Activities and Intents

Table of Contents

[Activities 3](#_Toc100898167)

[Activity Lifecycle Methods 4](#_Toc100898168)

[Intents 6](#_Toc100898169)

[Intent Filters 7](#_Toc100898170)

An Android application has four types of components:

1. **Activities** – These provide the user interface. We can think of a single activity as a single screen in the application. However, it can contain one or more views. Every activity extends the Activity class.
2. **Services** – These do not have any user interface. They work in the background. For example, if we play a song or download a file, there is a service running in the background. Every service extends the Service class.
3. **Broadcast Receivers** – These receive events from outside the application. For example, if we set an alarm, we do not constantly keep checking the time. Instead, once the specified time has been reached, the system sends out a broadcast that any application can receive. Our application will use a broadcast receiver to trigger the alarm once that broadcast is received. Broadcast receivers do not have any user interface, but they can trigger activities. They extend the BroadcastReceiver class.
4. **Content Providers** – These make data about the application available to other applications via some database, such as SQLite. For example, the contacts application does this to allow other applications to access the user’s contacts. They extent the ContentProvider class.

## Activities

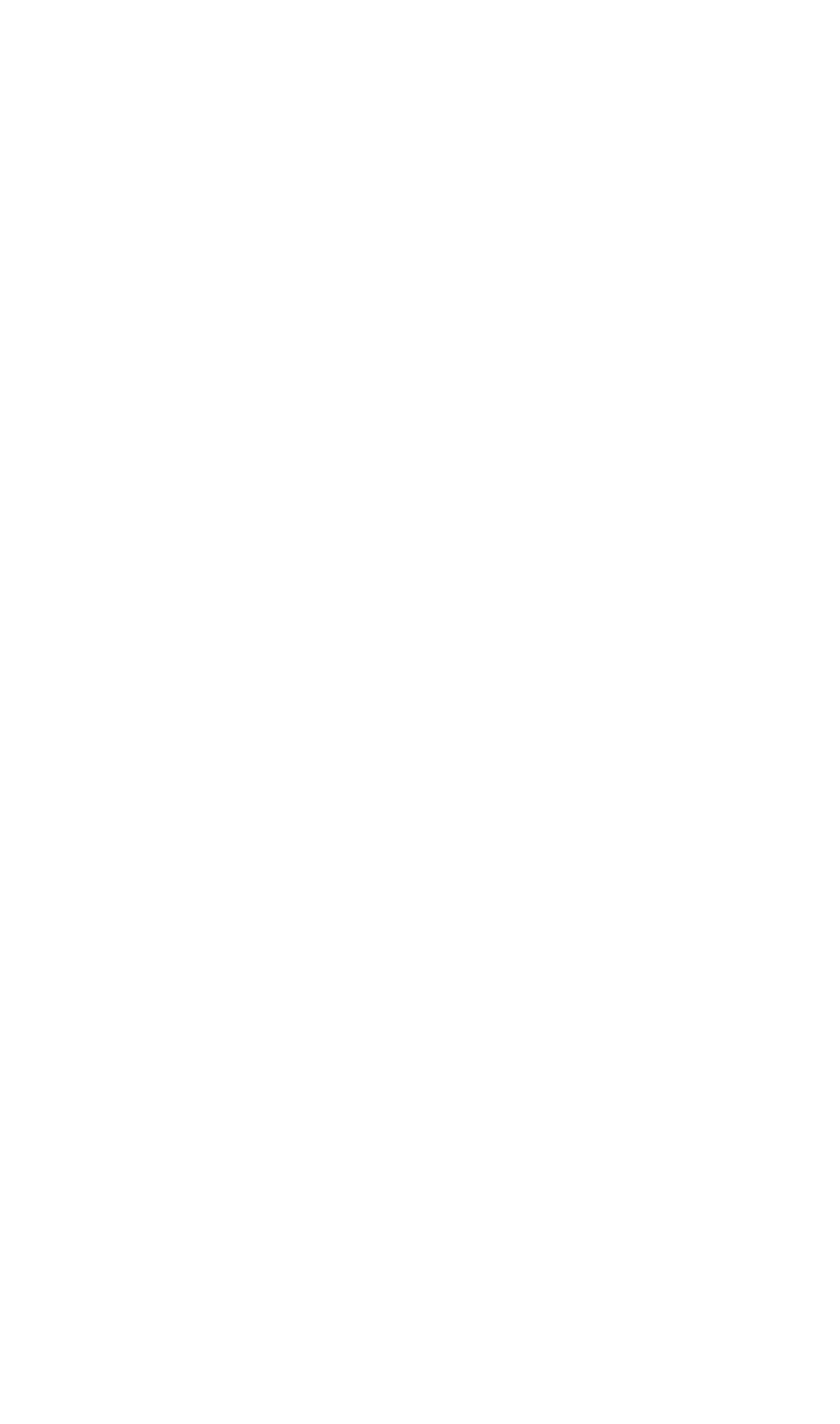
As mentioned before, an **activity** is a **single screen** with a **user interface**. Activities are the **main entry point** for the user to our application. When they first open the application, they will be greeted with an activity. Most applications have **multiple screens**, which means they have **multiple activities**. One of the activities is specified as the **main activity**, which is the one that first appears when the application opens. Each activity can start other activities in order to perform different actions.

For example, an email application could have one activity that shows a list of emails. Clicking on an email will start another activity to show the email.

Although activities work together, they are **independent**. This allows us to setup the activities so that other applications can start specific ones. For example, another application may trigger the compose email activity of the email application without starting the main activity.

