



APIs I

Curso: Desarrollando aplicaciones con Android

Contenidos

▶ Elementos multimedia

▶ Audio

- ▶ La clase `MediaPlayer`
- ▶ Reproducir audio
- ▶ Grabar audio

▶ Vídeo

- ▶ Reproducir vídeo

▶ Localización

- ▶ Obtener proveedores de localización
- ▶ Obtener coordenadas de posición
- ▶ Simular datos de localización



Audio

- ▶ Los archivos de audio pueden obtenerse desde distintas fuentes:
 - ▶ Directorio `/res/raw` (recursos de audio de la aplicación)
 - ▶ Archivos en el sistema (almacenados en el móvil)
 - ▶ Archivos en una URL (almacenados en una web)
- ▶ Los formatos de audio que reproduce Android son: AAC LC/LTP, HE-AACv1 (AAC+), HE-AACv2 (enhanced AAC+), AMR-NB, AMR-WB, MP3, MIDI, Ogg Vorbis, PCM/WAVE.

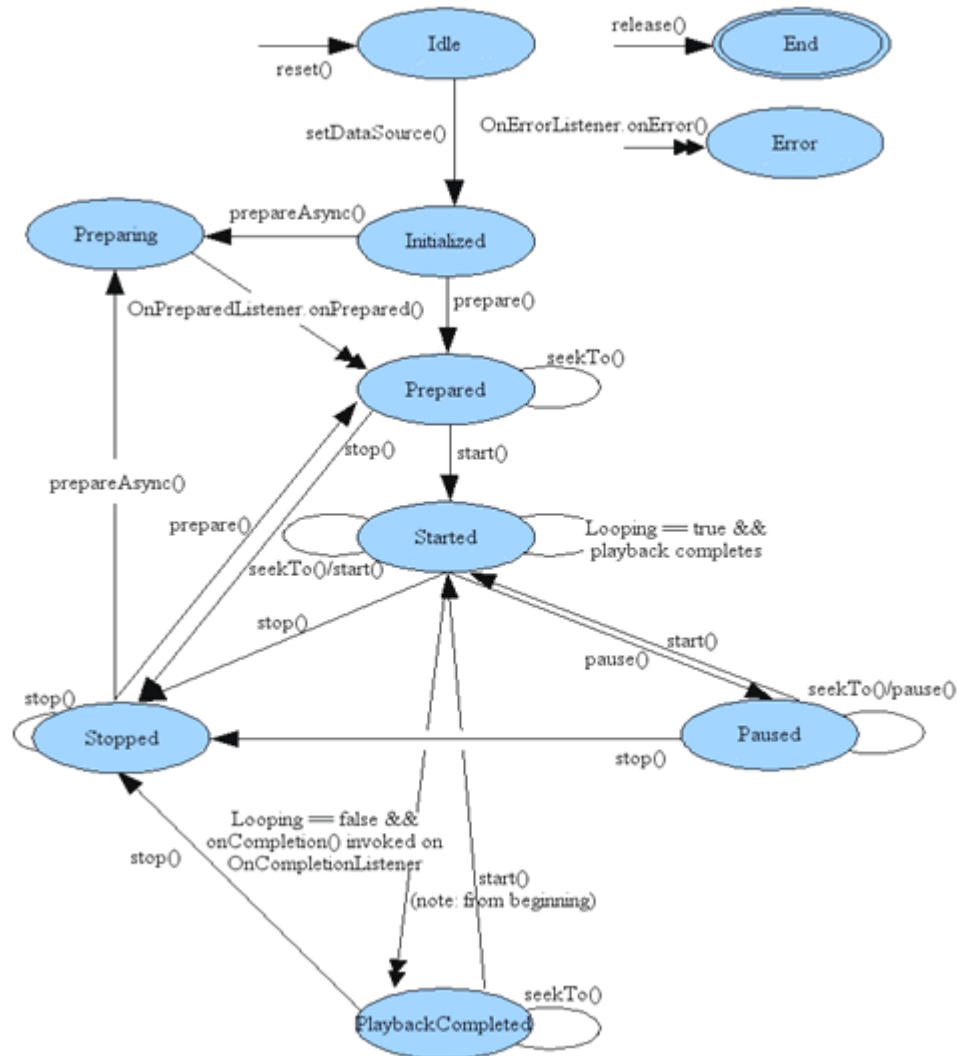


La clase MediaPlayer

- ▶ La clase `MediaPlayer` se utiliza para reproducir audio de una forma simple y directa. Permite pausar y parar la reproducción.
- ▶ El ciclo de vida de una reproducción en Android, que no contenga errores, pasa por los siguientes estados:
 - ▶ IDLE (Instancia creada)
 - ▶ INITIALIZED (Instancia informada de cuál es el archivo para reproducir)
