CS5530
Mobile/Wireless Systems
Android Services

Yanyan Zhuang

Department of Computer Science

http://www.cs.uccs.edu/~yzhuang

cat announce.txt_

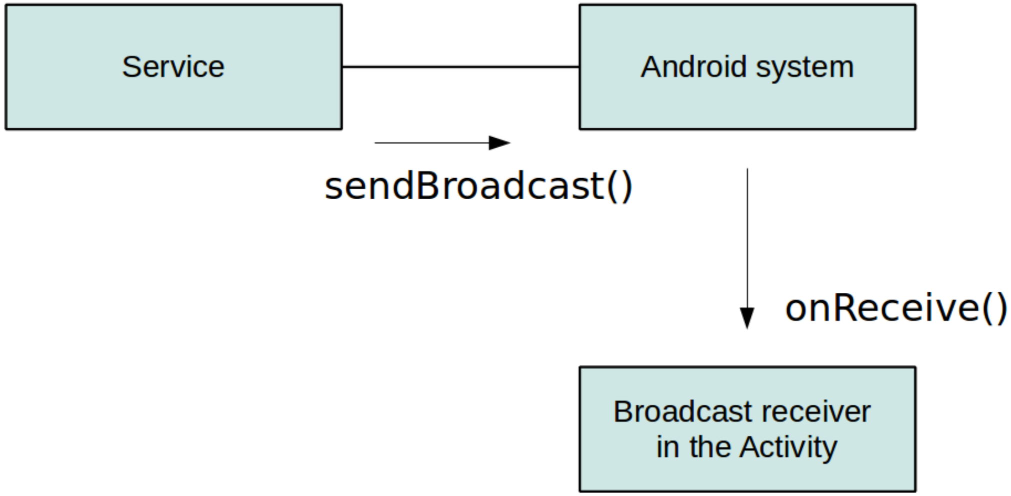
- FCQ next week
- If need extension, let me know asap
- Move last class to May 1st?

Android Services

- A Service is an app component that can perform long-running operations in background
  - It does not provide a UI
- Another app component (e.g., Activity) can start
  - Depending on the type of service, it can continue to run in background even if user switches to another app
- A component can bind & interact with it, and perform inter-process communication (IPC)

Android Services

- Ways to communicate with a service
  - Using intent data: no direct communication is required. The service receives the intent data and performs its work
  - Using receiver: another component register a broadcast receiver for an event that the service sends out (expensive)



Android Services

- Ways to communicate with a service
  - Using intent data: no direct communication is required. The service receives the intent data and performs its work
  - Using receiver: another component register a broadcast receiver for an event that the service sends out (expensive)
  - **Activity binding to a local service**: if the service is started in the same process as the activity, the activity can directly bind to the service (fast)

Bound Services

- A bound service is the server in a client-server interface
    - It allows components (e.g., activities) to bind to the service, send requests, receive responses, and perform IPC
    - A bound service typically lives only while it serves another application component and does not run in the background indefinitely

Bound Services

- A bound service is the server in a client-server interface
  - It allows components (e.g., activities) to bind to the service, send requests, receive responses, and perform IPC
  - A bound service typically lives only while it serves another application component and does not run in the background indefinitely
- If a service is used only by the local app and does not need to work across processes
  - Can implement a **Binder** class that provides client (e.g., activities) direct access to public methods in the service

Bound Services

- Steps
  - In service, create an instance of **Binder** class that
    - Creates the current Service instance, which has public methods the client can call
    - Return this instance of Binder from the onBind() callback method
  - In activity
    - Receive the Binder from the onServiceConnected() callback method
    - Make calls to the bound service using the methods provided
- Note
  - The service and client must be in the same app so the client can cast the returned object and properly call its APIs
  - The service and client must also be in the same process, because this technique does not perform any marshaling across processes

In Service

- Create an instance of Binder class that
  - Creates the current Service instance, which has public methods the client can call

```
public class LocalService extends Service {
    // Random number generator
    private final Random mGenerator = new Random();

    public class LocalBinder extends Binder {
        LocalService getService() {
            // Return this instance of LocalService so clients can call public methods
            return LocalService.this;
        }
    }
    /** method for clients */
    public int getRandomNumber() {
      return mGenerator.nextInt(100);
    }
}
```

In Service

- Create an instance of Binder class that
  - Return this instance of Binder from the onBind() callback method

```
public class LocalService extends Service {
    // Binder given to clients
    private final IBinder mBinder = new LocalBinder();

    @Override
    public IBinder onBind(Intent intent) {
        return mBinder;
    }
}
```

In Activity

- Receive the Binder from the onServiceConnected() callback method

  -
    ```
    public class MainActivity extends AppCompatActivity {
       @Override
       protected void onStart() {
          super.onStart();
          Intent intent = new Intent(this, LocalService.class);
          bindService(intent, mConnection, Context.BIND_AUTO_CREATE);
       }

       private ServiceConnection mConnection = new ServiceConnection() {
          @Override
          public void onServiceConnected(ComponentName className, IBinder service) {
             LocalService.LocalBinder binder = (LocalService.LocalBinder) service;
             mService = binder.getService();
             mBound = true;
          }
          @Override
          public void onServiceDisconnected(ComponentName arg0) {
             mBound = false;
          }
       };
    }
    ```

In Activity

- Make calls to the bound service using the methods provided

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onStop() {
        super.onStop();
        if (mBound) {
            unbindService(mConnection);
            mBound = false;
        }
    }

    public void onButtonClick(View v) {
        if (mBound) {
            int num = mService.getRandomNumber();
            Toast.makeText(this, "number: " + num, Toast.LENGTH_SHORT).show();
        }
    }
}
```