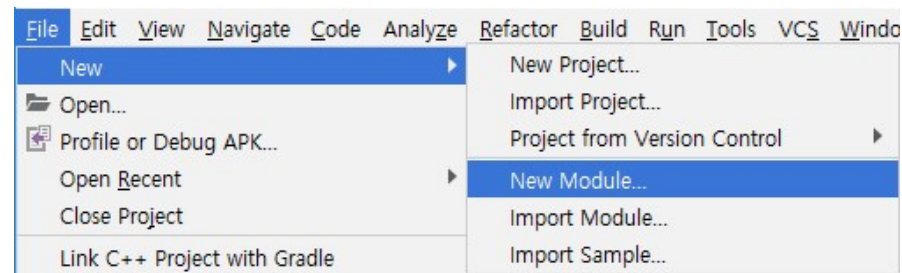
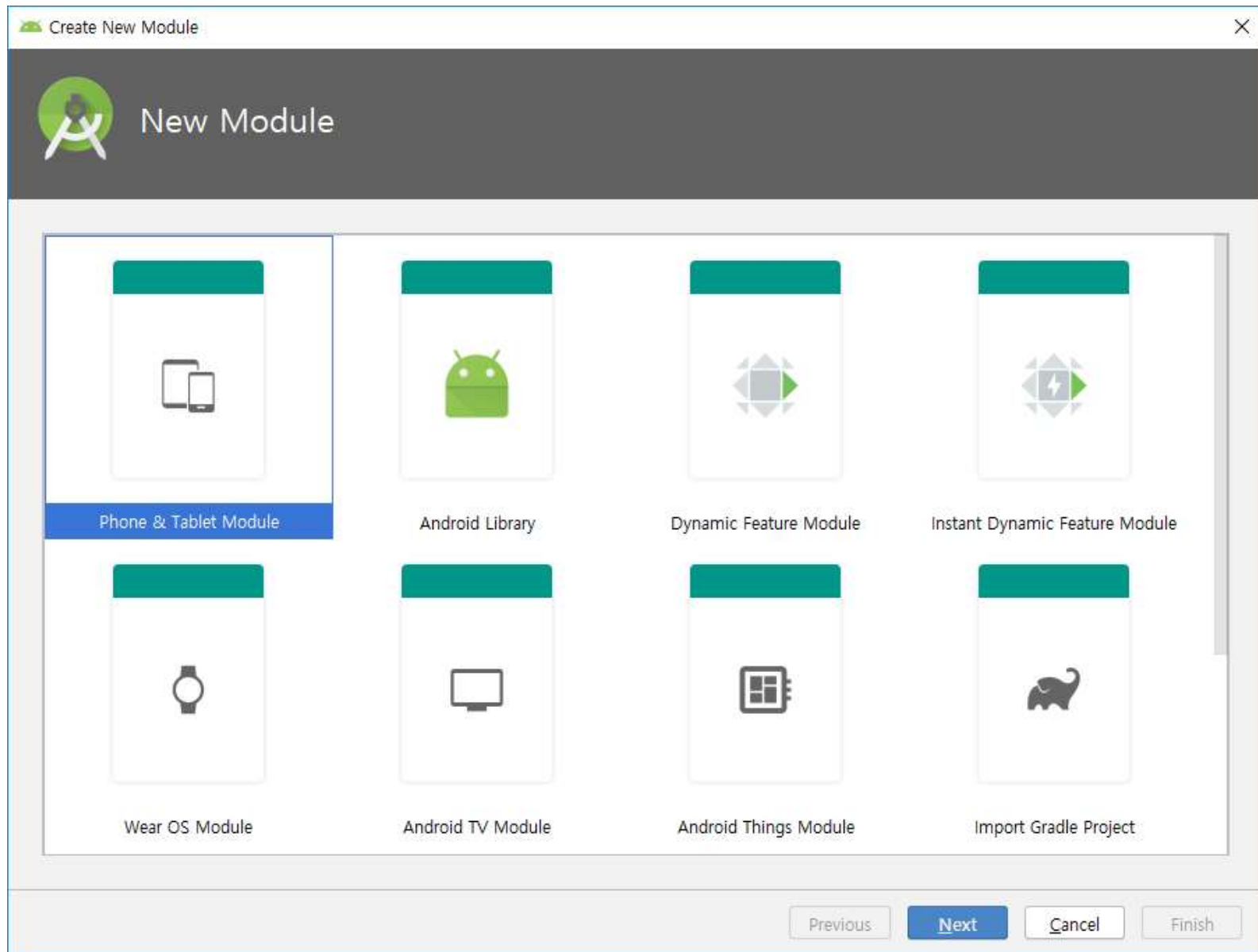




