# youtube播放器

# 参考文档

谷歌搜索：YouTube Android Player 停更

<https://stackoverflow.com/questions/75741191/youtube-android-player-api-deprecated-in-2023-so-how-can-implement-youtube-playe>

YouTube player api deprecated

https://support.google.com/googleplay/android-developer/thread/213353293/youtube-player-api-deprecated?hl=en

# 版本号

使用谷歌直接搜索下面的库：

例如搜索 com.google.api-client:google-api-client-android

com.google.apis:google-api-services-youtube 版本

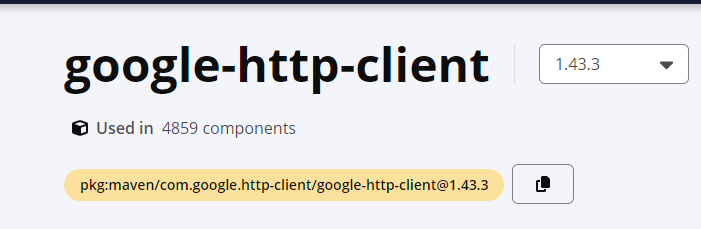
https://central.sonatype.com/artifact/com.google.apis/google-api-services-youtube/v3-rev20231011-2.0.0

com.google.api-client:google-api-client-android 版本

https://mvnrepository.com/artifact/com.google.api-client/google-api-client-android

google-http-client

https://central.sonatype.com/artifact/com.google.http-client/google-http-client



Class AndroidHttp过时

https://cloud.google.com/java/docs/reference/google-http-client/1.43.0/com.google.api.client.extensions.android.http.AndroidHttp

# 播放器测试代码

android-youtube-player-master

# 播放器接入

以下是文档翻译：

<https://github.com/PierfrancescoSoffritti/android-youtube-player>

## 快速开始

为了开始使用播放器，您需要在布局中添加YouTubePlayerView。

<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical" > <com.pierfrancescosoffritti.androidyoutubeplayer.core.player.views.YouTubePlayerView  
 android:id="@+id/youtube\_player\_view"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:videoId="S0Q4gqBUs7c"  
 app:autoPlay="true" />  
</LinearLayout>

建议您将YouTubePlayerView添加为其父“Activity/Fragment”的生命周期观察者。您可以在文档中阅读原因。

原因：

YouTubePlayerView实现了LifecycleObserver接口，这意味着它是一个感知生命周期的组件。

如果添加为Activity/Fragment生命周期的观察者，YouTubePlayerView将更智能。强烈建议您将YouTubePlayerView注册为LifecycleObserver。

lifecycleOwner.getLifecycle（）.addObserver（youTubePlayerView）；

将YouTubePlayerView作为观察者添加到生命周期中，将允许YouTubePlayerView在活动/片段停止时自动暂停播放（而不是在暂停时，以支持多窗口应用程序）。

如果您希望您的应用程序在Activity/Fragmen不可见时继续播放（请记住，如果您想在PlayStore上发布应用程序，这种行为是不允许的），请不要将YouTubePlayerView注册为生命周期观察员。但请记住，当活动/片段被销毁时，要手动调用release（）。

YouTubePlayerView youTubePlayerView = findViewById(R.id.youtube\_player\_view);  
getLifecycle().addObserver(youTubePlayerView);

（如果你在将YouTubePlayerView添加为LifecycleObserver时遇到问题，你可能没有使用androidx，我建议你迁移依赖项）

这就是你所需要的，一个YouTube视频正在你的应用程序中播放。

如果你想要更多的控制，一切都可以通过编程方式完成，方法是获取对YouTubePlayerView的引用，并向其中添加YouTubePlayerListener。

YouTubePlayerView youTubePlayerView = findViewById(R.id.youtube\_player\_view);  
getLifecycle().addObserver(youTubePlayerView);  
  
youTubePlayerView.addYouTubePlayerListener(new AbstractYouTubePlayerListener() {  
 @Override  
 public void onReady(@NonNull YouTubePlayer youTubePlayer) {  
 String videoId = "S0Q4gqBUs7c";  
 youTubePlayer.loadVideo(videoId, 0);  
 }  
});

