Zoom Windows-RTC-Stack

Modification History

Revision	Date	Originator	Comments
2	08/05/2016	Zoom custom engineering	Support login zoom user

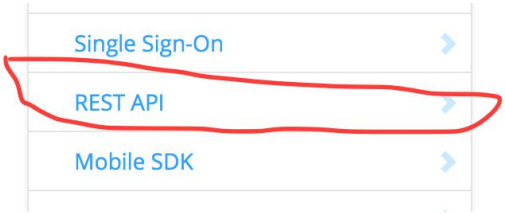
Note that the windows-RTC-Stack from Zoom is distributed under a separate RTC-Stack agreement. Please make sure that you read the terms and conditions of the RTC-Stack agreement before using the RTC-Stack.

1. Introduction and Pre-Requisite

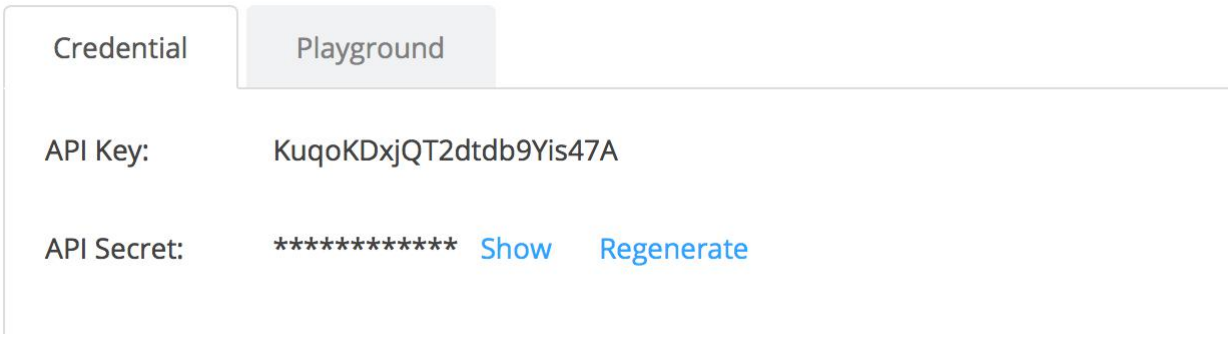
In order to use the SDK, you need to make sure that the RETS API and the mobile SDK is enabled in your account. If not, please contact your account rep or contact Zoom support. Our windows SDK is written in C++ - if you are writing a C#/.NET application, you need to make sure that you can call the C++ library functions from within your application space.

Note that our SDK support meeting services and do not support messaging currently.

