02

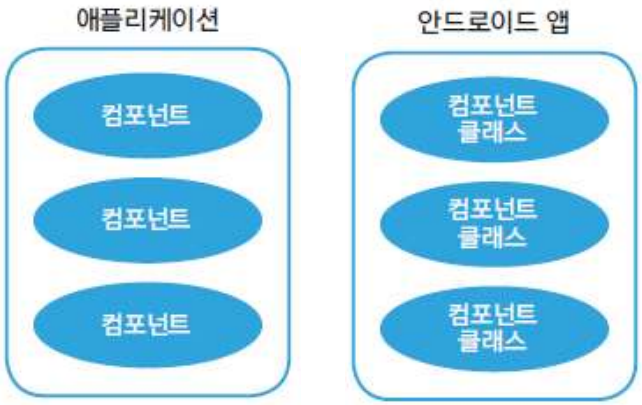
01-5. 앱 개발 특징

Android Overview

안드로이드 앱 개발의 특징

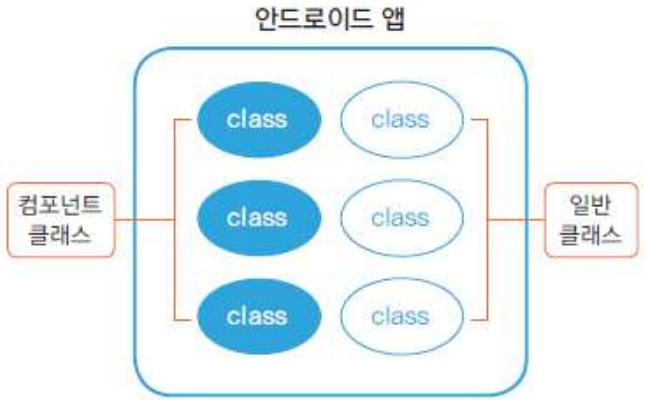
컴포넌트를 기반으로 한 개발

- 컴포넌트는 애플리케이션의 구성 요소다
  - 컴포넌트를 한마디로 정의하면 애플리케이션의 구성 요소라고 할 수 있습니다.
  - 안드로이드에서는 클래스로 컴포넌트를 개발합니다.



안드로이드 앱 개발의 특징

- 안드로이드 앱을 구성하는 클래스는 모두 컴포넌트인가?
  - 앱은 여러 클래스로 구성되는데 크게 컴포넌트 클래스와 일반 클래스로 구분합니다.
  - 클래스의 객체 생성부터 소멸까지 생명주기 관리를 개발자 코드에서 한다면 일반 클래스입니다
  - 생명주기를 안드로이드 시스템에서 관리한다면 컴포넌트 클래스입니다.



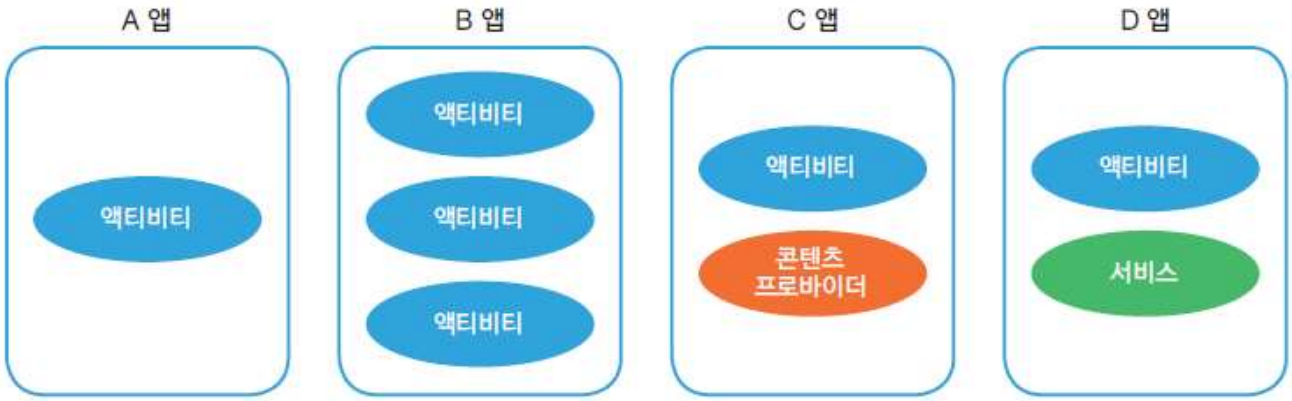
안드로이드 앱 개발의 특징

- 안드로이드 컴포넌트는 4종류다
  - 액티비티 화면을 구성하는 컴포넌트입니다.
  - 서비스 백그라운드 작업을 하는 컴포넌트입니다.
  - 콘텐츠 프로바이더 앱의 데이터를 공유하는 컴포넌트입니다.
  - 브로드캐스트 리시버 시스템 이벤트가 발생할 때 실행되게 하는 컴포넌트입니다.

