

# WaffleStudio 2023 Rookies 21.5 Seminar

Android Seminar 3  
2023.10.27 (금)

Instructor : 양주현 (@JuTaK97) / TA : 송동엽 (@eastshine2741)

# 오늘 배울 것

- Network
  - Error Handling
  - Authorization (Token)
- Local Data
- Fragment

# Network Error Handling (1)

Call과 CallBack을 이용해 response를 받을 경우...

```
lists.enqueue(object : Callback<List<SimpleWordList>> {  
    override fun onFailure(call: Call<List<SimpleWordList>>, t: Throwable) {  
        // (1)  
    }  
  
    override fun onResponse(  
        call: Call<List<SimpleWordList>>,  
        response: Response<List<SimpleWordList>>  
    ) {  
        if (response.isSuccessful) {  
            // (2)  
        } else {  
            // (3)  
        }  
    }  
})
```

## 1. 응답이 아예 안 온 경우(Http Fail)

- 네트워크 없음
- 틀린 endpoint 주소

## 2. Successful (code가 2XX)

## 3. Not Successful (다른 code)

# Network Error Handling (2)

suspend fun 과 Coroutine으로 받을 경우...

```
// ViewModel
suspend fun postWordList(...) {
    val lists = repository.postNewWordList(...)
    _liveData.postValue(lists)
}

// Activity
CoroutineScope(Dispatchers.IO).launch {
    try {
        viewModel.postWordList(...)
    } catch (e: Exception) {
        // (?)
    }
}
```

Exception을 받았는데...

내부의 error message는 어디에??

# Network Error Handling (3)

Retrofit 내부를 보면, `HttpException` 이라는 `Exception`으로 래핑해서 반환해 준다.  
(궁금하면 [링크](#))

```
public HttpException(Response<?> response) {  
    super(getMessage(response));  
    this.code = response.code();  
    this.message = response.message();  
    this.response = response;  
}
```

내부에는

1. code
2. message
3. response 가 있다.

서버가 보내준 에러 메시지는 아래와 같이 가져올 수 있다. (두 번은 못 가져온다! [관련 내용](#))

```
HttpException.response()?.errorBody()?.string()
```

# Network Error Handling (4)

에러 종류마다 어떻게 대응할 지 분기를 나눠야 하는데...

서버는 에러도 JSON 형식으로 보내 줄 텐데... 그러면 역직렬화도 해야 되는데...

이걸 전부 API 호출하는 곳의 catch에서 매번 하기??

(고급 내용)

키워드 : Call Adapter

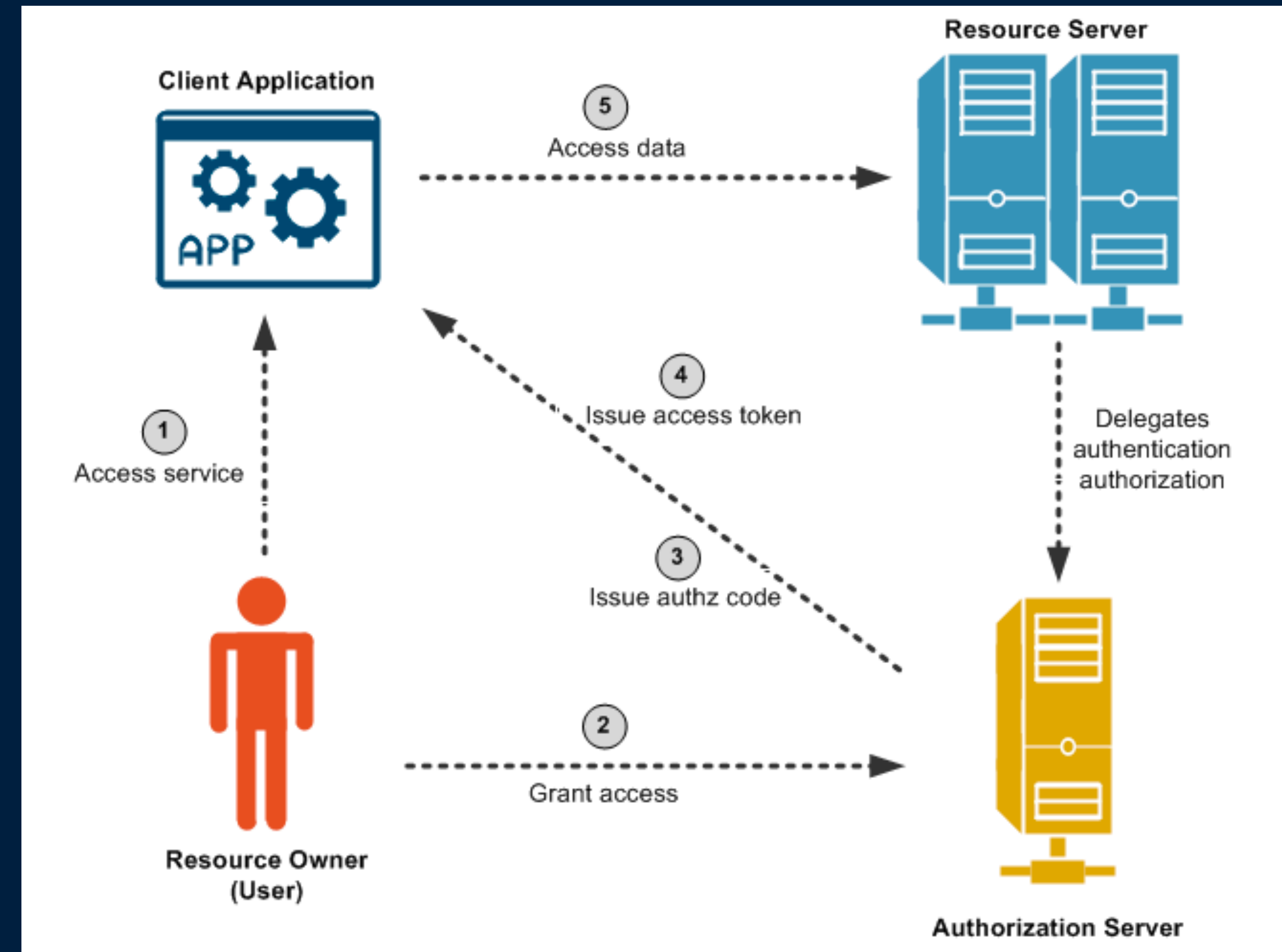
궁금하면 찾아보세요! (사용 예시: SNUTT 코드1 코드2 코드3)

