阅读准备

准备好一个保利威账号，并在直播后台中的设置 -> 开发者设置页面中获取账号直播系统中的`appId`（应用 ID），`appSecret`（应用密匙），`userId`（账号 ID）；直播->直播列表中获取`channelId`（频道号）。若为登录回放则额外需要频道号对应回放列表里的`vid`（回放视频 id）。

环境要求

名称	要求
Hbuilderx	>=4.85

使用方式

第一种：使用 PLVLiveScenesPluginHelper （推荐）

下载 PLVLiveScenesPluginHelper 并放入项目目录内;如果您使用 vscode 进行开发，那么可以将目录内的.d.ts 一并带入，获得更好的类型提示

使用 PLVLiveScenesPluginHelper 您将无需关心如何引入以及引入原生插件顺序。

```
import { PLVLiveScenesPluginHelper } from "项目目录/plv-live-scense-plugin-helper";
```

实例化插件

创建插件实例，后续所有操作都通过该实例进行。

```
const plugin = PLVLiveScenesPluginHelper(options);
```

`PLVLiveScenesPluginHelper(options)`

参数`options`：

参数名	类型	必填	描述
`type`	PluginType	否	插件类型，默认为 LiveScenesPlugin

参数名	类型	必填	描述
`env`	String	否	插件运行环境，默认为 development, 可选值包括 'development' 和 'production' 该值决定日志输出行为

```
enum PluginType {
  /** 纯观看插件 */
  LiveScenesPlugin = "PLV-LiveScenesPlugin",
  /** 观看 + 开播复合插件 */
  LiveScenesHostPlugin = "PLV-LiveUniPlugin",
}
```

`initService(options)`

初始化插件服务。在使用其他功能之前，必须先调用此方法。

参数 `options`：

参数名	类型	必填	描述
`viewerId`	String	是	观看者 ID，用于唯一标识用户，例如用户手机号。
`viewerName`	String	是	观看者昵称。
`viewerAvatar`	String	是	观看者头像 URL。
`code`	String	否	跑马灯配置代码（可选）。

示例:

```
await plugin.initService({
  viewerId: "user001",
  viewerName: "UniappDemo用户",
  viewerAvatar: "https://example.com/avatar.jpg",
});
```

`setUserInfo(options)`

为防止 APP 端被嗅探和反编译到开发者配置信息，强烈建议开发者设计自己的加密方式在服务端对 `userId`、`AppID`、`AppSecret` 进行加密，并通过 https 协议的接口获取后重新解密，再将参数传递给

SDK。

设置用户账号信息。在 `initService` 之后，进入直播间或回放前调用。

参数 `options`：

参数名	类型	必填	描述
`appId`	String	是	开发者在保利威平台申请的 App ID。
`userId`	String	是	开发者在保利威平台申请的 User ID。
`appSecret`	String	是	开发者在保利威平台申请的 App Secret。

示例：

```
await plugin.setUserInfo({
  appId: "YOUR_APP_ID",
  userId: "YOUR_USER_ID",
  appSecret: "YOUR_APP_SECRET",
});
```

`loginLiveRoom(options)`

登录并进入直播房间。

参数 `options`：

参数名	类型	必填	描述
`sceneType`	Number	是	场景类型：`1` 为云课堂，`2` 为直播带货。
`channelId`	String	是	频道的唯一标识 ID。

示例：

```
await plugin.loginLiveRoom({
  sceneType: 1, // 云课堂场景
  channelId: "YOUR_CHANNEL_ID",
});
```

`loginPlaybackRoom(options)`

登录并进入回放房间。

参数 `options`：

参数名	类型	必填	描述
`sceneType`	Number	是	场景类型：`1` 为云课堂，`2` 为直播带货。
`channelId`	String	是	频道的唯一标识 ID。
`videoId`	String	是	视频的 Vid。
`vodType`	Number	是	回放类型：`0` 为单个视频回放，`1` 为回放列表。

示例：

```
await plugin.loginPlaybackRoom({
  sceneType: 1,
  channelId: "YOUR_CHANNEL_ID",
  videoId: "YOUR_VIDEO_ID",
  vodType: 0, // 单个视频回放
});
```

`logoutRoomMessage(callback)`

注册一个回调函数，当用户退出房间时（目前仅支持 iOS）会触发。

参数 `callback`：

参数名	类型	必填	描述
`res`	Function	是	退出房间后执行的回调函数。

示例：

```
plugin.logoutRoomMessage((res) => {
  console.log("已退出房间", res);
});
```

