Contents

1 Broadcast receiver

Respond to system-wide broadcast announcements

- The user has not necessarily done something, but the OS / phone / another application has
- The screen has turned off, the battery is low, a new SMS has arrived, the phone has booted
- Can be sent by an application, or received by an application from the OS
- Create the process, instantiate a BroadcastReceiver (Resources). Process ddoesn't have to be running continuosly, can start when needed.

Intents are sent to specific Activities / Services. Explicitly or implicitly via Intent filters, URI provision. **Broadcasts** are sent to anything that cares to listen. System-wide Intent broadcasts

Is declared inside *Android.xml* file

- Need to specify broadcast intents we are interested in
- Registered and boot/install time.

2 Broadcast

Broadcasts are Intents

- Send an Intent to start an Activity
- Send an Intent to start a Service
- Send an Intent to trigger Broadcast Receivers that subscribe to that particular class of Intent
  - Can define our own Broadcast Intents
  - Cannot send system Intents (battery, screen etc), as the security model prevents it. Else could falsify system properties.
  - Again a messaging wrapper around Binder

2.1 Implementing a BroadcastReceiver

- **Subclass BroadcastReciever**: Specify which Intents we are interested in receiving. Permissions permitting
- Implement the onReceive() method

Then

- **Send a Message** to our application to change our behavior. Reduce the audio volume, play a sound
- **Start an Activity**. Never start an Activity in response to an incoming broadcast Intent. (Would stop current activity and disrupt user experience)
- **Start a Service**: A gateway to another component. How much work should a BroadcastReceiver actually do?
- **Show a Notification**: Alert the user that there is something that they need to interact with at some point

2.2 Sticky broadcast

Non-Sticky: If you werent listening at the time youve missed it. **Sticky**: Intents are cached by Android

- **Deprecated**
- New Intents overwrite older Intents they match
- Sticky broadcasts are global to the system
- And because of this, performing a sticky broadcast is multiple orders of **magnitude slower** than just implementing direct calls within your own app
- IPC for each receiver to register, IPC to the system to send it, IPC from the system back to your app to deliver it, marshalling and unmarshalling of all the data within the Intent over both IPCs
- More than that, there is **NO protection** on them, so any other application can watch your sticky broadcasts, or even send their own values back to you

No longer able to send implicit Broadcasts

2.3 Local Broadcasts

- Like the normal BroadcastReceiver but only supports local broadcasts (Within the application)
- Data broadcast will not leave the application (No data leakage)
- Other applications cannot send broadcasts to our application
- More efficient than sending a global broadcast (No IPC)
- Must register / sendBroadcasts using the LocalBroadcastManager. **Programmatically** rather than via the manifest

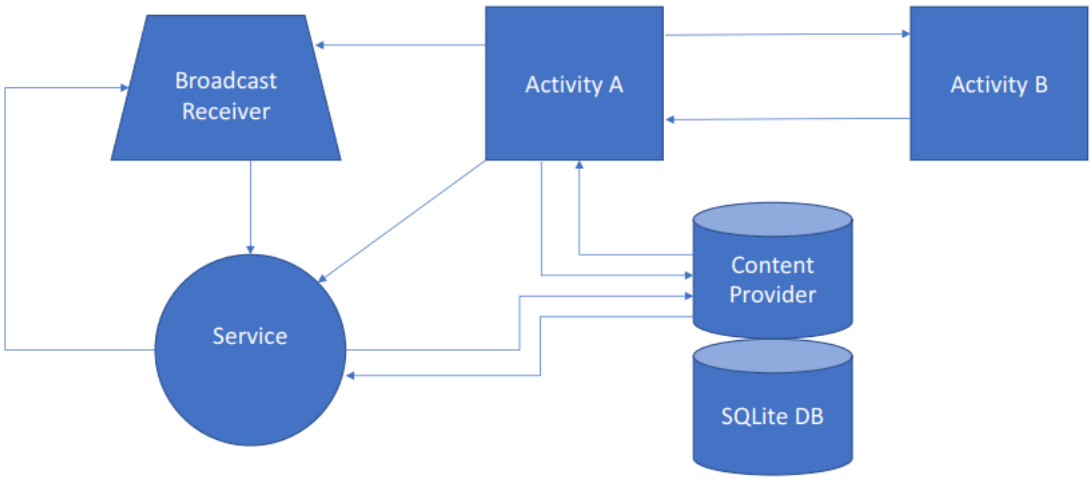
3 Caveats

Either the sender or receiver of a broadcast can implement permissions

- Control who can send broadcast intents to my application
- Control who can receive the intents that my application broadcasts
- Limit broadcasts to my application components only

Lifecycle

- A receiver handling broadcasts is considered to be running in the foreground (albeit briefly)
- Process is aggressively killed once onReceive() has returned. **No binding**, **no startActivityForResult** because they are asynchronous
- Long-running code should start a Service instead, as with any other Activity
- The application receiving the broadcasts must have been explicitly started / not explicitly stopped. Applications **cannot intercept broadcasts** without the user having some awareness that they have given it permission
- Can register / unregister for broadcast events programmatically. Most things that the manifest specifies can be done programmatically



Reference section

placeholder