커스텀 뷰

Step 1 _ 모듈 생성





Create New Module

 Phone & Tablet Module

Configure the new module

Application/Library name

Module name

Package name

com.example.user.lab11

Edit

Language

Java

Minimum SDK

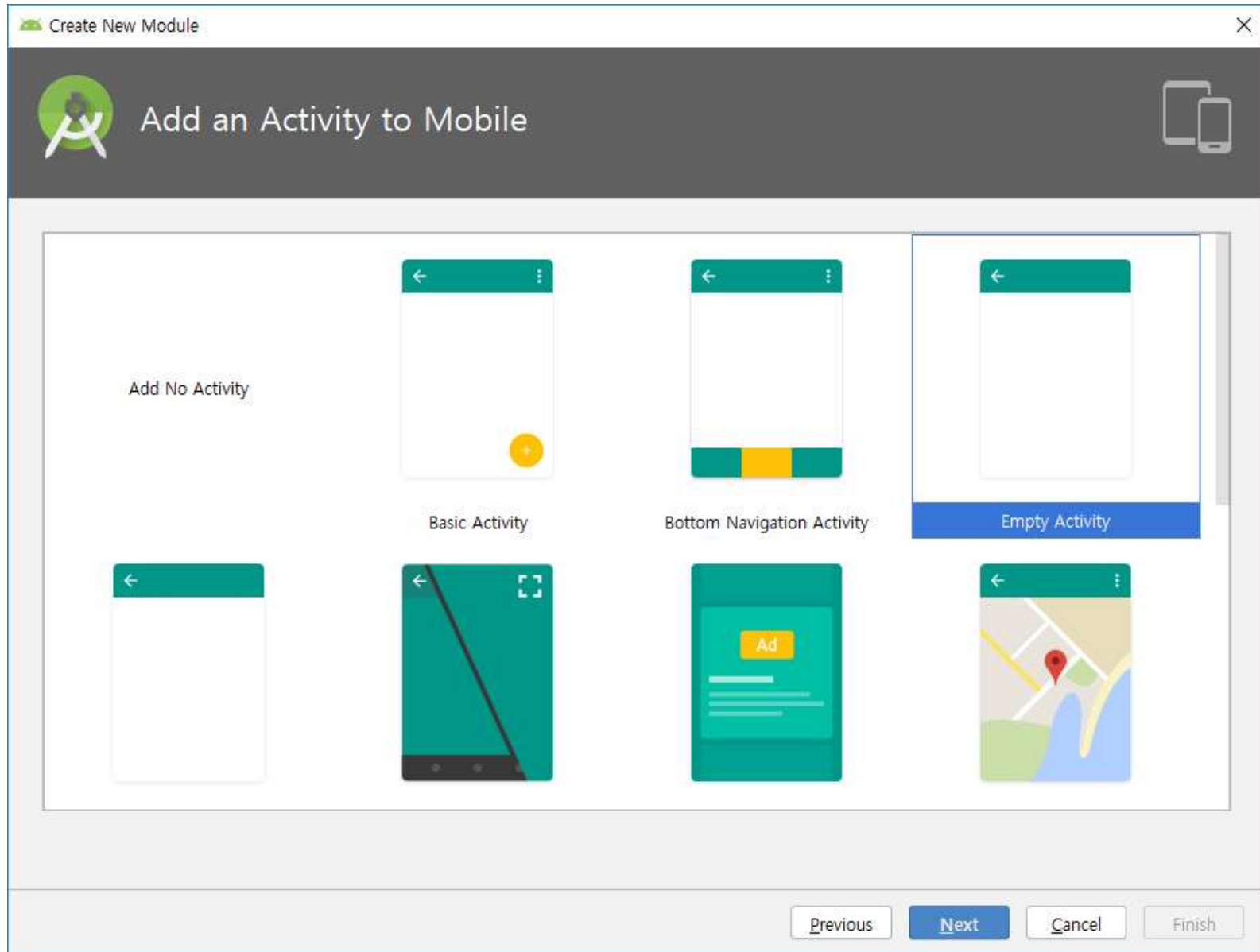
API 15: Android 4.0.3 (IceCreamSandwich)

