

세상의 속도를
따라잡고 싶다면

**Do
it!**

깡샘의
안드로이드
앱 프로그래밍
with **코틀린**

이지스퍼블리싱(주)

12

머티리얼 라이브러리

12-1 앱바 사용하기

12-2 탭 레이아웃 – 탭 버튼 구성

12-3 내비게이션 뷰 – 드로어 화면 구성

12-4 확장된 플로팅 액션 버튼

12-5 머티리얼 라이브러리로 화면 구성하기

12-1 앱바 사용하기

머티리얼 라이브러리란?

- 구글의 머티리얼 디자인은 모바일과 데스크톱, 그리고 그 밖에 다양한 장치를 아우르는 일관된 애플리케이션 디자인 지침

• 머티리얼 라이브러리 선언

```
implementation 'com.google.android.material:material:1.4.0'
```

12-1 앱바 사용하기

앱바 레이아웃 – 화면 위쪽 영역 꾸미기

- 앱바란 화면 위쪽의 꾸밀 수 있는 영역



- 툴바 포함하기

• 앱바 레이아웃에 툴바 포함

```
<com.google.android.material.appbar.AppBarLayout
    android:id="@+id/appbar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar">
    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize" />
</com.google.android.material.appbar.AppBarLayout>
```

툴바 포함

▶ 실행 결과

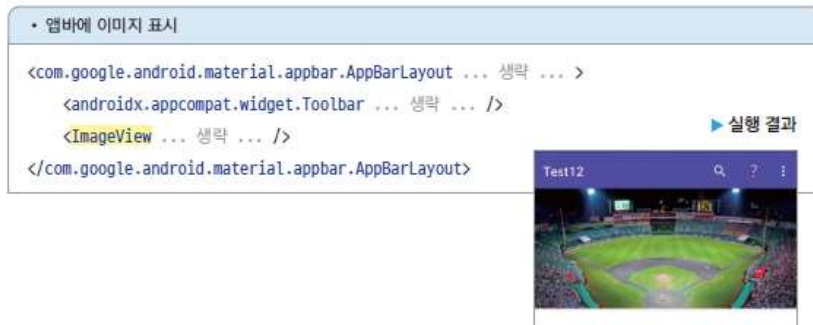
Test12

12-1 앱바 사용하기

- 크기 확장하기



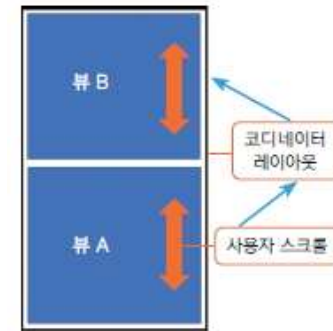
- 이미지 넣기



12-1 앱바 사용하기

코디네이터 레이아웃 – 뷰끼리 상호 작용하기

- 스크롤 연동하기
 - 레이아웃은 뷰끼리 상호 작용해야 할 때 사용
 - 뷰에서 발생한 스크롤 정보를 코디네이터 레이아웃이 받아서 다른 뷰에 전달
- 중첩 스크롤 뷰 사용하기
 - 코디네이터 레이아웃에 중첩 스크롤 뷰(NestedScrollView)를 포함하고 여기에 텍스트 뷰나 이미지 뷰를 넣으면 해당 뷰에서 발생하는 스크롤 정보를 코디네이터 레이아웃에 전달