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

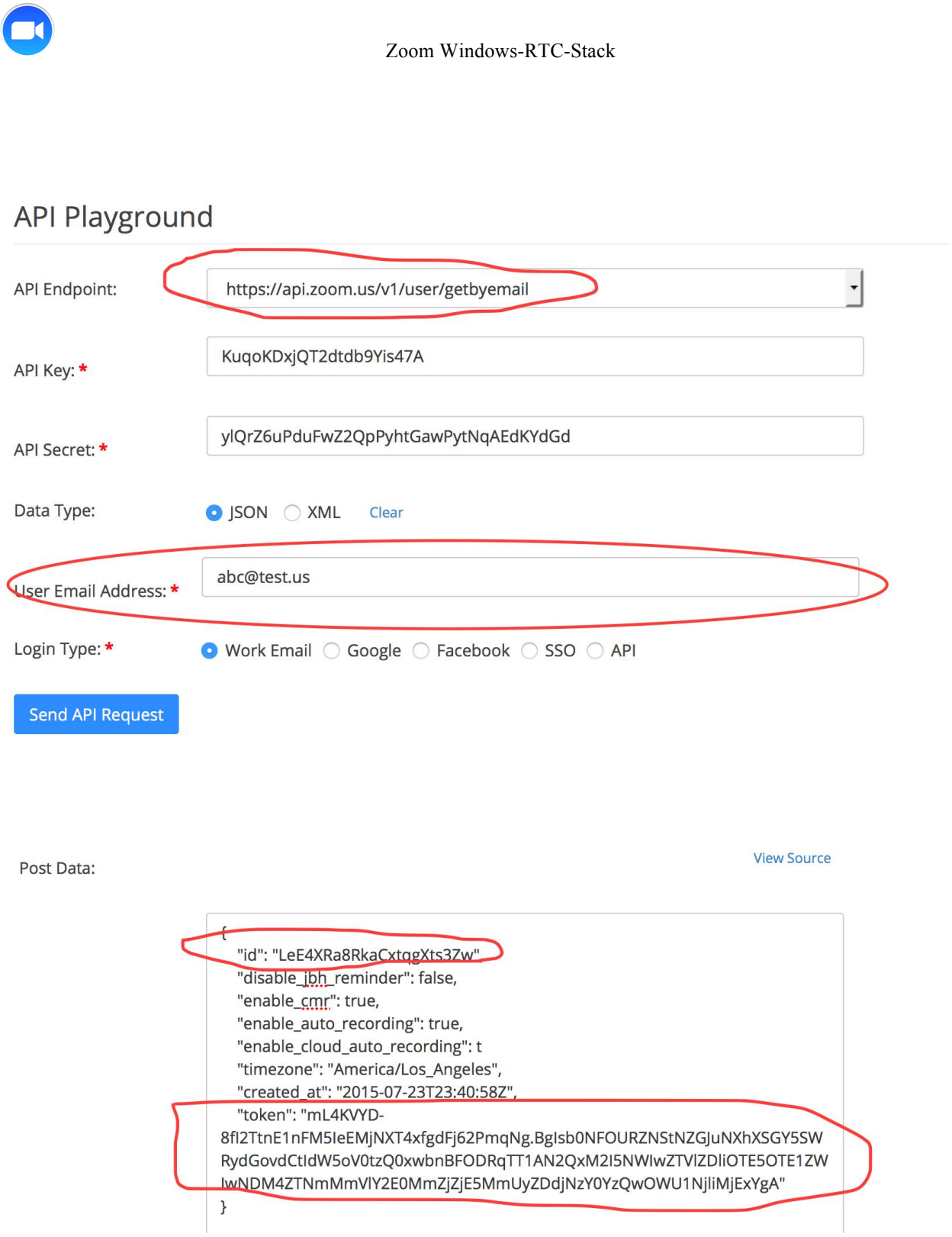


- Get the REST API Key/Sec from your zoom account

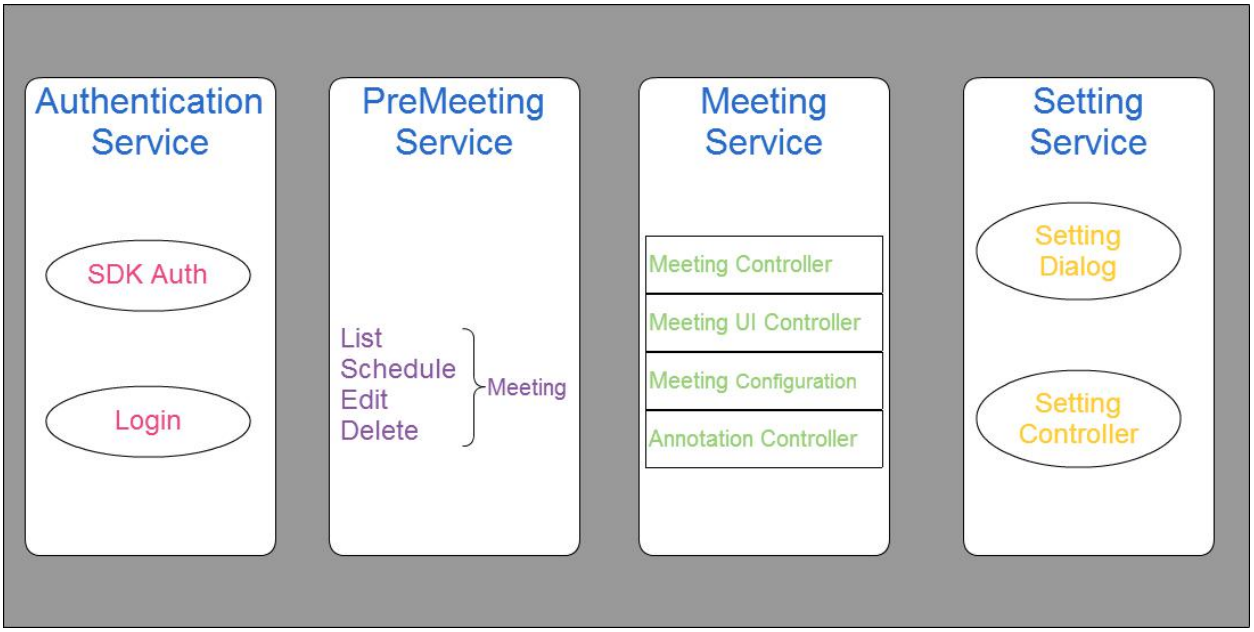


- Call REST API “getbyemail” and this should return the user id and user token

