TrueConf

TrueConf Mobile SDK

Руководство разработчика

Оглавление

1. Общее описание

TrueConf Mobile SDK позволяет создавать собственные приложения с поддержкой видеоконференцсвязи на мобильных платформах Android и iOS/iPadOS. Главное преимущество наших библиотек — это гарантия качественной связи в любых сетях и на большинстве представленных на рынке мобильных устройств. Вам не нужно переживать об адаптации видеопотоков под каналы связи или о тонкостях аппаратного кодирования для снижения нагрузки на устройства и экономии заряда батареи.

Для удобства интеграции видеосвязи в корпоративные приложения предоставляется возможность использовать разные среды разработки (IDE):

- Android Studio для разработки под Android
- Xcode для разработки под iOS/iPadOS
- Apache Cordova, React Native, .NET MAUI для кроссплатформенной разработки

Приложения, использующие SDK, требуют подключения к корпоративному серверу видеоконференцсвязи TrueConf Server или к облачному сервису TrueConf Online. Для управления пользователями, конференциями и их участниками необходимо использовать API TrueConf Server.

Актуальная версия SDK доступна в репозитории `git@git.trueconf.ru:SDK`, его надо будет импортировать как показано далее.

1.1. Возможности Mobile SDK

В настоящий момент в TrueConf Mobile SDK доступны следующие функции:

- Присоединение к выбранному серверу
- Авторизация выбранным логином/паролем
- Проверка статуса другого пользователя и получение информации о его изменении
- Звонок указанному пользователю
- Получение запроса о входящем звонке
- Присоединиться к групповой конференции
- Кастомизация элементов управления конференцией.

При этом возможности TrueConf Server по проведению конференций позволяют:

- запланировать конференцию на сервере заранее;
- создавать конференции средствами TrueConf Server API;
- добавлять в конференции абонентов ВКС-терминалов по протоколам SIP и H.323;
- присоединять к конференции RTSP-трансляции и IP-камеры;
- разрешить вход в конференцию по ссылке через браузер (с помощью WebRTC) через гостевой аккаунт;
- и многое другое, что указано в документации сервера видеосвязи.

Пользователи SDK во время конференции могут также просматривать транслируемый с других устройств контент и произвольно менять расположение участников на экране.

Наибольший функционал можно получить, используя SDK совместно с TrueConf Server API. SDK позволяет устанавливать поведение одного пользователя (аналогично клиентскому приложению), а TrueConf Server API — отслеживать состояние конференции и управлять профилями и правами пользователей в целом.

2. Схема работы Mobile SDK

TrueConf Server реализован в виде класса, предоставляющего доступ к основным функциям, и механизмам подписки на события. Основной функционал для iOS и Android унифицирован, тем не менее есть некоторые различия. Подробнее об этом вы узнаете далее.

Общий алгоритм работы:

1. Имеются сервер видеосвязи и клиентское приложение (терминал).
2. Для проведения сеанса видеосвязи к терминалу должен быть привязан пользователь, от имени которого производятся все операции. Доступно использование гостевых подключений.
3. Сеанс связи между двумя точками (терминалами) проводится по запросу одного из участников.
4. Для конференций с несколькими участниками на сервере создается отдельный объект-конференция с собственным идентификатором и временем жизни (текущая версия SDK не поддерживает эту функциональность, возможно лишь подключение к уже созданной групповой конференции). Терминал должен позвонить в такую конференцию сам или быть добавлен в список участников до её начала.

2.1. Соединение с сервером

Терминал, при наличии сетевого соединения на устройстве, пытается соединиться с сервером видеосвязи. После установки соединения оно продолжает оставаться активным в течении всего времени работы системы. При обрыве происходит автоматическая попытка восстановить связь. Если в течении установленного переподключение не удаётся, то происходит информирование приложения и попытки установить связь продолжаются с меньшей интенсивностью.

2.2. Авторизация

Терминал должен быть авторизован на сервере с помощью уникального идентификатора пользователя (TrueConf ID) и пароля. Если на стороне TrueConf Server имеются гостевые лицензии, то будет доступна генерация временных пользователей с одноразовыми паролями (гостей).

Одним идентификатором можно быть авторизованным только на одном терминале, то есть для SDK в отличие от обычных клиентских приложений Труконф не поддержана множественная авторизация. При логине на другом терминале, терминал, авторизованный ранее, будет автоматически отключен.

Пользователь считается авторизованным на конкретном терминале и доступным для вызова до тех пор, пока его терминал сохраняет соединение с сервером. При обрыве связи терминал пробует автоматически восстановить соединение, и, если это не удается в течение заданного времени, он считается отключенным. В зависимости от настроек терминала (включена опция автологина) при перезапуске клиента или восстановлении связи он может пытаться авторизоваться последним использованным идентификатором.

Помимо TrueConf ID (логина) используется отображаемое имя, которое используется для показа на элементах интерфейса. В некоторых случаях может отсутствовать, тогда вместо него используется TrueConf ID.

2.3. Конференция

Сеанс начинается по инициативе одного из пользователей. Для этого его приложение должно быть подключено к серверу, а пользователь на нём авторизован (см. общий алгоритм работы). При выполнении звонка терминал переходит в режим ожидания, который завершится либо началом конференции, либо уведомлением о невозможности её начать.

На первой стадии сервер проверяет что вызываемый абонент существует и находится в состоянии "онлайн". После этого на сторону вызываемого отправляется запрос. Он может быть принят или отвергнут. Если в течении определённого времени нет реакции, запрос автоматически считается отвергнутым. Если получен положительный ответ, вызывающая сторона информируется об этом, и через сервер устанавливается соединение двух участников. В случае, когда оба участника конференции с точки зрения топологии сети находятся в прямой видимости, возможно установление прямого соединения минуя сервер.

Сеанс может быть завершен либо по команде одной из сторон, либо при невозможности восстановить оборванное соединение в течении заданного времени.

Пользовательский интерфейс виджета конференции встроен в SDK и не может быть изменен кроме брендирования элементов интерфейса и ручного формирования раскладки. Вне конференции интерфейс и

поведение определяется использующей его программой.

2.3.1. Передача видео и аудио

При соединении в сеанс, качество и задержка передаваемых видео и аудио данных определяется состоянием канала и производительностью терминалов. Так работает технология SVC, используемая в системе видеосвязи Труконф. Оно подстраивается в зависимости при изменении состоянии канала и меняется динамически в течении сеанса связи. Существуют глобальные ограничения максимального потока в зависимости от типа используемого сетевого соединения (3G, Wi-Fi, проводное) и ограничения максимального потока, задаваемые на стороне сервера. При их наличии, выбирается вариант с минимальным битрейтом. Задача ограничения битрейта на стороне терминала в текущей версии SDK не поддерживается.

В процессе сеанса возможно переключение между видеокамерами и динамиками/гарнитурами. При падении качества канала ниже предельного значения, возможно принудительное отключение отправки и получения видео, с приоритезацией аудио потока.

2.4. Дополнительные сервисные функции

Сервер видеосвязи отслеживает статус авторизованных пользователей и может автоматически уведомлять о его изменении другие терминалы (используется событие `onUserStatusUpdate`).
Информирование о статусе осуществляется на основании подписки. Её можно сделать на сервере, через редактирование адресных книг пользователей, и на терминале, путем разового запроса статуса нужного пользователя (в пределах одного сеанса авторизации).

2.5. Состояния работы приложения с SDK

Операции, приводящие к изменению режима терминала, могут выполняться длительное время и в любой момент могут быть прерваны (например, при пропадании сетевого соединения). Потому SDK оперирует однозначно определёнными состояниями терминала и переходами между ними. Основными операциями являются команды изменения состояния и события, сообщающие о его фактическом изменении или произошедшей ошибке. Изменение состояний представлено на рисунке ниже:

2.6. Статусы пользователей

Состояние других абонентов, получаемые при подписке на изменение статусов, несколько отличаются от состояний терминала. Таблица соответствий приведена ниже.