如果您决定以编程方式初始化播放器，请记住从XML文件中的YouTubePlayerView中删除autoPlay和videoId属性。

# API documentation

以下部分提供了库中每个组件的详细文档。

如果您在文档中看到任何问题或错误，请随时通过打开issue和/或发送拉取请求来做出贡献。

## YouTubePlayerView

YouTubePlayer View是YouTubeLayer的接入点。

您可以将视图添加到布局中。

<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" >  
 <com.pierfrancescosoffritti.androidyoutubeplayer.core.player.views

.YouTubePlayerView

android:id="@+id/youtube\_player\_view"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"/>  
</LinearLayout>

或者，您可以通过编程方式创建它，并手动将其添加到ViewGroup中。

YouTubePlayerView youTubePlayerView = new YouTubePlayerView(this);  
layout.addView(youTubePlayerView);

如果视图的高度设置为wrap\_content，则视图将自动具有16:9的宽高比，以匹配大多数YouTube视频的宽高比。

### XML attributes

如果将视图添加到XML布局中，则可以设置一些自定义属性，以自定义视图的外观和行为。一切也可以通过编程完成。

videoId

autoPlay

enableAutomaticInitialization

handleNetworkEvents

#### videoId

此属性需要一个String，即YouTube视频的id。

如果设置，播放器将自动开始播放视频。

如果没有设置，播放器将不会自动播放。

一般来说，如果你想让你的播放器只播放一个视频，你应该使用这个属性。这不是规则，只是最佳实践。事实上，即使您设置了该属性，仍然可以以编程方式播放其他视频。

#### autoPlay

此属性需要一个布尔值。其默认值为false。

如果为真，则播放器开始播放带有videoId的视频，而无需等待用户输入。

如果为false，播放器将在播放带有videoId的视频之前等待用户输入。

如果未设置videoId，则此属性无效，因此，如果将其设置为true，YouTubePlayerView将抛出异常。

如果未将YouTubePlayerView添加为其父活动/片段的生命周期观察者，则autoPlay将无法工作。

#### enableAutomaticInitialization

此属性需要一个布尔值。其默认值为true。

如果为真，YouTubePlayerView将负责其初始化。

如果为false，则必须以编程方式初始化YouTubePlayerView。

一般来说，将此属性保留为true是有意义的。仅当需要使用IFramePlayerOptions初始化视图时，才可能需要将其设置为false。

#### handleNetworkEvents

此属性需要一个布尔值。其默认值为true。

如果为真，YouTubePlayerView将通过注册NetworkReceiver来处理网络事件。

如果为false，您将负责处理网络事件。

将此属性设置为true非常有用，这样，如果玩家在初始化YouTubePlayerView时连接中断，网络恢复后，YouTubeLayerView将能够自动恢复初始化。

如果您决定将其设置为false，您还应该禁用enableAutomaticInitialization并自行管理网络事件。

在此处阅读更多关于网络事件的信息。

### Initialization

如果需要以编程方式初始化YouTubePlayerView，可以将其xml属性enableAutomaticInitialization设置为false。您可以通过调用youTubeLayerView.setEnableAutomaticInitialization（false）以编程方式执行相同的操作。

禁用自动初始化后，您需要处理YouTubePlayerView的初始化。

您可以使用以下方法：

YouTubePlayerView.initialize(YouTubePlayerListener listener)  
YouTubePlayerView.initialize(YouTubePlayerListener listener, IFramePlayerOptions iframePlayerOptions)  
YouTubePlayerView.initialize(YouTubePlayerListener listener, boolean handleNetworkEvents)  
YouTubePlayerView.initialize(YouTubePlayerListener listener, boolean handleNetworkEvents, IFramePlayerOptions iframePlayerOptions)  
YouTubePlayerView.initialize(YouTubePlayerListener listener, boolean handleNetworkEvents, IFramePlayerOptions iframePlayerOptions, String videoId)

initialize(YouTubePlayerListener)