 - ▶ PREPARED (La instancia tiene toda la información que necesita para comenzar la reproducción)
 - ▶ STARTED (Instancia reproduciendo)
 - ▶ STOPPED (Instancia finalizada)



La clase MediaPlayer



Reproducir audio

- ▶ Para reproducir audio desde el directorio `/res/raw` de nuestro proyecto debemos:
 - ▶ Crear un directorio `raw` dentro de `/res` (directorío de recursos de nuestra aplicación) y poner el archivo multimedia dentro del directorio `/res/raw` de nuestro proyecto.
 - ▶ El plugin ADT de eclipse lo encontrará y lo convertirá en un recurso que se puede referenciar desde la clase R.
 - ▶ Crear una instancia de `MediaPlayer`, usar `MediaPlayer.create` para referenciar el recurso multimedia y llamar al método `start`.

```
MediaPlayer mp = new MediaPlayer();  
mp = MediaPlayer.create(this, R.raw.misonido1);  
mp.start();
```



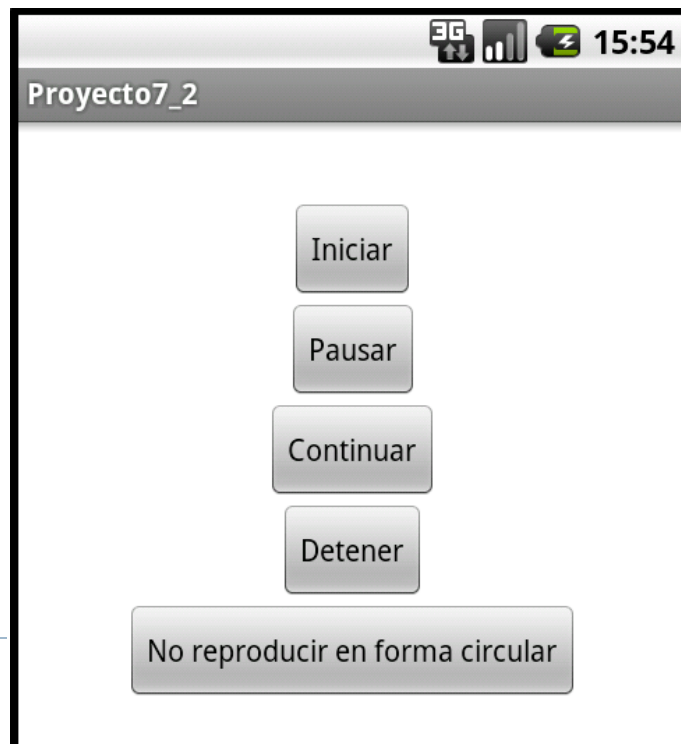
Ejercicio 5.1. Reproducción de audio

- ▶ Disponer dos botones con las etiquetas: Gato y León. Cuando se presione, reproducir el archivo de audio respectivo.
- ▶ Los archivos de audio deben almacenarlos en la misma aplicación.