Состояние терминала	Статус пользователя	Примечания

	Undefined	Технический статус, означает, что информация о статусе ещё не получена
	Unknown	Абонент с таким TrueConf ID не существует или для него не может быть получен статус
Disconnected	Offline	
Connected	Offline	
Logined	Online	Абонент зарегистрирован в сети и может принимать входящие звонки
In conference	Busy	Абонент находится в конференции и может принимать специальные запросы, но звонок типа точка-точка ему невозможен

2.7. Гостевые пользователи

Вы можете подключаться к публичным конференциям (вебинарам) в качестве гостя. Для этого нужно сгенерировать гостевые логин и пароль по следующей инструкции:

`LOGIN` — `*guest2*EXTID*DISPLAYNAME`

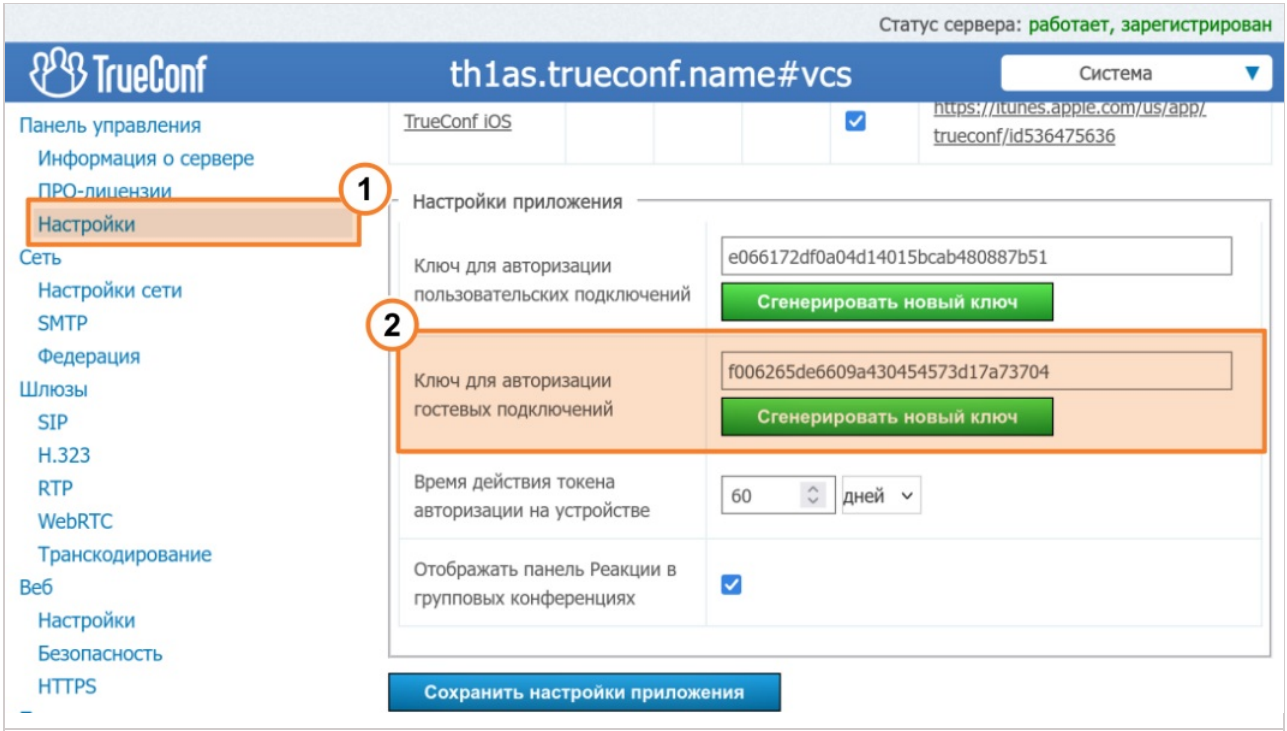
Где:

- `EXTID` — `External ID` (любой текст)
- `DISPLAYNAME` — отображаемое имя для гостя. Может быть пустым. Должен быть в UTF8.

`PASSWORD` — `$2RAND*TIMESTAMP*SIGN`

Где:

- `RAND` — любая строка без символа `*`
- `TIMESTAMP` — время истечения срока действия пароля в UTC (GMT) в секундах от 1 января 1970 (integer)
- `SIGN = md5(RAND + EXTID + TIMESTAMP + SECRET)`, где `SECRET` — параметр **Ключ для авторизации гостевых подключений** вашего сервера из раздела **Настройки**:



Если логин начинается с `guest2`, система ожидает пароль в специальном формате. Если пароль корректный и `TIMESTAMP` не старше времени сервера, тогда после подключения к конференции гость будет иметь Call ID `#guest2:EXTID@server.name`.

i Возможность подключения к публичным конференциям в качестве гостя и общее число таких доступных подключений зависит от лицензии на ваш TrueConf Server.

3. Получение доступа к SDK и его установка

Исходные коды TrueConf Mobile SDK для мобильных приложений находятся в удалённом репозитории, доступ к которому ограничен и выдаётся отделом продаж компании Труконф по запросу с вашей стороны. В репозитории, кроме самого SDK, вы найдёте примеры полноценных приложений с использованием SDK для демонстрации его возможностей.

Ниже мы подробно расскажем, как получить доступ и подключиться к репозиторию.

3.1. Шаг 1. Установите Git

Мы используем систему контроля версий Git, так что вам нужно будет установить себе ПО для работы с ней. Подробную инструкцию по установке для разных операционных систем можно найти на сайте Git `git-scm.com`.

3.2. Шаг 2. Сгенерируйте ssh-ключ

Как было сказано выше, доступ к репозиторию ограничен. Для идентификации пользователя и предоставления доступа используется стандартное средство Git – SSH-ключ.

Такой ключ генерируется самим пользователем и состоит из двух частей – открытой и закрытой (они хранятся в двух разных файлах). Открытую часть ключа вы передаёте нам, а закрытую оставляете у себя на компьютере в специальной директории `.ssh`. Остальное сделает Git. Для генерации ключа введите команду в терминале своей ОС:

```sh
ssh-keygen
```

В качестве папки для установки рекомендуем оставить ту, которая предлагается по умолчанию:

- на ОС Windows `C:\Users\MyUser\.ssh`
- на Linux `/home/MyUser/.ssh`
- на macOS `/Users/MyUser/.ssh`

где **MyUser** – имя вашего пользователя в ОС.

3.3. Шаг 3. Запросите пробный период использования SDK в отделе продаж

После того, как ключ сгенерирован, обратитесь в отдел продаж с просьбой предоставить доступ к Git репозиторию. Также к Android библиотекам необходим доступ к maven репозиторию по логину и паролю.

Отдел продаж может поинтересоваться, для каких целей вы собираетесь использовать SDK. Постарайтесь подробно описать ваш проект – если мы посчитаем его интересным, то предоставим вам дополнительно выделенного технического специалиста на период тестирования.

Когда отдел продаж примет решение о предоставлении вам доступа, отправьте им открытую часть своего ключа (файл с расширением `.pub` в директории, куда был помещён сгенерированный ключ).

3.4. Шаг 4. Скачайте репозиторий

Чтобы склонировать репозиторий себе на компьютер, выполните в терминале вашей ОС команду:

```sh
git clone git@git.trueconf.ru:SDK TARGET_PATH
```

где `TARGET_PATH` – путь к директории, куда вы хотите его скопировать.

Например, на Linux команда может выглядеть так:

```sh
git clone git@git.trueconf.ru:SDK /home/user/mobile-sdk
```

Если `TARGET_PATH` не указан, то репозиторий будет загружен в текущую директорию (из которой работает консоль).

С дополнительными параметрами команды `git clone` вы можете ознакомится на сайте Git - https://git-scm.com/docs/git-clone

Чтобы избежать работы с консолью, вы можете использовать графические оболочки Git с удобным интерфейсом, например:

- GitExtension - https://gitextensions.github.io/
- SourceTree - https://sourcetreeapp.com/

В них при клонировании в качестве адреса внешнего репозитория указывайте `git@git.trueconf.ru:SDK`

4. Технические требования и установка

4.1. xCode – разработка под iOS

Вам потребуется Xcode версии 12 или новее. Поддерживаемые версии iOS – 13 и новее. Для продолжения работы видеоконференции, когда ваше приложение уходит в фон, необходимо в настройках приложения включить соответствующие **Background Modes**. В файле `info.plist` должны быть такие ключи:

```
<key>UIBackgroundModes</key>
<array>
    <string>audio</string>
    <string>voip</string>
</array>
```

Для правильной работы приложения с камерой и микрофоном добавьте в файл `info.plist` следующие строки:

```
<key>NSCameraUsageDescription</key>
<string>To allow other people to see you</string>
<key>NSMicrophoneUsageDescription</key>
<string>To allow other people to hear you</string>
```

Для использования SDK нужно добавить в ваш проект файл `TrueConfSDK.xcframework`.

Далее обязательно убедитесь, что `TrueConfSDK.xcframework` прописался ва вашей IDE по пути: **Targets → Project-Name →General →Frameworks, Libraries and Embedded Content** где **Project-Name** – ваш проект.

Подключение SDK к классам Objective-C проекта производится следующими строками:

```
import "TrueConfSDK/TrueConf.h"
import "TrueConfSDK/TrueConfSDK.h"
```

Для Swift проектов достаточно прописать строку:

```
import TrueConfSDK
```

Framework поддерживает архитектуру **arm64** для iOS устройств, а также **x86_64** и **arm64** для симуляторов.

4.2. Android Studio

Проект создан для использования в среде разработки AndroidStudio. Поддерживаемая версия Android – не ниже API 24 (Android 7.0), targetSdk и compileSdk – API 34 (Android 14.0). Библиотеки собраны при использовании Java 17.

