



Do It! 안드로이드 앱 프로그래밍

개정6판

안드로이드 스튜디오 3.3 이상 기준

Mar. 2019

저자 : 정재곤



Do It! 안드로이드 앱 프로그래밍

둘째 마당 - Chapter 07

선택 위젯 만들기



이번 장에서는 무엇을 다룰까요?



아이콘이 들어간 리스트가 포함된 화면을 만들 때는 어떻게 하나요?



- 뷰를 직접 정의해볼까요?
- 리사이클러뷰가 포함된 화면을 만들어 볼까요?
- 콤보박스처럼 사용되는 스피너가 포함된 화면을 만들어 볼까요?





이번 장에서는 무엇을 다룰까요?



제공되는 뷰 말고 직접 뷰를 정의할 수도 있나요?

- 새로운 뷰 만들기



리스트 모양으로 보여주고 싶어요.

- 리사이클러뷰 사용하기



콤보박스 형태로 간단하게 보여줄 수 있나요?

- 스피너 사용하기

어댑터

아이템 #0

아이템 #1

아이템 #2





강의 주제

대표적인 선택 위젯의 이해와 실습



1

새로운 뷰 만들기

2

레이아웃 정의하고 카드뷰 넣기

3

리사이클러뷰 만들기

4

스피너 사용하기

1.

새로운 뷰 만들기



뷰 만들기

- 새로운 뷰를 직접 만들 수 있음

- 뷰를 상속하면 새로운 뷰를 정의할 수 있음
- 뷰의 크기를 결정할 수도 있고 뷰 위에 그래픽을 그릴 수도 있음

[Reference]

```
public void onMeasure (int widthMeasureSpec, int heightMeasureSpec)  
public void onDraw(Canvas canvas)
```

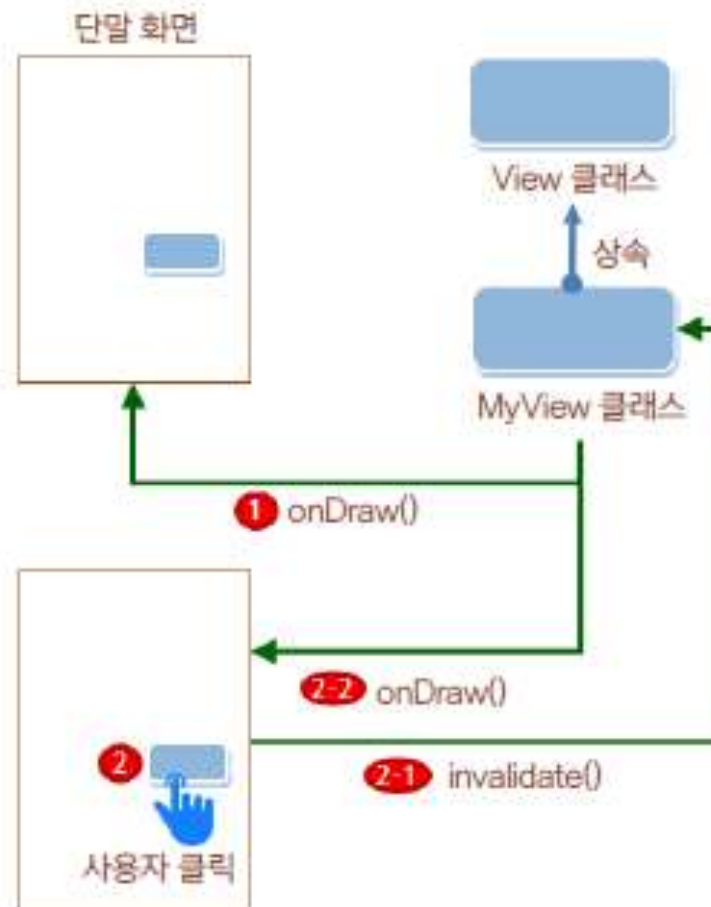
[Reference]

```
void setMeasuredDimension (int measuredWidth, int measuredHeight)
```



뷰 위에 그래픽을 그리는 과정

- 뷰에 그래픽이 그려질 때 onDraw() 메소드 호출됨
- 다시 그리기는 invalidate() 메소드 사용





버튼 만들기 예제

버튼 만들기 예제

-이미지를 보여주는 버튼

버튼을 상속한
새로운 클래스 정의

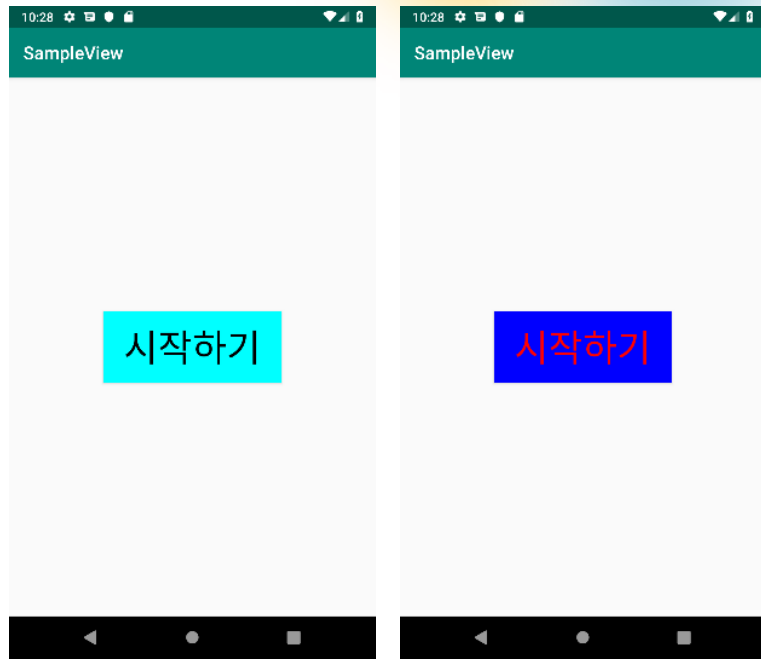
-새로운 버튼 클래스 정의

XML 레이아웃에 추가

-새로운 버튼 태그를 XML 레이아웃에 추가

메인 액티비티 코드 작성

-메인 액티비티 코드에서 참조하여 사용





새로운 버튼 클래스 정의

- 터치 이벤트에 따라 배경 이미지를 바꾸어주는 버튼 클래스 정의

참조파일 SampleView>/java/org.techtown.view/MyButton.java

```
public class MyButton extends AppCompatActivity { → ❶ AppCompatActivity 클래스 상속하여 새로운  
                                                    클래스 정의하기  
    public MyButton(Context context) {  
        super(context);  
        init(context);  
    }  
  
    public MyButton(Context context, AttributeSet attrs) {  
        super(context, attrs);  
        init(context);  
    }  
  
    private void init(Context context) {  
        setBackgroundColor(Color.CYAN);  
        setTextColor(Color.BLACK);  
  
        float textSize = getResources().getDimension(R.dimen.text_size);  
        setTextSize(textSize);  
    }  
}
```

❷ 초기화를 위한
메서드 정의하기



새로운 버튼 클래스 정의

- 터치 이벤트에 따라 배경 이미지를 바꾸어 줌

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    Log.d("MyButton", "onTouchEvent 호출됨");

    int action = event.getAction();
    switch (action) {
        case MotionEvent.ACTION_DOWN:
            setBackgroundColor(Color.BLUE);
            setTextColor(Color.RED);

            break;

        case MotionEvent.ACTION_OUTSIDE:
        case MotionEvent.ACTION_CANCEL:
        case MotionEvent.ACTION_UP:
            setBackgroundColor(Color.CYAN);
            setTextColor(Color.BLACK);

            break;
    }

    invalidate();

