ANDROID DEVELOPMENT

SERVICES

Tan Cher Wah (isstcw@nus.edu.sg)

© 2019 National University of Singapore. All Rights Reserved.

Services vs Activity

- Service is an application component to perform **background tasks** e.g.
  - Downloads file
  - Play music
  - Log no. of steps taken a day
- A Service has no user interface (UI)
- There are 3 types of Services
  - Started Service
  - Bound Service
  - IntentService

© 2019 National University of Singapore. All Rights Reserved.

What is Started Service?

- A Started Service is a service that runs within the **main UI Thread**
- Starts in response to a **startService()**
- Ends in response to a **stopService()** or itself calling **stopSelf()**
- For background work with a short duration (if using the main Thread)

© 2019 National University of Singapore. All Rights Reserved.

What is Started Service?

- A sample Started Service

```
public class MyStartedService extends Service {
    @Override
    public void onCreate() {
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        super.onStartCommand(intent, flags, startId);

        String action = intent.getAction();
        if (action != null) {
            if (action.compareTo(Consts.START_TASK1) == 0) {
                startTask1();
            }
        }

        return START_STICKY;
    }

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


...
```

- To start a Started Service

```
intent = new Intent(this, MyStartedService.class);
intent.setAction(Consts.START_TASK1);
startService(intent);
```

© 2019 National University of Singapore. All Rights Reserved.

To have a Started Service restarted

- A Started Service can be terminated by Android (e.g. when memory is low)
- Return **START_STICKY** in **onStartCommand()** if Android is to restart the service when circumstances allow (e.g. memory level is back to normal)

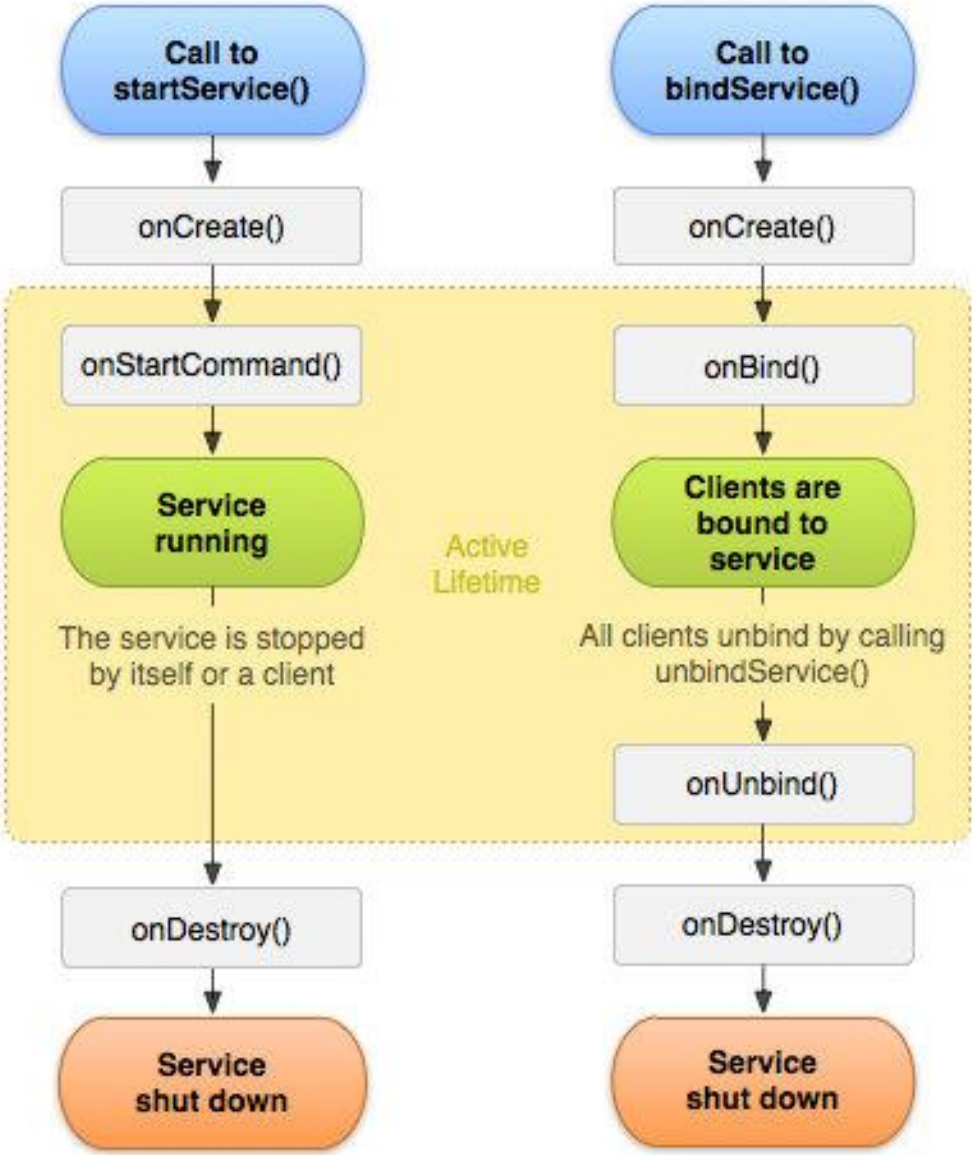
© 2019 National University of Singapore. All Rights Reserved.

