

프래그먼트

모바일게임프로그래밍
김지심교수

학습목표



프래그먼트의 개념을 설명할 수 있다.
프래그먼트를 구현할 수 있다.

01



프래그먼트란?



1. 프래그먼트란?

프래그먼트(Fragment)

액티비티보다 작은 단위의 화면(sub-activity)

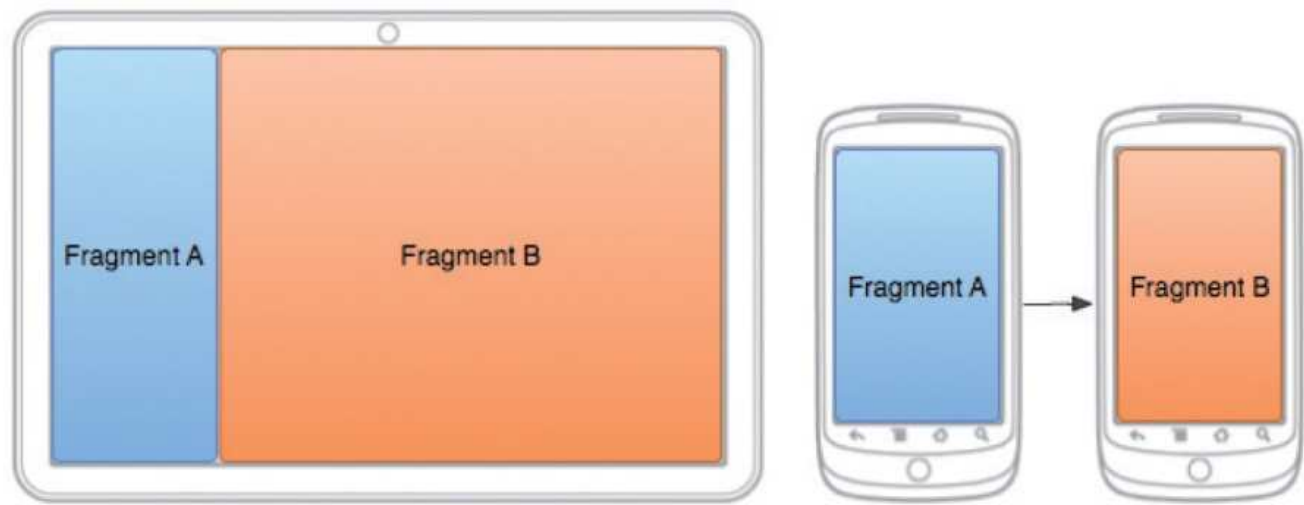
프래그먼트를 사용하면,

하나의 액티비티 안에 여러 UI를 구성할 수 있음

→ 대형화면에서 액티비티 화면을 분할해서 표현하거나,
스마트폰과 같은 소형화면에서는 화면의 분할보다는 실행 중에
화면을 동적으로 운영하는 데 더 많이 활용

하나의 Activity에 독립적으로 동작하는 부분화면을 만들 때 유용

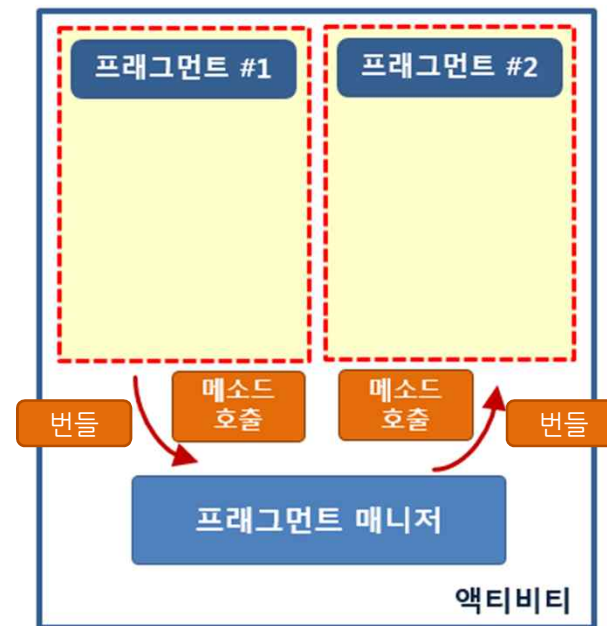
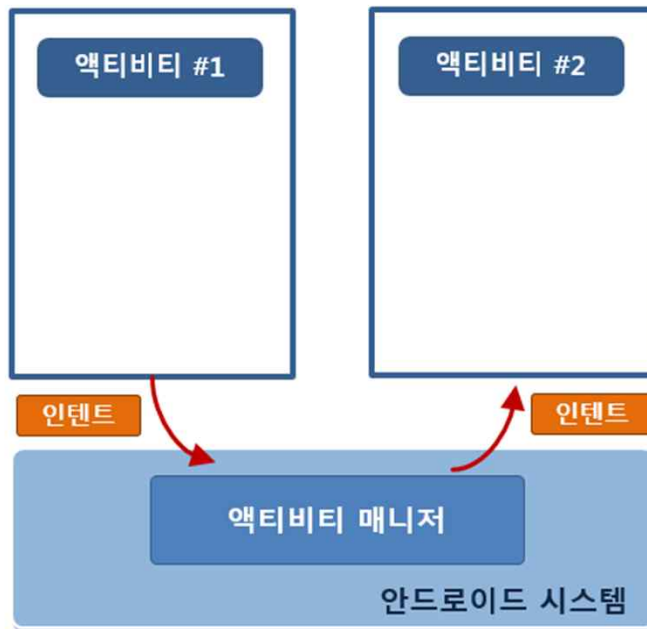
코드가 복잡해지는 문제를 해결하기 위해 부분화면의 코드를 분리시키고
안드로이드의 기본 원칙(뷰가 액티비티에 결속됨)을 피하여 유연성을 향상



1. 프래그먼트란?

프래그먼트 동작원리

액티비티를 본떠서 만든 것이나, 시스템에서 이해하는 객체인 intent를 사용하지 않고 메소드를 만들고 호출하는 방식으로 사용하며 FragmentManager로 구현



1. 프래그먼트란?

