Manejo de Video y Almacenamiento Interno en Android con Java

Este proceso delega la grabación del video a la aplicación de cámara del sistema mediante un **Intent**, y utiliza **FileProvider** para asegurar que el archivo resultante se guarde en el directorio privado de tu aplicación.

1. Permisos y Configuración Esencial

Para esta operación, solo necesitas el permiso para la cámara. El manejo de la Memoria Interna **no requiere** permisos de almacenamiento en *runtime*.

A. AndroidManifest.xml

XML

```
<uses-permission android:name="android.permission.CAMERA" />

<application ...>
    <provider
        android:name="androidx.core.content.FileProvider"
        android:authorities="${applicationId}.provider"
        android:exported="false"
        android:grantUriPermissions="true">
        <meta-data
            android:name="android.support.FILE_PROVIDER_PATHS"
            android:resource="@xml/file_paths" />
    </provider>
</application>
```

B. res/xml/file_paths.xml

El archivo de rutas debe contener el *path* para la Memoria Interna, igual que para las fotos:

XML

```xml
<?xml version="1.0" encoding="utf-8"?>
<paths>
    <files-path
        name="internal_files"
        path="/" />
</paths>
```

---

2. Preparar el Fichero de Destino en la Memoria Interna

Creamos un archivo temporal con extensión .mp4 (o similar) en el directorio devuelto por **getFilesDir()**.

Java

```java
// MainActivity.java
private String currentVideoPath; // Para almacenar la ruta absoluta del archivo

private File crearArchivoDeVideoInterno() throws IOException {
    // 1. Crea un nombre de archivo único para el video.
    String timeStamp = new SimpleDateFormat("yyyyMMdd_HHmmss", Locale.getDefault()).format(new Date());
    String videoFileName = "MP4_" + timeStamp + "_";

    // 2. Obtiene el directorio de destino: getFilesDir() para Memoria Interna.
    File storageDir = getFilesDir();

    // 3. Crea el archivo temporal con extensión .mp4.
    File video = File.createTempFile(
        videoFileName,  /* Prefijo */
```

```
        ".mp4",         /* Sufijo */
        storageDir      /* Directorio */
    );

    // Guarda la ruta para referencia posterior (reproducción o subida).
    currentVideoPath = video.getAbsolutePath();
    return video;
}
```

---

3. Llamar a la Grabadora de Video usando Intent

Utilizamos la acción **MediaStore.ACTION_VIDEO_CAPTURE** y le proporcionamos la URI de destino.

Java

```
private static final int REQUEST_VIDEO_CAPTURE = 3;

private void dispatchTakeVideoIntentInterno() {
    // 1. Verificar Permiso de CÁMARA aquí.

    Intent takeVideoIntent = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);

    // Asegúrate de que haya una aplicación de grabación de video.
    if (takeVideoIntent.resolveActivity(getPackageManager()) != null) {
        File videoFile = null;
        try {
            // Llama a la función que crea el archivo en Memoria Interna.
            videoFile = crearArchivoDeVideoInterno();
        } catch (IOException ex) {
```

```
            Log.e("VideoCapture", "Error creando archivo de video: " + ex.getMessage());
        }

        if (videoFile != null) {
            // 2. Obtiene la URI del archivo a través de FileProvider.
            Uri videoURI = FileProvider.getUriForFile(this,
                getApplicationContext().getPackageName() + ".provider",
                videoFile);

            // 3. Pasa la URI al Intent para que la cámara guarde allí el video.
            takeVideoIntent.putExtra(MediaStore.EXTRA_OUTPUT, videoURI);

            // Opcional: Limitar la duración o el tamaño del video (en segundos/bytes).
            // takeVideoIntent.putExtra(MediaStore.EXTRA_DURATION_LIMIT, 10); // Límite de 10s

            // 4. Lanza el Intent esperando un resultado.
            startActivityForResult(takeVideoIntent, REQUEST_VIDEO_CAPTURE);
        }
    } else {
        Toast.makeText(this, "No se encontró aplicación para grabar video.", Toast.LENGTH_SHORT).show();
    }
}
```

---

4. Manejar el Resultado y Reproducir el Video

Cuando el Intent regresa con éxito, el video está guardado de forma segura en currentVideoPath.

Java

```java
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == REQUEST_VIDEO_CAPTURE && resultCode == RESULT_OK) {
        // El video está guardado de forma privada en la ruta: currentVideoPath.

        // 1. Opcional: Muestra la ruta del archivo.
        Toast.makeText(this, "Video guardado en Memoria Interna.", Toast.LENGTH_LONG).show();

        // 2. Reproduce el video (requiere VideoView o MediaController).
        reproducirVideo(currentVideoPath);
    }
}

private void reproducirVideo(String path) {
    // Ejemplo de cómo cargar y reproducir el video en un VideoView
    // VideoView videoView = findViewById(R.id.mi_video_view);

    // Convertimos la ruta File a una URI que VideoView pueda usar
    Uri videoUri = Uri.parse(path);

    // videoView.setVideoURI(videoUri);
    // videoView.start();

    // Usar un MediaController para controles de reproducción es común.
    // MediaController mediaController = new MediaController(this);
```

```
    // mediaController.setAnchorView(videoView);
    // videoView.setMediaController(mediaController);
}
```

---

5. Consideraciones Clave para Videos

- **Tamaño del Archivo:** Los videos son grandes. Las operaciones de mover, copiar o subir videos deben realizarse **siempre en un hilo secundario** (como un ExecutorService) para no bloquear la interfaz de usuario.
- **Memoria Interna y Espacio:** Si grabas videos muy largos, la Memoria Interna puede llenarse rápidamente. Considera advertir al usuario si el espacio disponible es bajo o usar la Memoria Externa (con permisos) si el video debe ser muy largo.
- **Carga Asíncrona:** Si el video se va a reproducir inmediatamente, el VideoView maneja bien la carga desde la URI. Para previsualizaciones (miniaturas), usa **ThumbnailUtils** para generar miniaturas sin cargar todo el archivo.