Для подключения SDK требуется изменить несколько файлов в проекте:

1. В файле настроек `.gradle` (подробнее о нём в официальной документации Gradle):
- В разделе `repositories` добавьте maven репозиторий. `username` и `password` выдаются по запросу через вашего менеджера:

```
maven {
    credentials {
        username 'username'
        password 'password'
    }
    url 'https://sdk.trueconf.com/maven/repository/maven-public/'
}
```

- В разделе `dependencies` подключите библиотеки SDK нужной версии, например:

```
api 'com.trueconf:trueconfsdk:3.0.0.34@aar'
api 'com.trueconf:media:3.0.0.34@aar'
api 'com.trueconf:jnicore:3.0.0.34@aar'
```

- Также в `.gradle` должны быть прописаны следующие зависимости:

```
implementation 'androidx.work:work-runtime:2.9.0'
implementation 'androidx.concurrent:concurrent-futures:1.1.0'
implementation 'androidx.core:core:1.12.0'
implementation 'androidx.appcompat:appcompat:1.6.1'
implementation 'androidx.gridlayout:gridlayout:1.0.0'
implementation 'androidx.mediarouter:mediarouter:1.7.0'
implementation 'androidx.recyclerview:recyclerview:1.3.2'
implementation 'androidx.leanback:leanback:1.0.0'
implementation 'androidx.leanback:leanback-preference:1.0.0'
implementation 'androidx.multidex:multidex:2.0.1'
implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
implementation 'androidx.emoji:emoji-bundled:1.1.0'
implementation 'com.google.android.material:material:1.11.0'
implementation 'org.greenrobot:eventbus:3.3.1'
```

2. Подключите SDK к классам проекта в коде приложения (пример для Java):

```java
import com.trueconf.sdk.TrueConfSDK;// to work with SDK methods
import com.trueconf.sdk.TrueConfListener;    // to work with SDK events
```

3. В файле манифеста `AndroidManifest.xml`:

- Добавьте следующие возможности для вашего приложения:

```
<uses-feature
      android:glEsVersion="0x00020000"
      android:required="true"/>
  <uses-feature
      android:name="android.hardware.audio.low_latency"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.camera"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.camera.autofocus"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.camera.flash"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.camera.front"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.microphone"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.sensor.accelerometer"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.sensor.light"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.sensor.proximity"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.touchscreen"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.wifi"
      android:required="false"/>
  <uses-feature
      android:name="android.hardware.bluetooth"
      android:required="false"/>
```

- Добавить следующие разрешения :

```
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.permission.BLUETOOTH"/>
<uses-permission android:name="android.permission.VIBRATE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
android:maxSdkVersion="32"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

- Для повышения безопасности сетевых соединений рекомендуется запретить использование незащищённого HTTP протокола. Для этого в директории проекта `res` создайте файл `network_security_config.xml` с таким содержимым:

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
<base-config cleartextTrafficPermitted="false">
    <trust-anchors>
        <certificates src="system" />
    </trust-anchors>
</base-config>
</network-security-config>
```

После чего подключите этот xml в манифест `AndroidManifest.xml`. Для этого добавьте в секции `application` ссылку на созданный файл:

```
<application
        android:networkSecurityConfig="@xml/network_security_config"
        ... >
</application>
```

4.3. .NET

Для данного фреймворка TrueConf Mobile SDK упакован в NuGet пакет, который можно подключить с помощью менеджера пакетов в Visual Studio. Пакет содержит реализацию для обеих платформ Android и iOS, а также общий интерфейс для кроссплатформенной разработки. Библиотеки собраны на базе **.NET 8** (**net8.0-android** и **net8.0-ios**). В качестве примеров используются кроссплатформенные приложения MAUI.

Минимальная поддерживаемая версия Android - 7.0 (API 24), iOS - 13.0.

Для начала работы с TrueConf Mobile SDK необходимо выполнить установку NuGet пакета с SDK:

1. Правой клавишей мыши кликните по названию проекта.
2. Выберите пункт **Add →NuGet packages**.
3. В открывшемся окне в выпадающем списке перечня источников пакетов выберите пункт **Configure Sources...**.
4. В настройках источников пакетов нажмите кнопку **Add**.
5. В модальном окне добавления источника в пункте **Location** выберите директорию, которая содержит файл `TrueConfSDK.[version].nupkg`. Укажите название источника (например **Mobile SDK**), затем нажмите **Add Source**.
6. Закройте настройки источников пакетов чтобы вернуться в окно добавления пакета.
7. В выпадающем списке источников пакетов в окне добавления пакета выберите добавленный источник (например **Mobile SDK**) - в списке отобразится один пакет **TrueConfSDK**.
8. Выберите пакет **TrueConfSDK** и нажмите **Add Package**.

Основные объекты:

- `ITrueConfSDK` - общий интерфейс;
- `TrueConfAndroidSDKImpl` - реализация интерфейса `ITrueConfSDK` для Android;
- `TrueConfIOSSDKImpl` - реализация интерфейса `ITrueConfSDK` для iOS.

4.3.1. Дополнительные шаги для Android

1. Добавьте в манифест все необходимые для работы SDK разрешения и возможности (см. пункт 3 раздела **Android**).

2. Подключите дополнительные NuGet пакеты:

```
<PackageReference Include="Xamarin.AndroidX.MediaRouter" Version="1.7.0.4" />
<PackageReference Include="Xamarin.AndroidX.GridLayout" Version="1.0.0.27" />
<PackageReference Include="Xamarin.AndroidX.Leanback.Preference" Version="1.0.0.27" />
<PackageReference Include="Xamarin.AndroidX.Work.Runtime" Version="2.9.0.5" />
<PackageReference Include="Xamarin.AndroidX.Emoji.Bundled" Version="1.1.0.22" />
<PackageReference Include="Xamarin.AndroidX.MultiDex" Version="2.0.1.27" />
```

3. Добавьте файл обфускатора ProGuard с таким содержимым:

```
-dontwarn android.content.**
-dontwarn android.app.**
-dontwarn android.hardware.camera2.**
-dontwarn android.util.**
-dontwarn android.provider.**

-keep class androidx.appcompat.** { *; }
-keep class androidx.startup.InitializationProvider
-keep class com.microsoft.maui.** { *; }
-keep class com.trueconf.sdk.** { *; }
```

4.3.2. Дополнительные шаги для iOS

В iOS в файл `Info.plist` необходимо добавить “Privacy - Camera Usage Description” и “Privacy - Microphone Usage Description” с текстом, который будет отображаться при запросе разрешения использования оборудования.

4.3.3. Соответствие функций и событий для .NET

Описание функций и событий находится в основной документации к SDK, однако их названия отличаются:

Функции		События	
iOS, Android	.NET	iOS, Android	.NET
stop	Stop	onServerStatus	OnServerStatusEvent
stop	Stop	onServerStatus	OnServerStatusEvent
loginAs	LoginAs	onLogin	OnLoginEvent
logout	Logout	onLogout	OnLogoutEvent
callTo	CallTo	onStateChanged	OnStateChangedEvent
joinConf	JoinConf	onConferenceStart	OnConferenceStartEvent
hangup	Hangup	onConferenceEnd	OnConferenceEndEvent
acceptCall	AcceptCall	onInvite	OnInviteEvent
parseProtocolLink	ParseProtocolLink	onAccept	OnAcceptEvent
scheduleLoginAs	ScheduleLoginAs	onReject	OnRejectEvent
muteMicrophone	MuteMicrophone	microphoneMuted	MicrophoneMuted
muteCamera	MuteCamera	cameraMuted	CameraMuted
getMyId	GetMyId	onRejectTimeOut	OnRejectTimeOutEvent

getMyName	GetMyName	onUserStatusUpdate	OnUserStatusUpdateEvent
isConnectedToServer	IsConnectedToServer	onChatMessageReceived	OnChatMessageReceivedEvent
isLoggedIn	IsLoggedIn	onRecordRequest	OnRecordRequestEvent
isInConference	IsInConference		
getUserStatus	GetUserStatus		
acceptRecord	AcceptRecord		
sendChatMessage	SendChatMessage		

4.4. Cordova

Для данного фреймворка TrueConf Mobile SDK упакован в Cordova плагин. Минимальная поддерживаемая версия Android - 7.0 (API 24), iOS - 13.0.

Для начала работы создайте проект Cordova, выполнив команду:

```sh
cordova create DIRECTORY_NAME PROJECT_PACKAGE DISPLAYED_PROJECT_NAME
```

Затем перейдите в созданную папку и добавьте плагин TrueConf Mobile SDK командой:

```sh
cordova plugin add PATH_TO_TRUECONF_PLUGIN
```

Никаких дополнительных действий для начала использования TrueConf Mobile SDK не требуется. После добавления плагина в проект можно получить экземпляр SDK в js:

```js
let sdkBuilder = cordova.require('trueconf-sdk-plugin.sdk');
  sdkBuilder.getInstance("qa.trueconf.net").then(instance => {
      sdk = instance; // save the SDK instance for further use
  }, error => {
      console.log(error);
  });
```

А затем подписаться и обработать событие `onStart`:

```js
sdk.addEventListener(“onStart”, (event) => {
   // now you can use the SDK
  });
```

После этого уже можно получать события и работать с плагином SDK для Cordova.

