

1주차 개발일지

자료 조사

manifest file



manifest 파일이 무엇이며 안드로이드 앱에서 어떤 역할을 하며 그 안의 여러 tag가 가지는 의미에 대해 조사한다.

[App Manifest Overview | Android Developers](#)

Every app project must have an `AndroidManifest.xml` file (with precisely that name) at the root of the project source set. The manifest file describes essential information about your app to the Android build tools, the Android

 <https://developer.android.com/guide/topics/manifest/manifest-intro#perm>

Developers 

manifest 파일은 앱의 중요한 정보를 한 눈에 정리하고 있는 파일로 한 마디로 “summary of a app”라고 할 수 있다.

manifest 파일에 요약되어 있는 정보는 크게 4개이다.

1. 앱이 구동하는 하드웨어 소프트웨어 구동 정보 (api level)

앱은 모든 하드웨어와 os 버전에서 구현되지 않는다. 그래서 어떤 버전에서 정상 작동하는지 명시해야 한다.

2. 앱을 구성하고 있는 component의 종류

4대 component(activities, services, broadcast receivers, and content providers) 중 어떤 것인 든 앱에서 사용하려고 하면 반드시 manifest 파일에 어떤 component를 사용하고 있는지 명시해야 한다. 그리고 그 component들의 이름과 어떤 kotlin파일에 구현되어 있는지 적어야 한다.

3. 어떤 종류의 permission(허가)를 앱이 허용하고 있는지에 대한 정보

앱은 설치되는 폰의 모든 구역을 자유롭게 드나들 수 없다. 그래서 중요한 유저 정보에 대해 접근하기 위해서는 permission을 manifest 파일에 명시해야 한다.

4. 앱이 구동 중 사용하는 하드웨어와 소프트웨어에 대한 정보

카메라나 블루투스, 멀티 터치스크린과 연결해야 할 때 그들과 소통한다고 명시해야 한다.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="org.texchtown.kotlin_demo">
4
5      <application
6          android:allowBackup="true"
7          android:icon="@mipmap/ic_launcher"
8          android:label="Kotlin_demo"
9          android:roundIcon="@mipmap/ic_launcher_round"
10         android:supportsRtl="true"
11         android:theme="@style/Theme.Kotlin_demo">
12             <activity
13                 android:name=".MainActivity"
14                 android:exported="true">
15                 <intent-filter>
16                     <action android:name="android.intent.action.MAIN" />
17
18                     <category android:name="android.intent.category.LAUNCHER" />
19                 </intent-filter>
20             </activity>
21         </application>
22
23     </manifest>
```

▼ application 정보

| Android Developers

The declaration of the application. This element contains subelements that declare each of the application's components and has attributes that can affect all the components. Many of these attributes

 <https://developer.android.com/guide/topics/manifest/application-element>



```
<application
    android:allowBackup="true"           백업을 허용하는가
    android:icon="@mipmap/ic_launcher"   앱 아이콘 이미지 저장 위치
    android:label="@string/app_name"    표시되는 앱 이름
    android:roundIcon="@mipmap/ic_launcher_round"
                                         동그란 앱 아이콘 이미지 저장 위치
                                         일부 하드웨어에서만 동글 아이콘 사용
    android:supportsRtl="true"          right-to-left layout을 지원하는가
    android:theme="@style/Theme.Kotlin_demo">
                                         앱 스타일
<activity></activity>
<service></service>
<receiver></receiver>
<provider></provider>
```

```
</application>
```

<application> 태그 안에 작성되는 application에 대한 전체 정보로 앱의 아이콘이나 앱의 이름, 스타일에 대한 정보가 들어갈 수 있다.

<application> 태그 안에는 하위 태그로 여러 속성이 들어갈 수 있는데 그 종류는 다음과 같다.

1. <activity>, <activity-alias>, <service>, <receiver>, <provider>

앱 내에서 사용하는 component에 대한 정보

2. <meta-data>

component 사이 간단한 데이터를 전달할 때 사용되는 key-value 쌍에 대한 정보.

component 사이 데이터가 전달될 때 모든 meta-data 값들이 Bundle 객체에 수집되어 component가 생성 시 PackageManagerInfo.metaData 필드로 제공됨

3. <profileable>

profiler가 앱에 어떻게 접근할지 자세히 명시해준다.

고 하는데~ 잘 모르겠음 ㅋㅋㅋ profiler?

4. <uses-library>, <uses-native-library>

어플과 연결되어 있는 library에 대한 정보를 명시해준다

고 하는데~ 잘 모르겠음 ㅋㅋㅋ library?

▼ component 선언

```
<application>
    ...
    <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>
        <meta-data></meta-data>
        <layout></layout>
    </activity>

    <service>
        <intent-filter></intent-filter>
        <meta-data></meta-data>
    </service>

    <receiver>
```

```

<intent-filter></intent-filter>
<meta-data></meta-data>
</receiver>

<provider>
<intent-filter></intent-filter>
<meta-data></meta-data>
<grant-uri-permission></grant-uri-permission>
<path-permission></path-permission>
</provider>

</application>

```

4대 component에는 activity, service, receiver, provider가 있다. 이들을 이용하기 위해서는 반드시 manifest 파일에서 선언을 해줘야 한다. component의 종류와 역할에 대해서는 아래에서 더 자세히 설명을 하겠다.

사용하는 component를 manifest 파일에 선언 없이는 소스 코드가 있어도 앱에서 이들을 사용할 수가 없다.

그리고 각 component에는 고유의 이름을 android:name 속성을 이용해 선언해주어야 한다.

▼ permission 선언

Permissions on Android | Android Developers

App permissions help support user privacy by protecting access to the following: Restricted data, such as system state and users' contact information Restricted actions, such as connecting to a paired device

 <https://developer.android.com/guide/topics/permissions/overview>



1. 해당 앱이 다른 앱의 정보를 가져오거나 사진 파일을 가져오는 등 중요한 사용자 데이터에 접근을 하기 위해서는 허가가 필요하다.
2. 또한 특정 하드웨어(카메라, 인터넷, 마이크)에 접근하기 위해서도 허가가 필요하다.

여러가지 permission이 존재하고 각 허가를 위한 고유의 label이 존재한다. 예를 들어 해당 앱이 카메라에 접근을 하기 위해서는 해당 접근이 필요하다.

