

## Article

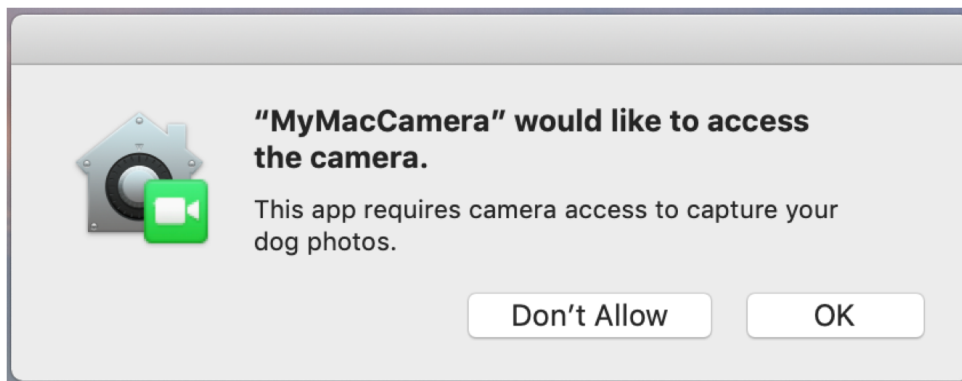
# Requesting Authorization for Media Capture on macOS

Prompt the user to authorize access to the camera and microphone.

[Framework](#)[Bundle Resources](#)[On This Page](#)[Overview](#) [See Also](#) 

## Overview


In macOS 10.14 and later, the user must explicitly grant permission for each app to access cameras and microphones. Before your app can use the capture system for the first time, macOS shows an alert asking the user to grant your app access to the camera, as shown below. macOS remembers the user's response to this alert, so subsequent uses of the capture system don't cause it to appear again. The user can change permission settings for your app in System Preferences > Security & Privacy. The request for authorization looks different from the alert UI in iOS.



To make sure your app has permission before capturing media, follow the steps below.

## Configure Your Camera and Microphone Apps

The information property list keys for Camera and Microphone in macOS operate the same way as they do in iOS. macOS 10.14 and later populates the static messages with these strings when

Documentation AVFoundation Cameras and Media Capture Requesting Authorization for Media Capture on macOS Language: Swift   
API  
Changes: None

- If your app uses device microphones, include the `NSMicrophoneUsageDescription` key in your app's `Info.plist` file.

For each key, provide a message that explains to the user why your app needs to capture media, so that the user can feel confident granting permission to your app.

Key	Type	Value
Information Property List		
	Dictionary	(17 items)
Privacy - Camera Usage Description	String	This app requires camera access to capture your dog photos.
Privacy - Microphone Usage Description	String	This app requires microphone access to record your greeting.
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
Localization native development region	String	\$(DEVELOPMENT_LANGUAGE)
Executable file	String	\$(EXECUTABLE_NAME)

### Important

If the appropriate key is not present in your app's `Info.plist` file when your app requests authorization or attempts to use a capture device, the system terminates your app. The Xcode debug console displays a message that explains the reason for the crash.

## Verify and Request Authorization for Capture

Always test the `AVCaptureDevice.authorizationStatus(for:)` method before setting up a capture session. If the user has not yet granted or denied capture permission, the authorization status is `AVAuthorizationStatus.notDetermined`. In this case, use the `requestAccess(for:completionHandler:)` method to have macOS prompt the user:

```
switch AVCaptureDevice.authorizationStatus(for: .video) {
    case .authorized: // The user has previously granted access to the camera.
        self.setupCaptureSession()

    case .notDetermined: // The user has not yet been asked for camera access.
        AVCaptureDevice.requestAccess(for: .video) { granted in
            if granted {
                self.setupCaptureSession()
            }
        }

    case .denied: // The user has previously denied access.
        return

    case .restricted: // The user can't grant access due to restrictions.
        return
}
```

The `requestAccess(for:completionHandler:)` method is asynchronous: Your app continues running while macOS shows the permission alert. When the user responds, the system calls your completion handler. If the completion handler's success parameter is `true`, you can proceed to set up and start a capture session.

### Note

Call `requestAccess(for:completionHandler:)` before starting capture, but only at a time that's appropriate for your app. For example, if photo or video recording isn't the main focus of your app, check for camera permission only when the user invokes your app's camera-related features.

## Request Authorization Before Saving Captured Media

After capturing photos or video, you may want to save them into the user's Photos library. Accessing the Photos library also requires user permission (separate from camera and microphone permission). For most photo and video capture workflows (including Live Photos and RAW format capture), use the `PHPhotoLibrary` and `PHAssetCreationRequest` classes. These classes require read/write access to the Photos library, so you must use the `NSPhotoLibraryUsageDescription` key in your information property list to provide a message to the user when asking for access. For details, see [Saving Captured Photos](#).

## Reset Authorization from Terminal

To reset the state of the user's decision to grant or reject Microphone access so the prompt shows again, open Terminal and input the command:

```
tccutil reset Microphone
```

To reset the authorization state for camera access, type:

```
tccutil reset Camera
```

This command resets the access authorization settings for all apps, so other apps will prompt the user again. Use this tool to debug the appearance of your privacy justification strings and their localized versions.


### Tip

You can use `tccutil` to reset authorization access settings for other system services as well, such as AddressBook, Calendar, and Finder.

## See Also

---

### User Privacy

 [Requesting Authorization for Media Capture on iOS](#)

Respect user privacy by seeking permission to capture and store photos, audio, and video.