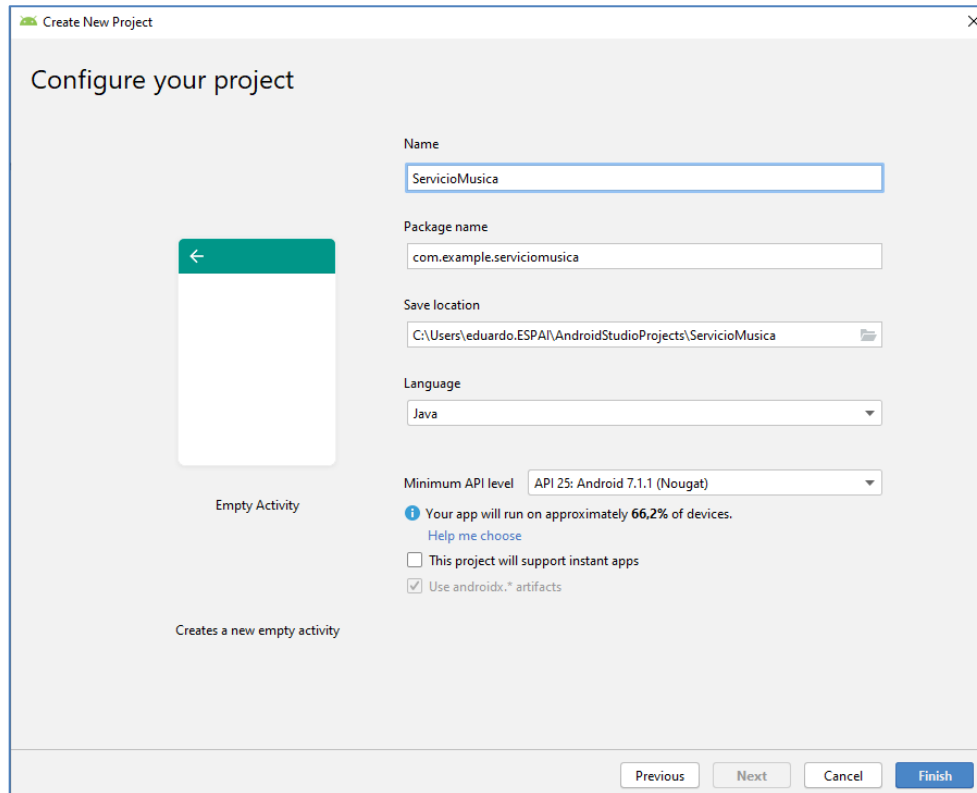


PRACTICA 15: SERVICIOMUSICA Y NOTIFICACIONES

Parte I: Un servicio para ejecución en segundo plano de reproducción de música.

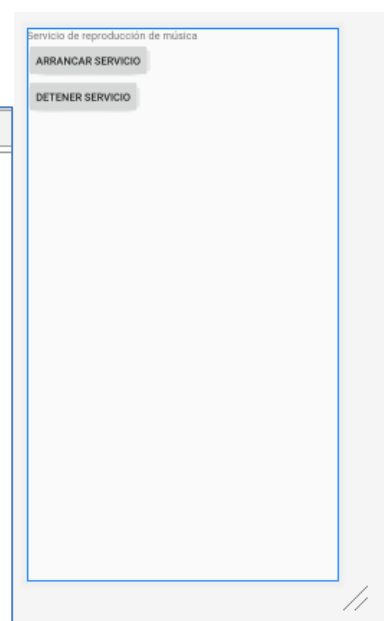
Veamos un ejemplo de servicio que corre en el mismo proceso de la aplicación que lo utiliza. El servicio será creado con la finalidad de reproducir una música de fondo y podrá ser arrancado y detenido desde la actividad principal.

Paso 1. Crea un nuevo proyecto con los siguientes datos:



Paso 2. Reemplaza el código del *layout* `activity_main.xml` por:

```
activity_main.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:orientation="vertical"
4     android:layout_width="fill_parent"
5     android:layout_height="fill_parent">
6     <TextView android:layout_width="fill_parent"
7         android:layout_height="wrap_content"
8         android:text="Servicio de reproducción de música"/>
9     <Button android:id="@+id/boton_arrancar"
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:text="Arrancar servicio"/>
13     <Button android:id="@+id/boton_detener"
14         android:layout_width="wrap_content"
15         android:layout_height="wrap_content"
16         android:text="Detener servicio"/>
17 </LinearLayout>
```



Se trata de un *layout* muy sencillo, con un texto y dos botones.

Paso 3. Reemplaza el código de la actividad por:

```
public class MainActivity extends AppCompatActivity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button arrancar = (Button) findViewById(R.id.boton_arrancar);
        arrancar.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                startService(new Intent( packageContext: MainActivity.this,
                    ServicioMusica.class));
            }
        });
        Button detener = (Button) findViewById(R.id.boton_detener);
        detener.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                stopService(new Intent( packageContext: MainActivity.this,
                    ServicioMusica.class));
            }
        });
    }
}
```