- 4가지 컴포넌트를 어떻게 구분하는가?
  - 개발자가 컴포넌트 클래스를 만들 때는 지정된 클래스를 상속받아야 하는데 이 상위 클래스를 보고 구분할 수 있습니다.
  - 액티비티는 Activity, 서비스는 Service, 콘텐츠 프로바이더는 ContentProvider, 브로드캐스트 리시버는 BroadcastReceiver 클래스를 상속받아서 만듭니다.

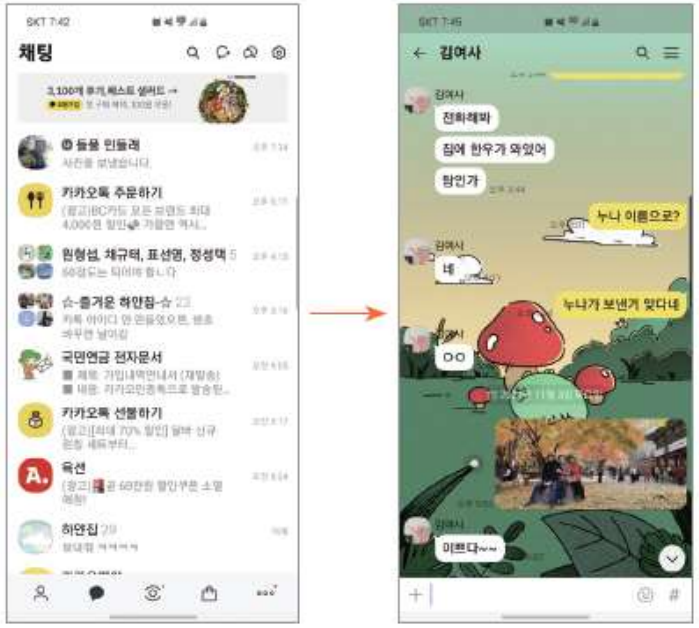
안드로이드 앱 개발의 특징

- 앱을 개발할 때 컴포넌트를 어떻게 구성해야 하는가?
  - 앱의 기능과 화면 등을 고려해 필요한 만큼 구성

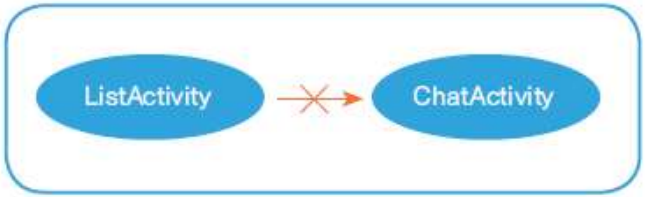


안드로이드 앱 개발의 특징

- 컴포넌트는 앱 안에서 독립된 실행 단위다
  - 컴포넌트끼리 서로 종속되지 않아서 코드 결합이 발생하지 않는다는 의미입니다.



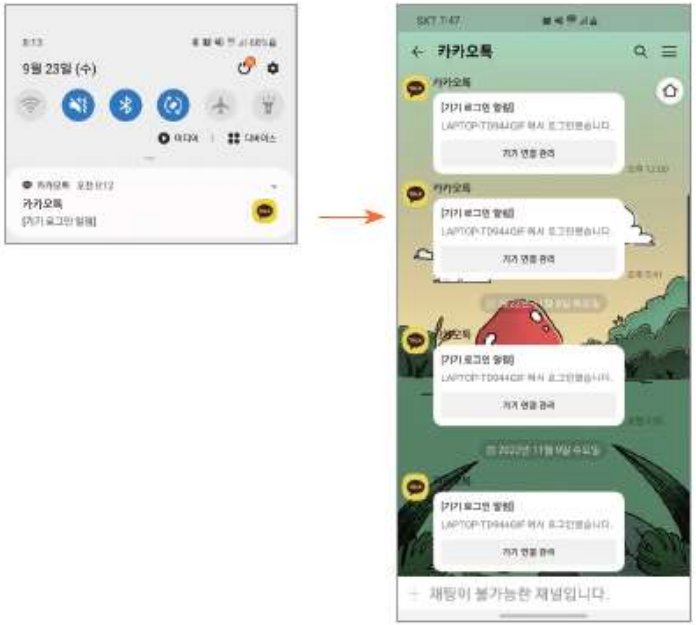
ListActivity ChatActivity



안드로이드 앱 개발의 특징

- 앱 실행 시점이 다양하다
  - 컴포넌트가 앱 내에서 독립해서 실행되는 특징 덕분에 앱의 실행 시점이 다양할 수 있습니다.
  - 안드로이드 앱에는 메인 함수main function 개념이 없다고 말합니다.