프래그먼트

※ Bundle

액티비티가 비정상적으로 종료된 경우 등에 액티비티의 상태를 저장된 데이터 세트로 가지고 있다가 액티비티가 호출되었을 때 액티비티의 인스턴스를 생성하여 사용자가 액티비티를 계속 사용할 수 있도록 함

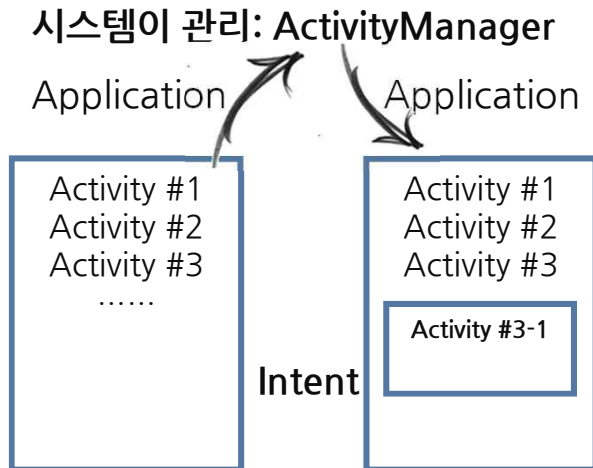
Bundle이란?

시스템이 이전 상태를 복원하기 위해 사용하는 객체로서, Key-value 형식으로 구성됨

<https://developer.android.com/training/basics/activity-lifecycle/recreating.html?hl=ko>

1. 프래그먼트란?

<Activity와 Intent>



시스템에서 관리하므로
이 원리를 앱 내에서 구현한다면
자원 낭비 등의 문제점 발생

Activity 내에
별도 설계한
Layout(XML)을
inflation하여
부분화면으로 사용

LayoutInflater

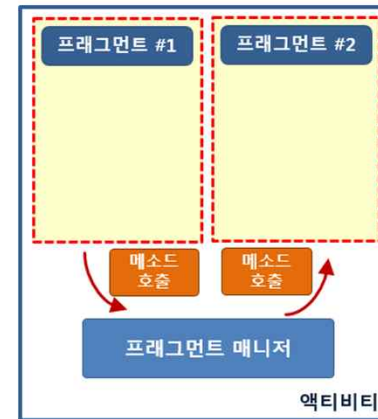
- Activity의 수명주기에 영향받음
- 복잡도 ↑

<Fragment>

등장: 허니콤(3.0), API11
목적: 화면분할,
독립적 리소스 관리

Activity가 관리: FragmentManager

여러 화면 구현 시
Fragment
(sub-Activity)
사용



하나의 Activity를
구성하기 위해,
activity_main.xml
MainActivity.java

하나의 Fragment를 구성하기 위해,
fragment.xml
Fragment.java

onCreateView(LayoutInflater, ViewGroup, Bundle)
nflater.inflate(R.layout.fragment1,container,false);
→ inflation한 View를 리턴

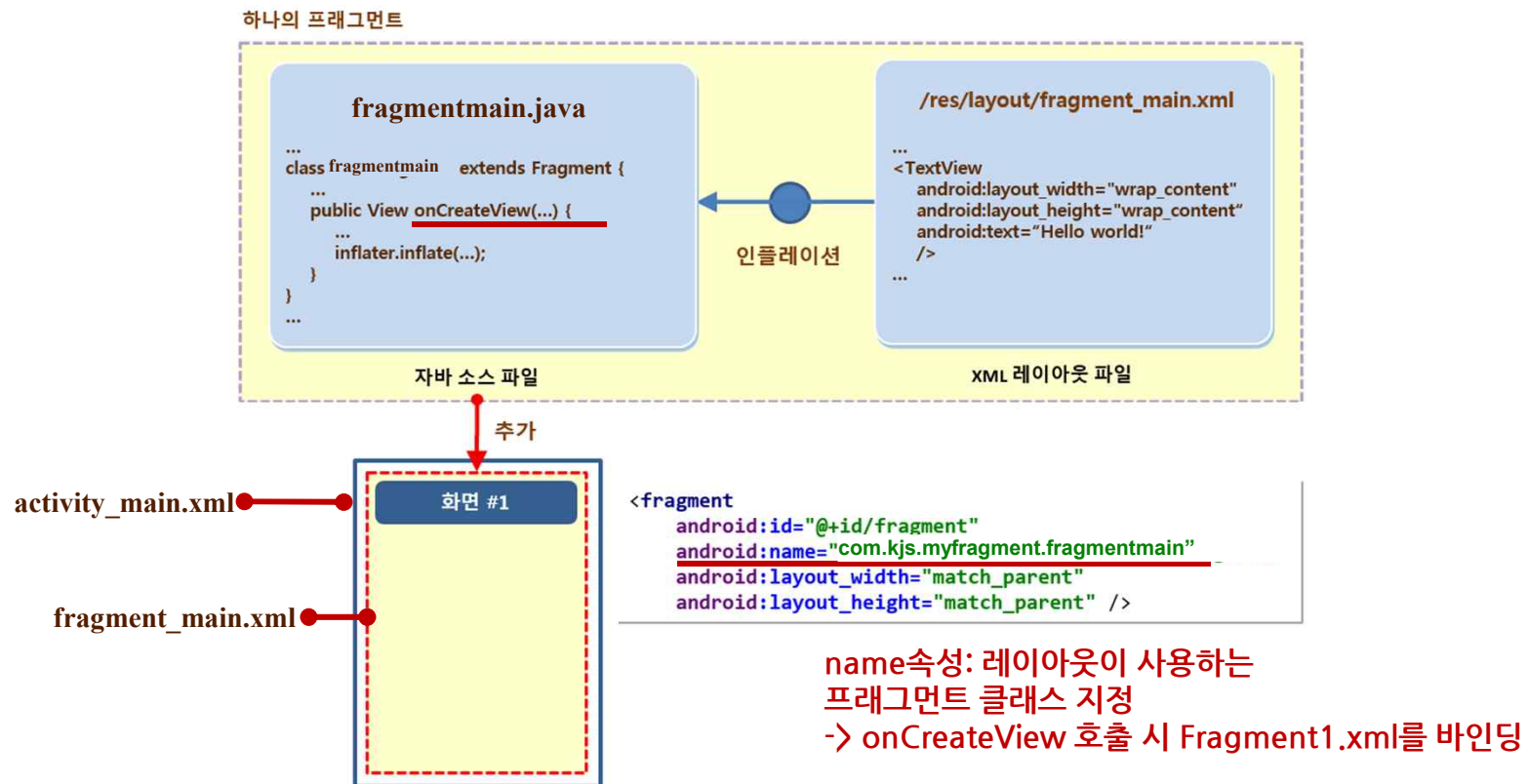
Fragment를 Activity 상의 뷰로 인플레이션

```
<Fragment  
    android:name=앱패키지명.프래그먼트클래스  
    ...  
>
```


1. 프래그먼트란?