```

<manifest ...>
<uses-permission android:name="android.permission.CAMERA"/>
<application ...>
...
</application>
</manifest>

```

component



안드로이드 앱의 4대 component에 대해 조사하고 각각 어떤 역할을 수행하는지 조사한다

▼ activity

[Introduction to activities | Android Developers](#)

An Activity is an application component that provides a screen with which users can interact in order to do something, such as dial the phone, take a photo, send an email, or view a map. Each activity is given a window in which to draw its user interface...

 <https://developer.android.com/guide/components/activities/intro-activities>

▷ activity의 개념: activity는 앱의 진입 지점이 되는 요소로 홈에서 아이콘으로 앱에 진입하거나 다른 앱에서 해당 앱으로 진입할 때 activity를 통해 해당 앱으로 진입한다. activity는 화면에 보이는 부분을 그리는 역할을 한다.

▷ activity의 수명주기:

[LifeCycle](#)

▼ service

[Services overview | Android Developers](#)

A is an application component that can perform long-running operations in the background. It does not provide a user interface. Once started, a service might continue running for some time, even after the user



 <https://developer.android.com/guide/components/services>

▷ service 개념: service는 view는 존재하지 않지만 background에서 사용자와의 상호 작용 없이 계속 동작하는 component이다.

▷ service가 사용되는 곳

1. 음악재생

2. 파일 입출력, 압축, 다운로드 등등

▷ 종류

- background service: 앱이 run하고 있을 때만 동작하고 앱이 terminate되면 동작을 멈춤

- foreground service: 앱이 run하고 있지 않아도 계속 살아서 동작하는 service이다. foreground service는 반드시 notification을 띄워야 한다.
- bound service: service와 연결된 component가 살아있을 때만 동작하는 service이다. 이때 service는 다른 component와 bindService()로 연결되어야 한다.

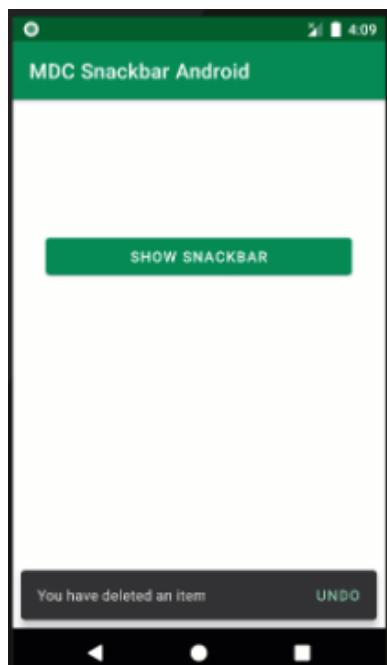
▷service 생명주기

1. startService(): 다른 component가 startService()를 통해 service를 호출하면 service는 생성된다.
2. stopSelf(), stopService(): service는 내부에서 stopSelf()를 스스로 호출하기 전까지 계속 background에서 존재한다. 또는 다른 component가 stopService()를 호출해서 종류할 수도 있다.
3. bindService(), unbindService(): service는 이 둘 메소드로 생성과 소멸이 가능하다.
(?????)

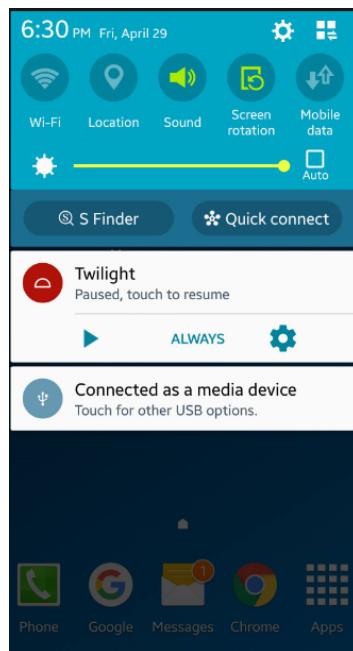
▷notification

service는 background에서 동작하고 있어서 사용자가 service가 동작 중인지 알게 하기 위해 notification을 띄워 동작중임을 알려야 한다.(background service의 경우)

- snack bar notification



- status bar notification



▼ receiver

Broadcasts overview | Android Developers

Android apps can send or receive broadcast messages from the Android system and other Android apps, similar to the publish-subscribe design pattern. These broadcasts are sent when an event of interest occurs.

 <https://developer.android.com/guide/components/broadcasts>



▷ receiver의 개념: 안드로이드 앱은 broadcast message를 안드로이드 시스템 뜯느 다른 앱들과 주고 받을 수 있다.

▷ broadcast의 개념: 어떤 중요 event나 일어날 때 특정 broadcast가 앱 전반에 걸쳐 발송된다. 예를 들어 폰이 켜지거나, 충전이 시작할 때이다. 또한 custom broadcast를 만들어서 발송을 할 수 있다. 그리고 broadcast message는 intent안에 들어가서 전달된다.

▷ standard broadcast action:

- ACTION_TIME_CHANGED: 현재 시간 변화
- ACTION_TIMEZONE_CHANGED: 시간 설정 완료
- ACTION_BOOT_COMPLETED: 부팅 완료
- ACTION_PACKAGE_ADDED: 새로운 앱 package 설치 완료
- ACTION_PACKAGE_CHANGED: 이미 존재하는 앱 package 내용 변화

- ACTION_PACKAGE_REMOVED: 존재하는 앱 package 삭제
- ACTION_PACKAGE_RESTARTED: package 재시작, 동작중이던 process 초기화
- ACTION_PACKAGE_DATA_CLEARED: 이미 존재하는 앱 package 안의 data 초기화
- ACTION_PACKAGES_SUSPENDED: package 사용중지
- ACTION_PACKAGES_UNSUSPENDED: package 재사용
- ACTION_UID_REMOVED: 시스템에서 UID(??)제거
- ACTION_BATTERY_CHANGED: 배터리 교체, 이때 배터리 관한 정보 변화
- ACTION_POWER_CONNECTED: 충전중
- ACTION_POWER_DISCONNECTED: 충전 중지

