SocialVid

Video Platform API

For Android client

V1.0

Democratizing Video Communication for Everyone to use

Table of Contents

Introduction

SocialVid Platform API is the interface for third-party Android application developers to build or include custom Video Communication client applications that seamlessly integrate into the Web infrastructure.

These API's allow video rendering using the WebRTC engine available for Android.

This document attempts to list the entire APIs that are supported and will be supported in future along with sample code snippets to use the APIs.

API Work Flow

SocialVid Platform APIs are implemented in the SocialvidClient object. The SocialvidClient exposes interfaces to make peer to peer or multiparty audio/video calls to a conference.

Developers can develop simple Android UI application and use the SocialvidClient to initiate the audio/video calls.

SocialvidClient

Create the SocialvidCient object

Create a SocialvidClient object. This opens a connection to the SocialVid server.

Contructor:
SocialvidClient(Point displaySize, RtcListener listener, EGLContext mEGLcontext)

displaySize: size of display in pixels, this is set as the video constraints for the video source
RtcListener: interface which implements the callback methods to handle events
EGLContext: context obtained from VideoRendererGui.getEGLContext() of the UI

```
Point displaySize = new Point();
getWindowManager().getDefaultDisplay().getSize(displaySize);
SocialvidClient  client = new SocialvidClient(displaySize, this,
      VideoRendererGui.getEGLContext());
```

Guest login

Login to the server with guest name and the conference ID. User should login first to join a video or an audio conference on the server. This call should be made after the *RtcListener.onConnectionReady()* callback.

SocialvidClient.guestLogin(String guestName, String confId)

In case of an error *RtcListener.handleError()* is called with the specific error number. Client will have to handle the error.

Refer to *RtcListener* class for more details.

Join a video conference

Join a Video conference with a specific guest name and valid Conference ID. In this case the device camera is accessed and the user video stream is sent to other users on the conference. This call should be made after the *RtcListener.onLoginCompleted()* callback.

SocialvidCient.joinVideoConference(String confId)

In case of an error *RtcListener.handleError()* is called with the specific error number. Client will have to handle the error.

Refer to *RtcListener* class for more details.

Join an Audio conference

Join an Audio conference with a specific guest name and valid Conference ID. In this case the device camera is not started. This call should be made after the *RtcListener.onLoginCompleted()* callback.

SocialvidCient.joinAudioConference(String confId)

In case of an error *RtcListener.handleError()* is called with the specific error number. Client will have to handle the error.

Refer to *RtcListener* class for more details.

Resume the conference

Resume the video source after it has been stopped. This has to be called is to restart the local video source. This can be called in the UI (Activity.onResume()). This is not same as unmute video option.

SocialvidClient.onResume()

Stop the conference

Stop the video stream from the UI. This stops the local video source. This can be called in the UI (Activity.onPause()). This is not same as mute video option.

SocialvidClient.onPause()

Close the connection

Close the connection and the video source at the end of the conference. This can be called in the UI (Activity.onStop()) or with explicit close button action. This stops the conference, stops transferring video source and closes the connection to the server.

SocialvidClient.closeConnection()

Mute Audio

Mute the audio source on from the client. A command is sent to the server to mute the audio.

SocialvidClient.muteAudio()

Unmute Audio

Unmute the audio source on from the client. A command is sent to the server to unmute the audio.

SocialvidClient.unMuteAudio()

Mute Video

Mute the video source on from the client. A command is sent to the server to mute the video.

SocialvidClient.muteVideo()

Unmute Video

Unmute the video source on from the client. A command is sent to the server to unmute the video.

SocialvidClient.unMuteVideo()

Send Application Data

Send application specific data to the server

SocialvidClient.unMuteVideo(String appData)

appData: application specific data

Callback handler: RtcListener

Provides an interface with method specifications that will be called on specific events while joining the conference. These callback methods should be implemented by the UI client accordingly.

onConnectionReady()

Callback when the connection to the SocialVid server is created. Client can login to the server only after this callback. The connection takes some time to be created.

onLoginCompleted()

Callback when the login to the SocialVid server is created. Client can issue events (join audio or video conference) only after this callback. The session id will be available for processing the events after login.

oncallReady()

Callback when the system is ready to start the call. On this call the client can prepare for using the local stream. The local camera is started in case of an audio conference and the stream is added to the server.

onStatusChanged(conferenceStatus newStatus)

Callback to update the client on the status of the call. The different statuses are as follows.

```
enum conferenceStatus {
        eStatusConnecting,
        eStatusInitiating,
        eStatusDisconnecting,
        eStatusNewCallerIn
};
```

onLocalStream(MediaStream localStream)

Callback to the UI client with the local stream when the local camera is started. Local Stream can be used by the client to play local user video. This is called only in case of a video conference.

onAddRemoteStream(MediaStream remoteStream, int endpoint)

Callback to the UI client with the remote stream when the remote user joins the conference. Remote Stream can be used by the client to play remote user video/audio. This is called only in case of a video conference.

onRemoveRemoteStream(int endpoint)

Callback to the UI client when the remote user disconnects from the conference. Client can stop playing specific remote user video/audio and also handle the user dropping out.

handleError(errorNumber)

Callback to the client to notify a login error or conference joining error.
Different errors supported are:

```
enum errorNumber {
        eNotSupported,
        eUserNotValid,
        eConferenceNotValid,
        eLicenseNotAvailable,
        eLicenseInvalid,
        eLicenseExpired,
        eConnectionError,
        eLoginRequired,
        eConnectionLost
}
```

- Application is not supported on the device because of Android version of the device. Currently the support is from Android 4.4.
- User is not valid – in case of user login.
- Conference ID is not valid – in case of guest joining a conference.
- License is not available or it is invalid or it has expired.
- Connection is not created to the server before issuing an event (joining the conference). Wait for the callback *onConnectionReady()*
- Login is required before issuing an event (joining the conference)
- Connection to the server is lost

Other info

1. The UI application should include the provided support libraries. The support libraries include autobahn.jar, libjingle_peerconnection.jar, SocialvidClient.jar and the libjingle_peerconnection_so.so
2. The UI application should have the permissions for : *android.permission.CAMERA* and *android.permission.INTERNET*

3. The UI application should have include *android.hardware.camera* and *android.hardware.camera.autofocus* features