

Incorporating HDR video with Dolby Vision into your apps

The new iPhone 12 models are capable of recording HDR video with Dolby Vision, so now your apps can capture, playback, edit, and export HDR video with Dolby Vision. Learn how to update your apps to take full advantage of this enhancement. To get started, iOS 14.1 or later is required.

Previewing and Editing HDR Video

Dolby Vision adds dynamic metadata to video, capturing important information, such as the dynamic range of each video frame or scene, to capture the intended look of the video. This metadata is essential in guiding an intelligent playback engine that is able to take full advantage of the final display, and render accurate color and contrast for a consistently stunning experience. If your app has Edit and Preview workflows, you may need to be aware of certain considerations.

The HDR video recorded with Dolby Vision is in Dolby Vision Profile 8, Cross-compatibility ID 4 (HLG) format. Learn more about <u>Dolby Vision profiles</u> and <u>levels</u>. It is designed to be backwards compatible with HLG, as it allows existing HEVC decoders to decode as HLG. The codec type is HEVC (10-bit). Dolby Vision 8.4-capable decoders decode as Dolby Vision and there is additional per-frame metadata to enhance the viewing experience.

Preview During Edit

For preview scenarios during edit, your app may be modifying the pixels, which invalidates the existing dynamic metadata and its usage. The Dolby Vision 8.4 metadata is completely transparent to you, hence you can drop the invalid metadata during preview-only scenarios using our APIs. Our APIs also allow you to update the dynamic metadata at export time to reflect any changes in the content.

To turn off the application of the Dolby Vision metadata during playback, set the appliesPerFrameHDRDisplayMetadata property to NO (Objective C) or false (Swift). At a lower level, if your application is using a VTDecompressionSession, the property kVTDecompressionPropertyKey_PropagatePerFrameHDRDisplayMetadata can be used to turn off the Dolby Vision tone mapping.

```
// Read/write, CFBoolean, Optional, default is kCFBooleanTrue
public let kVTDecompressionPropertyKey_PropagatePerFrameHDRDisplayMet
adata: CFString

AVPlayerItem.h
// Default is YES
open var appliesPerFrameHDRDisplayMetadata: Bool
```

Video Composition

If you use AVVideoComposition to edit Dolby Vision 8.4 video, by default (i.e., leave color properties in AVVideoComposition blank), the video composition will be in HLG.

When video frames reach the compositor, Dolby Vision 8.4 per frame metadata is discarded by the framework because the video editing can modify the pixels that invalidate the original metadata.

If you use a custom compositor, make sure your custom compositor supports HDR and indicates the HDR capability to the framework via the supportsHDRSourceFrames property. Your custom compositor needs to operate in 10-bit pixel formats and announce them in sourcePixelBufferAttributes and requiredPixelBufferAttributesForRenderContext properties.

```
AVVideoCompositing.h

optional var supportsHDRSourceFrames: Bool { get }
Summary
```

- Compositor supports HDR video.
- For preview scenarios during edit, make sure to look at options to discard the invalid Dolby Vision metadata if your workflow modifies pixels.

Exporting HDR Video

The HDR video recorded with Dolby Vision is in Dolby Vision Profile 8, Cross-compatibility ID 4 (HLG) format. Learn more about <u>Dolby Vision profiles</u> and <u>levels</u>. It is designed to be backwards compatible with HLG, as it allows existing HEVC decoders to decode as HLG. The codec type is HEVC (10-bit). Dolby Vision 8.4-capable decoders decode as Dolby Vision and there is additional per-frame metadata to enhance the viewing experience.

AVAssetExportSession

All HEVC presets have been upgraded to support HDR. The output format will match the source format, so if the source is Dolby Vision 8.4, the exported movie will also be Dolby Vision 8.4. If you need to change the output format, use AVAssetWriter.

H.264 presets will convert HDR to SDR.

