# IdeaPlayer集成开发

**1、概述**

IdeaPlayer依赖于Google开源项目ExoPlayer实现。相比MediaPlayer，ExoPlayer拥有更丰富的接口，更高的可定制性，并且功能更强大：支持DASH，SmoothStreaming，HLS自适应流以及常用媒体格式如MP4，M4A，FMP4，WebM，MKV，MP3，Ogg，WAV，MPEG-TS，MPEG-PS，FLV，ADTS(AAC)等，同时ExoPlayer实现了DRM方案。支持安卓7.0及以上。

IdeaPlayer模块支持HLS/DASH加密流的drm解密播放，提供快速集成方案。

ExoPlayer版本：2.8.4

IdeaPlayer版本：1.4

**2、功能接口**

*IIdeaPlayer.java*

|  |  |
| --- | --- |
| **类型** | **方法和描述** |
| **void** | **fastForward()**  步进播放，修改步进大小见seekFastForwardMs(int fastForwardMs)方法 |
| **long** | **getBufferedPosition()**  返回：当前媒体流的已缓冲时长（毫秒） |
| **long** | **getCurrentPosition()**  返回：当前媒体流的播放位置（毫秒） |
| **long** | **getDuration()**  播放器无法立即获得播放总时长，因此需要等待播放器解析视频信息。例：当{@link #isPlaying()}返回true时已能获取到正确时长值。  返回：当前媒体流的总播放时长（毫秒） |
| **String[]** | **getOttIDs()**  获得IdeaPlayer记录的OTT系统id信息  返回：string[0]:ottDeviceID string[1]:ottContentID |
| **float[]** | **getPlaybackParamas()**  获得相关播放参数，  返回：float[0]:播放速率 float[1]:声调 |
| **int** | **getPlaybackState()**  获得播放器当前状态，配合{@link #isPlayWhenReady()}可得到播放器精确状态，状态值详细描述见GosConstant.java STATE常量。  返回：播放器内部状态等效值（1~4） |
| **float** | **getPlaybackSpeed()**  返回：播放器当前播放速率（默认1.0） |
| **float** | **getVolume()**  获得播放器音量，此数值为播放器自身音量大小，不等同于实际系统媒体音量值。范围为0~1.0，1.0对应系统当前媒体音量大小  返回：播放器内部音量值（0~1.0） |
| **boolean** | **isLoading()**  检查播放器是否处于资源载入状态  返回：true:播放器正在载入资源 false:播放器未处于资源载入状态 |
| **boolean** | **isPlaying()**  检查播放器是否处于正在播放状态  返回：true:播放器正在播放或缓冲 false:播放器未开始播放或缓冲 |
| **boolean** | **isPlayingAd()**  检查是否正在播放VAST广告。  返回：true: 播放器正在播放广告 false: 未播放广告 |
| **boolean** | **isPlayWhenReady()**  检查播放器是否已经得到外部播放允许，  配合{@link #getPlaybackState()}方法可以获得播放器精确的状态  返回：true:外部允许立即播放 false:外部不允许播放 |
| **boolean** | **isPrepared()**  检查当前播放媒体是否可以进行跳跃播放，如果进度条长度为0则返回false  返回：true 允许使用{@link #seekTo}跳跃播放 false:不允许跳跃播放 |
| **boolean** | **isSeekable()**  检查当前播放媒体是否可以进行跳跃播放，如果进度条长度为0则返回false  返回：true 允许使用{@link #seekTo}跳跃播放 false:不允许跳跃播放 |
| **void** | **next()**  如果存在多个媒体资源，调用此方法播放下一个 |
| **void** | **pause()**  暂停播放，可调用{@link #start()}继续播放 |
| **void** | **previous()**  如果存在多个媒体资源，调用此方法播放上一个，否则重新播放当前流 |
| **Map<K, V>** | **queryKeyStatus()**  查询播放权限状态，只有ChinaDrm加密流存在状态查询，必须在播放器开始播放之后调用。详细键名称见GosConstant @code QKS常量  返回：Map<String, String>对象，包含当前详细的状态信息 |
| **void** | **release()**  销毁播放器并释放相关资源 |
| **void** | **restart()**  使当前播放的媒体流重新开始播放 |
| **void** | **rewind()**  步退播放，修改默认步退幅值请调用setRewindMs(int rewindMs)方法 |
| **void** | **seekTo(long positionMs)**  跳跃播放，在当前媒体流中跳跃到指定时间点。见{@link #seekTo}  positionMs：指定时间点（毫秒） |
| **void** | **seekTo(int windowIndex, long positionMs)**  跳跃播放，跳跃到指定媒体流中指定的时间点  windowIndex：当存在多个播放地址时，指定媒体流索引  positionMs：指定时间点（毫秒）  *可能的异常：*当指定非法窗口序号或者timeline尚未创建完成时进行跳转，此通知将会抛出IllegalSeekPositionException(timeline, windowIndex, positionMs)异常。 |
| **void** | **setAspectRatio(String ratio)**  设置视频宽高比。  ratio：画面比例，等效常量见GosConstant.java @code RATIO |
| **void** | **setDebugMode(boolean allowSaveLog)**  调试模式开关，日志文件优先保存到USB设备中，如果未发现USB设备则保存到内置存储卡根目录，文件名为gosLog.txt。  allowSaveLog：true:允许记录消息日志到本地文件 false:禁止记录日志 |
| **void** | **setFastForwardMs(int fastForwardMs)**  设置播放的步进增量，默认为10秒，最大允许设置范围为1000毫秒~600000毫秒。  fastForwardMs：步进大小（毫秒） |
| **void** | **setRepeatMode(@GosConstant.RepeatMode int repeatMode)**  立即改变播放器播放模式，可在播放后调用。  例1：REPEAT\_MODE\_OFF：正常播放  例：REPEAT\_MODE\_ONE：无限循环播放当前媒体流  例：REPEAT\_MODE\_ALL：无限循环播放所有媒体流  repeatMode：播放模式 |
| **void** | **setRewindMs(int rewindMs)**  设置播放的步退大小，默认为5秒，最大允许设置范围为1000毫秒~600000毫秒。  rewindMs：步退大小（毫秒） |
| **void** | **setSpeed(float speed)**  设置播放速率，默认为1.0  speed：播放速率，允许范围为0.5~2.0 |
| **void** | **setSpeed(float speed, float pitch)**  设置播放速率和声调，默认均为1.0，允许范围均为0.5~2.0  speed：播放速率  pitch：声调速率 |
| **void** | **setVolume(float audioVolume)**  设置播放器内部音量，默认为1.0，对应当前系统媒体音量  audioVolume：指定音量值，有效范围为0~1.0 |
| **void** | **start()**  使播放器开始播放。 |
| **void** | **stop()**  使播放器停止播放，等同于stop(false)。 |
| **void** | **stop(boolean reset)**  使播放器停止播放，将播放器状态置为Player.STATE\_IDLE*，*失去所有播放资源*，*属不可逆状态，如果希望在此之后能继续播放，请使用pause()方法。播放器停止后，如需继续使用，请重新配置IdeaPlayer。调用此方法并不会彻底释放播放器，因此需要手动调用{@link #release()}方法。  reset：是否重置播放器状态 |