ListActivity



ChatActivity

안드로이드 앱 개발의 특징

- 애플리케이션 라이브러리를 사용할 수 있다
  - 다른 애플리케이션을 라이브러리처럼 이용하는 것을 말합니다.

카카오톡 앱 카메라 앱

안드로이드 앱 개발의 특징

리소스를 활용한 개발

- 리소스란 코드에서 정적인 값을 분리한 것입니다.
- 문자열 이외에 색상, 크기, 레이아웃, 이미지, 메뉴 등 많은 요소를 리소스로 활용할 수 있습니다.
- 이미지 등 몇몇을 제외하면 대부분 리소스는 XML 파일로 작성합니다.

• 문자열을 리소스로 등록하기

```
<string name="mytxt">
    동해 물과 백두산이 마르고 닳도록
    하느님이 보우하사 우리나라 만세
    무궁화 삼천리 화려 강산
    대한 사람, 대한으로 길이 보전하세
</string>
```

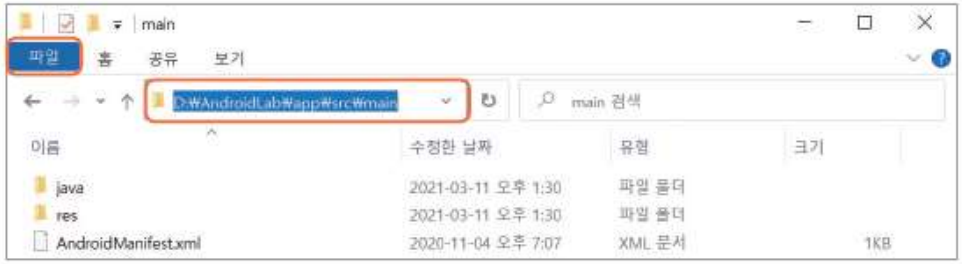
• 문자열 리소스 사용 예

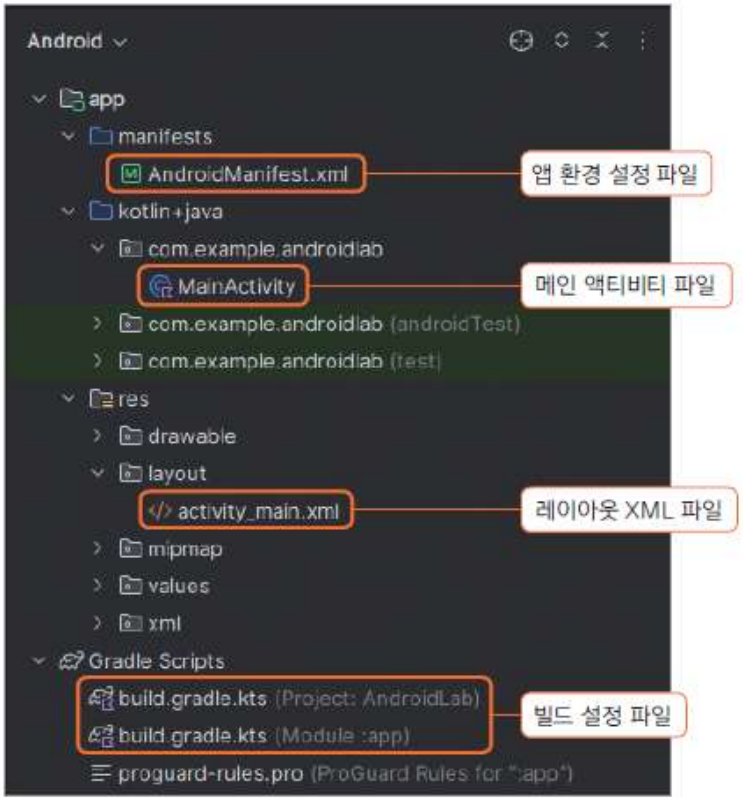
```
textView.text = resources.getString(R.string.mytxt)
```

앱 구성 파일 분석

프로젝트의 폴더 구성 알아보기

- 프로젝트 폴더에서 [모듈명 → src → main]





앱 구성 파일 분석

모듈의 폴더 구성 알아보기

이름	설명
build.gradle.kts	빌드 설정 파일
AndroidManifest.xml	앱의 메인 환경 파일
res	리소스 폴더
activity_main.xml	레이아웃 XML 파일
MainActivity.kt	메인 액티비티 파일

앱 구성 파일 분석

- 그래들 빌드 설정 파일
  - 그래들은 안드로이드 앱의 빌드 도구입니다.
  - 그래들의 설정 파일이 바로 build.gradle.kts
  - 프로젝트 수준의 build.gradle.kts (Project: AndroidLab)과 모듈 수준의 build.gradle.kts (Module: AndroidLab.app)

• 플러그인 선언

```
plugins {
    alias(libs.plugins.android.application)
    alias(libs.plugins.kotlin.android)
}
```

• 앱의 식별자 설정

```
applicationId = "com.example.androidlab"
```

앱 구성 파일 분석

- 그래들 빌드 설정 파일
  - targetSdk는 개발할 때 적용되는 SDK 버전입니다.
  - minSdk는 이 앱을 설치할 수 있는 기기의 최소 SDK 버전을 의미합니다.