Call Adapter에 대해 더 자세히 알고 싶으면 코모에서...

# Authorization (1)

Token : 서버가 유저를 인식하는 방식

1. 클라이언트가 로그인 시 서버는 토큰을 발급
2. 클라이언트는 매 request마다 헤더에 token을 넣어서 서버에 전달
3. 서버는 token을 검증해서 사용자를 인식





# Authorization (2)

OkHttp의 interceptor를 사용해서 헤더를 넣을 수 있다.

```
return OkHttpClient.Builder()
    .addInterceptor { chain ->
        val newRequest = chain.request()
            .newBuilder()
            .addHeader("x-access-token", "토큰 값")
            .build()
        chain.proceed(newRequest)
    }
```



# Local Data (1)

세미나 0에서...

1. 내가 원하는 UI 를 만드는 방법
2. 사용자 상호작용 (UX) 를 만드는 방법
3. 외부 서버와 통신하는 방법
4. 데이터를 저장하고 읽어오는 방법
5. 좀 더 **\*\*좋은\*\*** 코드를 작성하는 방법

대략 1~3번은 배웠으니, 다음은 4번의 차례

# Local Data (2)

## 1. Room DB

- 안드로이드 OS에서는 SQLite 기반의 로컬 데이터베이스인 Room DB를 제공
- 관계형 데이터베이스
- 기초적인 SQL 수준 지식만 있어도 쉽게 사용 가능!

<https://developer.android.com/training/data-storage/room?hl=ko>

-> 진짜 따라하기만 하면 똑딱 (짱쉬움)

# Local Data (2)

## 2. SharedPreferences

- DB가 필요가 없는 간단한 데이터를 저장하고 싶을 때는?

예시) token을 기기에 저장해놓고 자동 로그인을 하고 싶을 때

- Key-Value 기반의 간단한 값을 저장할 때 SharedPreferences 사용
- Int, Float, String 등 primitive type만 저장할 수 있다
- 하지만 Moshi 등의 직렬화/역직렬화 도구를 사용하면 객체를 String으로 바꿔 저장 가능

