
# 20장. 서비스 컴포넌트

# 20.1 서비스 기본 개념

## 20.1.1. 서비스 작성 방법

- 서비스는 백그라운드 작업을 위한 컴포넌트이며 화면과 상관없이 장시간 동안 처리해야 하는 업무를 구 현할 때 사용

- Service라는 클래스를 상속받아 작성

```
public class PlayService extends Service {     //...}
```

- AndroidManifest.xml 파일에 등록

```
<service android:name=".PlayService" ></service>
```

- 서비스를 실행

```
Intent intent=new Intent(this, PlayService.class);
startService(intent);
```

- 서비스 종료

```
Intent intent=new Intent(this, PlayService.class);
stopService(intent);
```

# 20.1 서비스 기본 개념

- bindService ( ) 함수로 실행

```
ServiceConnection connection=new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
    }
    @Override
    public void onServiceDisconnected(ComponentName name) {
    }
};
```

```
Intent bIntent=new Intent(this, BindService.class);
bindService(bIntent, connection, Context.BIND_AUTO_CREATE);
```

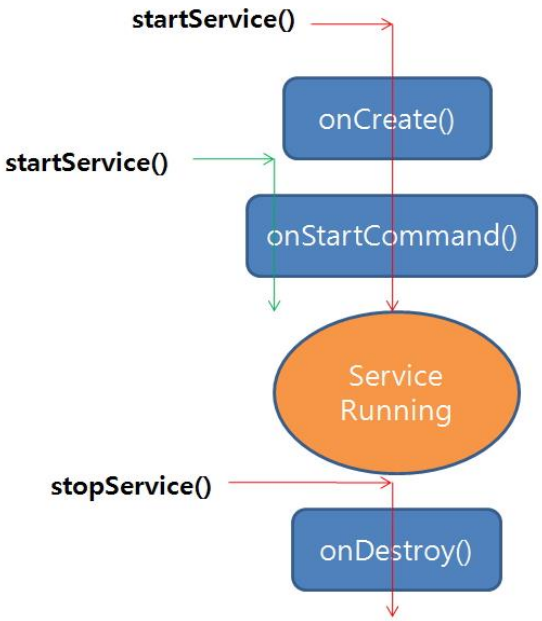
- unbindService ( ) 함수로 종료

```
unbindService(connection);
```