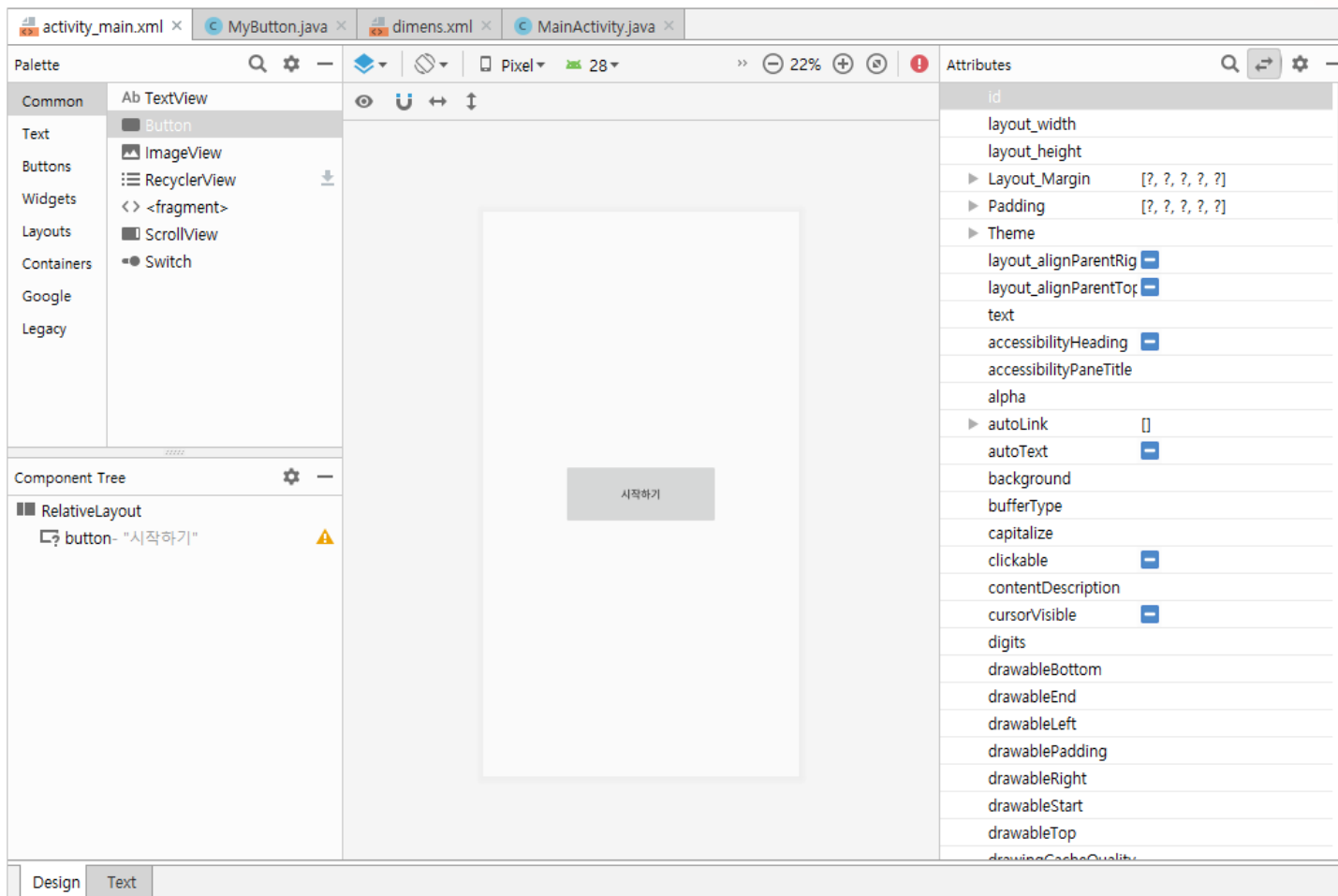
    return true;
}
```

② 뷰가 터치될 때 호출되는 함수에 기능 추가하기

종락...



레이아웃 만들기





레이아웃 만들기

참조파일 SampleView>/app/res/layout/activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
```

```
    <org.techtown.view.MyButton
        android:id="@+id/button"
        android:layout_width="200dp"
        android:layout_height="80dp"
        android:layout_centerInParent="true"
        android:text="시작하기"
    />
```

새로 만든 MyButton 클래스를 태그로 추가하기

```
</RelativeLayout>
```



```
public class MyButton extends AppCompatButton {  
    public MyButton(Context context) {  
        super(context);  
        init(context);  
    }  
    public MyButton(Context context, AttributeSet attrs) {  
        super(context, attrs);  
        init(context);  
    }  
  
    private void init(Context context) {  
        setBackgroundColor(Color.CYAN);  
        setTextColor(Color.BLACK);  
  
        float textSize = getResources().getDimension(R.dimen.text_size);  
        setTextColor(textSize);  
    }  
  
    @Override  
    protected void onDraw(Canvas canvas) {  
        super.onDraw(canvas);  
  
        Log.d("MyButton", "onDraw 호출됨");  
    }  
}
```

@Override

```
public boolean onTouchEvent(MotionEvent event) {
```

```
    Log.d("MyButton", "onTouchEvent 호출됨");
```

```
    int action = event.getAction();
```

```
    switch (action) {
```

```
        case MotionEvent.ACTION_DOWN:
```

```
            setBackgroundColor(Color.BLUE);
```

```
            setTextColor(Color.RED);
```

```
            break;
```

```
        case MotionEvent.ACTION_OUTSIDE:
```

```
        case MotionEvent.ACTION_CANCEL:
```

```
        case MotionEvent.ACTION_UP:
```

```
            setBackgroundColor(Color.CYAN);
```

```
            setTextColor(Color.BLACK);
```

```
            break;
```

```
    }
```

```
    invalidate();
```

```
    return true;
```

```
}
```

```
}
```




activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">
```

```
    <org.techtown.view.MyButton  
        android:id="@+id/button"  
        android:layout_width="200dp"  
        android:layout_height="80dp"  
        android:layout_centerInParent="true"  
        android:text="시작하기" />
```