▷ receiver의 생명주기: broadcast receiver는 activity의 생명주기를 이용하기 때문에 적절한 메서드를 추가해서 원하는 동작을 override 해주면된다.

▼ provider

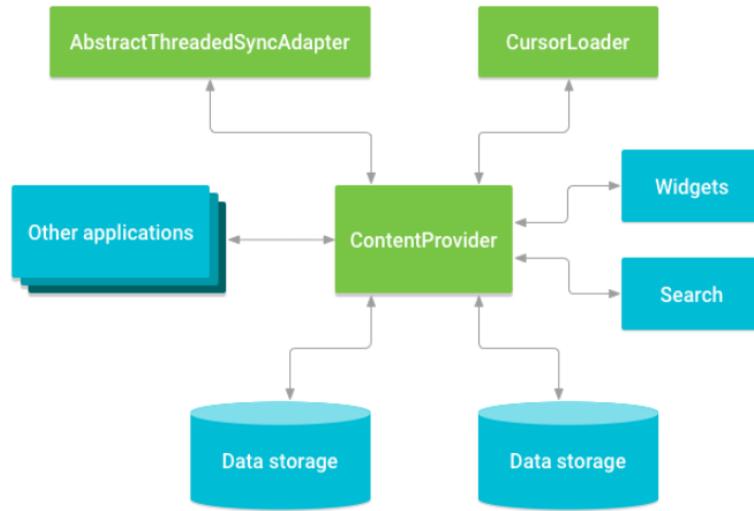
Content providers | Android Developers

Content providers can help an application manage access to data stored by itself, stored by other apps, and provide a way to share data with other apps. They encapsulate the data, and provide mechanisms for

 <https://developer.android.com/guide/topics/providers/content-providers>



▷ provider의 개념: content provider는 다른 앱이 저장한 데이터에 대한 액세스 권한을 관리하도록 돋고 다른 앱과 데이터를 공유할 방법을 제공합니다. 데이터를 캡슐화하고, 데이터 보안을 도와 다른 앱의 데이터에 안전하게 접근하고 수정할 수 있도록 도와줍니다.



▷content provider가 사용되는 곳

- 다른 애플리케이션과 내 애플리케이션 데이터에 대한 액세스 공유
- 위젯에 데이터 전송
- SearchRecentSuggestionsProvider를 사용하여 검색 프레임워크를 통해 애플리케이션의 맞춤 추천 검색어 반환
- AbstractThreadedSyncAdapter를 구현하여 서버와 애플리케이션 데이터 동기화
- CursorLoader를 사용하여 UI에서 데이터 로드

Android 프레임워크에는 오디오, 동영상, 이미지 및 개인 연락처 정보 등의 데이터를 관리하는 콘텐츠 제공자가 포함되어 있습니다.

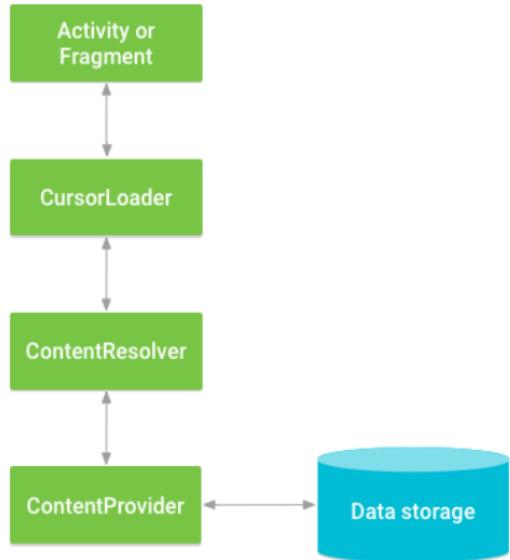
- calender provider: 달력 데이터에 접근
- contacts provider: 연락처 데이터에 접근

▷provider 사용법

일반적으로 content provider는 두 가지 경우에 사용된다.

1. 다른 앱의 provider에게 데이터를 제공하기 위한 코드를 구현하거나
2. 다른 앱으로부터 데이터를 제공받기 위해 내 앱에 content provider를 구현하는 것이다.

콘텐츠 제공자 내의 데이터에 액세스하고자 하는 경우, 애플리케이션의 Context에 있는 ContentResolver 객체를 사용하여 클라이언트로서 제공자와 통신을 주고받으면 됩니다. ContentResolver 객체가 제공자 객체와 통신하며, 이 객체는 ContentProvider를 구현하는 클래스의 인스턴스입니다.



intent



intent의 역할과 어떤 상황에서 intent가 쓰이는지 조사한다

인텐트 및 인텐트 필터 | Android 개발자 | Android Developers

An Intent is a messaging object you can use to request an action from another app component . Although intents facilitate communication between components in several ways, there are three fundamental use cases: An Activity represents a single screen in...

<https://developer.android.com/guide/components/intents-filters?hl=ko>

▷Intent이란

Intent는 메시징 객체로, 다른 앱 구성 요소로부터 작업을 요청하는 데 사용할 수 있습니다. 인텐트가 구성 요소 사이의 통신을 촉진하는 데는 여러 가지 방식이 있지만 기본적인 사용 사례는 크게 세 가지로 나눌 수 있습니다.

1. 액티비티 시작

Activity는 앱 안의 단일 화면을 나타냅니다. Activity의 새 인스턴스를 시작하려면 Intent를 startActivity()로 전달하면 됩니다. Intent는 시작할 액티비티를 설명하고 모든 필수 데이터를 담습니다.

2. 서비스 시작

Service는 사용자 인터페이스 없이 백그라운드에서 작업을 수행하는 구성 요소입니다. 서비스를 시작하여 일회성 작업을 수행하도록 하려면(예: 파일 다운로드) Intent를 startService()에 전달하면 됩니다. Intent는 시작할 서비스를 설명하고 모든 필수 데이터를 담고 있습니다.

3. 브로드캐스트 전달

브로드캐스트는 모든 앱이 수신할 수 있는 메시지입니다. 시스템은 시스템이 부팅될 때 또는 기기가 충전을 시작할 때 등 시스템 이벤트에 대한 다양한 브로드캐스트를 전달합니다. Intent를 sendBroadcast() 또는 sendOrderedBroadcast()에 전달하면 다른 앱에 브로드캐스트를 전달할 수 있습니다.

▷ Intent의 종류

- Explicit Intent(명시적)

Intent를 받을 명확한 대상의 이름이 정해져 있는 곳에 쓴다. 명시적 인텐트는 일반적으로 앱 안에서 구성 요소를 시작할 때 쓴다. 이는 구성 요소를 시작할 때 activity 또는 service의 클래스 이름을 알고 있기 때문이다.

- Implicit Intent(암시적)

Intent를 받을 명확한 대상의 이름이 정해져 있지 않은 곳에 쓴다. 대상은 정해져 있지 않지만 intent를 받는 대상이 수행할 action을 정해준다. 예를 들어 사용자의 위치를 지도에 보여주고 싶으면 implicit intent를 이용해 지도에 위치를 보여줄 수 있는 앱에 상용자 위치를 보여달라고 요청할 수 있다.

▷ Intent의 구조

Android system은 Intent 객체 안에 있는 정보를 이용해 어떤 component를 시작할지 결정하거나 intent를 받는 대상에게 올바른 action을 지시할 수 있다. Intent가 가지고 있는 정보는 다음과 같다.

- Component name: 시작하고자 하는 component의 이름
- Action: 어떤 action을 지시하고 싶은지

▼ Standard Activity Actions

- `ACTION_MAIN`
- `ACTION_VIEW`
- `ACTION_ATTACH_DATA`
- `ACTION_EDIT`
- `ACTION_PICK`

- `ACTION_CHOOSER`
- `ACTION_GET_CONTENT`
- `ACTION_DIAL`
- `ACTION_CALL`
- `ACTION_SEND`
- `ACTION_SENDTO`
- `ACTION_ANSWER`
