

## 第5章 后台服务与系统服务

### 5.1 后台服务

Android 系统的后台服务（Service）是一种类似于 Activity 的组件，但 Service 没有用户操作界面，也不能自己启动，其主要作用是提供后台服务调用。Service 不像 Activity 那样当用户关闭应用界面就停止运行，Service 会一直在后台运行，除非明确命令其停止。

通常使用 Service 为应用程序提供一些只需在后台运行的服务或不需要界面的功能，如从 Internet 下载文件、控制 Video 播放器等。

Service 的生命周期中只有 3 个阶段，即 onCreate、onStartCommand、onDestroy。Service 的常用方法如表 5-1 所示。

表 5-1 Service 的常用方法

| 方 法                                                    | 说 明                                          |
|--------------------------------------------------------|----------------------------------------------|
| onCreate()                                             | 创建后台服务                                       |
| onStartCommand (Intent intent, int flags, int startId) | 启动后台服务                                       |
| onDestroy()                                            | 销毁后台服务，并删除所有调用                               |
| sendBroadcast(Intent intent)                           | 继承父类 Context 的 sendBroadcast()方法，实现发送广播机制的消息 |
| onBind(Intent intent)                                  | 与服务通信的信道进行绑定，服务程序必须实现该方法                     |
| onUnbind(Intent intent)                                | 撤销与服务信道的绑定                                   |

通常 Service 要在一个 Activity 中启动，调用 Activity 的 startService(Intent)方法启动 Service。若要停止正在运行的 Service，则调用 Activity 的 stopService(Intent)方法关闭 Service。startService()和 stopService()方法均继承于 Activity 及 Service 共同的父类 android.content.Context。

一个服务只能创建一次，销毁一次，但可以开始多次，即 onCreate()和 onDestroy()方法只会被调用一次，而 onStartCommand()方法可以被调用多次。后台服务的具体操作一般应该放在 onStartCommand()方法里面。如果 Service 已经启动，当再次启动 Service 时则不调用 onCreate()而直接调用 onStartCommand()。

设计一个后台服务的应用程序大致有以下几个步骤。

(1) 创建 Service 的子类。

- ① 编写 onCreate()方法, 创建后台服务;
- ② 编写 onStartCommand()方法, 启动后台服务;
- ③ 编写 onDestroy()方法, 终止后台服务, 并删除所有调用。

(2) 创建启动和控制 Service 的 Activity。

- ① 创建 Intent 对象, 建立 Activity 与 Service 的关联;
- ② 调用 Activity 的 startService(Intent)方法启动 Service 后台服务;
- ③ 调用 Activity 的 stopService(Intent)方法关闭 Service 后台服务。

(3) 修改配置文件 AndroidManifest.xml。

在配置文件 AndroidManifest.xml 的<application>标签中添加以下代码:

```
<service android:enabled="true" android:name=".AudioSrv" />
```

**【例 5-1】** 一个简单的后台音频服务程序示例。

本例通过一个按钮启动后台服务, 在服务程序中播放音频文件, 演示服务程序的创建、启动, 再通过另一个按钮演示服务程序的销毁过程。新建项目 ex5\_1 后, 将音频文件 mtest.mp3 复制到应用程序新建的 res\raw 目录下。

(1) 设计界面布局。

在界面布局中, 设置两个按钮, 分别用于“启动后台服务程序”和“关闭后台服务程序”; 再设置一个文本标签, 用于显示后台服务程序的运行状态。

(2) 设计后台服务程序。

新建一个后台服务程序, 命名为 AudioSrv.java, 其代码如下。

```
1 package com.example.ex5_1;
2 import android.app.Service;
3 import android.content.Intent;
4 import android.media.MediaPlayer;
5 import android.os.IBinder;
6 import android.widget.Toast;
7
8 public class AudioSrv extends Service
9 {
10     MediaPlayer play;
11     @Override
12     public IBinder onBind(Intent intent)
13     {
14         return null;
15     }
16     public void onCreate()
17     {
18         super.onCreate();
19         play = MediaPlayer.create(this, R.raw.mtest); ◀创建调用资源音频文件对象
20         Toast.makeText(this, "创建后台服务...", Toast.LENGTH_LONG).show();
```

```

21     }
22     public int onStartCommand(Intent intent, int flags, int startId)
23     {
24         super.onStartCommand(intent, flags, startId);
25         play.start(); ← 开始播放音乐
26         Toast.makeText(this, "启动后台服务程序, 播放音乐...",
27             Toast.LENGTH_LONG).show();
28         return START_STICKY;
29     }
30     public void onDestroy()
31     {
32         play.release();
33         super.onDestroy();
34         Toast.makeText(this, "销毁后台服务!", Toast.LENGTH_LONG).show();
35     }
36 }