Paso 4. Crea la nueva clase, **ServicioMusica**, con el siguiente código:

```
public class ServicioMusica extends Service {
    MediaPlayer reproductor;
    @Override
    public void onCreate() {
        Toast.makeText( context: this, text: "Servicio creado", Toast.LENGTH_SHORT).show();
        reproductor = MediaPlayer.create(this, R.raw.audio);
    }

    @Override
    public int onStartCommand(Intent intenc, int flags, int idArranque) {
        Toast.makeText( context: this, text: "Servicio arrancado " + idArranque, Toast.LENGTH_SHORT).show();
        reproductor.start();
        return START_STICKY;
    }

    @Override
    public void onDestroy() {
        Toast.makeText( context: this, text: "Servicio detenido", Toast.LENGTH_SHORT).show();
        reproductor.stop();
    }

    @Override
    public IBinder onBind(Intent intencion) {
        return null;
    }
}
```

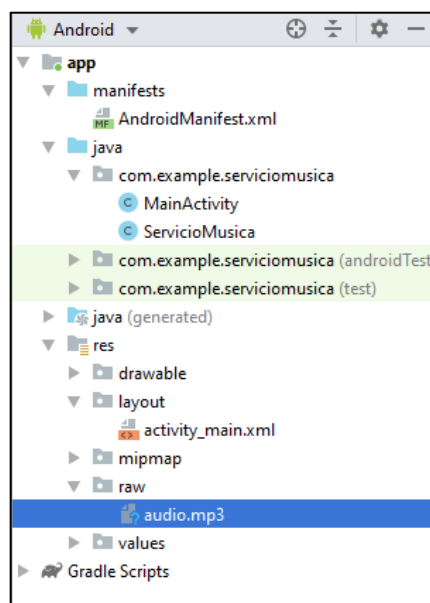
Paso 5. Edita el fichero `AndroidManifest.xml` y añade la siguiente línea dentro de la etiqueta `<application>`.

```
<service android:name=".ServicioMusica"/>
```

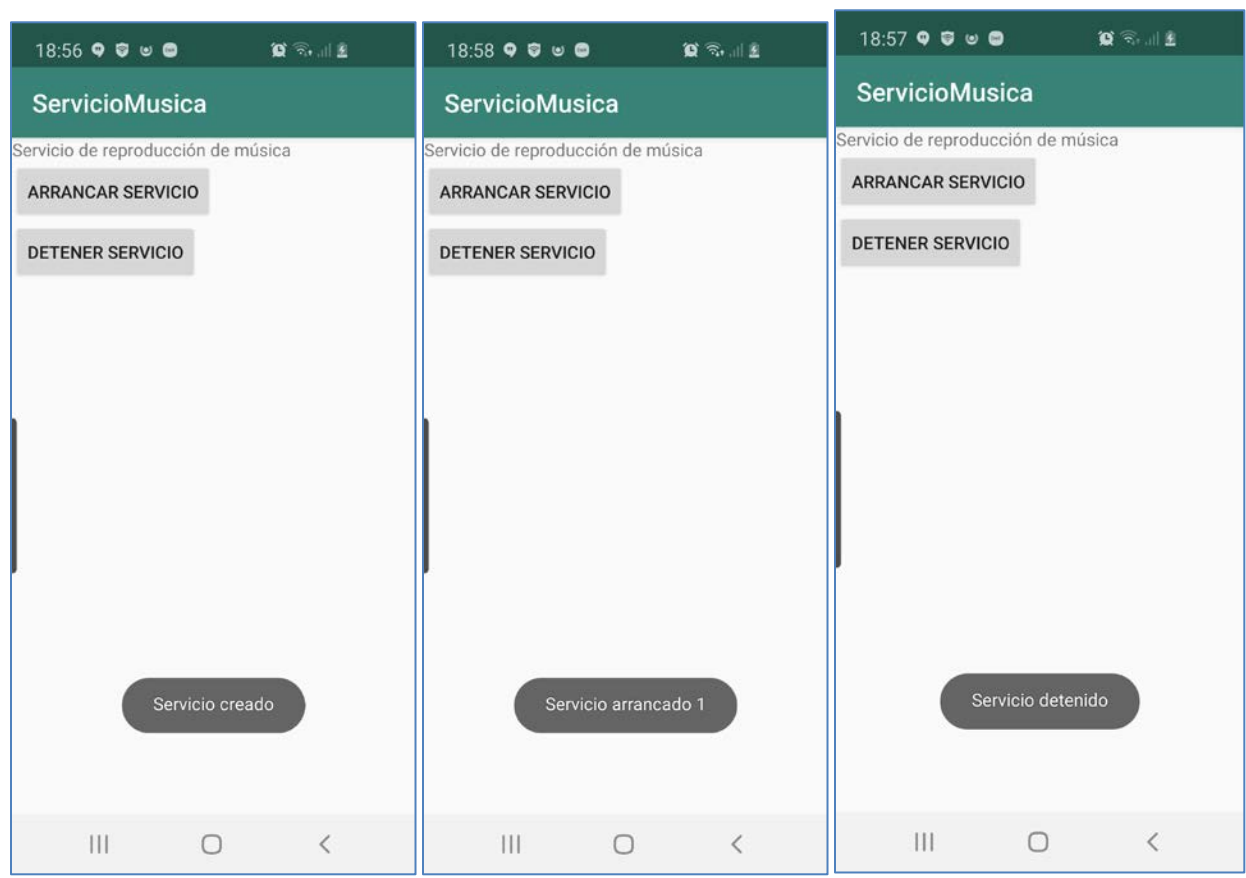


Paso 6. Crea una nueva carpeta con nombre `raw` dentro de la carpeta `res`. Arrastra a su interior el fichero `audio.mp3`.