• 중첩 스크롤 뷰

```
<androidx.coordinatorlayout.widget.CoordinatorLayout ... 생략 ... >
<com.google.android.material.appbar.AppBarLayout ... 생략 ... >
<androidx.appcompat.widget.Toolbar ... 생략 ...
    app:layout_scrollFlags="scroll|enterAlways" />
<ImageView ... 생략 ...
    app:layout_scrollFlags="scroll|enterAlways" />
</com.google.android.material.appbar.AppBarLayout>
<androidx.core.widget.NestedScrollView ... 생략 ...
    app:layout_behavior="@string/appbar_scrolling_view_behavior">
    <TextView ... 생략 ... />
</androidx.core.widget.NestedScrollView>
</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

12-1 앱바 사용하기

컬래핑 툴바 레이아웃 – 앱바 접히는 형태 설정하기

- 컬래핑 툴바 레이아웃(CollapsingToolbarLayout)은 앱바 레이아웃 하위에 선언하여 앱바가 접힐 때 다양한 설정을 할 수 있는 뷰
- title 속성으로 앱바의 제목을 설정
- expandedTitleMarginStart, expandedTitleMarginBottom 속성으로 앱바가 접히지 않았을 때 제목의 위치를 설정
- 내용이 정상으로 출력되지 못하는 상황이라면 contentScrim 속성에 지정한 색상으로 앱바를 출력



• 컬래핑 툴바 레이아웃 등록

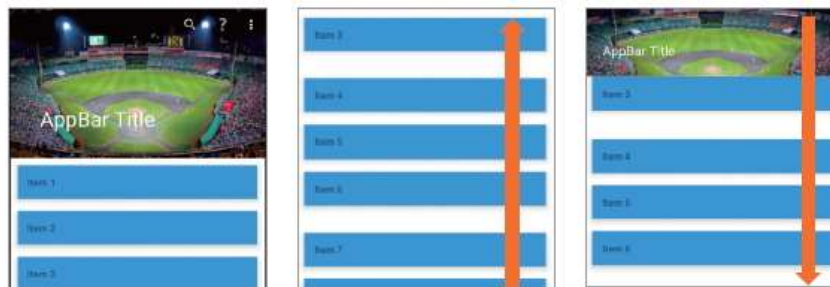
```
<androidx.coordinatorlayout.widget.CoordinatorLayout ... 생략 ... >
    <com.google.android.material.appbar.AppBarLayout ... 생략 ... >
        <com.google.android.material.appbar.CollapsingToolbarLayout ... 생략 ...
            app:contentScrim="?attr/colorPrimary"
            app:expandedTitleMarginBottom="50dp"
            app:expandedTitleMarginStart="48dp"
            app:layout_scrollFlags="scroll|exitUntilCollapsed"
            app:title="AppBar Title">
                <ImageView ... 생략 ...
                    app:layout_collapseMode="parallax" />
                <androidx.appcompat.widget.Toolbar ... 생략 ...
                    app:layout_collapseMode="pin" />
            </com.google.android.material.appbar.CollapsingToolbarLayout>
        </com.google.android.material.appbar.AppBarLayout>
        <androidx.recyclerview.widget.RecyclerView ... 생략 ...
            app:layout_behavior="@string/appbar_scrolling_view_behavior" />
    </androidx.coordinatorlayout.widget.CoordinatorLayout>
```

12-1 앱바 사용하기

- 스크롤 설정하기
 - 앱바가 스크롤될지를 설정하는 `layout_scrollFlags`
 - `scroll | enterAlways`: 스크롤 시 완전히 사라졌다가 거꾸로 스크롤 시 처음부터 다시 나타납니다.
 - `scroll | enterAlwaysCollapsed`: 스크롤 시 완전히 사라졌다가 거꾸로 스크롤 시 처음부터 나타나지 않고 메인 콘텐츠 부분이 끝까지 스크롤된 다음에 나타납니다.
 - `scroll | exitUntilCollapsed`: 스크롤 시 모두 사라지지 않고 툴바를 출력할 정도의 한 줄만 남을 때까지 스크롤됩니다.

• scroll | enterAlways 속성값

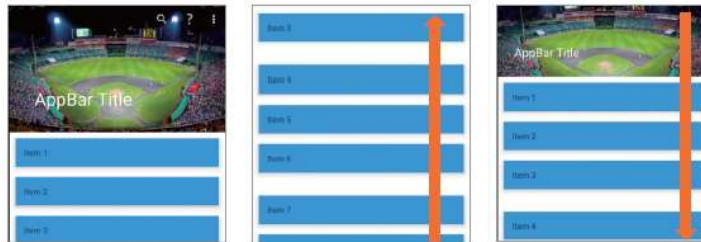
```
app:layout_scrollFlags="scroll|enterAlways"
```



12-1 앱바 사용하기

- scroll | enterAlwaysCollapsed 속성값