```

(3) 启动后台服务的主控程序 MainActivity.java。

```

1  package com.example.ex5_1;
2  import androidx.appcompat.app.AppCompatActivity;
3  import android.os.Bundle;
4  import android.content.Intent;
5  import android.view.View;
6  import android.view.View.OnClickListener;
7  import android.widget.Button;
8  import android.widget.TextView;
9  public class MainActivity extends Activity
10 {
11     Button startbtn, stopbtn;
12     Intent intent;
13     static TextView txt;
14     @Override
15     public void onCreate(Bundle savedInstanceState)
16     {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.main);
19         startbtn=(Button)findViewById(R.id.butn1);
20         stopbtn=(Button)findViewById(R.id.butn2);
21         startbtn.setOnClickListener(new mClick());
22         stopbtn.setOnClickListener(new mClick());
23         txt=(TextView)findViewById(R.id.text1);
24         intent = new Intent(MainActivity.this, AudioSrv.class); ← 创建 Intent 对象
25     }
26     class mClick implements OnClickListener //定义一个类实现监听接口
27     {
28         public void onClick(View v)

```

```

29     {
30         if(v == startbtn)
31         {
32             MainActivity.this.startService(intent); ← 启动 Intent 关联的 Service
33             txt.setText("start service .....");
34         }
35         else if(v == stopbtn)
36         {
37             MainActivity.this.stopService(intent); ← 终止后台服务
38         }
39     }
40 }
41 }

```

(4) 修改配置文件 AndroidManifest.xml。

在配置文件 AndroidManifest.xml 的<application>标签中添加以下代码：

```
<service android:enabled="true" android:name=".AudioSrv" />
```

修改后完整的 AndroidManifest.xml 文件如下：

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.example.ex5_1" />
4      <application
5          android:allowBackup="true"
6          android:icon="@mipmap/ic_launcher"
7          android:label="ex5_1 "
8          android:roundIcon="@mipmap/ic_launcher_round"
9          android:supportRtl="true"
10         android:theme="@style/Theme.ex5_1">
11         <activity
12             android:name=".MainActivity" >
13             <intent-filter>
14                 <action android:name="android.intent.action.MAIN" />
15                 <category android:name="android.intent.category.LAUNCHER" />
16             </intent-filter>
17         </activity>
18         <!-- 添加 AudioSrv 服务程序 -->
19         <service android:enabled="true" android:name=".AudioSrv" />
20     </application>
21 </manifest>

```

程序运行结果如图 5.1 所示。

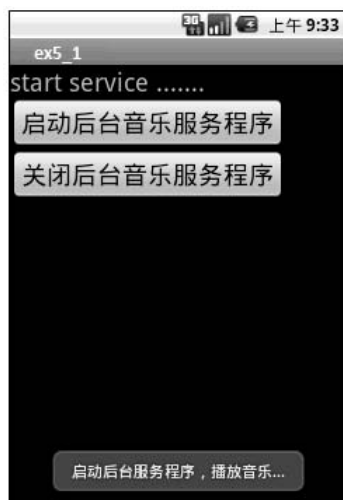


图 5.1 启动后台服务程序

## 5.2 信息广播机制

Broadcast 是 Android 系统应用程序之间传递信息的一种机制。当系统之间需要传递某些信息时，不是通过诸如单击按钮之类的组件来触发事件，而是由系统自身通过系统调用来引发事件。这种系统调用是由 `BroadcastReceiver` 类实现的，把这种系统调用称为广播。`BroadcastReceiver` 是“广播接收者”的意思，顾名思义，它用来接收来自系统和应用中的广播信息。

在 Android 系统中有很多广播信息。例如，当开机时，系统会产生一条广播信息，接收到这条广播信息就能实现开机启动服务的功能；当网络状态改变时，系统会产生一条广播信息，接收到这条广播信息就能及时地做出提示和保存数据等操作；当电池电量改变时，系统会产生一条广播信息，接收到这条广播信息就能在电量低时告知用户及时保存进度等。

下面通过简单的应用示例来说明信息广播的设计方法。

**【例 5-2】** 一个简单的信息广播程序示例。

(1) 设计主控文件 `MainActivity.java`。