Zoom Windows-RTC-Stack



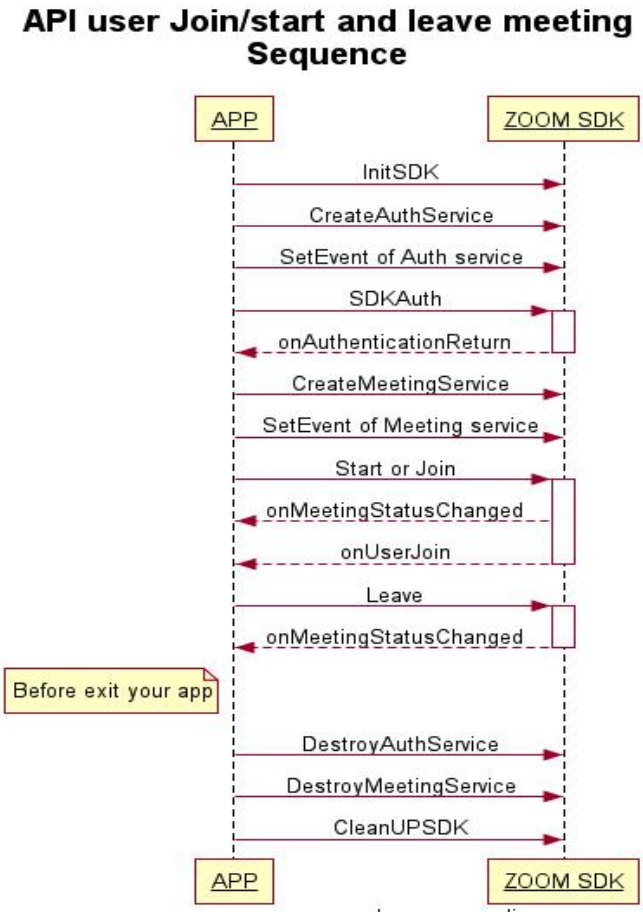
2. RTC-Stack Architecture and Workflow



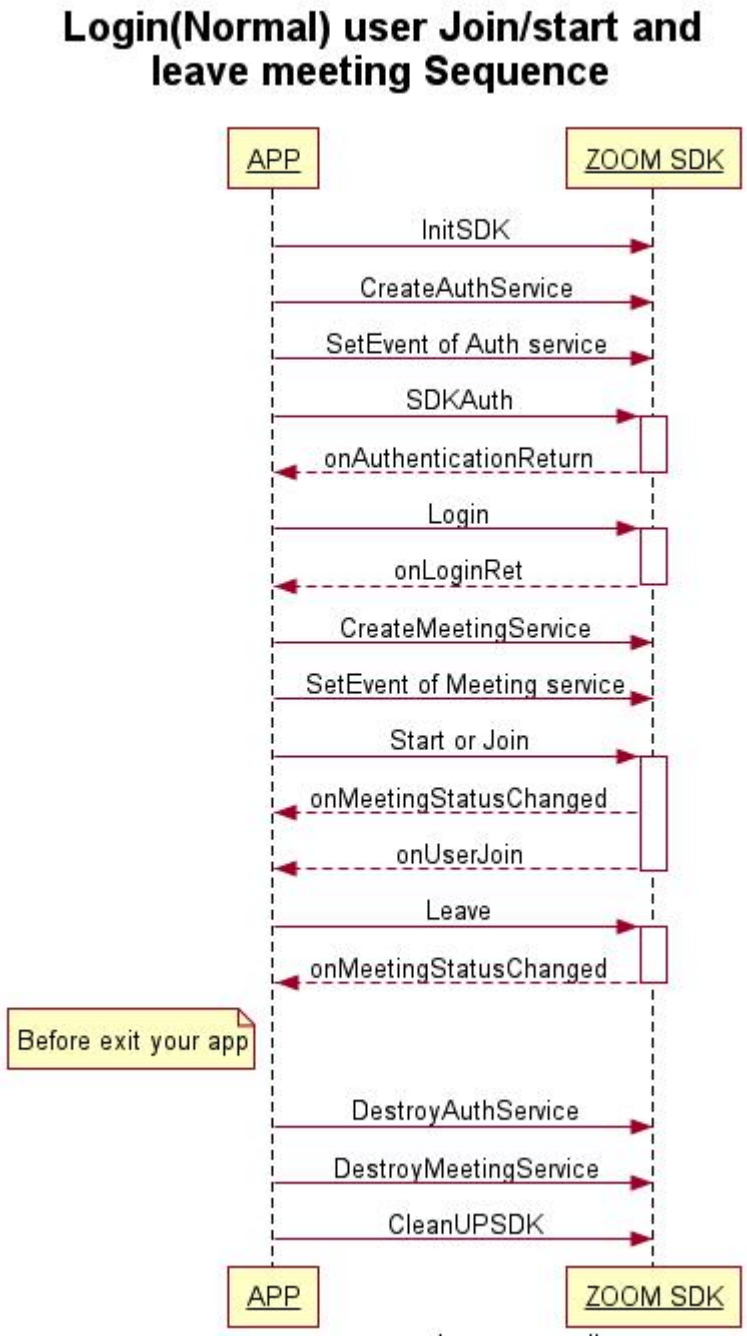
Zoom SDK supports two options for user authentication.