NOTA: Puedes utilizar cualquier fichero de música compatible con Android siempre que el nombre de fichero sea `audio`.

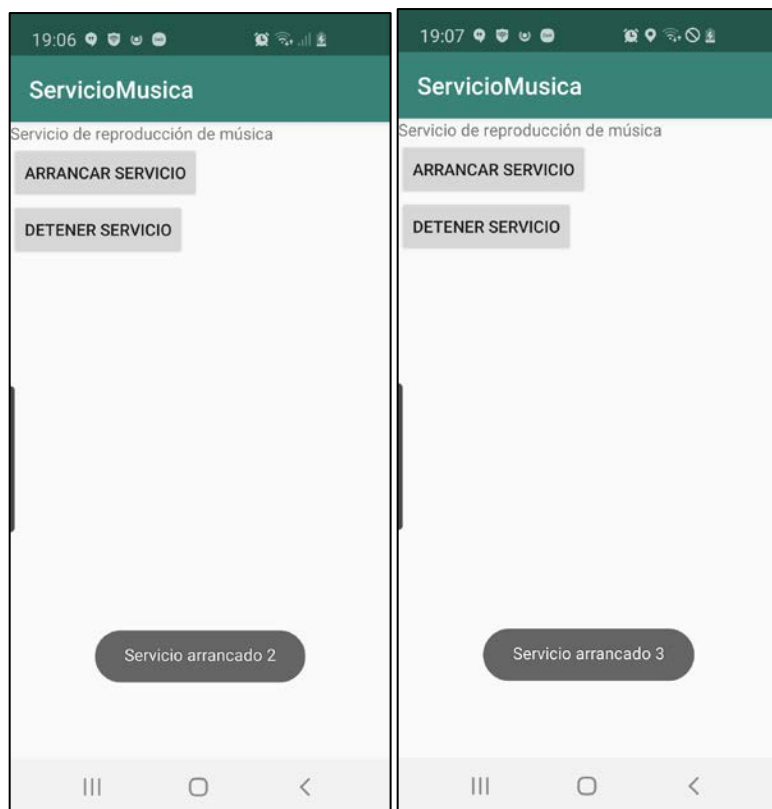


Paso 7. Ejecuta la aplicación y comprueba su funcionamiento. Puedes terminar la actividad pulsando el botón de retroceder y verificar que el servicio continúa en marcha.



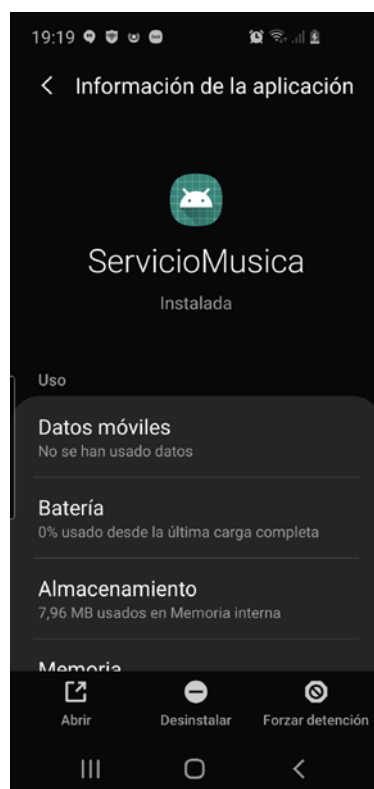
En esta versión de Android, al hacer click en el botón de detener servicio, el servicio no sigue activo y la música se para.

Paso 8. Verifica que aunque pulses varias veces el botón “Arrancar servicio”, este no vuelve a crearse, pero sí que vuelve a llamarse al método `onStartCommand()`. Además, con solo una vez que pulses en “Detener servicio” este parará.



Paso 9. Arranca la aplicación Ajustes / Aplicaciones / Servicios en ejecución / Servicio de Música. Desde aquí puedes obtener información y detener el servicio.

NOTA: Esta aplicación no está disponible en todas las versiones.



En esta versión de Android, no se encuentra disponible los servicios en ejecución.

Parte II: Los métodos onStartCommand() y onStart().

Paso 1. Comenta el en ejercicio anterior el método onStartCommand()

```
public class ServicioMusica extends Service {
    MediaPlayer reproductor;
    @Override
    public void onCreate() {
        Toast.makeText( context: this, text: "Servicio creado", Toast.LENGTH_SHORT).show();
        reproductor = MediaPlayer.create( context: this, R.raw.audio);
    }

    /*@Override
    public int onStartCommand(Intent intenc, int flags, int idArranque) {
        Toast.makeText(this, "Servicio arrancado " + idArranque, Toast.LENGTH_SHORT).show();
        reproductor.start();
        return START_STICKY;
    }*/
```

Paso 2. Añade el siguiente método onStartCommand:

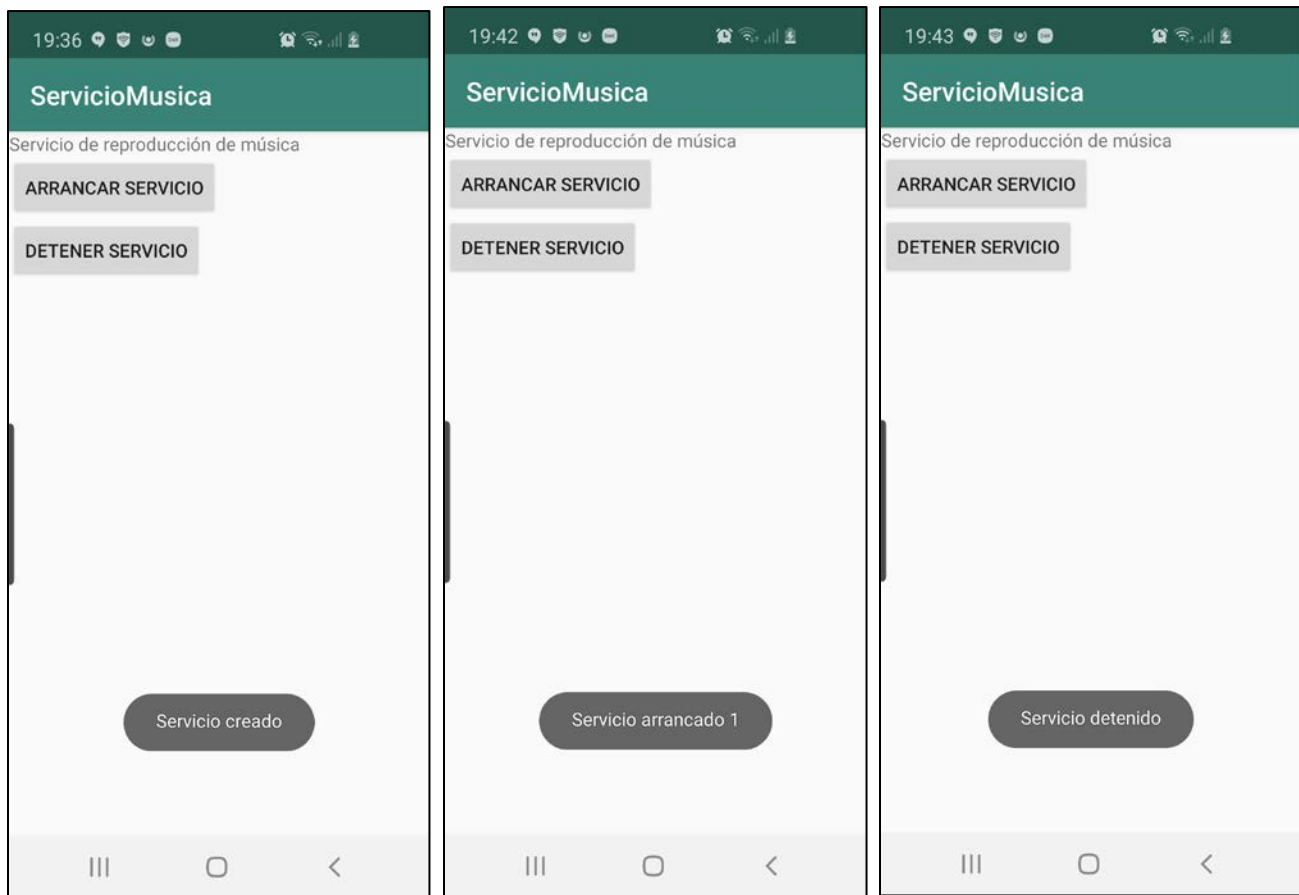
```
public class ServicioMusica extends Service {
    MediaPlayer reproductor;
    @Override
    public void onCreate() {
        Toast.makeText( context: this, text: "Servicio creado", Toast.LENGTH_SHORT).show();
        reproductor = MediaPlayer.create( context: this, R.raw.audio);
    }

    /*@Override
    public int onStartCommand(Intent intenc, int flags, int idArranque) {
        Toast.makeText(this, "Servicio arrancado " + idArranque, Toast.LENGTH_SHORT).show();
        reproductor.start();
        return START_STICKY;
    }*/

    @Override
    public void onStart(Intent intent, int startId) {
        Toast.makeText( context: this, text: "Servicio arrancado " + startId, Toast.LENGTH_SHORT).show();
        reproductor.start();
    }
}
```

En esta versión de Android, el método onStart aparece deprecated

Paso 3. Verifica que el programa funciona exactamente igual.



Paso 4. Comenta el método `onStart ()` y quita el comentario al método `onStartCommand()`.

```
public class ServicioMusica extends Service {
    MediaPlayer reproductor;
    @Override
    public void onCreate() {
        Toast.makeText( context: this, text: "Servicio creado", Toast.LENGTH_SHORT).show();
        reproductor = MediaPlayer.create( context: this, R.raw.audio);
    }

    @Override
    public int onStartCommand(Intent intenc, int flags, int idArranque) {
        Toast.makeText( context: this, text: "Servicio arrancado " + idArranque, Toast.LENGTH_SHORT).show();
        reproductor.start();
        return START_STICKY;
    }

    /*@Override
    public void onStart(Intent intent, int startId) {
        Toast.makeText(this, "Servicio arrancado " + startId, Toast.LENGTH_SHORT).show();
        reproductor.start();
    }*/
}
```

Parte III: Las notificaciones de la barra de estado

Paso 1. Abre el proyecto ServicioMusica.