```
1 package com.example.ex5_2;  
2 import androidx.appcompat.app.AppCompatActivity;  
3 import android.os.Bundle;  
4 import android.content.Intent;  
5 import android.view.View;  
6 import android.widget.Button;  
7 import android.widget.TextView;
```

```
8 public class MainActivity extends Activity
```

```

9  {
10     static TextView txt; ← 要在类外调用，设置成静态变量
11     private TestReceiver recevier;
12     private IntentFilter intentFilter;
13     @Override
14     public void onCreate(Bundle savedInstanceState)
15     {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.main);
18         txt = (TextView)findViewById(R.id.txtView);
19         Button btn=(Button)findViewById(R.id.button);
20         btn.setOnClickListener(new mClick());
21     }

22     class mClick implements View.OnClickListener
23     {
24         @Override
25         public void onClick(View v)
26         {
27             recevier = new TestReceiver();
28             intentFilter = new IntentFilter(); ← 创建过滤器
29             intentFilter.addAction("android.net.conn.CONNECTIVITY_CHANGE");
30             registerReceiver(recevier, intentFilter); ← 注册广播
31             System.out.println("广播已经发送");
32         }
33     }
34 }

```

(2) 设计广播接收器文件 TestReceiver.java。

```

1  package com.ex5_2;
2  import android.content.BroadcastReceiver;
3  import android.content.Context;
4  import android.content.Intent;
5
6  public class TestReceiver extends BroadcastReceiver ← 定义广播接收器
7  {
8      @Override
9      public void onReceive(Context context, Intent intent)
10     {
11         String str = intent.getExtras().getString("hello"); ← 接收的广播数据
12         MainActivity.txt.setText(str); ← 显示接收的数据
13     }
14 }

```

(3) 设计配置文件 AndroidManifest.xml。

```

1  <?xml version="1.0" encoding="utf-8"?>

```

```

2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.example.ex5_2">

4      <application
5          android:allowBackup="true"
6          android:icon="@mipmap/ic_launcher"
7          android:label="@string/app_name"
8          android:roundIcon="@mipmap/ic_launcher_round"
9          android:supportsRtl="true"
10         android:theme="@style/Theme.ex5_2">
11         <activity android:name=".MainActivity">
12             <intent-filter>
13                 <action android:name="android.intent.action.MAIN" />

14                 <category android:name="android.intent.category.LAUNCHER" />
15             </intent-filter>
16         </activity>
17         <!-- 注册对应的广播接收类 -->
18         <receiver android:name=".TestReceiver">
19             </receiver>
20     </application>
21 </manifest>

```

新增语句

程序运行结果如图 5.2 所示。

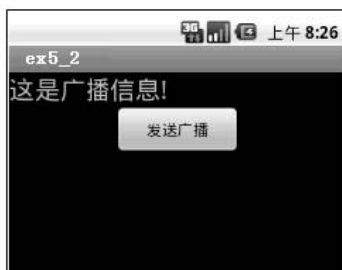


图 5.2 简单的广播示例

**【例 5-3】** 由一个后台服务广播音乐的播放、暂停或停止信息，接收器接收到信息后，执行改变用户界面按钮上文本的操作。

在本例中创建了 3 个类，即 MainActivity、AudioService 和 Broadcast，MainActivity 负责用户的交互界面，并启动后台服务；AudioService 是 Service 的子类，在后台提供播放、暂停或停止音乐等功能，同时发送改变交互界面的广播信息；Broadcast 是 BroadcastReceiver 的子类，负责接收广播信息，更改交互界面。

(1) 设计界面布局。

在界面布局中，设置一个文本标签和 3 个按钮，如图 5.3 所示。



图 5.3 后台服务广播音乐的界面布局设计

(2) 设计主控文件 MainActivity.java。

```

1  package com.example.ex5_3;
2  import androidx.appcompat.app.AppCompatActivity;
3  import android.os.Bundle;
4  import android.content.Intent;
5  import android.view.View;
6  import android.widget.Button;
7  import android.widget.TextView;