프래그먼트

액티비티 위에 올려야 프래그먼트로 동작함
Framgent를 상속받아 서브 클래스를 생성해야 함
인플레이션을 위한 콜백 메소드(onCreateView)를 제공함



1. 프래그먼트란?

Reference

android.app.Fragment가 아닌

android.support.v4.app.Fragment를 사용하자

Fragment 클래스: Fragment와 이를 가진 Activity를 관리

구분	내용
<code>onCreateView</code> (LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	인플레이션이 필요한 시점에 자동으로 호출되어 XML 파일의 내용을 인플레이션한 후 클래스에서 사용토록 함 - LayoutInflater : LayoutInflater.inflate를 호출하여 뷰를 인플레이팅 - ViewGroup : 뷰의 부모 - Bundle : 인플레이팅된 뷰를 뷰의 부모에게 추가할 것인지를 LayoutInflater에게 알려줌 (해당 액티비티에서 바로 인플레이팅된 뷰를 추가할 경우 false)
<code>public final Activity getActivity()</code>	해당 프래그먼트를 포함하는 액티비티를 리턴
<code>public final FragmentManager getFragmentManager()</code>	해당 프래그먼트를 포함하는 액티비티에서 프래그먼트 객체들과 의사소통하는 FragmentManager 리턴
<code>public final Fragment getParentFragment()</code>	해당 프래그먼트를 포함하는 부모가 프래그먼트일 경우 리턴, 액티비티이면 null 리턴

1. 프래그먼트란?

Reference

FragmentManager 클래스: Fragment를 운영(프래그먼트를 액티비티에 추가하거나 다른 프래그먼트로 바꾸거나 삭제 시 사용하며getFragmentManager()로 참조)

구분	내용
public abstract Fragment findFragmentById()	id로 프래그먼트 객체를 참조
public abstract FragmentTransaction beginTransaction()	프래그먼트를 변경하기 위한 트랜잭션을 시작

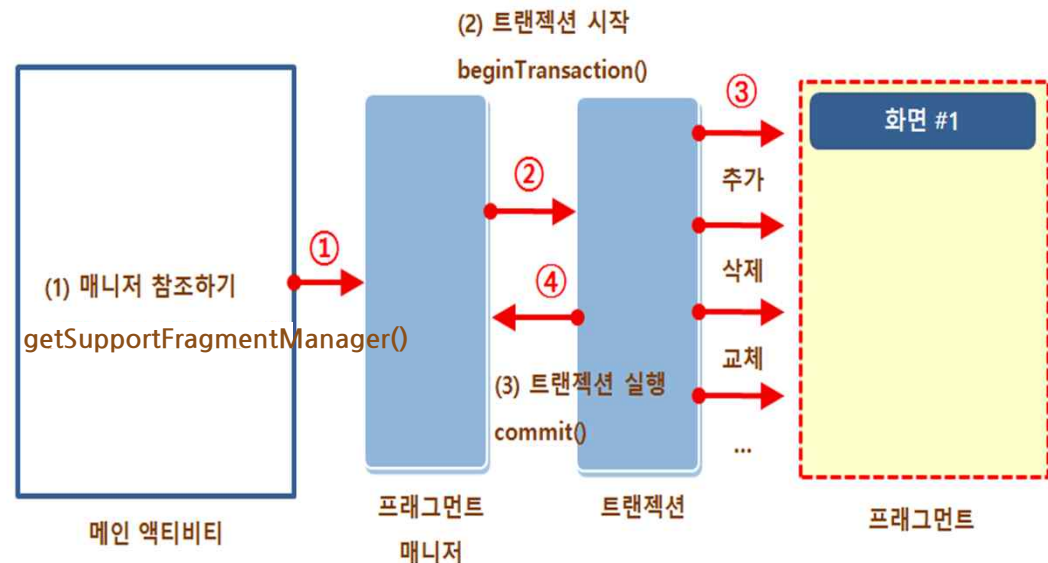
FragmentTransaion 클래스: Fragment의 트랜잭션(추가, 제거, 변경 등) 작업 수행

1. 프래그먼트란?

프래그먼트가 액티비티의 프래그먼트 매니저를 통해 의사소통하는 방식

ex. 화면 이동(교체)

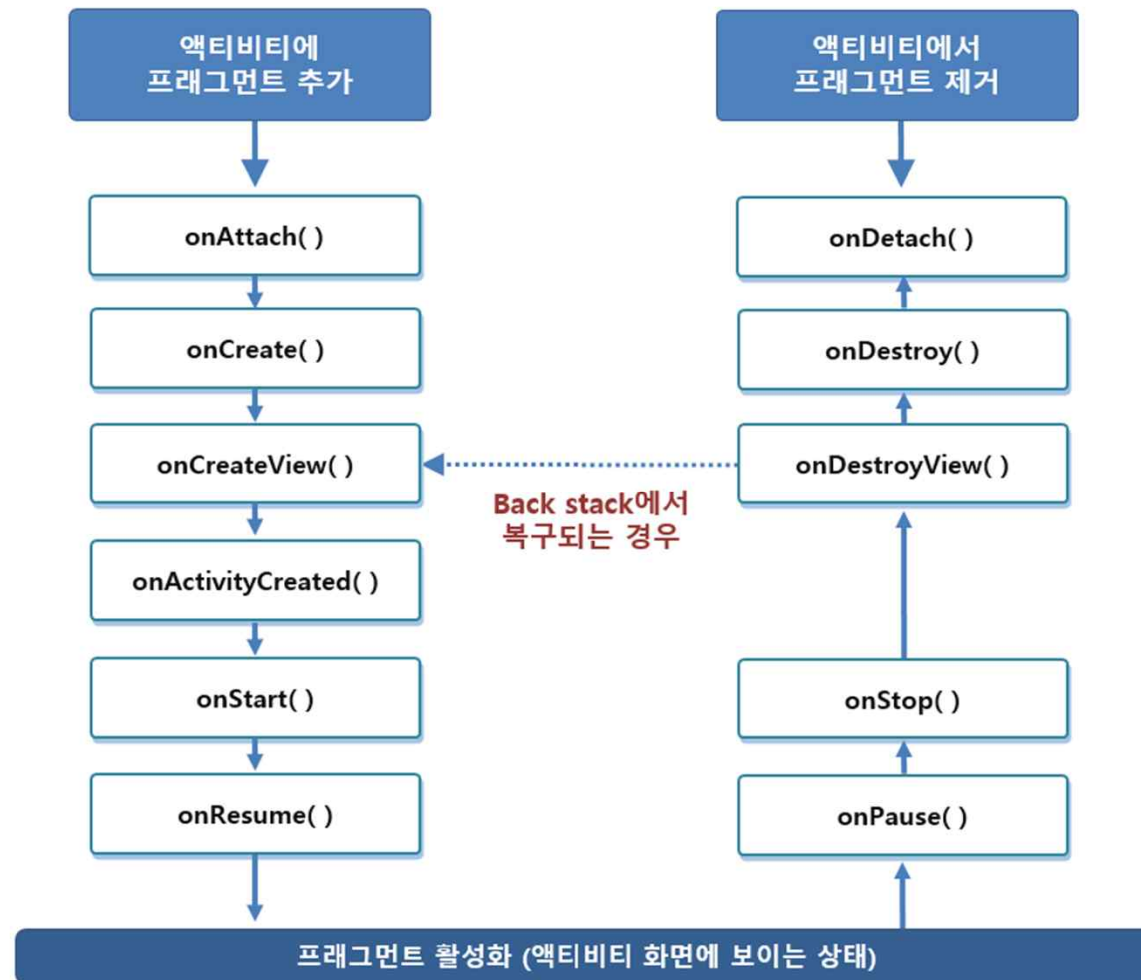
```
getFragmentManager();  
↓  
beginTransaction();  
↓  
replace();  
↓  
commit();
```



```
FragmentManager fm = getFragmentManager();  
FragmentTransaction fragmentTransaction = fm.beginTransaction();  
fragmentTransaction.replace(R.id.fragment_container, fr);  
fragmentTransaction.commit();
```