```
</RelativeLayout>
```



values/dimens.xml

```
<?xml version="1.0" encoding="utf-8"?>  
<resources>  
    <dimen name="text_size">16sp</dimen>  
</resources>
```

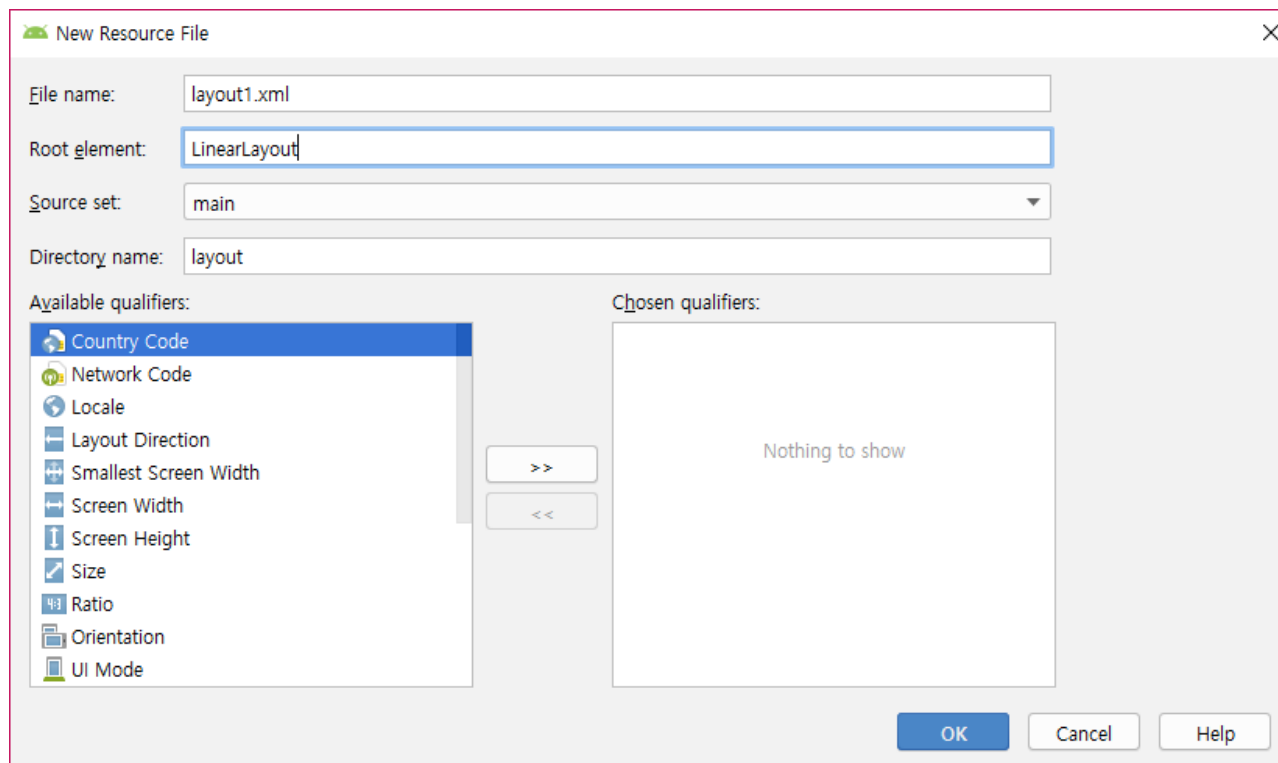
2.

레이아웃 정의하고 카드뷰 넣기



레이아웃 파일 추가

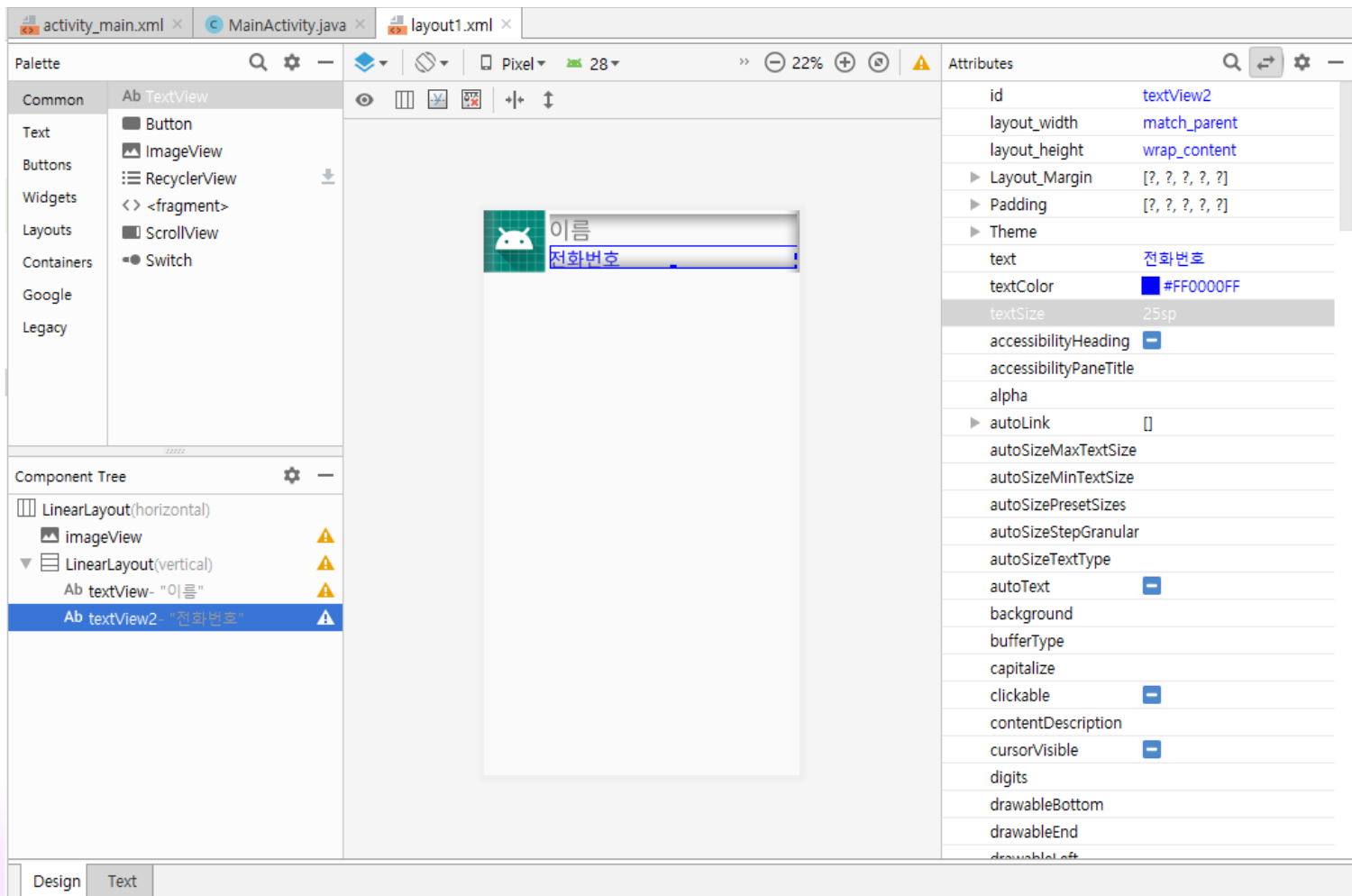
- /app/res/layout 폴더 안에 새로운 레이아웃 파일 만들기





레이아웃 파일 추가

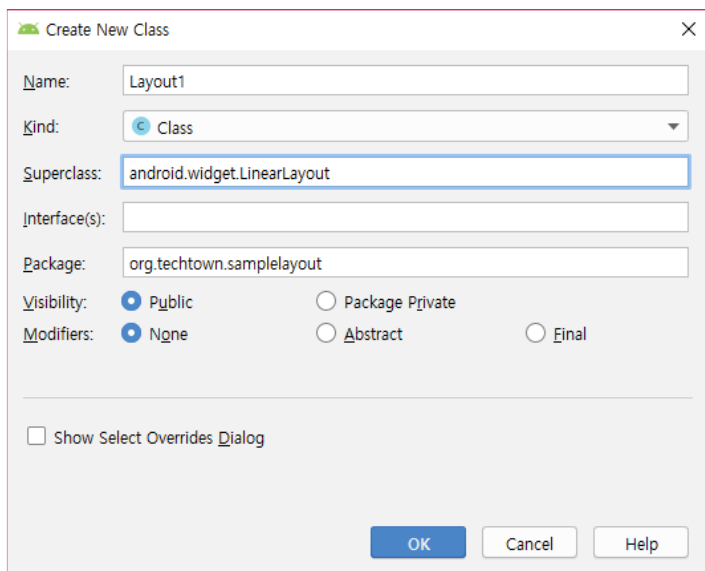
- LinearLayout 안에 하나의 이미지뷰와 텍스트뷰 두 개 추가하기



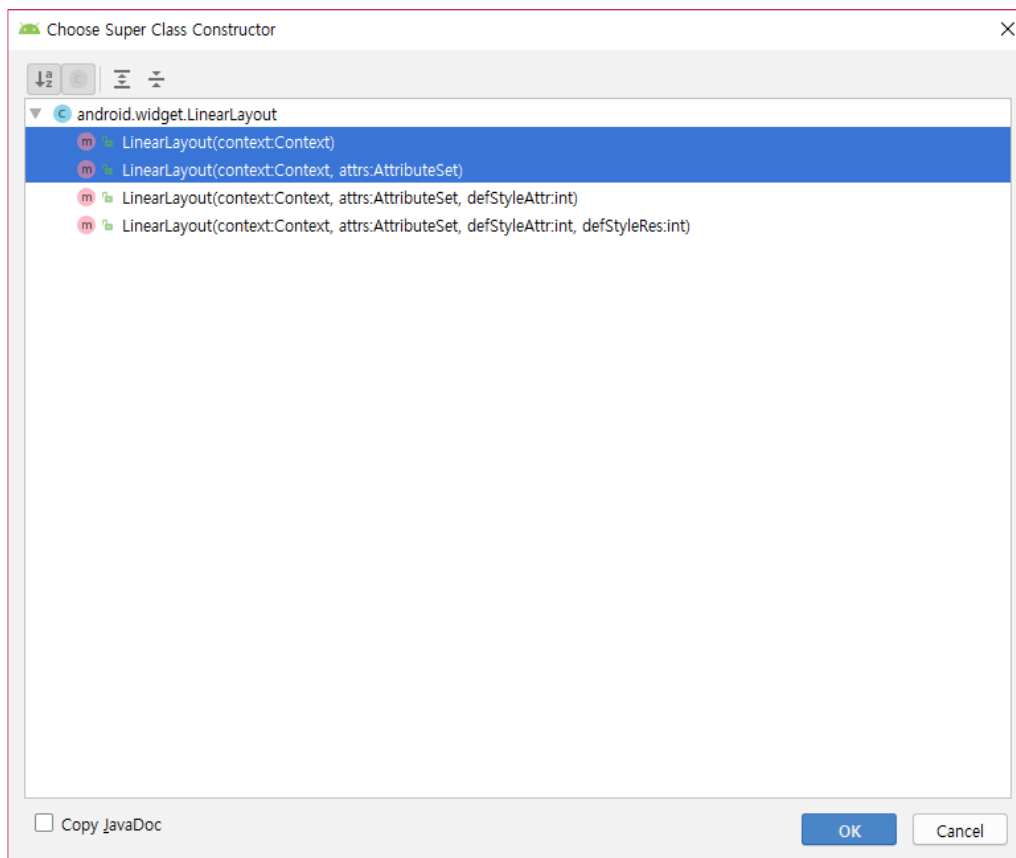


레이아웃 파일 추가

- 새로운 클래스 파일 추가하기
- 생성자는 2개 재정의



Create New Class dialog box. Fields: Name: Layout1, Kind: Class, Superclass: android.widget.LinearLayout, Interface(s):, Package: org.techtown.samplelayout. Visibility: Public (selected), Package Private, Abstract, Final. Modifiers: None (selected), Abstract, Final. Show Select Overrides Dialog: unchecked. Buttons: OK, Cancel, Help.



Choose Super Class Constructor dialog box. List of constructors for android.widget.LinearLayout: LinearLayout(context:Context), LinearLayout(context:Context, attrs:AttributeSet), LinearLayout(context:Context, attrs:AttributeSet, defStyleAttr:int), LinearLayout(context:Context, attrs:AttributeSet, defStyleAttr:int, defStyleRes:int). Buttons: OK, Cancel. Copy Javadoc: unchecked.



레이아웃 파일 추가

- 생성자 안에 인플레이션 진행하는 init 메서드 호출

참조파일 SampleLayout>/app/java/org.techtown.samplelayout/Layout1.java

```
public class Layout1 extends LinearLayout { —————> ❶ LinearLayout 클래스 상속하여 새로운 클래스
                                                         정의하기

    public Layout1(Context context) {
        super(context);
        init(context);
    }

    public Layout1(Context context, AttributeSet attrs) {
        super(context, attrs);
        init(context);
    }

    private void init(Context context) {
        LayoutInflater inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        inflater.inflate(R.layout.layout1, this, true);
    }
}
```

❷ 인플레이션 진행하기 ←



레이아웃 파일 추가

- XML 레이아웃 안에 들어있는 뷰 객체들을 찾아 변수에 할당

참조파일 SampleLayout>/app/java/org.techtown.samplelayout/Layout1.java

```
public class Layout1 extends LinearLayout {
    ImageView imageView;
    TextView textView;
    TextView textView2;
    종락...
    private void init(Context context) {
        LayoutInflater inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_
SERVICE);
        inflater.inflate(R.layout.layout1, this, true);

        imageView = findViewById(R.id.imageView);
        textView = findViewById(R.id.textView);
        textView2 = findViewById(R.id.textView2);
    }

    public void setImage(int resId) {
        imageView.setImageResource(resId);
    }