<https://developer.android.com/training/data-storage/shared-preferences?hl=ko>

# Local Data (3)

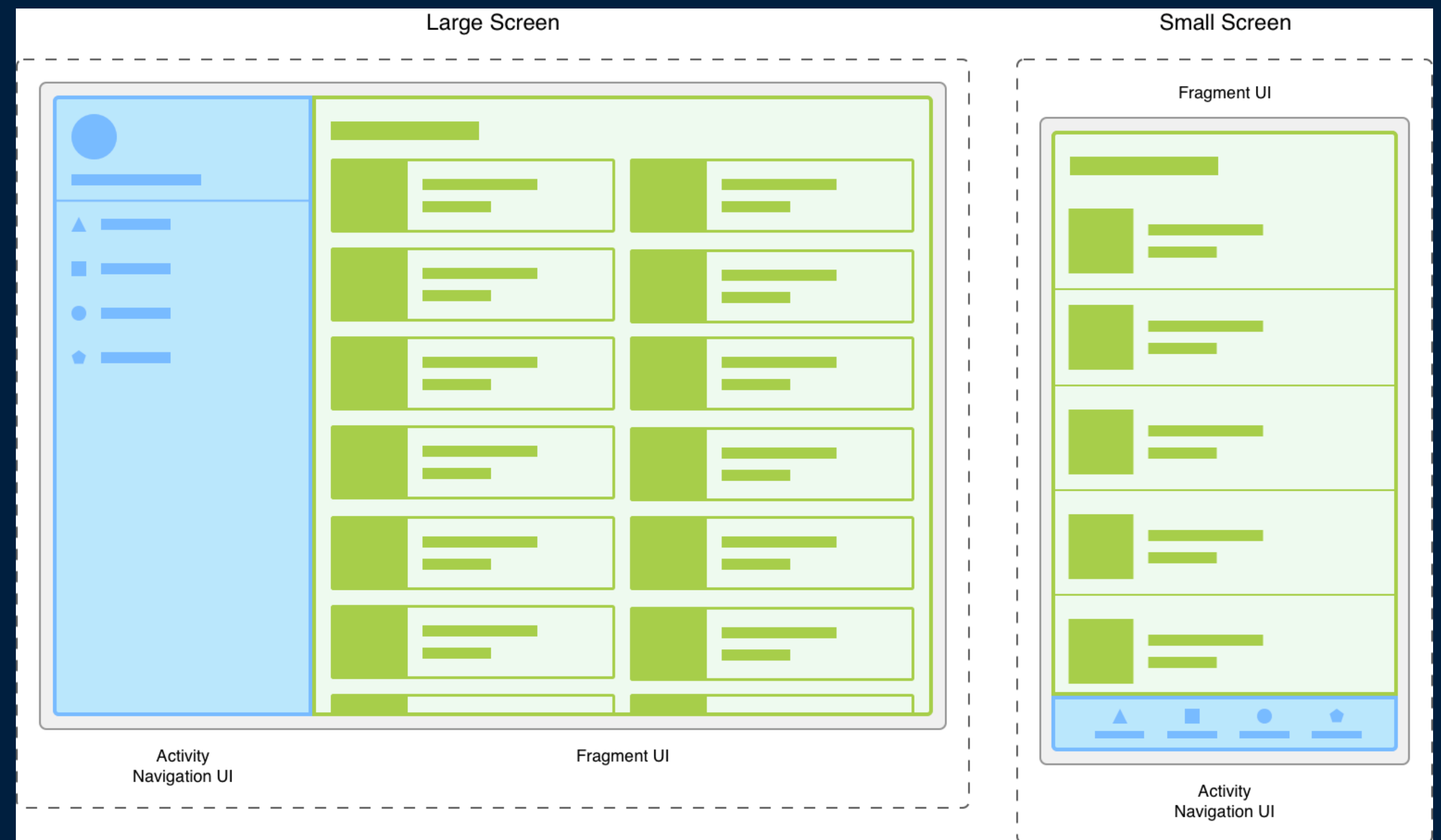
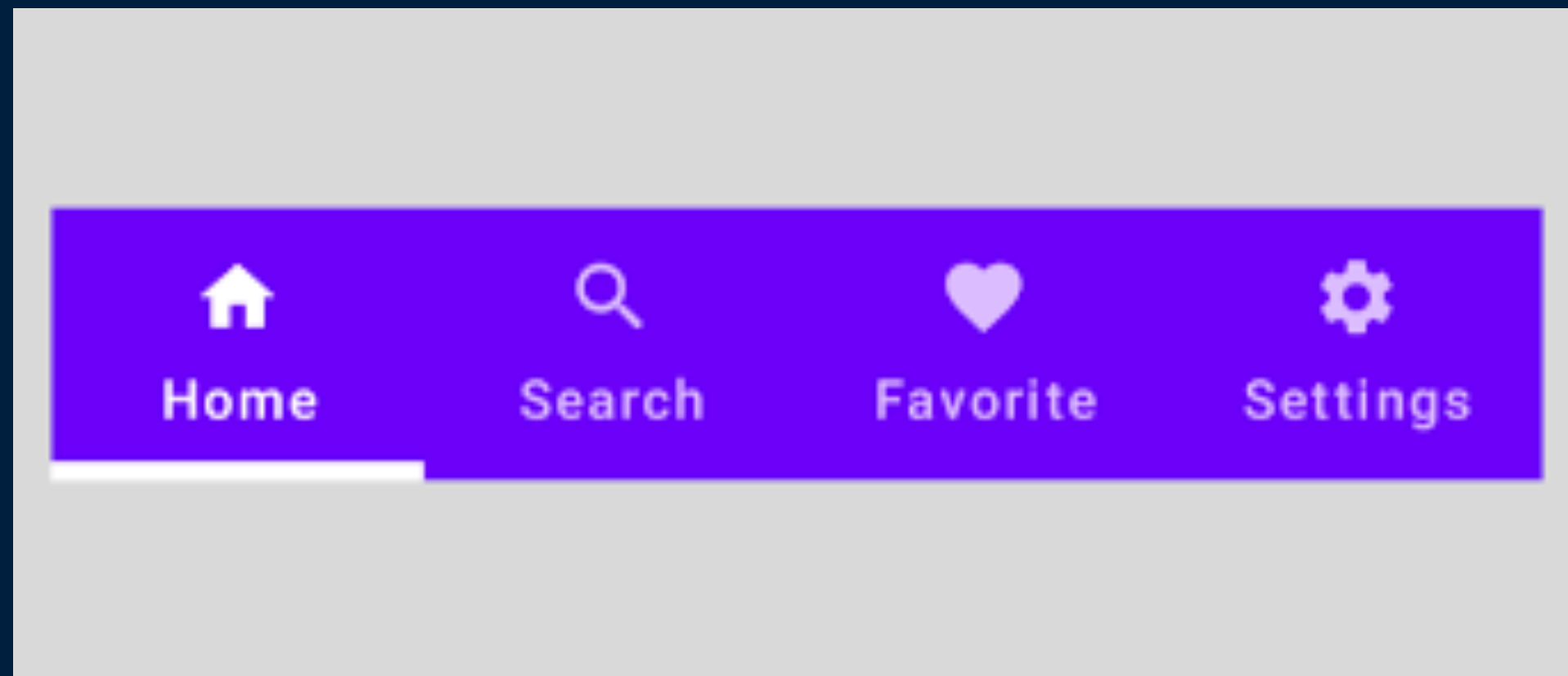
## 3. DataStore

