Reproducción de vídeo usando el control VideoView

La reproducción de vídeo es muy similar a la reproducción de audio, salvo dos particularidades. En primer lugar, no es posible reproducir un clip de vídeo almacenado como parte de los recursos de la aplicación. En este caso deberemos utilizar cualquiera de los otros tres medios (ficheros locales, streaming o proveedores de contenidos). Un poco más adelante veremos cómo añadir un clip de vídeo a la tarjeta de memoria de nuestro terminal emulado desde la propia interfaz de Eclipse. En segundo lugar, el vídeo necesitará de una superficie para poder reproducirse. Esta superficie se corresponderá con una vista dentro del layout de la actividad.

Existen varias alternativas para la reproducción de vídeo, teniendo en cuenta lo que acabamos de comentar. La más sencilla es hacer uso de un control de tipo *VideoView*, que encapsula tanto la creación de una superficie en la que reproducir el vídeo como el control del mismo mediante una instancia de la clase `MediaPlayer`. Este método será el que veamos en primer lugar.

El primer paso consistirá en añadir el control *VideoView* a la interfaz gráfica de la actividad en la que queramos que se reproduzca el vídeo. Podemos añadir algo como lo siguiente el fichero de layout correspondiente:

```
<VideoView android:id="@+id/superficie"
          android:layout_height="fill_parent"
          android:layout_width="fill_parent">
</VideoView>
```

Dentro del código Java podremos acceder a dicho elemento de la manera habitual, es decir, mediante el método `findViewById()`. Una vez hecho esto, asignaremos una fuente que se corresponderá con el contenido multimedia a reproducir. El control *VideoView* se encargará de la inicialización del objeto `MediaPlayer`. Para asignar un video a reproducir podemos utilizar cualquiera de estos dos métodos:

```
videoView1.setVideoUri("http://www.mysite.com/videos/myvideo.3gp");
videoView2.setVideoPath("/sdcard/test2.3gp");
```

Una vez inicializado el control se puede controlar la reproducción con los métodos `start()`, `stopPlayback()`, `pause()` y `seekTo()`. La clase `VideoView` también incorpora el método `setKeepScreenOn(boolean)`con la que se podrá controlar el comportamiento de la iluminación de la pantalla durante la reproducción del clip de vídeo. Si se pasa como parámetro el valor `true` ésta permanecerá constantemente iluminada.

El siguiente código muestra un ejemplo de asignación de un vídeo a un control *VideoView* y de su posterior reproducción. Dicho código puede ser utilizado a modo de esqueleto en nuestra propia actividad. También podemos ver un ejemplo de uso de `seekTo()`, en este caso para avanzar hasta la posición intermedia del video.

```
VideoView videoView = (VideoView)findViewById(R.id.superficie);
videoView.setKeepScreenOn(true);
videoView.setVideoPath("/sdcard/ejemplo.3gp");

if (videoView.canSeekForward())
    videoView.seekTo(videoView.getDuration()/2);

videoView.start();

// Hacer algo durante la reproducción

videoView.stopPlayback();
```

En esta sección veremos en último lugar, tal como se ha indicado anteriormente, la manera de añadir archivos a la tarjeta de memoria de nuestro dispositivo virtual, de tal forma que podamos almacenar clips de vídeo y resolver los ejercicios propuestos para la sesión. Se deben seguir los siguientes pasos:

En primer lugar el emulador debe encontrarse en funcionamiento, y por supuesto, el dispositivo emulado debe hacer uso de una tarjeta SD.

A continuación, seleccionamos la pestaña *File Explorer*. El contenido de la tarjeta de memoria se halla en la carpeta */mnt/sdcard/*.

En de dicha carpeta deberemos introducir nuestros archivos de vídeo, dentro del directorio *DCIM*. Al hacer esto ya podrán reproducirse desde la aplicación nativa de reproducción de vídeo y también desde nuestras propias aplicaciones. Podemos introducir un archivo de video con el ratón, arrastrando un fichero desde otra carpeta al interior de la carpeta *DCIM*, aunque también podemos hacer uso de los controles que aparecen en la parte superior derecha de la perspectiva *DDMS*, cuando la pestaña *File Explorer* está seleccionada. La función de estos botones es, respectivamente: guardar en nuestra máquina real algún archivo de la tarjeta de memoria virtual, guardar en la tarjeta de memoria virtual un archivo, y eliminar el archivo seleccionado.