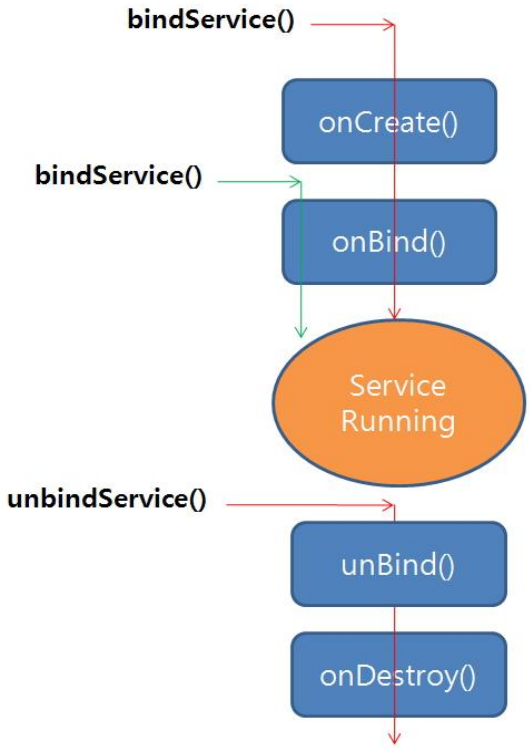
# 20.1 서비스 기본 개념

## 20.1.2. 서비스 생명주기

- startService( ) 함수 이용 시 생명주기



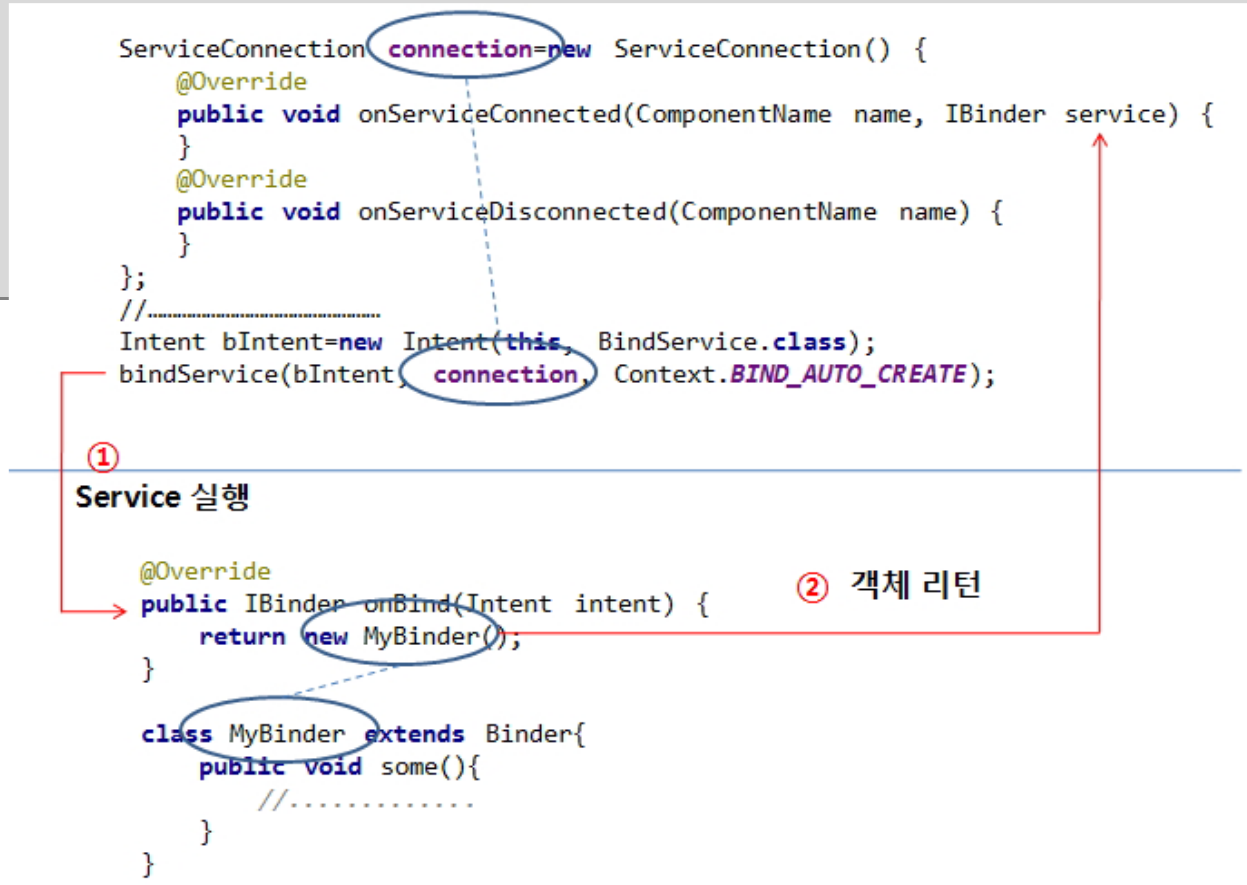

- bindService( ) 함수 이용 시 생명주기




# 20.1 서비스 기본 개념

- onBind( ) 함수

```
public IBinder onBind(Intent intent) {
    return new MyBinder();
}

class MyBinder extends Binder{
    public void some(){
        //...
    }
}
```