Paso 2. Declara la siguiente constante al comienzo de la clase:

```
public class ServicioMusica extends Service {  
  
    private static final int ID_NOTIFICATION_CREAR = 1;  
  
    MediaPlayer reproductor;
```

Paso 3. Para crear una nueva notificación añade al principio del método onStartCommand las siguientes líneas:

```
@Override  
public int onStartCommand(Intent intenc, int flags, int idArranque) {  
    Toast.makeText(context, this, text: "Servicio arrancado "+ idArranque, Toast.LENGTH_SHORT).show();  
    reproductor.start();  
  
    NotificationCompat.Builder nm = new NotificationCompat.Builder(context, this)  
        .setSmallIcon(R.drawable.ic_launcher_background)  
        .setContentTitle("hola")  
        .setContentText("adios");  
    NotificationManager notificationManager =  
        (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);  
    notificationManager.notify(ID_NOTIFICATION_CREAR, nm.build());  
  
    return START_STICKY;  
}
```

La notificación se crea utilizando una clase especial Builder que dispone de varios métodos set para configurarla, setContentTitle() permite indicar el título que describe la notificación y setContentText(), información más detallada. Con setSmallIcon() indicamos el icono a visualizar. En el ejemplo usamos el mismo que el de la aplicación, aunque no resulta muy adecuado dado que estos iconos han de seguir una estética concreta.

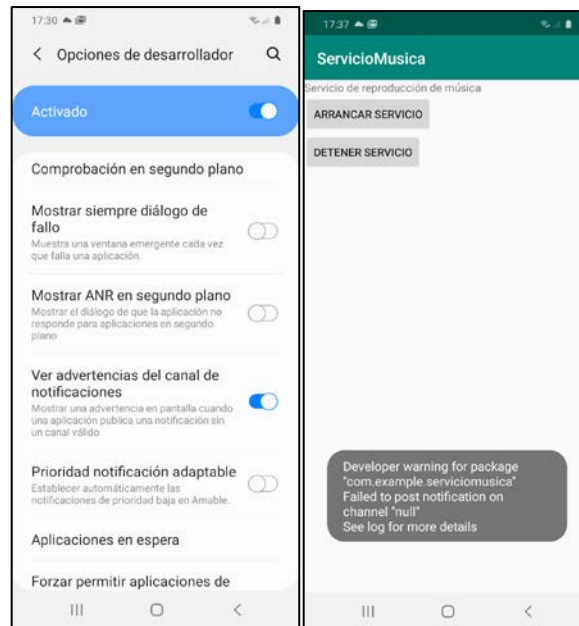
Para lanzar la notificación necesitamos una referencia al NotificationManger que permite manejar las notificaciones del sistema. El método notify es el encargado de lanzar la notificación. En el primer parámetro se indica un id para poder identificar esta notificación en un futuro y en el segundo la notificación.

NOTA: Una notificación puede tener otros parámetros, por ejemplo, puede reproducir un sonido, puede hacer vibrar el teléfono o puede hacer parpadear un LED del teléfono.

Paso 4. Ejecuta la aplicación y verifica el resultado.

NOTA: La clase NotificationCompat.Builder pertenece a la librería de compatibilidad v4. Si tu aplicación tiene un nivel mínimo de API 16 (v4.1) o superior ya no es necesario la librería de compatibilidad. En este caso utiliza la clase Notification.Builder.

No aparece nada. Tampoco ocurre ningún error. Activamos las advertencias del canal de notificaciones, y aparece un mensaje.



Paso 5. En las más nuevas versiones de Android hace falta crear un canal de notificaciones para que aparezcan correctamente.

```
@Override
public int onStartCommand(Intent intent, int flags, int idArranque) {
    Toast.makeText(context, this, text: "Servicio arrancado" + idArranque, Toast.LENGTH_SHORT).show();
    reproductor.start();

    NotificationManager notificationManager =
        (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
    String CHANNEL_ID = "my_channel_01";

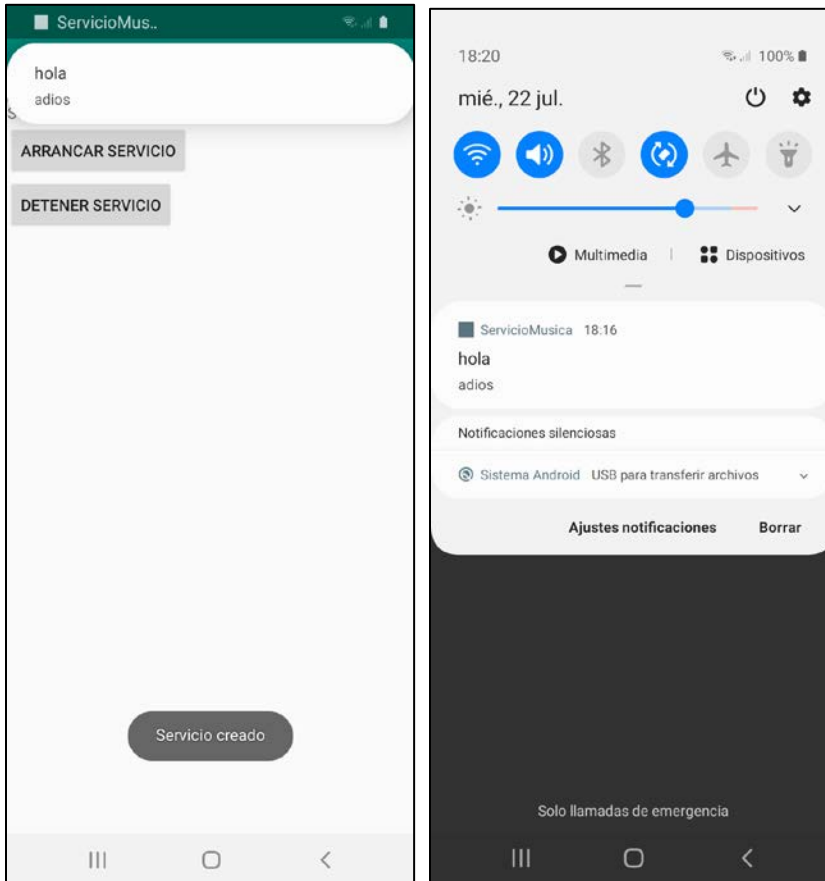
    if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
        CharSequence name = "my_channel";
        String Description = "This is my channel";
        int importance = NotificationManager.IMPORTANCE_HIGH;
        NotificationChannel mChannel = new NotificationChannel(CHANNEL_ID, name, importance);
        mChannel.setDescription(Description);
        mChannel.enableLights(true);
        mChannel.setLightColor(Color.RED);
        mChannel.enableVibration(true);
        mChannel.setVibrationPattern(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});
        mChannel.setShowBadge(false);
        notificationManager.createNotificationChannel(mChannel);
    }

    NotificationCompat.Builder nm = new NotificationCompat.Builder(context, CHANNEL_ID)
        .setSmallIcon(R.drawable.ic_launcher_background)
        .setContentTitle("hola")
        .setContentText("adios");

    notificationManager.notify(ID_NOTIFICATION_CREAR, nm.build());

    return START_STICKY;
}
```

