



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 구현할 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);
```

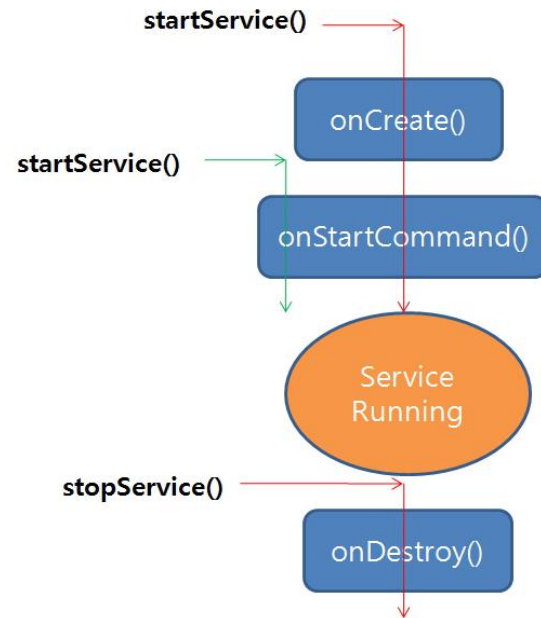
- unbindService () 함수로 종료

```
unbindService(connection);
```

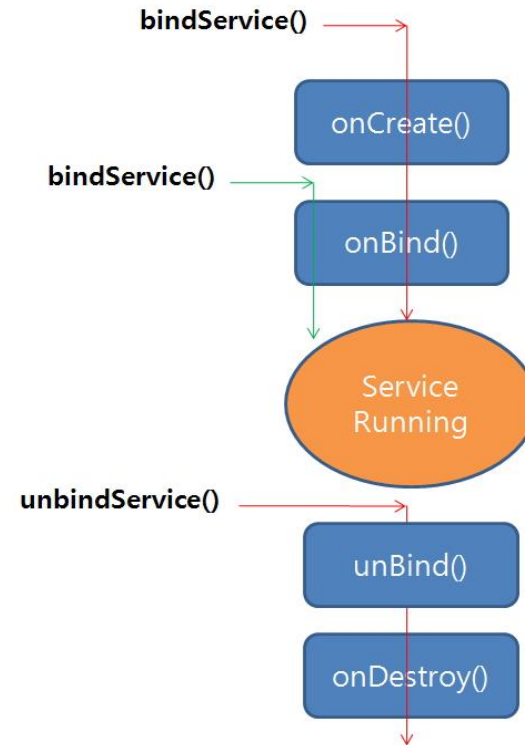
20.1 서비스 기본 개념

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(){  
        //...  
    }  
}
```

```
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);
```

①
Service 실행

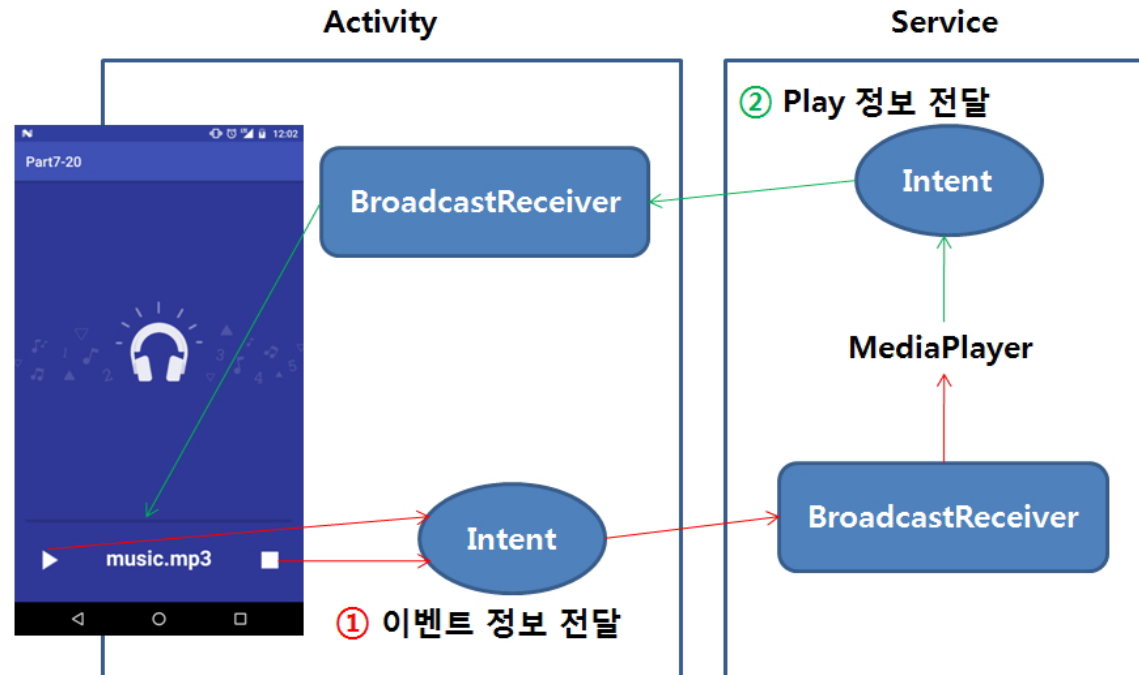
```
@Override  
public IBinder onBind(Intent intent) {  
    return new MyBinder();  
}  
  
class MyBinder extends Binder{  
    public void some(){  
        //.....  
    }  
}
```