Статическая функция получения экземпляра TrueConf Mobile SDK:

```js
(static) getInstance() → {Promise.<TrueConfSDK>}
```

Возвращает:

```js
// A promise to be called after starting SDK, it gets the SDK instance.
Type - Promise.<TrueConfSDK>
```

Событие, означающие возможность начала работы с TrueConf Mobile SDK:

```js
onStart
```

Содержит параметры:

Имя	Тип	Описание
`connected`	`boolean`	Статус подключения
`serverName`	`string`	Имя сервера или IP, `undefined` - при отсутствии коннекта
`loggedIn`	`boolean`	Статус авторизации
`userID`	`string`	TrueConf ID, пустой если не авторизован

4.4.1. Дополнительные шаги для Android

В `build.gradle` в разделе `repositories` добавьте maven репозиторий, в котором находятся библиотеки Android SDK (как для Android Studio). `username` и `password` выдаются по запросу через вашего менеджера.

```
maven {
    credentials {
        username 'username'
        password 'password'
    }
    url 'https://sdk.trueconf.com/maven/repository/maven-public/'
}
```

До инициализации SDK (до вызова метода `start`) вызовите следующие методы:

- `registerApp` – в него нужно передать `Application` (или его наследника, который используется в проекте);
- `setFallbackActivity` – в него нужно передать класс `Activity`, к которому следует возвращаться в случае завершения звонка.

Это можно сделать, к примеру, в классе `MainActivity` в `onCreate`:

```js
public class MainActivity extends CordovaActivity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate();
        TrueConfSDK.getInstance().registerApp(getApplication());
        TrueConfSDK.getInstance().setFallbackActivity(MainActivity.class);
       ...
```

4.5. React Native

Для данного фреймворка TrueConf Mobile SDK упакован в npm пакет, который можно добавить в проект с

помощью **npm** или **yarn**.

Минимальная поддерживаемая версия Android - 7.0 (API 24), iOS - 13.0.

Для создания проекта выполните команду:

```sh
npx react-native init PROJECT_NAME
```

После добавьте модуль SDK с помощью **npm** или **yarn**:

```sh
npm install PATH_TO_TRUECONF_MODULE --install-links=true
```

или

```sh
yarn add PATH_TO_TRUECONF_MODULE
```

4.5.1. Дополнительные шаги для iOS

Для подключения модуля SDK через CocoaPods из папки проекта выполните команду:

```sh
cd ios && pod install && cd ..
```

Добавьте в `Info.plist` разрешения на использование камеры и микрофона непосредственно в Xcode проекте, либо выполнив следующие команды из папки проекта:

```sh
cd ios/PROJECT_NAME
plutil -insert NSCameraUsageDescription -string '' Info.plist
plutil -insert NSMicrophoneUsageDescription -string '' Info.plist
```

4.5.2. Дополнительные шаги для Android

В `build.gradle` (в `Project:Example`) для всех проектов в разделе `repositories` добавьте maven репозиторий, в котором находятся библиотеки Android SDK (как для Android Studio). `username` и `password` выдаются по запросу через вашего менеджера.

```
maven {
    credentials {
        username 'username'
        password 'password'
    }
    url 'https://sdk.trueconf.com/maven/repository/maven-public/'
}
```

До инициализации SDK (до вызова метода `start`) необходимо вызвать следующие методы:

- `registerApp` - в него нужно передать `Application` (или его наследника, который используется в проекте);
- `setFallbackActivity` - в него нужно передать класс `Activity`, к которому следует возвращаться в случае завершения звонка.

Это можно сделать, к примеру, в классе `MainApplication` в `onCreate`:

```js
public class MainApplication extends Application implements ReactApplication {
    @Override
    public void onCreate() {
        super.onCreate();
        TrueConfSDK.getInstance().registerApp(this);
        TrueConfSDK.getInstance().setFallbackActivity(MainActivity.class);
       ...
```

5. Основные типы данных и перечисления

5.1. Статусы

Список статусов пользователей, в том числе самого экземпляра SDK, одинаковый на всех платформах для разработки, но названия констант в перечислении могут отличаться.

5.1.1. Для Android Studio

При разработке на Android Studio доступны следующие статусы, которые предоставляются перечислением `PresenceStatus`:

- `UNDEFINED` (-127) — информация о статусе пользователя отсутствует
- `INVALID` (-1) — серверу неизвестен статус пользователя
- `LOGOFF` (0) — пользователь не подключен к серверу
- `ONLINE` (1) — пользователь авторизован на сервере и доступен для звонка
- `BUSY` (2) — пользователь находится в звонке или конференции
- `MULTIHOST` (5) — пользователь находится в конференции и является её владельцем. Разные типы существуют для поддержки разных видов конференции. Проверку можно делать так: `status >= BUSY`.

6. Функции Android

Далее идёт список функций, которые применяются при разработке на IDE Android Studio.

6.1. Функции экземпляра SDK

Вызываются на уровне экземпляра TrueConf Mobile SDK, получаемого функцией `TrueConfSDK.getInstance()`. Например, `TrueConfSDK.getInstance().registerApp(this)`.

6.1.1. registerApp

Записывает класс-наследник от `Application`. Этот метод нужно обязательно вызвать до `start`.

Параметры:

- `application` – класс `Application` (`Class`).

6.1.2. addTrueconfListener

Подключение слушателя.

Параметры:

- `listener` – класс слушатель (`TrueconfListener`).

6.1.3. removeTrueconfListener

Отключение слушателя.

Параметры:

- `listener` – класс слушатель (`TrueconfListener`).

6.1.4. setFallbackActivity

Записывает класс `Activity`, к которому нужно вернуться в случае завершения звонка. Этот метод нужно вызвать обязательно.

Параметры:

- `activity` – класс `Activity` (`Class`).

6.1.5. start

Запуск SDK.

Параметры:

- `serverList` – список серверов (`String`) (необязательный параметр);
- `checkPermissions` – проверять ли разрешения при необходимости (`boolean`).

6.1.6. stop

Остановка работы экземпляра SDK и освобождение ресурсов.

6.1.7. isStarted

Проверка состояния работы экземпляра SDK, результат типа `boolean`.

6.2. IServerManager

Данные функции вызываются на уровне интерфейса `IServerManager`, который получаем вызовом функции `TrueConfSDK.getServerManager()`.

6.2.1. isConnectedToServer

Проверка соединения с сервером.

Возвращаемое значение (булевое) – `true`, если есть соединение с сервером, `false` – соединение отсутствует.

6.2.2. isLoggedIn

Проверка состояния авторизации.

Возвращаемое значение (булевое) – `true`, пользователь авторизован на сервере, `false` – пользователь не авторизован.

6.2.3. loginAs

Авторизация под определённым пользователем с указанными параметрами на сервере.

Параметры:

- `user` – идентификатор пользователя (строка);
- `pwd` – пароль (строка);
- `encryptPassword` – `true`, если пароль передается в открытом виде (и SDK требуется его зашифровать), `false`, если он уже в зашифрованном виде (булевая);
- `enableAutoLogin` – нужно ли автоматически авторизовать пользователя этим идентификатором при повторном запуске (булевая).

Возвращаемое значение (булевое) – `true`, если запрос на авторизации отправлен на сервер, `false` – если авторизацию невозможно произвести.

6.2.4. scheduleLoginAs

Получение списка операций, которые должны осуществиться последовательно, в том числе сервер для присоединения, идентификационные данные аккаунта и адресата звонка. Является эквивалентом функции `parseProtocolLink`, в которой параметры передаются не в виде строки протокола, а по отдельности. Строковые параметры могут быть пустыми.

Параметры:

- `login` – идентификатор пользователя (строка);
- `pwd` – пароль (строка);
- `encryptPassword` – `true`, если пароль передается в открытом виде (и SDK требуется его зашифровать), `false`, если он уже в зашифрованном виде (булевая);
- `callToUser` – идентификатор вызываемого абонента (строка);
- `autoClose` – нужно ли закрывать сеанс работы с сервером после выполнения звонка или прерывания выполнения (булевая);
- `loginTemp` – признак временного логина. Означает, что после звонка нужно деавторизовать клиент (булевая);
- `loginForce` – принудительный логин. Авторизация будет произведена даже в том случае, если клиент уже авторизован на сервере (булевая);
- `domain` – домен, в котором будет осуществляться автоматический поиск сервера (строка);
- `serversList` – список серверов, к которым необходимо подключиться (строка);
- `isPublic` – параметр, указывающий, что callToUser является именем конференции, а не именем пользователя. При неверном значении этого параметра звонок пользователю или подключение к конференции не выполнится (булевая).

6.2.5. logout

Деавторизация (выход из системы) аккаунта без отключения SDK от сервера видеосвязи.

Возвращаемое значение (булевое) – `true`, если запрос на деавторизации отправлен на сервер, `false` – если деавторизацию невозможно произвести.