AVAssetWriter

In order to preserve Dolby Vision 8.4 when exporting via AVAssetWriter, you must choose an output format and color properties that support Dolby Vision. The profile level also needs to be 10-bit. Additionally, you must specify the key kVTCompressionPropertyKey_HDRMetadataInsertionMode and value kVTHDRMetadataInsertionMode_Auto pairs for AVVideoCompressionPropertiesKey in your video output settings for exporting the Dolby Vision format.

```
VTCompressionProperties.h
// Read/write, CFString, Optional, default is
// kVTHDRMetadataInsertionMode_Auto
public let kVTCompressionPropertyKey_HDRMetadataInsertionMode:
CFString
let videoOutputSettings: [String: Any]
  = [AVVideoCodecKey : AVVideoCodecType.hevc,
      AVVideoProfileLevelKey:
            kVTProfileLevel_HEVC_Main10_AutoLevel,
      AVVideoColorPropertiesKey: [
           AVVideoColorPrimariesKey:
                 AVVideoColorPrimaries_ITU_R_2020,
           AVVideoTransferFunctionKey:
                 AVVideoTransferFunction ITU R 2100 HLG,
           AVVideoYCbCrMatrixKey:
                 AVVideoYCbCrMatrix_ITU_R_2020],
      AVVideoCompressionPropertiesKey: [
            kVTCompressionPropertyKey_HDRMetadataInsertionMode:
                 kVTHDRMetadataInsertionMode Auto]
1
```

Also, you should ensure that the pixel buffers presented to AVAssetWriterInput are a 10-bit format, such as x420.

Some applications will use a separate AVAssetReader/AVAssetWriter model for export. kVTCompressionPropertyKey_PreserveDynamicHDRMetadata can be set to kCFBooleanFalse and the AVAssetWriter will recompute the Dolby Vision 8.4 metadata when creating the new file. This should be done when the application modifies the output frames from the AVAssetReader.

```
// CFBoolean, Write, Optional
```

public let kVTCompressionPropertyKey_PreserveDynamicHDRMetadata: CFString

Summary

- Export workflows support HDR video with Dolby Vision.
- APIs now allow you to generate exports in both HDR with Dolby Vision 8.4 format and SDR format.
- Apple platforms handle the advanced Dolby Vision dynamic metadata for you, so metadata can be updated at export.

Playing Back HDR Video

iPhone 12 HDR video captured with Dolby Vision supports high dynamic range, which provides amazing contrast and colors that make each detail pop with remarkable vividness. This allows better preservation of detail in both bright and dark areas of an image, without compromise. Dolby Vision also adds dynamic metadata to the video, capturing important information, such as the dynamic range of each video frame or scene, to capture the intended look of the video. This metadata is essential in guiding an intelligent playback engine that is able to take full advantage of the final display, and render accurate color and contrast for a consistently stunning experience.

The HDR Video recorded with Dolby Vision is in Dolby Vision Profile 8, Cross-compatibility ID 4 (HLG) format. Learn more about <u>Dolby Vision profiles</u> and <u>levels</u>. It is designed to be backwards compatible with HLG, as it allows existing HEVC decoders to decode as HLG. The codec type is HEVC (10-bit). Dolby Vision 8.4-capable decoders decode as Dolby Vision and there is additional per-frame metadata to enhance the viewing experience. The video recorded by the Camera app is in the QuickTime File Format (QTFF) movie (.mov extension). Signaling for Dolby Vision in a QTFF movie is similar to signaling in Dolby Vision Streams <u>within the ISO base media file format</u>.

Asset Inspection

A new media characteristic, AVMediaCharacteristicContainsHDRVideo, identifies whether any segment in a track contains HDR.

AVMediaFormat.h

AVMediaCharacteristic const AVMediaCharacteristicContainsHDRVideo;

```
Example usages:
public static let containsHDRVideo: AVMediaCharacteristic
// After loading "tracks" property via
// loadValuesAsynchronously(forKeys:completionHandler:)
let hdrTracks =
asset.tracks(withMediaCharacteristic: .containsHDRVideo)
if track.hasMediaCharacteristic(.containsHDRVideo)
```

Playback

Dolby Vision 8.4 playback is supported via AVPlayer+AVPlayerLayer and AVSampleBufferDisplayLayer. You can use the static property eligibleForHDRPlayback on AVPlayer to check if the device supports HDR playback.

