Zoom Android SDK

Revision History

Date	History	Author
Sept 18, 2016	First version	Zoom Engineering

Configuration

Minimum supported Android OS version

Android 4.0 +

CPU

armeabi-v7a and above.

For x86, it can be supported via Intel ARM binary translator (Houdini). But your app should not contain other x86 targeted native binaries.

1. In order for the app to run on x86, all the binary files in your app should target on ARM, binary translator will translate them to support x86.

2. If your app contains other binary files that target on x86, you should delete those files to remain only ARM targets. Because on x86 devices, if an app have binary files target on x86, it also should have the file that target on ARM. Because Zoom Android SDK does not contains any x86 binaries, so other files also can't have binaries that target on x86 in order for the SDK to run.

3. The ARM binary translator works at run time, not at compile time.

API Details

You can find the API details in **zoom-sdk/doc** in the SDK package.

Deployment

Android Studio Project “gradle project”

To start using Zoom Android SDK in your project.

- Create a new project with targeted SDK and above.

- Import Modules **zoomsdk**, **zoomcommonlib** and **zoomshareextensions** (optional) into your project

ADT Project

To start using Zoom Android SDK in your project.

Import **zoomsdk** and **zoomcommonlib** as lib projects into your project

Build example projects

- Change constants values defined in src/us/zoom/sdkexample/Constants.java to your values.
- How to build example project in zoom-sdk-adt.zip?

a) Use Eclipse (with ADT plug-in installed)

1) Import 3 projects as Android project
2) Build example project

b) Use ant

1) Define ANDROID_SDK_HOME to your Android SDK root path as a system environment variable;
2) Open command line tool

> cd example
> ant

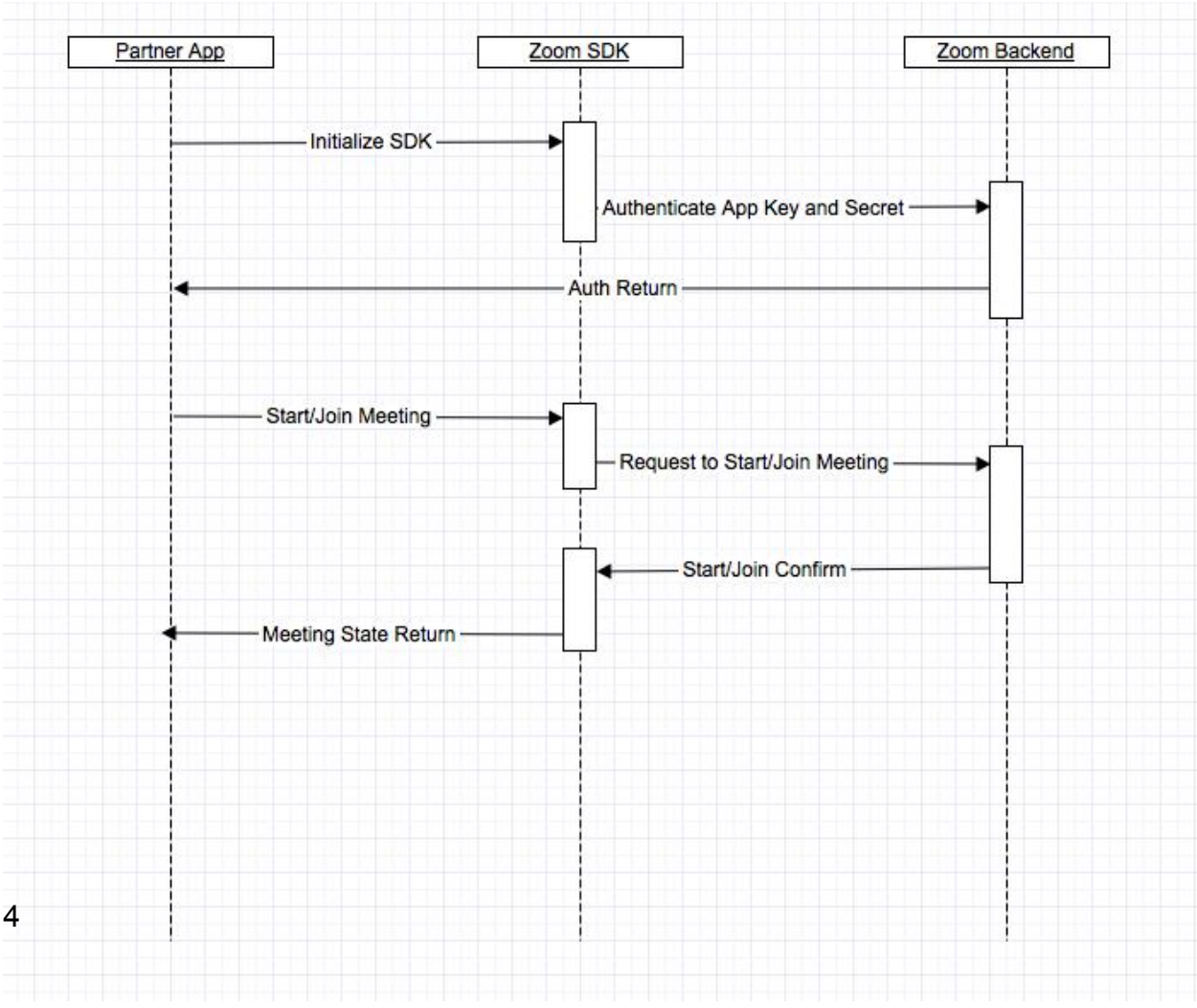
- How to build example project in zoom-sdk-android-studio.zip