6.3. IConferenceManager

Данные функции вызываются на уровне интерфейса `IConferenceManager`, который получаем вызовом

функции `TrueConfSDK.getConferenceManager()`.

6.3.1. isInConference

Проверка нахождения в конференции.

Возвращаемое значение (булевое) – `true`, клиент находится в конференции, `false` – клиент вне конференции.

6.3.2. callTo

Звонок указанном абоненту.

Параметры:

- `user` – TrueConf ID вызываемого абонента (строка).

Возвращаемое значение (булевое) – `true`, если звонок отправлен на сервер, `false` – если звонок невозможно произвести.

6.3.3. joinConf

Подключение к групповой конференции.

Параметры:

- `conf_ID` – идентификатор конференции (строка).

Возвращаемое значение (булевое) – `true`, если звонок отправлен на сервер, `false` – если звонок невозможно произвести.

6.3.4. hangup

Завершение текущего звонка или конференции.

Параметры:

- `forAll` (опциональный) – в случае конференции завершать ли её для всех участников при наличии полномочий (булевое). По умолчанию `true`.

Возвращаемое значение (булевое) – `true`, если звонок завершить возможно, `false` – если звонок завершить невозможно (обычно по причине неверного состояния).

6.3.5. acceptCall

Ответ на входящий звонок.

Параметры:

- `accept` – принять или отклонить вызов (булевое).

Возвращаемое значение (булевое) – `true`, если ответ на запрос можно произвести, `false` – если ответ невозможен.

6.3.6. acceptRecord

Ответ на входящий запрос на видеозапись.

Параметры:

- `accept` – принять или отклонить запрос на видеозапись (булевое);
- `userID` – TrueConf ID пользователя, которому даётся ответ на запрос (строка).

6.3.7. returnToCall

Возврат к activity конференции.

Параметры:

- `currentContext` – текущий контекст (`Context`).

6.3.8. sendPincode

Отправка пин-кода для доступа к конференции с пин-кодом.

Параметры:

- `confId` – ID конференции (`String`);
- `pin` - пин-код (`String`).

6.4. IVideoDeviceController

Данные функции вызываются на уровне интерфейса `IVideoDeviceController` , который получаем вызовом функции `TrueConfSDK.getVideoDeviceController()` .

6.4.1. isCameraMuted

Проверка состояния камеры.

Возвращаемое значение (булевое) – `true` , когда камера отключена, `false` – камера включена.

6.4.2. muteCamera

Изменение состояния камеры.

Параметры:

- `mute` – состояние, в которое требуется перевести камеру: `true` – камера выключена, `false` – камера включена (булевая).

6.4.3. isCameraEnabledByDefault

Возвращает статус камеры по умолчанию (`boolean`)

6.4.4. setDefaultCameraEnabled

Установка состояния камеры по умолчанию.

Параметры:

- `isEnabled` – `true` включает захват видео, `false` выключает (`boolean`).

6.5. IAudioDeviceController

Данные функции вызываются на уровне интерфейса `IAudioDeviceController` , который получаем вызовом функции `TrueConfSDK.getAudioDeviceController()` .

6.5.1. muteMicrophone

Изменение состояния микрофона.

Параметры:

- `mute` – состояние, в которое требуется перевести микрофон: `true` – микрофон выключен, `false` – микрофон включен (булевая).

6.5.2. isMicrophoneMuted

Проверка состояния микрофона.

Возвращаемое значение (булевое) – `true` , когда микрофон отключен, `false` – микрофон включен.

6.5.3. muteSpeaker

Включает либо отключает звук из динамиков.

Параметры:

- `mute` – `true` выключает звук, `false` включает (`boolean`).

6.5.4. requestAudioState

Запрос текущего состояния аудио устройств. В свою очередь вызывает метод onAudioDeviceResponse из AudioDeviceCallback.

6.5.5. isMicEnabledByDefault

Возвращает статус микрофона по умолчанию (`boolean`).

6.5.6. setDefaultMicEnabled

Установка состояния микрофона по умолчанию.

Параметры:

- `isEnabled` – `true` включает захват звука, `false` выключает (`boolean`).

6.5.7. isSpeakerEnabledByDefault

Возвращает статус динамика по умолчанию (`boolean`).

6.5.8. setDefaultSpeakerEnabled

Установка состояния динамика по умолчанию.

Параметры:

- `isEnabled` – `true` включает вывод звука, `false` выключает (`boolean`).

6.5.9. setDefaultAudioDevice

Установка устройства вывода звука по умолчанию.

Параметры:

- `audioDeviceInfo` – устройство вывода (`AudioDeviceInfo`).

6.5.10. changeAudioDevice

Меняет устройство вывода звука во время конференции.

Параметры:

- `audioDeviceInfo` – устройство вывода (`AudioDeviceInfo`).

6.6. IChatManager

Данные функции вызываются на уровне интерфейса `IChatManager` , который получаем вызовом функции `TrueConfSDK.getChatManager()` .

6.6.1. sendChatMessage

Отправка текстового сообщения.

Параметры:

- `toID` – (строка) TrueConf ID пользователя, которому отправляется сообщение (рекомендуется использовать полный ID вида `user@server.name`). Для отправки сообщения в чат текущей групповой конференции этот параметр должен быть пустым.
- `text` – (строка) текст сообщения.

6.7. IContactsManager

Данные функции вызываются на уровне интерфейса `IContactsManager` , который получаем вызовом функции `TrueConfSDK.getContactsManager()` .

6.7.1. getMyId

Получение собственного идентификатора.

Возвращаемое значение (строка) – идентификатор текущего пользователя в системе. Если пользователь не авторизован на сервере, возвращается Nil.

6.7.2. getMyName

Получение собственного имени для показа в интерфейсе.

Возвращаемое значение (строка) – имя текущего пользователя. Содержит Nil если пользователь не авторизован на сервере, или строку, равную идентификатору пользователя, если пользователь не имеет собственного имени.

6.7.3. getUserStatus

Получение статуса другого пользователя. Если статус известен, он возвращается немедленно. Если нет – статус запрашивается на сервере, а клиент подписывается на получение уведомлений о его изменении.

Параметры:

- `user` – TrueConf ID пользователя, статус которого запрашивается (строка).

Возвращаемое значение (`UserPresStatus`) – текущий статус пользователя.

6.7.4. getUsers

Получение списка абонентов из адресной книги пользователя, под которым авторизованы в SDK. Возвращает массив объектов типа `ContactInfo` , каждый объект содержит `userId` (String, TrueConf ID пользователя) и его текущий статус (`PresenceStatus`).

6.8. IVisicallManager

Данные функции вызываются на уровне интерфейса `IVisicallManager` , который получаем вызовом функции `TrueConfSDK.getVisicallManager()` .

6.8.1. parseProtocolLink

Функция получает в виде строки команду, содержащую указания каким аккаунтом авторизоваться и какой звонок осуществить. После чего автоматически выполняет все эти операции. В случае, если на какой-то стадии происходит остановка выполнения команды.

Параметры:

- `cmd` – обрабатываемая строка (строка).

6.9. IExtraButtonController

Данные функции вызываются на уровне интерфейса `IExtraButtonController` , который получаем вызовом функции `TrueConfSDK.getExtraButtonController()` .

6.9.1. setNewExtraButtons

Добавить дополнительные кнопки в панель управления конференцией. Кнопки будут добавлены в порядке расположения их в переданном массиве в список, который открывается по тапу на кнопку "троеточия" (правая кнопка в панели). Смотрите Пример 5.

Параметры:

- `btns` – массив из объектов типа `TCExtraButton` .

6.10. ICallScreenController

Данные функции вызываются на уровне интерфейса `ICallScreenController` , который получаем вызовом функции `TrueConfSDK.getCallScreenController()` .

6.10.1. setReciveCallFragment

Переопределить экран входящего вызова.

Параметры:

- `fragment` – кастомный фрагмент входящего вызова (`Fragment`)

6.10.2. setPlaceCallFragment

Переопределить экран исходящего вызова.

Параметры:

- `fragment` – кастомный фрагмент исходящего вызова (`Fragment`)

6.10.3. setConferenceFragment

Переопределить экран конференции.

Параметры:

- `fragment` – кастомный фрагмент конференции (`Fragment`)

7. События для Android

Данные события вызываются на уровне интерфейса `TrueConfListener` (см. Пример 7). Он в свою очередь содержит указанные ниже разные интерфейсы, которые при разработке требуется имплементировать в своём классе.

7.1. ServerStatusEventsCallback

7.1.1. onServerStatus

Событие, вызываемое при присоединении/отсоединении сервера или ошибке при попытке соединения.

Параметры:

- `connected` – есть ли соединение с сервером (boolean);
- `serverName` – имя текущего сервера (строковое);
- `serverPort` – номер порта используемый для соединения с сервером (числовое).

7.1.2. onStateChanged

Событие, вызываемое при изменении собственного статуса пользователя. Текущий статус можно получить через функции запроса текущего состояния.

