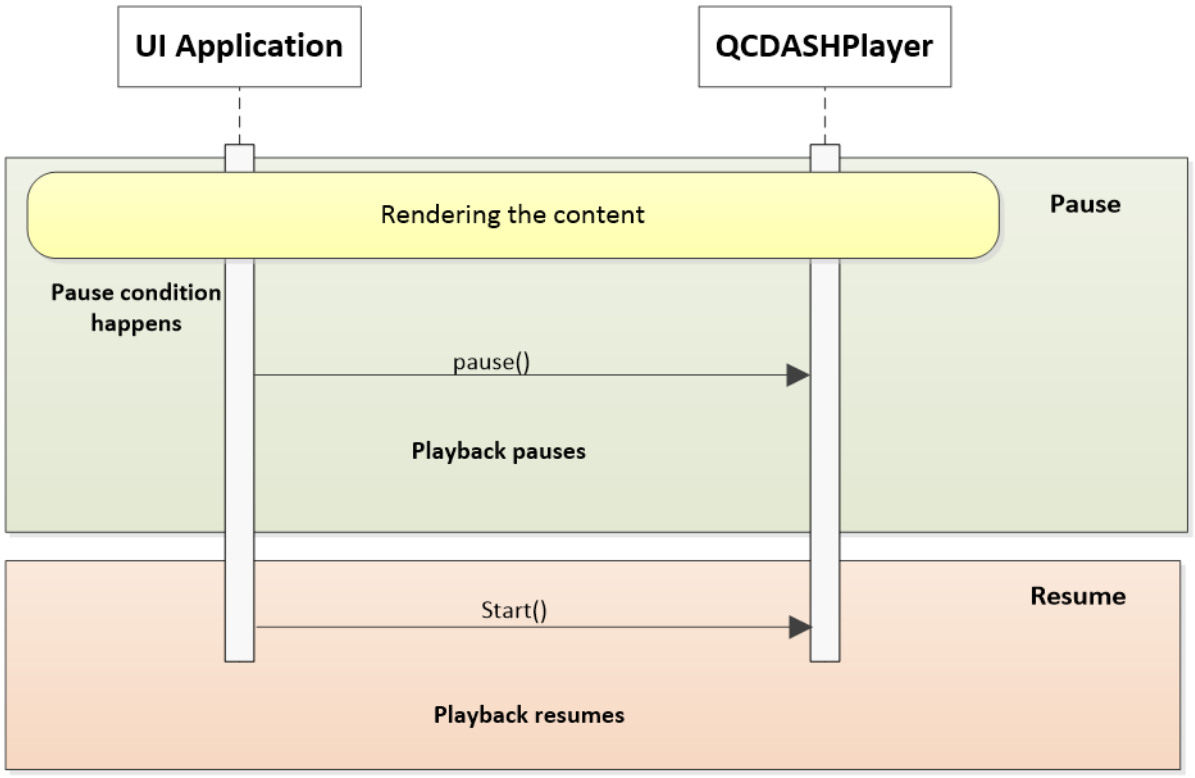
3.2 Pause and resume

When the application is paused, either by clicking the pause icon or going to the background, `QCDashPlayer.pause()` should be called.

When the application is ready to resume playback, it calls `QCDashPlayer.start()` to resume rendering.

Note: If the Surface object has changed, `QCDashPlayer.setSurface()` should be called again.

3.2.1 Call flow

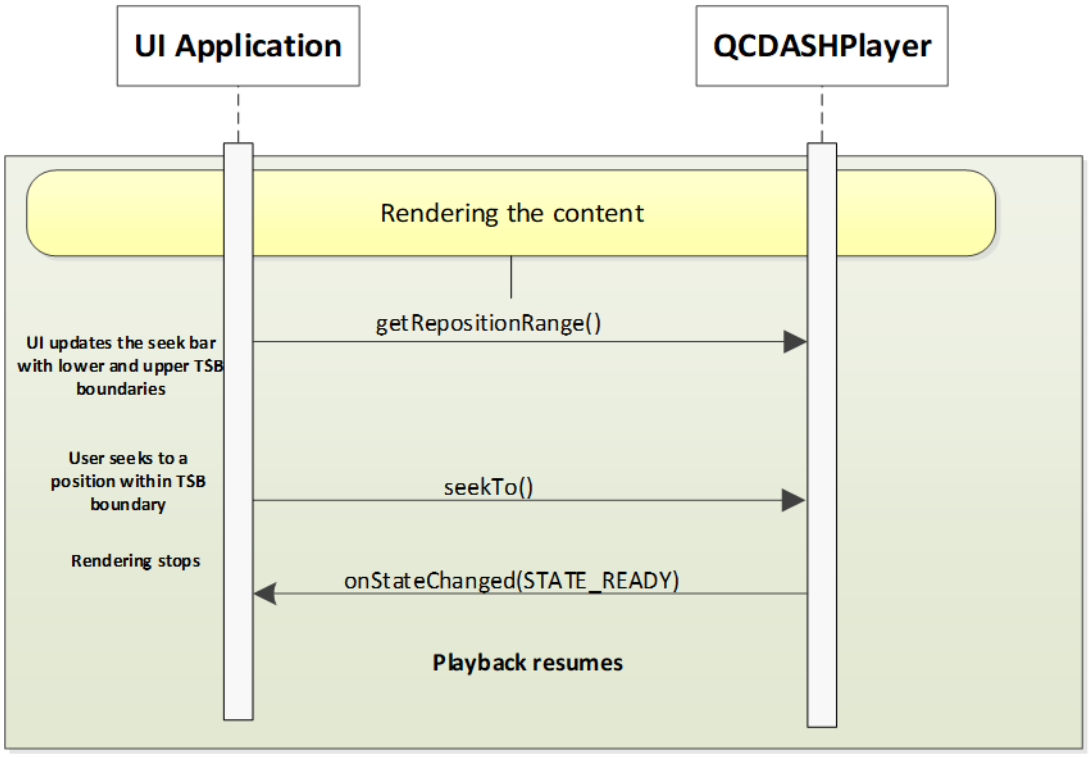


3.3 Seek

3.3.1 Time Shift Buffer boundaries

The application updates the seek bar boundaries using `QCDashPlayer.getRepositionRange()` to get the lower and upper boundary based on available Time Shift Buffer (TSB) boundaries. The application calls `QCDashPlayer.seekTo()` to reposition playback within TSB boundaries.