> cd zoom-sdk-android-studio
> gradlew.bat example:assembleDebug

Integration

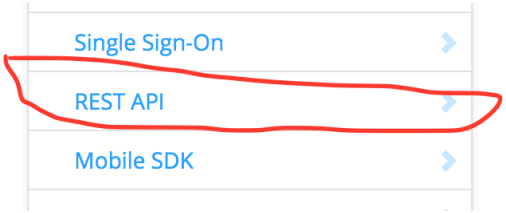
We provided a project named Zoom SDK Android Sample. For the detailed usage, please refer to the sample project.
https://github.com/zoomvideo/Android-Sample-Apps

You can find a sample code for each feature in the sdk folder (example, example 2 and loginexample).

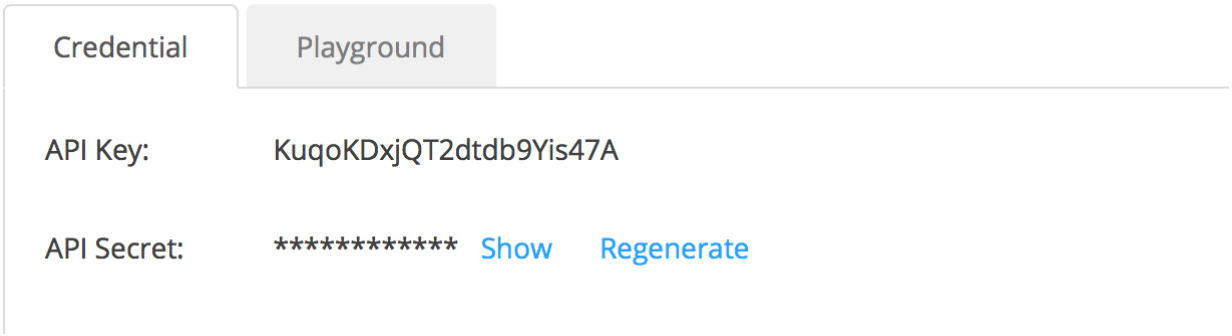


1. SDK Initialization

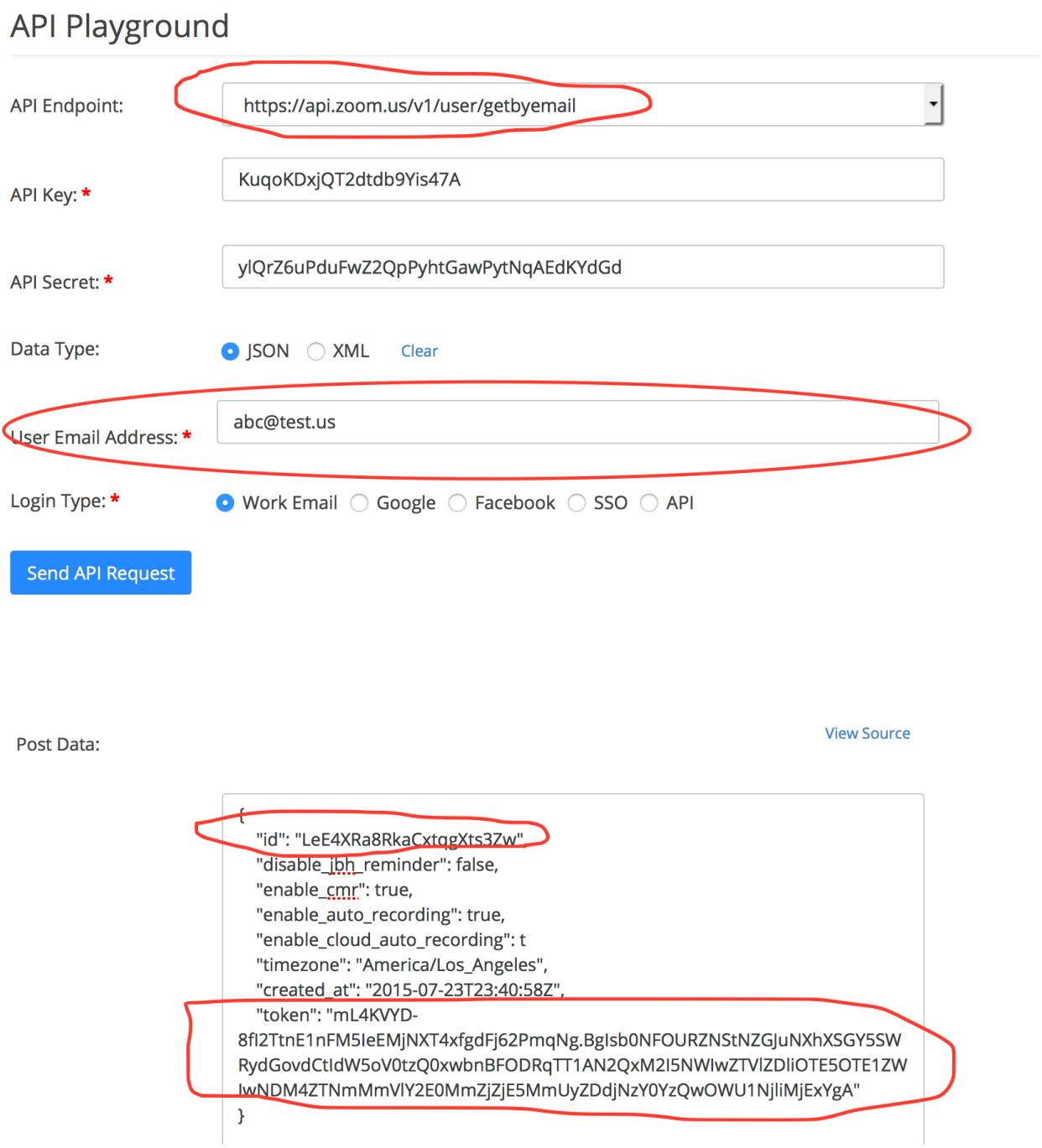
- Get the SDK key and secret from your zoom account. This key/sec is same as the one used for mobile SDK



- Get the REST API Key/Sec from your zoom account



- Call REST API “getbyemail” and this should return the user id and user token



Here is what SDK initialize code looks like:

```
ZoomSDK sdk = ZoomSDK.getInstance();
sdk.initialize(this, APP_KEY, APP_SECRET, WEB_DOMAIN, this);

//set your own keys for dropbox , oneDrive and googleDrive
sdk.setDropBoxAppKeyPair(this, DROPBOX_APP_KEY, DROPBOX_APP_SECRET);
sdk.setOneDriveClientId(this, ONEDRIVE_CLIENT_ID);
sdk.setGoogleDriveClientId(this, GOOGLE_DRIVE_CLIENT_ID);
```

Register ***ZoomSDKInitializeListener*** to get initialize result:

2. Login

There are two options for your app to pass login credentials to the SDK.

