Jacob Taggart
Week 3 Day 4 Research

Week 3 Day 4 Research

1. What is interprocess communication?

   Interprocess communication (IPC) is the communication between threads across the boundaries of processes. It allows communication across processes between the client and the service

2. Define each of the following and a short description of how each is implemented:
   a) AIDL

   AIDL stands for Android Interface Definition Language. It is also a way to carry out interprocess communication. It allows for sending primitive data across from client to server without any extra work-around. To implement it, one needs to create an AIDL file (an interface) in the server application. Two methods will be set up in this AIDL file - addNumbers and onBind. Then we need to make a java file that extends the Service class and overrides the addNumbers and onBind methods. Then the service needs to be defined in the manifest with a unique name and setting it to .remote. Then in a client app, we need to create a package with the same package name that holds the AIDL file in the server app. Then we need to put the exact code from the server's AIDL file into the package. Then we need to make an object of the server's AIDL interface. In the Main Activity, make a method called initConnection. Then all you have to do is call the appropriate functions. To pass object(s) make sure to use Parcelable.

   b) Messenger (for IPC)

   Messenger is a way to use IPC without having to use AIDL directly. Messenger class needs to be instantiated which allows communication across processes between the client and the service. A Messenger uses a built-in handler so that the thread handling is sequential by design. A service will pass a Messenger reference to the client, then using the passed reference the client can send as many messages as needed back to the server process. The entire communication process takes place in the Binder framework.

   c) Binder

Binder is a base class with the purpose of setting up a remote object. Most developers will not use this class, instead opting for use of the AIDL or Messenger. It's an IPC primitive and has no impact on the service lifecycle, so will keep running until the process that created is destroyed.

3. What are the restrictions for background services and what are some ways to accommodate for the restrictions?

Because backgrounds services can drain battery and other limited resources on the user's device without the user knowing, so a couple of important restrictions include an application idling for too long the service may either have to notify the user somehow or will have to end. Also broadcast receivers now must be explicitly stated in the code and cannot be implicitly defined (with some exceptions). This is so that receivers aren't constantly listening for signals unless there is a reason for such. Bound services are not affected by these restrictions. A good way to get around this by using job schedulers which allows an application to work while not being ran actively. This is a good way to perform operations without needing the heavy duty work of a long running service or broadcast receiver.

4. What are the restrictions for broadcast receivers?

Three of the restrictions for broadcast receivers include using permissions, Local Broadcast Manager (restricts broadcast to its own application), and System Receivers (restricts the receivers and broadcast to system-defined processes such as when someone uses airplane mode on a phone.)

5. What is client/server relationships?

In terms of services and IPC, the client/server relationships are the communication and relative affect the client application and service application have with/on each other. This mainly deals with sending information and/or threads to and from the processes as far as services are concerned.