开播功能，需要集成全功能版本插件，并设置type为 LiveScenesHostPlugin

```
import {
  PLVLiveScenesPluginHelper,
  PluginType,
} from "../../tools/plv-live-scense-plugin-helper.mjs";
const plugin = PLVLiveScenesPluginHelper({
  type: PluginType.LiveScenesHostPlugin,
});
```

`loginStreamer`

登录主播端开播

参数 `options`:

参数名	类型	必填	描述
`channelId`	String	是	频道的唯一标识 ID。
`password`	String	是	频道登录密码。
`nickname`	Number	是	主播名称。

示例:

```
plugin.loginStreamer({
  channelId: "YOUR_CHANNEL_ID",
  password: "YOUR_CHANNEL_PASSWORD",
  nickname: "YOUR_NAME",
});
```

`setScreenShareGroup`

设置屏幕共享（仅 iOS 需要）

参数 `options`:

参数名	类型	必填	描述
`appGroup`	String	是	屏幕共享组标识

示例:

```
plugin.setScreenShareGroup(YOUR_APP_GROUP);
```

完整使用流程

一个典型的使用流程如下：

1. **引入并实例化插件**。
2. 调用 `initService` **初始化服务**。
3. 调用 `setUserInfo` **设置用户信息**。
4. 根据需要调用 `loginLiveRoom` 或 `loginPlaybackRoom` **进入直播或回放**。

```
import { PLVLiveScenesPluginHelper } from "./tools/plv-live-scense-plugin-helper.js";

const plugin = PLVLiveScenesPluginHelper();

async function start() {
  try {
    // 1. 初始化服务
    await plugin.initService({
      viewerId: "user001",
      viewerName: "Demo用户",
      viewerAvatar: "https://example.com/avatar.jpg",
    });

    // 2. 设置用户信息
    await plugin.setUserInfo({
      appId: "YOUR_APP_ID",
      userId: "YOUR_USER_ID",
      appSecret: "YOUR_APP_SECRET",
    });

    // 3. 登录直播间
    await plugin.loginLiveRoom({
      sceneType: 1,
      channelId: "YOUR_CHANNEL_ID",
    });

    console.log("登录成功");
  } catch (error) {
    console.error("发生错误:", error);
  }
}

start();
```

第二种：自行引入原生插件

自行引入原生插件

```
const playModule = uni.requireNativePlugin("PLV-LiveScenesPlugin-PlayModule");
const configModule = uni.requireNativePlugin(
  "PLV-LiveScenesPlugin-ConfigModule"
);
```

全功能版则是

```
var playModule = uni.requireNativePlugin("PLV-LiveUniPlugin-WatchPlayModule")
var configModule = uni.requireNativePlugin("PLV-LiveUniPlugin-WatchConfigModule")
```

观看端配置模块 - ConfigModule

ConfigModule 封装了账号信息、用户信息、SDK 配置功能。 开发者要播放保利威视频，需先到 保利威官网 注册账号，登录账号后，进入**云直播 - 开发设置** 获取 `userId` 、 `AppID` 、 `AppSecret` ，并将加密得到加密串放到自己的服务器，再在移动端通过网络获取加密串，app 本地解密，并设置给 `setConfig` 方法。配置模块信息如果没有特殊说明都需要在进入直播间前配置。

1. setViewerInfo

配置直播间用户信息。(直播间用户的属性在初始化后不允许修改) 方法： `setViewerInfo()`

示例：

```
configModule.setViewerInfo({ viewerId:"", viewerName: "", viewerAvatar: "", },
(result) => { })
```

参数：

名称	类型	说明
params.viewerId	String	对应观看日志中的 用户 ID
params.viewerName	String	对应观看日志中的 用户昵称
params.viewerAvatar	String	观看用户的头像
callback	function `{"isSuccess": 0, @"errMsg": ""}`	执行结果的 callback，返回是否成功的状态和描述

2. setConfig

设置多场景的账号属性。**为防止 APP 端被嗅探和反编译到开发者配置信息，强烈建议开发者设计自己的加密方式在服务端对 `userId` 、 `AppID` 、 `AppSecret` 进行加密，并通过 https 协议的接口获取后重新解密，再将参数传递给 SDK。** 方法： `setConfig()`

示例：

```
configModule.setConfig({
 appId: "",
 userId: "",
 appSecret: ""
}, (result) => {
});
```