1. 프래그먼트란?

프래그먼트 생명주기



02



프래그먼트 실습



2. 프래그먼트 실습

실습: 01_FragmentTablet

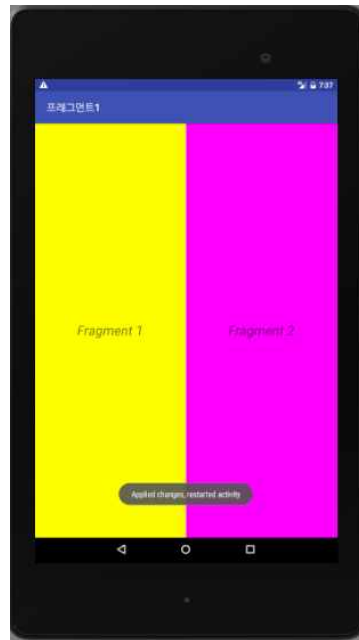
화면 크기에 따라 다른 프래그먼트를 출력

스마트폰



activity_main.xml
fragment1.xml
fragment2.xml

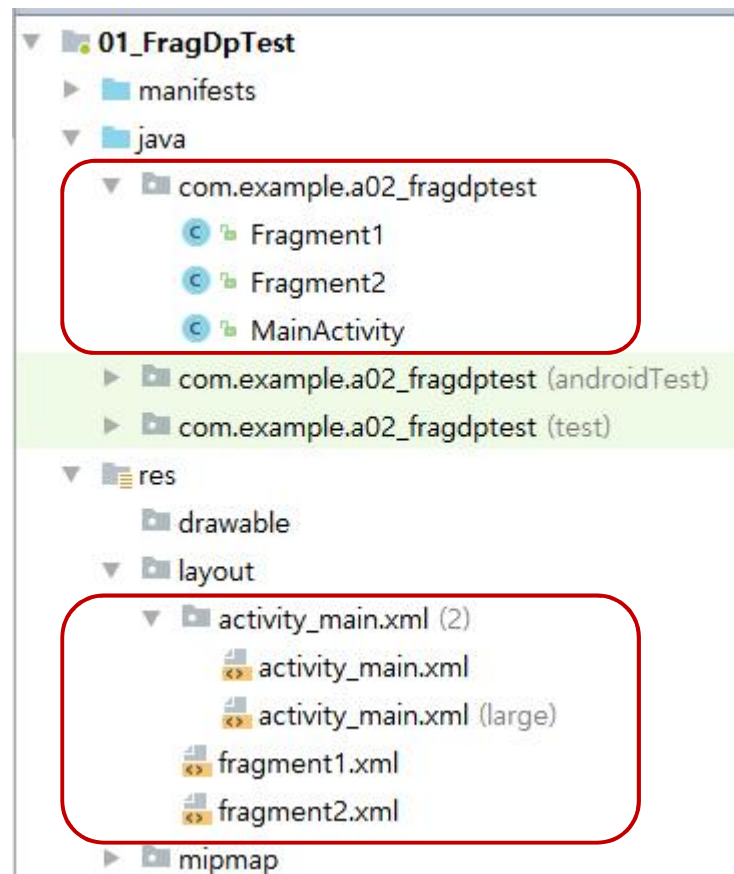
타블릿



activity_main(large).xml
fragment1.xml
fragment2.xml

2. 프래그먼트 실습

실습: 01_FragmentTablet

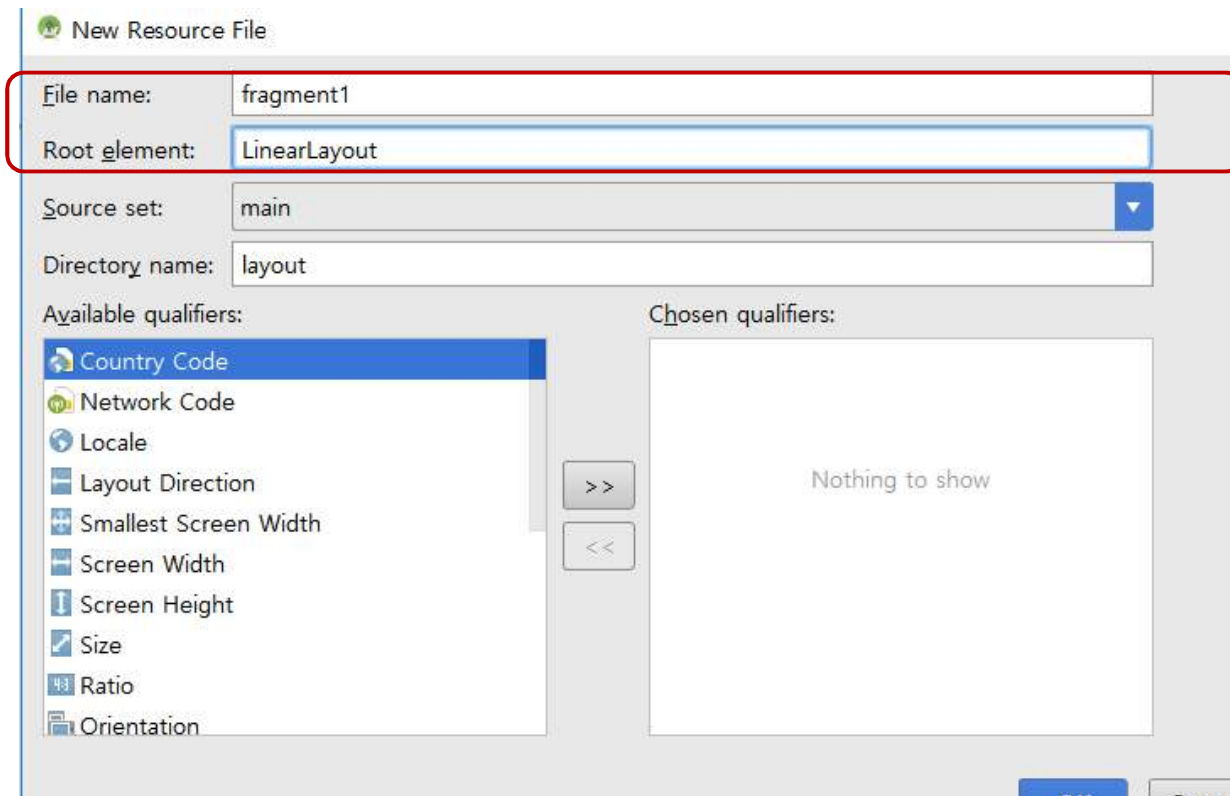


2. 프래그먼트 실습

실습: 01_FragmentTablet

프래그먼트 화면을 설계하기 위해 리소스로 fragment1.xml 생성

프로젝트 트리 창에서 [res] > [layout] 오른쪽 메뉴 > [New] > [Layout Resource File]에서 아래와 같이 생성



2. 프래그먼트 실습

실습: 01_FragmentTablet
fragment1.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:orientation="vertical"
7      android:background="#ffff00"
8      android:gravity="center">
9
10     <TextView
11         android:id="@+id/textView1"
12         android:layout_width="wrap_content"
13         android:layout_height="wrap_content"
14         android:text="Fragment1"
15         android:textSize="50dp"
16         android:textColor="@color/colorPrimaryDark"
17     />
18
19 </LinearLayout>
```



2. 프래그먼트 실습

실습: 01_FragmentTablet
fragment2.xml