- 가장 최신 솔루션
- SharedPreferences와 비슷하게 키-값 쌍으로 값을 저장
- 내부적으로 protobuf 라는 것을 사용 (하지만 사용자에게는 고수준 API가 제공된다)
- Kotlin Coroutine, Kotlin flow를 이용해 비동기적으로 데이터를 읽고 쓸 수 있다.

<https://developer.android.com/topic/libraries/architecture/datastore?hl=ko>

# Fragment (1)

지금까지의 화면 전환과 내비게이션은? Activity, Intent, startActivity



이런 레이아웃을 여러 개의 Activity로 구성한다면? 너무 무겁다.

# Fragment (2)

- 재사용 가능한 일종의 sub-activity (모듈)
- Activity의 한계를 극복할 수 있다.
  1. 근본적으로 full-screen인 Activity와 다르게 유동적으로 구성 가능
  2. Intent를 사용한 Activity 전환보다 쉽고 가볍게 Fragment끼리 화면 전환 가능
- Activity의 생명 주기에 종속된 자체 생명 주기가 존재
- Fragment를 사용하기 위해서는 FragmentActivity 하위 액티비티를 사용  
예: ComponentActivity, AppCompatActivity

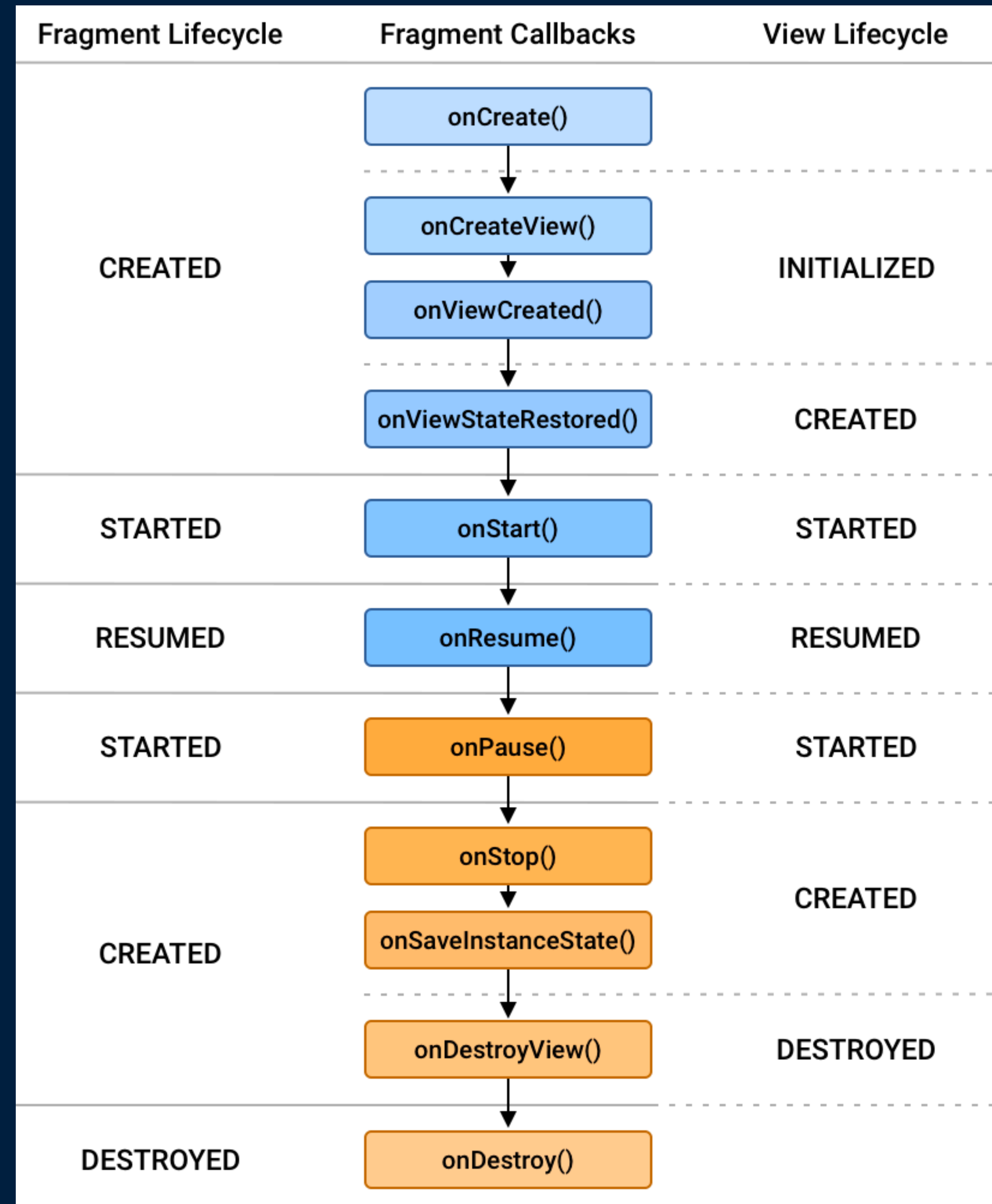


# Fragment (3)

프래그먼트 또한 생명 주기를 갖는다.