7.2. LoginEventsCallback

7.2.1. onLogin

Событие, вызываемое при авторизации или ошибке авторизации на сервере.

Параметры:

- `loggedIn` – авторизован ли пользователь на сервере (boolean);
- `userId` – TrueConf ID пользователя (строка).

7.2.2. onLogout

Событие, вызываемое при деавторизации на сервере.

7.3. ConferenceEventsCallback

7.3.1. onConferenceStart

Событие, вызываемое при начале конференции.

7.3.2. onConferenceEnd

Событие, вызываемое при завершении конференции.

7.3.3. onInvite

Событие, вызываемое при поступлении входящего звонка.

Параметры:

- `userId` – TrueConf ID вызывающего пользователя (строка);
- `userName` – имя вызывающего пользователя (строка).

7.3.4. onAccept

Событие, вызываемое при приеме вызываемым абонентом звонка.

Параметры:

- `userId` – TrueConf ID вызываемого пользователя (строка);
- `userName` – имя вызываемого пользователя (строка).

7.3.5. onReject

Событие, вызываемое при отклонении вызываемым абонентом звонка.

Параметры:

- `userId` – TrueConf ID вызываемого пользователя (строка);
- `userName` – имя вызываемого пользователя (строка).

7.3.6. onRejectTimeout

Событие, вызываемое при отсутствии реакции вызываемого абонента в течении определенного времени.

Параметры:

- `userId` – TrueConf ID вызываемого пользователя (строка);
- `userName` – имя вызываемого пользователя (строка).

7.3.7. onRecordRequest

Событие, вызываемое при получении запроса на видеозапись.

Параметры:

- `userID` – TrueConf ID пользователя, который запрашивает видеозапись (строка);
- `userName` – имя пользователя, который запрашивает видеозапись (строка).

7.3.8. onConferencePasswordRequired

Вызывается при запросе ПИН-кода во время подключения к конференции с ПИН-кодом.

Параметры:

- `confId` – ID конференции (`String`)

7.3.9. onConferenceWrongPassword

Вызывается при отправке неверного ПИН-кода при попытке подключения к конференции с ПИН-кодом.

Параметры:

- `confId` – ID конференции (`String`)

7.4. ChatEventsCallback

7.4.1. onChatMessageReceived

Событие, вызываемое при получении текстового сообщения.

Параметры:

- `fromID` – идентификатор пользователя, который отправлял сообщение (строка);
- `fromName` – имя пользователя, который отправлял сообщение (строка);
- `text` – текст сообщения (строка);
- `toID` – идентификатор пользователя, которому отправлялось сообщение (строка).

7.5. UserStatusEventsCallback

7.5.1. onUserStatusUpdate

Событие, вызываемое при изменении статуса другого пользователя.

Параметры:

- `userID` – идентификатор пользователя с изменившимся статусом (строка);
- `state` – новый статус пользователя (`UserPresStatus`).

7.5.2. onContactListUpdate

Вызывается при загрузке списка контактов и их статусов после авторизации пользователем на сервере.

7.6. AudioDeviceCallback

7.6.1. onAudioDeviceChanged

Вызывается при изменении устройства вывода во время конференции.

Параметры:

- `playerMute` – состояние динамика (вкл/выкл) (boolean)
- `pair` – устройство вывода (`AudioDeviceInfo`).

7.6.2. onAudioDeviceUpdate

Вызывается при изменении состояния микрофона или динамика.

Параметры:

- `playerMute` – состояние динамика (вкл/выкл) (boolean)
- `recorderMute` – состояние микрофона (вкл/выкл) (boolean)
- `pair` – устройство вывода (`AudioDeviceInfo`)

7.6.3. onAudioDeviceResponse

Возвращает текущее состояние аудио устройств по запросу requestAudioState.

Параметры:

- `playerMute` – состояние динамика (вкл/выкл) (boolean)
- `recorderMute` – состояние микрофона (вкл/выкл) (boolean)
- `active` – текущее устройство вывода (AudioDeviceInfo)
- `pairs` – список доступных устройств вывода (`List<AudioDeviceInfo>`)

7.7. VideoDeviceCallback

7.7.1. onVideoDeviceUpdate

Вызывается при изменении состояния камеры.

Параметры:

- `videoEnabled` – состояние камеры (вкл/выкл) (boolean)

7.8. LayoutCallback

7.8.1. onCalculateCustomLayouts

Метод для изменения координат и размеров видеоокон участников конференции. Смотрите Пример 7.

7.8.2. onLayoutApplied

Вызывается, когда страница завершает анимацию и становится активной.

8. Изменения между версиями SDK для Android

8.1. Что изменилось в версии 3.0.0 по сравнению с 2.2.0

1. Методы теперь сгруппированы по разным интерфейсам:

- IServerManager
- IConferenceManager
- IChatManager
- IContactsManager
- IVisicallManager
- IVideoDeviceController
- IAudioDeviceController
- IExtraButtonController
- ICallScreenController

Например, было:

```java
TrueConfSDK.getInstance().joinConf(confId);
TrueConfSDK.getInstance().logout();
```

Стало:

```java
TrueConfSDK.getConferenceManager().joinConf(confId);
TrueConfSDK.getServerManager().logout();
```

2. Внесены существенные изменения в управление звуком:

- удалены методы `isSpeakerMuted`, `getAudioDevices`, `onAudioPairChanged`;
- для получения актуального списка устройств вывода звука необходимо выполнить запрос `requestAudioState` и реализовать метод `onAudioDeviceResponse` из интерфейса `AudioDeviceCallback`. Также данный callback содержит методы `onAudioDeviceChanged` и `onAudioDeviceUpdate`;
- для изменения устройства вывода по умолчанию сначала необходимо выполнить запрос `requestAudioState` и только после этого установить устройство с помощью метода `setDefaultAudioDevice`;
- во фрагментах `ConferenceFragment`, `IncomingCallFragment` и `PlaceCallFragment` удалены методы `onSwitchMicApplied`, `onMuteSpeakerApplied`. Вместо них необходимо реализовать интерфейс `AudioDeviceCallback`;
- методы `muteMicrophone`, `muteSpeaker`, `changeAudioDevice` должны использоваться только во время конференции, иначе они не сработают из-за проверки.

3. Изменения в управлении видео:

- добавлен метод `onVideoDeviceUpdate` для уведомления об изменении состояния камеры;
- во фрагментах `ConferenceFragment`, `IncomingCallFragment` и `PlaceCallFragment` удалён метод `onSwitchCameraApplied`
- метод `muteCamera` должен использоваться только во время конференции, иначе он не сработает из-за проверки.

4. Изменены пути до некоторых важных классов, например класс кастомной кнопки, а также `TrueConfListener` и `CallCast`.

Было:


```java
com.trueconf.sdk.data.TCSDKExtraButton
com.trueconf.sdk.interfaces.TrueConfListener
com.trueconf.sdk.gui.activities.CallCast
```

Стало:

```java
com.trueconf.sdk.presentation.views.TCExtraButton
com.trueconf.sdk.TrueConfListener
com.trueconf.sdk.presentation.activities.CallCast
```

5. Метод `onStateChanged` получил параметр `FSM.State newState`, который возвращает новый собственный статус (`userOffline`, `userOnline`, `userBusy` и т.д.). Список статусов представлен в таблице в разделе “Типы. UserPresStatus”.

6. Изменены некоторые названия иконок для кастомизации:

- вместо `ic_selfie_icon` теперь используется `ic_rotate`. Также на собственном видео добавлена иконка изменения размера `ic_minimize_fullscreen`;
- иконки `shape_circle_background_red_pressed` и `shape_circle_background_red` теперь не используются;
- добавлена иконка `conf_button_back` для изменения заднего фона кнопок.

Полный список смотреть в 5 примере "Кастомизация интерфейса".

7. Библиотеки собраны при использовании Java 17 (было Java 11).

8. В составе SDK обновлены библиотеки OpenSSL 1.1.1w и SQLite 3.47.0.

9. Поддержано перетягивание собственного видео на экране конференции.

10. Изменены версии некоторых зависимостей, необходимых для работы SDK. Полный список перечислен в разделе интеграции с Android Studio.

8.2. Что изменилось в версии 2.2.0 по сравнению с 1.3.3

1. Теперь Android SDK распространяется через maven репозиторий с доступом по логину и паролю, которые выдаются по запросу через вашего менеджера. Необходимо в файле gradle в разделе repositories добавить maven репозиторий следующим образом:

```
maven {
    credentials {
        username 'username'
        password 'password'
    }
    url 'https://sdk.trueconf.com/maven/repository/maven-public/'
}
```

2. Чтобы подключить SDK, необходимо в **dependencies** в файле настроек `.gradle` добавить 3 библиотеки:

```
api 'com.trueconf:trueconfsdk:2.2.0.101@aar'
api 'com.trueconf:media:2.2.0.101@aar'
api 'com.trueconf:jnicore:2.2.0.101@aar'
```

3. Минимальная версия Android теперь 7.0 (minSdkVersion 24).