이전 방법과 동일하게 리소스 파일로 추가하여 디자인

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     android:layout_width="match_parent"
5     android:layout_height="match_parent"
6     android:orientation="vertical"
7     android:background="#ff00ff"
8     android:gravity="center">
9
10    <TextView
11        android:id="@+id/textView1"
12        android:layout_width="wrap_content"
13        android:layout_height="wrap_content"
14        android:text="Fragment1"
15        android:textSize="50dp"
16        android:textColor="@color/colorPrimaryDark"
17    />
18
19 </LinearLayout>
```

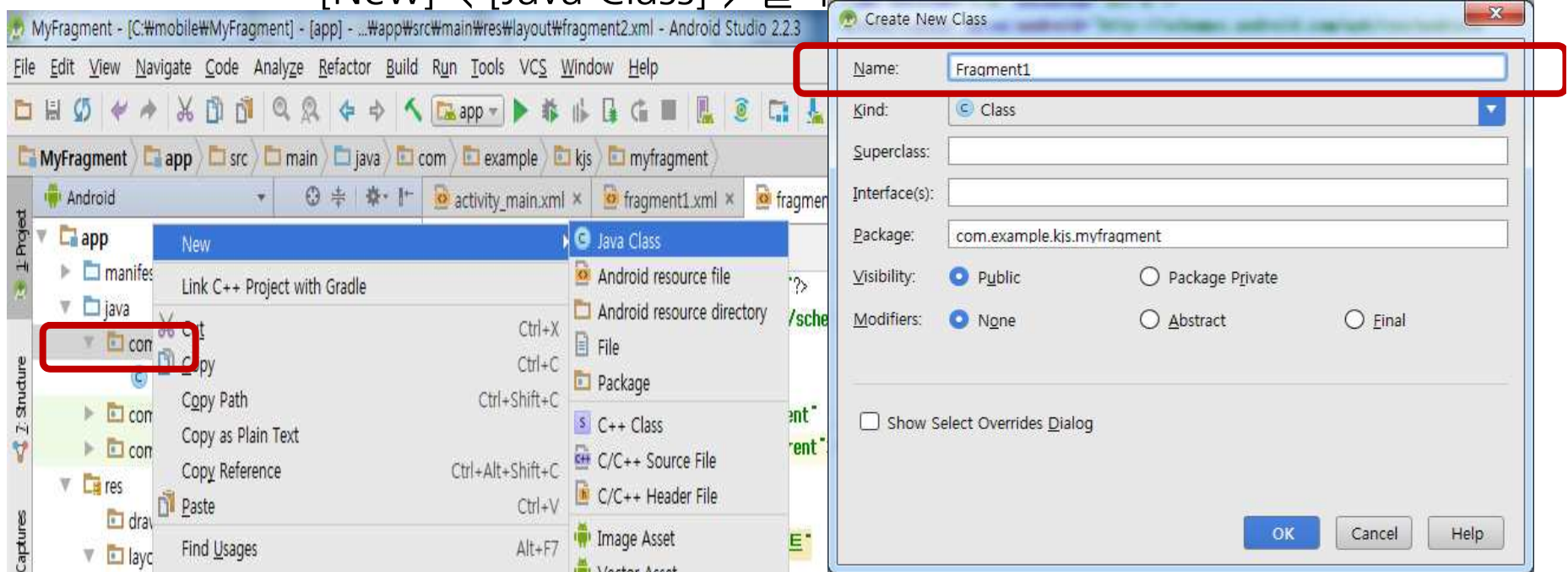


2. 프래그먼트 실습

실습: 01_FragmentTablet
Fragment1.java

fragment1 화면의 동작을 코딩하기 위해,

프로젝트 트리 창에서 패키지명 오른쪽 메뉴 >
[New] < [Java Class] > 클래스명: Fragment1



2. 프래그먼트 실습

실습: 01_FragmentTablet Fragment1.java

```
14 public class Fragment1 extends Fragment {  
15  
16     @Nullable  
17     @Override  
18     public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {  
19         //화면 inflate  
20         return inflater.inflate(R.layout.fragment1,container,false);  
21     }  
22 }  
23
```