1. You can pass the user id of the user to the SDK initialization and all the meetings will start on that user's account. This is referred to as API user
2. There are times it might be cumbersome to get the user id and you might want to distribute your app to all Zoom users – in this case, you can ask the user to enter the Zoom login credentials (username /password) in your app and then pass it to the SDK. This is referred to as normal user

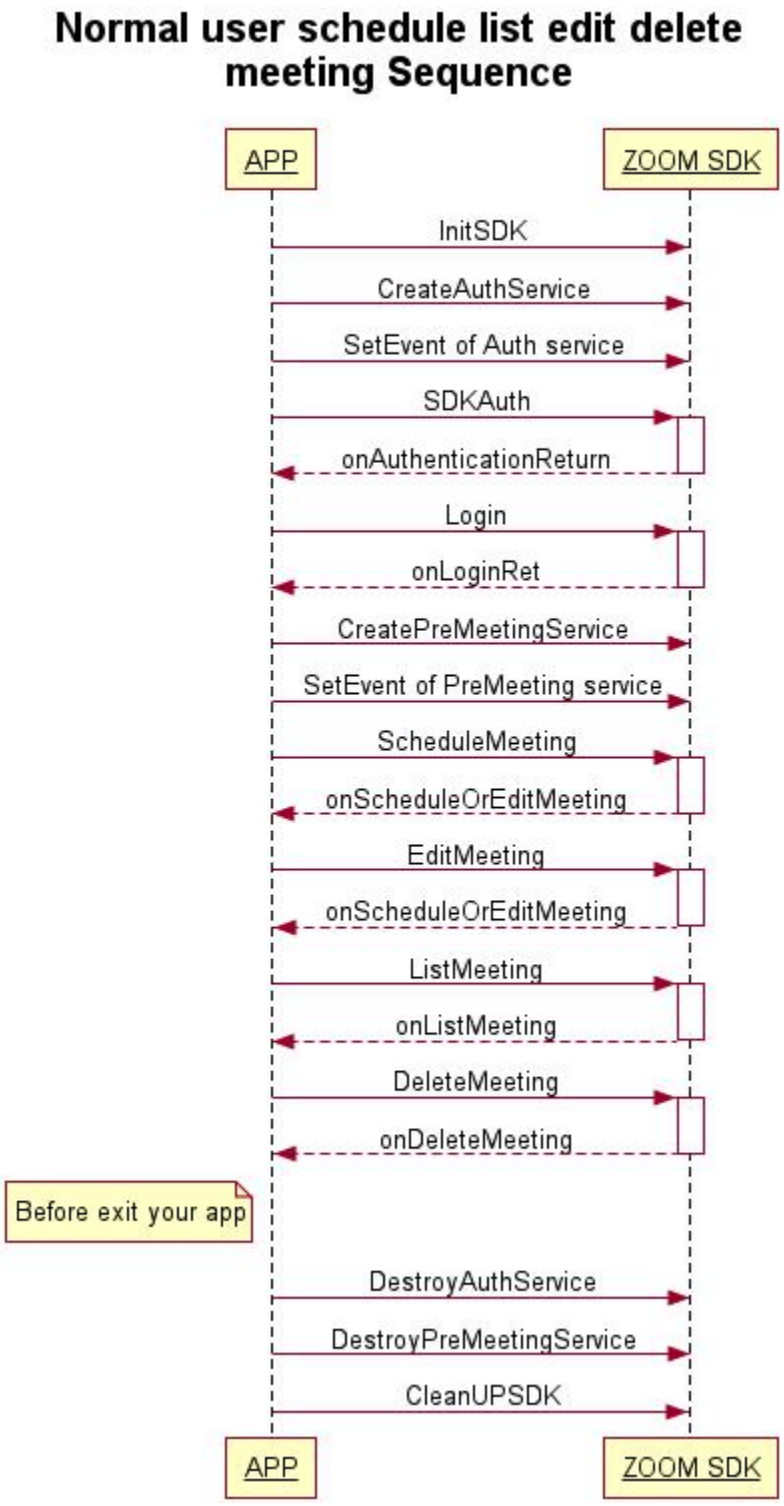
- API User join/start and leave



- Normal User join/start and leave meeting



- Normal User list/schedule/edit/delete meeting



3. RTC-Stack Functions

3.1. Initialization

You can find these APIs in zoom_sdk.h. Please call zoom sdk APIs in main ui thread for thread safe operation.

Initialize zoom sdk:

```
//Initialization code
ZOOM_SDK_NAMESPACE::InitParam param_;
param_.strWebDomain = _T("https://zoom.us/");
aram_.emLanguageID = ZOOM_SDK_NAMESPACE::LANGUAGE_English;//ZOOM_SDK_NAMESPACE::SDK_LANGUAGE_ID
//branding, if you don't config, use zoom as defeault
param_.strBrandingName = "Zoom";
param_.strSupportUrl = "https://support.zoom.us/";
param_.hResInstance = your resource module handle;
param_.uiWindowIconBigID = your windows title big icon id;
param_.uiWindowIconBigID = your windows title samll icon id;
ZOOM_SDK_NAMESPACE::SDKError err = ZOOM_SDK_NAMESPACE::InitSDK(param_);
switch (err)
{
case ZOOM_SDK_NAMESPACE::SDKERR_SUCCESS:
    {
    //success
    }
    break;
default:
    {
    //error handle
    }
    break;
}
```

Cleanup zoom sdk before you exit your application.

```
//Cleanup code