Previous



Next

Cancel

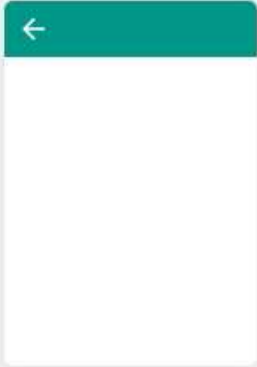
Finish



Create New Module

 Configure Activity

Creates a new empty activity



Activity Name:

MainActivity

☒ Generate Layout File

Layout Name:

activity_main

Source Language:

Java

The name of the activity class to create

Previous

Next

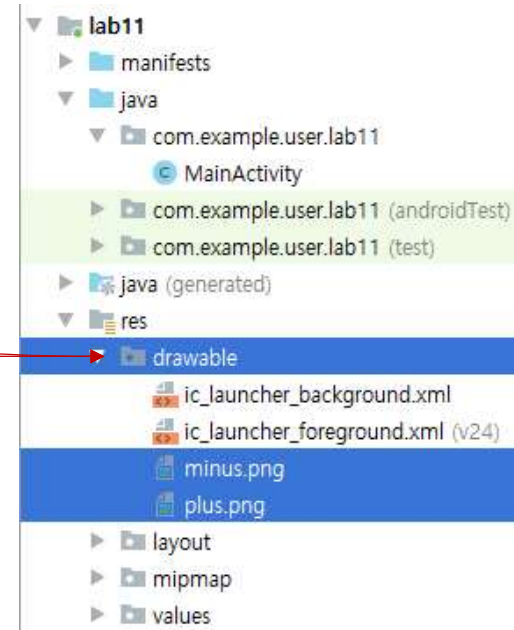
Cancel

Finish

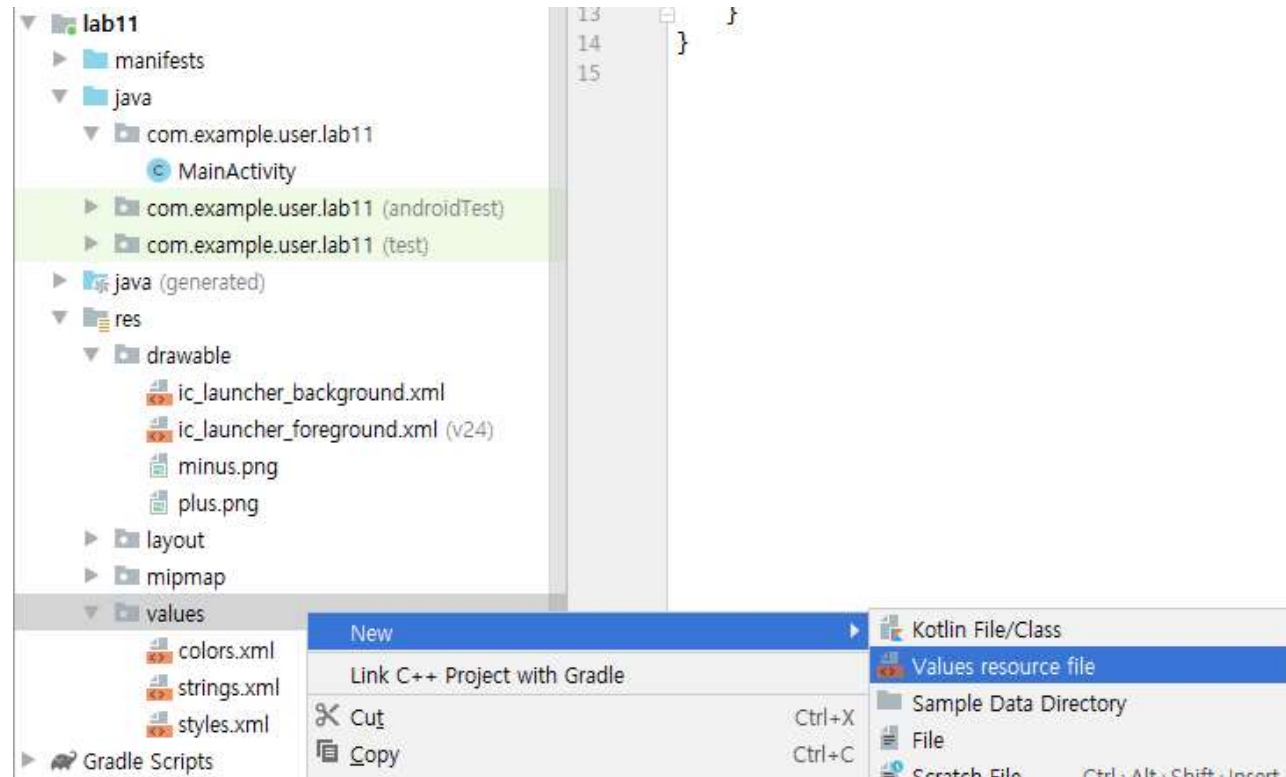
Step 2 _ 파일 복사

plus.png

minus.png



Step 3 _ attrs.xml 생성



New Resource File

×

File name:

attrs

Source set:

main

Directory name:

values

Available qualifiers:

Country Code

Network Code

Locale

Layout Direction

Smallest Screen Width

Screen Width

Screen Height

Size

Ratio

Orientation

UI Mode

Night Mode

Density

>>

<<

Chosen qualifiers:

Nothing to show

OK

Cancel

Help

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<resources>
```

```
  <declare-styleable name="MyView">
```

```
    <attr name="customTextColor" format="color" />
```

```
  </declare-styleable>
```

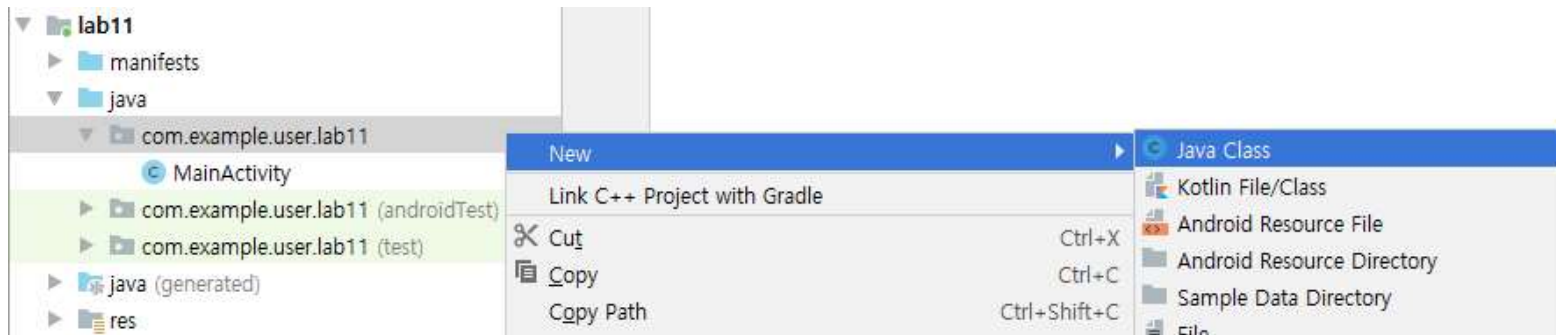
```
</resources>
```

<declare-styleable>은 여러 <attr>을 묶어 한꺼번에 관리하기 위한 태그

<attr> 태그 하나가 속성 하나를 정의하게 됨

속성값으로 색상을 등록해야 한다는 것,