初始化YouTubePlayer。网络事件由玩家自动处理。

这个参数是一个YouTubePlayerListener，你可以在这里阅读更多。

initialize(YouTubePlayerListener, boolean)

初始化YouTubePlayer。通过使用布尔值可以决定玩家是否应该处理网络事件，请在此处阅读更多关于网络事件的信息。

initialize(YouTubePlayerListener, boolean, IFramePlayerOptions)

通过将IFramePlayerOptions传递给initialize方法，可以设置IFrame YouTubePlayer的一些参数。在此处阅读更多关于IFramePlayerOptions的信息。

此处列出了所有可能的参数和值。并非所有这些都在这个库中得到支持，因为有些在这种情况下没有意义。如果需要当前不支持的参数，请打开问题。

initialize(YouTubePlayerListener, boolean, IFramePlayerOptions, String)

通过传递videoId，视频将尽快加载

### IFramePlayerOptions

IFramePlayerOptions 是一个可选的参数，可以传递给 YouTubePlayerView.initialize(YouTubePlayerListener, boolean, IFramePlayerOptions)，它可以用来设置 YouTubePlayer IFrame 的一些参数。

如何使用IFramePlayerOptions的简单示例可以在示例应用程序中找到。

使用 Builder 获取 IFramePlayerOptions 对象。

IFramePlayerOptions iFramePlayerOptions = new IFramePlayerOptions.Builder()

.controls(1)

.build();

所有可能的参数和值都列在这里。并非所有这些都在这个库中得到支持，因为有些在此上下文中没有意义。如果您需要目前不支持的参数，请打开问题。

#### Supported options

**controls**

此选项指示 IFrame 播放器的基于 Web 的 UI 是隐藏还是可见。

如果设置为 0:Web UI 不可见。

如果设置为 1:Web UI 可见。

**rel**

此选项控制视频末尾显示的相关视频。

如果设置为0：相关视频将来自与刚才播放的视频相同的频道。

如果设置为1：相关视频将来自多个频道。

**ivLoadPolicy**

此选项控制视频注释(调试信息)。

如果设置为1：玩家将显示注释(调试信息)。

如果设置为3：玩家将不显示注释(调试信息)。

**ccLoadPolicy**

此选项控制视频字幕。它不适用于自动生成的字幕。

如果设置为0：玩家将显示字幕。

如果设置为1：玩家将不显示字幕。

**start**

此参数使播放器从视频开始后的给定秒数开始播放视频。参数值为正整数。

它的工作原理与seekTo方法以及loadVideo和cueVideo的startSeconds参数类似。

**end**

此参数指定播放器应停止播放视频的时间，以秒为单位，从视频开始算起。参数

### Full screen

全屏按钮可以通过使用IFramePlayerOptions添加到播放器

IFramePlayerOptions iFramePlayerOptions = new IFramePlayerOptions.Builder()  
 .controls(1)  
 // enable full screen button  
 .fullscreen(1)  
 .build();

您可以通过向YouTubePlayerView添加一个全屏Listener来收听全屏事件

youTubePlayerView.addFullscreenListener(new FullscreenListener() {  
 @Override  
 public void onEnterFullscreen(@NonNull View fullscreenView, @NonNull Function0<Unit> exitFullscreen) {  
   
 }  
  
 @Override  
 public void onExitFullscreen() {  
 }  
});

请参阅示例应用程序。

当调用全屏监听器#onEnterFullscreen时，播放器将在全屏视图而不是YouTubePlayerView中呈现，直到调用全屏监听器#onExitFullscreen。因此，您需要在全屏启动时将全屏视图添加到应用程序的视图层次结构中，并在全屏结束时将其删除。

您还可以使用YouTubePlayerView#matchParent和YouTubeLayerView#wrapContent展开视图以填充其父级。开发人员有责任隐藏活动中的其他视图，更改活动的方向等。示例应用程序包含一个帮助类，可以帮助您更新应用程序状态，但这不是库的一部分。

