Class

AVCapturePhotoOutput

A capture output for still image, Live Photos, and other photography workflows.

iOS 10.0+ | iPadOS 10.0+ | Mac Catalyst 14.0+ | macOS 10.15+ | tvOS 17.0+

```
class AVCapturePhotoOutput
```

Mentioned in

- Configuring camera capture to collect a Portrait Effects matte
- Setting up a capture session
- Capturing and saving Live Photos
- Capturing thumbnail and preview images
- Capturing a bracketed photo sequence

Overview

`AVCapturePhotoOutput` provides an interface for capture workflows related to still photography. In addition to basic capture of still images, a photo output supports RAW-format capture, bracketed capture of multiple images, Live Photos, and wide-gamut color. You can output captured photos in a variety of formats and codecs, including RAW format DNG files, HEVC format HEIF files, and JPEG files.

To capture photos with the `AVCapturePhotoOutput` class, follow these steps:

1. Create an `AVCapturePhotoOutput` object. Use its properties to determine supported capture settings and to enable certain features (for example, whether to capture Live Photos).

2. Create and configure an AVCapturePhotoSettings object to choose features and settings for a specific capture (for example, whether to enable image stabilization or flash).

3. Capture an image by passing your photo settings object to the capturePhoto(with:delegate:) method along with a delegate object implementing the AVCapturePhotoCaptureDelegate protocol. The photo capture output then calls your delegate to notify you of significant events during the capture process.

Some photo capture settings, such as the flashMode property, include options for automatic behavior. For such settings, the photo output determines whether to use that feature at the moment of capture—you don't know when requesting a capture whether the feature will be enabled when the capture completes. When the photo capture output calls your AVCapturePhotoCaptureDelegate methods with information about the completed or in-progress capture, it also provides an AVCaptureResolvedPhotoSettings object that details which automatic features are set for that capture. The resolved settings object's uniqueID property matches the uniqueID value of the AVCapturePhotoSettings object you used to request capture.

Enabling certain photo features (Live Photo capture and high resolution capture) requires a reconfiguration of the capture render pipeline. To opt into these features, set the isHighResolutionCaptureEnabled, isLivePhotoCaptureEnabled, and isLivePhotoAutoTrimmingEnabled properties before calling your AVCaptureSession object's startRunning() method. Changing any of these properties while the session is running disrupts the capture render pipeline: Live Photo captures in progress end immediately, unfulfilled photo requests abort, and video preview temporarily freezes.

Using a photo capture output adds other requirements to your AVCaptureSession object:

- A capture session can't support both Live Photo capture and movie file output. If your capture session includes an AVCaptureMovieFileOutput object, the isLivePhotoCaptureSupported property becomes false. (As an alternative, you can use the AVCaptureVideoDataOutput class to output video buffers at the same resolution as a simultaneous Live Photo capture).

- A capture session can't contain both an AVCapturePhotoOutput object and an AVCaptureStillImageOutput object. The AVCapturePhotoOutput class includes all functionality of (and deprecates) the AVCaptureStillImageOutput class.

The AVCapturePhotoOutput class implicitly supports wide-gamut color photography. If the source AVCaptureDevice object's activeColorSpace value is AVCaptureColorSpace.P3_D65, the capture output produces photos with wide color information (unless your AVCapturePhotoSettings object specifies an output format that doesn't support wide color).

Topics

Creating a photo output

`init()`

Creates a new photo capture output object.

Capturing a photo

`func capturePhoto(with: AVCapturePhotoSettings, delegate: any AVCapturePhotoCaptureDelegate)`

Initiates a photo capture using the specified settings.

Managing responsive capture

`var captureReadiness: AVCapturePhotoOutput.CaptureReadiness`

A value that specifies whether the photo output is ready to respond to new capture requests in a timely manner.

`enum CaptureReadiness`

Constants that indicate whether the output is ready to receive capture requests.

`var isAutoDeferredPhotoDeliveryEnabled: Bool`

A Boolean value that indicates the enabled state of automatic deferred photo delivery.

`var isAutoDeferredPhotoDeliverySupported: Bool`

A Boolean value that indicates whether the photo output supports deferred photo delivery.

`var isFastCapturePrioritizationSupported: Bool`

A Boolean value that indicates whether the photo output supports fast capture prioritization.

`var isFastCapturePrioritizationEnabled: Bool`

A Boolean value that indicates whether the output enables fast capture prioritization.

`var isResponsiveCaptureSupported: Bool`

A Boolean value that indicates whether the photo output supports responsive capture.