Paso 6. Ejecuta la aplicación y verifica el nuevo resultado.



Ahora sí que se publica la notificación.

Paso 7. Cuando arrastras la barra de notificaciones hacia abajo, para mostrar la lista de notificaciones, y pulsas sobre una de ellas, es habitual que se abra una actividad para realizar acciones relacionadas con la notificación. En este ejercicio aprenderemos a asociar una actividad a una notificación.

Añade el siguiente código:

```
mChannel.setVibrationPattern(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});
mChannel.setShowBadge(false);
notificationManager.createNotificationChannel(mChannel);
}

Intent intent = new Intent(this, MainActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent, 0);

NotificationCompat.Builder nm = new NotificationCompat.Builder(this, CHANNEL_ID)
    .setContentIntent(pendingIntent)
    .setSmallIcon(R.drawable.ic_launcher_background)
    .setContentTitle("hola")
    .setContentText("adios");

notificationManager.notify(ID_NOTIFICATION_CREAR, nm.build());

return START_STICKY;
}
```

Este código asocia una actividad que se ejecutará cuando el usuario pulse sobre la notificación. Para ello se crea un **PendingIntent** (intención pendiente que se ejecutará más adelante) asociado a la actividad **ActividadPrincipal**. Por supuesto, también puedes crear una nueva actividad para usarla exclusivamente con este fin. En un ejemplo más complejo, puedes pasar los parámetros adecuados a través del **Intent**, para que la actividad conozca los detalles específicos que provocaron la notificación (por ejemplo, el número de teléfono que provocó la llamada perdida).

Paso 8. Queremos que si el servicio deja de estar activo, eliminamos la notificación. Para ello añade en **onDestroy()**:

```
@Override
public void onDestroy() {
    Toast.makeText( context: this, text: "Servicio detenido", Toast.LENGTH_SHORT).show();

    NotificationManager notificationManager =
        (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
    notificationManager.cancel(ID_NOTIFICATION_CREAR);

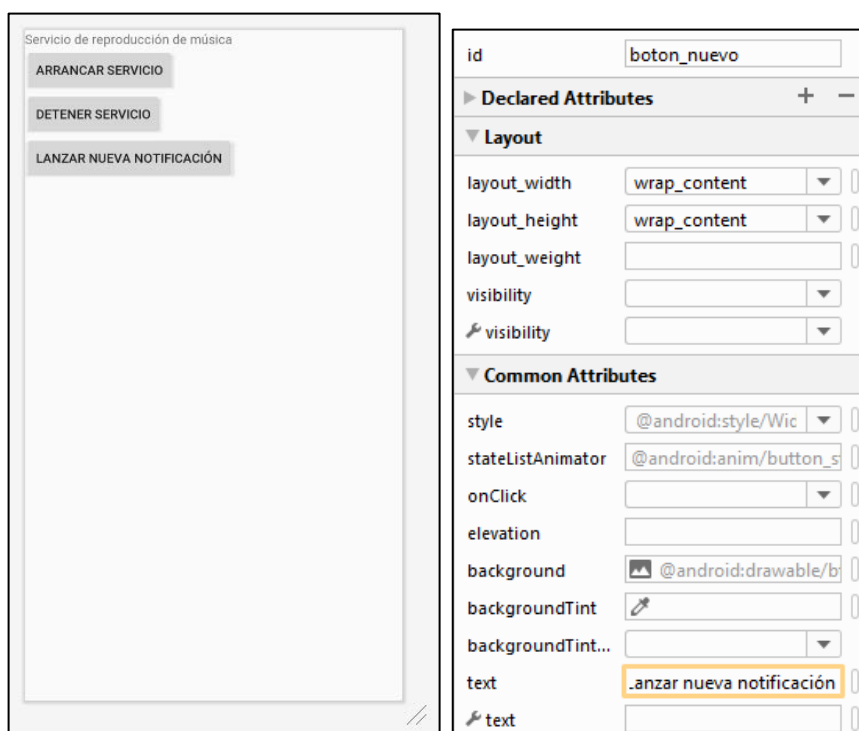
    reproductor.stop();
}
```

Este paso es opcional. Muchas notificaciones han de permanecer visibles aunque el servicio que las creo sea destruido. En nuestro caso, dado que estamos anunciando que un servicio de reproducción de música está activado, la notificación deja de tener sentido al desaparecer el servicio.