[간략 설명]

- onAttach와 onDetach는 Fragment가 자신의 호스팅 Activity와 연결/해제되는 과정
- onCreate와 onDestroy는 Fragment 객체가 생성되고 파괴되는 과정
- onStart&onResume, onPause&onStop은 Activity 수명 주기와 비슷한 과정
- onCreateView, onDestroyView는 Fragment의 UI View와 관련된 과정





# Fragment (4)

<https://developer.android.com/guide/fragments/create?hl=ko>

방법 1 : XML을 통해 프래그먼트 추가

```
<!-- res/layout/example_activity.xml -->
<androidx.fragment.app.FragmentContainerView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragment_container_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:name="com.example.ExampleFragment" /> <!-- 여기가 중요 -->
```

# Fragment (5)

## 방법 2 : 직접 추가

```
<!-- res/layout/example_activity.xml -->
<androidx.fragment.app.FragmentContainerView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragment_container_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>

// Activity의 onCreate
if (savedInstanceState == null) {
    supportFragmentManager.commit {
        setReorderingAllowed(true)
        add<ExampleFragment>(R.id.fragment_container_view)
    }
}
```

# Fragment (5)

## 방법 2 : 직접 추가

```
<!-- res/layout/example_activity.xml -->
<androidx.fragment.app.FragmentContainerView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragment_container_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>

// Activity의 onCreate
if (savedInstanceState == null) {
    supportFragmentManager.commit {
        setReorderingAllowed(true)
        add<ExampleFragment>(R.id.fragment_container_view)
    }
}
```

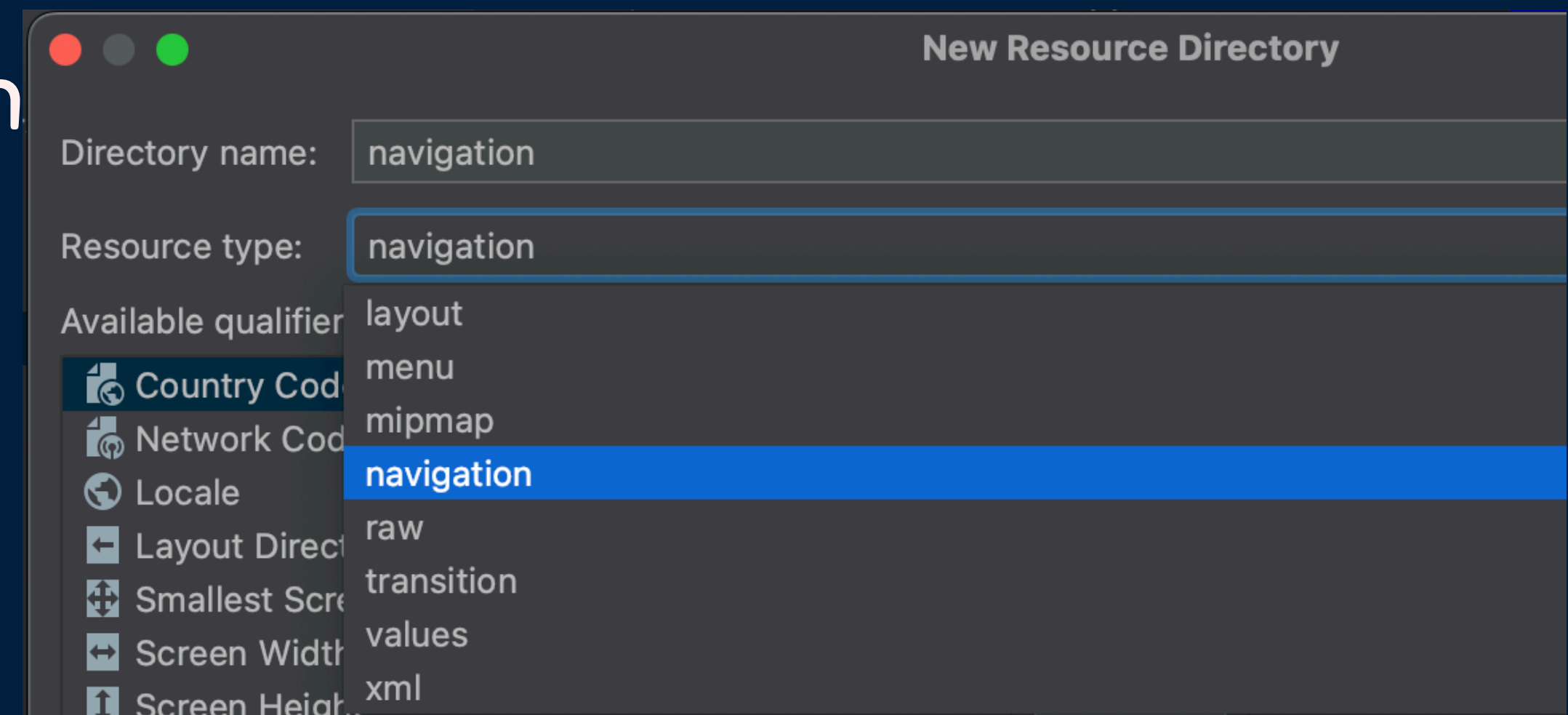
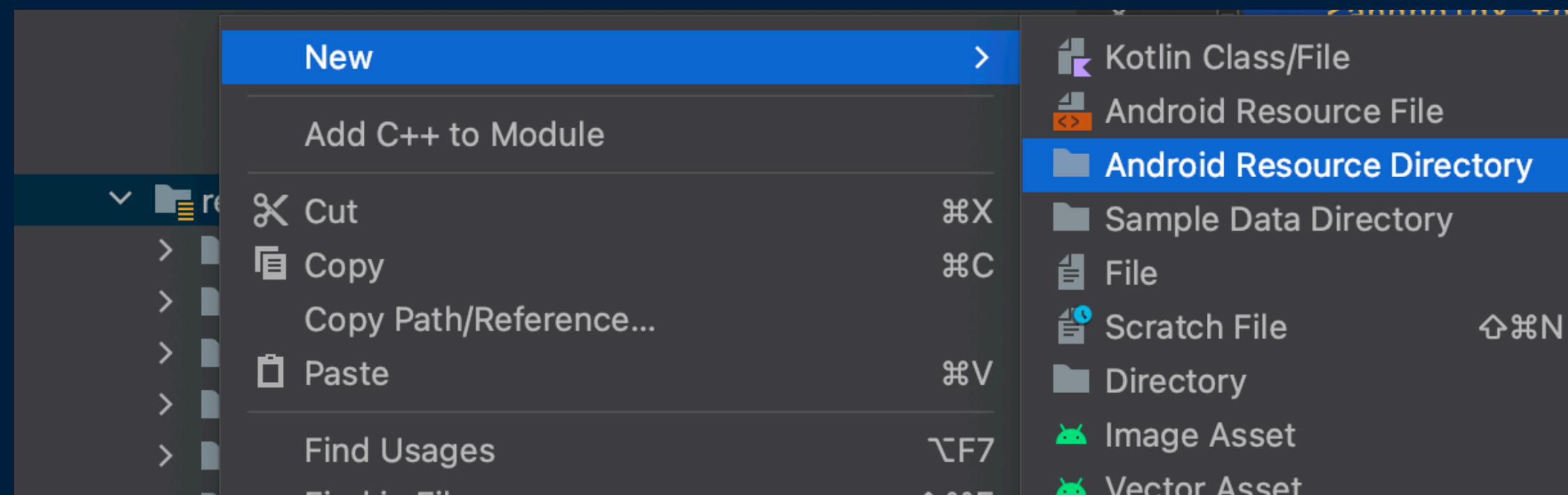
# Jetpack Navigation (1)