如果您需要更改活动/片段的方向，请记住，默认情况下，Android会在方向更改时重新创建活动和片段。通过在清单中的manifest文件中定义中添加属性android:configChanges，确保手动处理方向更改。

<application >

<activity android:configChanges="orientation|screenSize|keyboardHidden|smallestScreenSize|screenLayout" />

</application>

### Playlist

您可以初始化播放器以播放播放列表而不是视频。这可以通过将listType设置为播放列表，然后提供要列出的播放列表的id来完成。

val iFramePlayerOptions = IFramePlayerOptions.Builder()  
 .controls(1)  
 .listType("playlist")  
 .list(PLAYLIST\_ID)  
 .build()

### Release the YouTubePlayerView

使用完毕后，记得通过调用YouTubePlayerView.release（）来释放YouTubeLayerView。

@Override

public void onDestroy() {

super.onDestroy();

youTubePlayerView.release();

}

如果您将播放器注册为活动/片段生命周期的观察者，则不需要手动释放它。

### LifecycleObserver

YouTubePlayerView实现了生命周期观察者接口，这意味着它是一个生命周期感知组件。

如果添加为Activity/Fragment的生命周期的观察者，YouTubePlayerView将更智能。强烈建议您将YouTubePlayerView注册为生命周期观察者。

lifecycleOwner.getLifecycle().addObserver(youTubePlayerView);

将YouTubePlayerView作为观察者添加到生命周期中，将允许YouTubeLayerView在Activity/Fragment停止时自动暂停播放（而不是在暂停时，以支持多窗口应用程序）。

如果你想让你的应用在Activity/Fragment不可见时继续播放（记住，如果你想在PlayStore上发布你的应用，这种行为是不允许的），不要将YouTubePlayerView注册为生命周期观察者。但记住在销毁活动/片段时手动调用release（）。

## YouTubePlayer

YouTubePlayer是负责控制YouTube视频播放的组件。你可以在这里看到它的合同。

每个YouTubePlayer视图都包含一个YouTubeLayer。

### Get a reference to YouTubePlayer

有两种方法可以通过YouTubePlayerView获取YouTubePlayer的引用。

#### YouTubePlayerView.getYouTubePlayerWhenReady

有两种方法可以通过YouTubePlayerView获取YouTubePlayer的引用。YouTubePlayerView.getYouTubeLayerWhenReady可用于获取YouTubePlayer的引用。正如该方法的名称所说，只有当玩家准备就绪时，你才能得到它。

因此，此函数将回调作为参数，当YouTubePlayer准备就绪时，将调用回调。

youTubePlayerView.getYouTubePlayerWhenReady(youTubePlayer -> {

// do stuff with it

})

#### YouTubePlayerListener

YouTubePlayerListener的每个方法都有YouTubePlayer作为参数。

### Load videos

要加载视频，您可以使用两种方法。

YouTubePlayer.loadVideo(String videoId, float startTime)

或者

YouTubePlayer.cueVideo(String videoId, float startTime)

两者之间的区别在于loadVideo加载并自动播放视频，而cueVideo只加载视频和缩略图，但不自动播放。

### Utility for loading videos

如果Activity/Fragment在后台，但您创建了一个YouTubePlayerListener，在调用onReady时调用loadVideo，则即使活动在后台，视频也会开始播放。

要解决这个问题，您应该使用loadOrCueVideo函数。

在Java中作为实用函数提供。

YouTubePlayerUtils.loadOrCueVideo(  
 youTubePlayer,  
 getLifecycle(),  
 videoId,  
 startTime  
);

作为Kotlin中的扩展函数。

youTubePlayer.loadOrCueVideo(lifeCycle, videoId, startTime)