`var isResponsiveCaptureEnabled: Bool`

A Boolean value that indicates whether the photo output configuration enables responsive capture.

`var isZeroShutterLagSupported: Bool`

A Boolean value that indicates whether the photo output supports zero shutter lag.

`var isZeroShutterLagEnabled: Bool`

A Boolean value that indicates whether the photo output configuration enables zero shutter lag.

Determining supported pixel formats

`var availablePhotoPixelFormatTypes: [OSType]`

The pixel formats the capture output supports for photo capture.

`var availableRawPhotoPixelFormatTypes: [OSType]`

The pixel formats the capture output supports for RAW photo capture.

`func supportedPhotoPixelFormatTypes(for: AVFileType) -> [OSType]`

Returns the list of uncompressed pixel formats supported for photo data in the specified file type.

`func supportedRawPhotoPixelFormatTypes(for: AVFileType) -> [OSType]`

Returns the list of Bayer RAW pixel formats supported for photo data in the specified file type.

`class func isAppleProRAWPixelFormat(OSType) -> Bool`

Returns a Boolean value that indicates whether the pixel format is an Apple ProRAW format.

`class func isBayerRAWPixelFormat(OSType) -> Bool`

Returns a Boolean value that indicates whether the pixel format is a Bayer RAW format.

Determining supported codec types

`var availablePhotoCodecTypes: [AVVideoCodecType]`

The compression codecs this capture output currently supports for photo capture.

`func supportedPhotoCodecTypes(for: AVFileType) -> [AVVideoCodecType]`

Returns the list of photo codecs (such as JPEG or HEVC) supported for photo data in the specified file type.

Determining supported file types

`var availablePhotoFileTypes: [AVFileType]`

The list of file types currently supported for photo capture and output.

`var availableRawPhotoFileTypes: [AVFileType]`

The list of file types currently supported for RAW format capture and output.

Suppressing the shutter sound

`var isShutterSoundSuppressionSupported: Bool`

A Boolean value that indicates whether the photo output supports suppressing the system shutter sound.

Configuring ProRAW support

`var isAppleProRAWSupported: Bool`

A Boolean value that indicates whether the current device and configuration supports Apple ProRAW pixel formats.

`var isAppleProRAWEnabled: Bool`

A Boolean value that indicates whether you've configured the photo output to deliver Apple ProRAW formats.

Determining available settings

`var isContentAwareDistortionCorrectionSupported: Bool`

A Boolean value that indicates whether the session's current configuration supports content-aware distortion correction.

`var isContentAwareDistortionCorrectionEnabled: Bool`

A Boolean value that indicates whether the photo render pipeline can perform content-aware distortion correction.

`var isLensStabilizationDuringBracketedCaptureSupported: Bool`

A Boolean value indicating whether the capture output currently supports lens stabilization during bracketed image capture.

`var maxBracketedCapturePhotoCount: Int`

The maximum number of images that the photo capture output can support in a single bracketed capture.

`var supportedFlashModes: [AVCaptureDevice.FlashMode]`

A Swift array of flash settings this capture output currently supports.

`var isAutoRedEyeReductionSupported: Bool`

A Boolean value indicating whether the capture output supports automatic red-eye reduction.

Monitoring the visible scene

`var isFlashScene: Bool`

A Boolean value indicating whether the scene currently being previewed by the camera warrants use of the flash.

`var photoSettingsForSceneMonitoring: AVCapturePhotoSettings?`

A photo settings object that controls how the photo output detects and handles automatic flash and stabilization modes.

Configuring high-resolution still capture

`var maxPhotoDimensions: CMVideoDimensions`

The maximum resolution of the requested photo.

Configuring Live Photo capture

`var isLivePhotoCaptureSupported: Bool`

A Boolean value that indicates whether the capture output currently supports Live Photo capture.

`var isLivePhotoCaptureEnabled: Bool`

A Boolean value that indicates whether to configure the capture pipeline for Live Photo capture.

`var isLivePhotoCaptureSuspended: Bool`

A Boolean value that indicates whether Live Photo capture is currently in a suspended state.

`var preservesLivePhotoCaptureSuspendedOnSessionStop: Bool`

A Boolean value that indicates whether to preserve the suspended state of Live Photo capture when the session stops.

`var isLivePhotoAutoTrimmingEnabled: Bool`

A Boolean value that indicates whether to automatically trim Live Photo movie captures to avoid excessive movement.

`var availableLivePhotoVideoCodecTypes: [AVVideoCodecType]`

An array of video codecs currently available for Live Photo movie captures.