3.3.2 Call flow



3.4 Subtitle and track selection

The application creates and sets the subtitle layout on the player using `QCDashPlayer.setSubtitleLayout()`.

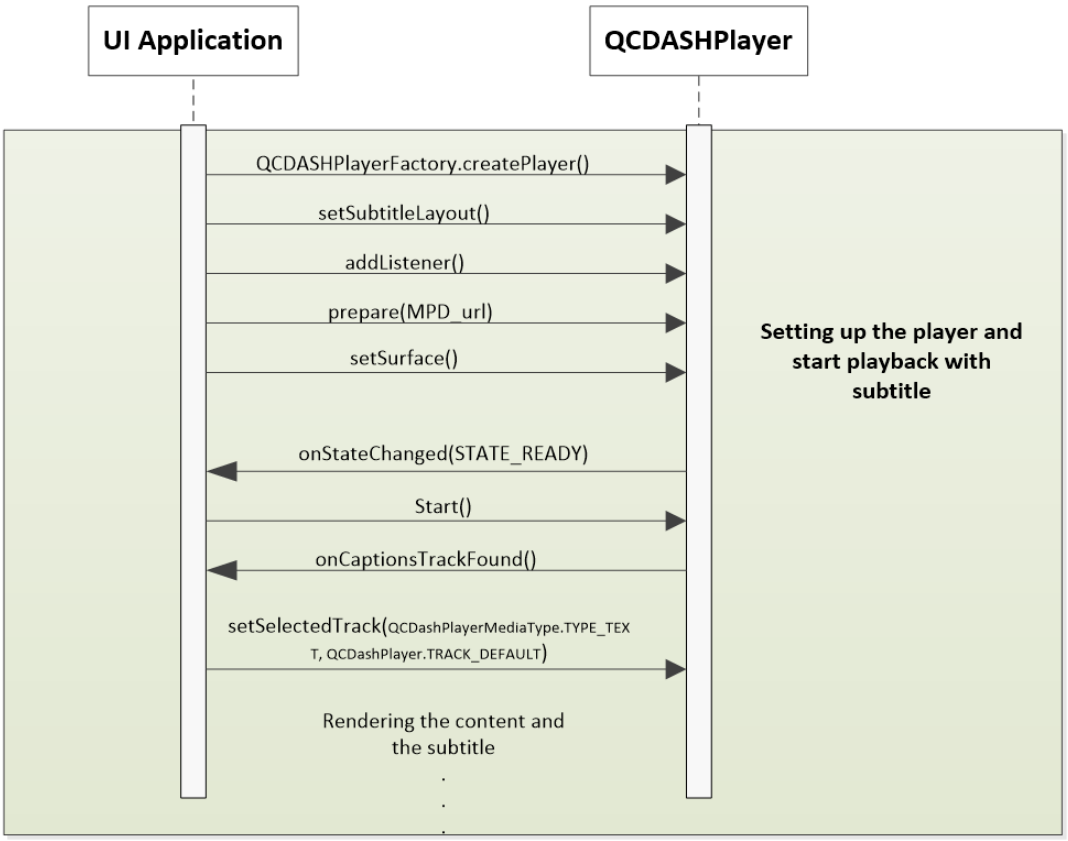
The Qualcomm DASH Player API provides an easy way to render the subtitle of the content. Subtitle rendering is handled by the player. The application should create and set subtitle layout on the player using the `QCDashPlayer.setSubtitleLayout()` method.

Note: The player cannot create a subtitle layout internally and have the layout set on it.

In addition, the application should register for `QCDashPlayer.QCDashPlayerListener()` to receive the captions found callback. The `onCaptionsTrackFound()` callback is sent whenever a closed captions channel is found in the video stream. Then, `QCDashPlayer.setSelectedTrack()` is called to set the mediaType and track index and start rendering the subtitle, `setSelectedTrack(QCDashPlayerMediaType.TYPE_TEXT, QCDashPlayer.TRACK_DEFAULT)`

To disable the subtitle rendering the app should call `setSelectedTrack(QCDashPlayerMediaType.TYPE_TEXT, QCDashPlayer.TRACK_DISABLED)`