ZOOM_SDK_NAMESPACE::CleanUPSDK();
```

3.2. Authentication Service

You should do authentication first when you start to use zoom sdk and include auth_service_interface.h

- Create authentication service

```
//Create Auth Service
ZOOM_SDK_NAMESPACE::IAuthService* pAuthService(NULL);
ZOOM_SDK_NAMESPACE::IAuthServiceEvent* pAuthServiceEvent;//implement yourself
ZOOM_SDK_NAMESPACE::SDKError err = ZOOM_SDK_NAMESPACE::CreateAuthService(&pAuthService);
if (ZOOM_SDK_NAMESPACE::SDKERR_SUCCESS == err && pAuthService)
{
    pAuthService->SetEvent(pAuthServiceEvent);
    //do you logic
}
```

- Auth SDK

```
//Auth sdk
ZOOM_SDK_NAMESPACE::AuthParam authParam;
authParam.appKey = L"your app key";
authParam.appSecret = L"your app secret";
pAuthService->SDKAuth(authParam);

//Auth sdk callback
//implement yourself
virtual void IAuthServiceEvent::onAuthenticationReturn(AuthResult ret) = 0;
```

- Login end user

```
//Login end user
ZOOM_SDK_NAMESPACE::LoginParam param_;
param_.bRememberMe = true or false;
param_.userName = L"your account";
param_.password = L"your password";
pAuthService->Login(param_);

//Login callback
//implement yourself
virtual void IAuthServiceEvent::onLoginRet(LOGINSTATUS ret, IAccountInfo* pAccountInfo) = 0;
```

- Logout end user

```
//logout end user
pAuthService->LogOut();

//Logout callback
//implement yourself
virtual void IAuthServiceEvent::onLogout() = 0;
```

- Destroy auth service

```
//Destroy auth service
ZOOM_SDK_NAMESPACE::DestroyAuthService(pAuthService);
```

3.3. Pre-meeting Service

If you want to use this service, include premeeting_service_interface.h.