```
app:layout_scrollFlags="scroll|enterAlwaysCollapsed"
```



- scroll | exitUntilCollapsed 속성값

```
app:layout_scrollFlags="scroll|exitUntilCollapsed"
```

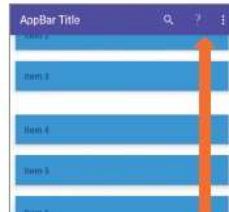


12-1 앱바 사용하기

- 개별 뷰의 스크롤 설정하기
 - 하위 뷰마다 스크롤 설정은 layout_collapse Mode 속성
 - pin: 고정되어 스크롤되지 않습니다.
 - parallax: 함께 스크롤됩니다.

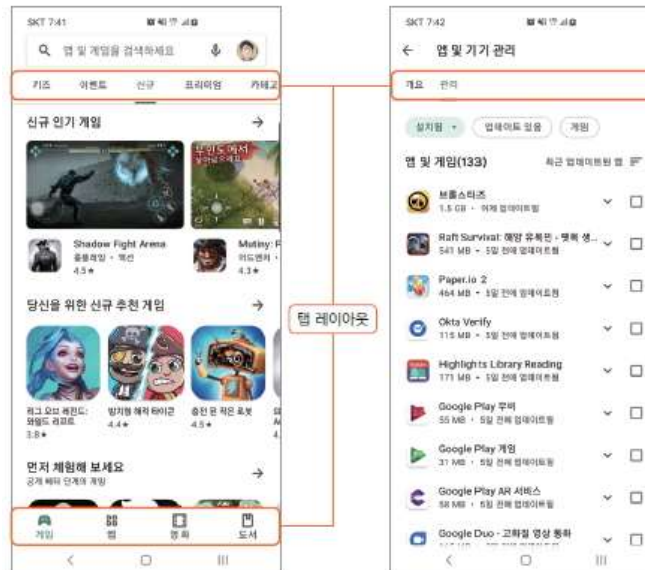
• 개별 뷰의 스크롤 설정

```
<com.google.android.material.appbar.CollapsingToolbarLayout ... 생략 ...  
  app:layout_scrollFlags="scroll|exitUntilCollapsed">  
    <ImageView ... 생략 ...  
      app:layout_collapseMode="parallax" />  
    <androidx.appcompat.widget.Toolbar ... 생략 ...  
      app:layout_collapseMode="pin" />  
</com.google.android.material.appbar.CollapsingToolbarLayout>
```



12-2 탭 레이아웃 - 탭 버튼 구성

- 탭 레이아웃은 탭tab으로 구분하는 화면에서 탭 버튼을 배치하는 레이아웃



12-2 탭 레이아웃 - 탭 버튼 구성

• 탭 레이아웃 등록

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <com.google.android.material.tabs.TabLayout
        android:id="@+id/tabs"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <FrameLayout
        android:id="@+id/tabContent"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

• 코드에서 탭 버튼 정의

```
val tab1: TabLayout.Tab = tabLayout.newTab()
tab1.text="Tab1"
tabLayout.addTab(tab1)

val tab2: TabLayout.Tab = tabLayout.newTab()
tab2.text="Tab2"
tabLayout.addTab(tab2)

val tab3: TabLayout.Tab = tabLayout.newTab()
tab3.text="Tab3"
tabLayout.addTab(tab3)
```

12-2 탭 레이아웃 – 탭 버튼 구성

- 탭 버튼을 코드에서 정의하지 않고 레이아웃 XML 파일의 TabItem으로 정의

• XML 파일에서 탭 버튼 정의