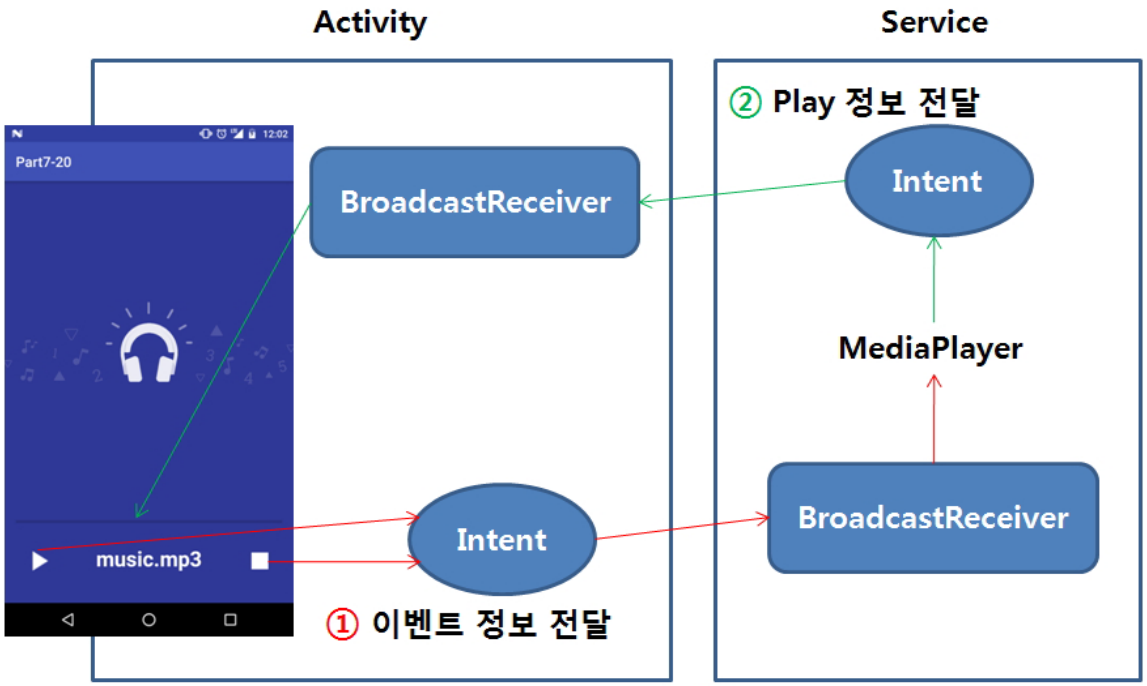


# 20.1 서비스 기본 개념

## 20.1.3. 데이터 공유

- 서비스에서 데이터가 발생한 순간 다른 컴포넌트에 전

- startService( ) 함수로 실행, 브로드캐스트 리시버를 이용하는 방법
- 서비스나 액티비티 내부에 브로드캐스트 리시버를 정의하고, 데이터 전달이 필요할 때 이를 실행하여 브로드캐스트 인텐트에 Extra 데이터로 전달하는 방법




# 20.1 서비스 기본 개념

- 인텐트로 서비스 내부에 구현된 브로드캐스트 리시버를 실행

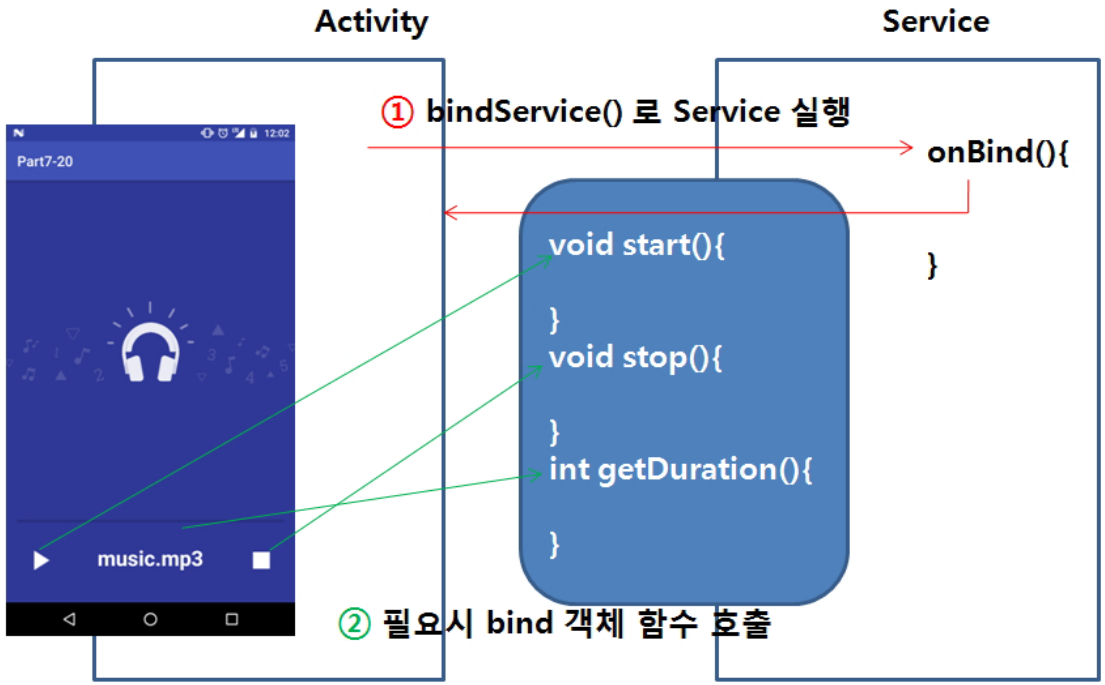
```
BroadcastReceiver receiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        String mode = intent.getStringExtra("mode");

        }
    }
};
```

```
public void onCreate() {
    super.onCreate();
    registerReceiver(receiver, new IntentFilter("com.example.PLAY_TO_SERVICE"));
}
```

```
public void onClick(View v) {
    if(v==playBtn){
        Intent intent=new Intent("com.example.PLAY_TO_SERVICE");
        intent.putExtra("mode","start");
        sendBroadcast(intent);
    }else if(v==stopBtn){
        Intent intent=new Intent("com.example.PLAY_TO_SERVICE");
        intent.putExtra("mode","stop");
        sendBroadcast(intent);
    }
}
```

