使用ExoPlayer播放音频，可控制播放速度和音调

and2long 关注

0.712 2019.07.20 20:50:22 字数 347 阅读 3,612

日常开发中，我们一般使用MediaPlayer来快速实现音频的播放。
但是功能实在有限，最近遇到一个需求，需要控制音频的播放速度、音调。
用MediaPlayer只能干瞪眼，实在没辙，做不到啊。

ExoPlayer

> https://exoplayer.dev

> ExoPlayer is an application level media player for Android. It provides an alternative to Android's MediaPlayer API for playing audio and video both locally and over the Internet. ExoPlayer supports features not currently supported by Android's MediaPlayer API, including DASH and SmoothStreaming adaptive playbacks. Unlike the MediaPlayer API, ExoPlayer is easy to customize and extend, and can be updated through Play Store application updates.

简介中的这前两句话，明确说明ExoPlayer是用来替代MediaPlayer的。
看来以后再遇到音视频相关的功能，直接上ExoPlayer就好啦。

话不多说，直接进入主题：用ExoPlayer先播放个音频看看效果。

添加依赖，接入到项目中

> 这里不做赘述，直接看文档：https://exoplayer.dev/hello-world.html

播放个音频

```
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        btn_play.setOnClickListener {
            play(radioUrl = Uri.fromFile(File("/sdcard/care.mp3")))
        }
    }

    private fun play(radioUrl: Uri) {
        //创建exoPlayer实例
        val player = ExoPlayerFactory.newSimpleInstance(this)
        val dataSourceFactory = DefaultDataSourceFactory(
            this,
            Util.getUserAgent(this, application.packageName)
        )
        //创建mediaSource
        val mediaSource = ProgressiveMediaSource.Factory(dataSourceFactory)
            .createMediaSource(radioUrl)
        //准备播放
        player.prepare(mediaSource)
        //设置播放速度和音调均为2倍速
```

```
        player.playbackParameters = PlaybackParameters(2.0f, 2.0f)
        //资源加载后立即播放
        player.playWhenReady = true
    }
}
```

play()方法就简单实现了播放本地mp3文件，并且设置了播放速度和音调均为2倍。效果不错！

获取播放状态

```
val state = player.playWhenReady
```

控制播放和暂停

- 暂停播放

  `player.playWhenReady = false`

- 开始播放

  `player.playWhenReady = true`

获取媒体资源的时长

```
//给mediaSsource添加监听器，当媒体资源加载完成后，会回调onLoadCompleted方法
mediaSource?.addEventListener(mHandler, object : DefaultMediaSourceEventListener() {
            override fun onLoadCompleted(windowIndex: Int, mediaPeriodId: MediaSource.
                                         loadEventInfo: MediaSourceEventListener.LoadE
                                         mediaLoadData: MediaSourceEventListener.Media
                //移除监听器，避免重复回调
                mediaSource.removeEventListener(this)
                val duration = player.duration.toInt()
                seekBar.max = duration
            }
        })
```

监听播放器的状态

```
player.addListener(object : Player.EventListener {
                override fun onPlayerStateChanged(playWhenReady: Boolean, playbackState:
                    when (playbackState) {
                        Player.STATE_IDLE -> {
                            Log.i(TAG, "STATE_IDLE")
                        }
                        Player.STATE_BUFFERING -> {
                            Log.i(TAG, "STATE_BUFFERING")
                        }
                        Player.STATE_READY -> {
                            Log.i(TAG, "STATE_READY")
                        }
                        Player.STATE_ENDED -> {
                            Log.i(TAG, "STATE_ENDED")
                        }
                    }
                }
            })
```

控制循环播放

```
val loopResource =  LoopingMediaSource(mediaSource)
player.prepare(loopResource)
```