此函数仅在Activity恢复时调用loadVideo，否则将调用cueVideo，以便视频开始加载但不播放。

### Events

在其存在期间，玩家将不断发出事件，您可以通过向其添加YouTubePlayerListener来轻松收听所有事件。

### YouTubePlayerTracker

YouTubePlayerTracker是库提供的一个实用程序，可以轻松跟踪YouTubePlayer的状态和其他信息。

YouTubePlayerTracker是一个YouTubeLayerListener，因此要使用它，您需要将其添加为YouTubePlayer的监听器。

然后，您可以使用跟踪器获取玩家的状态以及有关正在播放的视频的各种信息。

YouTubePlayerTracker tracker = new YouTubePlayerTracker();  
youTubePlayer.addListener(tracker);  
  
tracker.getState();  
tracker.getCurrentSecond();  
tracker.getVideoDuration();  
tracker.getVideoId();

## YouTubePlayerListener

YouTubePlayer监听器用于拦截YouTubeLayer发出的事件。

YouTubePlayer在其存在期间会不断发出事件，您可以通过向其添加YouTubeLayer Listener来监听它们。

youTubePlayer.addListener(YouTubePlayerListener listener);  
youTubePlayer.removeListener(YouTubePlayerListener listener);

这些是YouTubePlayer必须实现的方法，每个方法都引用YouTubeLayer和其他一些参数。

// Called when the player is ready to play videos.  
// You should start using the player only after this method is called.  
void onReady(@NonNull YouTubePlayer youTubePlayer)  
  
// Called every time the state of the player changes.  
void onStateChange(@NonNull YouTubePlayer youTubePlayer, @NonNull PlayerConstants.PlayerState state)  
  
// Called every time the quality of the playback changes.  
void onPlaybackQualityChange(@NonNull YouTubePlayer youTubePlayer, @NonNull PlayerConstants.PlaybackQuality playbackQuality)  
  
// Called every time the speed of the playback changes.  
void onPlaybackRateChange(@NonNull YouTubePlayer youTubePlayer, @NonNull PlayerConstants.PlaybackRate playbackRate)  
  
// Called when an error occurs in the player.  
void onError(@NonNull YouTubePlayer youTubePlayer, @NonNull PlayerConstants.PlayerError error)  
  
// Called periodically by the player, the argument is the number of seconds that have been played.  
void onCurrentSecond(@NonNull YouTubePlayer youTubePlayer, float second)  
  
// Called when the total duration of the video is loaded.  
// Note that getDuration() will return 0 until the video's metadata is loaded, which normally happens just after the video starts playing.  
void onVideoDuration(@NonNull YouTubePlayer youTubePlayer, float duration)  
  
// Called periodically by the player, the argument is the percentage of the video that has been buffered.  
void onVideoLoadedFraction(@NonNull YouTubePlayer youTubePlayer, float loadedFraction)  
  
// Called when the id of the current video is loaded  
void onVideoId(@NonNull YouTubePlayer youTubePlayer, String videoId)  
  
void onApiChange(@NonNull YouTubePlayer youTubePlayer)

如果你不想实现这个接口的所有方法，你可以扩展AbstractYouTubeLayerListener，而不是实现YouTubePlayerListener，只覆盖你感兴趣的方法。

有关YouTubeLayerListener接口中定义的方法的更多信息，请参阅代码库中每个方法上面定义的文档。

### onReady callback

当YouTubePlayer准备好首次使用时，YouTubeLayer Listener的onReady回调会被调用一次。在YouTubePlayer准备就绪之前，您无法使用它。

### onStateChanged callback

YouTubePlayer有一个状态，它会随着播放的变化而相应地变化。可能的状态列表与YouTube IFrame Player API相同。

UNKNOWN  
UNSTARTED  
ENDED  
PLAYING  
PAUSED  
BUFFERING  
VIDEO\_CUED

## Create your own custom UI

定制是这个库的一个重要方面。如果需要，您可以完全替换播放器的默认UI。

YouTubePlayerView有这样的方法。

View inflateCustomPlayerUi(@LayoutRes int customUiLayoutID)  
void setCustomPlayerUi(View view)

此方法接受布局资源的id，这是一个包含布局或视图定义的常规XML文件。

新的用户界面将覆盖在播放器上。因此，建议通过使用IFramePlayerOptions初始化YouTubePlayerView来禁用IFrame播放器的UI。

// disable web ui  
IFramePlayerOptions options = new IFramePlayerOptions.Builder().controls(0).build();  
youTubePlayerView.initialize(listener, options);

