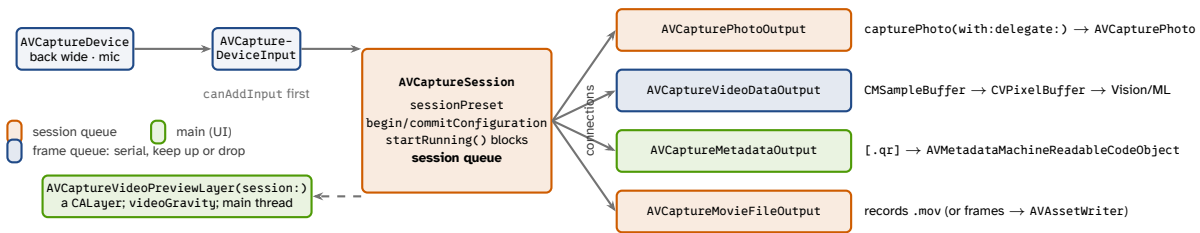


In one line: AVCaptureSession is a **graph**: devices → inputs → session → outputs, configured and started on a **private serial queue** (startRunning() blocks), frames delivered on **another** serial queue, UI on main. AVAudioSession is **policy, not playback**: category + mode + options tell iOS whether you mix, duck, obey the silent switch, record, and keep playing in the background.



How it works — capture

- Permission:** NSCameraUsageDescription (+ ...Microphone...) — missing key = **crash** on access. AVCaptureDevice.authorizationStatus(for:) → requestAccess(for:) (completion off main; async overload); .denied/.restricted → Settings. Mic alone: AVAudioApplication.requestRecordPermission (iOS 17).
- Configure** between beginConfiguration() and commitConfiguration() (atomic camera/preset swap); canAdd... before every add... Focus/exposure/torch: lockForConfiguration() → set → unlock (check is...Supported).
- Frames:** captureOutput(_:didOutput:from:) at frame rate; alwaysDiscardsLateVideoFrames (default true) drops rather than queues — drops arrive in didDrop.
- Photo:** a **new** AVCapturePhotoSettings per shot; photoQualityPrioritization .speed/.balanced/.quality. Saving is the **Photos** framework's job (PHPPhotoLibrary, add-only usage key).
- Rotation** (iOS 17): videoRotationAngle + AVCaptureDevice.RotationCoordinator replace videoOrientation.
- Interruptions:** AVCaptureSessionWasInterrupted/...InterruptionEnded; reasons e.g. videoDeviceNotAvailableInBackground, ...WithMultipleForegroundApps (iPad multitasking), ...InUseByAnotherClient. Runtime error → restart on the session queue. Hot device (ProcessInfo.thermalState) → lower preset/fps.

Example — configure off main

```
func configure() { // runs on sessionQueue
    session.beginConfiguration()
    defer { session.commitConfiguration() }
    guard let cam = AVCaptureDevice.default(.builtinWideAngleCamera,
        for: .video, position: .back),
        let input = try? AVCaptureDeviceInput(device: cam),
        session.canAddInput(input) else { return }
    session.addInput(input)
    let frames = AVCaptureVideoDataOutput()
    frames.setSampleBufferDelegate(self, queue: videoQueue) // serial
    if session.canAddOutput(frames) { session.addOutput(frames) }
    let meta = AVCaptureMetadataOutput(); session.addOutput(meta)
    meta.setMetadataObjectsDelegate(self, queue: .main)
    meta.metadataObjectTypes = [.qr] // AFTER addOutput
}
sessionQueue.async { configure(); session.startRunning() }
```

Audio session — the category sets defaults

Category	silent sw.	mixes	BG *	for
.ambient	silenced	yes	no	game/UI sound
.soloAmbient	silenced	no	no	default
.playback	plays	no [†]	yes	media, podcasts
.record	—	no	yes	recorder
.playAndRecord	plays	no [†]	yes	VoIP, video call

* also needs UIBackgroundModes audio. [†] unless .mixWithOthers / .duckOthers.

.ambient/.soloAmbient also stop on screen lock.

- setCategory(_:mode:options:) → setActive(true) (fails during a call); end with .notifyOthersOnDeactivation.
- Modes** tune DSP/routing: .moviePlayback, .spokenAudio, .voiceChat (echo cancel), .videoChat, .measurement. **Options:** .duckOthers, .mixWithOthers, .defaultToSpeaker (else playAndRecord uses the *earpiece*), .allowBluetoothA2DP.
- Interruption:** .began — already stopped, update UI; .ended + .shouldResume → reactivate, resume. An .ended is not guaranteed.
- Route change** .oldDeviceUnavailable (headphones out) → **pause**. mediaServicesWereReset → rebuild all audio objects.
- Player:** AVAudioPlayer = local file, simple · AVPlayer = streams/HLS, video · AVAudioEngine = real-time node graph (mixer, effects, installTap); restart it on AVAudioEngineConfigurationChange.
- Lock screen = MPNowPlayingInfoCenter + MPRemoteCommand Center; a .mixWithOthers app can't be Now Playing.

Interview traps

- ! startRunning() on main = UI freeze; config from two threads = races. One serial session queue owns the session.
- ! metadataObjectTypes before addOutput (or an unsupported type) **raises an exception** — not a silent no-op.
- ! Keeping CMSampleBuffers past the callback starves the capture pool → dropped frames. Copy what you need.
- ! “No sound on silent” = default .soloAmbient; use .playback. AVAudioPlayer in a local let is deallocated and stops.

Remember

Three queues, one graph · category + mode + options = policy
· pause on **began** and on **headphones out**.

Likely questions

- QR scanner? — metadata output, [.qr] after addOutput (or VisionKit DataScannerViewController, iOS 16).
- Nav voice over Spotify? — .playback + .duckOthers, then .notifyOthersOnDeactivation.