1. Use the SDK key/secret from the user account (called SDK user)
2. Pass the Zoom user credentials (login/password) (called login user)

To login using option 2:

```
ZoomSDK zoomSDK = ZoomSDK.getInstance();
if(!(zoomSDK.loginWithZoom(userName, password) ==
ZoomApiError.ZOOM_API_ERROR _SUCCESS))
      { //Error message
} else {
      //something else
}
```

For logout:

```
ZoomSDK zoomSDK = ZoomSDK.getInstance();
if(!zoomSDK.logoutZoom()) {
      //Error
message } else {
      //something else
}
```

Register ***ZoomSDKAuthenticationListener*** to get login/logout result:

3. Start/Join Meeting

Create Meeting Service

```
ZoomSDK zoomSDK = ZoomSDK.getInstance();

if(!zoomSDK.isInitialized()) {
     //Error message
     return;
}

MeetingService meetingService = zoomSDK.getMeetingService();
```

Start Meeting

After SDK was initialized, the app can start a zoom meeting.

For API user:

```
MeetingOptions opts = new MeetingOptions();
//              opts.no_driving_mode = true;
//              opts.no_meeting_end_message = true;
   //              opts.no_titlebar = true;
   //              opts.no_bottom_toolbar = true;
//              opts.no_invite = true;
//              ……
        int ret = meetingService.startMeeting(this, USER_ID, ZOOM_TOKEN,
USER _TYPE, meetingNo, DISPLAY_NAME, opts);
```

For Login user:

```
MeetingOptions opts = new MeetingOptions();
//              opts.no_driving_mode = true;
//              opts.no_meeting_end_message = true;
   //              opts.no_titlebar = true;
   //              opts.no_bottom_toolbar = true;
//              opts.no_invite = true;
//              ……
int ret = meetingService.startMeeting(this, meetingNo, opts);
```

Join Meeting

After SDK was initialized, partner app can join a zoom meeting

```
ZoomSDK zoomSDK = ZoomSDK.getInstance();

if(!zoomSDK.isInitialized()) {
     //Error message
     return;
}

MeetingService meetingService = zoomSDK.getMeetingService();

MeetingOptions opts = new MeetingOptions();
//              opts.no_driving_mode = true;
//              opts.no_meeting_end_message = true;
   //              opts.no_titlebar = true;
   //              opts.no_bottom_toolbar = true;
//              opts.no_invite = true;
//              ……
 int ret = meetingService.joinMeeting(this, meetingNo,
DISPLAY_NAME, meetingPassword, opts);
```

Listener for Meeting Service

Register ***MeetingServiceListener*** to get meeting event.

4. Pre-Meeting Functions

After Zoom user is logged in, the app can schedule, edit, delete or get a meeting item

Create Pre-Meeting Service

```
ZoomSDK zoomSDK = ZoomSDK.getInstance();

if(zoomSDK.isInitialized()) {
      PreMeetingService preMeetingServic =
zoomSDK.getPreMeetingService();
}
```

Pre-Meeting Service function Api

```
publicboolean editMeeting(String meetingId, MeetingItem item);//edit a meeting
publicboolean scheduleMeeting(MeetingItem item);// schedule a meeting
publicboolean deleteMeeting(MeetingItem item); // delete a meeting
publicMeetingItem getMeetingItemByIndex(int index); // get a meeting item by index
publicMeetingItem getMeetingItemByNumber(long meetingNumber); //get a meeting itemby meeting number
```

Listener for Pre-Meeting Service

Register ***PreMeetingServiceListener*** to get function call result.

5. Meeting Options

You can join a meeting with customized options such as disable driver mode, disable meeting invitation, customize invitation action list , show/hide meeting views etc.

```
MeetingOptions opts = new MeetingOptions();
opts.no_driving_mode = true;
opts.no_invite = true;
opts.no_meeting_end_message = true;
opts.no_meeting_error_message = true
opts.no_titlebar = true;
opts.no_bottom_toolbar = true;
opts.no_dial_in_via_phone = true;
opts.no_dial_out_to_phone = true;
opts.no_disconnect_audio = true;
opts.invite_options = InviteOptions.INVITE_VIA_EMAIL + InviteOptions.INVITE_VIA_SMS + InviteOptions.INVITE_COPY_URL;
opts.meeting_views_options= MeetingViewsOptions.NO_BUTTON_SHARE + MeetingViewsOptions.NO_BUTTON_VIDEO;
…
…
int ret = meetingService.startMeeting(this, USER_ID, ZOOM_TOKEN, STYPE, meetingNo, DISPLAY_NAME, opts);
```

For details, please see ***MeetingOptions*** and ***MeetingViewsOptions*** class in api docs

