

S H O U T · O U R · P A S S I O N · T O G E T H E R

ODo IT SOPT O

안드로이드 3차 세미나

SHOUT OUR PASSION TOGETHER
SOPT

01 FragmentStatePagerAdapter

02 RecyclerView



01

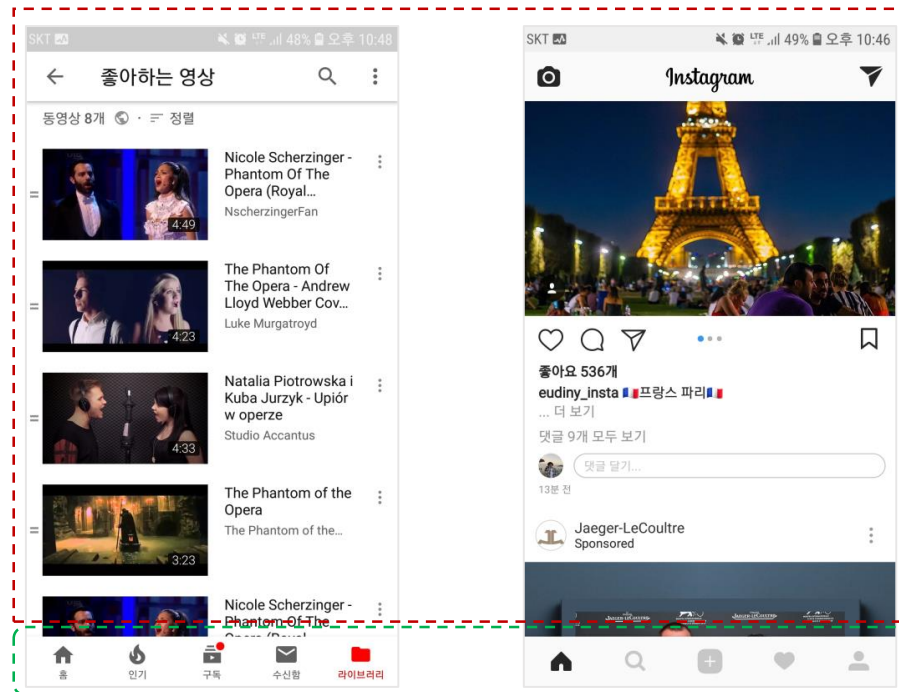


FragmentStatPagerAdapter

FragmentStatePagerAdapter란?

1. 일단, 안드로이드에서 Pager라는 단어가 View에 들어가면 책장을 넘기듯 화면을 slide 시키는 View라고 생각하면 됩니다!
2. FragmentStatePagerAdapter는 slide되는 view가 Fragment인 pager에서 Fragment를 관리해주는 Adapter입니다!!! 말이 어렵죠?
➔ 그냥 우리는 Fragment가 슬라이드 되는 View를 만들 것인데 그것을 관리해주는 겁니다.
3. FragmentPagerAdapter란 것도 있는데, 이것은 고정된 개수의 Fragment에 적합한 Adapter입니다!
한번 생성되면 Fragment들이 FragmentManager(아까 배웠죠?)에 박제되어 Activity가 종료되지 않는 한 제거되지 않아요! 그래서 Fragment의 개수가 많아지면 메모리 누수가 발생할 수 있어요!
4. **FragmentStatePagerAdapter**는 범위를 지정할 수 있는데, 범위 밖의 Fragment는 FragmentManager에서 지워주고 Adapter 내부에 저장시켜 둔 뒤 범위 안에 들어왔을 때 재생성 시키고 상태가 복원됩니다!
즉, Adapter가 유연하게 Fragment를 관리해줍니다!

FragmentStatePagerAdapter 사용 예



탭을 누를 때,
Activity 교체 없이
빨간 박스 UI만 변화한다.

1. 이런 UI를 구성했을 때, Fragment 상태를 보존 시켜 불필요한 서버 통신을 줄일 수 있다.
2. 앱을 많이 터뜨려봐야 이게 얼마나 용이한지 아는데, 우리는 고생하지 말고 미리 알고 가는 걸로!
3. TabLayout과 콜라보를 이뤄서 예쁜 탭 구성 가능

실습 0-1) 일단 Fragment 3개를 만들어봅시다!