Ejercicio 5.2. Reproducción, pausa, continuación y detención de un archivo de audio

- ▶ Desarrollar una aplicación que permita 1) iniciar un archivo mp3, 2) pausar, 3) continuar, detener y activación o no la reproducción en forma circular.
- ▶ <http://developer.android.com/reference/android/media/MediaPlayer.html>



Reproducir audio

- ▶ También podemos reproducir audio desde los archivos del sistema o desde una URL siguiendo los siguientes pasos:
 - ▶ Crear una instancia de `MediaPlayer`.
 - ▶ Llamar al método `setDataSource` con una cadena de texto que contenga la ruta del archivo o la URL del archivo que queremos reproducir. (Window → Show View → File Explorer)
 - ▶ Llamar a los métodos `prepare`, dentro de una excepción y `start` para que comience la reproducción.

```
MediaPlayer mp = new MediaPlayer();
try {
    mp.setDataSource("/mnt/sdcard/misonido1.mp3");
    mp.prepare ();
} catch (IOException e) {}
mp.start();
```

Grabar audio

- ▶ La plataforma Android nos permite grabar audio y vídeo por medio del hardware disponible en el dispositivo móvil que estemos utilizando (micrófono y cámara).
- ▶ Para grabar audio o vídeo en Android se utiliza la clase `MediaRecorder`.
- ▶ Pasos
 1. Poner los siguientes permisos en el `Androidmanifest.xml` para
 2. poder grabar audio y poder guardar el archivo que grabemos:

```
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```
 3. Crear una instancia de `MediaPlayer`.
 4. Añadir la fuente del audio, el formato y la codificación del mismo con `setAudioSource` y `setAudioEncoder`.
 5. Indicar en que ruta se guardará el sonido grabado con `setOutputFile`.
 6. Indicar los segundos de grabación con `setMaxDuration`.
 7. Llamar a los métodos `prepare`, dentro de una excepción y `start` para comenzar la grabación.



Grabar audio

```
MediaRecorder mRecorder = new MediaRecorder();
mRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);
mRecorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
mRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
mRecorder.setOutputFile("/mnt/sdcard/sonido1.amr");
mRecorder.setMaxDuration(4000);
try {
    mRecorder.prepare();
} catch (IOException e) {}
mRecorder.start();

//Se recomienda usar
//Environment.getExternalStorageDirectory().getPath();
//Para obtener la url de la carpeta de datos
```



Ejercicio 5.3 Grabar audio

- ▶ Disponer tres objetos de la clase `Button` con las etiquetas "Grabar", "Detener" y "Reproducir".
- ▶ Disponer además un `TextView` para informar del estado actual.
- ▶ Cuando se presione el botón "Grabar" permitir registrar todos los sonidos hasta que se presione el botón "Detener". Cuando se presione el botón "Reproducir" emitir el archivo de audio previamente generado.



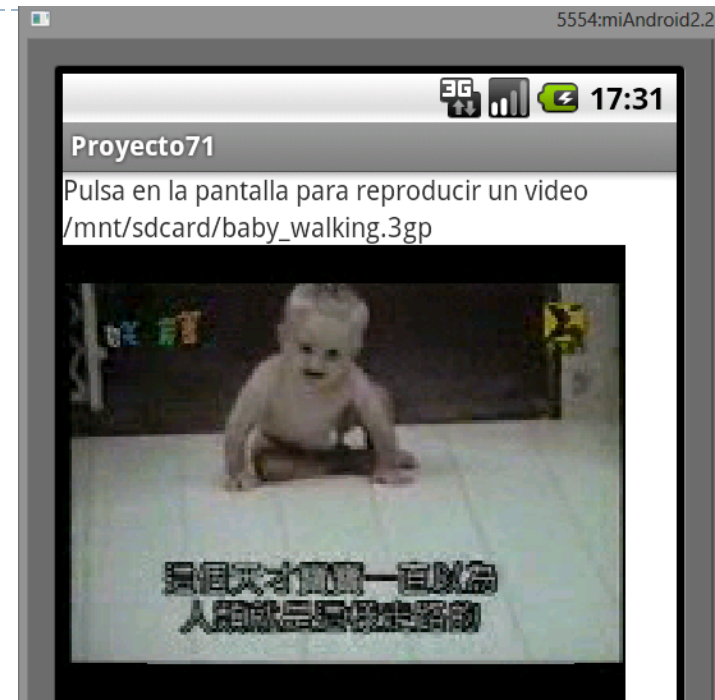
Video

- ▶ Los archivos de vídeo pueden obtenerse desde distintas fuentes:
 - ▶ Archivos en el sistema (almacenados en el móvil)
 - ▶ Archivos de una URL (almacenados en una web)
- ▶ Los formatos de vídeo que soporta Android para reproducir son: H.263, H.264 AVC, MPEG-4 SP.
- ▶ <http://developer.android.com/guide/appendix/media-formats.html>
- ▶ Con el control VideoView podemos mostrarlo.



Ejercicio 5.4 - Reproducir video

- ▶ Desarrollar un programa que reproduzca un video residente en la tarjeta SD
- ▶ Tendremos que pulsar en la pantalla para ver algo parecido a la figura en el AVD.