# 20.1 서비스 기본 개념

- bindService( ) 함수로 실행, 바인드 객체를 이용하는 방법




# Step by Step 20-1 - Service

Service 테스트

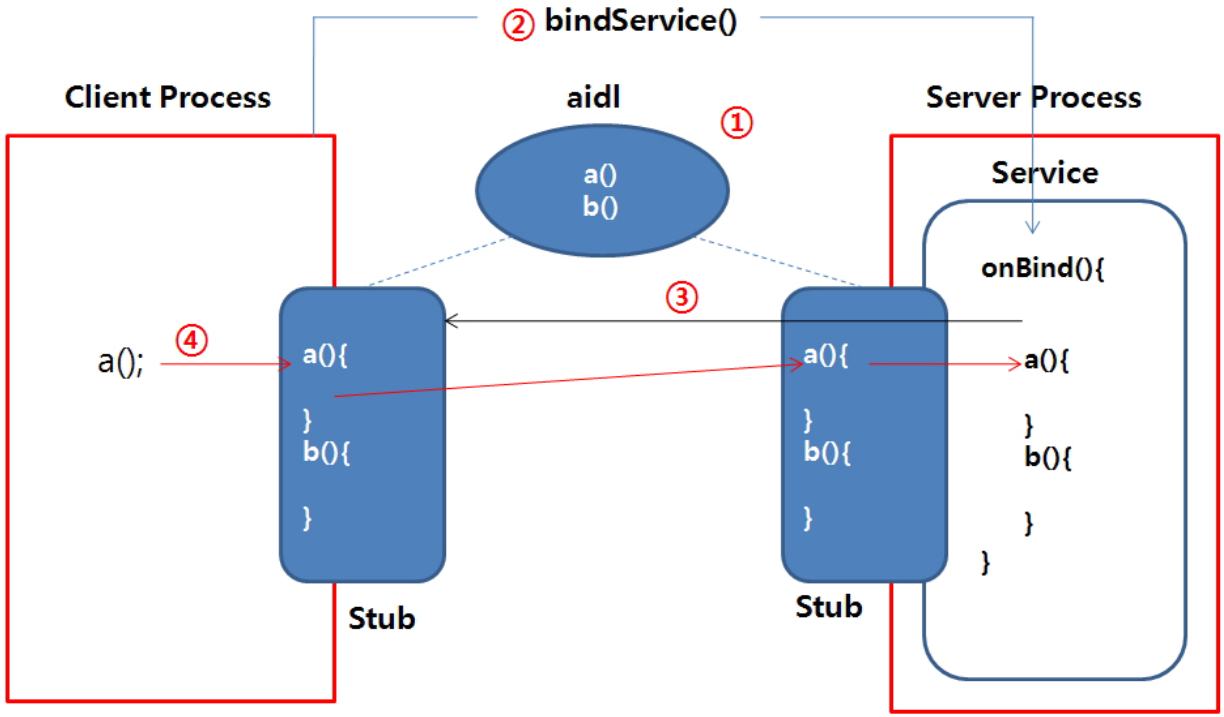
음악 Player이며 테스트 용이성을 위해 지정된 음악을 play, stop하고 ProgressBar에 currentTime 출력
Activity 가 종료되어 다른 앱을 수행하고 있더라도 음악이 계속 플레이
Activity와 Service 구조이며 두 Component간의 데이터를 BroadcastReceiver 방법으로 주고 받는것에 초점
플레이 하는 mp3 파일은 SDCard의 Music 폴더에  music.mp3 라는 이름의 파일이 있다는 가정하에 진행