```
<com.google.android.material.tabs.TabLayout
    android:id="@+id/tabs"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <com.google.android.material.tabs.TabItem
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tab1" />
    <com.google.android.material.tabs.TabItem
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tab2" />
    <com.google.android.material.tabs.TabItem
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tab3" />
</com.google.android.material.tabs.TabLayout>
```

12-2 탭 레이아웃 - 탭 버튼 구성

- 탭 버튼의 이벤트 핸들러

• 탭 버튼 이벤트 처리

```
tabLayout.addTabSelectedListener(object: TabLayout.OnTabSelectedListener {  
    // 탭 버튼을 선택할 때 이벤트  
    override fun onTabSelected(tab: TabLayout.Tab?) {  
        val transaction = supportFragmentManager.beginTransaction()  
        when (tab?.text) {  
            "Tab1" -> transaction.replace(R.id.tabContent, OneFragment())  
            "Tab2" -> transaction.replace(R.id.tabContent, TwoFragment())  
            "Tab3" -> transaction.replace(R.id.tabContent, ThreeFragment())  
        }  
        transaction.commit()  
    }  
    // 선택된 탭 버튼을 다시 선택할 때 이벤트  
    override fun onTabReselected(tab: TabLayout.Tab?) {  
        (... 생략 ...)  
    }  
    // 다른 탭 버튼을 눌러 선택된 탭 버튼이 해제될 때 이벤트  
    override fun onTabUnselected(tab: TabLayout.Tab?) {  
        (... 생략 ...)  
    }  
})
```

▶ 실행 결과



12-2 탭 레이아웃 - 탭 버튼 구성

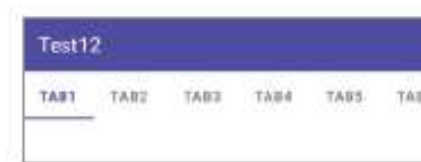
- 탭 버튼 정렬하기

- tabGravity는 탭 버튼을 정렬하는 속성



- 스크롤 설정하기

- tabMode 속성은 탭 버튼을 스크롤할 수 있는지를 설정
- fixed은 스크롤을 지원하지 않는다는 의미
- scrollable로 설정하면 탭 버튼이 왼쪽부터 나열되고 모두 출력할 수 없다면 자동으로 가로 스크롤



12-2 탭 레이아웃 - 탭 버튼 구성

- 뷰 페이지저 연동하기
 - TabLayout과 ViewPager2를 등록한 후 코드에서 TabLayoutMediator를 이용해 둘을 연동

• 탭 레이아웃과 뷰 페이지 등록

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical">
    <com.google.android.material.tabs.TabLayout
        android:id="@+id/tabs"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:tabMode="scrollable">
    </com.google.android.material.tabs.TabLayout>
    <androidx.viewpager2.widget.ViewPager2
        android:id="@+id/viewpager"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

• 탭 레이아웃과 뷰 페이지 연동

```
TabLayoutMediator(tabLayout, viewPager) { tab, position ->
    tab.text = "Tab${(position + 1)}"
}.attach()
```


12-3 내비게이션 뷰 - 드로어 화면 구성

- 드로우어 레이아웃 화면에서 위쪽은 아이콘과 문자열 등을 배치했고 아래쪽은 메뉴 항목을 나열


• 내비게이션 뷰 등록

```
<androidx.drawerlayout.widget.DrawerLayout ... 생략 ...>
  <LinearLayout ... 생략 ...>
    (... 생략 ...)
  </LinearLayout>

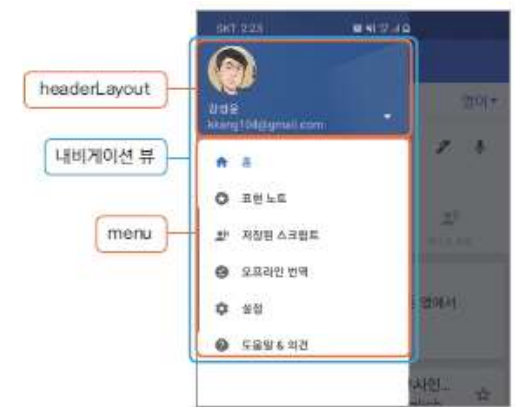
  <com.google.android.material.navigation.NavigationView
    android:id="@+id/main_drawer_view"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:layout_gravity="start"