    public void setName(String name) {
        textView.setText(name);
    }

    public void setMobile(String mobile) {
        textView2.setText(mobile);
    }
}
```

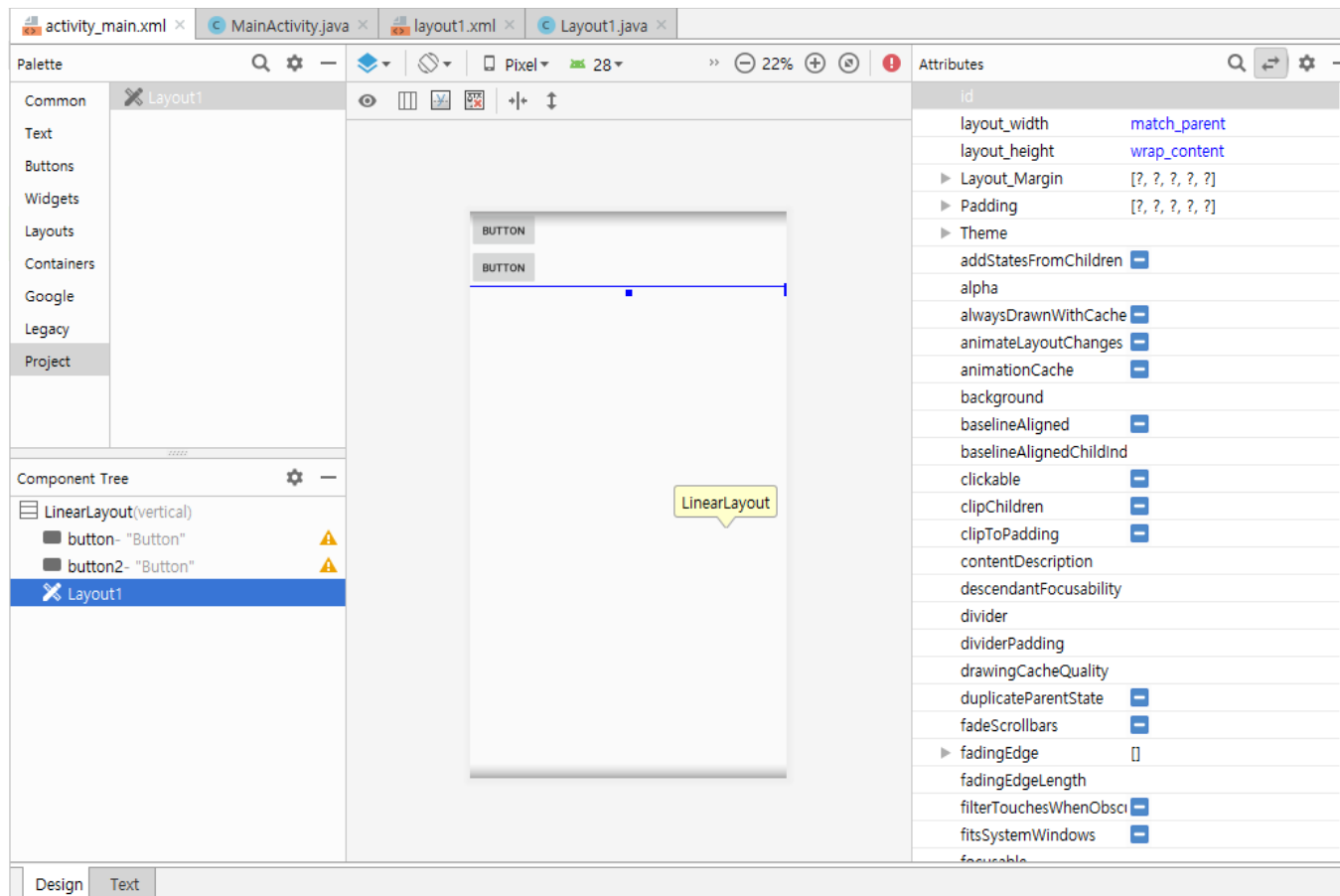
① XML 레이아웃에서 정의했던 뷰 참조하기

② 뷰에 데이터 설정하기



레이아웃 파일 추가

- activity_main.xml 파일에 새로 만든 뷰 추가하기





레이아웃 파일 추가

- 메인 액티비티 소스에서 데이터 설정

참조파일 SampleLayout>/app/java/org.techtown.samplelayout/MainActivity.java

```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        Layout1 layout1 = findViewById(R.id.layout1); —————→ ❶ XML 레이아웃에 추가한 뷰 참조하기  
  
        layout1.setImage(R.drawable.ic_launcher_foreground);  
        layout1.setName("김민수");  
        layout1.setMobile("010-1000-1000");  
    }  
}
```

❷ 뷰의 메서드 호출하여 데이터 설정하기



레이아웃 파일 추가

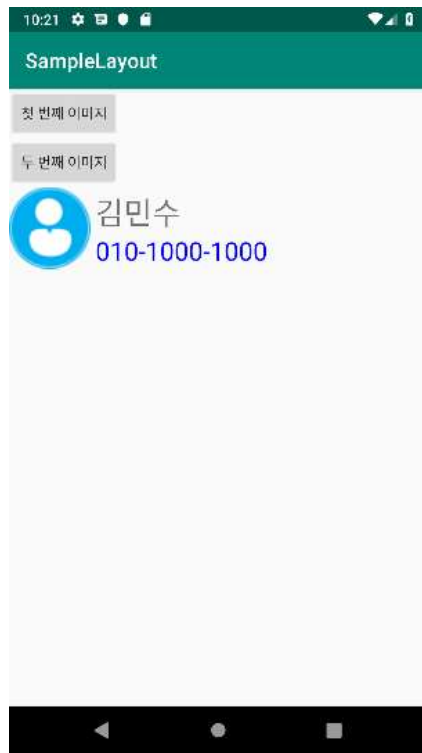
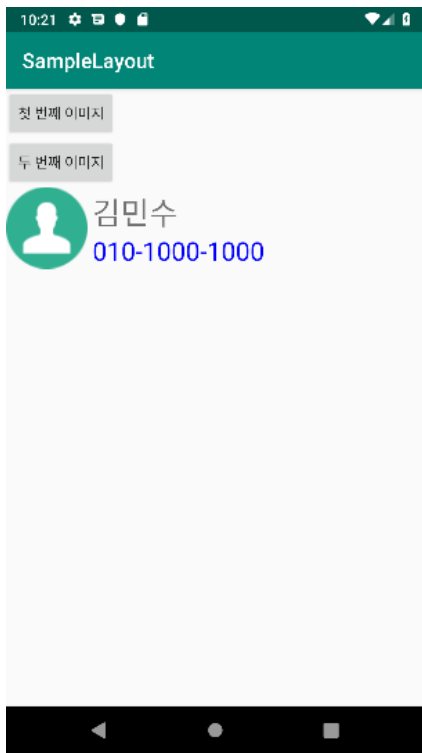
- 버튼 클릭 시의 이벤트 처리

```
Button button2 = findViewById(R.id.button2);
button2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        layout1.setImage(R.drawable.profile2);
    }
});
}
```



레이아웃 파일 추가

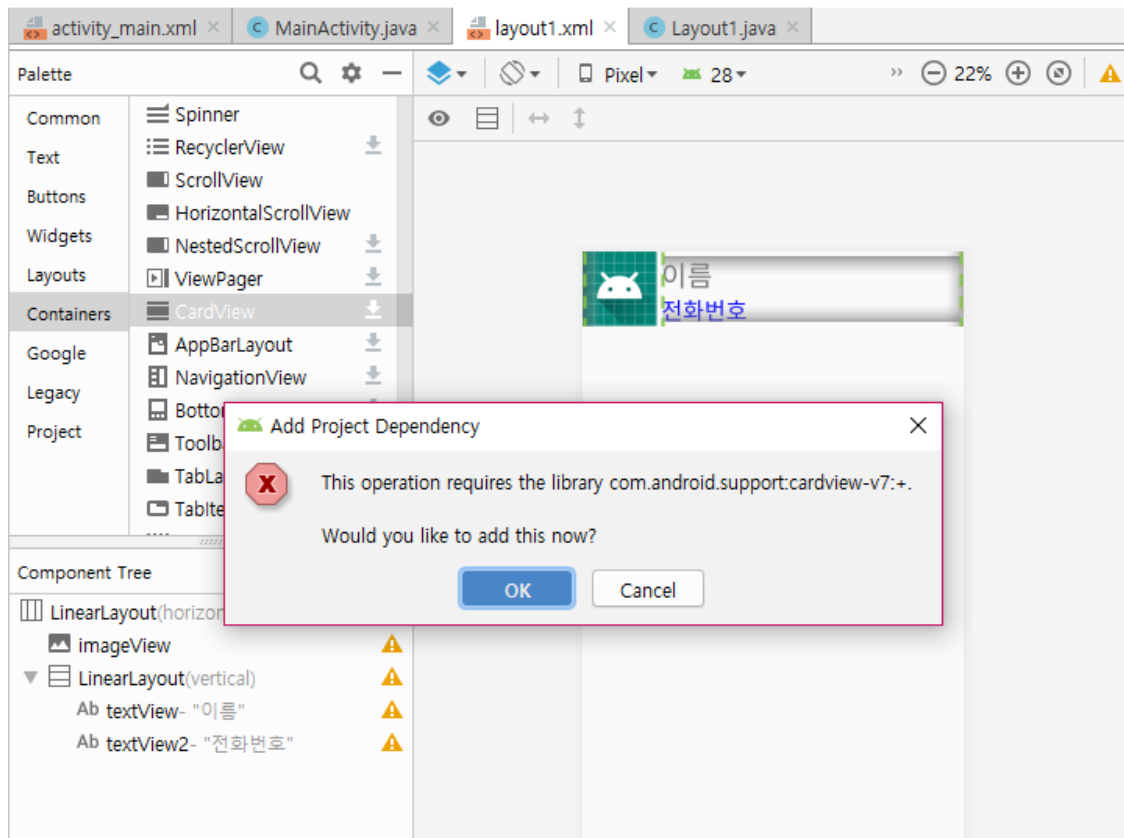
- 앱 실행





카드뷰 모양으로 바꾸기

- 카드뷰를 위한 외부 라이브러리 추가





카드뷰 모양으로 바꾸기

- 카드뷰 태그 추가

참조파일 SampleLayout>/app/res/layout/layout1.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

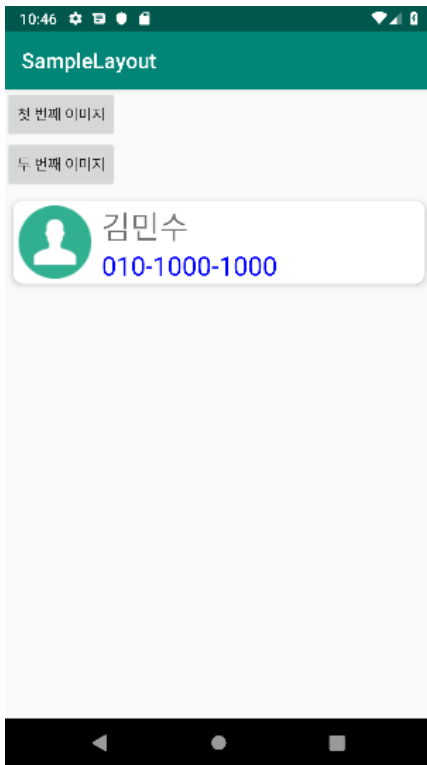
    <android.support.v7.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:cardBackgroundColor="#FFFFFF"
        app:cardCornerRadius="10dp"
        app:cardElevation="5dp"
        app:cardUseCompatPadding="true" >
```