Fragment2.java

위와 유사하게 코딩

MainActivity.jaga

수정사항 없음

2. 프래그먼트 실습

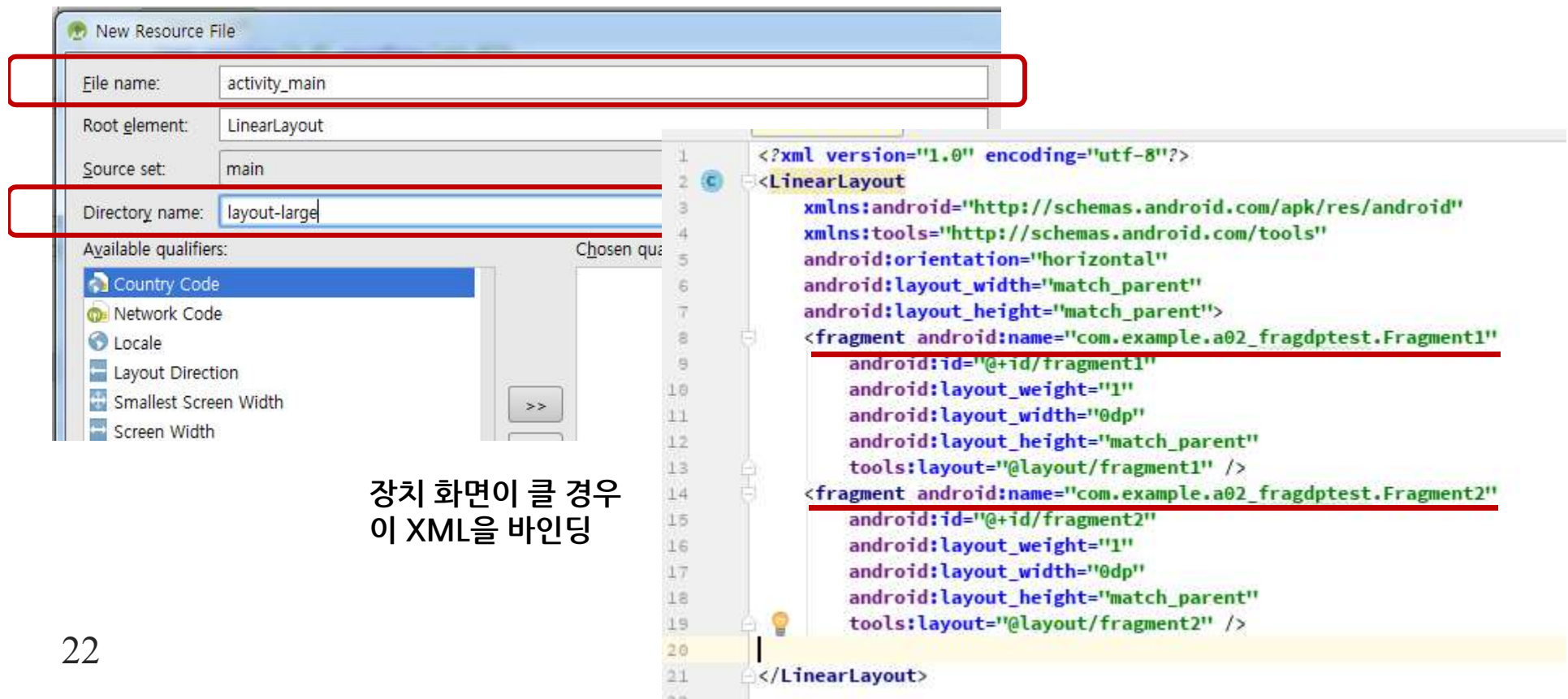
실습: 01_FragmentTablet
activity_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:orientation="horizontal">
8
9      <fragment
10         android:id="@+id/fragment1"
11         android:name="com.example.a02_fragdptest.Fragment1"
12         android:layout_width="0dp"
13         android:layout_height="match_parent"
14         android:layout_weight="1"
15         tools:layout="@layout/fragment1" />
16  </LinearLayout>
17
```


2. 프래그먼트 실습

실습: 01_FragmentTablet activity_main.xml (large)

프로젝트 트리 창에서 [res] > [layout] 오른쪽 메뉴 > [New] >
[Layout Resource File]에서 아래와 같이 생성
[Directory_name]: 'layout-large'



The screenshot shows the 'New Resource File' dialog on the left and the generated XML code on the right. The dialog has 'File name' set to 'activity_main', 'Root element' set to 'LinearLayout', and 'Source set' set to 'main'. The 'Directory name' is set to 'layout-large', which is highlighted with a red box. Below this, a list of 'Available qualifiers' includes 'Country Code', 'Network Code', 'Locale', 'Layout Direction', 'Smallest Screen Width', and 'Screen Width'. The XML code on the right is for 'activity_main.xml' and contains two fragments, 'Fragment1' and 'Fragment2', each with a red underline. The code is as follows:

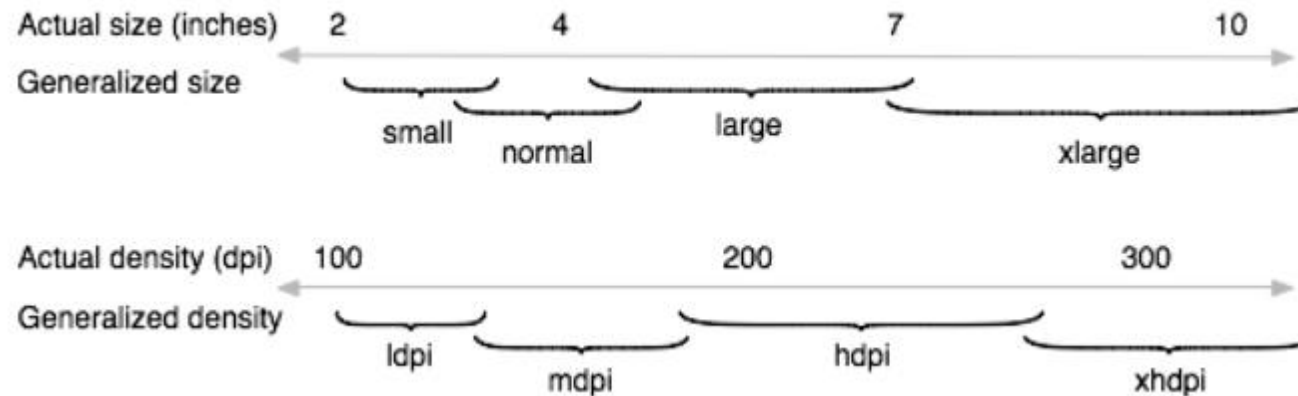
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <fragment android:name="com.example.a02_fragdptest.Fragment1"
        android:id="@+id/fragment1"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        tools:layout="@layout/fragment1" />
    <fragment android:name="com.example.a02_fragdptest.Fragment2"
        android:id="@+id/fragment2"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        tools:layout="@layout/fragment2" />
</LinearLayout>
```