- `ACTION_INSERT`
- `ACTION_DELETE`
- `ACTION_RUN`
- `ACTION_SYNC`
- `ACTION_PICK_ACTIVITY`
- `ACTION_SEARCH`
- `ACTION_WEB_SEARCH`
- `ACTION_FACTORY_TEST`

▼ Standard Broadcast Actions

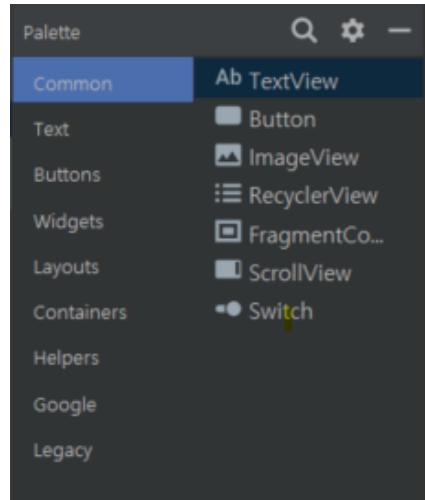
- `ACTION_TIME_TICK`
- `ACTION_TIME_CHANGED`
- `ACTION_TIMEZONE_CHANGED`
- `ACTION_BOOT_COMPLETED`
- `ACTION_PACKAGE_ADDED`
- `ACTION_PACKAGE_CHANGED`
- `ACTION_PACKAGE_REMOVED`
- `ACTION_PACKAGE_RESTARTED`
- `ACTION_PACKAGE_DATA_CLEARED`
- `ACTION_PACKAGES_SUSPENDED`
- `ACTION_PACKAGES_UNSUSPENDED`
- `ACTION_UID_REMOVED`

- ACTION_BATTERY_CHANGED
 - ACTION_POWER_CONNECTED
 - ACTION_POWER_DISCONNECTED
 - ACTION_SHUTDOWN
- Data: action에 필요한 데이터를 담고 있다. 해당 데이터를 참조하는 Uri형태로 제공된다.
 - Category: 인텐트를 처리해야 하는 구성 요소의 종류에 관한 추가 정보를 담은 문자열입니다.
 - Extras: Data를 이용해 전달한 데이터 외에 추가적인 데이터를 key-value 형태로 제공한다. Bundle 객체 형태로 제공된다.
 - Flags: intent에 대한 추가 정보로 activity를 시작할 방법에 대한 지침을 줄 수도 있고 activity를 시작한 다음에 어떻게 처리해야 하는지도 알려줄 수 있다.

pallette



xml파일에 있는 pallette안에 있는 화면을 구성하는 여러 화면구성 요소들의 종류와 역할에 대해 조사한다



▼ Common

- TextView: 글자 형태의 view를 만드는 요소
- Button: 버튼 형태의 view를 만드는 요소
- ImageView: 이미지를 추가하기 위한 view

- RecyclerView:

RecyclerView

- FragmentContainerView: fragment를 담기 위한 공간을 할당하는 요소
- ScrollView: 스크롤 가능한 view를 만드는 요소
- Switch: 스위치 형태의 view를 만드는 요소

▼ Text

▼ Button

▼ Widgets

▼ Layouts

▼ Containers

▼ Helpers

▼ Google

▼ Legacy

layout

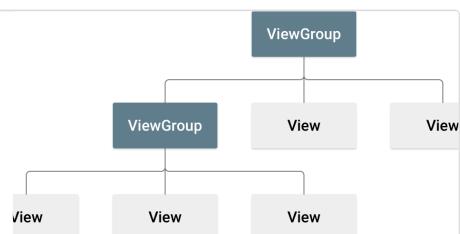


안드로이드 앱의 6대 layout에 대해 조사하고 각각의 특징과 공통점 차이점에 대해 조사한다

Layouts | Android Developers

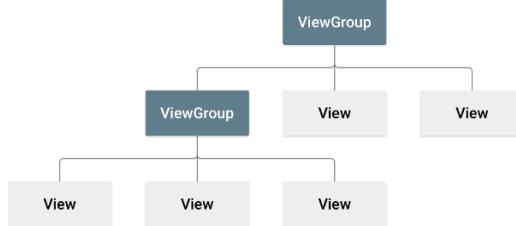
A layout defines the visual structure for a user interface, such as the UI for an activity or app widget . You can declare a layout in two ways: The Android framework gives you the flexibility to use either or both of these methods for

 <https://developer.android.com/develop/ui/views/layout/declaring-layout>



Layout은 앱에서 User Interface를 담당하는 부분에서 보여지는 요소들을 어떻게 배열할 지 구조를 담당하는 부분이다.

Layout의 모든 요소는 view와 viewgroup으로 이루어져 있다.



- View: 화면에서 보여지는 부분 중 사용자와 상호작용하는 부분으로 View 객체를 widget이라고 한다.
- ViewGroup: 여러개의 View와 ViewGroup을 묶는 보여지지 않는 묶음으로 ViewGroup 객체를 layout이라고 한다.

▼ Linear Layout

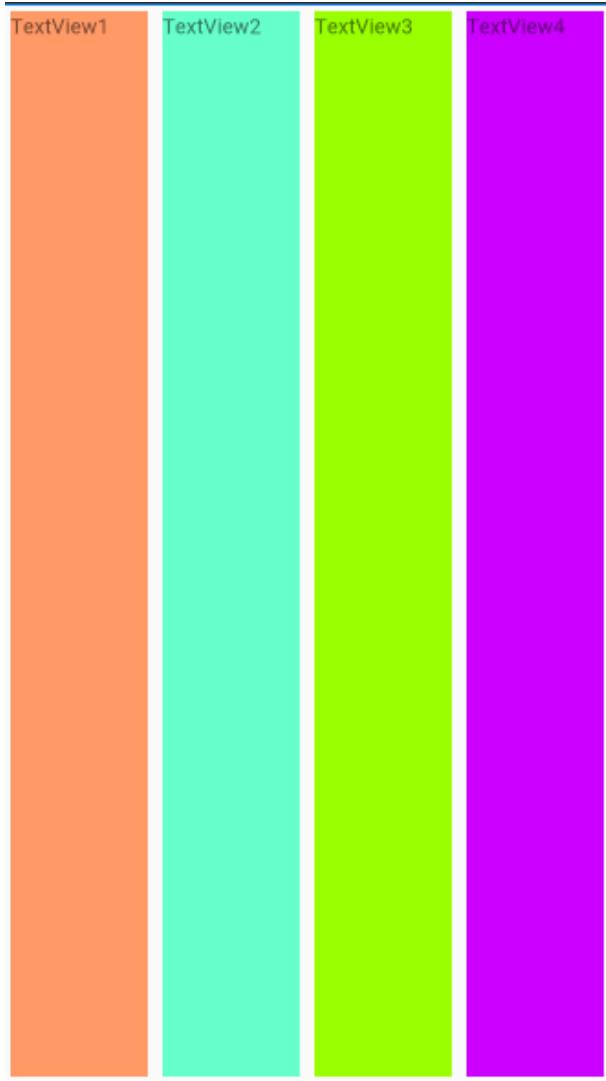
Linear Layout | Android Developers

LinearLayout is a view group that aligns all children in a single direction, vertically or horizontally. You can specify the layout direction with the LinearLayout are stacked one after the other, so a vertical list will only

 <https://developer.android.com/develop/ui/views/layout/linear>

▷ Linear layout 개념

Linear layout은 안의 요소들을 한 방향으로 정렬하는 ViewGroup이다.



▷View 배열 방법

- android:orientation 속성을 이용해 가로 세로 정렬 방향을 설정한다.
- android:weight 속성을 이용해 layout 안의 View들의 크기 비율을 설정한다.
- android:gravity 속성을 이용해 요소들이 상하좌우 어디로 치우칠지 정해준다.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:orientation="vertical" >
    안에 있는 요소들이 차례대로 위에서 아래로 공간 차지
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```

        android:hint="@string/to" />
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="@string/subject" />
<EditText
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:gravity="top"
    android:hint="@string/message" />
<Button
    android:layout_width="100dp"
    android:layout_height="wrap_content"
    android:layout_gravity="right"
    android:text="@string/send" />
</LinearLayout>
```