CardView 태그 추가하기



카드뷰 모양으로 바꾸기

- 카드뷰 모양으로 바꾼 결과





layout1.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <androidx.cardview.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:cardBackgroundColor="#FFFFFF"
        app:cardCornerRadius="10dp"
        app:cardElevation="5dp"
        app:cardUseCompatPadding="true">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">
            <ImageView
                android:id="@+id/imageView"
                android:layout_width="80dp"
                android:layout_height="80dp"
                android:padding="5dp"
                app:srcCompat="@mipmap/ic_launcher" />
```




layout1.xml

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="5dp"
    android:layout_weight="1"
    android:orientation="vertical">
    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="이름"
        android:textSize="30sp" />
    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="전화번호"
        android:textColor="#FF0000FF"
        android:textSize="25sp" />
</LinearLayout>
</LinearLayout>
</androidx.cardview.widget.CardView>
</LinearLayout>
```



activity_main.xml

```
<LinearLayout 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"
    android:orientation="vertical"
    tools:context=".MainActivity" >
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="첫 번째 이미지" />
    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="두 번째 이미지" />
    <com.example.mycardview.Layout1
        android:id="@+id/layout1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

```
public class Layout1 extends LinearLayout {
```

```
    ImageView imageView;
```

```
    TextView textView;
```

```
    TextView textView2;
```

```
    public Layout1(Context context) {
```

```
        super(context);
```

```
        init(context);
```

```
    }
```

```
    public Layout1(Context context, AttributeSet attrs) {
```

```
        super(context, attrs);
```

```
        init(context);
```

```
    }
```

```
    private void init(Context context) {
```

```
        LayoutInflater inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
```

```
        inflater.inflate(R.layout.layout1, this, true);
```

```
        imageView = findViewById(R.id.imageView);
```

```
        textView = findViewById(R.id.textView);
```

```
        textView2 = findViewById(R.id.textView2);
```

```
    }
```

```
public void setImage(int resId) {  
    imageView.setImageResource(resId);  
}
```

```
public void setName(String name) {  
    textView.setText(name);  
}
```

```
public void setMobile(String mobile) {  
    textView2.setText(mobile);  
}
```

```
}
```

```
public class MainActivity extends AppCompatActivity {  
    Layout1 layout1;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        layout1 = findViewById(R.id.layout1);  
        layout1.setImage(R.drawable.profile1);  
        layout1.setName("김민수");  
        layout1.setMobile("010-1000-1000");  
  
        Button button = findViewById(R.id.button);  
        button.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                layout1.setImage(R.drawable.profile1);  
            }  
        });  
        Button button2 = findViewById(R.id.button2);  
        button2.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                layout1.setImage(R.drawable.profile2);  
            }  
        });  
    }  
}
```

3.

리싸이클러뷰 만들기



왜 굳이 선택위젯이라는 이름으로 구분할까?

- 안드로이드에서는 여러 아이템 중의 하나를 선택하는 선택위젯은 별도의 패턴을 사용함
- 여러 개의 아이템 중에서 하나를 선택하는 방식의 선택 위젯은 **어댑터**를 사용하여야 함
- 이 **어댑터에서 데이터를 관리**하도록 해야 할 뿐만 아니라 **화면에 보여지는 뷰도 어댑터의 getView() 메소드에서 결정함**
- 선택위젯의 가장 큰 특징은 **원본 데이터를 위젯에 직접 설정하지 않고 어댑터라는 클래스를 사용**하도록 되어 있다는 점으로 이 패턴을 잘 기억해 두어야 함



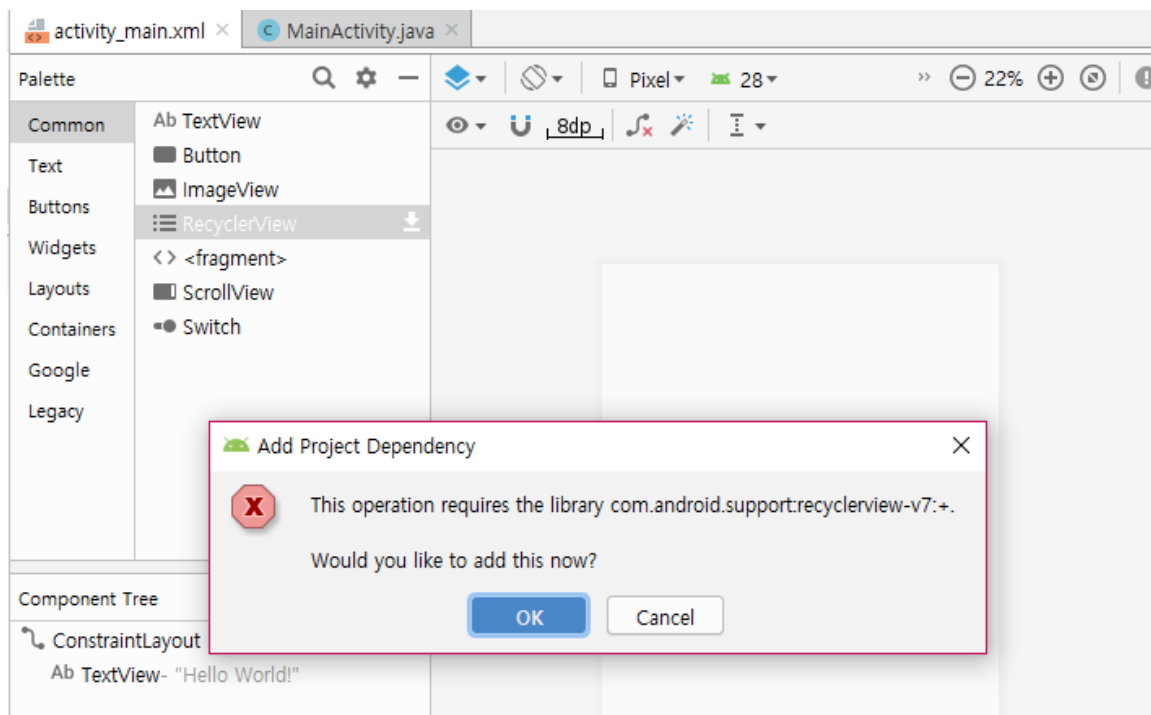
선택 위젯에 보이는 각각의 아이템이 화면에 디스플레이되기 전에 어댑터의 getView() 메서드가 호출됨
getView()메서드에서 반환하는 뷰가 하나의 아이템으로 디스플레이됨

[선택 위젯과 어댑터]



리사이클러뷰 만들기

- 팔레트에서 RecyclerView 오른쪽에 있는 아이콘 눌러 외부 라이브러리 추가





리사이클러뷰 만들기

- activity_main.xml 에 RecyclerView 태그 추가

참조파일 SampleRecyclerView>/app/res/layout/activity_main.xml

```
<?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:orientation="vertical"

    <android.support.v7.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
```

리사이클러뷰 태그 추가하기

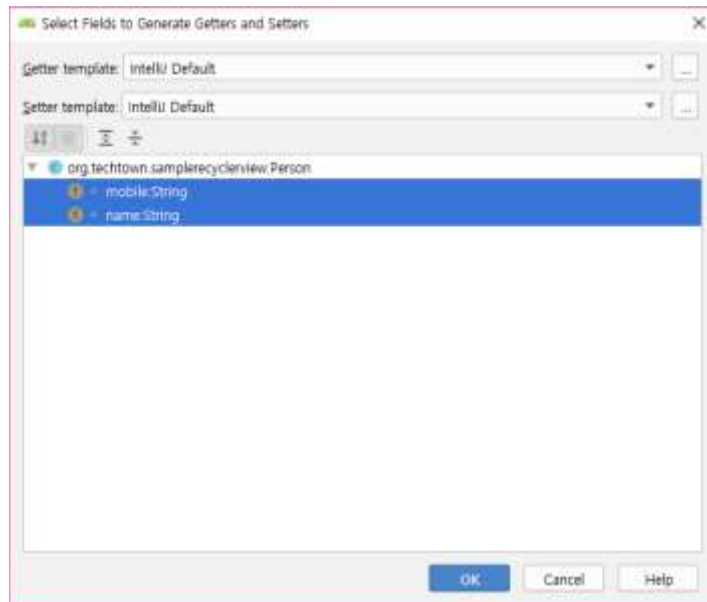
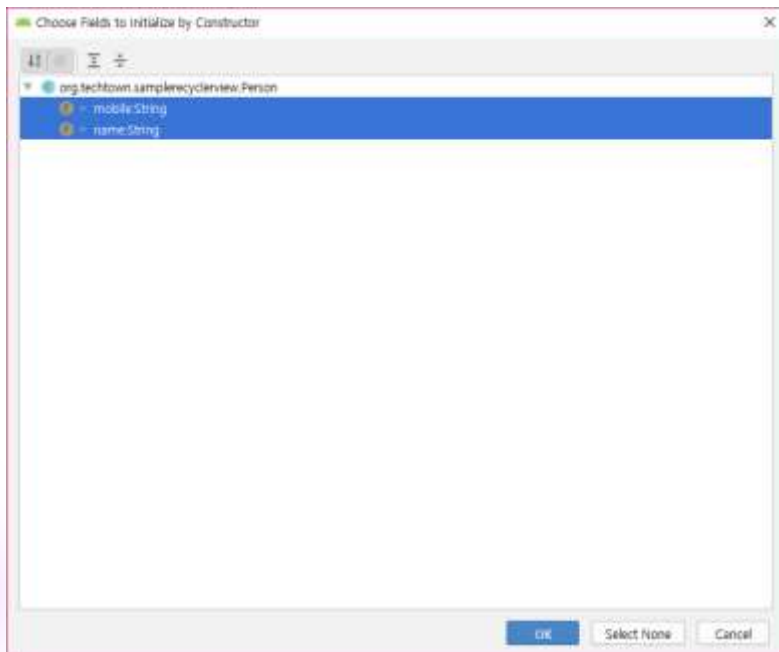


리사이클러뷰 만들기

- 각 아이템을 위한 데이터를 담아두기 위해 Person 클래스 추가
- 생성자와 get, set 메서드 추가

참조파일 SampleRecyclerView>/app/java/org.techtown.samplerectyclerview/Person.java

