Android : Service

난이도 3 단계 (고급) | 학습시간 150 분

| 학습주제

- ✓ bindService
- ✓ foregroundService

| 학습목표

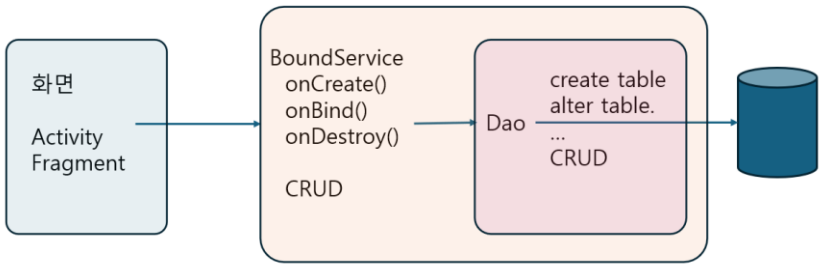
- ✓ SQLite 에 접근 시 ServiceBinder 를 이용하여 접근할 수 있다.
- ✓ Service 를 이용하여 상가의 물품 정보를 가져올 수 있다.

| 지문

상가의 물품을 관리하기 위한 어플리케이션을 만들고자 한다. 아래 내용을 만족하도록 처리하세요.

기존에 DB 를 사용하지 않던 예제를 DB 에 저장하는 것으로 변경하고자 한다.
DB 작업은 화면 UI 작업을 담당하는 Main Thread 에서 처리하는 것이 적절하지 않으므로, 이 작업은 Local Binder 를 사용하여 Service 에서 담당하도록 수정하고자 한다.

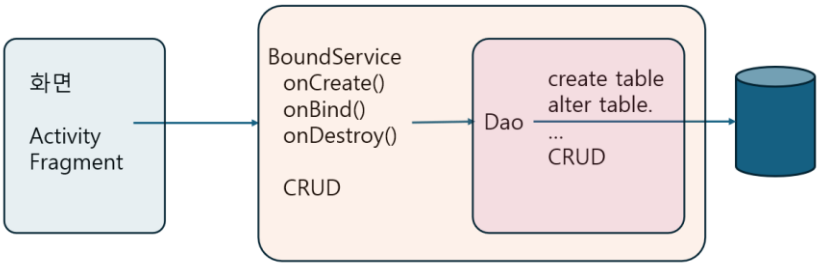
1. **환경설정**
   A. 프로젝트 이름은 ws_android_component_service 로 지정한다.
2. **구현**
   A. 환경 구성
      i. Androidmanifest.xml 을 수정한다.
         1) Service 를 상속받는 BoundService 를 manifest 에 등록한다.
      ii. 서비스 연결은 Activity 에서 해도 되고, Fragment 에서 해도 되겠지만, Fragment 에서 연결하는 것을 전제로 작성한다. (Activity 에서 연결을 원하면 그렇게 해도 됨.)
   B. 동작구현
      i. **Stuff 를 수정한다.**
         1) Primary Key 설정을 위한 id 변수를 추가한다. 기존에 id 대신으로 전달하던 position 은 필요 없어지게 된다.
      ii. **StuffDao 를 구현한다.**
         1) DBHelper 와 SQLiteDatabase 를 이용하여 Stuff Database 를 구축하고 접근할 수 있다.
      iii. **BoundService 를 구현한다.**
         1) Service class 를 상속받아 구현한다.
         2) LocalBinder 를 이용하여 Service 객체를 관리한다.
         3) onCreate 에서 DB 에 접속하기 위하여 dao 를 생성한다.
         4) StuffDao 에 정의된 함수를 호출할 수 있다.



iv. **StuffFragment 를 수정한다.**

1) Service Bind 를 위한 ServiceConnection 객체를 선언한다. (구현한 내용에 따라서 Connection 객체는 Activit, Fragment 등 어디에서나 생성해도 무관함)
2) onStart 에서 service 를 bind 한다. (bindService 를 호출)
3) bind 성공하면 목록을 조회하여 화면에 출력한다.
   DB 를 조회하는 로직은 onServiceConnected 성공 이후에 수행해야 한다. !!!
4) StuffEditFragment 의 결과에 따라 BoundService 의 함수를 호출하여 DB 를 조작한다.

v. **주의!! Fragment 에서 직접 DAO 를 직접 호출하지 말아야 한다.**
화면에서는 binder 로 Service 를 부르고, Service 에서 background DB 작업을 담당하는 DAO 를 호출하는 구조가 된다.



| 자료 1: 샘플 코드(StuffDao.kt)

```
1.  class StuffDao(
2.      context: Context,
3.      name: String,
4.      factory: SQLiteDatabase.CursorFactory?,
5.      version: Int
6.  ) : SQLiteOpenHelper(context, name, factory, version) {
7.
8.      // 테이블 생성
9.      override fun onCreate(db: SQLiteDatabase) {
10.         db.execSQL("""CREATE TABLE if not exists $TABLE_NAME (
11.                     $STUFF_ID INTEGER PRIMARY KEY AUTOINCREMENT,
12.                     $STUFF_NAME VARCHAR(50),
13.                     $STUFF_CNT INTEGER
14.                 )
15.                 """)
16.     }
17.
18.     override fun onUpgrade(db: SQLiteDatabase, p1: Int, p2: Int) { //테이블 삭제 후 생성
19.         db.execSQL("DROP TABLE IF EXISTS $TABLE_NAME")
20.         onCreate(db)
21.     }
22.
23.     override fun onOpen(db: SQLiteDatabase) {
```

```
24.         super.onOpen(db)
25.         this.db = db
26.         Log.d(TAG, "onOpen: database 준비 완료")
27.     }
28.          //DB 선언부
29.     lateinit var db: SQLiteDatabase
30.
31.     private val TABLE_NAME = "Stuff"
32.     private val STUFF_ID = "_id"
33.     private val STUFF_NAME = "name"
34.     private val STUFF_CNT = "count"
35.
36.      // 물품 CRUD 구현 !!!
37.
38.     // 물품 등록
39.     // 특정 물품 조회 method
40.     // 물품 조회 method
41.     // 물품정보 변경
42.     // 물품 삭제 method
43.
44.
```

| 자료 2: 서비스 코드(BoundService.kt)

```
1.  // 이 클래스를 activity 에서 호출한다.
2.  class BoundService : Service() {
3.
4.
5.      private lateinit var stuffDAO: StuffDao
6.
7.      override fun onCreate() {
8.          super.onCreate()
9.          stuffDAO = StuffDao(this, "clean_store.db",null, 1)
10.         stuffDAO.writableDatabase
11.     }
12.
13. ...
14.     // onbind() 및 물품추가, 수정, 삭제 등 activity 에서 호출할 method 구현, (dao 의 method 호출)



15.     override fun onDestroy() {
16.         stuffDAO.close()
17.         super.onDestroy()
18.     }
19. }
```

Android : Service