1. Module 생성
2. PlayService 생성
3. 파일 복사
4. AndroidManifest.xml 작업
5. MainActivity.java 작성
6. PlayService.java 작성
7. 실행


# 20.2 AIDL 의 이해

## 20.2.1. AIDL 프로그램 구조

- AIDL(Android Interface Definition Language)은 안드로이드에서 프로세스 간의 통신을 지칭하는 용어
- AIDL은 bindService ( ) 함수로 서비스 컴포넌트를 실행하여 객체를 공유하고, 공유된 객체의 함수를 호출하면 서 프로세스 간의 통신을 수행




# 20.2 AIDL 의 이해

## 20.2.2. AIDL 작성 방법

- aidl 파일 작성

```
interface IPlayService {
   void start();
   void stop();
}
```

- 서비스 작성

```
public IBinder onBind(Intent intent) {
    return new IPlayService.Stub() {

        @Override
        public void start() throws RemoteException {
            //...
        }
        @Override
        public void stop() throws RemoteException {
            //...        }
    };
}
```

# 20.2 AIDL 의 이해

- bindService( ) 함수에 의해 서비스 실행

```
Intent intent=new Intent("com.example.ACTION_PLAY");
intent.setPackage("com.example.test7_20_aidl");
bindService(intent,connection, Context.BIND_AUTO_CREATE);
```

- 바인드 객체 획득 및 이용

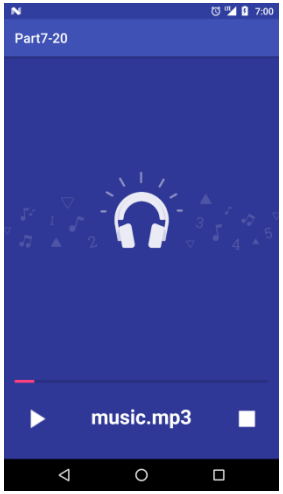
```
ServiceConnection connection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        pService = IPlayService.Stub.asInterface(service);
        //...
    }

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

# Step by Step 20-2 - AIDL

AIDL 테스트

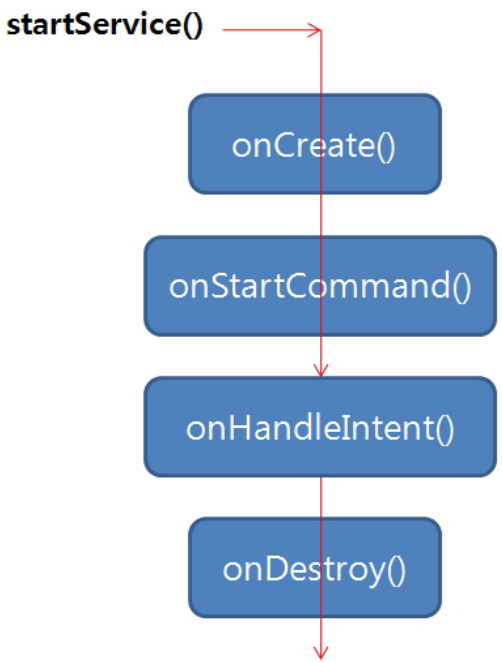
실습 20-1에서 startService() 방법으로 음악을 플레이 했던것을 동일하게 AIDL 로 처리
실습 20-1 과 마찮가지로 SDCard 에 Music 폴더가 있고 그 폴더에 music.mp3 파일이 있다는 가정에 진행

1. Activity 생성
2. 새로운 모듈 생성
3. 파일 복사
4. Part7-20-aidl 모듈 작업 - aidl 파일 준비
5. Part7-20-aidl 모듈 작업 - PlayService 작성
6. Part7-20-aidl 모듈 작업 - AndroidManifest.xml 추가
7. Part7-20 모듈 작업 - aidl 파일 추가
8. Lab20_2Activity.java 작성
9. Part7-20-aidl 모듈 실행
10. Lab20_2Activity.java 실행




# 20.3 IntentService

- 인텐트 서비스는 자신에게 주어진 업 무만 끝나면 자동으로 종료되는 특징




# 20.3 IntentService

