Apple VisionableSDK v1.0 Release Notes (June 24, 2024)

NOTE:
This is the first major release of the Visionable SDK. It contains three main modules:

- MeetingSDK
- ModeratorSDK
- VisionableAPI

These are all represented as separate singletons bundled in an Apple XCFramework file named MeetingSDK.xcframework and which has slices for both macOS and iOS. All APIs are documented in our readme.io-based documentation repository. The release notes below will cover changes made to the original MeetingSDK singleton; please consult the readme.io repository for information on new singletons (ModeratorSDK, VisionableAPI)

API CHANGES

The initializeMeeting and initializeMeetingWithToken APIs have been moved to the VisionableAPI singleton.

```
public func enableVideoPreview(camera: String, withMode: String,
andBlurring: Bool = false, lowLevel: Bool, completion: @escaping
(Bool) -> ())
```

Add parameter to enable background blurring for previews. If omitted, blurring will be disabled by default.

CHANGES/FIXES

Any functionality associated with calling Visionable's web service APIs has been moved into the new ModeratorSDK singleton.

When connecting to a V3 or later server, the SDK now establishes a web socket connection to Visionable's RTN service for the purposes of information updates from the Visionable back-end as well as out-of-band commands that may be sent to the client from a meeting Moderator or for the purposes of receiving PTZ commands (for supported cameras) from remote users.

KNOWN ISSUES
When screen sharing from iOS and placing the application in the background, iOS may suspend the app (and cause screen sharing to be paused) if the iOS device is running low on system resources.

When sharing a window into the meeting, the remote user may see the share freeze if the shared window is resized while being actively shared.

The new `previewVideoUpdated` delegate method may *not* be called when a device rotates. This will be resolved in a future SDK release.

In support of the new threading model, all delegate methods are executed on a serial OperationQueue that is created by the SDK. Future versions will allow you to specify an OperationQueue that you create (or use the main queue)