3.4.1 Call flow

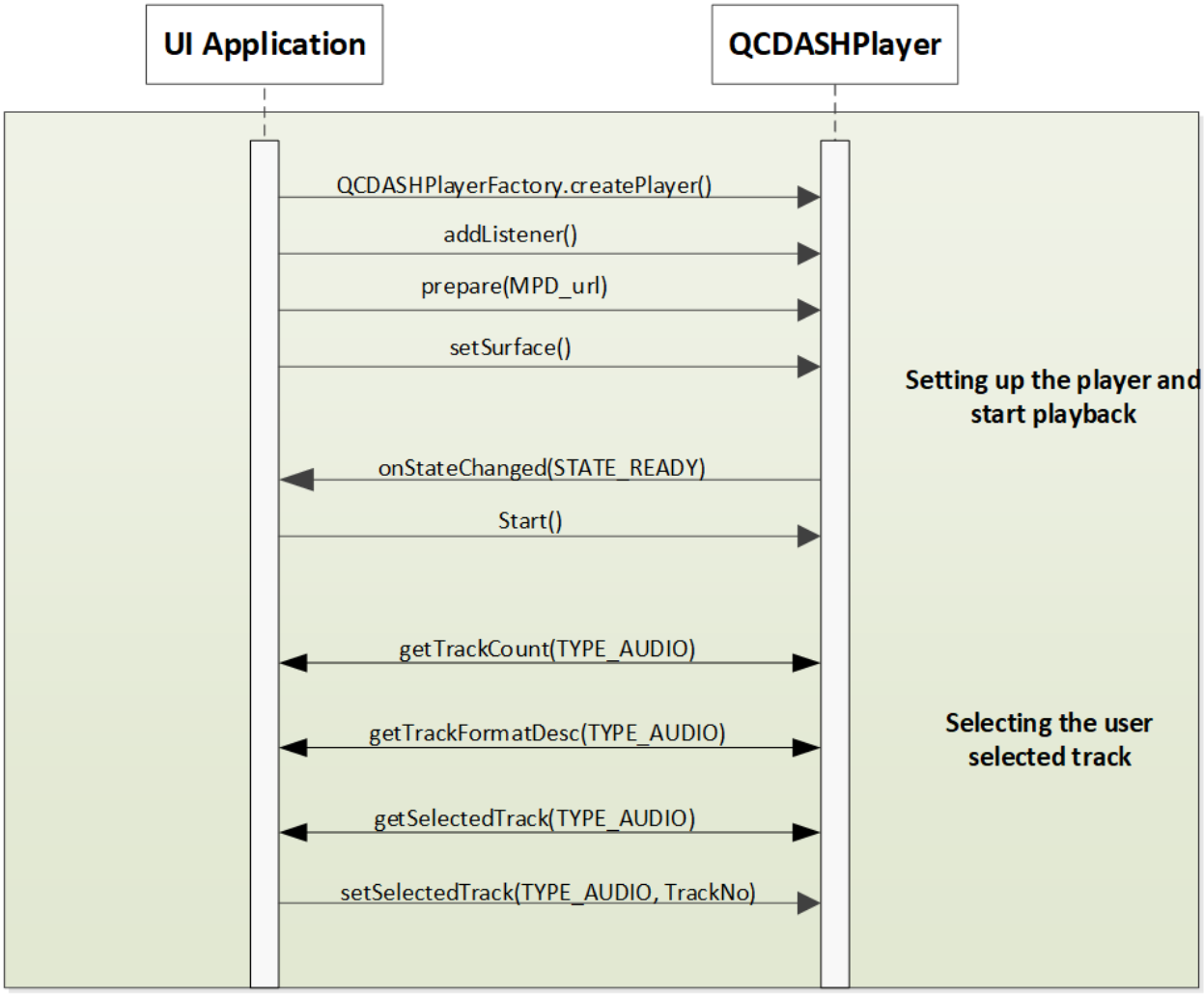


3.5 Multiple audio and track selection

The Qualcomm DASH Player API provides a way to get the number of audio tracks if the stream consists of multiple audio tracks. The application can select an audio track from the available tracks of a particular stream using the following APIs:

1. `QCDashPlayer.getTrackCount()` gets the number of available tracks for each media type, `getTrackCount(QCDashPlayerMediaType.TYPE_AUDIO)`.
2. `QCDashPlayer.getTrackFormatDesc()` gets the track format description for each track of a media type, `getTrackFormatDesc(QCDashPlayerMediaType.TYPE_AUDIO)`.
   - The application can use `QCDashPlayer.getSelectedTrack()` to get the current selected track number, `getSelectedTrack(QCDashPlayerMediaType.TYPE_AUDIO)`.
   - The application can use the method `QCDashPlayer.setSelectedTrack()` to select the user requested track, `setSelectedTrack(QCDashPlayerMediaType.TYPE_AUDIO, trackNo)`.