4. Подключение SDK к классам проекта производится следующими строками:

```java
import com.trueconf.sdk.TrueConfSDK;// to work with SDK methods
import com.trueconf.sdk.TrueConfListener;    // to work with SDK events
```

5. Перед вызовом метода `start` необходимо вызвать метод `registerApp` и передать в него дочерний класс от `Application`, например:

```java
public class TestApp extends Application {
    @Override
    public void onCreate() {
        super.onCreate();
        TrueConfSDK.getInstance().registerApp(this);
        TrueConfSDK.getInstance().start("server.name", true);
       ...
    }
}
```

6. Необходимо вызвать метод `setFallbackActivity`, в котором нужно указать класс `Activity`, к которому следует возвращаться в случае завершения звонка, например:

```java
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TrueConfSDK.getInstance().setFallbackActivity(MainActivity.class);
   ...
    }
}
```

7. В файл манифеста теперь не нужно подключать activity `com.vc.gui.activities.Call`, `com.vc.gui.activities.PermissionActivity` и `service com.vc.service.ExternalSchemeHelperService`.

8. Расширены возможности кастомизации: добавлена возможность размещения собственного видео и видео участников конференции в отдельном Fragment, а также изменение размеров и координат видеоокон участников конференции. Подробности описаны в документации к 7 примеру.

9. Некоторые изменения в методах, а именно:

- в методе `start` убран параметр `Context`;
- добавлен метод `registerApp`, в котором нужно указать дочерний класс от `Application`;
- добавлен метод `setFallbackActivity`, в котором нужно указать класс `Activity`, к которому следует возвращаться в случае завершения звонка;
- добавлены методы для управления динамиками (`muteSpeaker`, `isSpeakerMuted`, `setDefaultSpeakerEnabled`, `getAudioDevices`, `getCurrentAudioDevice`, `setAudioDevice`);
- `onContactListUpdate` - событие в `TrueConfListener.UserStatusEventsCallback`, которое приходит при загрузке списка контактов и их статусов после авторизации пользователем на сервере;
- добавлен новый интерфейс - `TrueConfListener.AudioDeviceCallback`, который содержит 2 метода - `onAudioPairChanged` (вызывается при изменении устройства вывода) и `onAudioDeviceListChanged` (вызывается при изменении списка устройств);
- добавлены методы `setDefaultAudioEnabled` и `setDefaultCameraEnabled` для установления дефолтного состояния микрофона и камеры соответственно;
- методы `microphoneMuted` и `cameraMuted` переименованы в `isMicrophoneMuted` и `isCameraMuted`

соответственно;

- убран флаг `trueConfSDKLogEnable`, который включал расширенные логи SDK (теперь они пишутся всегда). Также убраны методы `startSavingLogs` и `stopSavingLogs`, т.к. логи автоматически собираются в подкаталог `./files/logs`.

9. Функции iOS

Далее идёт список функций, которые применяются при разработке на IDE Xcode под iOS.

9.1. initWithViewController

Функция инициализации объекта SDK.

Параметры:

- `vc` – используемый для инициализации вьюконтроллер (`UIViewController`);
- `serverIP` – сервер и порт, к которому нужно присоединиться в виде `<server address>:<server port>` (`NSString`);
- `directConnection` (опциональный) – использование прямого соединения (`BOOL`). По умолчанию `YES` .

9.2. start

Запуск SDK после инициализации.

9.3. startWithServersList

Запуск SDK после инициализации со списком серверов. Происходит перебор адресов из переданного списка и подключение к первому рабочему серверу.

Параметры:

- `serversList` – список адресов серверов, разделённых запятой (`NSString`).

9.4. isStarted

Проверка состояния работы экземпляра SDK, результат типа `boolean` .

9.5. stop

Остановка работы экземпляра SDK и освобождение ресурсов.

9.6. loginAs

Авторизация под определённым пользователем с указанными параметрами на сервере.

Параметры:

- `user` – идентификатор пользователя (строка);
- `pwd` – пароль (строка);
- `encryptPassword` – `true` , если пароль передается в открытом виде (и SDK требуется его зашифровать), `false` , если он уже в зашифрованном виде (булевая);
- `enableAutoLogin` – нужно ли автоматически авторизовать пользователя этим идентификатором при повторном запуске (булевая).

Возвращаемое значение (булевое) – `true` , если запрос на авторизации отправлен на сервер, `false` – если авторизацию невозможно произвести.

9.7. logout

Деавторизация (выход из системы) аккаунта без отключения SDK от сервера видеосвязи.

Возвращаемое значение (булевое) – `true` , если запрос на деавторизации отправлен на сервер, `false` – если деавторизацию невозможно произвести.

9.8. callTo

Звонок указанном абоненту.

Параметры:

- `user` – TrueConf ID вызываемого абонента (строка).

Возвращаемое значение (булевое) – `true`, если звонок отправлен на сервер, `false` – если звонок невозможно произвести.

9.9. joinConf

Подключение к групповой конференции.

Параметры:

- `conf_ID` – идентификатор конференции (строка).

Возвращаемое значение (булевое) – `true`, если звонок отправлен на сервер, `false` – если звонок невозможно произвести.

9.10. hangup

Завершение текущего звонка или конференции.

Параметры:

- `forAll` (опциональный) – в случае конференции завершать ли её для всех участников при наличии полномочий (булевое). По умолчанию `true`.

Возвращаемое значение (булевое) – `true`, если звонок завершить возможно, `false` – если звонок завершить невозможно (обычно по причине неверного состояния).

9.11. acceptCall

Ответ на входящий звонок.

Параметры:

- `accept` – принять или отклонить вызов (булевое).

Возвращаемое значение (булевое) – `true`, если ответ на запрос можно произвести, `false` – если ответ невозможен.

9.12. parseProtocolLink

Функция получает в виде строки команду, содержащую указания каким аккаунтом авторизоваться и какой звонок осуществить. После чего автоматически выполняет все эти операции. В случае, если на какой-то стадии происходит остановка выполнения команды.

Параметры:

- `cmd` – обрабатываемая строка (строка).

9.13. scheduleLoginAs

Получение списка операций, которые должны осуществиться последовательно, в том числе сервер для присоединения, идентификационные данные аккаунта и адресата звонка. Является эквивалентом функции `parseProtocolLink`, в которой параметры передаются не в виде строки протокола, а по отдельности. Строковые параметры могут быть пустыми.

Параметры:

- `login` – идентификатор пользователя (строка);
- `pwd` – пароль (строка);
- `encryptPassword` – `true`, если пароль передается в открытом виде (и SDK требуется его зашифровать), `false`, если он уже в зашифрованном виде (булевая);
- `callToUser` – идентификатор вызываемого абонента (строка);
- `autoClose` – нужно ли закрывать сеанс работы с сервером после выполнения звонка или прерывания выполнения (булевая);
- `loginTemp` – признак временного логина. Означает, что после звонка нужно деавторизовать клиент (булевая);

- `loginForce` – принудительный логин. Авторизация будет произведена даже в том случае, если клиент уже авторизован на сервере (булевая);
- `domain` – домен, в котором будет осуществляться автоматический поиск сервера (строка);
- `serversList` – список серверов, к которым необходимо подключиться (строка);
- `isPublic` – параметр, указывающий, что `callToUser` является именем конференции, а не именем пользователя. При неверном значении этого параметра звонок пользователю или подключение к конференции не выполнится (булевая).

9.14. muteMicrophone

Изменение состояния микрофона.

Параметры:

- `mute` – состояние, в которое требуется перевести микрофон: `true` – микрофон выключен, `false` – микрофон включен (булевая).

9.15. muteCamera

Изменение состояния камеры.

Параметры:

- `mute` – состояние, в которое требуется перевести камеру: `true` – камера выключена, `false` – камера включена (булевая).

9.16. getMyId

Получение собственного идентификатора.

Возвращаемое значение (строка) – идентификатор текущего пользователя в системе. Если пользователь не авторизован на сервере, возвращается Nil.

9.17. getMyName

Получение собственного имени для показа в интерфейсе.

Возвращаемое значение (строка) – имя текущего пользователя. Содержит Nil если пользователь не авторизован на сервере.

9.18. isConnectedToServer

Проверка соединения с сервером.

Возвращаемое значение (булевое) – `true` , если есть соединение с сервером, `false` – соединение отсутствует.

9.19. isLoggedIn

Проверка состояния авторизации.

Возвращаемое значение (булевое) – `true` , пользователь авторизован на сервере, `false` – пользователь не авторизован.

9.20. isInConference

Проверка нахождения в конференции.

Возвращаемое значение (булевое) – `true` , клиент находится в конференции, `false` – клиент вне конференции.

9.21. getUserStatus

Получение статуса другого пользователя. Если статус известен, он возвращается немедленно. Если нет – статус запрашивается на сервере, а клиент подписывается на получение уведомлений о его изменении.

Параметры:

- `user` – TrueConf ID пользователя, статус которого запрашивается (строка).

Возвращаемое значение (`UserPresStatus`) – текущий статус пользователя.

