Android MeetingSDK v0.5.1 Release Notes. (August 19, 2022)

NOTE:
Version 0.5.1 of the Visionable Anddroid MeetingSDK is a transitional release that begins to expose functionality available in Visionable's "V3" server architecture. As this architecture is not expected to be in production until 4Q2022, any 0.5.x releases of the MeetingSDK are likely to be unstable and change frequently.

If you are looking to write an application that interfaces with Visionable's V2 architecture, you should remain with v0.4 of the MeetingSDK.

We expect that stability will be established against the V3 architecture with v0.6 of the SDK due when the V3 architecture officially goes into production.

OVERARCHING CHANGES (this text present for ALL v0.5.x releases)
Starting with v0.5, the Visionable MeetingSDK has been re-architected to rely on a layer of cross-platform C++-based code to manage the parsing of XML objects coming from our audio/video engine, for establishing model objects representing Meetings and Participants, and for firing "delegate methods/callbacks" notifying your application of changes in state for the current meeting. Prior to v0.5, these functions were individually implemented per-platform supported in that platform's native language. Moving all of this functionality into a common, C++ codebase should result in consistent behavior when dealing with Visionable back-end servers.

CONNECTING TO V3 SERVERS (this text present for ALL v0.5.x releases)
In Visionable's V3 architecture, a special token (referred to as an MJWT token) is required to join a meeting. There are two types of MJWT tokens: a *guest* MJWT token that doesn't correspond to Visionable user and an *authenticated* MJWT token that is obtained by passing a JWT token obtained from Visionable's authentication system (not covered here). To retrieve an MJWT token, use the new `initializeMeetingWithToken` API call *instead of* the original `initializeMeeting` API call (which is used only with V2 servers).

```
public static void initializeMeetingWithToken(
        String token,
        String server,
        String uuid,
        IInitializeMeetingCompleteCallback initDoneCallback
)
```

This function still takes a `uuid` and a `server` name but now also takes a `token` parameter that is either `NULL` if you wish to obtain a guest MJWT or it contains a JWT token if you want to obtain an authenticated MJWT.

The completion routine for `initializeMeetingWithToken` now is called with a second parameter (String) that contains the MJWT token (guest or authenticated).

Once you obtain an MJWT, you now join the meeting with a call to joinMeeting (same call for either V2 or V3 servers):

```
public static int joinMeeting(
        String name,
        String userUUID,
        IJoinMeetingCompleteCallback joinDoneCallback)
```

The `initializeMeetingWithToken` function caches the `server` name you are connecting to, the `uuid` for the meeting and the MJWT needed to connect to the meeting. When calling `joinMeeting`, you only need to pass the `name` of the user to be shown in the meeting as well as the optional `userUUID` to be associated with this user (pass an empty string to have the SDK generate a UUID for this user). This behavior is slightly different than the APIs in the MeetingSDK for non-Android platforms, and the Android MeetingSDK will likely be refactored in the future to behave in a manner similar to other platforms.

Using these two calls will allow you to connect to a V3 meeting. Once connected, all other SDK functionality is the same as with V2 servers.

CONNECTING TO V2 SERVERS

The APIs for connecting to V2 servers are the same as they were for the v0.4 release of the Android MeetingSDK.

See previous release notes in the v0.5.x series for API changes that were new in previous releases. The rest of these release notes pertain only to the v0.5.1 release.

API CHANGES

To retrieve a `Participant` object based on a known user's UUID, this function is now available:

```
public static Participant findParticipantByUUID(final String uuid)
```

When setting the delegate with the `MeetingSDK` singleton's `setDelegate` method, you can now provide a `Looper` that all SDK callbacks will be executed on. A new, overloaded `setDelegate` method has been provided to allow specification of this argument. The original version is still available which, if used, will cause the SDK callbacks to be executed on whatever thread the underlying C++ code that invokes them is on. Additionally, you may now pass NULL for the delegate to tell the `MeetingSDK` singleton that no delegate is available.

```
public static void setDelegate(@Nullable INotificationCallback delegate,
final Looper looper)

public static void setDelegate(@Nullable INotificationCallback delegate)
```

Routines previously available in the `VMeeting` class to retrieve participants based on audio and video stream IDs have been removed. Use new APIs introduced in v0.5 in the `MeetingSDK` singleton for these purposes.

CHANGES/FIXES

You will no longer receive multiple `participantAdded` events for the local user (with slightly different user UUIDs) when connecting to a V3 server.

Removed excessive logging statements that were coming in through the `logMessage` `INotificationCallback` method.

The `joinMeeting` API call is no longer executed on the main thread. This helps avoid user interface lockups.

Fixed issue with not being able to retrieve AudioInfo and AudioStreamId from the underlying JNI code.

Fixed issue where the `getLocalParticipant()` method was returning NULL instead of the local `Participant`.

Fixed issue where the camera name associated with the local user was the string "true".

Log messages from the underlying CoreMeeting library will now appear via the `logMessage` delegate method.

KNOWN ISSUES

Screen sharing works, but has not been heavily tested.

The MeetingSDK Reference App has exhibited some issues displaying (some) remote videos, and has occasionally crashed.

The SDK will not recognize all supported video modes for cameras.

Some applications have received multiple `participantAdded` delegate method calls for the local user with different user UUIDs.