MainFragment.kt / MapFragment.kt / MyPageFragment.kt

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <TextView
            android:text="I'm Main Fragment"
            android:layout_centerInParent="true"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        </RelativeLayout>
    </RelativeLayout>
```

<fragment_main.xml>

```
class MainFragment : Fragment(){
    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        return inflater.inflate(R.layout.fragment_main, container, false)
    }
}
```

<MainFragment.kt>

나머지 두 Fragment는 복붙을 통해 5분 안에 다 만들어 봅시다!!!

실습 0-2) 그 후 FragmentStatePagerAdapter를 만들어 봅시다!

```
import android.support.v4.app.Fragment
import android.support.v4.app.FragmentManager
import android.support.v4.app.FragmentStatePagerAdapter

class MyFragmentStatePagerAdapter(fm : FragmentManager, val fragmentCount : Int): FragmentStatePagerAdapter(fm){
    override fun getItem(position: Int): Fragment? {
        when(position){
            0 -> return MainFragment()
            1 -> return MapFragment()
            2 -> return MyPageFragment()
            else -> return null
        }
    }
    override fun getCount(): Int = fragmentCount
}
```

<MyFragmentStatePagerAdapter.kt>

어댑터는 매우 간단히 생성할 수 있지만,
그 외 작업이 쪼~금 귀찮아요!

01 FragmentStatePagerAdapter

실습 1-1) BottomNaviActivity를 만들고, view는 아래와 같이 구성해주세요!

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".BottomNaviActivity">

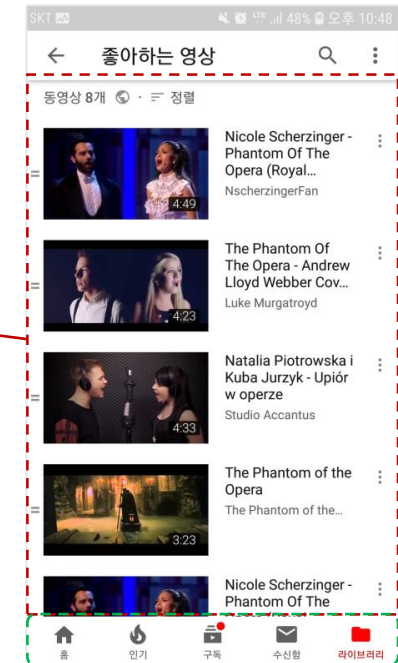
    <android.support.v4.view.ViewPager
        android:id="@+id/vp_bottom_navi_act_frag_pager"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_above="@+id/tl_bottom_navi_act_bottom_menu">

    </android.support.v4.view.ViewPager>

    <android.support.design.widget.TabLayout
        android:id="@+id/tl_bottom_navi_act_bottom_menu"
        android:background="#FFFFFF"
        android:elevation="5dp"
        app:tabIndicatorColor="#40D39F"
        android:layout_width="match_parent"
        android:layout_height="56dp"
        android:layout_alignParentBottom="true">

    </android.support.design.widget.TabLayout>

</RelativeLayout>
```



<activity_bottom_navi.xml>

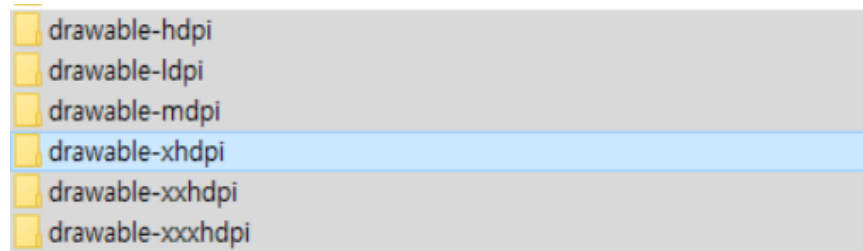
만약 TabLayout이 없다면 com.android.support.design 라이브러리가 주입되지 않은 것!


elevation은 높이를 줌으로 그림자가 약간 보이도록 하는 옵션인데요, 반드시 배경 색이 있어야 정상 작동해요!



실습 1-2) TabLayout에 들어갈 Custom 메뉴 View를 만들어 볼 텐데, 그전에

1. 일단 이미지 소스를 다운받아 봅시다. (*디자인 파트 **최형운** 제공*)
2. **프로젝트 이름\app\src\main\res** 내에 아래 6개 폴더를 통채로 옮깁니다.