您负责使用自己的代码管理自定义UI。含义：你应该编写自己的类来管理UI。这里可以看到一个简单但完整的示例，在示例应用程序中，我建议花几分钟时间阅读它，它应该很容易理解。

示例（取自示例应用程序）：

View customPlayerUi = youTubePlayerView.inflateCustomPlayerUi(R.layout.custom\_player\_ui);  
  
YouTubePlayerListener listener = new AbstractYouTubePlayerListener() {  
  
 @Override  
 public void onReady(@NonNull YouTubePlayer youTubePlayer) {  
 CustomPlayerUiController customPlayerUiController = new CustomPlayerUiController(CustomUiActivity.this, customPlayerUi, youTubePlayer, youTubePlayerView);  
 youTubePlayer.addListener(customPlayerUiController);  
  
 YouTubePlayerUtils.loadOrCueVideo(  
 youTubePlayer, getLifecycle(),  
 VideoIdsProvider.getNextVideoId(),0f  
 );  
 }  
};  
  
// disable iframe ui  
IFramePlayerOptions options = new IFramePlayerOptions.Builder().controls(0).build();  
youTubePlayerView.initialize(listener, options);

一篇更深入的博客文章可以在这个链接上找到。

自定义UI示例：（这只是一个丑陋的示例，但在这里你的设计技能是极限：）

自定义ui示例

警告：更换IFrame UI时，请注意不要违反YouTube的服务条款。如果你打算在PlayStore上发布你的应用程序，改变玩家的外观和感觉可能是一个问题。

### DefaultPlayerUiController

DefaultPlayerUiController是一个预先制作好的即用型自定义UI。

从库的12.0.0版本开始，此UI作为一个单独的模块提供，需要与核心模块结合使用。要导入库，请将以下内容添加到gradle.build文件中的依赖项中：

dependencies {  
 implementation 'com.pierfrancescosoffritti.androidyoutubeplayer:core:{latest-version}'  
 implementation 'com.pierfrancescosoffritti.androidyoutubeplayer:custom-ui:{latest-version}'  
}

导入依赖关系后，您可以这样使用它：

YouTubePlayerListener listener = new AbstractYouTubePlayerListener() {  
 @Override  
 public void onReady(@NonNull YouTubePlayer youTubePlayer) {  
 // using pre-made custom ui  
 DefaultPlayerUiController defaultPlayerUiController = new DefaultPlayerUiController(youTubePlayerView, youTubePlayer);  
 youTubePlayerView.setCustomPlayerUi(defaultPlayerUiController.getRootView());  
 }  
};  
  
// disable iframe ui  
IFramePlayerOptions options = new IFramePlayerOptions.Builder().controls(0).build();  
youTubePlayerView.initialize(listener, options);

UI看起来像这样。

YouTubePlayerSeekBar

您可以使用DefaultPlayerUiController隐藏视图、添加新视图等。

#### Menu

DefaultPlayerUiController有一个可选菜单。您可以使用以下方法来控制菜单的行为：

PlayerUiController.showMenuButton(boolean show);  
PlayerUiController.setMenuButtonClickListener(@NonNull View.OnClickListener customMenuButtonClickListener);

默认情况下，菜单图标不可见。

单击菜单图标时，默认的OnClickListener会打开菜单。您可以更改此行为，例如打开具有不同用户体验的菜单，如底部面板。显然，如果你想要一个与库提供的用户体验不同的用户体验，你有责任创建自己的组件。

Menu screenshot:

menu screenshot

#### YouTubePlayerMenu

您可以从PlayerUiController获取YouTubePlayerMenu的参考。

YouTubePlayerMenu PlayerUiController.getMenu()

一旦你得到一个YouTubePlayerMenu对象，你就可以向其中添加和删除项目，显示和关闭它。

YouTubePlayerMenu addItem(MenuItem menuItem)  
YouTubePlayerMenu removeItem(MenuItem menuItem)  
YouTubePlayerMenu removeItem(int itemIndex)  
  
void show(View anchorView)  
void dismiss()

