UA03. PROGRAMANDO...

3.7. Sonidos

Los sonidos de nuestra app pueden alertar al usuario de un mensaje o un estado especial. Una forma sencilla de generar un sonido es un generador de tonos. Veamos un ejemplo:

```
val tg = ToneGenerator(AudioManager.STREAM_NOTIFICATION, ToneGeneratc
override fun onResume() {
super.onResume()
tg.startTone(ToneGenerator.TONE_CDMA_ABBR_ALERT)
}
override fun onStop() {
super.onStop()
tg.release()
}
```

Creamos una instancia de la clase **ToneGenerator**, que sonará en el canal de notificaciones con el volumen máximo. Cuando la ventana se hace activa, llama a **onResume**, donde utilizamos el método **startTone** del generador con uno de los muchos tipos de sonidos de que dispone. Es una buena práctica liberar los recursos del generador una vez que no los necesitemos, por ejemplo, en **onStop**. Es imprescindible elegir adecuadamente en qué canal sonará el aviso, pues el sistema de audio de Android enrutará el tono que generemos en ese canal, mezclándolo con todos los sonidos que lleguen de otras fuentes.

Quizá los sonidos disponibles en el generador de tonos sean demasiado simples para el efecto que buscamos. Si necesitamos sonidos más complejos, utilizaremos archivos que

hayamos creado con algún editor de audio o descargado de internet. Para reproducir varios sonidos de forma rápida y eficiente, lo mejor es utilizar un **SoundPool**. Este objeto guardará en memoria los sonidos y los reproducirá cuando se lo pidamos, mezclándolos en el canal correspondiente.

Para reproducir un fichero de audio tendremos que seguir una secuencia de pasos. En primer lugar deberemos crear una instancia de la clase MediaPlayer e indicar qué fichero será el que se reproducirá. Tenemos dos opciones para hacer esto.

1. Inicializar la reproducción multimedia es por medio del método **setDataSource**, el cual asigna una fuente multimedia a una instancia ya existente de la clase **MediaPlayer**.

```
MediaPlayer mediaPlayer = new MediaPlayer();
mediaPlayer.setDataSource("/sdcard/test.mp3");
mediaPlayer.prepare();
```

Al instanciar la clase MediaPlayer se encontrará en estado idle. En este estado lo primero que debemos hacer es indicar el fichero a reproducir. Una vez hecho esto pasa a estado inicializado. En este estado ya sabe qué fichero ha de reproducir, pero todavía no se ha preparado para ello (inicializar bufferes, etc), por lo que no podrá comenzar la reproducción. Para prepararlo deberemos llamar al método prepare(), con lo que tendremos el reproductor listo para empezar a reproducir el audio.

2. Crear una instancia de la clase **MediaPlayer** por medio del método **create**. En este caso se deberá pasar como parámetro, además del contexto de la aplicación, el identificador del recurso, como se puede ver en el siguiente ejemplo:

```
Context appContext = getApplicationContext();

// Recurso de la aplicación
MediaPlayer resourcePlayer =
    MediaPlayer.create(appContext, R.raw.my_audio);
// Fichero local (en la tarjeta de memoria)
MediaPlayer filePlayer =
```

```
        MediaPlayer.create(appContext, Uri.parse("file:///sdcard/localfil
    // URL
    MediaPlayer urlPlayer =
        MediaPlayer.create(appContext, Uri.parse("http://site.com/audio/a
    // Proveedor de contenido
    MediaPlayer contentPlayer =
        MediaPlayer.create(appContext, Settings.System.DEFAULT_RINGTONE_U
```

En este caso el método create() se encarga de asignar la fuente de audio y además pasar el reproductor a estado preparado. Por lo tanto, en este caso no será necesario llamar a prepare(), sino que podremos reproducir el medio directamente. Aunque es más sencillo que la primera opción, también resulta menos flexible.

Más información

https://developer.android.com/guide/topics/media/mediaplayer?hl=es-419
<https://developer.android.com/guide/topics/media/mediaplayer?hl=es-419>