```
public class MyIntentService extends IntentService {
    public MyIntentService() {
        super("MyIntentService");
    }
    public void onCreate() {
        super.onCreate();
        Log.d("kkang","onCreate....");
    }
    public int onStartCommand(Intent intent, int flags, int startId) {
        Log.d("kkang"," onStartCommand....");
        return super.onStartCommand(intent, flags, startId);
    }
    public void onDestroy() {
        Log.d("kkang","onDestroy....");
        super.onDestroy();
    }
    protected void onHandleIntent(Intent intent) {
        Log.d("kkang","onHandleIntent start....");
        int sum=0;
        for(int i=1;i<11;i++){
            sum += i;
            SystemClock.sleep(1000);
        }
        Log.d("kkang","onHandleIntent result:"+sum);
        Log.d("kkang","onHandleIntent end....");
    }
}
```

# 20.3 IntentService

- 실행 결과 로그

```
03-15 19:30:08.888 21374-21374/com.example.test7_20 D/kkang: onCreate....
03-15 19:30:08.888 21374-21374/com.example.test7_20 D/kkang: onStartCommand....
03-15 19:30:08.888 21374-21429/com.example.test7_20 D/kkang: onHandleIntent start....
03-15 19:30:18.892 21374-21429/com.example.test7_20 D/kkang: onHandleIntent result:55
03-15 19:30:18.892 21374-21429/com.example.test7_20 D/kkang: onHandleIntent end....
03-15 19:30:18.894 21374-21374/com.example.test7_20 D/kkang: onDestroy....
```

- 여러 번 수행하면 onHandleIntent( ) 함수가 차례로 실행

```
protected void onHandleIntent(Intent intent) {
    String who=intent.getStringExtra("who");
    Log.d("kkang", who+" in.....");
    int sum=0;
    for(int i=1;i<11;i++){
        sum += i;
        SystemClock.sleep(1000);
    }
    Log.d("kkang", who+" out.....");

}
```

# 20.3 IntentService

- 실행 결과 로그

```
03-15 19:38:58.594 22042-22042/? D/kkang: onCreate....
03-15 19:38:58.594 22042-22042/? D/kkang: onStartCommand....
03-15 19:38:58.594 22042-22042/? D/kkang: onStartCommand....
03-15 19:38:58.594 22042-22042/? D/kkang: onStartCommand....
03-15 19:38:58.597 22042-22128/? D/kkang: one in.....
03-15 19:39:08.603 22042-22128/com.example.test7_20 D/kkang: one out.....
03-15 19:39:08.604 22042-22128/com.example.test7_20 D/kkang: two in.....
03-15 19:39:18.612 22042-22128/com.example.test7_20 D/kkang: two out.....
03-15 19:39:18.614 22042-22128/com.example.test7_20 D/kkang: three in.....
03-15 19:39:28.623 22042-22128/com.example.test7_20 D/kkang: three out.....
03-15 19:39:28.626 22042-22042/com.example.test7_20 D/kkang: onDestroy....
```

# 20.4 시스템 서비스

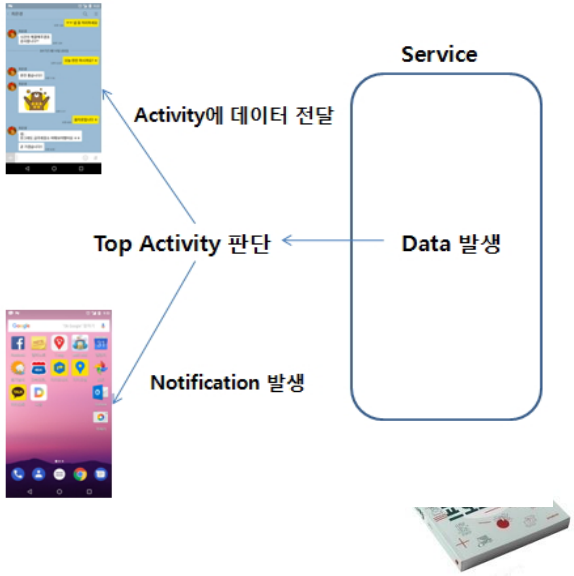
## 20.4.1. ActivityManager

- 앱의 실행 상태와 관련된 다양한 정보를 획득할 목적
- 구동 중인 서비스 목록 획득