나머지 요소들의 크기를 제외하고 나머지 영역 모두 차지
자신한테 할당된 공간 내에서 위쪽 정렬

자신한테 할당된 공간 내에서 오른쪽 정렬

▼ RELATIVE

Relative Layout | Android Developers

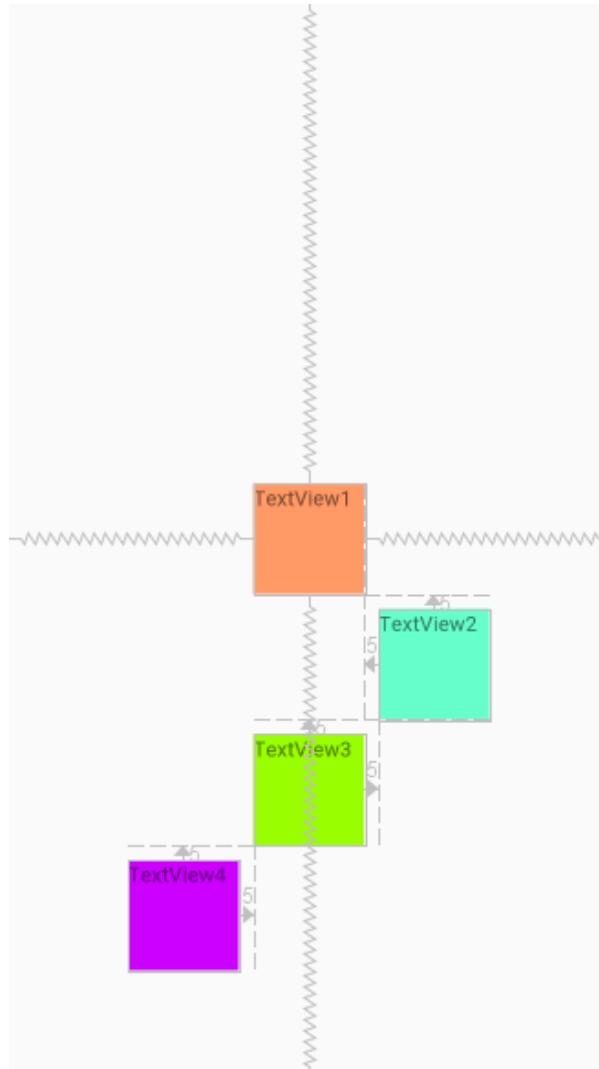
RelativeLayout is a view group that displays child views in relative positions. The position of each view can be specified as relative to sibling elements (such as to the left-of or below another view) or in

 <https://developer.android.com/develop/ui/views/layout/relative>

▷ Relative layout 개념

Relative layout은 안의 요소들의 위치가 서로의 상대적인 위치에 의해 결정되는 ViewGroup이다.

ViewGroup간의 nesting 없이 다양한 형태의 배치를 만들 수 있다는 장점이 있다.



▷View 배열 방법

View가 가지고 있는 고유의 id를 이용하거나 View간 관계(parent, child)를 이용해서 상대적 위치를 정할 수 있다.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    ...
    >
    <EditText
        android:id="@+id/name"
        ...
        />
    <Spinner
        ...
        android:id="@+id/dates"
        android:layout_alignParentLeft="true"          parent의 왼쪽에 요소 붙임
        android:layout_toLeftOf="@+id/times" />      spinner의 오른쪽을 id times의 왼쪽에 붙임
```