参数：

名称	类型	说明
params.appId `(必需)`	String	polyv 云直播应用 ID （可在云直播后台插件）
params.userId `(必需)`	String	polyv 云直播账号 ID
params.appSecret `(必需)`	String	用户头像地址 polyv 云直播应用密匙
callback	function `{"isSuccess": 0, @"errMsg": ""}`	执行结果的 callback，返回是否成功的状态和描述

3. setMarqueeConfig

设置自定义跑马灯的配置属性 方法：`setMarqueeConfig()`

示例

```
configModule.setMarqueeConfig({
  code: "",
}, (result) => {
})
```

参数：

名称	类型	说明
params.code	String	跑马灯 code 参数配置
callback	function `{"isSuccess": 0, @"errMsg": ""}`	执行结果的 callback，返回是否成功的状态和描述

观看端播放模块 - PlayConfig

playModule 封装了云课堂、直播带货的直播和回放功能。

4. showFullScreenButtonOnIPad

是否在 iPad 上显示全屏按钮（仅在 iPad 云课堂场景），需要在进入房间前调用

方法：`showFullScreenButtonOnIPad()`

示例：

```
playModule.showFullScreenButtonOnIPad({show:true});
```

参数：

名称	类型	说明
show `（必需）`	Bool(true 显示全屏按钮，false 显示)	

5. loginLiveRoom

直播登录（云课堂场景，带货直播场景登录）

方法：`loginLiveRoom()`

示例：

```
// 直播
playModule.loginLiveRoom(
  1,
  {
    channelId: "",
    liveParam4: "",
    liveParam5: "",
  },
  (result) => {}
);
```

参数：

名称	类型	说明
sceneType `（必需）`	Number(1 云课堂场景, 2 直播带货场景)	直播室类型
params.channelId `（必需）`	String	直播的频道号

名称	类型	说明
params.liveParam4 `(非必需)`		
params.liveParam5 `(非必需)`		
callback	function `{"isSuccess": 0, @"errMsg": ""}`	执行结果的 callback，返回是否成功的状态和描述

6. loginPlaybackRoom

登录回放直播间（云课堂，带货直播回放登录）

方法：`loginPlaybackRoom()`

示例：

```
playModule.loginPlaybackRoom(
  0,
  {
    channlId: "",
    liveParam4: "",
    liveParam5: "",
    videoId: "",
    vodType: 0,
  },
  (result) => {}
);
```

参数：

名称	类型	说明
sceneType `(必需)`	Number(1 云课堂场景, 2 直播带货场景)	直播室类型
params.channelId `(必需)`	String	直播的频道号
params.liveParam4 `(非必需)`		
params.liveParam5 `(非必需)`		
params.videoId `(必需)`	String	回放视频 id
params.vodType `(必需)`	Number 0 回放视频 1 回放列表	

名称	类型	说明
callback	function `{"isSuccess": 0, @"errMsg": ""}`	执行结果的 callback，返回是否成功的状态和描述

7. exitRoomCallback (仅 ios)

退出直播间回调

方法：`setExitRoomCallback()`

示例：

```
playModule.setExitRoomCallback((res) => {
  Logger.info("退出房间成功");
  cb && cb(res);
});
```

主播端配置模块

```
var liveModule = uni.requireNativePlugin("PLV-LiveUniPlugin-StreamerLiveModule")
var configModule = uni.requireNativePlugin("PLV-LiveUniPlugin-StreamerConfigModule")
```

1. setAppGroup【仅iOS有效】

屏幕共享帐号配置的App Group 方法：`setAppGroup()`

示例：

```
configModule.setAppGroup({
    appGroup:""
}, (result) => {
})
```

参数：

名称	类型	说明
params.appGroup	String	帐号配置的App Group
callback	function `{"isSuccess": 0, @"errMsg": ""}`	执行结果的callback，返回是否成功的状态和描述

主播端直播模块 - LiveModule

登录开播推流

方法：`loginStreamer()`

示例：

```
playModule.loginStreamer({
    channelId: "",
    password: "",
    nickname: ""
}, (result) => {
});
```

参数：

名称	类型	说明
`channelId` (必需)	String	登录频道号
`password` (必需)	String	登录密码
`nickname` (非必需)	String	昵称
callback	function `{"isSuccess": 0, @"errMsg": ""}`	执行结果的callback，返回是否成功的状态和描述