    app:headerLayout="@layout/navigation_header"
    app:menu="@menu/menu_navigation" />
</androidx.drawerlayout.widget.DrawerLayout>
```

▶ 실행 결과



The screenshot shows a mobile app interface. At the top is a red header bar with the text "Navigation Title". Below the header is a white navigation drawer that is currently open, displaying a list of four items: "아이템1", "아이템2", "아이템3", and "아이템4". Each item has a small icon to its left. The background of the app is dark grey.



12-3 내비게이션 뷰 – 드로어 화면 구성

- 메뉴를 구성한 XML 파일을 menu 속성에 지정만 해도 실행 결과처럼 항목이 자동으로 출력
- 항목 선택 이벤트는 DrawerLayout의 setNavigationItemSelectedListener() 함수로 이벤트 핸들러를 지정

• 항목 선택 이벤트 핸들러

```
binding.mainDrawerView.setNavigationItemSelectedListener {  
    Log.d("kkang", "navigation item click... ${it.title}")  
    true  
}
```

12-4 확장된 플로팅 액션 버튼

- 화면에 떠 있는 듯한 버튼

• 확장된 플로팅 액션 버튼

```
<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton  
    (... 생략 ...)   
    android:text="extended FAB"   
    app:icon="@android:drawable/ic_input_add" />
```



• 확장된 플로팅 액션 버튼 조절

```
binding.extendedFab.setOnClickListener {   
    when (binding.extendedFab.isExtended) {   
        true -> binding.extendedFab.shrink()   
        false -> binding.extendedFab.extend()   
    }   
}
```

아이콘만 표시

아이콘과 문자열 함께 표시했다면

머티리얼 라이브러리로 화면 구성하기

1단계. 새 모듈 생성하기

- Ch12_Material이라는 이름으로 새로운 모듈을 만듭니다.

2단계. 빌드 그래들 작성하기

- 뷰 바인딩을 위한 설정

3단계. 준비된 파일 복사하기

- drawable, layout, menu, values 디렉터리를 현재 모듈의 res 디렉터리에 복사
- 소스가 있는 디렉터리의 kt 파일을 소스 영역에 복사

시계 앱의 스톱워치 기능 만들기

4단계. 레이아웃 XML 파일 작성하기

- activity_main.xml 파일을 열고 작성합니다.

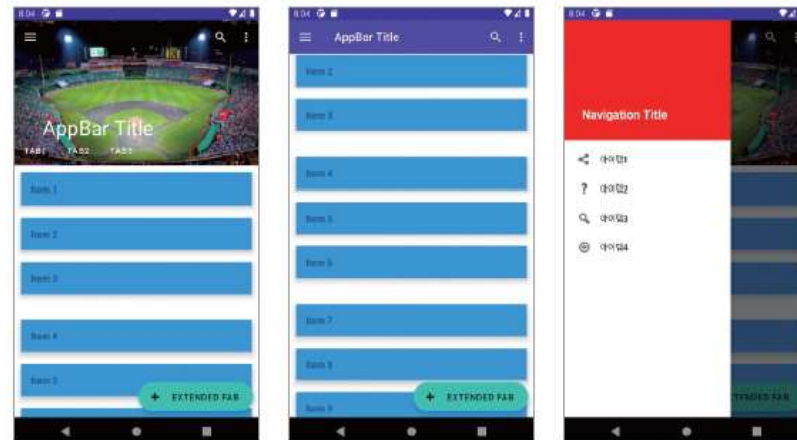
5단계. 메인 액티비티 파일 작성하기

- MainActivity.kt 파일을 열고 onCreate() 함수에 코드를 추가합니다.

Do it! 실습

시계 앱의 스톱워치 기능 만들기

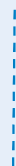
6단계. 앱 실행하기





감사합니다

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



윌리엄 셰익스피어
William Shakespeare