3.5.1 Call flow

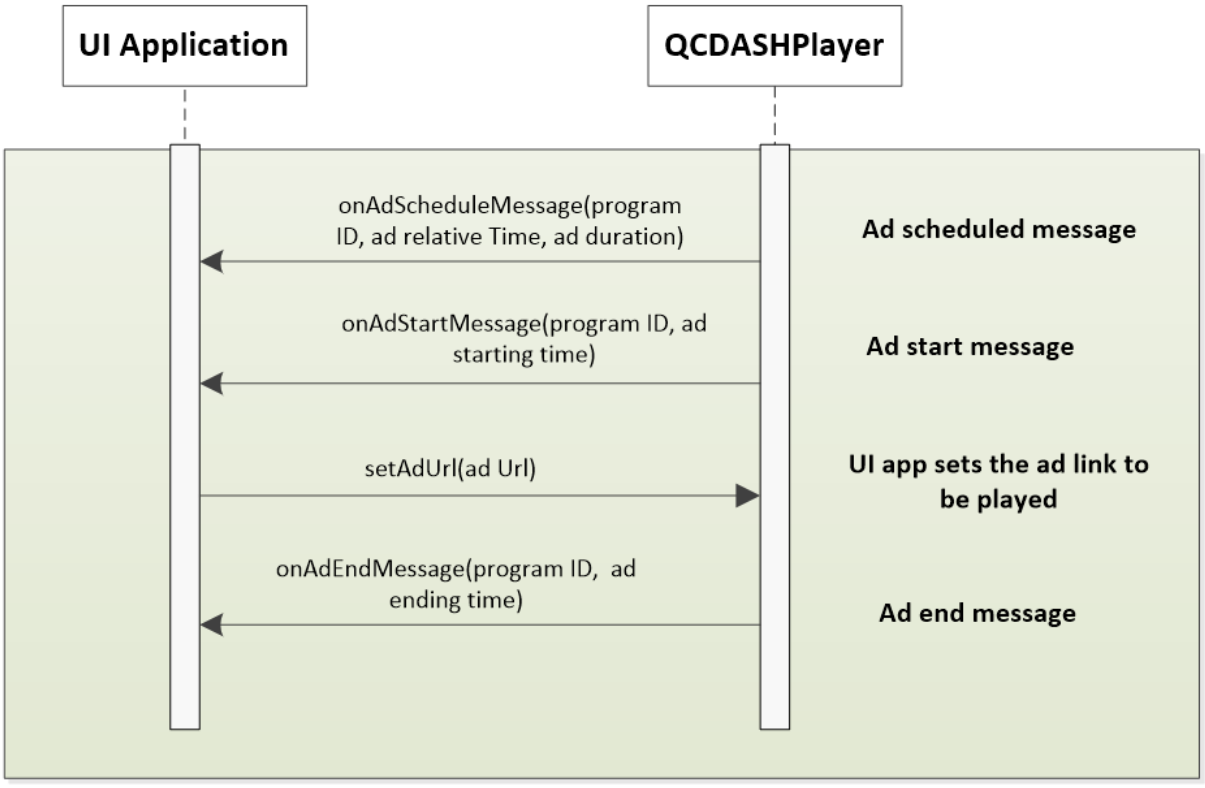


3.6 Ad-insertion during streaming play

The Qualcomm DASH Player API supports the ad-insertion feature. The DASH API provides information on ad schedule, ad start, and ad end to the UI application. The UI application can set the appropriate ad link (local file or HTTP) to the DASH API to play the ad video. A unique program ID will be mentioned on DASH APIs so that the UI application can set the ad link according to the program ID.

- `setAdUrl(String adUrl)` sets the ad link by UI app.
- `onAdScheduleMessage(int programID, int adrelativePtsTime, int adDuration)` provides information on an upcoming ad event to the UI application.
- `onAdStartMessage(int programID, int adStartingInSec)` tells the UI application that the ad will start with a specified time and UI app to set the ad link.
- `onAdEndMessage(int programID, int adEndinginSec)` informs the UI application that the ad currently playing will end at a specified time.

3.6.1 Call flow



4 Sample Code

This chapter provides sample code that shows how to use the DASH API.

4.1 Initialization

```
// Media player instance
   private QCDashPlayer mQCDashPlayer = null;
   private QCDashPlayerSubtitleLayout mQCDashPlayerSubtitleLayout = null;

mQCDashPlayer = QCDashPlayerFactory.createPlayer(this.getApplicationContext());

//Get the subtitle layout
QCDashPlayerSubtitleLayout mQcDashPlayerSubTSubtitleLayout =
(QCDashPlayerSubtitleLayout) streamingFragmentView
                        .findViewById(R.id.qc_dashplayer_subtitle_layout);

//set the subtitle layout to handle closed caption
mQCDashPlayer.setSubtitleLayout(this.mQCDashPlayerSubtitleLayout);


//Add listeners to handle the notifications
addListeners();

//Set the MPD URL
String url = theMPD_URL;
mQCDashPlayer.prepare(url, "SAMPLE");

//set surface
mQCDashPlayer.setSurface(surfaceObj);
```

4.2 Subtitle layout sample in res/layout/layout.xml

```
<com.qualcomm.dashplayer.api.QCDashPlayerSubtitleLayout
                  android:id="@+id/qc_dashplayer_subtitle_layout"
                  android:layout_alignBottom="@id/videosurfaceview"
                  android:layout_width="match_parent"
                  android:layout_height="match_parent"/>
```

4.3 Adding listeners and implementing the callback

```
private void addListeners() {

  mQCDashPlayer.addListener(new QCDashPlayerListener() {

      @Override
      public void onStateChanged(QCDashPlayer qcDashPlayer, QCDashPlayerState
      qcDashPlayerState)
      {
           Log.i(TAG,"onStateChanged QCDashPlayerState = " + qcDashPlayerState);
           if (qcDashPlayerState == QCDashPlayerState.STATE_READY) {
             mQCDashPlayer.start();

           //Add your code here to handle other state change notifications


      @Override
      public void onError(QCDashPlayer qcDashPlayer, Exception e)

          Log.i(TAG,"onError e = " + e.toString());
          //Reset the player and retry


      @Override
      public void onVideoSizeChanged(QCDashPlayer qcDashPlayer, int width, int
      height, int unappliedRotationDegrees, float pixelWidthHeightRatio)
      {
           Log.i(TAG,"onVideoSizeChanged w = "+width+", h = "+height
                  +", unappliedRotationDegress = "+unappliedRotationDegrees
                  +", pixelWidthHeightRatio = "+pixelWidthHeightRatio);



      @Override
      public void onCaptionsTrackFound(QCDashPlayer qcDashPlayer)
      {
           mQCDashPlayer.setSelectedTrack(QCDashPlayerMediaType.TYPE_TEXT,
                                          QCDashPlayer.TRACK_DEFAULT);


));
```

4.4 Getting TSB boundaries and seek

```
private long[] getTSBRangeValues()
{
  long lowerBoundaryValue = -1,upperBoundaryValue = -1;

  QCDashPlayerRepositionRangeinfo repositionRangeinfo =
                                  mQCDashPlayer.getRepositionRange();
  if(repositionRangeinfo != null){
    lowerBoundaryValue = repositionRangeinfo.getLowerBoundary();
    upperBoundaryValue = repositionRangeinfo.getUpperBoundary() ;
    Log.i("getTSBRangeValues  mSeekLowerBoundary = "
                             + lowerBoundaryValue + " mSeekUpperBoundary
                             +upperBoundaryValue);
     else{
        Log.i("getTSBRangeValues :repositionRangeinfo is null");


  long[] _tsbRanges = new long[2];
   tsbRanges[0] = lowerBoundaryValue;
   tsbRanges[1] = upperBoundaryValue;
  Log.i("getTSBRangeValues :return lowerBoundary   "+_tsbRanges[0]+", upperBoundary
                                                   "+ tsbRanges[1]);

  return  tsbRanges;



private void handleSeek(int progress)
{
  Log.i("handleSeek");
  mQCDashPlayer.seekTo(progress) ;
```