*IdeaPlayer$Builder*

*使用Builder模式对IdeaPlayer进行创建和初始化。最后调用{@link #build()}方法可初始化IdeaPlayer，并且得到IdeaPlayer单例。这是创建IdeaPlayer的唯一途径。*

*注：除了特别注明的方法，均返回{@link Builder}对象。*

|  |  |
| --- | --- |
| **Builder** | **addEventListener(GosEventListener listener)**  为IdeaPlayer添加播放器状态事件监听器，属于低级事件监听。  listener：监听器对象，为GosEventListener子类 |
| **Builder** | **setPath(String... mediaPath)**  播放器设置媒体地址，可以传入至少一个媒体地址进行顺序播放，但当调用{@link #setAds(String)}方法后，将只有第一个地址有效。并且，仅支持以下媒体流组合形式：   1. 未加密流 2. 相同加密方案的加密流 3. 未加密流 + 相同加密方案的加密流   mediaPath：一个或多个媒体地址 |
| **Builder** | **setLicenseUrl(String licenseUrl)**  设置RI服务器地址，如未设置则使用内置RI服务器地址。  license Url：RI服务器地址 |
| **Builder** | **setAds(String adsTag)**  添加VAST广告。如果需要使用此功能，必须为module添加依赖'com.google.android.gms:play-services-ads:11.4.2'。  adsTag可以是一个有效URL，也可以是一个符合规范的VAST标签。详细描述见VASTv3.0标准规范。当前只能为单一媒体流匹配VAST广告，因此调用此方法后无论传入多少个媒体路径，都将只使用第一个媒体路径。  adsTag：一个URL或者符合规范的VAST标签字符串 |
| **Builder** | **bindView()**  IdeaPlayer记录一个PlayerView对象，并在播放器初始化时自动地与此对象进行绑定以显示画面。  playView 必须为com.google.android.exoplayer2.ui.PlayerView类或其子类对象 |
| **Builder** | **setLoadControl(GosLoadControl loadControl)**  使用一个自定义轨道选择策略替换默认策略。  loadControl：自定义缓冲策略，必须为{@link GosLoadControl}子类 |
| **Builder** | **setDataSourceFactory()**  使用一个自定义数据源工厂替换默认实现类。  factory：自定义数据源工厂，必须为{@link GosDataSourceFactory}子类 |
| **Builder** | **setRenderersFactory(GosRenderersFactory factory)**  使用一个自定义渲染器工厂替换默认实现类。  factory：自定义渲染器工厂，必须为{@link GosRenderersFactory}子类 |
| **Builder** | **setTrackSelector(GosTrackSelector trackSelector)**  置一个自定义轨道选择策略，替换默认策略。  trackSelector：自定义轨道选择策略，必须为{@link GosTrackSelector}子类 |
| **Builder** | **setOttIDs(String ottDevicesID, String ottContentID)**  IdeaPlayer记录OTT系统的一些id信息。***目前只对ChinaDRM加密流有效。***  ottDeviceID：盒子唯一识别id(UserId)  ottContentID：OTT节目id |
| **IdeaPlayer** | **build()**  完成一系列初始化工作。  返回：{@link IdeaPlayer}对象 |