<https://developer.android.com/guide/navigation?hl=ko>

## 1. 의존성 추가

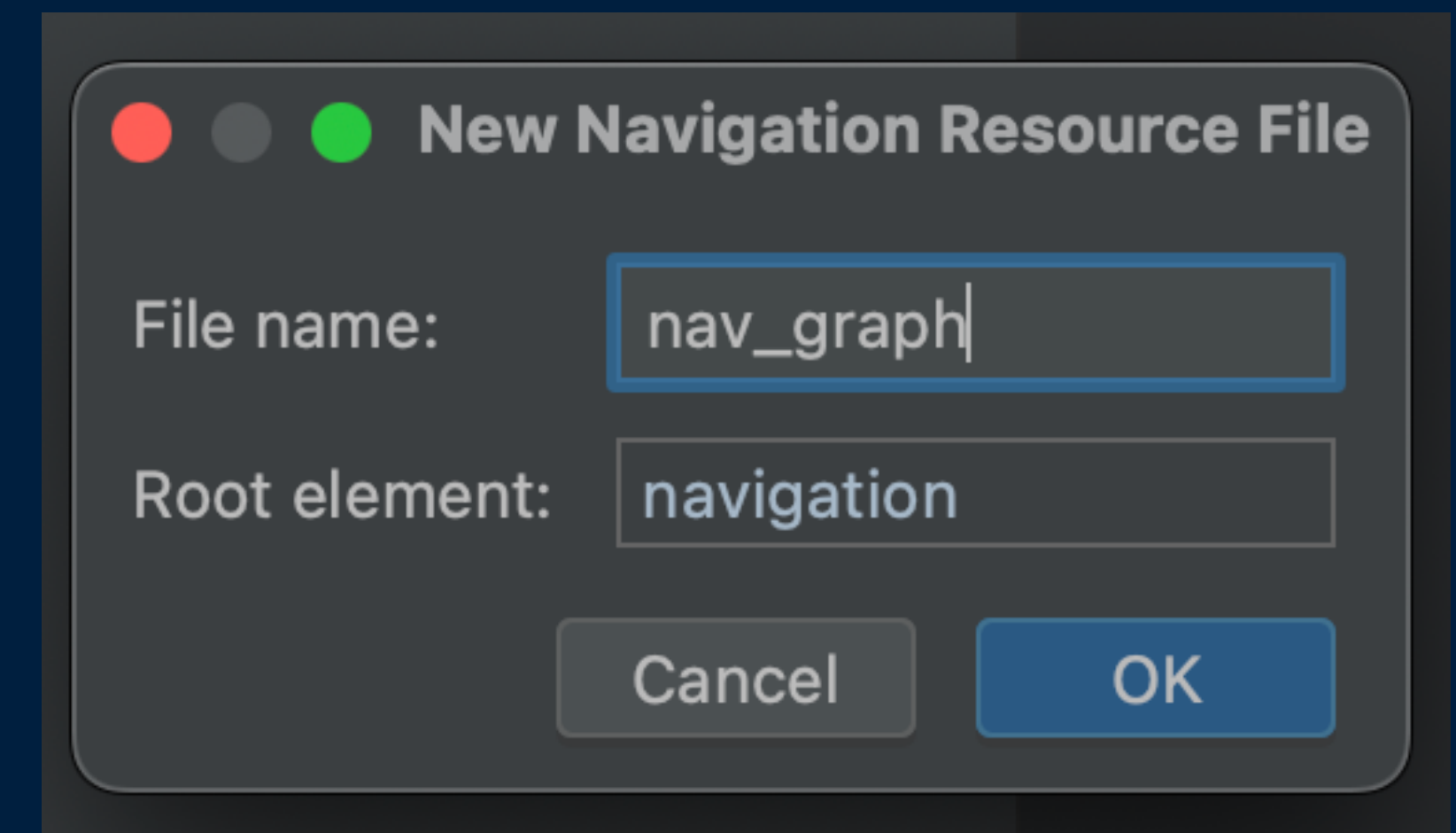
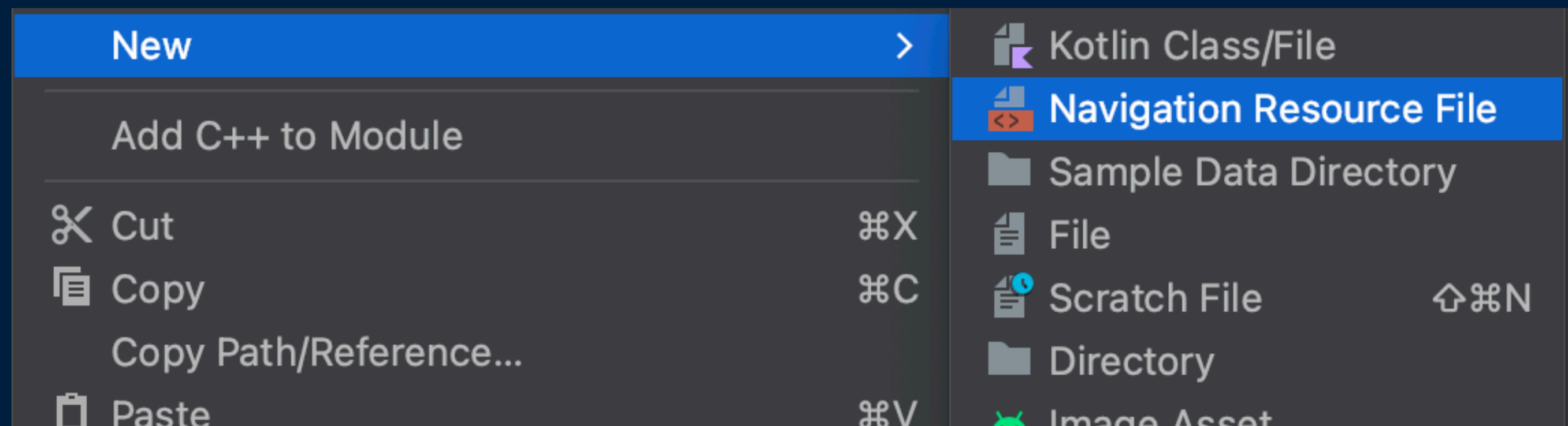
```
implementation ("androidx.navigation:navigation-fragment-ktx:2.7.4")  
implementation ("androidx.navigation:navigation-ui-ktx:2.7.4")
```

## 2. res 폴더에서 새 directory 추가 : navigation



# Jetpack Navigation (2)

## 3. res/navigation 폴더에 nav\_graph 생성



# Jetpack Navigation (3)


## 4. activity\_main.xml 에 NavHost 추가

```
<androidx.fragment.app.FragmentContainerView
    android:id="@+id/nav_host_fragment"
    android:name="androidx.navigation.fragment.NavHostFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintBottom_toBottomOf="parent"
    app:defaultNavHost="true"
    app:navGraph="@navigation/nav_graph" />
```



# Jetpack Navigation (4)

## 5. NavGraph에서 Fragment 및 action 추가

- design 탭에서  버튼 클릭해서 추가해도 되고 (기본 뼈대 코드가 자동 작성)
- Fragment를 미리 만들어 놓으면 목록에서 선택 가능
- design 탭의 GUI를 이용해 쉽게 Fragment 간 action 정의 가능



# Jetpack Navigation (5)

## Activity에서 NavHost에 접근 & navigation

```
val navHostFragment =  
    supportFragmentManager.findFragmentById(R.id.nav_host_fragment) as NavHostFragment  
val navController = navHostFragment.navController  
navController.navigate(R.id.action_greenFragment_to_redFragment)
```

## Fragment에서 NavHost에 접근 & navigation

```
findNavController().navigate(R.id.action_redFragment_to_blueFragment)
```

# Jetpack Navigation (6)

그 외의 다양한 기능들...