3. 탭을 누른 상태면 Green색, 누르지 않은 상태면 Gray색 아이콘을 적용시키기 위해 Selector를 만들어 줍니다.
res/drawable에 마우스 오른쪽을 누른 뒤  Drawable resource file 을 통해 **selector_bottom_navi_main_icon.xml** 이라는 파일을 만들어 준 뒤 내용물은 아래와 같이 작성합니다.

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_selected="true" android:drawable="@drawable/tab_main_icon_green"/>
    <item android:state_selected="false" android:drawable="@drawable/tab_main_icon_gray"/>
</selector>
```

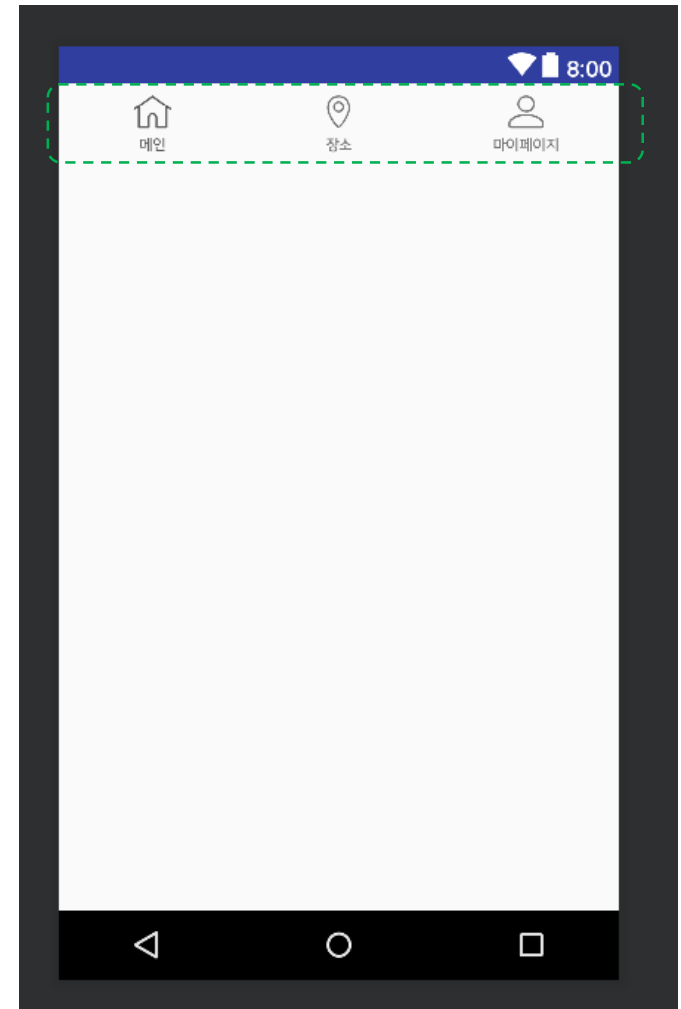
4. 같은 방법으로 **selector_bottom_navi_map_icon.xml**,
selector_bottom_navi_my_page_icon.xml을 만들어주세요.
(복붙을 잘 활용하면 빠르게 만들 수 있습니다!)

실습 1-3) 본격적으로 TabLayout에 들어갈 Custom 메뉴 View를 만들어 봅시다!

1. bottom_navigation_tab.xml이라는 layout폴더에 파일을 하나 만듭니다. 내용은 아래와 같습니다.

```
<?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="56dp"
    android:orientation="horizontal">
    <RelativeLayout
        android:id="@+id/btn_bottom_navi_main_tab"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1">
        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"
            android:src="@drawable/selector_bottom_navi_main_icon" />
    </RelativeLayout>
    <RelativeLayout
        android:id="@+id/btn_bottom_navi_map_tab"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1">
        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"
            android:src="@drawable/selector_bottom_navi_map_icon" />
    </RelativeLayout>
    <RelativeLayout
        android:id="@+id/btn_bottom_navi_my_page_tab"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1">
        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"
            android:src="@drawable/selector_bottom_navi_my_page_icon" />
    </RelativeLayout>
</LinearLayout>
```

반복...!
복붙 활용



01 FragmentStatePagerAdapter

실습 1-4) 방금 만든 뷰를 TabLayout에 달아봅시다!

1. BottomNaviActivity내에 아래 코드를 넣습니다.