② 객체 리턴

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

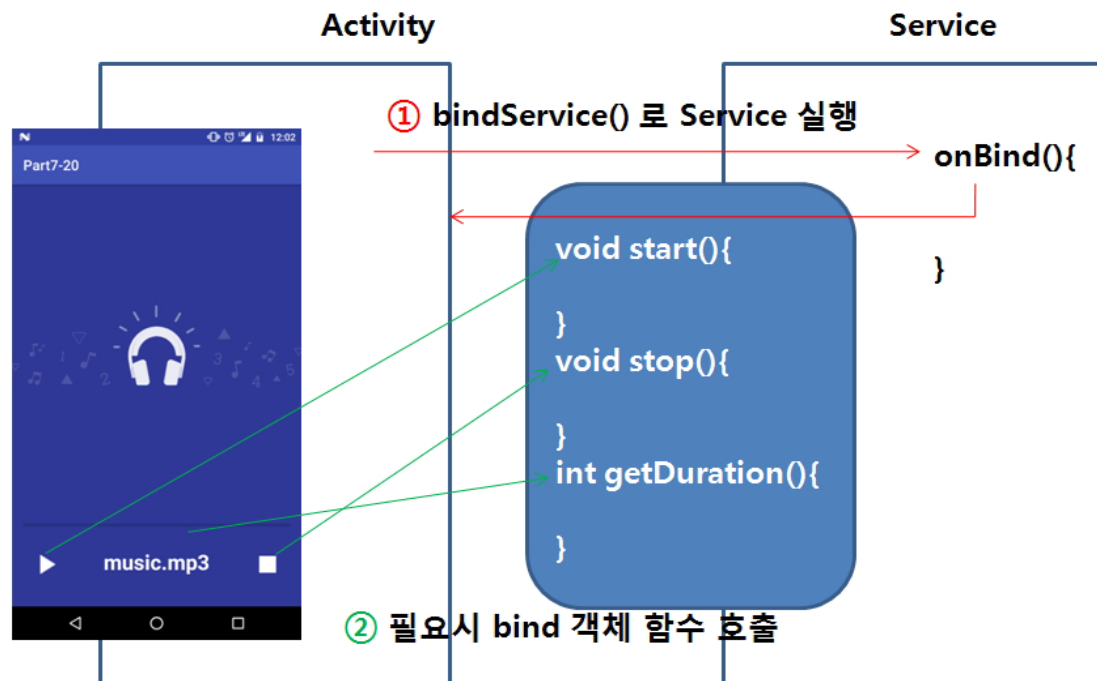
```
BroadcastReceiver receiver = new BroadcastReceiver() {  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        String mode = intent.getStringExtra("mode");  
    }  
};
```

```
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);  
    }  
}
```