```
public class Person {  
    String name;  
    String mobile;  
}
```





리사이클러뷰 만들기

- 어댑터 소스 추가하고 ViewHolder 정의

참조파일 SampleRecyclerView>/app/java/org.techtown.samplerectyclerview/PersonAdapter.java

```
public class PersonAdapter {  
  
    static class ViewHolder extends RecyclerView.ViewHolder {  
        TextView textView;  
        TextView textView2;  
  
        public ViewHolder(View itemView) {  
            super(itemView);  
  
            textView = itemView.findViewById(R.id.textView);  
            textView2 = itemView.findViewById(R.id.textView2);  
        }  
  
        public void setItem(Person item) {  
            textView.setText(item.getName());  
            textView2.setText(item.getMobile());  
        }  
    }  
}
```

1 뷰홀더 생성자로 전달되는 뷰 객체 참조하기

2 뷰 객체에 들어 있는 텍스트뷰 참조하기



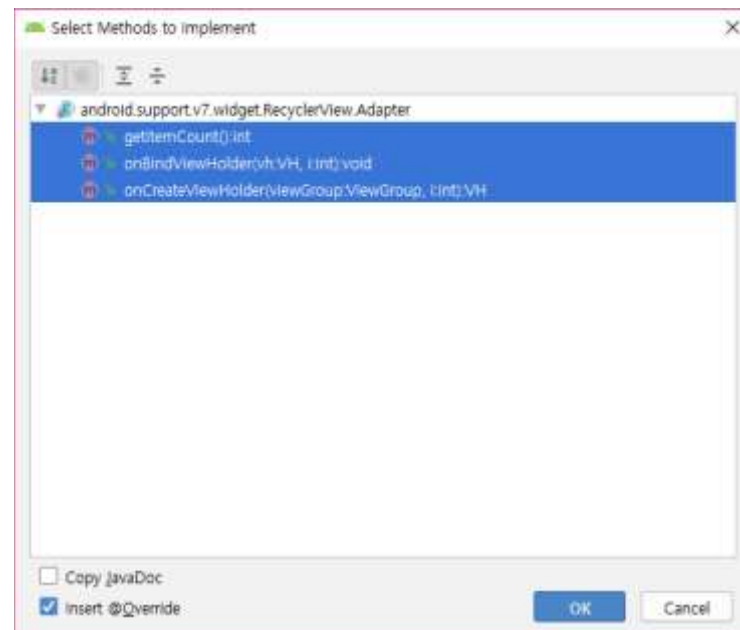
리사이클러뷰 만들기

- PersonAdapter가 상속할 클래스 지정

참조파일 SampleRecyclerView>/app/java/org.techtown.samplerecyclerview/PersonAdapter.java

```
public class PersonAdapter extends RecyclerView.Adapter<PersonAdapter.ViewHolder> {
```

중략...





리사이클러뷰 만들기

- onCreateViewHolder와 onBindViewHolder 메서드 재정의

참조파일 SampleRecyclerView>/app/java/org.techtown.samplerectyclerview/PersonAdapter.java

```
public class PersonAdapter extends RecyclerView.Adapter<PersonAdapter.ViewHolder> {
    ArrayList<Person> items = new ArrayList<Person>();

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int viewType) {
        LayoutInflater inflater = LayoutInflater.from(viewGroup.getContext());
        View itemView = inflater.inflate(R.layout.person_item, viewGroup, false); → ❶ 인플레이션을
        return new ViewHolder(itemView); → ❷ 뷰홀더 객체를 생성하면서 뷰 객체를 통해 뷰 객체
    }                                     전달하고 그 뷰홀더 객체를 반환하기 만들기

    @Override
    public void onBindViewHolder(@NonNull ViewHolder viewHolder, int position) {
        Person item = items.get(position);
        viewHolder.setItem(item);
    }

    @Override
    public int getItemCount() {
        return items.size();
    }
}
```

종락...



리사이클러뷰 만들기

- person_item.xml 파일 정의

참조파일 SampleRecyclerView>/app/res/layout/person_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <android.support.v7.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:cardBackgroundColor="#FFFFFF"
        app:cardCornerRadius="10dp"
        app:cardElevation="5dp"
        app:cardUseCompatPadding="true" >

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <ImageView
                android:id="@+id/imageView"
                android:layout_width="80dp"
                android:layout_height="80dp"
                android:padding="5dp"
                app:srcCompat="@mipmap/ic_launcher" />
```




리싸이클러뷰 만들기

- add, set, get 등의 메서드 추가

```
public void addItem(Person item) {  
    items.add(item);  
}  
  
public void setItems(ArrayList<Person> items) {  
    this.items = items;  
}  
  
public Person getItem(int position) {  
    return items.get(position);  
}  
  
public void setItem(int position, Person item) {  
    return items.set(position, item);  
}
```

종락...



리사이클러뷰 만들기

- MainActivity의 onCreate 메서드 안에 리사이클러뷰와 어댑터를 위한 코드 추가

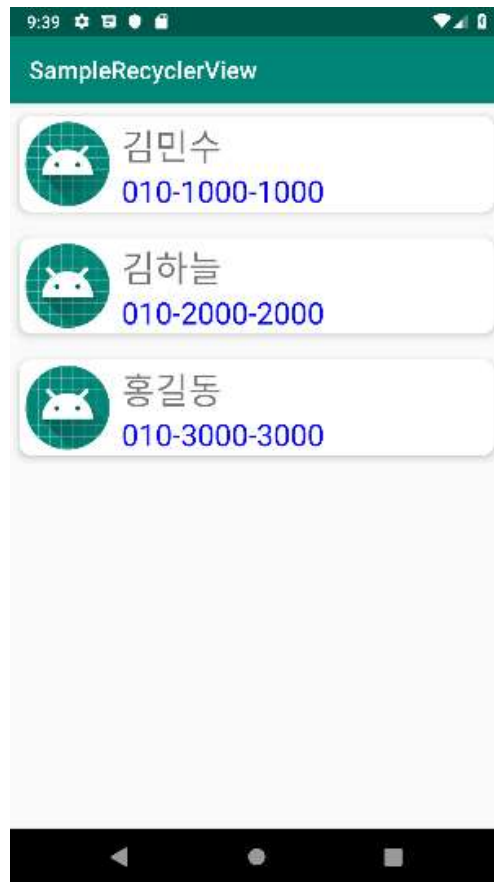
참조파일 SampleRecyclerView>/app/java/org.techtown.samplerecyclerview/MainActivity.java

```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        RecyclerView recyclerView = findViewById(R.id.recyclerView);  
  
        LinearLayoutManager layoutManager =  
            new LinearLayoutManager(this, LinearLayoutManager.VERTICAL, false);  
        recyclerView.setLayoutManager(layoutManager);  
        PersonAdapter adapter = new PersonAdapter();  
  
        adapter.addItem(new Person("김민수", "010-1000-1000"));  
        adapter.addItem(new Person("김하늘", "010-2000-2000"));  
        adapter.addItem(new Person("홍길동", "010-3000-3000"));  
    }  
}
```




리싸이클러뷰 만들기

- 앱 실행





activity_main.xml

```
<LinearLayout 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"
    android:orientation="vertical"
    tools:context=".MainActivity" >
```

```
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
</LinearLayout>
```



person_item.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <androidx.cardview.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:cardBackgroundColor="#FFFFFF"
        app:cardCornerRadius="10dp"
        app:cardElevation="5dp"
        app:cardUseCompatPadding="true">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">
            <ImageView
                android:id="@+id/imageView"
                android:layout_width="80dp"
                android:layout_height="80dp"
                android:padding="5dp"
                app:srcCompat="@mipmap/ic_launcher" />
```



person_item.xml

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="5dp"
    android:layout_weight="1"
    android:orientation="vertical">
    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="이름"
        android:textSize="30sp" />
    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="전화번호"
        android:textColor="#FF0000FF"
        android:textSize="25sp" />
</LinearLayout>
</LinearLayout>
</androidx.cardview.widget.CardView>
</LinearLayout>
```

```
public class Person {  
    String name;  
    String mobile;  
  
    public Person(String name, String mobile) {  
        this.name = name;  
        this.mobile = mobile;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name = name;  
    }  
  
    public String getMobile() {  
        return mobile;  
    }  
  
    public void setMobile(String mobile) {  
        this.mobile = mobile;  
    }  
}
```

```
public class MainActivity extends AppCompatActivity {
```

```
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_main);
```

```
        RecyclerView recyclerView = findViewById(R.id.recyclerView);
```

```
        LinearLayoutManager layoutManager
```

```
            = new LinearLayoutManager(this,
```

```
                LinearLayoutManager.VERTICAL, false);
```

```
        recyclerView.setLayoutManager(layoutManager);
```

```
        PersonAdapter adapter = new PersonAdapter();
```

```
        adapter.addItem(new Person("김민수", "010-1000-1000"));
```

```
        adapter.addItem(new Person("김하늘", "010-2000-2000"));
```

```
        adapter.addItem(new Person("홍길동", "010-3000-3000"));
```

```
        recyclerView.setAdapter(adapter);
```

```
    }
```

```
}
```