장치 화면이 클 경우
이 XML을 바인딩

2. 프래그먼트 실습

※ Multiple Screen size

https://developer.android.com/guide/practices/screens_support.html



- 초대형 화면은 최소 960dp x 720dp
- 대형 화면은 최소 640dp x 480dp
- 보통 화면은 최소 470dp x 320dp
- 소형 화면은 최소 426dp x 320dp

2. 프래그먼트 실습

실습: 01_FragmentTablet

각 에뮬레이터(스마트폰, 태블릿)에서 실행

[Tools] > [Android] > [AVD Manager]에서

태블릿 카테고리에서 'Nexus7' 선택 후, (64비트) 오레오 등 설치

The first screenshot shows the 'Select Hardware' dialog in the AVD Manager. A red box highlights the 'Tablet' category, and another red box highlights the 'Nexus 7' device definition in the list.

Category	Name	Size	Resolution	Density
TV	Pixel C	9.94"	2560x1800	xhdpi
Wear	Nexus 9	8.86"	2048x1536	xhdpi
Phone	Nexus 7 (2012)	7.0"	800x1280	hdpi
Tablet	Nexus 7	7.02"	1200x1920	xhdpi
	Nexus 10	10.05"	2560x1600	xhdpi
	7" WSVGA (Tablet)	7.0"	600x1024	mdpi
	10.1" WXGA (Tablet)	10.1"	800x1280	mdpi

The second screenshot shows the 'Verify Configuration' dialog. The AVD Name is 'Nexus 7 API 25'. The device is 'Nexus 7' with resolution '7.02 1200x1920 xhdpi' and OS 'Nougat' with version 'Android 7.1.1 x86'. The 'Startup orientation' is set to 'Portrait'. The 'Emulated Performance' section shows 'Graphics: Automatic' and 'Multi-Core CPU' checked with '2' cores. The 'Device Frame' is checked. A red box highlights the 'Show Advanced Settings' button at the bottom.

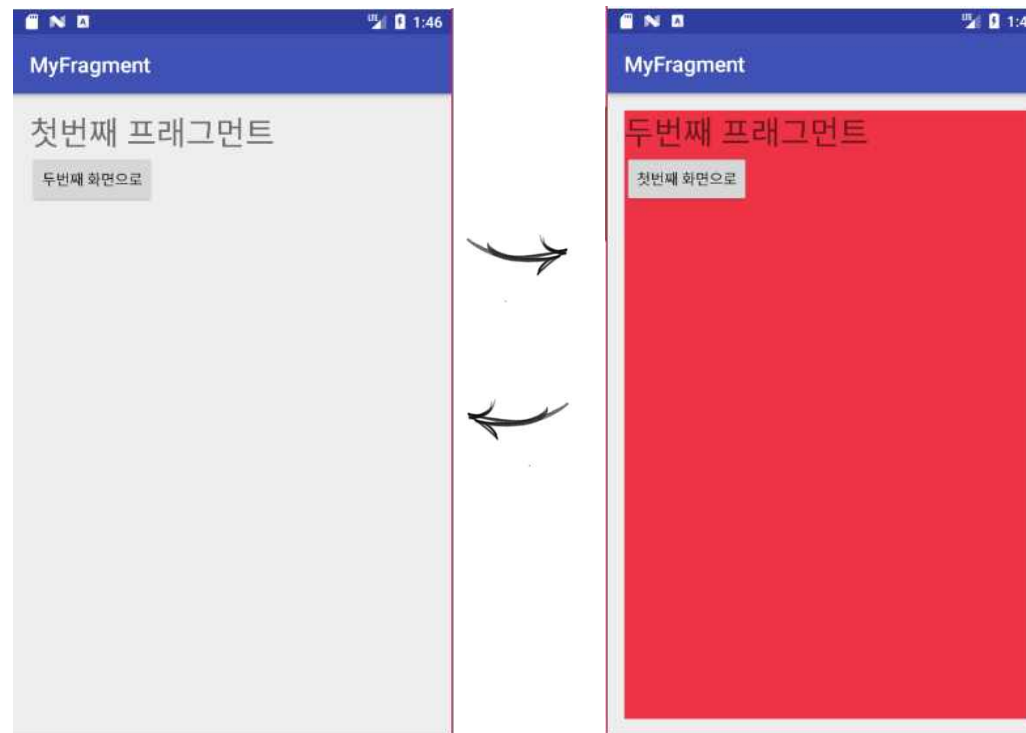
The third screenshot shows the 'Verify Configuration' dialog with advanced settings expanded. A red box highlights the 'RAM' field, which is set to '200' MB. Other settings include 'VM heap: 64 MB', 'Internal Storage: 800 MB', 'SD card: Studio-managed 100 MB', and 'Custom skin definition: nexus_7_2013'.