```
VideoView videoView = (VideoView) findViewById(R.id.videoView1);  
videoView.setVideoPath("mnt/sdcard/movie.mp4");  
//videoView.setVideoPath("http://www.ebookfrenzy.com/android_book/movie.mp4");  
videoView.start();
```

Localización

- ▶ Android nos ofrece la posibilidad de obtener y gestionar coordenadas y otros parámetros de posición dentro de las aplicaciones que desarrollemos.
- ▶ Android dispone de un servicio que hace posible localizar nuestras coordenadas de posición llamado `LocationManager`.
- ▶ Con el servicio `LocationManager` podemos obtener la lista de proveedores de localización (`LocationProviders`) de los que dispone nuestro dispositivo móvil.
- ▶ Eligiendo un proveedor de localización podremos obtener los siguientes parámetros entre otros: latitud, longitud, altitud, velocidad, orientación y tiempo.



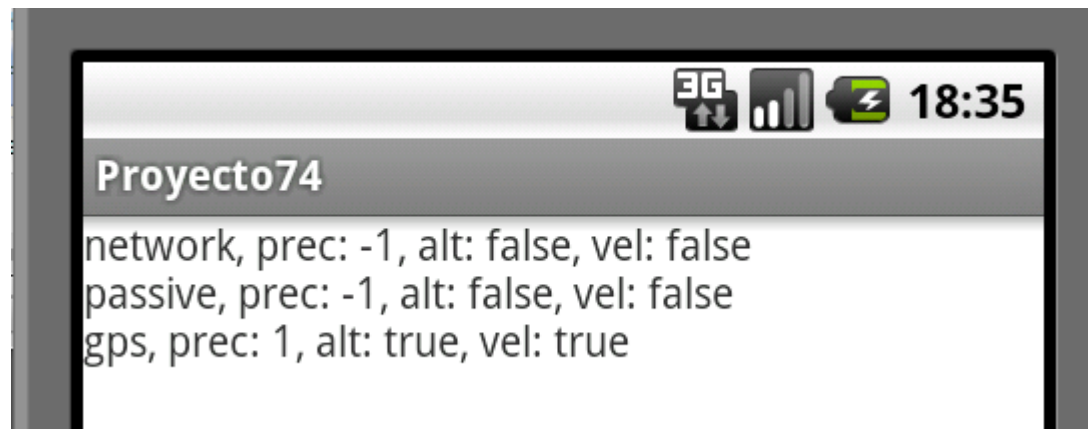
Localización

- ▶ No todos los dispositivos poseen los mismos proveedores de localización. Pueden tener uno o varios de los siguientes:
 - ▶ **GPS:** Responde a las siglas Global Position System y es un proveedor que determina la localización utilizando satélites.
 - ▶ **Network:** Este proveedor determina la localización basándose en antenas y puntos de acceso WIFI.
 - ▶ **Passive:** Este proveedor puede ser utilizado para recibir pasivamente actualizaciones de localización cuando otras aplicaciones o servicios las solicitan, sin tener que solicitarlas por si mismo.



Ejercicio 5.5.a Obtener proveedores de localización

- ▶ Obtener los proveedores de localización de nuestro dispositivo móvil (o AVD) y las características de cada uno, en este caso obtendremos:
 - ▶ El nombre del proveedor
 - ▶ La precisión que tiene
 - ▶ Si detecta la altitud a la que estamos
 - ▶ Si detecta la velocidad a la que nos movemos



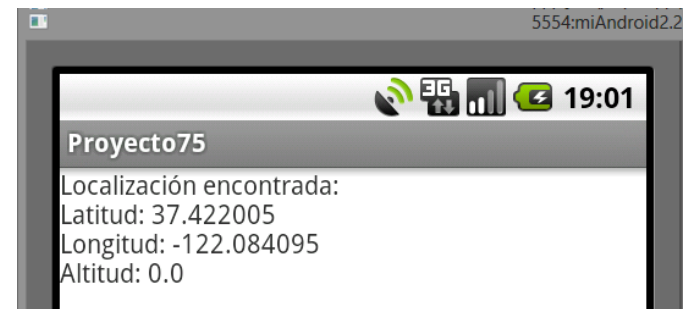
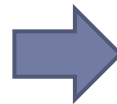
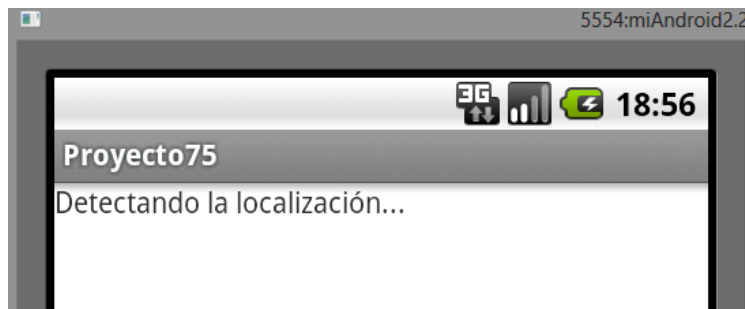
5.5.b Proveedores de localización