- Create Pre-Meeting Service:

```
//Create premeeting service
ZOOM_SDK_NAMESPACE::IPreMeetingService* pPreMeetingService(NULL);
ZOOM_SDK_NAMESPACE::IPreMeetingServiceEvent* pPreMeetingServiceEvent(NULL);//implement yourself
ZOOM_SDK_NAMESPACE::SDKError err = ZOOM_SDK_NAMESPACE::CreatePreMeetingService(&pPreMeetingService);
if (ZOOM_SDK_NAMESPACE::SDKERR_SUCCESS == err && pPreMeetingService)
{
    pPreMeetingService->SetEvent(pPreMeetingServiceEvent);
    //do your logic
}
```

- Schedule Meeting

```
//Schedule meeting
IScheduleMeetingItem* pScheduleMeetingItem = ZOOM_SDK_NAMESPACE::IPreMeetingService::CreateScheduleMeetingItem();
if (pScheduleMeetingItem)
{
    //optional
    pScheduleMeetingItem->SetAllowJoinBeforeHost(true);
    pScheduleMeetingItem->SetMeetingPassword(password);
    pScheduleMeetingItem->SetUsePMIAsMeetingID(true);
    pScheduleMeetingItem->TurnOffVideoForHost(true);
    pScheduleMeetingItem->TurnOffVideoForAttendee(true);
    //

    pScheduleMeetingItem->SetStartTime(starttime)
    pScheduleMeetingItem->SetDurationInMinutes(duration);
    pScheduleMeetingItem->SetMeetingTopic(topic);
    pPreMeetingService->ScheduleMeeting(pScheduleMeetingItem);
    ZOOM_SDK_NAMESPACE::IPreMeetingService::DestoryScheduleMeetingItem(pScheduleMeetingItem);
}

//Schedule meeting callback
//implement yourself
IPreMeetingServiceEvent
virtual void onScheduleOrEditMeeting(PremeetingAPIResult result, UINT64 meetingUniqueID) = 0;
```

- Edit Meeting

```
//Edit meeting
IScheduleMeetingItem* pEditMeetingItem = ZOOM_SDK_NAMESPACE::IPreMeetingService::CreateScheduleMeeting
Item();
if (pEditMeetingItem)
{
    //optional
    pEditMeetingItem->SetAllowJoinBeforeHost(true);
    pEditMeetingItem->SetMeetingPassword(password);
    pEditMeetingItem->SetUsePMIAsMeetingID(true);
    pEditMeetingItem->TurnOffVideoForHost(true);
    pEditMeetingItem->TurnOffVideoForAttendee(true);
    //

    pEditMeetingItem->SetStartTime(starttime)
    pEditMeetingItem->SetDurationInMinutes(duration);
    pEditMeetingItem->SetMeetingTopic(topic);
    pPreMeetingService->EditMeeting(meetingUniqueID, pEditMeetingItem);
    ZOOM_SDK_NAMESPACE::IPreMeetingService::DestoryScheduleMeetingItem(pEditMeetingItem);
}

//Edit meeting callback
//implement yourself
IPreMeetingServiceEvent
virtual void onScheduleOrEditMeeting(PremeetingAPIResult result, UINT64 meetingUniqueID) = 0;
```

- Delete Meeting

```
//Delete meeting
pPreMeetingService->DeleteMeeting(meetingUniqueID);
//Delete meeting callback
//implement yourself
IPreMeetingServiceEvent
virtual void onDeleteMeeting(PremeetingAPIResult result) = 0;
```

- List Meeting

```
//List meeting
pPreMeetingService->ListMeeting();
//List meeting callback
//implement yourself
IPreMeetingServiceEvent
virtual void onListMeeting(PremeetingAPIResult result, IList<UINT64 >* lstMeetingList) = 0;
```

- Get meeting item by ID

Refer the header file for the details.

- Destroy Pre-Meeting Service

```
//Destroy premeeting service
ZOOM_SDK_NAMESPACE::DestroyPreMeetingService(pPreMeetingService);
```

3.4. Meeting Service

If you want to use this service, include meeting_service_interface.h.