9.22. microphoneMuted

Проверка состояния микрофона.

Возвращаемое значение (булевое) – `true` , когда микрофон отключен, `false` – микрофон включен.

9.23. cameraMuted

Проверка состояния камеры.

Возвращаемое значение (булевое) – `true` , когда камера отключена, `false` – камера включена.

9.24. acceptRecord

Ответ на входящий запрос на видеозапись.

Параметры:

- `accept` – принять или отклонить запрос на видеозапись (булевое);
- `userID` – TrueConf ID пользователя, которому даётся ответ на запрос (строка).

9.25. sendChatMessage

Отправка текстового сообщения.

Параметры:

- `toID` – (строка) TrueConf ID пользователя, которому отправляется сообщение (рекомендуется использовать полный ID вида `user@server.name`). Для отправки сообщения в чат текущей групповой конференции этот параметр должен быть пустым.
- `text` – (строка) текст сообщения.

Возвращаемое значение (булевое) – `true` , когда сообщение отправлено на сервер, `false` – если сообщение отправить не удалось из-за отсутствия соединения с сервером.

9.26. setInitViewController

Вызывается перед запуском интерфейса звонка. Позволяет переопределить базовый вьюконтроллер в SDK. Имеет приоритет перед вьюконтроллером заданным при инициализации SDK.

Параметры:

- `aInitViewController` – вьюконтроллер, который нужно использовать (`UIViewController`).

9.27. setNewExtraButtons

Добавить дополнительные кнопки в панель управления конференцией. Кнопки будут добавлены в список, который открывается по тапу на кнопку троеточия (правая кнопка в панели) в порядке расположения их в переданном массиве. Смотрите Пример 5.

Параметры:

- `btns` – массив из объектов типа `UIAlertAction` (`NSArray`).

9.28. orientationWillChangeTo

Должна вызываться при начале смены ориентации интерфейса. Ретрансляция события главного контроллера приложения.

Параметры:

- `toOrientation` – новая ориентация интерфейса (`UIInterfaceOrientation`).

9.29. orientationDidChangeTo

Должна вызываться при окончании смены ориентации интерфейса. Ретрансляция события главного

контроллера приложения.

Параметры:

- `toOrientation` – новая ориентация интерфейса (`UIInterfaceOrientation`)

9.30. trueConfSDKLogEnable

Свойство типа `BOOL` , которое включает либо отключает расширенное логирование. По умолчанию `NO` .

9.31. getUserName

Функция возвращает отображаемое имя другого пользователя.

Параметры:

- `user` – TrueConf ID пользователя, имя которого запрашивается (строка).

Возвращаемое значение (строка) – имя пользователя.

10. Изменения между версиями SDK для iOS/iPadOS

10.1. Версия 3.4.3

Что изменилось в версии 3.4.3 по сравнению с 3.2.6:

- убраны свойства `hideSelfViewButtonsInSmallSize` и `muteButtonVisible` по причине неактуальности;
- функция `setNewExtraButtons` в качестве параметра теперь принимает массив `UIAlertAction` вместо `TCSDKExtraButton`. Кастомные кнопки теперь добавляются в список, открывающийся по кнопке “троеточие" (см. Пример 5);
- из инициализатора `initWithViewController` убран параметр `confCustomControlsImages` по причине неактуальности. Для замены картинок стандартных кнопок управления конференцией теперь достаточно в `Assets` добавить картинки с соответствующими именами (см. Пример 5).

11. Примеры для Android

Несколько примеров для IDE Android Studio при разработке под Android. Скачать их вы можете на нашем GitHub.

11.1. Пример 1. Демонстрация основных возможностей SDK

Одностраничное приложение где реализованы все основные функции TrueConf Mobile SDK:

1. Инициализация и подключение к серверу (`start`).
2. Ручной (по нажатию кнопки) логин на указанном сервере под учётной записью пользователя и ручной выход из аккаунта.
3. Звонок абоненту по его TrueConf ID (`callTo`).
4. Возможность принимать входящие видеозвонки.

В классе `TestApp` в `onCreate()` мы сначала вызываем метод `registerApp()` , в который передаём дочерний класс от `Application` , а затем запускаем SDK методом `start()` .

В `MainActivity` в `onCreate()` мы вызываем метод `setFallbackActivity()` с указываем класса `Activity` , к которому нужно вернуться в случае завершения звонка.

В `PlaceholderFragment` мы имплементируем интерфейс `TrueconfListener` и переопределяем его функции для приема `callback` .

11.2. Пример 2. Демонстрация работы с trueconf-ссылками

Возможность подключиться к серверу, позвонить по протоколу `trueconf:` с автоматическим логином и звонком конкретному пользователю по его TrueConf ID, либо в конкретную групповую конференцию по её ID.

Звонок выполняется методом `parseProtocolLink` в качестве параметра которого идет строка вызова в формате `String` .

Подробнее об управляющем протоколе `trueconf:` можно почитать в данной статье.

11.3. Пример 3. Работа с групповыми конференциями

Одной из важнейших возможностей TrueConf Mobile SDK является возможность участия в видеоконференциях с несколькими пользователями одновременно. На данный момент возможно только подключение к существующим конференциям.

Пример выглядит также, как и первый, за исключением, что вместо метода `callTo` для подключения к конференции мы используем метод `joinConf(conferenceId)` .

11.4. Пример 4. Работа со статусами пользователей

В примере показано отслеживание статусов других пользователей на сервере.

Пример на 2 экрана: на первом происходит подключение к серверу и авторизация, после этого открывается второй экран со списком всех пользователей из нашей адресной книги.

В примере также продемонстрировано получение текущего статуса пользователя с помощью метода `getUserStatus(user)` , а также реакция на `callback` в `onUserStatusUpdate` и `onContactListUpdate` .

11.5. Пример 5. Кастомизация интерфейса

Пример показывает, как добавить кнопки на экран внутри звонка. Для этого используется метод `setNewExtraButtons` , в аргумент которого необходимо поместить массив объектов типа `TCExtraButton` . Список кнопок визуально появится при нажатии на кнопку “троеточие” в конференции. В качестве примера добавлены 2 кнопки, которые при нажатии открывают новое окно поверх окна конференции. Первая кнопка открывает `Fragment` , а вторая - `Activity` .

Для замены иконок внутри звонка нужно загрузить ресурсы в папку `drawables` со следующими именами:


```
ic_bluetooth_audio_arrow_off
ic_bluetooth_audio_arrow
ic_bluetooth_audio
ic_call_end
ic_camera_off
ic_camera_on
ic_dyn_arrow_off
ic_dyn_arrow
ic_empty_invoice
ic_headphones_arrow_off
ic_headphones_arrow
ic_headset_mic_arrow_off
ic_headset_mic_arrow
ic_headset_mic
ic_headset
ic_mic_off
ic_mic_on
ic_more_horiz
ic_phone_in_talk_arrow_off
ic_phone_in_talk_arrow
ic_phone_in_talk_off
ic_phone_in_talk
ic_sound_off
ic_sound_on
ic_rotate
ic_minimize_fullscreen
conf_button_back
```

11.6. Пример 6. Чат

В примере показана возможность использования функции отправки текстовых сообщений `sendChatMessage` и обработки события о новых входящих сообщениях `onChatMessageReceived`.

Если сообщение отправляется пользователю, который оффлайн, то оно ему придёт, как только он станет онлайн. Если сообщение отправляется в момент, когда отсутствует связь с сервером, то оно ставится в очередь и отправится в момент, когда связь с сервером возобновится.

11.7. Пример 7. Кастомизация вывода видеоокон в конференции

В примере реализована возможность размещения собственного изображения и изображения участников конференции в отдельном Fragment. Показана самостоятельная реализация кнопок управления конференцией, настройка оборудования перед началом конференции, а также замена окон исходящего/входящего вызовов на кастомные Fragment.

Показанные в примере возможности и использованные для этого фрагменты кода:

1. Создание своего окна входящего вызова на базе наследника класса `IncomingCallFragment`. Если необходимо изменить поведение при принятии или отклонении звонка, нужно переопределить методы `onAcceptClick` и `onDeclineClick` соответственно.
2. Кастомизация окна исходящего вызова на базе наследника класса `PlaceCallFragment`. Если необходимо изменить поведение при отмене звонка, нужно переопределить метод `onHangupClick`.
3. Изменение окна конференции на базе наследника класса `ConferenceFragment`.
4. Изменение размера окна звонка с помощью метода `setCallLayoutParams`.
5. Управление микрофоном с помощью метода `setDefaultMicEnabled()` и пр.
6. Управление камерой с помощью метода метода `setDefaultCameraEnabled()` и пр.
7. Настройка динамика с помощью метода `setDefaultSpeakerEnabled()` и др.

8. Изменение размеров и координат видеоокон участников конференции.

> *i* Чтобы посмотреть работу функционала изменения размеров и координат окон, раскомментируйте содержимое метода `onCalculateCustomLayouts` в `MainActivity` в примере.