```
protected void onCreate(Bundle savedInstanceState) {  
    ...  
    tV = (TextView)findViewById(R.id.textView1);  
  
    LocationManager IM = (LocationManager) getSystemService(LOCATION_SERVICE);  
    List<String> listProviders = IM.getAllProviders();  
    String listado = "";  
    LocationProvider provider;  
    for (int i = 0; i < listProviders.size(); i++) {  
        provider = IM.getProvider(listProviders.get(i));  
        listado = listado + provider.getName() + ", prec: " + provider.getAccuracy() + ", alt: "  
        + provider.supportsAltitude() + ", vel: " + provider.supportsSpeed() + "\n";  
    }  
  
    tV.setText(listado);  
}
```



Ejercicio 5.6a – Obtener coordenadas de posición

- ▶ Desarrollar una aplicación en la que obtengamos las coordenadas de posición (latitud, longitud y altitud) desde un dispositivo móvil.
- ▶ Si no podemos obtener estas coordenadas reales, podremos simularlas en el AVD.



5.6.b Implementar MyLocationListener

```
private TextView tV;  
private LocationManager IM;  
private LocationListener mLocationListener;  
...  
protected void onCreate(Bundle savedInstanceState) {  
...  
    tV = (TextView)findViewById(R.id.textV1);  
    IM = (LocationManager)getSystemService(LOCATION_SERVICE);  
    mLocationListener = new MyLocationListener();  
    IM.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0, mLocationList  
}  
private class MyLocationListener implements LocationListener  
    {  
        @Override  
        public void onLocationChanged(Location loc) {  
            // TODO Auto-generated method stub  
            if (loc != null) {  
                tV.setText("Localización encontrada: \nLatitud: " + loc.getLatitude()  