最初的YouTubePlayerMenu不包含任何item。你需要添加它们。

#### MenuItem

MenuItems是YouTubePlayerMenu的item。它们有一个标题、一个可选图标和一个OnClickListener，当单击项目时会调用它。

### Reusable UI components

该库提供了一些预构建的UI组件，这些组件有助于减少构建自己的UI和控制器所需的时间。

从12.0.0版本开始，这些组件存在于自定义ui模块中。因此，您需要将其添加到应用程序的依赖项中才能使用它们。

#### YouTubePlayerSeekBar

此组件可用于显示和控制播放时间。它显示当前时间、视频的总持续时间和搜索栏。

YouTubePlayerSeekBar

您可以通过编程或xml将其添加到布局中。

<com.pierfrancescosoffritti.androidyoutubeplayer.core.customui.views.YouTubePlayerSeekBar  
 android:id="@+id/youtube\_player\_seekbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
  
 app:fontSize="12sp"  
 app:color="@color/red" />

可以使用fontSize和color属性更改字体大小和颜色。

YouTubePlayerSeekBar实现了YouTubeLayerListener。为了使其工作，您需要将其添加为YouTubePlayer对象的监听器。

youTubePlayer.addListener(youTubePlayerSeekBar);

您可能希望收听YouTubePlayer SeekBar的事件，以便在用户移动触摸栏时更新YouTubeLayer的当前时间。为此，请将YouTubeLayerSeekBarListener传递给YouTubePlayerSeekBar。

youTubePlayerSeekBar.setYoutubePlayerSeekBarListener(new YouTubePlayerSeekBarListener() {  
 @Override  
 public void seekTo(float time) {  
 youTubePlayer.seekTo(time);  
 }  
 });

#### FadeViewHelper

一个助手类，在不使用时自动淡出视图。它可用于自动淡入淡出玩家控件的容器，以便在适当的时候自动淡入淡出。

FadingFrameLayout是一个YouTubePlayerListener，因此它可以根据玩家的状态改变其行为。例如：如果视频暂停，它不会自动淡出。

您可以通过将要淡入淡出的视图传递给构造函数来初始化它。

FadeViewHelper fadeViewHelper = new FadeViewHelper(controlsContainer);

可以使用设置器方法更改动画持续时间和淡出延迟。

fadeViewHelper.setAnimationDuration(FadeViewHelper.DEFAULT\_ANIMATION\_DURATION);  
fadeViewHelper.setFadeOutDelay(FadeViewHelper.DEFAULT\_FADE\_OUT\_DELAY);

它们都以毫秒为单位。

为了使FadeViewHelper正常工作，您需要将其添加为YouTubePlayer对象的监听器。

youTubePlayer.addListener(fadeViewHelper);

使用FadeViewHelper.setDisabled（布尔值）方法禁用自动淡入淡出。

使用FadeViewHelper.toggleVisibility（）方法通过淡入淡出动画切换目标视图的可见性。

#### TimeUtilities

一组可用于格式化时间字符串（如视频的持续时间和当前时间）的实用程序

String TimeUtilities.formatTime(float timeInSeconds)

接收以秒为单位的时间，并返回一个时间格式为“M:SS”的字符串。（M=分钟，S=秒）。

## Network events

YouTubePlayerView使用内部BroadcastReceiver自动处理网络事件。您可以在初始化播放器时选择启用或禁用此功能，也可以通过设置xml属性app:handleNetworkEvents=“false”来启用或禁用。

使用内部BroadcastReceiver是处理网络事件的最简单和推荐的方法。该库能够处理连接断开且播放无法继续的情况，或者在播放器处于初始化过程中连接断开的情况。

如果你想使用自己的BroadcastReceiver，请确保涵盖所有可能的场景，以提供良好的用户体验。

# 播放器API

式样

微软雅黑 Light 5号

段落 固定值 23磅

# 目录1

## 目录2

### 目录3

#### 目录4

##### 目录5