*IdeaPlayer*

|  |  |
| --- | --- |
| **IdeaPlayer** | **getIdeaPlayer()**  返回：IdeaPlayer单例，如未创建则返回null |

*GosEventListener*

|  |  |
| --- | --- |
| **void** | **onPlayerStateChanged(boolean playWhenReady, int state)**  当播放器状态发生变更时，将会通知此事件。  playWhenReady：true:已经调用了{@link IdeaPlayer#start()}方法以表明外部允许播放 false:外部未允许播放  state：状态等效值，见{@link GosConstant @code STATE} |
| **void** | **onError(String info, int code)**  播放器错误通知事件。当播放器中任一过程发生不可预期的错误，将会通知此事件。  info：错误详细描述  code：错误码等效常量，见{@link GosConstant @code CODE} |
| **void** | **onPlaybackParametersChanged(float speed, float pitch, boolean keepSilence)**  播放参数变更事件。当调用{@linkIdeaPlayer#setSpeed}方法时，通知此事件。  speed：视频播放速率  pitch：声调  keepSilence：静音开关 |
| **void** | **onSeekProcessed()**  进度跳转事件。   1. 当调用{@link #seekTo}方法时，将会通知此事件。 2. 此方法伴随着{@link IdeaPlayer#onPositionDiscontinuity(int)}事件的通知。 |
| **void** | **onPositionDiscontinuity(int reason)**  播放连续性被中断事件。进行{@link IdeaPlayer#seekTo}方法时通知此事件。  reason：连续性被中断的原因等效值，见GosConstant @code DC。 |
| **void** | **onRepeatModeChanged(int repeatMode)**  重复播放模式变更事件。  调用{@link IdeaPlayer#setRepeatMode(int)}方法时，将会通知此事件。  repeatMode：重复模式等效值，见GosConstant @code REPEAT\_MODE |
| **void** | **onLoadingChanged(boolean isLoading)**  资源下载状态变更事件。当播放器变更到GosConstant @code STATE\_BUFFERING状态或者从此状态离开时，将会通知此事件。  isLoading：true:资源需要开始下载 false:未处于资源下载状态 |
| **void** | **onTracksChanged(Object trackGroups, Object trackSelections)**  当可用的或者当前选定的轨道变更时，将会回调此方法。  trackGroups：（缺省）  trackSelections：（缺省） |
| **void** | **onShuffleModeEnabledChanged(boolean isShuffle)**  暂未提供相关方法，因此IdeaPlayer不会通知此事件。  isShuffle：（缺省） |

**3、使用步骤**

**3.1、导入IdeaPlayer-dev库**

**3.1.1 Android Studio导入aar包**

**①** 导入visioncryptdev库和默认UI库：将**IdeaPlayer-dev**.aar拷贝到目标modular的libs文件夹中。

**②** 在目标modular的build.gradle中dependencies节点下添加aar包依赖：

implementation fileTree(dir: 'libs', include: ['IdeaPlayer-dev.aar'])

