LOS SERVICIOS

UNIVERSIDAD POLITECNICA DE VALENCIA

- Una aplicación en Android puede tener tres tipo de componentes principales:
  - **Actividades:** pantallas del interfaz de usuario
  - **Receptores de anuncios:** reacciona ante mensajes
  - **Servicios:** estudiados en esta presentación

- ¿Cuándo definir un servicio?
  - Queremos parte del código que se ejecute siempre en segundo plano.
  - No precise interacción con el usuario

Doble función de los servicios en Android

UNIVERSIDAD POLITECNICA DE VALENCIA

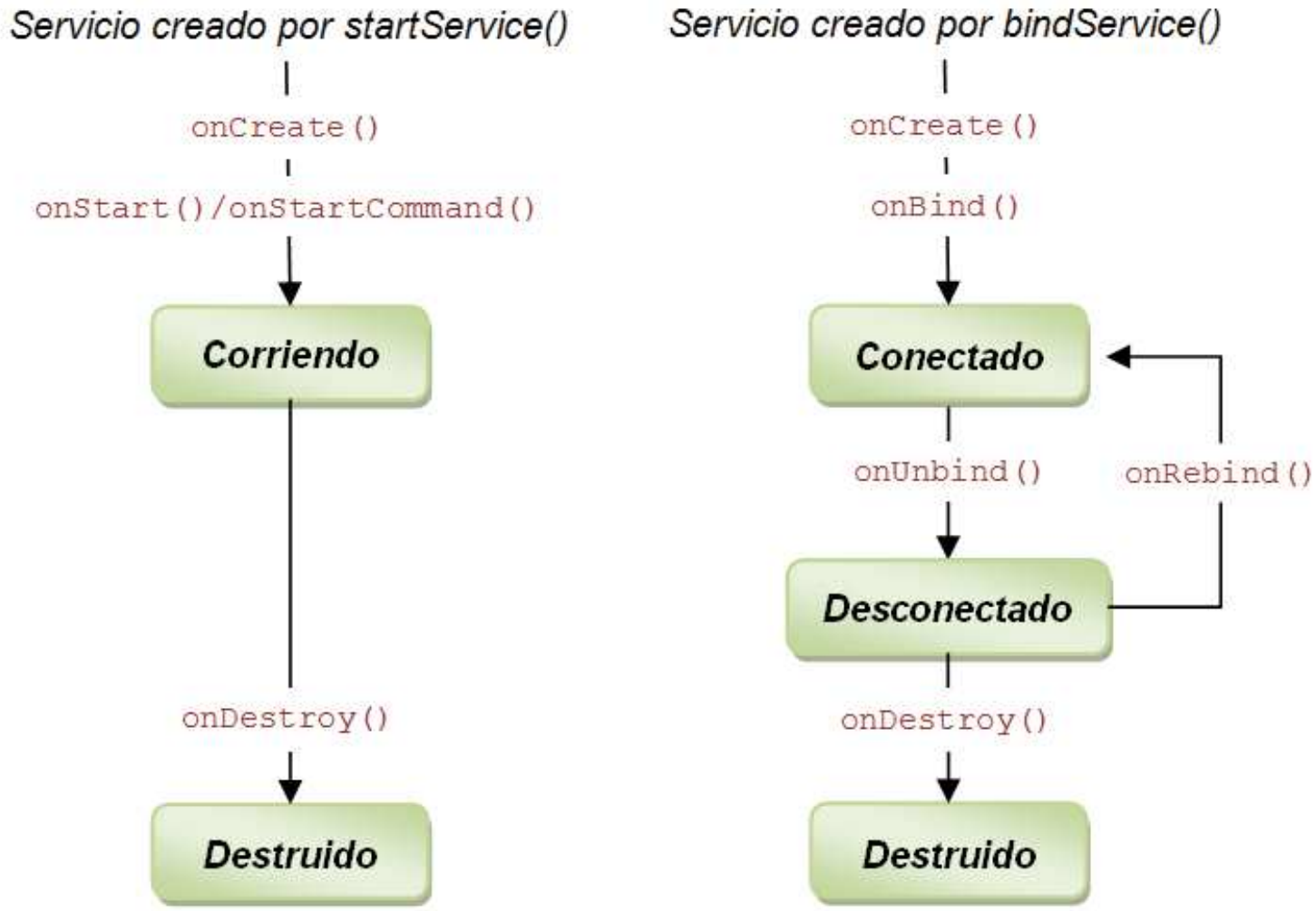
- Ejecutar un componente en segundo plano:
  - Normalmente durante un largo período de tiempo y activo mientras cambiamos entre actividades.
  - Se inicia con startService(), que indica al sistema que lo ejecute de forma indefinida hasta que sea detenido.
- Permite que otras aplicaciones se comuniquen con nuestra aplicación:
  - Ofreceremos ciertas funciones que podrán ser llamada desde otras aplicaciones.
  - Se conectarán con bindService(), que permite que se establezca una conexión con el servicio para poder invocar alguno de los métodos que se publican.

