05

# 05-1. Service

서비스

# 15-1 서비스 이해하기

## 서비스 생성과 실행

- 서비스 컴포넌트는 Service 클래스를 상속받아서 작성
- 서비스도 컴포넌트이므로 매니페스트에 등록

• 서비스 컴포넌트 생성

```
class MyService : Service() {
    override fun onBind(intent: Intent): IBinder? {
        return null
    }
}
```

• 서비스 컴포넌트 등록

```
<service
    android:name=".MyService"
    android:enabled="true"
    android:exported="true"></service>
```

# 15-1 서비스 이해하기

- startService() 함수로 실행
  - startService() 함수로 서비스를 실행하려면 해당 서비스를 인텐트에 담아서 매개변수로 전달
  - 외부 앱의 서비스라면 암시적 인텐트로 실행해야 하므로 setPackage() 함수를 이용해 앱의 패키지명을 명시
  - 서비스를 종료하려면 stopService() 함수로 인텐트를 전달

• 서비스 실행

```
val intent = Intent(this, MyService::class.java)
startService(intent)
```

• 서비스 종료

```
val intent = Intent(this, MyService::class.java)
stopService(intent)
```

# 15-1 서비스 이해하기

- startService() 함수로 실행
  - 만약 외부 앱의 서비스라면 암시적 인텐트로 실행해야 하므로 setPackage() 함수를 이용해 앱의 패키지명을 명시
  - setPackage()를 사용하였다면 매니페스트 파일에 연동하고 자 하는 앱의 패키지가 등록되어 있어야 합니다.

• 암시적 인텐트로 실행

```
val intent = Intent("ACTION_OUTER_SERVICE")
intent.setPackage("com.example.test_outer")
startService(intent)
```

• 외부 앱 패키지 공개 상태 지정

```
<queries>
    <package android:name="com.example.test_outer" />
</queries>
```

# 15-1 서비스 이해하기

- bindService() 함수로 실행
  - ServiceConnection 인터페이스를 구현한 객체를 준비
  - onServiceConnected()는 bindService() 함수로 서비스를 구동할 때 자동으로 호출
  - onServiceDisconnected()는 unbindService() 함수로 서비스를 종료할 때 자동으로 호출

• ServiceConnection 인터페이스 구현

```
val connection: ServiceConnection = object : ServiceConnection {
    override fun onServiceConnected(name: ComponentName?, service: IBinder?) { }
    override fun onServiceDisconnected(name: ComponentName?) { }
}
```

• 서비스 실행

```
val intent = Intent(this, MyService::class.java)
bindService(intent, connection, Context.BIND_AUTO_CREATE)
```

• 서비스 종료

```
unbindService(connection)
```

# 15-1 서비스 이해하기

## 서비스 생명주기

- 서비스를 실행하는 2가지 방법은 startService()와 bindService()이므로 어느 함수를 이용해 서비스를 실행하는지에 따라 생명주기가 나뉩니다.
- startService() 함수에서 서비스 객체를 생성하면 onCreate() → onStartCommand() 함수가 호출
- onCreate() 함수는 서비스 객체가 생성될 때 처음에 한 번만 호출
- onStartCommand() 함수는 startSer vice() 함수가 실행될 때마다 반복해서 호출
- stopService() 함수로 서비스가 종료되면 바로 전에 onDestory() 함수가 호출

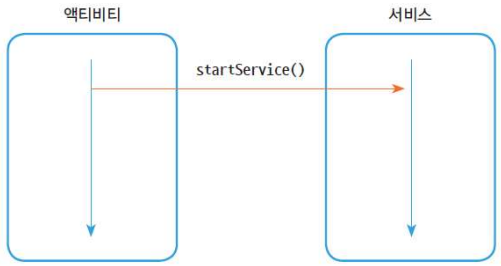
# 15-1 서비스 이해하기

- bindService() 함수에서 서비스 객체를 생성하면 onCreate → onBind() 함수가 호출
- 다시 bindService() 함수로 실행하면 onBind() 함수만 다시 호출
- unbindService() 함수로 서비스를 종료하면 onUnbind() → onDestory() 함수까지 실행

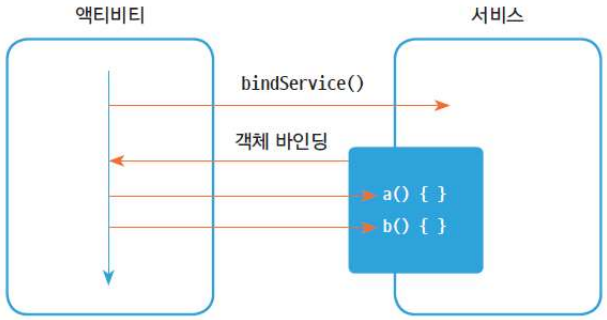
# 15-2 바인딩 서비스

## IBinder 객체 바인딩

- startService() 함수로 서비스를 실행했다고 가정




- 서비스와 액티비티가 상호작용 해야 할 때 bindService()



# 15-2 바인딩 서비스

- 서비스 코드
  - onBind() 함수의 반환 타입은 IBinder 인터페이스
  - 서비스를 실행한 곳에서 이 클래스의 함수를 호출하면서 매개변수와 반환값으로 데이터를 주고받습니다.

• 서비스에서 객체 바인딩

```
class MyBinder : Binder() {
    fun funA(arg: Int) {
    }
    fun funB(arg: Int): Int {
        return arg * arg
    }
}
override fun onBind(intent: Intent): IBinder? {
    return MyBinder()
}
```

# 15-2 바인딩 서비스

- 액티비티 코드
  - onBind() 함수에서 반환한 객체를 ServiceConnection 인터페이스를 구현한 객체의 onServiceConnected() 함수로 받을 수 있습니다.

• 액티비티에서 객체를 전달받아 사용

```
val connection: ServiceConnection = object : ServiceConnection {
    override fun onServiceConnected(name: ComponentName?, service: IBinder?) {
        serviceBinder = service as MyService.MyBinder
    }
    override fun onServiceDisconnected(name: ComponentName?) {
    }
}
```

• 서비스에서 바인딩한 객체의 함수 호출

```
serviceBinder.funA(10)
```

# 감사합니다

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

윌리엄 셰익스피어
William Shakespeare