• SDK 버전 설정

```
minSdk = 24
targetSdk = 35
```

• 앱의 버전 설정

```
versionCode = 1
versionName = "1.0"
```

앱 구성 파일 분석

- targetSdk에 명시한 안드로이드 SDK는 기본으로 적용되지만 그 외에 개발자가 추가하는 오픈소스 라이브러리나 구글의 androidx 라이브러리 등 SDK 라이브러리가 아닌 것들은 모두 dependencies에 선언해야 합니다.

• 라이브러리 설정

```
dependencies {
    implementation(libs.androidx.core.ktx)
    implementation(libs.androidx.appcompat)
    implementation(libs.material)
    implementation(libs.androidx.activity)
    implementation(libs.androidx.constraintlayout)
    testImplementation(libs.junit)
    androidTestImplementation(libs.androidx.junit)
    androidTestImplementation(libs.androidx.espresso.core)
}
```

라이브러리 버전은 시기에 따라 책과 다를 수 있어요.

앱 구성 파일 분석

- 메인 환경 파일
  - AndroidManifest.xml은 안드로이드 앱의 메인 환경 파일입니다
  - URL이 http://schemas.android.com/apk/res/android로 선언되었다면 안드로이드 표준 네임스페이스입니다.

• 네임스페이스 선언

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
```

앱 구성 파일 분석

- <application> 태그는 앱 전체를 대상으로 하는 설정
- icon 속성이 있는데 이곳에 지정한 이미지가 앱을 설치한 사용자의 폰에 보이는 실행 아이콘
- label 속성에는 앱의 이름을 등록
- theme 설정은 앱에 적용해야 하는 테마를 설정

• 네임스페이스 선언

```
<application
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.AndroidLab"
    tools:targetApi="31">
    (... 생략 ...)
</application>
```

앱 구성 파일 분석

- 액티비티는 <activity> 태그로, 서비스는 <service> 태그로, 브로드캐스트 리시버는 <receiver> 태그로, 그리고 콘텐츠 프로바이더는 <provider> 태그로 등록
- name 속성에는 클래스 이름을 등록
- <intent-filter>가 선언되었고 그 안에 <action> 태그의 name값이 android.intent.action.MAIN 문자열로, <category> 태그의 name값이 android.intent.category.LAUNCHER로 선언되면 이 액티비티는 앱 아이콘을 클릭했을 때 실행되는 액티비티라는 의미

• 액티비티 선언

```
<activity
    android:name=".MainActivity"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

앱 구성 파일 분석

- 리소스 폴더
  - drawable: 이미지 리소스
  - layout: UI 구성에 필요한 XML 리소스
  - mipmap: 앱 아이콘 이미지
  - values: 문자열 등의 값으로 이용되는 리소스

  - 리소스를 식별하기 위한 int형 변수가 R.java 파일에 등록
  - res/layout/test.xml 파일이라면 R.layout.test라고 이용

  - res 하위의 폴더명은 지정된 폴더명을 사용해야 합니다.
  - 각 리소스 폴더에 다시 하위 폴더를 정의할 수는 없습니다.
  - 리소스 파일명은 자바의 이름 규칙을 위배할 수 없습니다.
  - 리소스 파일명에는 알파벳 대문자를 이용할 수 없습니다.

앱 구성 파일 분석

- 레이아웃 XML 파일
  - 화면을 구성하는 레이아웃 XML 파일

• 레이아웃 XML 파일

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.an-
droid.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

앱 구성 파일 분석

- 메인 액티비티 파일
  - setContentView() 함수는 매개변수에 지정한 내용을 액티비티 화면에 출력합니다.
  - R.layout.activity_main으로 지정했으므로 res/layout/activity_main.xml 파일에 구성한 내용을 화면에 출력합니다.
  - enableEdgeToEdge()는 액티비티 화면이 상단의 배터리 표시가 있는 영역(이를 Status Bar라고 합니다.)과 하단의 안드로이드 버튼이 있는 영역(이를 Navigation Bar라고 합니다.)까지 나오게 하는 설정입니다
  - setOnApplyWindowInsetsListener() 부분은 액티비티에 출력되는 내용이 Navigation Bar 등과 겹치지 않게 하기 위한 설정입니다.

• 메인 액티비티 파일

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
            val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.
bottom)
            insets
        }
    }
}
```

감사합니다

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

윌리엄 셰익스피어
William Shakespeare