```
class BottomNaviActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_bottom_navi)  
  
        configureBottomNavigation()  
    }  
  
    private fun configureBottomNavigation(){  
        vp_bottom_navi_act_frag_pager.adapter = MyFragmentStatePagerAdapter(supportFragmentManager, 3)  
        //vp_bottom_navi_act_frag_pager.offscreenPageLimit = 3  
  
        // ViewPager와 Tablayout을 엮어줍니다!  
        tl_bottom_navi_act_bottom_menu.setupWithViewPager(vp_bottom_navi_act_frag_pager)  
  
        //TabLayout에 붙일 layout을 찾아준 다음  
        val bottomNaviLayout : View = this.layoutInflater.inflate(R.layout.bottom_navigation_tab, null, false)  
  
        //탭 하나하나 TabLayout에 연결시켜줍니다.  
        tl_bottom_navi_act_bottom_menu.getTabAt(0)!!.customView = bottomNaviLayout.findViewById(R.id.btn_bottom_navi_main_tab) as RelativeLayout  
        tl_bottom_navi_act_bottom_menu.getTabAt(1)!!.customView = bottomNaviLayout.findViewById(R.id.btn_bottom_navi_map_tab) as RelativeLayout  
        tl_bottom_navi_act_bottom_menu.getTabAt(2)!!.customView = bottomNaviLayout.findViewById(R.id.btn_bottom_navi_my_page_tab) as RelativeLayout  
    }  
}
```

2. 끝!!! 이제 Run을 시켜보아요



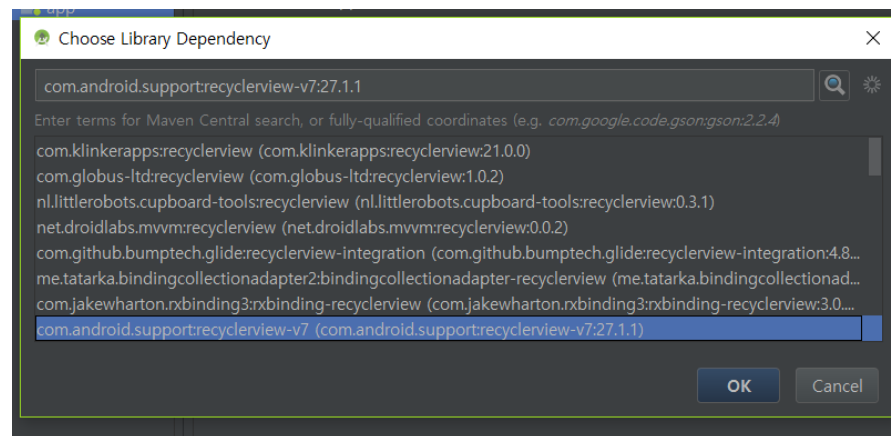
02



RecyclerView

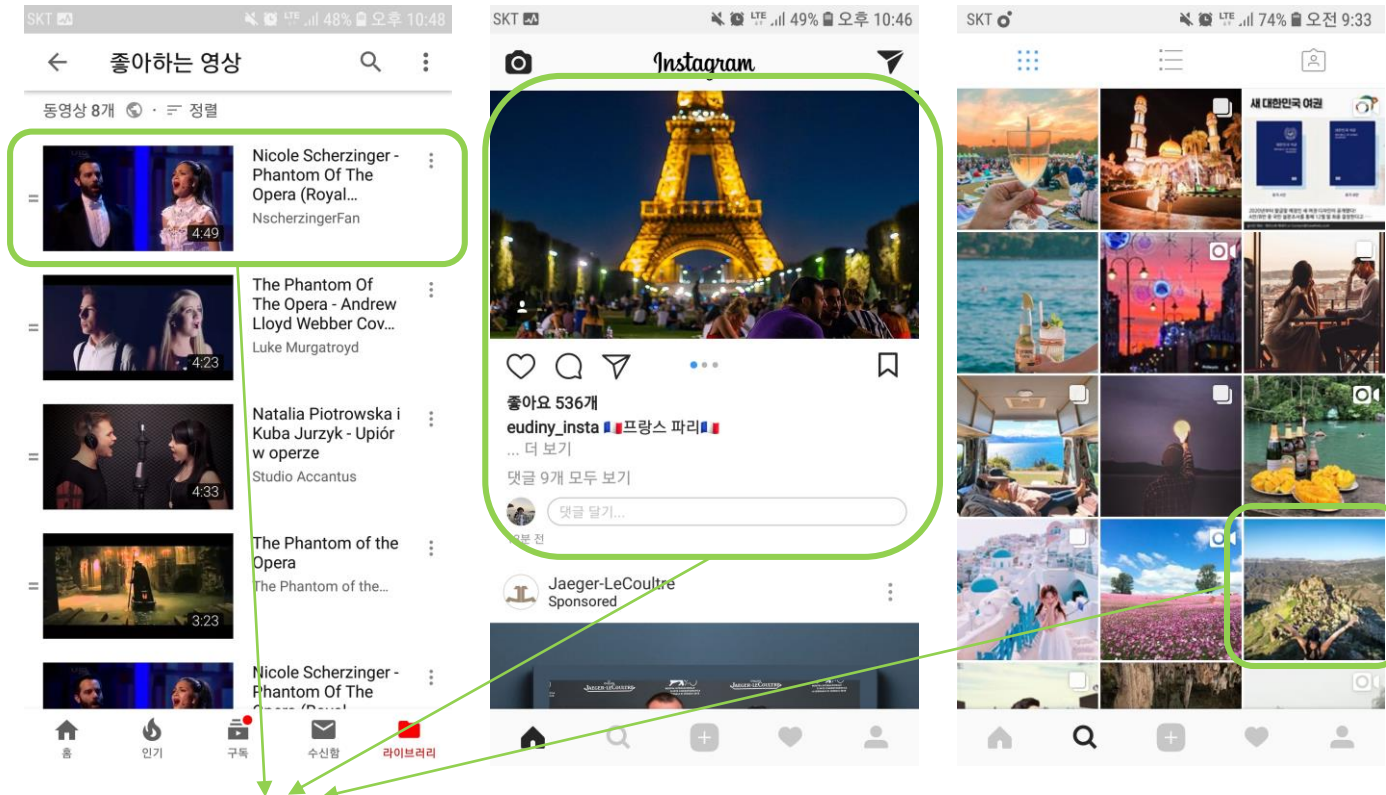
일단 이론에 들어가기 전 RecyclerView 라이브러리를 추가해줍니다!

- 안드로이드 스튜디오 메뉴를 통해 추가 검색어 : recyclerview