                    + " \nLongitud: " + loc.getLongitude() + "\nAltitud: " +  
                    loc.getAltitude());  
            }  
        }  
    }  
}
```

...

Simular datos de localización

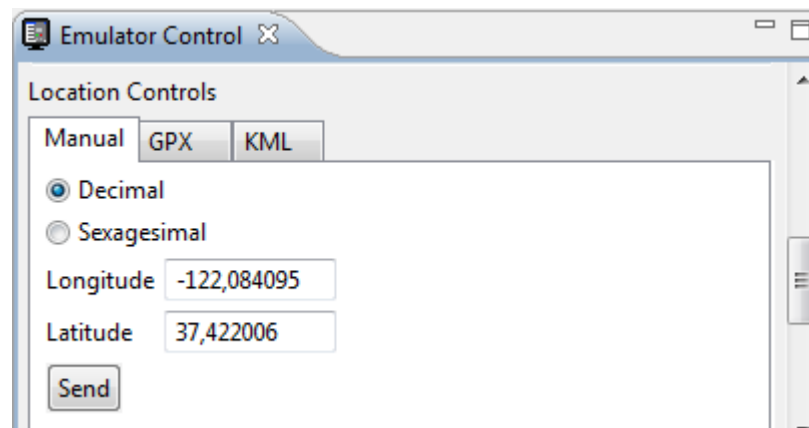
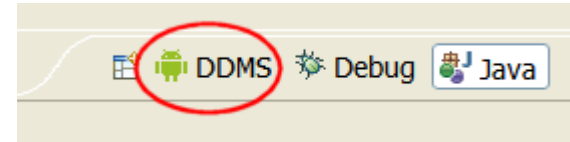
- ▶ En Eclipse podemos simular datos de localización ...:
 - ▶ Usando la perspectiva DMMS (Dalvik Debug Monitor Service):
 - ▶ **Enviando las coordenadas de longitud y latitud manualmente.**
 - ▶ Utilizando un archivo GPX, describiendo una ruta.
 - ▶ Utilizando un archivo KML describiendo marcas individuales.
 - ▶ Con la opción “geo” en la línea de comandos.



Simular datos de localización

► DDMS

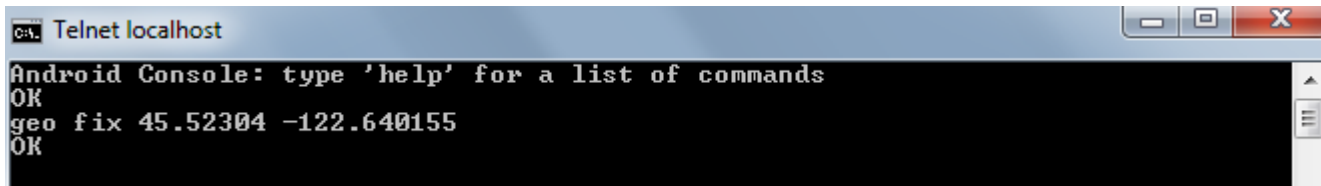
1. Cambiar a la perspectiva DDMS
2. Bajo la pestaña Emulator Control, podremos introducir la latitud y la longitud en sistema decimal o en sistema sexagesimal.
3. Cuando tengamos estos datos introducidos debemos pulsar el botón Send para enviárselos a nuestra aplicación.



Simular datos de localización

► Comando `geo`

1. Abrir una consola y ejecutar “abd devices” en el directorio platform-tools, donde tengamos nuestra instalación de Android, para conocer los números asignados a los AVDs que tengamos creados en Eclipse.
2. Ejecutar “telnet localhost 5554”.
 - ‘5554’ sería el número de nuestro terminal elegido.
3. Nos encontraremos en una consola de Android donde
4. podremos ejecutar “geo fix latitud longitud” para enviar a nuestra aplicación las coordenadas de latitud y longitud.



```

C:\> Telnet localhost
Android Console: type 'help' for a list of commands
OK
geo fix 45.52304 -122.640155
OK

```