### Activity Lifecycle Methods

Each activity goes through a **lifecycle**, during which different methods are triggered by the Android system at different stages, as shown below:



* onCreate() – Triggered when the system first starts creating the activity.
* onStart() – Triggered when the activity is completely created and is made visible to the user.
* onResume() – Triggered when the activity is brought to the foreground to allow the user to interact with it. Triggered immediately after onStart().
* onPause() – Triggered when the activity is sent to the background and cannot be interacted with, but is still visible to the user. This may happen due to a different application opening, a different activity in the same application opening or a pop-up dialog opening.
* onStop() – Triggered when the activity is sent to the background and is not visible to the user anymore, for example due to switching to a different application.
* onRestart() – Triggered when the activity comes back to the foreground after onStop() was called, for example if the user comes back to the application after going to a different one.
* onDestroy() – Triggered when the activity is completely removed and does not exist, even in the background.

## Intents

An **intent** is a message that we can use to request an action from a different application component. There are three major use cases of intents:

1. Starting an activity
2. Starting a service
3. Delivering a broadcast

Intents can be of two types, explicit and implicit.

An **explicit intent** is used to start a component within the same application. For example, when the user clicks the download button, we can use an explicit intent to start the download service in the background. Explicit intents require us to specify a package or class name.

An **implicit intent** does not specify any component, but declares a general action to perform. This action can then be performed by a different application. For example, we can send a location to Google Maps and tell it to show the location, which will cause the application to open up at the required activity. Implicit intents require the receiving application to include a compatible **intent filter**.

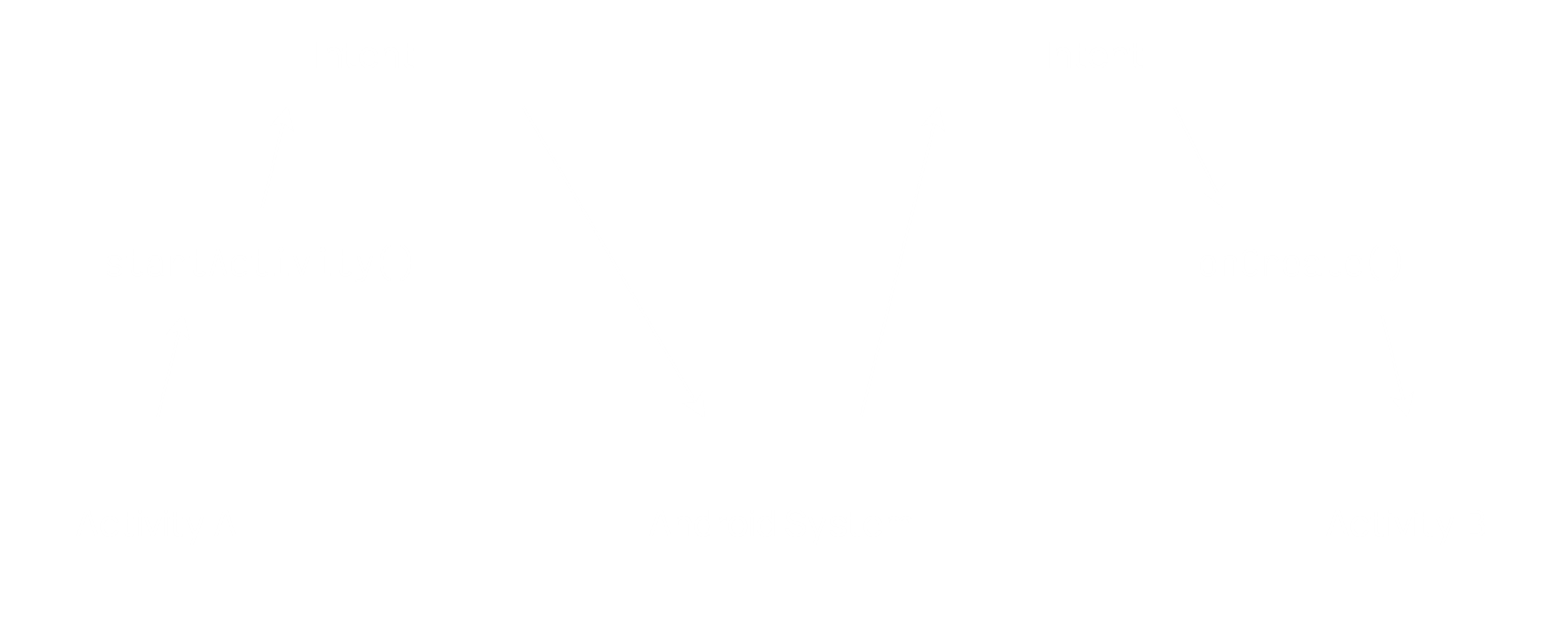
### Intent Filters

**Intent Filters** allow our application to handle incoming requests from other applications. They contain three parts, the **action**, the **category** and the **data**.

<activity android:name="ShareActivity" android:exported="false">  
 <intent-filter>  
 <action android:name="android.intent.action.SEND" />  
 <category android:name="android.intent.category.DEFAULT" />  
 <data android:mimeType="text/plain" />  
 </intent-filter>  
</activity>

XML

This is how implicit intents work. One application creates an intent and an action description and passes it to the startActivity() method. This method tells the Android system to search all other applications for an intent filter that matches the created intent. Once the required intent filter is found, the corresponding activity is started.



Thus, **implicit intents** make use of intent filters.