Paso 9. Ejecuta la aplicación y verifica que al parar el servicio la notificación desaparece.
Efectivamente la notificación desaparece.

Parte IV: Una notificación de socorro

Paso 1. En el proyecto anterior, ServicioMusica, crea un nuevo botón.



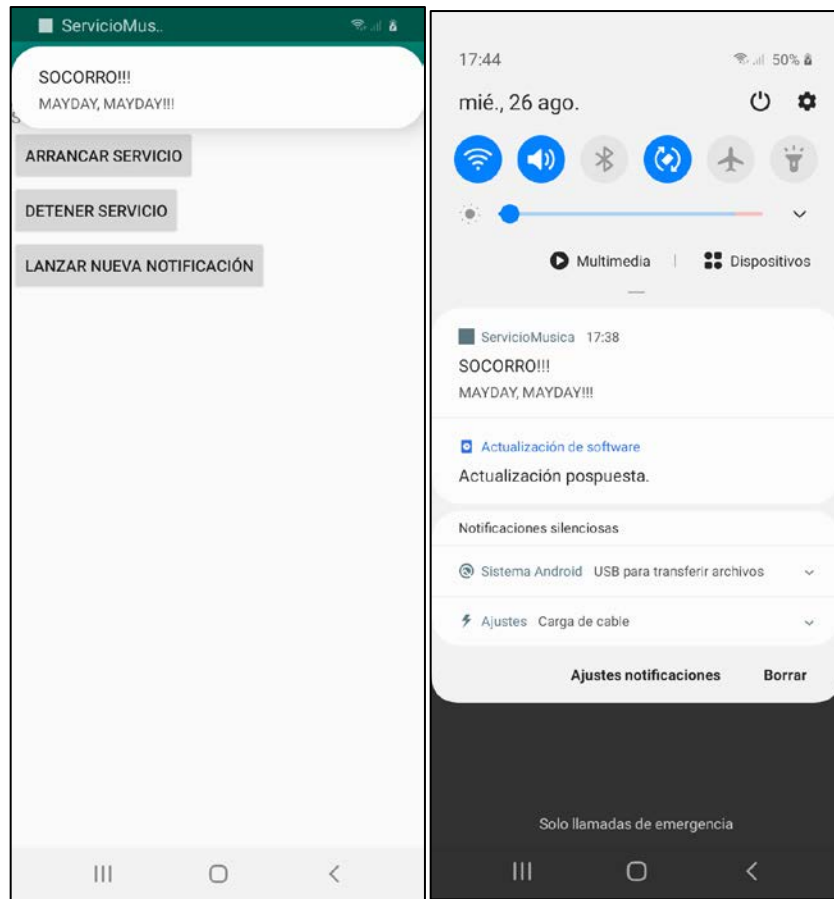
Paso 2. Al pulsar este botón se lanzará una nueva notificación que mostrará el texto “¡SOCORRO!”.

```
Button nuevo = (Button) findViewById(R.id.boton_nuevo);
nuevo.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
        enviaNotificacion();
    }
});
```

```
public void enviaNotificacion() {
    NotificationManager notificationManager =
        (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
    String CHANNEL_ID = "my_channel_01";

    if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
        CharSequence name = "my_channel";
        String Description = "This is my channel";
        int importance = NotificationManager.IMPORTANCE_HIGH;
        NotificationChannel mChannel = new NotificationChannel(CHANNEL_ID, name, importance);
        mChannel.setDescription(Description);
        mChannel.enableLights(true);
        mChannel.setLightColor(Color.RED);
        mChannel.enableVibration(true);
        mChannel.setVibrationPattern(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});
        mChannel.setShowBadge(false);
        notificationManager.createNotificationChannel(mChannel);
    }
    Intent intent = new Intent( packageContext: this, MainActivity.class);
    PendingIntent pendingIntent = PendingIntent.getActivity( context: this, requestCode: 0, intent, flags: 0);

    NotificationCompat.Builder nm = new NotificationCompat.Builder( context: this, CHANNEL_ID)
        .setContentIntent(pendingIntent)
        .setSmallIcon(R.drawable.ic_launcher_background)
        .setContentTitle("SOCORRO!!!")
        .setContentText("MAYDAY, MAYDAY!!!");
    int ID_NOTIFICATION_CREAR=1;
    notificationManager.notify(ID_NOTIFICATION_CREAR, nm.build());
}
```



Paso 3. El audio de la notificación será una grabación de voz que diga “¡SOCORRO!”.

Iremos a la aplicación Grabadora (que hicimos en temas anteriores) y ésta nos generará un archivo con nuestra voz en el directorio `/mnt/sdcard/Android/data/com.example.grabadora/files/audio.3gp`.