If AVPlayer is used to play back Dolby Vision 8.4 video, nothing extra is needed from the client code. AVFoundation framework will set up an HDR playback pipeline and handle Dolby Vision 8.4 rendering automatically, if the asset is Dolby Vision and the device supports HDR playback.

Some apps may use AVSampleBufferDisplayLayer to render video. To play back Dolby Vision 8.4 assets, make sure the sample buffers passed to the sample buffer display layer are in formats suitable for HDR and carry Dolby Vision 8.4 per-frame metadata. The sample buffers need to have 10-bit or higher bit-depth. A commonly used 10-bit pixel format is kCVPixelFormatType_420YpCbCr10BiPlanarVideoRange (x420). If your sample buffers are decoded with a VTDecompressionSession, leave kVTDecompressionPropertyKey_PropagatePerFrameHDRDisplayMetadata to the default (true) to carry the Dolby Vision 8.4 per-frame metadata in the buffers.

Summary

- iPhone 12 captures HDR video with Dolby Vision video, supporting high dynamic range, and providing amazing contrast and colors that make each detail pop with remarkable vividness.
- Apple platforms handle the advanced Dolby Vision dynamic metadata for you.
- Review your playback pipeline to take advantage of HDR video playback with AVPlayer+AVPlayerLayer and AVSampleBufferDisplayLayer.

Converting HDR Video to SDR Video

We encourage you to adopt HDR workflows in your apps so that the content is presented in HDR with Dolby Vision. However, if your app is not ready for HDR video workflows, for compatibility and as a "stop gap" solution, one option is to convert HDR video to SDR video. This conversion via the AVFoundation API will ensure the best possible representation of the original colors.

How to Convert HDR Video to SDR Video

Down-converting 10-bit HDR to 8-bit SDR involves two independent transformations: HDR to SDR color space conversion and 10-bit to 8-bit bit-depth conversion. You need to ensure that you perform the color conversion before or, optimally, at the same time as the 10-bit to 8-bit pixel format conversion. If 10-bit to 8-bit pixel format conversion occurs before the color conversion, then undesirable banding artifacts will be present in the output video.

For the color space conversion, the only thing you need to do is explicitly specify an SDR color space and an 8-bit codec and profile level. A general rule of thumb is to set the color properties to BT.709. Specifically, if you include video composition in your playback or editing, you would set the following SDR color properties in your AVVideoComposition.

```
videoComposition.colorPrimaries = AVVideoColorPrimaries_ITU_R_709_2
videoComposition.colorTransferFunction =
AVVideoTransferFunction_ITU_R_709_2 videoComposition.colorYCbCrMatrix
= AVVideoYCbCrMatrix_ITU_R_709_2
```

If your workflow includes AVAssetReader and AVAssetReaderOutput, or AVAssetWriter and AVAssetWriterInput, make sure you specify SDR color properties in the corresponding outputSettings. For example:

If you are using VTDecompressionSession directly, make sure you set the kVTPixelTransferPropertyKey_PixelTransferProperties dictionary with SDR color properties.

If your workflow involves the use of AVPlayerItemVideoOutput, you can use initWithPixelBufferAttributes with SDR color properties.

let videoOutput = AVPlayerItemVideoOutput(pixelBufferAttributes:
attributes as [String: Any])

Summary

Tone mapping is handled automatically through the AVFoundation conversion methods. Converting HDR to SDR through other methods (e.g., a server-side conversion) requires special considerations for handling the tone mapping to ensure SDR content looks similar to what was intended in the original HDR capture. Therefore:

- If your app is not ready for HDR workflows yet, convert HDR video to SDR video.
- The HDR to SDR conversion via the AVFoundation API will ensure the best possible representation of the original colors.

Learn More

- Export HDR media in your app with AVFoundation (WWDC20 session)
- Edit and play back HDR video with AVFoundation (WWDC20 session)
- Dolby Vision profiles and levels
- Dolby Vision within ISO base media file format