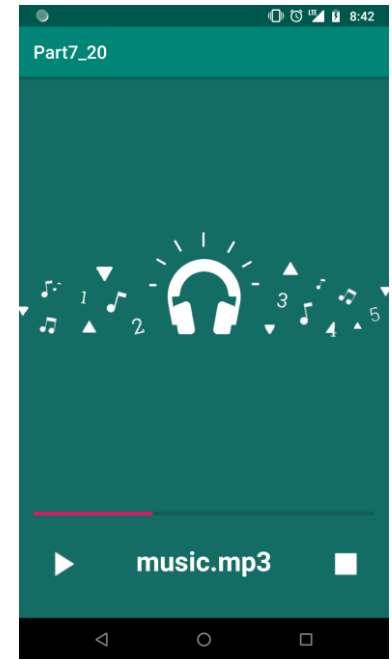
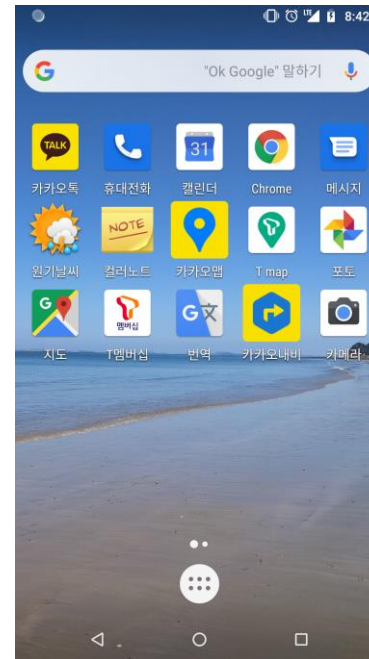
20.1 서비스 기본 개념

- `bindService()` 함수로 실행, 바인드 객체를 이용하는 방법



Step by Step 20-1 - Service

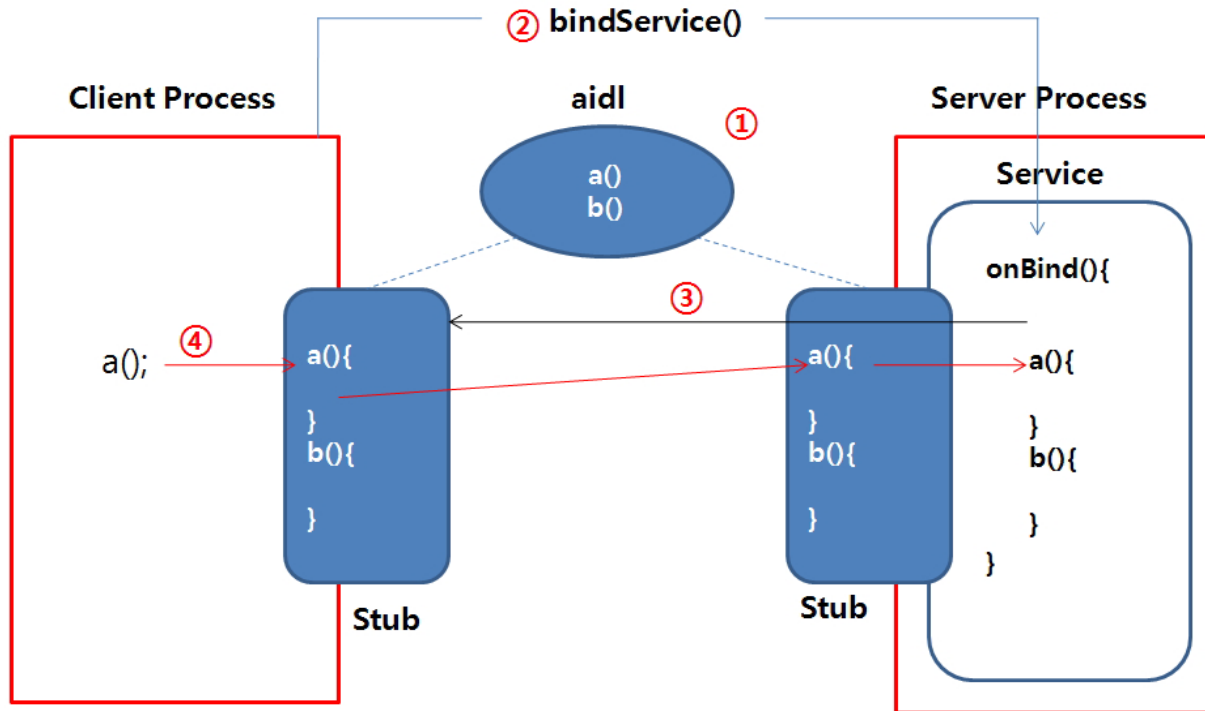
- 모듈 생성
- PlayService 생성
- 파일 복사
- AndroidManifest.xml 작업
- MainActivity 작성
- PlayService 작성
- 실행