```
ActivityManager am = (ActivityManager) getSystemService(Context.ACTIVITY_SERVICE);
List<ActivityManager.RunningServiceInfo> list = am.getRunningServices(100);

for(ActivityManager.RunningServiceInfo info : list) {
    String className=info.service.getClassName();
    String packageName=info.service.getPackageName();    //...
}
```

- 최상위 액티비티 판단

```
ActivityManager manager = (ActivityManager) getSystemService(
Activity.ACTIVITY_SERVICE );
//api level 21.. deprecated..
List<ActivityManager.RunningTaskInfo> list = manager.getRunningTasks(1);
ActivityManager.RunningTaskInfo info=list.get(0);
if(info.topActivity.getClassName().equals("com.example.test7_20.Lab4Activit
y")){
    //...
}else {
    //...
}
```



# 20.4 시스템 서비스

## 20.4.2. PackageManager

- 설치된 앱 목록

```
PackageManager pm = getPackageManager();
List<ApplicationInfo> list = pm.getInstalledApplications(PackageManager.GET_META_DATA);

for(ApplicationInfo info : list) {
    String label = info.loadLabel(pm).toString();
    String packageName=info.packageName;
}
```

- 인텐트에 반응할 컴포넌트 정보

```
PackageManager pm = getPackageManager();
List<ResolveInfo> activities = pm.queryIntentActivities(new Intent(Intent.ACTION_PICK), 0);

for(ResolveInfo info : activities){
    String name=info.loadLabel(pm).toString();
    String packageName=info.activityInfo.applicationInfo.packageName;
}
```

# 20.4 시스템 서비스

## 20.4.3. AlarmManager

- 시스템에 특정 시간이나 주기를 등록하는 기능

```
AlarmManager manager = (AlarmManager)getSystemService(Context.ALARM_SERVICE);
```

```
Intent intent = new Intent(this, Lab4Service.class);
PendingIntent pIntent = PendingIntent.getService(this, 50, intent, PendingIntent.FLAG_UPDATE_CURRENT);
```

```
manager.set(AlarmManager.RTC, System.currentTimeMillis()+10000, pIntent);
```

- set(int type, long triggerAtMillis, PendingIntent operation): 지정된 시간에 한 번 알림
- setRepeating(int type, long triggerAtMillis, long intervalMillis, PendingIntent operation): 반복 알림

- AlarmManager.ELAPSED_REALTIME: 스마트폰이 부팅 후 경과 시간을 기준
- AlarmManager.ELAPSED_REALTIME_WAKEUP: ELAPSED_REALTIME와 동일. 스마트폰이 대기 상태이면 활성 상태로 전환 후 알림
- AlarmManager.RTC: 실제 시간을 기준
- AlarmManager.RTC_WAKEUP: RTC와 동일. 대기 상태이면 스마트폰을 활성 상태로 전환 후 알림

# 20.5 백그라운드 서비스 제한

- 앱이 포그라운드(foreground)에 있는 경우의 서비스 이용에는 아무런 제약사항이 없다.

- 액티비티가 시작되거나 일시 중지되거나 상관없이 보이는 액티비티가 있는 경우
- 포그라운드 서비스가 있는 경우
- 앱의 서비스 중 하나에 바인드하거나 앱의 콘텐츠 제공자 중 하나를 사용하여 앱에 또 다른 포그라운드 앱이 연결된 경우

- 위의 상황과 상관없이 실행되는 서비스는 인텐트에 의한 서비스 실행에 제약

- java.lang.IllegalStateException: Not allowed to start service Intent

```
if (Build.VERSION.SDK_INT>=Build.VERSION_CODES.O) {
context.startForegroundService(intent1);
}
else {
context.startService(intent);
}
```

- 부팅 완료 시점에 실행되는 리시버의 코드이며 Android Oreo일 때는 startForegroundService ( ) 함수에 의해 서비스 인텐트를 발생시키면 서비스가 정상 실행
- startForegroundService ( ) 함수에 의해 실행된 서비스는 빠른 시간 내에(몇초 이내로) 알림 등을 위한 startFore ground( ) 함수를 호출해야 함.

# Step by Step 20-3 - SystemService

ActivityManager, PackageManager, AlarmManager를 테스트

1. Activity 생성
2. 파일 복사
3. Lab4Service.java 생성
4. AndroidManifest.xml 작업
5. Lab4Service 작성
6. Lab20_3Activity 작성
7. Lab20_3Activity.java 실행