```
implementation 'com.android.support:recyclerview-v7:27.1.1'
```

RecyclerView란?



- 반복적인 View를 가진 **Item**들을 보여주는, **일종의 스크롤 되는 목록(List)**을 구현하기 위해 사용됩니다.
- 목록(List) UI가 없는 앱을 찾아보기 어려울 만큼 자주 쓰이는 핵심 위젯이므로 사용 방법을 반드시 익혀야 합니다.
- 이전에는 ListView라는 것이 있었는데 이것의 장점을 이어받고 단점을 보완한 것이 RecyclerView이므로, 목록(List)을 구현할 때는 RecyclerView를 사용하면 됩니다!

실습 전 RecyclerView를 위한 준비물 순서

1. **Data Class** → Item에 들어갈 Data들

2. 반복적인 View UI를 구성할 **xml 파일** → Item UI

3. **View Holder Class** →

ViewHolder pattern: findViewById() 호출에 대한 비용을 줄여줍니다.

들어가는 로직은, ViewHolder를 상속 받아서 Item UI에 있는 View들의 ID를 할당합니다!

4. **Recycler View Adapter Class** → View와 Data를 붙여주는 용도

RV 실습 1) Item에 들어갈 Data를 위한 **Data Class** 만들기



➔ 이미지, 제목, 내용, 인원, 시간 필요!

KakaoTalkRoomData.kr이란 파일을 만든 뒤 아래와 같은 코드를 구성해주세요!

