05

05-3. Message

서비스

15-2 바인딩 서비스

메신저 바인딩

- API에서 제공하는 Messenger 객체를 바인딩하는 방법
- Messenger 객체를 이용하는 방법은 프로세스 간 통신할 때도 사용

- handleMessage() 함수는 외부에서 서비스에 데이터를 전달할 때 자동으로 호출
- 전달한 데이터는 Message 타입
- Message의 what값으로는 어떤 성격의 데이터인지를 구분하며 obj 속성으로는 전달된 데이터를 가져옵니다.
- onBind() 함수의 반환값으로 Messenger 객체를 생성하면서 생성자 매개변수로 Handler를 구현한 객체를 지정

15-2 바인딩 서비스

• 메신저 객체를 이용하는 서비스 코드

```
class MyService : Service() {
    lateinit var messenger: Messenger
    internal class IncomingHandler(
        context: Context,
        private val applicationContext: Context = context.applicationContext
    ) : Handler(Looper.getMainLooper()) {
        override fun handleMessage(msg: Message) {
            when (msg.what) {
                10 ->
                    Toast.makeText(applicationContext, "${msg.obj}",
                                Toast.LENGTH_SHORT).show()
                20 ->
                    Toast.makeText(applicationContext, "${msg.obj}",
                                Toast.LENGTH_SHORT).show()
                else -> super.handleMessage(msg)
            }
        }
    }
    override fun onBind(intent: Intent): IBinder? {
        messenger = Messenger(IncomingHandler(this))
        return messenger.binder
    }
}
```

15-2 바인딩 서비스

- 액티비티 코드
  - onServiceConnected() 함수의 매개변수로 넘어온 객체를 Messenger의 생성자 매개변수에 지정

• 메신저 객체를 이용하는 액티비티 코드

```
class MainActivity : AppCompatActivity() {
    lateinit var messenger: Messenger
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        (... 생략 ...)
        val intent = Intent(this, MyService::class.java)
        bindService(intent, connection, Context.BIND_AUTO_CREATE)
    }
    (... 생략 ...)
    val connection: ServiceConnection = object : ServiceConnection {
        override fun onServiceConnected(name: ComponentName?, service: IBinder?) {
            messenger = Messenger(service)
        }
        override fun onServiceDisconnected(name: ComponentName?) {
        }
    }
}
```

• 서비스에 데이터 전달

```
val msg = Message()
msg.what = 10
msg.obj = "hello"
messenger.send(msg)
```

15-2 바인딩 서비스

- 외부 앱 연동
  - 외부 앱의 서비스를 bindService() 함수로 실행하려면 먼저 서비스를 등록한 매니페스트에 외부 앱을 연동할 수 있게끔 <intent-filter>가 선언
  - bindService() 함수로 실행하는 앱에서는 외부 앱에 접근할 수 있도록 매니페스트에 queries 등록

• 매니페스트에 인텐트 필터 선언

```
<service
    android:name=".MyService"
    android:exported="true">
    <intent-filter>
        <action android:name="ACTION_OUTER_SERVICE" />
    </intent-filter>
</service>
```

• 매니페스트에 패키지 정보 등록

```
<manifest  ... 생략 ... >
    (... 생략 ...)
    <queries>
        <package android:name="com.example.test_outter" />
    </queries>
</manifest>
```

15-2 바인딩 서비스

- 외부 앱을 연동하고자 한다면 bindService() 함수로 발생하는 인텐트에 실행 대상인 앱의 패키지명을 명시
- 프로세스 간 통신에서는 주고받는 데이터는 Parcelable이나 Bundle 타입이어야

• 실행할 앱의 패키지명 명시

```
val intent = Intent("ACTION_OUTER_SERVICE")
intent.setPackage("com.example.test_outter")
bindService(intent, connection, Context.BIND_AUTO_CREATE)
```

• 번들 객체 이용

```
val bundle = Bundle()
bundle.putString("data1", "hello")
bundle.putInt("data2", 10)

val msg = Message()
msg.what = 10
msg.obj = bundle
messenger.send(msg)
```

감사합니다

단단히 마음먹고 떠난 사람은
산꼭대기에 도착할 수 있다.
산은 올라가는 사람에게만 정복된다.

윌리엄 셰익스피어
William Shakespeare