- 전환 시 인수 전달
- 전환 애니메이션
- 중첩 그래프
- 백 스택 관리
- 딥 링크 관리

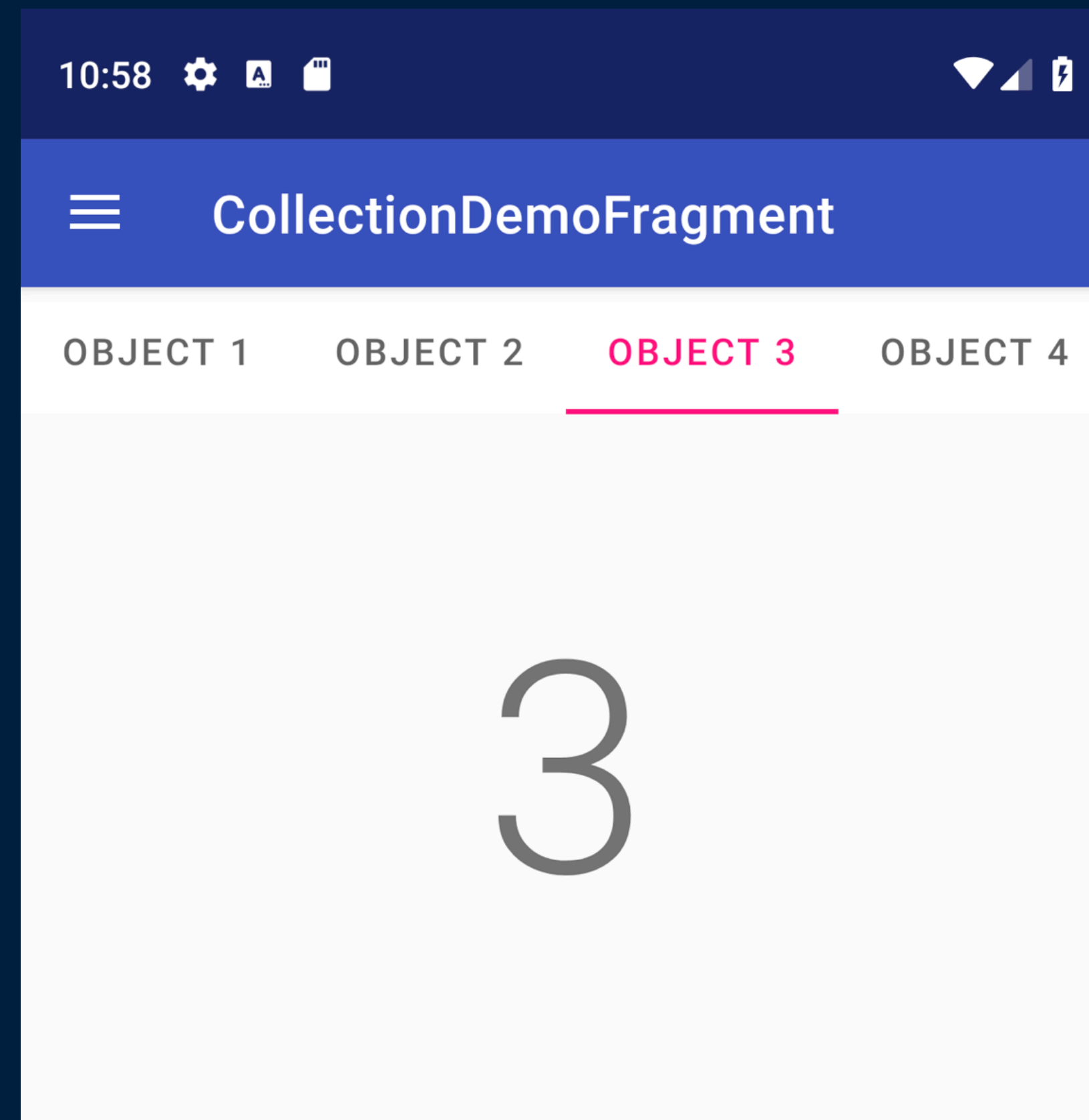
전부 <https://developer.android.com/guide/fragments/create?hl=ko> 에 있다!

# 그 외의 Fragment 활용

[ViewPager2]

<https://developer.android.com/guide/navigation/navigation-swipe-view-2?hl=ko>

간단하고 쉽게 탭 UI를 만들 수 있다.



# 과제 공지

## 과제 4: (유사) 넷플릭스 앱 만들기

- 과제 테마 : Network Error Handling, SharedPreferences, Fragment (Jetpack Navigation)
- 과제 4 기한 : 11월 9일 목 자정 (목~금 넘어가는 밤)
- 과제 링크 : <https://github.com/wafflestudio/seminar-2023-android-assignment/blob/assignment4/assignment-4/README.md>