RecyclerView에는 레이아웃 매니저를 설정할 수 있다
레이아웃 매니저는 RecyclerView 가 보일 기본적인 형태를
설정할 때 사용
자주 사용하는 형태는 세로방향, 가로방향, 격자모양임

```
public class PersonAdapter extends RecyclerView.Adapter<PersonAdapter.ViewHolder> {
```

```
    ArrayList<Person> items = new ArrayList<Person>();
```

```
    @NonNull
```

```
    @Override
```

```
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int viewType) {
```

```
        LayoutInflater inflater = LayoutInflater.from(viewGroup.getContext());
```

```
        View itemView = inflater.inflate(R.layout.person_item, viewGroup, false);
```

```
        return new ViewHolder(itemView);
```

```
    }
```

```
    @Override
```

```
    public void onBindViewHolder(@NonNull ViewHolder viewHolder, int position) {
```

```
        Person item = items.get(position);
```

```
        viewHolder.setItem(item);
```

```
    }
```

```
    @Override
```

```
    public int getItemCount() {
```

```
        return items.size();
```

```
    }
```

```
    public void addItem(Person item) {
```

```
        items.add(item);
```

```
    }
```

onCreateViewHolder()에서 파라미터로 전달되는 뷰그룹 객체는 각 아이템을 위한 뷰그룹 객체이므로 XML레이아웃을 인플레이션하여 이 뷰그룹 객체에 설정함

XML레이아웃을 ViewGroup객체에 인플레이션한 후 ViewHolder객체에 넣어둔다

onBindViewHolder()-재활용할 수 있는 뷰홀더 객체를 파라미터로 전달하기 때문에 그 뷰홀더에 현재 아이템에 맞는 데이터만 설정함

getItemCount() - 어댑터에서 관리하는 아이템의 개수를 반환
onCreateViewHolder() - 뷰홀더 객체가 만들어질 때 자동으로 호출
뷰홀더가 새로 만들어지는 시점에 호출되므로 그 안에서는 각 아이템을 위해 정의한 XML 레이아웃을 이용해 뷰 객체를 만들어 줌
그리고 뷰객체를 새로 만든 뷰 홀더 객체에 담아 반환
onBindViewHolder()-재사용될 때 자동으로 호출
뷰홀더가 재사용될 때 호출되므로 뷰객체는 기존것을 그대로 사용하고 데이터만 바꿔줌

```

public void setItems(ArrayList<Person> items) {
    this.items = items;
}

public Person getItem(int position) {
    return items.get(position);
}

public void setItem(int position, Person item) {
    items.set(position, item);
}

static class ViewHolder extends RecyclerView.ViewHolder {
    TextView textView;
    TextView textView2;

    public ViewHolder(View itemView) {
        super(itemView);
        textView = itemView.findViewById(R.id.textView);
        textView2 = itemView.findViewById(R.id.textView2);
    }

    public void setItem(Person item) {
        textView.setText(item.getName());
        textView2.setText(item.getMobile());
    }
}
}

```

리스트형태로 보일 때 각각의 아이템은 뷰로 만들어지며
 각각의 아이템을 위한 뷰는 뷰홀더에 담아두게 됨
 이 뷰 홀더 역할을 하는 클래스를 PersonAdapter 클래스
 안에 넣어둔다고 생각하면 됨
 ViewHolder 클래스의 생성자에는 뷰 객체가 전달됨
 setItem() 메서드는 이 뷰홀더에 들어있는 뷰 객체의
 데이터를 다른 것으로 보이도록 하는 역할을 함



리사이클러뷰 만들기

• 여러 칼럼의 뷰로 보여주기

격자모양으로 보이도록 변경하고, 각 아이템을 클릭했을 때 동작하도록 만든다

참조파일 SampleRecyclerView2>/app/java/org.techtown.recyclerview/MainActivity.java

```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        RecyclerView recyclerView = findViewById(R.id.recyclerView);  
  
        GridLayoutManager layoutManager = new GridLayoutManager(this, 2);  
        recyclerView.setLayoutManager(layoutManager);  
    }  
}
```

중략...

리사이클러뷰에 GridLayoutManager를 레이아웃 매니저로 설정하기





리사이클러뷰 만들기

• 클릭했을 때 이벤트 처리

참조파일 SampleRecyclerView2>

중략...

```
static class ViewHolder extends RecyclerView.ViewHolder {
    TextView textView;
    TextView textView2;

    public ViewHolder(View itemView, final OnPersonItemClickListener listener) {
        super(itemView);

        textView = itemView.findViewById(R.id.textView);
        textView2 = itemView.findViewById(R.id.textView2);

        itemView.setOnClickListener(new View.OnClickListener() { → ❶ 아이템 뷰에 OnClickListener
            @Override
            public void onClick(View view) {
                int position = getAdapterPosition();

                if (listener != null) {
                    listener.onItemClick(ViewHolder.this, view, position); } ❷ 아이템 뷰 클릭 시
                }
            }
        });
    }
};
```

참조파일 SampleRecyclerView2>/app/java/org.techtown.recyclerview/OnPersonItemClickListener.java

```
public interface OnPersonItemClickListener {
    public void onItemClick(PersonAdapter.ViewHolder holder, View view, int position);
}
```



리사이클러뷰 만들기

- 어댑터에 리스너를 위한 변수 선언

참조파일 SampleRecyclerView2>/app/java/org.techtown.recyclerview/PersonAdapter.java

중략...

```
public class PersonAdapter extends RecyclerView.Adapter<PersonAdapter.ViewHolder>
    implements OnPersonItemClickListener {
    ArrayList<Person> items = new ArrayList<Person>();
    OnPersonItemClickListener listener;

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int viewType) {
        LayoutInflater inflater = LayoutInflater.from(viewGroup.getContext());
        View itemView = inflater.inflate(R.layout.person_item, viewGroup, false);

        return new ViewHolder(itemView, this);
    }
}
```

중략...



리사이클러뷰 만들기

- 어댑터에 리스너를 위한 변수 선언

```
public void setOnItemClickListener(OnPersonItemClickListener listener) {  
    this.listener = listener;  
}  
  
@Override  
public void onItemClick(ViewHolder holder, View view, int position) {  
    if (listener != null) {  
        listener.onItemClick(holder, view, position);  
    }  
}
```

종락...



리사이클러뷰 만들기

• 메인 액티비티에 추가

```
recyclerView.setAdapter(adapter);

adapter.setOnItemClickListener(new OnPersonItemClickListener() { → ❶
    @Override
    public void onItemClick(PersonAdapter.ViewHolder holder, View view, int position) {

        Person item = adapter.getItem(position); → ❷
        Toast.makeText(getApplicationContext(), "아이템 선택됨: " + item.getName(),
            Toast.LENGTH_LONG).show();
    }
});
}
```




```
public interface OnPersonItemClickListener {  
    public void onItemClick(PersonAdapter.ViewHolder holder, View view, int position);  
}
```

각 아이템을 클릭했을 때 토스트 메시지가 표시되도록 수정
클릭이벤트는 리사이클러뷰가 아니라 각 아이템에 발생하게 되므로 뷰홀더 안에서 클릭이벤트를 처리할 수 있도록 만드는 것이 좋다
뷰홀더의 생성자로 뷰 객체가 전달되므로 이 뷰 객체에 OnClickListener를 설정
그러면 이 뷰를 클릭했을 때 그 리스너의 onClick()메서드가 호출됨
이 리스너안에서 토스트 메시지를 띄우면 클릭했을 대의 기능이 변경될 때마다 어댑터를 수정해야 하는 문제가 생김

- ⇒ 어댑터 객체 밖에서 리스너를 설정하고 설정된 리스너 쪽으로 이벤트를 전달받도록 하는 것이 좋다
- ⇒ 이를 위해 OnPersonItemClickListener 인터페이스를 정의
- ⇒ onItemClick()메서드가 호출될 때 파라미터로 뷰홀더 객체, 뷰 객체, 뷰의 position 정보가 전달되도록 한다
- ⇒ position-몇번째 아이템인지 구분할 수 있는 인덱스값

```
public class MainActivity extends AppCompatActivity {  
    RecyclerView recyclerView;  
    PersonAdapter adapter;
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    recyclerView = findViewById(R.id.recyclerView);
```

```
    GridLayoutManager layoutManager = new GridLayoutManager(this, 2);  
    recyclerView.setLayoutManager(layoutManager);
```

```
    adapter = new PersonAdapter();
```

```
    adapter.addItem(new Person("김민수", "010-1000-1000"));  
    adapter.addItem(new Person("김하늘", "010-2000-2000"));  
    adapter.addItem(new Person("홍길동", "010-3000-3000"));  
    adapter.addItem(new Person("내이름1", "010-4000-4000"));  
    adapter.addItem(new Person("내이름2", "010-4000-4000"));  
    adapter.addItem(new Person("내이름3", "010-4000-4000"));  
    adapter.addItem(new Person("내이름4", "010-4000-4000"));  
    adapter.addItem(new Person("내이름5", "010-4000-4000"));  
    adapter.addItem(new Person("내이름6", "010-4000-4000"));  
    adapter.addItem(new Person("내이름7", "010-4000-4000"));
```

```
adapter.addItem(new Person("내 이름8", "010-4000-4000"));
adapter.addItem(new Person("내 이름9", "010-4000-4000"));
adapter.addItem(new Person("내 이름10", "010-4000-4000"));
```

```
recyclerView.setAdapter(adapter);
```

```
adapter.setOnItemClickListener(new OnPersonItemClickListener() {
    @Override
    public void onItemClick(PersonAdapter.ViewHolder holder, View view, int position) {
        Person item = adapter.getItem(position);

        Toast.makeText(getApplicationContext(), "아이템 선택됨 : " + item.getName(), Toast.LENGTH_LONG).show();
    }
});
```