```

<Spinner
    ...weight, height
    android:id="@+id/times"
    android:layout_below="@+id/name"
    android:layout_alignParentRight="true" />
    id times 아래에 요소 붙임
    parent의 오른쪽에 요소 붙임
<Button
    ...weight, height
    android:layout_alignParentRight="true" />
    parent의 오른쪽에 요소 붙임
/>/RelativeLayout>

```

<u>android:layout_above</u>	Positions the bottom edge of this view above the given anchor view ID.
<u>android:layout_alignBaseline</u>	Positions the baseline of this view on the baseline of the given anchor view ID.
<u>android:layout_alignBottom</u>	Makes the bottom edge of this view match the bottom edge of the given anchor view ID.
<u>android:layout_alignEnd</u>	Makes the end edge of this view match the end edge of the given anchor view ID.
<u>android:layout_alignLeft</u>	Makes the left edge of this view match the left edge of the given anchor view ID.
<u>android:layout_alignParentBottom</u>	If true, makes the bottom edge of this view match the bottom edge of the parent.
<u>android:layout_alignParentEnd</u>	If true, makes the end edge of this view match the end edge of the parent.
<u>android:layout_alignParentLeft</u>	If true, makes the left edge of this view match the left edge of the parent.
<u>android:layout_alignParentRight</u>	If true, makes the right edge of this view match the right edge of the parent.
<u>android:layout_alignParentStart</u>	If true, makes the start edge of this view match the start edge of the parent.
<u>android:layout_alignParentTop</u>	If true, makes the top edge of this view match the top edge of the parent.
<u>android:layout_alignRight</u>	Makes the right edge of this view match the right edge of the given anchor view ID.
<u>android:layout_alignStart</u>	Makes the start edge of this view match the start edge of the given anchor view ID.
<u>android:layout_alignTop</u>	Makes the top edge of this view match the top edge of the given anchor view ID.
<u>android:layout_alignWithParentIfMissing</u>	If set to true, the parent will be used as the anchor when the anchor cannot be found for

	layout_toLeftOf, layout_toRightOf, etc.
<code>android:layout_below</code>	Positions the top edge of this view below the given anchor view ID.