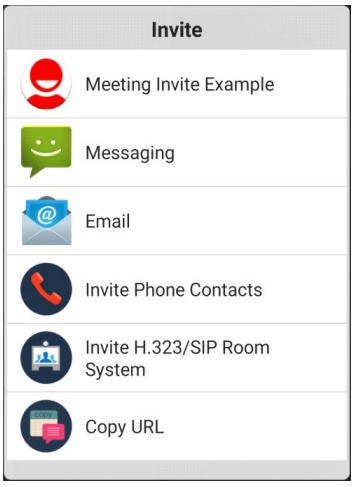
6. Customize Invitation Method

Define an activity with intent-filter as below:

```
        <activity android:name="us.zoom.sdkexample.MyInviteActivity"
                  android:label="@string/invite_acitivity_name"
                  android:icon="@drawable/ic_launcher" >
            <intent-filter>
                <action android:name="us.zoom.sdkexample.intent.action.MeetingInvite"
/>
                <category android:name="android.intent.category.DEFAULT"
            /> </intent-filter>
        </activity>
```

The action name should be your application package name plus "*.intent.action.MeetingInvite*". This activity even can be defined in another application.

Then the activity will be listed at the top of the invite methods list.



Click the item that will open the activity with join meeting URL, invitation topic and invitation content as arguments in the intent. Retrieve them as below:

```
Intent intent = getIntent();
Uri uri = intent.getData();
String subject = intent.getStringExtra(AndroidAppUtil.EXTRA_SUBJECT);
String text = intent.getStringExtra(AndroidAppUtil.EXTRA_TEXT);
```

You can also define this bool value in config.xml to remove all default invite options and remain only your totally customized invite activity.

```
<bool name="zm_config_invite_by_only_action_meeting_invite">true</bool>
```

7. Customize Joining Before Host View

Define an activity with intent-filter as below:

```
<activity android:name="us.zoom.sdkexample.MyWaitJoinActivity"
              android:icon="@drawable/ic_launcher" >
        <intent-filter>
            <action
android:name="us.zoom.sdkexample.intent.action.JoinBeforeHost" />
            <category android:name="android.intent.category.DEFAULT"
        /> </intent-filter>
    </activity>
```

The action name should be your application package name plus "*.intent.action.JoinBeforeHost*". Meeting topic, meeting id, time and meeting type (is repeat or not) can retrieve from the intent arguments. Retrieve them as below:

```
Intent intent = getIntent();
       Uri uri = intent.getData();
       String topic = intent.getStringExtra(AndroidAppUtil.EXTRA_TOPIC); long
 meetingId = intent.getLongExtra(AndroidAppUtil.EXTRA_MEETING_ID, 0);
```

Finish your custom waiting join activity when the meeting is ready to join.
Register ***MeetingServiceListener*** and implement callback method (***onMeetingEvent***) , handler the event MeetingEvent.MEETING_READY_TO_JOIN to return meeting.

8. Customize Invitation email subject and content, SMS content, Copy URL text

Add your invitation content generator class which must implement interface:
com.zipow.videobox.util.InviteContentGenerator

Override and implement genEmailTopic and getEmailContent method and set your Email subject and content as the method return value.
Override and implement genSmsContent method and set your Sms content as the method return value.
Override and implement genCopyUrlText method and set your copy url content as the method return value.
If the return value is null or empty, it will use the zoom default invitation template.

Four input parameters have been provided, you can use them to generate your subject and content:
Application's context, meeting's id, meeting url ,user screen name and meeting password

Add a string resource named "*zm_config_invite_content_generator*" and set your content generator class full name as the string's value.

For detail, you can see the "My*InviteContentGenerator*" and config.xml in example

9. Embed Meeting UI into another Activity

The example2 in Zoom Android SDK package shows how to embed Zoom meeting UI into another activity.

As you can see in example2, in order for MyMeetingActivity to run, you should put this into a separate process. You can look at example2's AndroidManifest.xml. MyMeetingActivity can not be sub activity of other activities, it needs to bounded at the most outside of other activities. Other activities can be sub activities of MyMeetingActivity. The process where MyMeetingActivity is located will auto-restart for several situations, for example: disconnection of meeting service, crashing of meeting service. If there is no meeting, there is also no meeting process, so MyMeetingActivity also won't start. **you have to follow this rule in your application - otherwise Zoom SDK can't handle it.**