**3.1.2 Eclipse导入jar包**

导入IdeaPlayer-dev.jar包到工程中，并且将此文件夹中的res文件夹合并到module名称/src/main/res目录中。

**3.2、IdeaPlayer使用步骤（*更多实现见Demo工程）***

**Player图层：**使用com.google.android.exoplayer2.ui.PlayerView或其子类，用于显示帧：

<com.google.android.exoplayer2.ui.PlayerView

android:id="@+id/playerView"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

**UI界面：**如需使用自定义UI界面：只需要为PlayerView添加属性app:use\_controller="false"，并使UI布局位于PlayerView图层上层即可。

**java代码：**使用IdeaPlayer快速创建和使用播放器

IdeaPlayer player = new IdeaPlayer.Builder(context)

.setPath(paths)

.setOttIDs(ottUserId, ottContentId)

.bindView(playerView)

.build();

player.start();

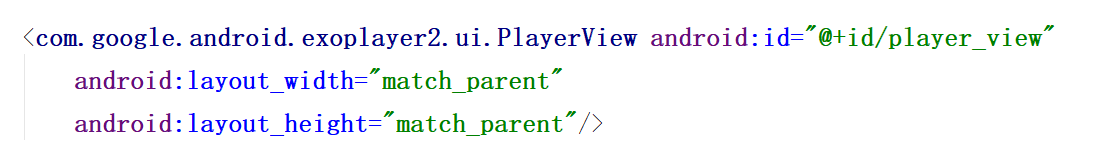
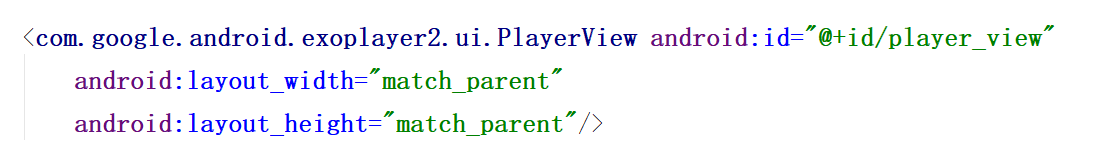
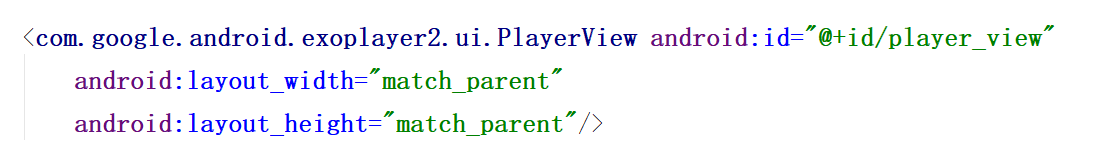
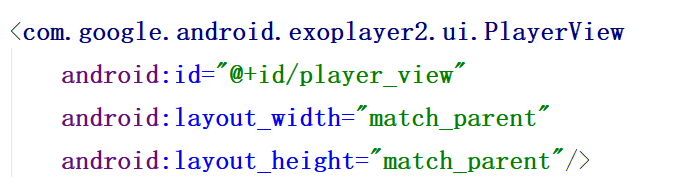
可能需要的权限：

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

**4、ExoPlayer定制UI**

① 自定义LoadControl：setLoadControl(loadControl)。loadControl必须为GosLoadControl子类对象，默认缓冲策略实现类见DefaultLoadControl.java。

② 自定义TrackSelector：setTrackSelector(trackSelector)。trackSelector必须为GosTrackSelector子类对象，默认策略实现类见DefaultTrackSelector.java。

③ 自定义RenderersFactory：setRenderersFactory(factory)。factory必须为GosRenderersFactory子类对象，默认实现为DefaultRenderersFactory.java

④ 自定义DataSource：setDataSourceFactory(factory)。factory必须为GosDataSourceFactory子类对象，默认实现为DefaultDataSourceFactory.java

⑤ 自定义UI样式但保留原生功能：在PlayerView中添加布局，在布局中对控件设置预置id（播放器原生UI模块预置id见下方），并启用控制条：

app:controller\_layout\_id="@layout/customLayout"

app:use\_controller="true"

原生UI 模块预置id（见ids.xml）：

exo\_play //播放

exo\_pause //暂停

exo\_rew //上一个

exo\_ffwd //步进

exo\_prev //步退

exo\_next //下一个

exo\_duration //总时长

exo\_position //当前播放时间点

exo\_progress *//*进度条，使用com.google.android.exoplayer2.ui.DefaultTimeBar或其子类

如需其它改动*，*需重写com.google.android.exoplayer2.ui.PlayerView及com.google.android.exoplayer2.ui.PlayerControlView类甚至修改UI library源码以实现更多自定义可能性。