8  public class MainActivity extends AppCompatActivity
9  {
10     static Button startbtn, stopbtn, paubtn;
11     static TextView txt;
12     Intent intent;

13     @Override
14     public void onCreate(Bundle savedInstanceState)
15     {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.main);
18         startbtn=(Button)findViewById(R.id.button);
19         paubtn = (Button)findViewById(R.id.button2);
20         stopbtn=(Button)findViewById(R.id.button3);
21         txt=(TextView)findViewById(R.id.textView);
22         intent = new Intent(MainActivity.this, AudioSrv.class);
23         startbtn.setOnClickListener(new mClick());
24         paubtn.setOnClickListener(new mClick());
25         stopbtn.setOnClickListener(new mClick());
26     }

27     class mClick implements View.OnClickListener //定义实现监听接口类
28     {
29         public void onClick(View v)
30         {
31             if(v == startbtn)
32             {
33                 intent.putExtra("play",1);
34                 MainActivity.this.startService(intent);//开启服务
35                 MainActivity.this.startService(intent);
36                 System.out.println("(1)启动音乐");
37             }
38             else if(v == paubtn)
39             {
40                 intent.putExtra("play",2);
41                 MainActivity.this.startService(intent);
42                 System.out.println("(2)暂停音乐");

```

```

43         }
44         else if(v == stopbtn)
45         {
46             intent.putExtra("play",3);
47             MainActivity.this.startService(intent);
48             System.out.println("(3)停止音乐");
49         }
50     }
51 }
52 }

```

(3) 设计后台服务程序 AudioService.java。

```

1  package com.example.ex5_3;
2  import android.app.Service;
3  import android.content.Intent;
4  import android.content.IntentFilter;
5  import android.media.MediaPlayer;
6  import android.os.IBinder;
7  import android.widget.Toast;

8  public class AudioService extends Service
9  {
10     public MediaPlayer play;
11     private AudioReceiver receiver; //接收广播类
12     private IntentFilter intentFilter;
13     public IBinder onBind(Intent intent)
14     {
15         return null;
16     }
17     public void onCreate() //创建后台服务
18     {
19         super.onCreate();
20         play = MediaPlayer.create(this, R.raw.mtest);
21         Toast.makeText(this, "创建后台服务...", Toast.LENGTH_LONG).show();
22     }
23     void onStart(){ //创建广播
24         receiver = new AudioReceiver();
25         intentFilter = new IntentFilter();
26         intentFilter.addAction("android.net.conn.CONNECTIVITY_CHANGE");
27         registerReceiver(receiver, intentFilter);
28     }
29     public int onStartCommand(Intent intent, int flags, int startId)
30     {
31         super.onStartCommand(intent, flags, startId);
32         switch (intent.getIntExtra("play",-1)){
33             case 1:

```

```

34         play.start(); //开始
35         on_Start();    //启动广播
36         Toast.makeText(this, "启动后台服务, 播放音乐...",
37                         Toast.LENGTH_SHORT).show();//提示
38         break;
39     case 2:
40         if (play!=null&&play.isPlaying()){
41             play.pause();//暂停
42             Toast.makeText(this, "暂停...", Toast.LENGTH_SHORT).show();
43         }
44         else {
45             play.start();//继续
46             Toast.makeText(this, "继续...", Toast.LENGTH_SHORT).show();
47         }
48         break;
49     case 3:
50         play.stop();//停止
51         play.release();//释放内存
52         Toast.makeText(this, "销毁后台服务...", Toast.LENGTH_SHORT).show();
53         break;
54     }
55     return START_STICKY;
56 }
57 public void onDestroy()
58 {
59     play.release();
60     super.onDestroy();
61     Toast.makeText(this, "销毁后台服务!", Toast.LENGTH_LONG).show();
62 }
63 }

```

(4) 设计广播接收器程序 Broadcast.java。

```

1  package com.example.ex5_3;
2  import android.content.BroadcastReceiver;
3  import android.content.Context;
4  import android.content.Intent;
5  /* 广播接收器 */
6  class Broadcast extends BroadcastReceiver
7  {
8      @Override
9      public void onReceive(Context context, Intent intent)
10     {
11         String action = intent.getAction();
12         if(action != null) {
13             String str = "正在播放音乐";
14             System.out.println(str);

```