Hilos de ejecución

UNIVERSIDAD POLITECNICA DE VALENCIA

- Un servicio, como el resto de componentes de una aplicación, se ejecuta en el hilo principal.
- Por lo tanto, si el servicio necesita un uso intensivo de CPU o puede quedar bloqueado en ciertas operaciones, como uso de redes, debes crear su propio hilo.
- También puedes utilizar la clase `IntentService` para lanzar un servicio en su propio hilo.

CICLO DE VIDA DE UN SERVICIO



Nota: A diferencia de una actividad, *no* debes llamar al super

CREACIÓN DE UN SERVICIO

UNIVERSIDAD POLITECNICA DE VALENCIA

```
public class ServicioMusica extends Service {
  MediaPlayer reproductor;

  @Override public void onCreate() {
     reproductor = MediaPlayer.create(this, R.raw.audio);
  }

  @Override public int onStart (Intent intencion,
                                int idArranque) {
     reproductor.start();
  }

  @Override public void onDestroy() {
     reproductor.stop();
     reproductor.release();
  }
}
```

Creación de un Servicio

UNIVERSIDAD POLITECNICA DE VALENCIA

- En `AndroidManifest.xml` añade en la etiqueta `<application>`

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

- Para arrancar el servicio:

```
startService(new Intent(MiActividad.this,
                        ServicioMusica.class));
```

- Para parar el servicio:

```
stopService(new Intent(MiActividad.this,
                       ServicioMusica.class));
```

A PARTIR DE LA VERSIÓN 2.0

UNIVERSIDAD POLITECNICA DE VALENCIA

- En cualquier versión:

```
@Override public void onStart(Intent intencion,
                              int idArranque) {
   reproductor.start();
}
```

- A partir de la 2.0:

```
@Override public int onStartCommand(Intent intencion,
                    int flags, int idArranque) {
   reproductor.start();
   return START_STICKY;
}
```

onStartCommand

UNIVERSIDAD POLITECNICA DE VALENCIA

```
public int onStartCommand (Intent intencion,
                           int flags, int idArranque)
```

`intencion` Un objeto `Intent` que se indicó en la llamada `startService(Intent)`.

`flags` Información sobre como comienza la solicitud. Puede ser 0, `START_FLAG_REDELIVERY` o `START_FLAG_RETRY`. Un valor distinto de 0 se utiliza para reiniciar un servicio tras detectar algún problema (ver siguiente transparencia).

`idArranque` Un entero único representando la solicitud de arranque específica. Usar este mismo estero en el método `stopSelfResult(int idArranque)`.

onStartCommand

UNIVERSIDAD POLITECNICA DE VALENCIA

retorna cómo ha de comportarse el sistema cuando el proceso del servicio sea matado con el servicio inicializado en situaciones de baja memoria.

START_STICKY: El sistema tratará de recrear el servicio, se realizará una llamada a onStartCommand() pero con el parámetro intent igual a null. Usar cuando el servicio puede arrancar sin información adicional.

START_NOT_STICKY: El sistema no tratará de volver a crear el servicio, por lo tanto el parámetro intent nunca podrá ser igual a null. Usar cuando el servicio no puede reanudarse una vez interrumpido.

START_REDELIVER_INTENT: El sistema tratará de volver a crear el servicio. El parámetro intent será el que se utilizó en la última llamada startService(Intent).

START_STICKY_COMPATIBILITY: Versión compatible de START_STICKY, que no garantiza que onStartCommand() sea llamado después de que el proceso sea matado.

La clase IntentService

UNIVERSIDAD POLITECNICA DE VALENCIA

- Usa `IntentService` en lugar de `Service` cuando quieras un servicio en su propio hilo.

```
public class MiServicio extends IntentService {
  public MiServicio () {
        super("Nombre de mi servicio");
  }
  @Override
  protected void onHandleIntent(Intent intencion) {
        ...
  }
}
```

- Las peticiones son encoladas. Se atienden una tras otra sin que haya dos a la vez en ejecución.
- Llamar a un `IntentService` es igual que `Service`