- **Create meeting service**

```
//Create meeting service
ZOOM_SDK_NAMESPACE::IMeetingService* pMeetingService(NULL);
ZOOM_SDK_NAMESPACE::IMeetingServiceEvent* pMeetingServiceEvent;//implement yourself
ZOOM_SDK_NAMESPACE::SDKError err = ZOOM_SDK_NAMESPACE::CreateMeetingService(&pMeetingService);
if (ZOOM_SDK_NAMESPACE::SDKERR_SUCCESS == err && pMeetingService)
{
    pMeetingService->SetEvent(pMeetingServiceEvent);
    //do your logic
}
```

- **Destroy meeting service**

```
//Destroy meeting service
ZOOM_SDK_NAMESPACE::DestroyMeetingService(pMeetingService);
```

3.4.1. Meeting controller

- Start meeting for API user

```
//Start meeting for api user
ZOOM_SDK_NAMESPACE::StartParam startParam;
startParam.userType = ZOOM_SDK_NAMESPACE::SDK_UT_APIUSER;
ZOOM_SDK_NAMESPACE::StartParam4APIUser& apiuserParam = startParam.param.apiuserStart;
apiuserParam.userID = L"your user id, get this via RestAPI";
apiuserParam.userToken = L"your user id, get this via RestAPI";
apiuserParam.userName = L"your display name"
apiuserParam.meetingNumber = 123456789;///you can schedule meeting via RestAPI
apiuserParam.isDirectShareDesktop = false;///direct share desktop or not when you start meeting
apiuserParam.hDirectShareAppWnd = NULL;///direct share window or not when you start meeting
pMeetingService->Start(startParam);

//Start meeting callback
//implement yourself, for more details please read comments of IMeetingServiceEvent in head file
IMeetingServiceEvent
virtual void onMeetingStatusChanged(MeetingStatus status, int iResult = 0) = 0;
virtual void onUserJoin(IList<unsigned int >* lstUserID, const wchar_t* strUserList = NULL) = 0;
virtual void onUserAudioStatusChange(IList<IUserAudioStatus* >* lstAudioStatusChange, const wchar_t* s
trAudioStatusList = NULL) = 0;
```

- Start meeting for End user

```
//Start meeting for end user
ZOOM_SDK_NAMESPACE::StartParam startParam;
startParam.userType = ZOOM_SDK_NAMESPACE::SDK_UT_NORMALUSER;
ZOOM_SDK_NAMESPACE::StartParam4NormalUser& normalParam = startParam.param.normaluserStart;
normalParam.meetingNumber = 0;//your meeting number or 0 for instance meeting
normalParam.isDirectShareDesktop = false;///direct share desktop or not when you start meeting
normalParam.hDirectShareAppWnd = NULL;///direct share window or not when you start meeting
pMeetingService->Start(startParam);

//Start meeting callback
//implement yourself, for more details please read comments of IMeetingServiceEvent in head file
IMeetingServiceEvent
virtual void onMeetingStatusChanged(MeetingStatus status, int iResult = 0) = 0;
virtual void onUserJoin(IList<unsigned int >* lstUserID, const wchar_t* strUserList = NULL) = 0;
virtual void onUserAudioStatusChange(IList<IUserAudioStatus* >* lstAudioStatusChange, const wchar_t* s
trAudioStatusList = NULL) = 0;
```

- Join Meeting for API user

```
//Join meeting for api user
ZOOM_SDK_NAMESPACE::JoinParam joinParam;
joinParam.userType = ZOOM_SDK_NAMESPACE::SDK_UT_APIUSER;
ZOOM_SDK_NAMESPACE::JoinParam4APIUser& apiParam = joinParam.param.apiuserJoin;
apiParam.meetingNumber = 123456789;//meeting number you want to join
apiParam.userName = L"your display name";
apiParam.psw = L"Meeting's password";
apiParam.isDirectShareDesktop = false;///direct share desktop or not when you start meeting
apiParam.hDirectShareAppWnd = NULL;///direct share window or not when you start meeting
apiParam.toke4enfrocelogin = L"token for enforce login meeting";//if the meeting is just for signed us
er.
apiParam.participantId = L"participant Id";///for meeting participant report list, need web backend en
able.
pMeetingService->Join(joinParam);

//join meeting callback
//implement yourself, for more details please read comments of IMeetingServiceEvent in head file
IMeetingServiceEvent
virtual void onMeetingStatusChanged(MeetingStatus status, int iResult = 0) = 0;
virtual void onUserJoin(IList<unsigned int >* lstUserID, const wchar_t* strUserList = NULL) = 0;
virtual void onUserAudioStatusChange(IList<IUserAudioStatus* >* lstAudioStatusChange, const wchar_t* s
trAudioStatusList = NULL) = 0;
```