```
data class KakaoTalkRoomData(var title : String, var content : String, var person_cnt : Int, var time : String)
```


02 RecyclerView

RV 실습 2) Item의 UI 구성하기!



[DoIT_SOPT] 안드로이드파트 53

어제

하자~~!!

- layout 디렉토리 마우스 오른쪽 클릭 후,
new → Layout resource file 클릭하여
파일명 rv_item_kakao_talk_room으로
xml파일을 생성해주세요!!
- 시간 관계 상 오른쪽 코드는 독방으로 보내
드리겠습니다!! 복붙하기

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="80dp">
    <ImageView
        android:id="@+id/iv_rv_item_kakao_talk_room_image"
        android:src="@drawable/ic_account_circle_black_48dp"
        android:layout_centerVertical="true"
        android:layout_marginLeft="16dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
    <RelativeLayout
        android:layout_centerVertical="true"
        android:layout_marginHorizontal="16dp"
        android:layout_toRightOf="@+id/iv_rv_item_kakao_talk_room_image"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <LinearLayout
            android:orientation="vertical"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <LinearLayout
                android:orientation="horizontal"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content">
                <TextView
                    android:id="@+id/tv_rv_item_kakao_talk_room_title"
                    android:textColor="#000000"
                    android:textSize="15dp"
                    android:text="[DoIT_SOPT]안드로이드 파트"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content" />
                <TextView
                    android:id="@+id/tv_rv_item_kakao_talk_room_person_cnt"
                    android:text="53"
                    android:textSize="12dp"
                    android:layout_marginLeft="4dp"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content" />
            </LinearLayout>
            <TextView
                android:id="@+id/tv_rv_item_kakao_talk_room_content"
                android:text="하자~~!!"
                android:textSize="14dp"
                android:layout_marginTop="2dp"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />
        </LinearLayout>
    </RelativeLayout>
    <TextView
        android:id="@+id/tv_rv_item_kakao_talk_room_time"
        android:text="어제"
        android:layout_marginRight="16dp"
        android:textSize="10dp"
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:paddingBottom="16dp"/>
</RelativeLayout>
```

RV 실습 3) 파일 생성 후 inner class로 ViewHolder 만들기

```
class KakaoTalkRoomRecyclerViewAdapter {  
  
    inner class Holder(itemView : View) : RecyclerView.ViewHolder(itemView){  
        val title : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_title) as TextView  
        val content : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_content) as TextView  
        val person_cnt : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_person_cnt) as TextView  
        val time : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_time) as TextView  
    }  
}
```

<KakaoTalkRoomRecyclerViewAdapter.kt>

- KakaoTalkRoomRecyclerViewAdapter 클래스 내부 클래스로 Holder 클래스를 만듭니다!
(내부 클래스가 아니고 따로 빼도 되며 본인 스타일대로 코딩하시면 됩니다!)
- 이전 장에서 만든 Item UI의 View들을 ID를 통해 찾아서 인스턴스 변수로 만들어 줍니다!

RV 실습 4) 본격적으로 Adapter를 통해 View와 Data를 연결시키는 작업을 해봅시다!

```
class KakaoTalkRoomRecyclerViewAdapter(val ctx : Context, val dataList : ArrayList<KakaoTalkRoomData>) : RecyclerView.Adapter<KakaoTalkRoomRecyclerViewAdapter.Holder>() {
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): Holder {
        //뷰 인플레이트!!
        val view : View = LayoutInflater.from(ctx).inflate(R.layout.rv_item_kakao_talk_room, parent, false)
        return Holder(view)
    }

    override fun getItemCount(): Int = dataList.size

    override fun onBindViewHolder(holder: Holder, position: Int) {
        //뷰 바인딩!!
        holder.title.text = dataList[position].title
        holder.content.text = dataList[position].content
        holder.person_cnt.text = dataList[position].person_cnt.toString()
        holder.time.text = dataList[position].time
    }

    inner class Holder(itemView : View) : RecyclerView.ViewHolder(itemView){
        val title : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_title) as TextView
        val content : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_content) as TextView
        val person_cnt : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_person_cnt) as TextView
        val time : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_time) as TextView
    }
}
```

<KakaoTalkRoomRecyclerViewAdapter.kt>

- 여기는 설명과 동시에 코딩하면서 설명하겠습니다!! → 설명 끝난 후 직접 해보기

RV 실습 5) 드디어 마지막!!! Adapter를 연결 시켜봐요! (MainFragment에 RecyclerView를 달아보겠습니다!)

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">
    <android.support.v7.widget.RecyclerView
        android:id="@+id/rv_main_frag_kakao_talk_room_list"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
    </android.support.v7.widget.RecyclerView>
</RelativeLayout>
```

<fragment_main.xml>

- 일단 기존에 만들었던 레이아웃 뷰를 다 지우고 위와 같이 작성해주세요! recyclerView 위젯 달기!

RV 실습 6) MainFragment에서 앞서 작업한 것들 조립하는 프로그래밍 시작!