<code>android:layout_centerHorizontal</code>	If true, centers this child horizontally within its parent.
<code>android:layout_centerInParent</code>	If true, centers this child horizontally and vertically within its parent.
<code>android:layout_centerVertical</code>	If true, centers this child vertically within its parent.
<code>android:layout_toEndOf</code>	Positions the start edge of this view to the end of the given anchor view ID.
<code>android:layout_toLeftOf</code>	Positions the right edge of this view to the left of the given anchor view ID.
<code>android:layout_toRightOf</code>	Positions the left edge of this view to the right of the given anchor view ID.
<code>android:layout_toStartOf</code>	Positions the end edge of this view to the start of the given anchor view ID.

▼ CONSTRAINT

Build a Responsive UI with ConstraintLayout | Android Developers

ConstraintLayout allows you to create large and complex layouts with a flat view hierarchy (no nested view groups). It's similar to RelativeLayout in that all views are laid out according to relationships between sibling views and

 <https://developer.android.com/develop/ui/views/layout/constraint-layout>

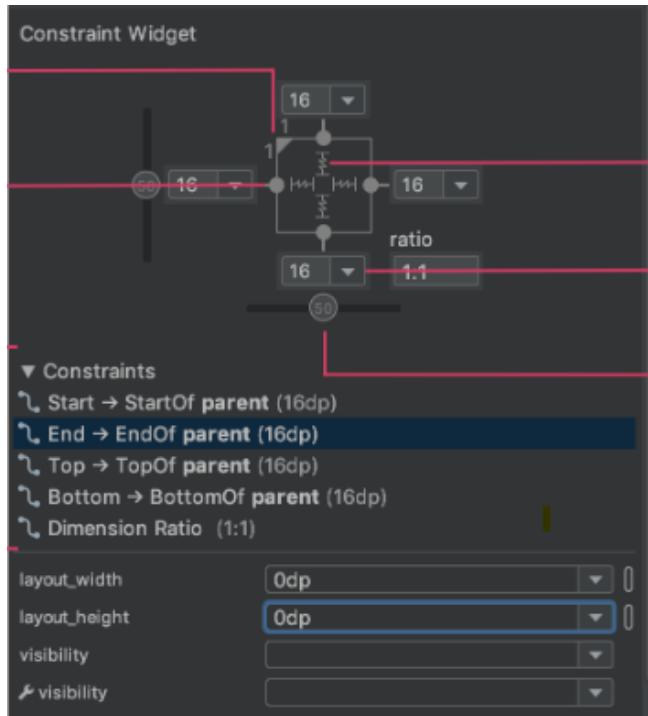
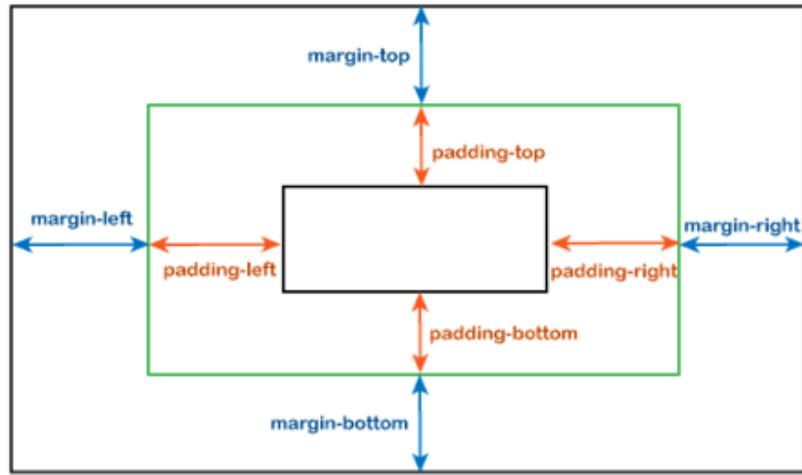
▷ Constraint layout 개념

Constraint layout은 안에 있는 요소들의 크기와 위치가 조건들에 의해서 결정되는 ViewGroup이다. 하나의 View의 크기와 위치가 결정되기 위해서는 최소한의 개수의 조건이 존재한다. 방법에 따라서는 크기와 위치가 한 두 가지 조건에 의해 결정될 수도 있다. 이런 방법을 통해 더 적은 코드로 UI를 구성할 수 있다.

▷ Constraint의 종류

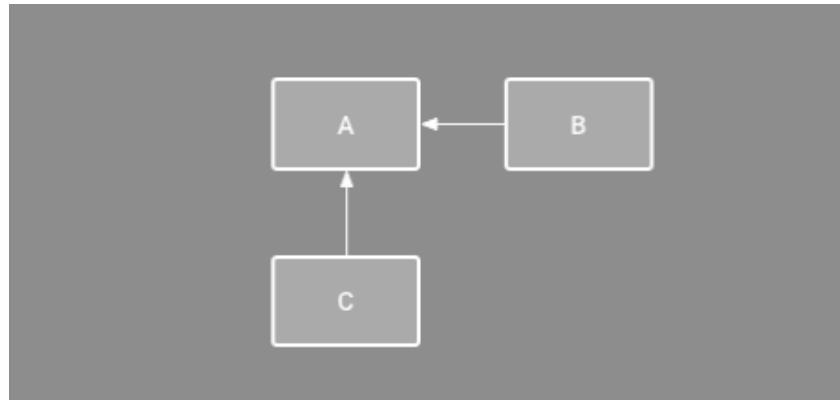
- View 크기 직접 조정하는 조건

View 주변의 margin과 View 안의 padding을 직접 조정해서 View 전체 크기와 상대 위치를 조정할 수 있다.

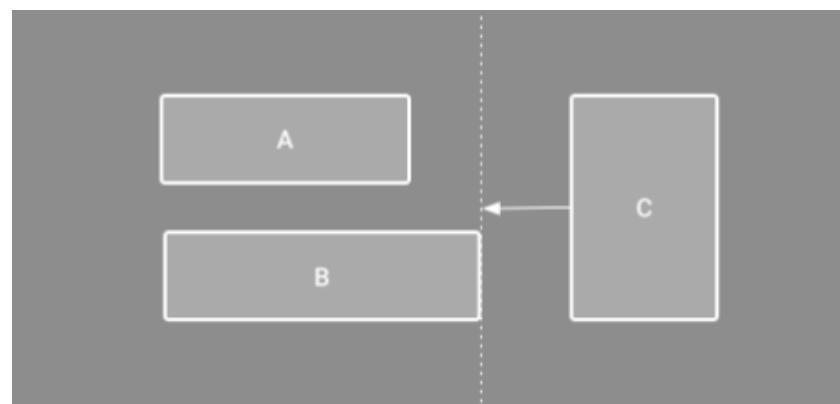


- View간 또는 baseline 과의 위치 조정

View와 View 사이 관계를 지정해서 View의 위치와 크기를 조정할 수 있다.



보이지 않는 baseline을 설정한 다음 그것과 View간의 위치와 거리를 조정해 View의 위치와 크기를 조정할 수 있다.



▼ TABLE

Table | Android Developers

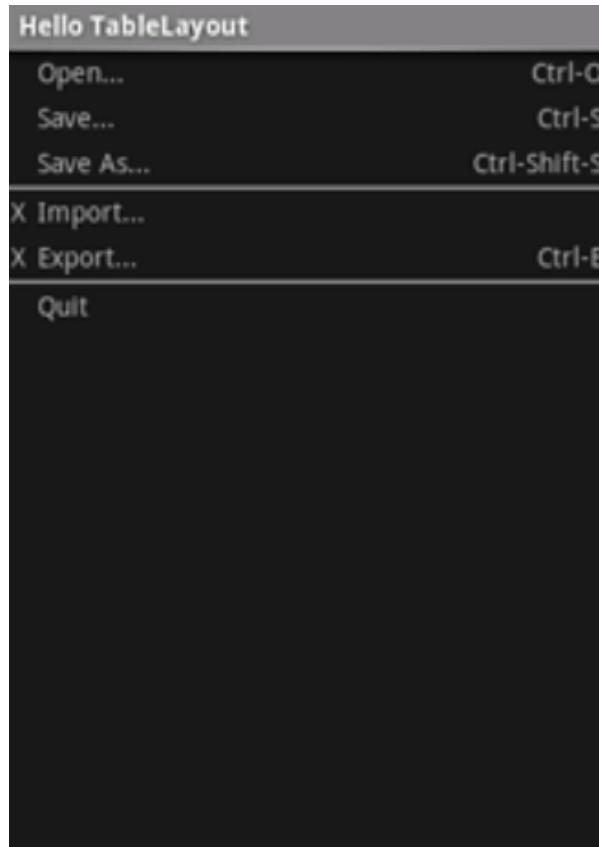
TableLayout is a ViewGroup that displays child View elements in rows and columns. TableLayout positions its children into rows and columns. TableLayout containers do not display border lines for their rows,