- Join meeting for end user

```
//Join meeting for end user
ZOOM_SDK_NAMESPACE::JoinParam joinParam;
joinParam.userType = ZOOM_SDK_NAMESPACE::SDK_UT_NORMALUSER;
ZOOM_SDK_NAMESPACE::JoinParam4NormalUser& normalParam = joinParam.param.normaluserJoin;
normalParam.meetingNumber = 123456789;//meeting number you want to join
normalParam.userName = L"your display name";
normalParam.psw = L"Meeting's password";
normalParam.isDirectShareDesktop = false;///direct share desktop or not when you start meeting
normalParam.hDirectShareAppWnd = NULL;///direct share window or not when you start meeting
pMeetingService->Join(joinParam);

//join meeting callback
//implement yourself, for more details please read comments of IMeetingServiceEvent in head file
IMeetingServiceEvent
virtual void onMeetingStatusChanged(MeetingStatus status, int iResult = 0) = 0;
virtual void onUserJoin(IList<unsigned int >* lstUserID, const wchar_t* strUserList = NULL) = 0;
virtual void onUserAudioStatusChange(IList<IUserAudioStatus* >* lstAudioStatusChange, const wchar_t* s
trAudioStatusList = NULL) = 0;
```

- Other interfaces

Please refer the Zoom SDK.chm

3.4.2. Meeting UI controller

You can get this controller after you have been in the meeting. If not, you can't use this controller.

- Usage

```
//How to user meeting ui controller
ZOOM_SDK_NAMESPACE::IMeetingUIController* pMeetingUIController = pMeetingService->GetUIController();
if (pMeetingUIController)
{
    //control meeting ui, such as
    pMeetingUIController->EnterFullScreen(true, false); //Enter full screen mode
    pMeetingUIController->ExitFullScreen(true, false);//Exit full screen mode
    pMeetingUIController->ShowChatDlg(param_);//show chat
    pMeetingUIController->SwitchToVideoWall();//switch to video wall mode
    //for more details, you can read the IMeetingUIController comments or Zoom SDK.chm
}
```

3.4.3. Meeting Configuration controller

You can set configuration before you call start/join meeting API. And if the meeting is ended or left, the configuration will be reset automatically.

- Usage

```
//How to user meeting configuration
ZOOM_SDK_NAMESPACE::IMeetingConfiguration* pMeetingConfig = pMeetingService->GetMeetingConfiguration();
if (pMeetingConfig)
{
    //popup wrong password error dialog or not when join meeting.
    pMeetingConfig->DisablePopupMeetingWrongPSWDlg(true);
    //don't show zoom meeting id in meeting ui title
    pMeetingConfig->HideMeetingInfoFromMeetingUITitle(true);
    //popup waiting for host join dialog or not when join meeting.
    pMeetingConfig->DisableWaitingForHostDialog(true);
    //for more details, you can read the IMeetingConfiguration comments in head file or Zoom SDK.chm
}
```

3.4.4. Meeting Annotation controller

You can get this controller when someone or you are sharing. If not, you can't use this controller.

- Usage

```
//How to user meeting annotation
ZOOM_SDK_NAMESPACE::IAnnotationController* pAnnoCtrl = pMeetingService->GetAnnotationController();
if (pAnnoController)
{
    //show zoom annotation bar
    pAnnoController->StartAnnotation(leftpos, toppos);
    pAnnoController->StopAnnotation();
    //set color of the tools
    pAnnoController->SetColor(color);
    //set tool of the annotation
    pAnnoController->SetTool(type);
    //for more details, you can read the IAnnotationController comments in head file or Zoom SDK.chm
}
```

- **Meeting Event**

All of meeting event define at IMeetingServiceEvent. You can refer the header file or Zoom SDK.chm

3.5. Setting Service

If you want to user this service, include setting_service_interface.h.

- Usage


```
//How to use setting service
ZOOM_SDK_NAMESPACE::ISettingService* pSettingService(NULL);
ZOOM_SDK_NAMESPACE::SDKError err = ZOOM_SDK_NAMESPACE::CreateSettingService(&pSettingService);
if (ZOOM_SDK_NAMESPACE::SDKERR_SUCCESS == err && pSettingService)
{
    pSettingService->ShowSettingDlg(param_);//show zoom setting dialog
    pSettingService->SetRecordingPath(path);//set record path
    pSettingService->EnableAutoJoinAudio(true);//auto join audio when you join/start meeting
    //other interface, you can read the ISettingService comments in head file or Zoom SDK.chm
}

//Destroy setting service
ZOOM_SDK_NAMESPACE::DestroySettingService(pSettingService);
```