```
class MainFragment : Fragment() {
    lateinit var kakaoTalkRoomRecyclerViewAdapter: KakaoTalkRoomRecyclerViewAdapter

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        return inflater.inflate(R.layout.fragment_main, container, false)
    }

    override fun onActivityCreated(savedInstanceState: Bundle?) {
        super.onActivityCreated(savedInstanceState)

        setRecyclerView()
    }

    private fun setRecyclerView() {
        //임시 데이터
        var dataList: ArrayList<KakaoTalkRoomData> = ArrayList()
        dataList.add(KakaoTalkRoomData("[DoIT_SOPT] 안드로이드파트", "인터뷰 하자~!!!", 53, "오후 6:53"))
        dataList.add(KakaoTalkRoomData("[DoIT_SOPT] iOS파트", "승수!!!", 36, "오후 4:43"))
        dataList.add(KakaoTalkRoomData("[DoIT_SOPT] 서버파트", "배다슬!!!", 55, "오후 3:03"))
        dataList.add(KakaoTalkRoomData("[DoIT_SOPT] 기획파트", "나성수!!!", 42, "오후 2:33"))
        dataList.add(KakaoTalkRoomData("[DoIT_SOPT] 디자인파트", "승미;;;", 39, "오후 1:13"))
        dataList.add(KakaoTalkRoomData("[DoIT_SOPT] 23대 윤영진", "회의요!", 10, "오전 5:53"))
        dataList.add(KakaoTalkRoomData("23기 버디버디조", "코다차야?!", 10, "오전 6:53"))
        dataList.add(KakaoTalkRoomData("23기 상반기 엠티조", "디제잉 시작한다!!!", 10, "오전 7:53"))

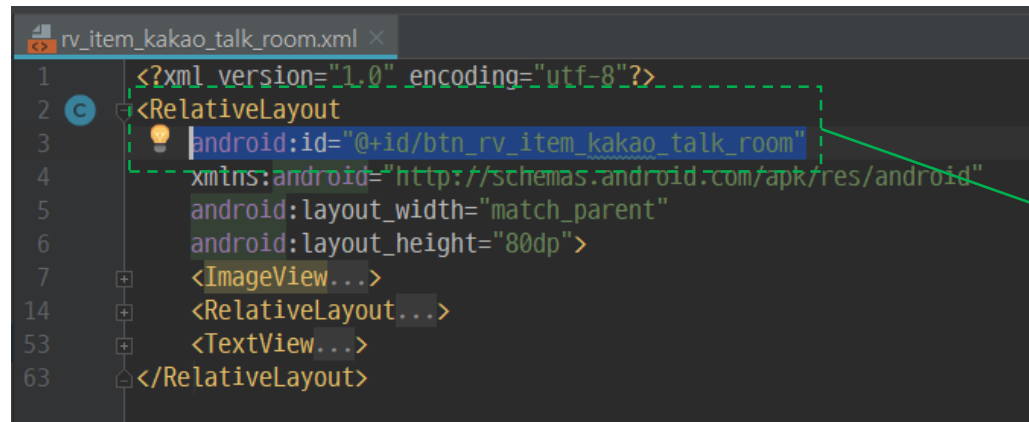
        kakaoTalkRoomRecyclerViewAdapter = KakaoTalkRoomRecyclerViewAdapter(activity!!, dataList)
        rv_main_frag_kakao_talk_room_list.adapter = kakaoTalkRoomRecyclerViewAdapter
        rv_main_frag_kakao_talk_room_list.layoutManager = LinearLayoutManager(activity)
    }
}
```

<MainFragment.kt>

layoutManager!!!!!! ppt뒷장에서 다시 쓰여요!

RV 실습 7) 진짜 마지막! 아이템 클릭 리스너 달기! - id설정

- 앞서 작업한 Item UI의 가장 안쪽(root) View Group에 id를 줍니다!



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout
3     android:id="@+id/btn_rv_item_kakao_talk_room"
4     xmlns:android="http://schemas.android.com/apk/res/android"
5     android:layout_width="match_parent"
6     android:layout_height="80dp">
7     <ImageView...>
14    <RelativeLayout...>
53    <TextView...>
63 </RelativeLayout>
```

root View Group인
RelativeLayout에
btn_rv_item_kakao_talk_room
라는 id를 주었습니다!