4.5 Audio track selection

```
int audioTrackCount =
          mQCDashPlayer.getTrackCount(QCDashPlayerMediaType.TYPE_AUDIO);

if(audioTrackCount > 1){
    String[] audioTrackArray = new String[audioTrackCount];
    List<String> audioTracks =
          mQCDashPlayer.getTrackFormatDesc(QCDashPlayerMediaType.TYPE_AUDIO);
     if(audioTracks!=null && audioTracks.size()>1){
        int i=0;
         for(String atrack: audioTracks){
             MSDCUILog.i("Audio Track Desc "+atrack +" "+i);
             audioTrackArray[i] = atrack+" "+i;
             i++;
        }
        int currentlyRunningAudioTrack =
            mQCDashPlayer.getSelectedTrack(QCDashPlayerMediaType.TYPE_AUDIO);
        MSDCUILog.i("Running audio track is "+ currentlyRunningAudioTrack);
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle(R.string.lang_desc)
             .setSingleChoiceItems(audioTrackArray, currentlyRunningAudioTrack,
             new DialogInterface.OnClickListener()
             {
               @Override
               public void onClick(DialogInterface dialogInterface, int i)
               {
                MSDCUILog.i("Changing the audio track to "+ i);

              mQCDashPlayer.setSelectedTrack(QCDashPlayerMediaType.TYPE_AUDIO, i);
                 dialogInterface.dismiss();
               }
             });
        AlertDialog mDialog = builder.create();
        mDialog.show();
    }
```

4.6 Release player

```
private void releaseMediaPlayer()
{
      if (mQCDashPlayer != null) {
            mQCDashPlayer.stop();
            mQCDashPlayer.release();
            mQCDashPlayer = null;
      }
}
```

5 Android Pie Requirement

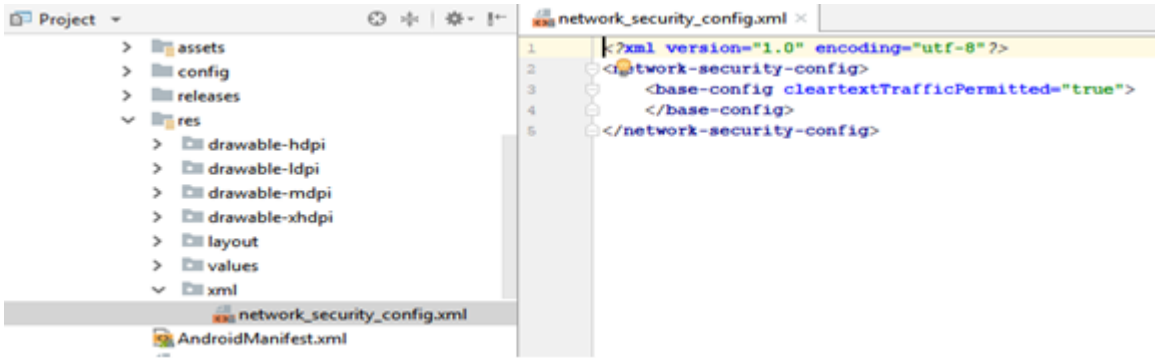
Android Pie and above must add “networkSecurityConfig” in AndroidManifest.xml to bypass the Secure Sockets Layer (SSL) if your server does not support SSL:

```
<application
        android:networkSecurityConfig="@xml/network_security_config"
    ...
</application>
```

Create the `network_security_config.xml` file under the `res/xml` directory and copy the following:

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <base-config cleartextTrafficPermitted="true">
    </base-config>
</network-security-config>
```

Android Pie example



6 Data Structure Documentation

6.1 QCDashPlayer

6.1.1 Class Documentation

6.1.1.1 interface com::qualcomm::dashplayer::api::QCDashPlayer

The Qualcomm DASH player is built on the ExoPlayer framework, exposing traditional high-level media player functionality, such as the ability to play, pause, seek, etc.

Public Member Functions

- String getPlayerVersion ()
- void addListener (QCDashPlayerListener listener)
- void removeListener (QCDashPlayerListener listener)
- void prepare (String uri, String userAgent)
- void start ()
- void pause ()
- void seekTo (long positionMs)
- void stop ()
- void release ()
- void setSurface (android.view.Surface surface)
- Surface getSurface ()
- void setSubtitleLayout (QCDashPlayerSubtitleLayout layout)
- QCDashPlayerSubtitleLayout getSubtitleLayout ()
- int getTrackCount (QCDashPlayerMediaType type)
- android.media.MediaFormat getTrackFormat (QCDashPlayerMediaType type, int index)
- int getSelectedTrack (QCDashPlayerMediaType type)
- void setSelectedTrack (QCDashPlayerMediaType type, int index)
- long getDuration ()
- long getCurrentPosition ()

- QCDashPlayerState getPlaybackState ()
- boolean isPlaying ()
- void setMute (boolean mute)
- QCDashPlayerRepositionRangeInfo getRepositionRange ()
- void setAdUrl (String adUrl)
- public String [] getCaptionsLanguages();
- public void enableCaptions(boolean setCaptionOn);
- public void enableTimedText(boolean setTimedTextOn);

Static Public Attributes

- static final int TRACK_DISABLED = -1
- static final int static final int TRACK_DEFAULT = 0
- static final int static final int static final long UNKNOWN_TIME = -1

6.1.1.1.1 Member Function Documentation

6.1.1.1.1. 6 static final int static final int static final long String com.qualcomm.dashplayer.api.QCDash- Player.getPlayerVersion ( )

AST update.

CUE message update. Mood Update type. Returns the version of QCDashPlayer.

Returns

QCDashPlayer version as a string. Format: a.b.c.d.

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.addListener (QCDashPlayerListener *listener*)

Registers a listener to receive events from the player.

The listener's methods will be invoked on the player thread.

Parameters

listener	The listener QCDashPlayerListener to register.

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.removeListener (QCDashPlayerListener *listener*)

Unregisters a listener.

The listener will no longer receive events from the player.

Parameters

listener	The listener QCDashPlayerListener to unregister.

6.1.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayer.prepare (String *uri*, String *userAgent*)

Prepares the player for playback.

Parameters

uri	DASH (MPD) URL.
userAgent	User-agent string to be used in network messages (optional).

6.1.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayer.start ( )

Starts playback when player is ready.

If the player is paused, then this method can be used to resume playback.

6.1.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayer.pause ( )

Pauses playback.

6.1.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayer.seekTo (long *positionMs*)

Seeks to a specified playback position.

Parameters

positionMs	The seek position (in milliseconds).

6.1.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayer.stop ( )

Stops playback.

Use pause() rather than this method if the intention is to pause playback.

Calling this method will cause the playback state to transition to QCDashPlayerState#STATE_IDLE. The player instance can still be used, and release() must still be called on the player if it is no longer required.

6.1.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayer.release ( )

Releases the player.

This method must be called when the player is no longer required.

The player must not be used after calling this method.

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.setSurface (android.view.Surface *surface*)

Sets the drawing surface on the player.

If invoked with a null surface, clears the existing surface.

Parameters

surface	The surface `android.view.Surface` to set.

6.1.1.1.1. 6 Surface com.qualcomm.dashplayer.api.QCDashPlayer.getSurface ( )

Gets the current drawing surface from the player.

Returns

The current drawing surface `android.view.Surface`.

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.setSubtitleLayout (QCDashPlayer- SubtitleLayout *layout*)

Sets the subtitle layout view on the player, which it uses to render subtitles.

This call is needed for the player to handle subtitle rendering. QCDashPlayerSubtitleLayout is a view instance that should be created by the application and passed on to the player, for the player to render subtitles.

Note

There is currently no caption listener interface exposed that allows the caller to register and receive subtitles. In a future event when an application wants to handle the subtitles, rendering a caption listener will be added.

Parameters

layout	The QCSubtitleLayout QCDashPlayerSubtitleLayout to set.

6.1.1.1.1. 6 QCDashPlayerSubtitleLayout com.qualcomm.dashplayer.api.QCDashPlayer.getSubtitle- Layout ( )

Gets the subtitle view from the player.

Returns

The current subtitle view QCDashPlayerSubtitleLayout.

6.1.1.1.1. 6 int com.qualcomm.dashplayer.api.QCDashPlayer.getTrackCount (QCDashPlayerMedia- Type *type*)

Returns the number of tracks for the specified media type.

Parameters

type	The media type QCDashPlayerMediaType.

Returns

The number of tracks.

6.1.1.1.1. 6 android.media.MediaFormat com.qualcomm.dashplayer.api.QCDashPlayer.getTrackFormat (QCDashPlayerMediaType *type*, int *index*)

Returns the media format of the specified track.

Parameters

type	The media type QCDashPlayerMediaType of the track.
index	The index of the track.

Returns

The format of the track (android.media.MediaFormat).

6.1.1.1.1. 6 int com.qualcomm.dashplayer.api.QCDashPlayer.getSelectedTrack (QCDashPlayerMedia- Type *type*)

Returns the index of the currently selected track for the specified media type.

Parameters

type	The media type QCDashPlayerMediaType of the selected track.

Returns

The selected track index. A negative value, or a value greater than or equal to the track count, indicates that the track is disabled.

6.1.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayer.setSelectedTrack (QCDashPlayer- MediaType *type*, int *index*)

Selects a track for the specified media type.

Parameters

type	The media type QCDashPlayerMediaType of the selected track.
index	The index of the track. A negative value or a value greater than or equal to the track count will disable the renderer.

6.1.1.1.1.6 com.qualcomm.dashplayer.api.QCDashPlayer.getDuration ( ) long

Gets the duration in milliseconds.