 <https://developer.android.com/guide/topics/ui/layout/grid>

Views>Layouts>TableLayout>04_Stretchable

Open...	Ctrl-O
Save As...	Ctrl-Shift-S

TableLayout은 내부에 있는 View를 행과 열로 배치하는 layout이다.



▼ FRAME

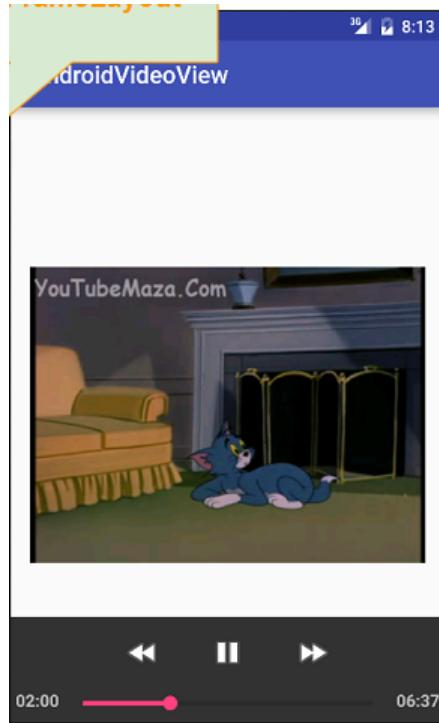
FrameLayout | Android Developers

AccessibilityService.MagnificationController.OnMagnificationChangedListener

 <https://developer.android.com/reference/android/widget/FrameLayout>

Developers 

FrameLayout은 단 하나의 item을 보여주는 공간이다. 여러 item이나 화면을 하나의 고정된 공간에 보여줄 때 자주 사용한다.



▼ DRAWER

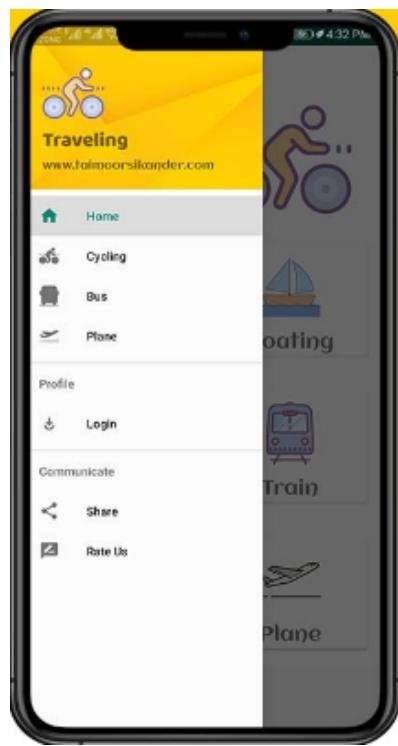
DrawerLayout | Android Developers

MediaSessionCompat.OnActiveChangeListener

👉 <https://developer.android.com/reference/androidx/drawerlayout/widget/DrawerLayout>



DrawerLayout은 왼쪽 또는 오른쪽에서 스와이프 액션으로 이미 있는 화면 위에 겹쳐지는 화면을 만드는 layout이다. 이 layout의 위치는 `android:layout_gravity` 속성을 이용해서 지정할 수 있다. 그리고 `DrawerListener`를 이용해서 drawer view의 상태를 모니터링할 수 있다.



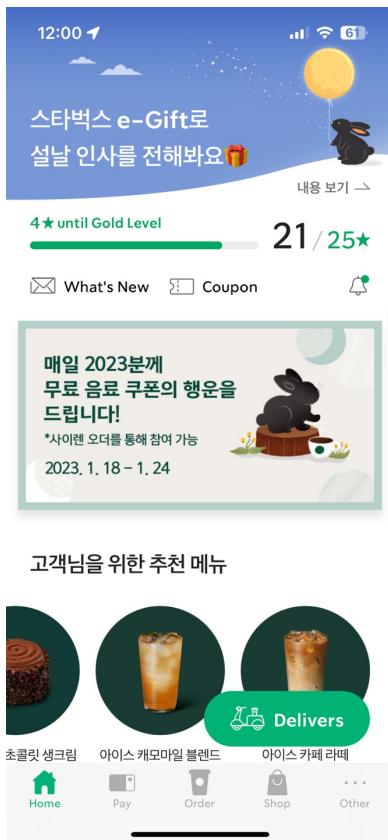
보통 navigation drawer를 구현하기 위해 사용되는 layout이다. navigation drawer는 앱의 menu와 연결되어서 앱에 있는 여러 activity, fragment 사이를 쉽게 오갈 수 있게 도와준다.

구현

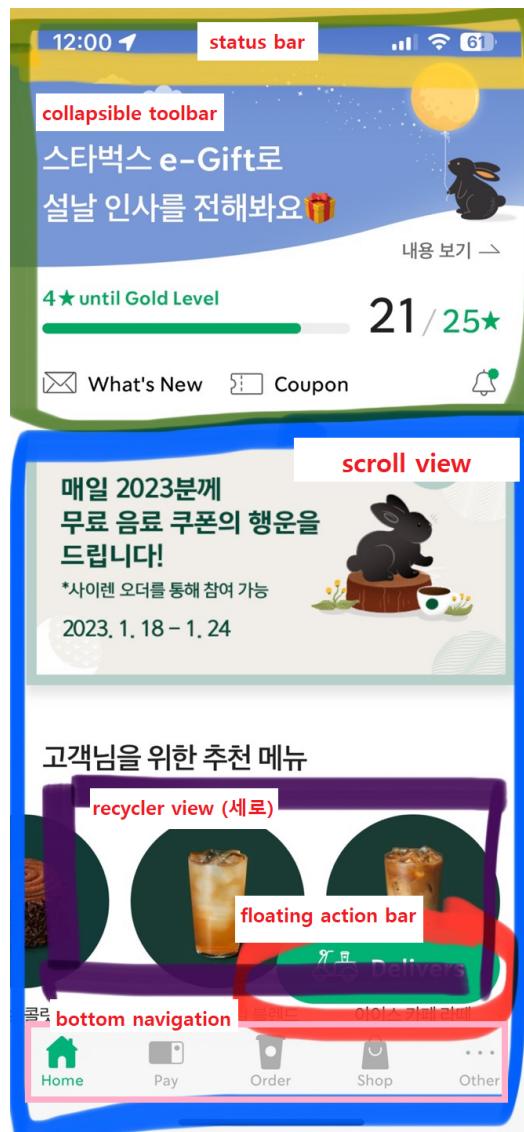


xml 파일만을 이용해서 상용되고 있는 앱의 한 화면을 똑같이 구현한다.

💡 Starbucks 기존 앱 디자인



🌟 Starbucks 기존 앱 디자인 분석

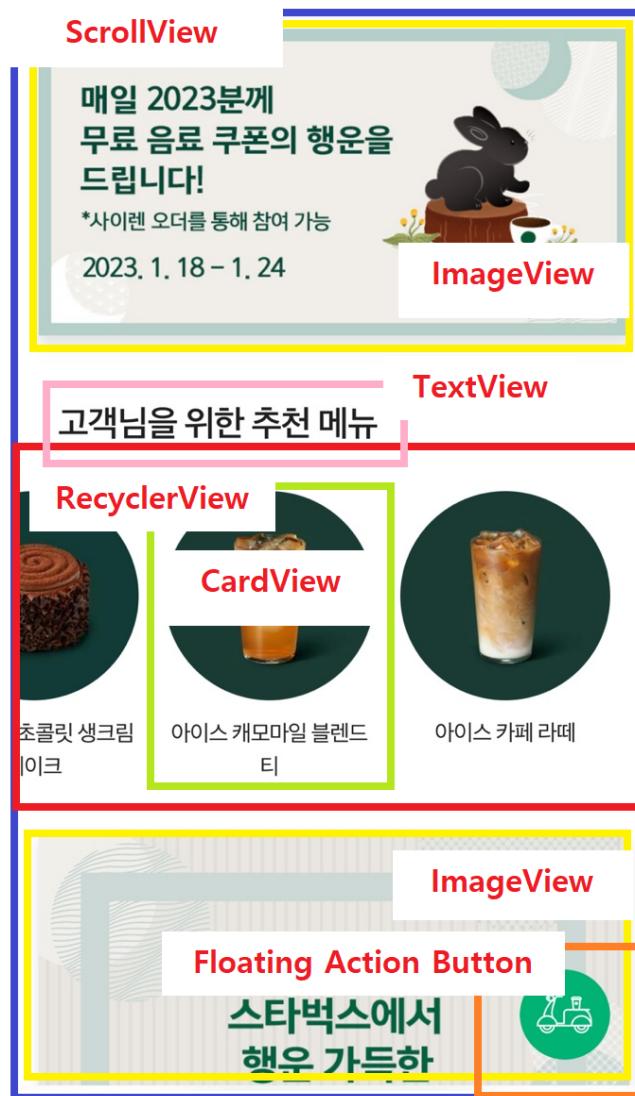


1. Collapsible toolbar & expandable



collapsible toolbar를 이용해서 스크롤 액션과 함께 toolbar의 크기가 변하게 해야 한다. 아래 스크롤 액션 시 최하단에 있는 버튼 3개만을 제외하고 위로 없어지는 animation이 필요하다.

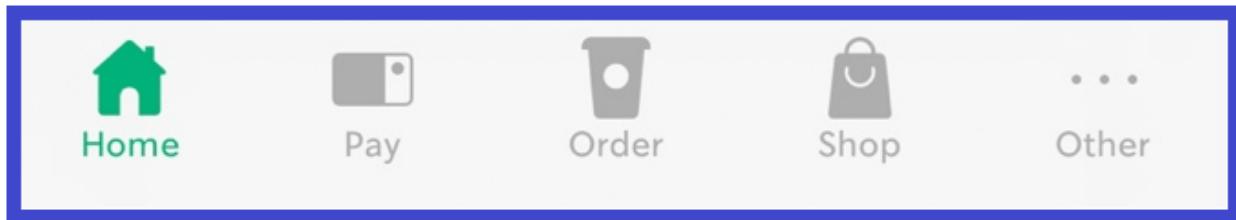
2. ScrollView



- 가장 상위 layout으로 ScrollView를 설정해서 스크롤을 하면서 여러 쿠폰과 이벤트를 알려주는 ImageView와 제목을 알려주는 TextView가 보이게 해야 한다.
- 그리고 그 안에 RecyclerView를 nesting시켜서 여러 커피 메뉴를 오른쪽으로 스크롤 할 수 있게 만 들어줘야 한다.

- 마지막으로 floating action button을 만들어서 클릭했을 때 원하는 액션을 가능하게 해야 한다. 그리고 collapsible toolbar와 연동해서 floating action button의 모양이 변해야 한다.

3. bottom navigation



bottom navigation에는 5개의 항목이 icon과 text가 동시에 표시되어야 하고 클릭 시 색이 초록으로 바뀌어야 한다.

⭐️ Starbucks 앱 결과물

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/e1c7870a-a2e5-43d5-ac4-ce38bbe2a69f/Untitled.mp4>

⭐️ 만들면서 사용한 새로운 구현방법

[Collapsible Toolbar 구현](#)

[Toolbar 안에 원하는 위치에 button 넣기](#)

[Expanded Floating Action Button 구현](#)

[Bottom Navigation](#)