▼ mnt	drwxr-xr-x	2020-08-24 17:25	320 B
▶ appfuse	drwx--x--x	2020-08-24 17:25	40 B
▶ asec	drwxr-xr-x	2020-08-24 17:25	40 B
▶ expand	drwxrwx--x	2020-08-24 17:25	40 B
▶ Knox	drwx-----	2020-08-24 17:25	100 B
▶ media_rw	drwxr-x---	2020-08-24 17:25	100 B
▶ obb	drwxr-xr-x	2020-08-24 17:25	40 B
▶ runtime	drwx-----	2020-08-24 17:25	120 B
▼ sdcards	lrwxrwxrwx	2020-08-24 17:25	21 B
▶ Alarms	drwxrwx--x	2020-06-19 17:36	4 KB
▼ Android	drwxrwx--x	2020-06-19 17:36	4 KB
▼ data	drwxrwx--x	2020-08-27 16:24	4 KB
▶ android.auto_generated_rro_vendor__	drwxrwx--x	2020-08-27 16:22	4 KB
▶ com.android.chrome	drwxrwx--x	2020-07-23 17:26	4 KB
▶ com.android.vending	drwxrwx--x	2020-06-19 17:37	4 KB
▶ com.diotek.sec.lookup.dictionary	drwxrwx--x	2020-08-27 16:23	4 KB
▶ com.example.ejemplogooglemaps	drwxrwx--x	2020-07-27 19:27	4 KB
▼ com.example.grabadora	drwxrwx--x	2020-08-26 17:51	4 KB
▼ files	drwxrwx--x	2020-08-26 17:51	4 KB
▶ audio.3gp	-rw-rw----	2020-08-26 17:51	6,2 KB
▶ com.example.mislugares	drwxrwx--x	2020-08-05 12:45	4 KB
▶ com.example.notification	drwxrwx--x	2020-08-27 16:23	4 KB
▶ com.example.nuevopermiso	drwxrwx--x	2020-08-27 16:22	4 KB
▶ com.example.pruebamapa	drwxrwx--x	2020-08-03 12:52	4 KB

Descargamos el audio, lo renombramos y lo copiamos en la carpeta `res/raw` para utilizarlo como sonido de fondo en la notificación.

```

if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
    CharSequence name = "my_channel";
    String Description = "This is my channel";
    int importance = NotificationManager.IMPORTANCE_HIGH;
    NotificationChannel mChannel = new NotificationChannel(CHANNEL_ID, name, importance);
    mChannel.setDescription(Description);
    mChannel.enableLights(true);
    mChannel.setLightColor(Color.RED);
    mChannel.enableVibration(true);

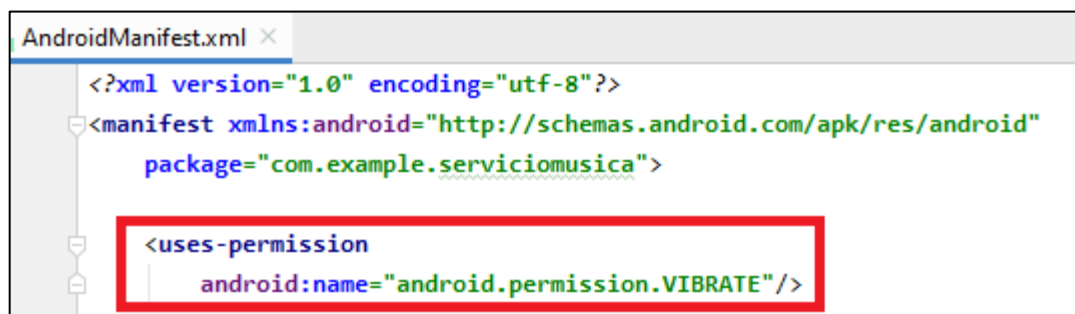
    Uri fichero=Uri.parse("Android.resource://" + getPackageName() + "/" + R.raw.audio3gp);
    AudioAttributes audioAttributes = new AudioAttributes.Builder()
        .setContentType(AudioAttributes.CONTENT_TYPE_SPEECH)
        .build();
    mChannel.setSound(fichero,audioAttributes);

    long [] vibrate = new long[]{50, 50, 50, 300, 300, 300, 50, 50, 50};
    mChannel.setVibrationPattern(vibrate);

    mChannel.setShowBadge(false);
    notificationManager.createNotificationChannel(mChannel);
}

```

Paso 4. La notificación hará vibrar el teléfono con el mensaje internacional de socorro S.O.S. codificado en Morse. Para conseguir esto haz vibrar el teléfono con una sucesión de tres pulsos cortos, tres largos y otros tres cortos (. . . - - - . . .).



```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.serviciomusica">

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

```

Parte V: Arranque automático del Servicio de Música.

Modifica el proyecto ServicioMusica para que el servicio se active desde el arranque del sistema operativo.

Paso 1. Creamos el receptor de anuncios ReceptorArranque, donde en su evento onReceive arrancaremos el servicio ServicioMusica con un intent:

```

public class ReceptorArranque extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        context.startService(new Intent(context, ServicioMusica.class));
    }
}

```

Paso 2. En AndroidManifest.xml registramos el receptor de anuncios y pedimos el permiso RECEIVE_BOOT_COMPLETED:


```
AndroidManifest.xml x
<?xml version="1.0" encoding="utf-8"?>
<manifest>
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="ServicioMusica"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <service android:name=".ServicioMusica" />
        <receiver android:name=".ReceptorArranque">
            <intent-filter>
                <action android:name="android.intent.action.BOOT_COMPLETED"/>
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

Paso 3. Ejecuta el proyecto en el móvil una primera vez. A continuación lo apagamos y al reiniciarlo de nuevo se inicia el servicio de música.