If there are multiple tracks in the stream, the maximum duration across all tracks is returned.

Returns

The duration (of the stream) in milliseconds, or QCDashPlayer#UNKNOWN_TIME if the duration is not known.

6.1.1.1.1.6 com.qualcomm.dashplayer.api.QCDashPlayer.getCurrentPosition ( ) long

Gets the current playback position in milliseconds.

Returns

The current playback position in milliseconds.

6.1.1.1.1.6 com.qualcomm.dashplayer.api.QCDashPlayer.getPlaybackState ( ) QCDashPlayerState

Gets the player state.

Returns

The current playback state - one of the QCDashPlayerState values as defined in the QCDashPlayerState interface.

6.1.1.1.1.6 com.qualcomm.dashplayer.api.QCDashPlayer.isPlaying ( ) boolean

Checks if the player is playing.

Returns

Whether player is playing (TRUE) or not (FALSE).

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.setMute (boolean *mute*)

Instructs audio to be muted/unmuted.

Parameters

mute	Whether audio should mute (when mute is true) or unmute (mute is false).

6.1.1.1.1. 6 QCDashPlayerRepositionRangeInfo com.qualcomm.dashplayer.api.QCDashPlayer.get- RepositionRange ( )

Gets the reposition range information (e.g., for repositioning in a stored viewing of a live stream).

Returns

The reposition range information QCDashPlayerRepositionRangeInfo.

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.setAdUrl (String *adUrl*)

Sends Ad Event Url information in response to onAdStartMessage.

Parameters

adUrl	The ad URL, can be a local file path or HTTP link..

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.getCaptionsLanguages ()

Gets the list of languages supported with CEA enabled stream.

Returns

List of Caption language.

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.enableCaptions (boolean setCaptionOn)

Instructs to enable/disable captions.

Parameters

setCaptionOn	When the setCaptionOn is set to true will enable Captions.

6.1.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayer.enableTimedText (boolean setTimedTextOn)

Instructs to enable/disable timed text.

Parameters

setTimedTextOn	When the setTimedTextOn is set to true will enable timed text.

6.1.1.1.2 Member Data Documentation

6.1.1.1.2. 6 final int com.qualcomm.dashplayer.api.QCDashPlayer.TRACK_DISABLED = -1

A value that can be passed as the second argument to setSelectedTrack to disable a track.

6.1.1.1.2. 6 final int static final int com.qualcomm.dashplayer.api.QCDashPlayer.TRACK_DEFAULT = 0

A value that can be passed as the second argument to setSelectedTrack to select the default track.

6.1.1.1.2. 6 final int static final int static final long com.qualcomm.dashplayer.api.QCDashPlayer.UNKNO- WN_TIME = -1

Represents an unknown time or duration.

6.2 QCDashPlayerFactory

6.2.1 Class Documentation

6.2.1.1 class com::qualcomm::dashplayer::api::QCDashPlayerFactory

A factory for creating the Qualcomm DASH player.

Static Public Member Functions

- static QCDashPlayer createPlayer (Context context)

6.2.1.1.1 Member Function Documentation

6.2.1.1.1.1 static QCDashPlayer com.qualcomm.dashplayer.api.QCDashPlayerFactory.createPlayer (Context *context*)

Creates and obtains an instance of the Qualcomm DASH player.

Parameters

context	Application context `android.content.Context`.

Returns

The player QCDashPlayer instance.

6.3 QCDashPlayerListener

6.3.1 Class Documentation

6.3.1.1 interface com::qualcomm::dashplayer::api::QCDashPlayerListener

Qualcomm DASH player listener definition for core player event notifications.

Public Member Functions

- void onStateChanged (QCDashPlayer qdp, QCDashPlayerState playbackState)
- void onError (QCDashPlayer qdp, Exception e)
- void onVideoSizeChanged (QCDashPlayer qdp, int width, int height, int unappliedRotationDegrees, float pixelWidthHeightRatio)
- void onCaptionsTrackFound (QCDashPlayer qdp)
- void onTimedTextReceived(String timedTextString);
- void onAdScheduleMessage (int programID, int adrelativePtsTime, int adDuration)
- void onAdStartMessage (int programID, int adStartingInSec)
- void onAdEndMessage (int programID, int adEndinginSec)

6.3.1.1.1 Member Function Documentation

6.3.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayerListener.onStateChanged (QCDash- Player *qdp*, QCDashPlayerState *playbackState*)

Invoked when the player state changes.

Parameters

qdp	Reference to DASH player QCDashPlayer.
playbackState	One of the QCDashPlayer state constants as defined in the QCDashPlayerState interface.

6.3.1.1.1.6 void com.qualcomm.dashplayer.api.QCDashPlayerListener.onError (QCDashPlayer *qdp*, Exception *e*)

Invoked when an error occurs.

The player instance may still be used (based on the error), and QCDashPlayer#release() must be called on the player should it no longer be required.

Parameters

qdp	Reference to DASH player QCDashPlayer.
e	The error `java.lang.Exception`.

6.3.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayerListener.onVideoSizeChanged (QCDash- Player *qdp*, int *width*, int *height*, int *unappliedRotationDegrees*, float *pixelWidthHeightRatio*)

Invoked when video size is first known or updated.

Parameters

qdp	Reference to DASH player QCDashPlayer.
width	The video width in pixels.
height	The video height in pixels.
unappliedRotation-Degrees	For videos that require rotation, this is the clockwise rotation in degrees that the application should apply to the video for it to be rendered in the correct orientation. This value will always be zero on API levels 21 and above, since the renderer will apply all necessary rotations internally. On earlier API levels, this is not possible. Applications that use `android.view.TextureView` can apply the rotation by calling `android.view.TextureView::setTransform`. Applications that do not expect to encounter rotated videos can safely ignore this parameter.
pixelWidthHeight-Ratio	The width-to-height ratio of each pixel. In normal cases with square pixels this will equal 1.0. Different values are indicative of anamorphic content.

6.3.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayerListener.onCaptionsTrackFound (QCDashPlayer *qdp*)

Invoked when a closed-caption channel is found in the video stream.

Parameters

qdp	Reference to DASH player QCDashPlayer.

6.3.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayerListener.onTimedTextReceived (String timedTextString)

Invoked when timed text is enabled and a timedTextString is received.

Parameters

timedTextString	Timed text string.

6.3.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayerListener.onAdScheduleMessage (int *programID*, int *adrelativePtsTime*, int *adDuration*)

Invoked when MPD receives an Ad scheduled cue message.

Parameters

programID	The uniqueID identifying the cue message.
adrelativePtsTime	The relative time in seconds after the ad is scheduled to play.
adDuration	The duration of time which an ad will play.