<rv_item_kakao_talk_room.xml>

RV 실습 7-2) 진짜 마지막! 아이템 클릭 리스너 달기! – 리스너 달기

- Holder에서 RelativeLayout의 ID를 잡아주고, onBindViewHolder에서 리스너를 달아주면 끝!

```
class KakaoTalkRoomRecyclerViewAdapter(val ctx : Context, val dataList : ArrayList<KakaoTalkRoomData>) :
    RecyclerView.Adapter<KakaoTalkRoomRecyclerViewAdapter.Holder>() {
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): Holder {
        //뷰 인플레이트!!
        val view : View = LayoutInflater.from(ctx).inflate(R.layout.rv_item_kakao_talk_room, parent, false)
        return Holder(view)
    }

    override fun getItemCount(): Int = dataList.size

    override fun onBindViewHolder(holder: Holder, position: Int) {
        //뷰 바인딩!!
        holder.title.text = dataList[position].title
        holder.content.text = dataList[position].content
        holder.person_cnt.text = dataList[position].person_cnt.toString()
        holder.time.text = dataList[position].time

        holder.item_btn.setOnClickListener {
            ctx.toast("메인 액티비티로~")
            ctx.startActivity<MainActivity>()
        }
    }

    inner class Holder(itemView : View) : RecyclerView.ViewHolder(itemView){
        val title : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_title) as TextView
        val content : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_content) as TextView
        val person_cnt : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_person_cnt) as TextView
        val time : TextView = itemView.findViewById(R.id.tv_rv_item_kakao_talk_room_time) as TextView

        val item_btn : RelativeLayout = itemView.findViewById(R.id.btn_rv_item_kakao_talk_room) as RelativeLayout
    }
}
```

<KakaoTalkRoomRecyclerViewAdapter.kt>

More) 다른 형태의 List를 만들고 싶다면?! LayoutManager!!!

- layoutManager를 통해 다양한 형태로 Item들을 배치할 수 있습니다! 실습 코드 기준으로 설명해드릴게요!

첫번째, **수직 스크롤 형태**: default가 수직 스크롤이므로 둘 중 아무거나 써도 됩니다!

```
rv_main_frag_kakao_talk_room_list.layoutManager = LinearLayoutManager(activity)
```

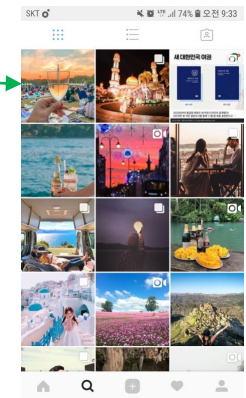
```
rv_main_frag_kakao_talk_room_list.layoutManager = LinearLayoutManager(activity, LinearLayoutManager.VERTICAL, false)
```

두번째, **수평 스크롤 형태**

```
rv_main_frag_kakao_talk_room_list.layoutManager = LinearLayoutManager(activity, LinearLayoutManager.HORIZONTAL, false)
```

세번째, **그리드 형태**: 두번째 매개변수는 한 줄에 몇 개의 Item이 보일지에 대한 값

```
rv_main_frag_kakao_talk_room_list.layoutManager = GridLayoutManager(activity, 3)
```



네번째, 불규칙 그리드 형태: 울퉁불퉁하게 지그재그인 그리드 형태!

```
rv_main_frag_kakao_talk_room_list.layoutManager = StaggeredGridLayoutManager(2, StaggeredGridLayoutManager.VERTICAL)
```

매개변수는 컨트롤+P를 통해 무엇을 넣으면 되는지 알 수 있으므로 시간 날 때 사용해보세요~!

S H O U T · O U R · P A S S I O N · T O G E T H E R

과제

과 제

과제3) Instagram 그리드 형태 목록을 만들어 봅시다!



- 첫번째 조건: 아래 데이터 클래스로 구현하기
(isLike가 true일 때 Item의 우측 상단에 하트 보이도록)

```
data class MyItemData(  
    val counter : Int,  
    val isLike : Boolean  
)
```

- 두번째 조건: Item 클릭 시 해당되는 숫자 toast 메시지로 띄우기

- 힌트) 하트 숨기는 방법, Adapter class에서 해당 로직 구현

```
if (!dataList[position].islike){  
    holder.like_img.visibility = View.GONE  
}
```

S H O U T · O U R · P A S S I O N · T O G E T H E R

ODo IT SOPT O

THANK U

PPT 디자인 한승미 세미나 자료 남윤환

SHOUT OUR PASSION TOGETHER
SOPT