Configuring depth data capture

`var isDepthDataDeliverySupported: Bool`

A Boolean value indicating whether the capture output currently supports depth data capture.

`var isDepthDataDeliveryEnabled: Bool`

A Boolean value that specifies whether to configure the capture pipeline for depth data capture.

Configuring Portrait Effects matte capture

`var isPortraitEffectsMatteDeliveryEnabled: Bool`

A Boolean value indicating whether the capture output generates a portrait effects matte.

`var isPortraitEffectsMatteDeliverySupported: Bool`

A Boolean value indicating whether the capture output currently supports delivery of a portrait effects matte.

`var portraitEffectsMatte: AVPortraitEffectsMatte?`

The portrait effects matte captured with the photo.

Configuring constant color

`var isConstantColorSupported: Bool`

A Boolean value that indicates whether a photo output supports constant color capture.

`var isConstantColorEnabled: Bool`

A Boolean value that indicates whether the photo output configures the render pipeline to perform constant color capture.

Configuring orientation compensation

`var isCameraSensorOrientationCompensationSupported: Bool`

`var isCameraSensorOrientationCompensationEnabled: Bool`

Configuring virtual device capture

`var isVirtualDeviceFusionSupported: Bool`

A Boolean value that indicates whether the device supports virtual device image fusion.

`var isVirtualDeviceConstituentPhotoDeliverySupported: Bool`

A Boolean value that indicates whether the photo output configuration supports delivery of photos from constituent cameras of a virtual device.

`var isVirtualDeviceConstituentPhotoDeliveryEnabled: Bool`

A Boolean value that indicates whether the photo output delivers photos from constituent cameras of a virtual device.

Preparing for resource-intensive captures

`var preparedPhotoSettingsArray: [AVCapturePhotoSettings]`

An array of photo settings for which the photo output has prepared capture resources.

`func setPreparedPhotoSettingsArray([AVCapturePhotoSettings], completionHandler: ((Bool, (any Error)?) -> Void)?)`

Tells the photo capture output to prepare resources for future capture requests with the specified settings.

Getting segmentation mattes

`var availableSemanticSegmentationMatteTypes: [AVSemanticSegmentationMatte.MatteType]`

An array of semantic segmentation matte types that may be captured and delivered along with the primary photo.

`var enabledSemanticSegmentationMatteTypes: [AVSemanticSegmentationMatte.MatteType]`

The semantic segmentation matte types that the photo render pipeline delivers.

Setting the capture prioritization

`var maxPhotoQualityPrioritization: AVCapturePhotoOutput.QualityPrioritization`

The highest quality the photo output should prepare to deliver on a capture-by-capture basis.

`enum QualityPrioritization`

Constants that indicate how to prioritize photo quality relative to capture speed.

Determining calibration data delivery support

`var isCameraCalibrationDataDeliverySupported: Bool`

A Boolean value indicating whether the capture output currently supports delivery of camera calibration data.

Deprecated

Deprecated symbols

Review unsupported symbols and their replacements.

Instance properties

`var availableRawPhotoCodecTypes: [AVVideoCodecType]`

Instance methods

`func supportedRawPhotoCodecTypes(forRawPhotoPixelFormatType: OSType, fileType: AVFileType) -> [AVVideoCodecType]`

Relationships

Inherits From

`AVCaptureOutput`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`
`CustomStringConvertible`
`Equatable`
`Hashable`
`NSObjectProtocol`

See Also

Photo capture

{} Capturing consistent color images

Add the power of a photography studio and lighting rig to your app with the new Constant Color API.

Capturing still and Live Photos

Configure and capture single or multiple still images, Live Photos, and other forms of photography.

Capturing photos in RAW and Apple ProRAW formats

Support professional photography workflows by enabling minimally processed image capture in your camera app.

Supporting Continuity Camera in Your Mac App

Incorporate scanned documents and pictures from a user's iPhone, iPad, or iPod touch into your Mac app using Continuity Camera.

`class AVCapturePhoto`

A container for image data from a photo capture output.

`class AVCaptureDeferredPhotoProxy`

A lightly-processed photo with data that the system may use to process and fetch a higher-resolution asset at a later time.

`protocol AVCapturePhotoCaptureDelegate`

Methods for monitoring progress and receiving results from a photo capture output.

`class AVCapturePhotoOutputReadinessCoordinator`

An object that monitors changes to a photo output's capture readiness.

`protocol AVCapturePhotoOutputReadinessCoordinatorDelegate`

A delegate protocol to receive updates about a photo output's capture readiness.

~~`class AVCaptureStillImageOutput`~~

A capture output for capturing still photos.

Deprecated