24

(옵션) 마지막 창의 [Show Advanced Setting]에서 RAM을 200MB로 할 것

2. 프래그먼트 실습

실습: 02_FragmentNext



2. 프래그먼트 실습

실습: 02_FragmentNext

activity_main.xml

ConstraintLayout이나 FrameLayout으로 작성할 것

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:id="@+id/container"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     android:background="#eee"
9     android:orientation="vertical"
10    android:padding="16dp">
11
12    <fragment
13        android:id="@+id/Fragment1"
14        android:name="com.example.kjs.myfragment.Fragment1"
15        android:layout_width="336dp"
16        android:layout_height="463dp"
17        app:layout_constraintLeft_toLeftOf="parent"
18        app:layout_constraintTop_toTopOf="parent"
19        tools:layout_editor_absoluteX="8dp"
20        tools:layout_editor_absoluteY="8dp" />
21
22 </android.support.constraint.ConstraintLayout>
```

fragment를 container로 사용할
레이아웃 안에 배치

2. 프래그먼트 실습

실습: 02_FragmentNext

fragment1.xml

리소스로 추가하여 디자인

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      android:orientation="vertical"
4      android:background="#eee"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent">
7
8      <TextView
9          android:text="첫번째 프래그먼트"
10         android:textSize="30sp"
11         android:layout_width="wrap_content"
12         android:layout_height="wrap_content" />
13
14     <Button
15         android:id="@+id/button"
16         android:text="두번째 화면으로"
17         android:layout_width="wrap_content"
18         android:layout_height="wrap_content" />
19 </LinearLayout>
```

2. 프래그먼트 실습

실습: 02_FragmentNext fragment2.xml

이전과 동일한 방법으로 리소스로 추가하여 디자인

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent"
5      android:background="#e34"
6      android:orientation="vertical">
7
8      <TextView
9          android:layout_width="wrap_content"
10         android:layout_height="wrap_content"
11         android:text="두번째 프래그먼트"
12         android:textSize="30sp" />
13
14     <Button
15         android:id="@+id/button"
16         android:layout_width="wrap_content"
17         android:layout_height="wrap_content"
18         android:text="첫번째 화면으로" />
19 </LinearLayout>
```

2. 프래그먼트 실습

실습: 02_FragmentNext Fragment1.java

```
15 public class Fragment1 extends Fragment{
16
17     @Nullable
18     @Override
19     public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
20         //화면 inflate
21         View view = inflater.inflate(R.layout.fragment1, container, false);
22
23         //버튼 이벤트정의
24         Button button = (Button)view.findViewById(R.id.button);
25         button.setOnClickListener(new View.OnClickListener(){
26             @Override
27             public void onClick(View v) {
28                 //fragment 를 호출한 activity 얻기
29                 MainActivity activity = (MainActivity)getActivity();
30                 activity.onFragmentChanged(0);
31             }
32         });
33         return view;
34     }
35 }
```

android.support.v4.app를 import

1) Fragment를 호출한 activity를 얻어
해당 activity상에 선언된 메소드 호출

2. 프래그먼트 실습

실습: 02_FragmentNext Fragment2.java

```
15 public class Fragment2 extends Fragment{
16
17     @Nullable
18     @Override
19     public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
20
21         View view = inflater.inflate(R.layout.fragment2, container, false);
22
23         Button button = (Button)view.findViewById(R.id.button);
24         button.setOnClickListener(new View.OnClickListener(){
25             @Override
26             public void onClick(View v) {
27                 MainActivity activity = (MainActivity)getActivity();
28                 activity.onFragmentChanged(1);
29             }
30         });
31         return view;
32     }
33 }
```

2. 프래그먼트 실습

실습: 02_FragmentNext
MainActivity.java

```
6 public class MainActivity extends AppCompatActivity {
7
8     Fragment1 fragment1;
9     Fragment2 fragment2;
10
11
12     @Override
13     protected void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15         setContentView(R.layout.activity_main);
16
17         fragment1 = new Fragment1();
18         fragment2 = new Fragment2();
19     }
20
21     //fragment 교체
22     public void onFragmentChanged(int index) {
23         if (index == 0) {
24             getSupportFragmentManager().beginTransaction().replace(R.id.container, fragment2).commit();
25         } else if (index == 1) {
26             getSupportFragmentManager().beginTransaction().replace(R.id.container, fragment1).commit();
27         }
28     }
29 }
```

2) 액티비티에서
FragmentManager의
트랜잭션으로 프래그먼트를 replace

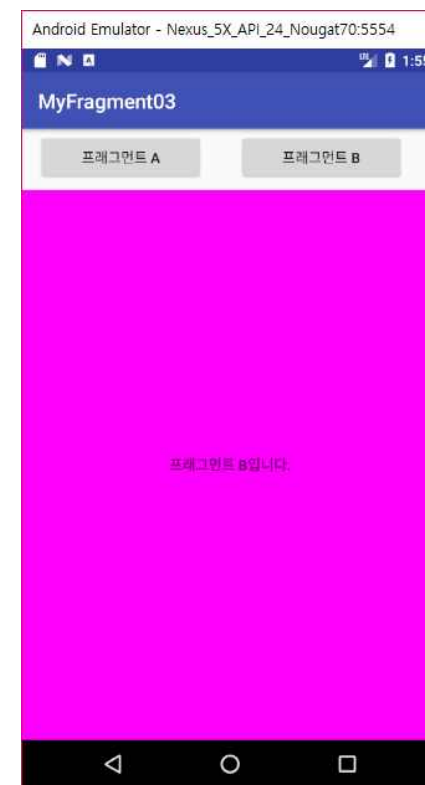
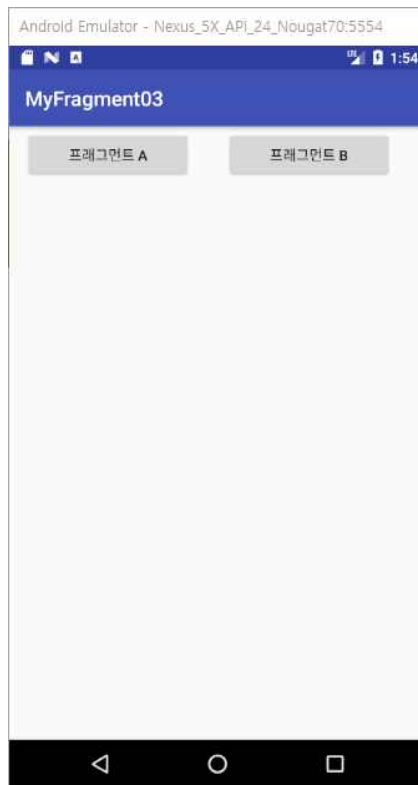
31* **addToBackStack()**: 화면에 안 보이게 되는 순간 제거하지 않고 스택에 저장했다가 다시 이용할 수 있도록 함

2. 프래그먼트 실습

실습: 03_FragBtn

버튼 클릭시 해당 프래그먼트 출력 > 완성해보자!

activity_main.xml에서 각 버튼에 onClick 속성으로 메소드를 추가하여 해결하기



참고문헌

정재곤(2018). Do it! 안드로이드 앱 프로그래밍(개정 5판), 이지스퍼블리싱.
천인국(2016). 그림으로 쉽게 설명하는 안드로이드 프로그래밍, 생능출판사.

참고 자료

<https://academy.realm.io/kr/posts/michael-yotive-state-of-fragments-2017/>