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;
    }
};
```

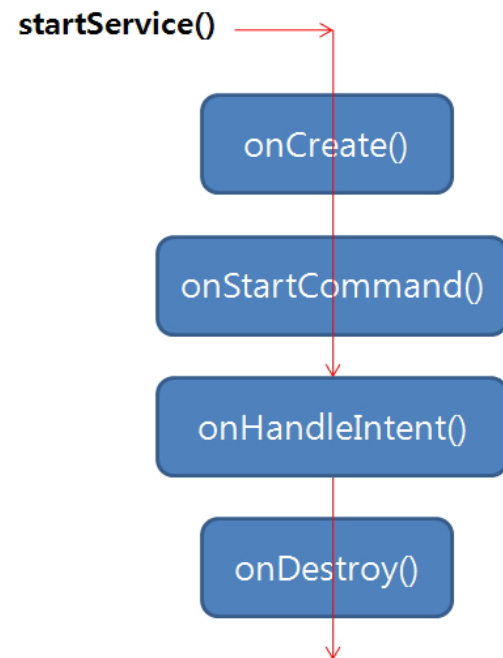
Step by Step 20-2 - AIDL

- 액티비티 생성
- 새로운 모듈 생성
- 파일 복사
- Part7_20_aidl 모듈 작업
- Part7_20 모듈 작업



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

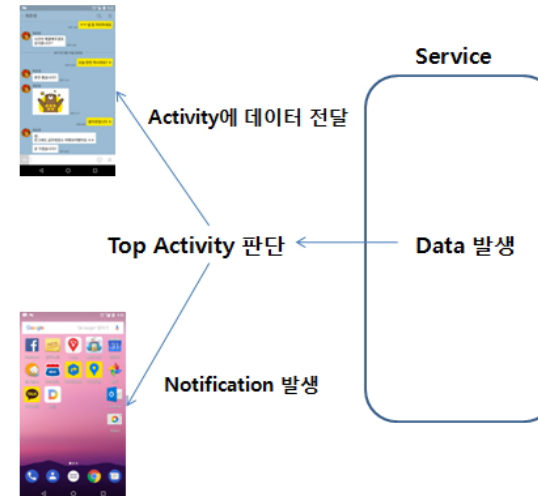
```
03-15 19:38:58.594 22042-22042/? D/kkang: onCreate....
03-15 19:38:58.594 22042-22042/? D/kkang: onStartCommand....
03-15 19:38:58.594 22042-22042/? D/kkang: onStartCommand....
03-15 19:38:58.594 22042-22042/? D/kkang: onStartCommand....
03-15 19:38:58.597 22042-22128/? D/kkang: one in.....
03-15 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
- 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;
}
```

20.5 백그라운드 서비스 제한

Android Oreo 버전부터는 백그라운드 실행 제한

안드로이드 매뉴얼을 보면 앱의 포그라운드를 다음의 경우

- 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

앱이 포그라운드에 있더라도 암시적 인텐트로 서비스를 구동하는 것은 API Level 26부터 불가능

외부 앱의 프로세스가 살아있다면, 해당 앱의 서비스는 `startService ()`와 `bindService ()` 메서드 모두 정상적으로 실행
만약 실행해야 하는 앱이 구동 상태가 아니라면, `startService ()` 메서드로 실행하면 에러가 발생
앱이 실행되지 않았더라도 `bindService ()` 메서드로 실행 가능.

앱이 백그라운드 상황에 있을때 서비스를 정상적으로 실행시키기 위해서는 `startForegroundService ()`
함수에 의해 서비스 인텐트를 발생

`startForegroundService ()` 함수에 의해 실행된 서비스는 빠른 시간 내에(몇초 이내로) 알림 등을 위한 `startForeground ()` 함수를 호출해야한다.

20.5 백그라운드 서비스 제한

```
Notification notification = new NotificationCompat.Builder(this, CHANNEL_ID)
    .setContentTitle("")
    .setContentText(" ").build();
startForeground(1, notification);
```

API Level 28부터는 startForeground() 메서드를 이용해 앱을 포그라운드로 실행하려면 퍼미션 필요

```
<uses-permission android:name="android.permission.FOREGROUND_SERVICE"/>
```