A veces es necesario volver a arrancar el terminal emulado para poder acceder a los vídeos insertados siguiendo este método en la tarjeta de memoria desde la aplicación *Galería de Android*.

Reproducción de vídeo basada en MediaPlayer

La segunda alternativa para la reproducción de video consiste en la creación de una superficie sobre la que dicho vídeo se reproducirá y en el uso directo de la clase `MediaPlayer`. La superficie deberá ser asignada a la instancia correspondiente de dicha clase. En caso contrario el vídeo no se mostrará. Además, la clase `MediaPlayer` requiere que la superficie sea un objeto de tipo `SurfaceHolder`.

Para añadir una vista de tipo *SurfaceHolder* a la interfaz gráfica de la actividad debemos incluir una vista `SurfaceView` en el archivo de layout correspondiente

```
<SurfaceView
    android:id="@+id/superficie"
    android:layout_width="200px"
    android:layout_height="200px"
    android:layout_gravity="center">
</SurfaceView>
```

El siguiente paso será la inicialización el objeto `SurfaceView` y la asignación del mismo a la instancia de la clase `MediaPlayer` encargada de reproducir el vídeo. El siguiente código muestra cómo hacer esto. Obsérvese que es necesario que la actividad implemente la interfaz `SurfaceHolder.Callback`. Esto es así porque los objetos de la clase `SurfaceHolder` se crean de manera asíncrona, por lo que debemos añadir un mecanismo que permita esperar a que dicho objeto haya sido creado antes de poder empezar a reproducir el vídeo.

```
// La clase debe implementar la interfaz SurfaceHolder.Callback
public class MiActividad extends Activity implements SurfaceHolder.Callback
{
    private MediaPlayer mediaPlayer;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        mediaPlayer = new MediaPlayer();
        SurfaceView superficie = (SurfaceView)findViewById(R.id.superficie);

        // Hacemos que la actividad maneje los eventos del SurfaceHolder
        SurfaceHolder holder = superficie.getHolder();
        holder.addCallback(this);
        holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    }

    // No comenzamos la reproducción hasta que no se cree
    // la superficie
    public void surfaceCreated(SurfaceHolder holder) {
        try {
            mediaPlayer.setDisplay(holder);
            // Reproducir vídeo a continuación
        } catch (IllegalArgumentException e) {
            Log.d("MEDIA_PLAYER", e.getMessage());
        } catch (IllegalStateException e) {
            Log.d("MEDIA_PLAYER", e.getMessage());
        }
    }

    // Liberamos la memoria de la instancia de MediaPlayer
    // cuando se vaya a destruir la superficie
    public void surfaceDestroyed(SurfaceHolder holder) {
        mediaPlayer.release();
    }

    public void surfaceChanged(SurfaceHolder holder, int format, int width,
    int height) { }
    }
}
```

Una vez que hemos asociado la superficie al objeto de la clase `MediaPlayer` debemos asignar a dicho objeto el identificador del clip de vídeo a reproducir. Ya que habremos creado la instancia del objeto `MediaPlayer` previamente, la única posibilidad que tendremos será utilizar el método `setDataSource()`, como se muestra en el siguiente ejemplo. Recuerda que cuando se utiliza dicho método es necesario llamar también al método `prepare()`. En este ejemplo, se muestra la forma síncrona, pero recordad que si la preparación del video tarda demasiado, tenemos el método `prepareAsync()`, que lo gestiona de forma asíncrona.

```
public void surfaceCreated(SurfaceHolder holder) {
    try {
        mediaPlayer.setDisplay(holder);
        // Inicio de la reproducción una vez la superficie
        // ha sido asociada a la instancia de MediaPlayer
        mediaPlayer.setDataSource("/sdcard/prueba.3gp");
        mediaPlayer.prepare();
        mediaPlayer.start();
    } catch (IllegalArgumentException e) {
        Log.d("MEDIA_PLAYER", e.getMessage());
    } catch (IllegalStateException e) {
        Log.d("MEDIA_PLAYER", e.getMessage());
    } catch (IOException e) {
        Log.d("MEDIA_PLAYER", e.getMessage());
    }
}
```