What is a Bound Service?

- A Service that allows Android components **to send requests to it and receive responses** (key point)
- **Starts** when the **first** Android component **binds** to it via **bindService()**
- **Stops** when the **last** Android component **unbinds** from it via **unbindService()**

© 2019 National University of Singapore. All Rights Reserved.

Services Lifecycle



© 2019 National University of Singapore. All Rights Reserved.

Setting up a Bound Service

- The Bound Service must provide a Binder object to caller

```
public class MyBoundService extends Service {

    private final IBinder binder = new LocalBinder();
    ServiceCallback callback;

    public class LocalBinder extends Binder {
        MyBoundService getService() {
            return MyBoundService.this;
        }
    }

    public void setCallback(ServiceCallback callback) {
        this.callback = callback;
        this.callback.svcToActivity("Hello from Service!");
    }

    public interface ServiceCallback {
        void svcToActivity(String msg);
    }


...
```

© 2019 National University of Singapore. All Rights Reserved.

Setting up a Bound Service

- The Bound Service will also setup methods that the calling Android component can call

```
public void startTask() {
    carryOn = true;

    runner = new Thread(new Runnable() {
        @Override
        public void run() {
            for (int i=0; carryOn; i++) {
                callback.svcToActivity("Completed item: " + i);
            }
        }
    });
    System.out.println("Thread " + runner.getId() + " has started ...");
    runner.start();
}

public void stopTask() {
    carryOn = false;
}
```

© 2019 National University of Singapore. All Rights Reserved.

Setting up a Bound Service

- The calling Android component will create a **ServiceConnection** object and wait for the Bound Service to connect

```
private ServiceConnection svcConn = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        MyBoundService.LocalBinder binder = (MyBoundService.LocalBinder) service;
        if (binder != null) {
            svc = binder.getService();
            svc.setCallback(MainActivity.this);
        }
    }

    @Override
    public void onServiceDisconnected(ComponentName name) {
        svc = null;
    }
};
```

- The Bound Service connects when the calling Android component calls **bindService()**

```
intent = new Intent(this, MyBoundService.class);
bindService(intent, svcConn, BIND_AUTO_CREATE);
```

© 2019 National University of Singapore. All Rights Reserved.

Setting up a Bound Service

- Once connected, the calling Android component can then use the Bound Service

```
@Override
public void onClick(View v) {
    Intent intent = null;

    switch (v.getId()) {
        case R.id.startTaskBtn:
            if (svc != null)
                svc.startTask()
            break;

        case R.id.stopTaskBtn:
            if (svc != null)
                svc.stopTask();
            break;


 ...
```

© 2019 National University of Singapore. All Rights Reserved.

What is IntentService?

- Spawns **a new worker thread** on your behalf
  - Only a single background thread is running
  - Tasks execute sequentially - internally, it maintains a queue for multiple requests
  - Need not destroy the service; it ends when all tasks have been completed
- onStartCommand() is called first
  - Runs in main UI thread
- OnHandleIntent() is called next
  - Runs in its own worker thread

© 2019 National University of Singapore. All Rights Reserved.

What is a IntentService?

- An IntentService can be started N times
  - But **only one** IntentService will be created (i.e. onCreate() is called once only)
  - Both onStartCommand() and onHandleIntent will be called N times
- Specify the Action to be taken in the Intent before calling IntentService
- In onHandleIntent, check the Action needed and perform the required task

© 2019 National University of Singapore. All Rights Reserved.

IntentService

- A sample IntentService

```
public class MyIntentService extends IntentService {

    public MyIntentService() {
        super("MyIntentService");
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        return super.onStartCommand(intent, flags, startId);
    }

    @Override
    protected void onHandleIntent(Intent intent)
    {
        String action = intent.getAction();
        if (action != null) {
            runTask(action);
        }
    }
```

- Starts the IntentService

```
intent = new Intent(this, MyIntentService.class);
intent.setAction(Consts.START_TASK1);
startService(intent);
```

© 2019 National University of Singapore. All Rights Reserved.

Manifest

- A service must be declared in the manifest
- If the service is called “MyService”,
  - Short form - “.MyService”
  - Long form - “com.example.myapplication.MyService”

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.myapplication">

   <application>
      <service android:name=".MyService" />
   </application>

</manifest>
```

© 2019 National University of Singapore. All Rights Reserved.

References

- Services - https://developer.android.com/guide/components/services

- Android Thread Constructs: Comparisons - http://techtej.blogspot.com/2011/03/android-thread-constructspart-4.html

© 2017 National University of Singapore. All Rights Reserved.

Active Recall

© 2019 National University of Singapore. All Rights Reserved.