리사이클러뷰 객체와 어댑터 객체는 이 클래스안의 어디서든 접근할 수 있도록 클래스 안에 선언된 변수에 할당
어댑터 객체에는 `setOnItemClickListener()` 메서드를 호출하면서 리스트 객체를 설정
이렇게 하면 각 아이템이 클릭되었을 때 이 리스너의 `onItemClick()` 메서드가 호출됨
`onItemClick()` 안에서는 어댑터 객체의 `getItem()` 메서드를 이용해 클릭된 아이템 객체를 확인


```
public class PersonAdapter extends RecyclerView.Adapter<PersonAdapter.ViewHolder>
```

```
    implements OnItemClickListener {
```

```
        ArrayList<Person> items = new ArrayList<Person>();
```

```
        OnItemClickListener listener;
```

```
        @NonNull
```

```
        @Override
```

```
        public ViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int viewType) {
```

```
            LayoutInflater inflater = LayoutInflater.from(viewGroup.getContext());
```

```
            View itemView = inflater.inflate(R.layout.person_item, viewGroup, false);
```

```
            return new ViewHolder(itemView, this);
```

```
        }
```

```
        @Override
```

```
        public void onBindViewHolder(@NonNull ViewHolder viewHolder, int position) {
```

```
            Person item = items.get(position);
```

```
            viewHolder.setItem(item);
```

```
        }
```

```
        @Override
```

```
        public int getItemCount() {
```

```
            return items.size();
```

```
        }
```

OnItemClickListener 인터페이스를 구현

onItemClick()메서드를 추가

이 메서드는 뷰홀더 클래스 안에서 뷰가 클릭되었을 때 호출되는 메서드

이 어댑터 클래스 안에서가 아니라 밖에서 이벤트를 처리를 하는 것이

일반적이므로 listener라는 이름의 변수를 하나 선언하고

setOnItemClickListener()메서드를 추가하여 이 메서드가 호출되었을 때

리스너객체를 변수에 할당하도록 함

이렇게 하면 onItemClick()메서드가 호출되었을 때 다시 외부에서 설정된

메서드가 호출되도록 만들 수 있다

```
public void addItem(Person item) {  
    items.add(item);  
}
```

```
public void setItems(ArrayList<Person> items) {  
    this.items = items;  
}
```

```
public Person getItem(int position) {  
    return items.get(position);  
}
```

```
public void setItem(int position, Person item) {  
    items.set(position, item);  
}
```

```
public void setOnItemClickListener(OnItemClickListener listener) {  
    this.listener = listener;  
}
```

@Override

```
public void onItemClick(ViewHolder holder, View view, int position) {  
    if (listener != null) {  
        listener.onItemClick(holder, view, position);  
    }  
}
```

```
static class ViewHolder extends RecyclerView.ViewHolder {
```

```
    TextView textView;
```

```
    TextView textView2;
```

```
    public ViewHolder(View itemView, final OnPersonItemClickListener listener) {
```

```
        super(itemView);
```

```
        textView = itemView.findViewById(R.id.textView);
```

```
        textView2 = itemView.findViewById(R.id.textView2);
```

```
        itemView.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
```

```
            public void onClick(View view) {
```

```
                int position = getAdapterPosition();
```

```
                if (listener != null) {
```

```
                    listener.onItemClick(ViewHolder.this, view, position);
```

```
                }
```

```
            }
```

```
        });
```

```
    }
```

```
    public void setItem(Person item) {
```

```
        textView.setText(item.getName());
```

```
        textView2.setText(item.getMobile());
```

```
    }
```

```
}
```

ViewHolder()생성자가 호출될 때 리스너 객체가
파라미터로 전달되도록 수정
이 리스너 객체는 어댑터 밖에서 설정할 것이며
뷰홀더까지 전달됨
이렇게 전달된 리스너 객체의 onItemClick() 이벤트는
뷰가 클릭되었을 때 호출됨

getAdapterPosition()-이 뷰홀더에 표시할 아이템이
어댑터에서 몇 번째인지 정보를 반환함

4.

스피너 사용하기



스피너 사용하기 예제

스피너 사용하기 예제

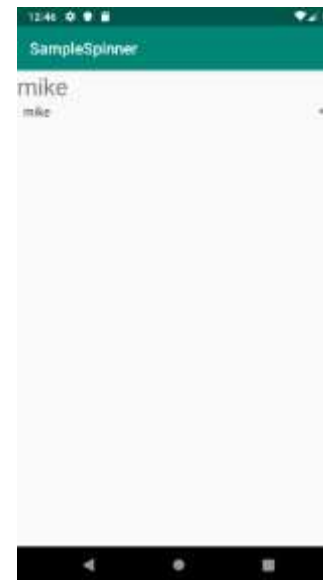
- 콤보박스처럼 선택할 수 있는 스피너 사용
- XML 레이아웃에 정의한 스피너 참조

메인 액티비티의 XML 레이아웃

- 스피너를 포함하는 메인 화면의
XML 레이아웃 정의

메인 액티비티 코드 작성

- 스피너 객체를 참조하여 설정





XML 레이아웃 만들기

- <Spinner> 태그를 사용하여 레이아웃에 추가

```
<Spinner  
    android:id="@+id/spinner"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
>
```

1

스피너 정의



메인 액티비티 코드 만들기

- 안드로이드에서 미리 만들어 제공하는 어댑터를 사용할 수 있음

```
String[] items = { "mike", "angel", "crow", "john", "ginnie", "sally", "cohen", "rice" };
```

1 스피너에 표시될 아이템들의 데이터를 배열로 정의

```
selection = (TextView) findViewById(R.id.selection);  
Spinner spinner = (Spinner) findViewById(R.id.spinner);  
spinner.setOnItemSelectedListener(this);
```

Continued..



메인 액티비티 코드 만들기 (계속)

```
ArrayAdapter<String> adapter = new ArrayAdapter<String>(
    this, android.R.layout.simple_spinner_item, items);
adapter.setDropDownViewResource(
    android.R.layout.simple_spinner_dropdown_item);
```

2 ArrayAdapter를 이용해 어댑터 객체 생성

```
spinner.setAdapter(adapter);
```

3 스피너에 어댑터 설정

```
public void onItemSelected(AdapterView<?> parent, View v, int position, long id) {
    selection.setText(items[position]);
}
```

4 스피너의 아이템이 선택되었을 때 처리하는 메소드 정의

```
public void onNothingSelected(AdapterView<?> parent) {
    selection.setText("");
}
```

setDropDownViewResource() 메서드는 스피너의 각 아이템들을 보여줄 뷰에 사용되는 레이아웃을 지정하는 데 사용되며 안드로이드에서 미리 정의한 리소스인 `android.R.layout.simple_spinner_dropdown_item` 값을 전달하면 가장 단순한 형태의 뷰가 보이게 됨

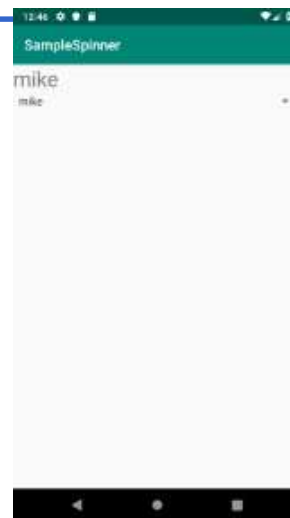


ArrayAdapter의 사용

[Reference]

```
public ArrayAdapter (Context context, int textViewResourceId, T[] objects)
```

- 첫 번째 파라미터는 Context 객체이므로 액티비티인 this를 전달하면 됨
- 두 번째 파라미터는 뷰를 초기화할 때 사용되는 XML 레이아웃의 리소스 ID 값으로 이 코드에서는 `android.R.layout.simple_spinner_item`을 전달하였음
 - 이 ID값은 안드로이드에서 미리 정의한 레이아웃으로 문자열을 아이템으로 보여주는 단순 스피너 아이템의 레이아웃이라고 보면 됨
- 세 번째 파라미터는 아이템으로 보일 문자열 데이터들의 배열임





activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:orientation="vertical"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="선택한 아이 템"
        android:textSize="30sp" />

    <Spinner
        android:id="@+id/spinner"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

```
public class MainActivity extends AppCompatActivity {
    TextView textView;
    String[] items = {"mike", "angel", "crow", "john", "ginnie", "sally", "cohen", "rice"};

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        textView = findViewById(R.id.textView);
        Spinner spinner = findViewById(R.id.spinner);

        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this, android.R.layout.simple_spinner_item, items);
        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        spinner.setAdapter(adapter);

        spinner.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view, int position, long id) {
                textView.setText(items[position]);
            }
            @Override
            public void onNothingSelected(AdapterView<?> adapterView) {
                textView.setText("");
            }
        });
    }
}
```



[References]

- 기본 서적
2019, 정재곤, “Do it! 안드로이드 앱 프로그래밍(개정6판)”, 이지스퍼블리싱(주)
- Android Website
<http://www.android.com/>
- Google Developer’s Conference
<http://code.google.com/events/io/>
- Android SDK Documentation