6.3.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayerListener.onAdStartMessage (int *programID*, int *adStartingInSec*)

Invoked when MPD receives Ad start cue message.

Parameters

programID	The uniqueID identifying the cue message.
adStartingInSec	The time in seconds before an add starts playing.

6.3.1.1.1. 6 void com.qualcomm.dashplayer.api.QCDashPlayerListener.onAdEndMessage (int *programID*, int *adEndinginSec*)

Invoked when MPD receives an Ad end cue message.

Parameters

programID	The uniqueID identifying the cue message.
adEndinginSec	The time in seconds that an ad will end.

6.4 QCDashPlayerMediaType

6.4.1 Class Documentation

6.4.1.1 enum com::qualcomm::dashplayer::api::QCDashPlayerMediaType

Public Member Functions

- int getCode ()

Static Public Member Functions

- [static initializer]
- static QCDashPlayerMediaType get (int code)

Public Attributes

- TYPE_VIDEO = (0)
- TYPE_AUDIO = (1)
- TYPE_TEXT = (2)
- TYPE_METADATA = (3)
- TYPE_UNKNOWN = (4)

6.4.1.1.1 Member Function Documentation

6.4.1.1.1. 6 com.qualcomm.dashplayer.api.QCDashPlayerMediaType.[static initializer] ( )

Static initializer of QCDashPlayerMediaType class when it gets loaded in memory.

6.4.1.1.1. 6 int com.qualcomm.dashplayer.api.QCDashPlayerMediaType.getCode ( )

Gets QCDashPlayerMediaType as an integer type.

6.4.1.1.1. 6 static QCDashPlayerMediaType com.qualcomm.dashplayer.api.QCDashPlayerMediaType.get (int *code*)

Gets QCDashPlayerState as an enumeration based on the integer value passed.

6.4.1.1.2 Member Data Documentation

6.4.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerMediaType.TYPE_VIDEO = (0)

Video media type.

6.4.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerMediaType.TYPE_AUDIO=(1)

Audio media type.

6.4.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerMediaType.TYPE_TEXT=(2)

Text media type.

6.4.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerMediaType.TYPE_METADATA = (3)

Metadata media type.

6.4.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerMediaType.TYPE_UNKNOWN = (4)

Unknown media type.

6.5 QCDashPlayerRepositionRangeInfo

6.5.1 Class Documentation

6.5.1.1 class com::qualcomm::dashplayer::api::QCDashPlayerRepositionRangeInfo

Reposition range information (e.g., TSB window parameters - left edge, right edge, depth).

Public Member Functions

- QCDashPlayerRepositionRangeInfo (long min, long max, long depth)
- QCDashPlayerRepositionRangeInfo ()
- final long final long final long long getLowerBoundary ()
- long getUpperBoundary ()
- long getDepth ()

Package Attributes

- final long mLowerBoundary
- final long final long mUpperBoundary
- final long final long final long mDepth

6.5.1.1.1 Constructor & Destructor Documentation

6.5.1.1.1.6 com.qualcomm.dashplayer.api.QCDashPlayerRepositionRangeInfo.QCDashPlayer- RepositionRangeInfo (long *min*, long *max*, long *depth*)

6.5.1.1.1.6 com.qualcomm.dashplayer.api.QCDashPlayerRepositionRangeInfo.QCDashPlayer- RepositionRangeInfo ( )

Constructor of QCDashPlayerRepositionRangeInfo with no argument.

6.5.1.1.2 Member Function Documentation

6.5.1.1.2.6 final long final long final long long com.qualcomm.dashplayer.api.QCDashPlayerReposition- RangeInfo.getLowerBoundary ( )

Gets the lower boundary in MS.

Returns

Lower boundary of the reposition range.

6.5.1.1.2. 6 long com.qualcomm.dashplayer.api.QCDashPlayerRepositionRangeInfo.getUpperBoundary ( )

Gets the upper boundary in MS.

Returns

Upper boundary of the reposition range.

6.5.1.1.2. 6 long com.qualcomm.dashplayer.api.QCDashPlayerRepositionRangeInfo.getDepth ( )

Gets the depth in MS.

Returns

Depth of the reposition range (i.e., upper boundary - lower boundary).

6.5.1.1.3 Member Data Documentation

6.5.1.1.3. 6 final long com.qualcomm.dashplayer.api.QCDashPlayerRepositionRangeInfo.mLower-Boundary [package]

Constructor of QCDashPlayerRepositionRangeInfo with minimum, maximum and depth arguments.

Represents lower boundary of the reposition range.

6.5.1.1.3. 6 final long final long com.qualcomm.dashplayer.api.QCDashPlayerRepositionRangeInfo.m- UpperBoundary [package]

Represents upper boundary of the reposition range.

6.5.1.1.3. 6 final long final long final long com.qualcomm.dashplayer.api.QCDashPlayerReposition- RangeInfo.mDepth [package]

Represents depth of the reposition range.

6.6 QCDashPlayerState

6.6.1 Class Documentation

6.6.1.1 enum com::qualcomm::dashplayer::api::QCDashPlayerState

Public Member Functions

- int getCode ()

Static Public Member Functions

- [static initializer]
- static QCDashPlayerState get (int code)

Public Attributes

- STATE_IDLE = (1)
- STATE_BUFFERING = (2)
- STATE_READY = (3)
- STATE_ENDED = (4)

6.6.1.1.1 Member Function Documentation

6.6.1.1.1. 6 com.qualcomm.dashplayer.api.QCDashPlayerState.[static initializer] ( )

Static initializer of QCDashPlayerState class when it gets loaded in memory.

**6.6.1.1.1. 6 int com.qualcomm.dashplayer.api.QCDashPlayerState.getCode ( ) **

Get QCDashPlayerState as an integer type.

6.6.1.1.1. 6 static QCDashPlayerState com.qualcomm.dashplayer.api.QCDashPlayerState.get (int *code*)

Gets QCDashPlayerState as an enumeration based on the integer value passed.

6.6.1.1.2 Member Data Documentation

6.6.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerState.STATE_IDLE = (1)

Player is neither prepared nor being prepared.

6.6.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerState.STATE_BUFFERING=(2)

The player is prepared, but unable to play immediately from the current position.

This state typically occurs when more data must be buffered for playback to start.

6.6.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerState.STATE_READY=(3)

The player is prepared and able to play immediately from the current position.

The player will play if QCDashPlayer#start() was called, and pause if QCDashPlayer#pause() was called.

6.6.1.1.2. 6 com.qualcomm.dashplayer.api.QCDashPlayerState.STATE_ENDED=(4)

Player has finished playing media.

6.7 QCDashPlayerSubtitleLayout

6.7.1 Class Documentation

6.7.1.1 class com::qualcomm::dashplayer::api::QCDashPlayerSubtitleLayout

A view for rendering rich-formatted captions.

QCDashPlayerSubtitleLayout is a subclass of android.view.View.

Public Member Functions

- QCDashPlayerSubtitleLayout (android.content.Context context)

6.7.1.1.1 Constructor & Destructor Documentation

6.7.1.1.1.1 com.qualcomm.dashplayer.api.QCDashPlayerSubtitleLayout.QCDashPlayerSubtitleLayout (android.content.Context *context*)

Creates QCDashPlayerSubtitleLayout constructor with specified context.

A References

A.1 Acronyms and Terms

Acronym or term	Definition
DASH	Dynamic Adaptive Streaming over HTTP
HTTP	Hypertext Transfer Protocol
MMF	Multimedia Framework
MPD	Media Presentation Description
SCTE	Society of Cable and Telecommunications Engineers
TSB	Time Shift Buffer