color 이외에 string, float, integer, dimension, enum 등을 지정할 수 있음.

Step 4 _ OnMyChangeListener 인터페이스 생성



Create New Class

Name: OnMyChangeListener

Kind: Interface

Interface(s):

Package:

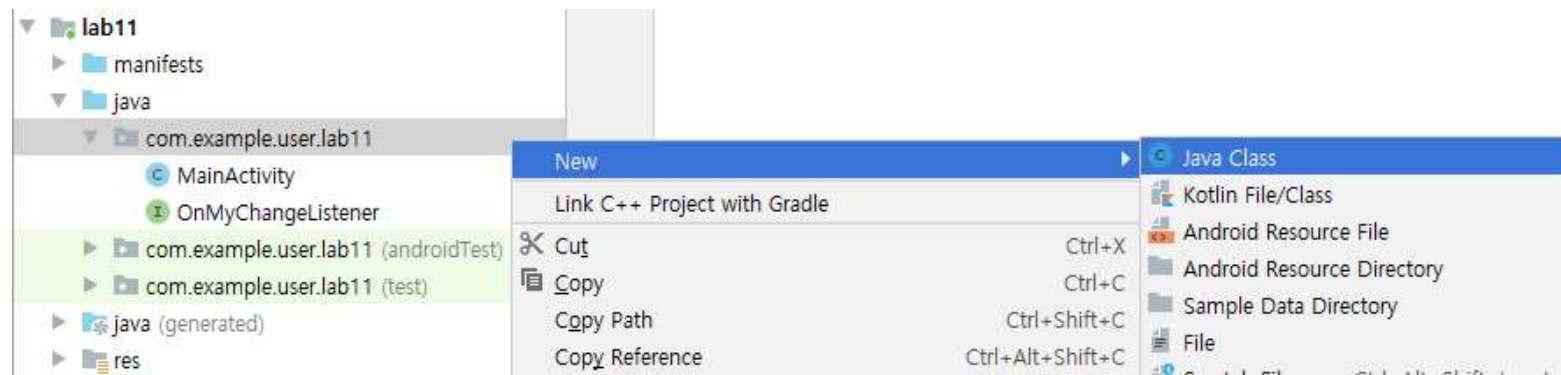
Visibility:

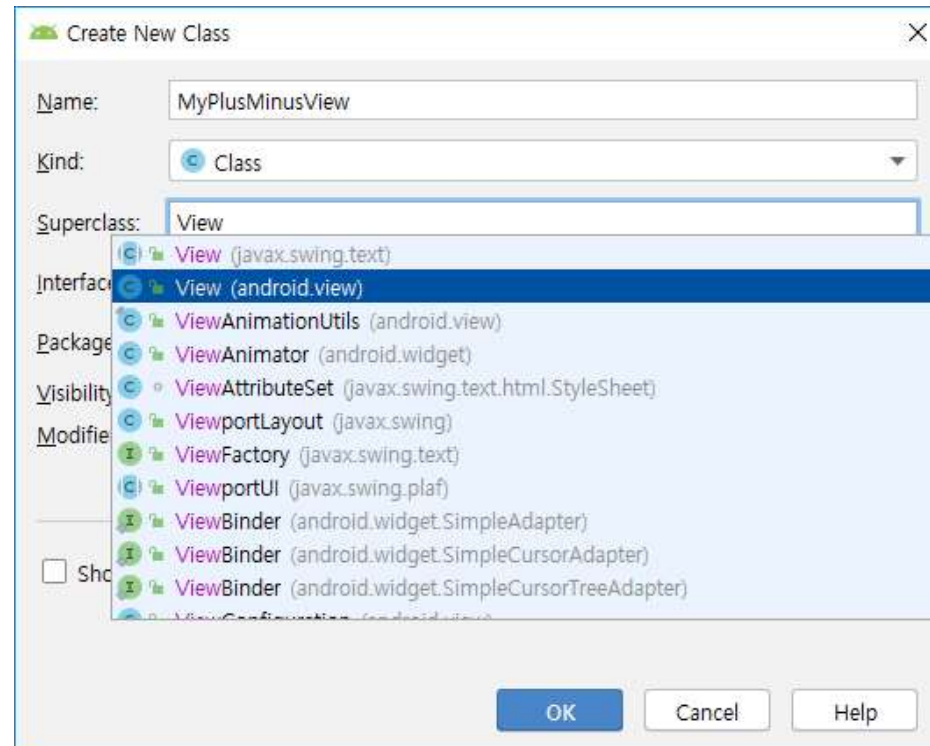
- Class
- Interface
- Enum
- Annotation
- Singleton

OK Cancel Help

```
public interface OnMyChangeListener {  
    void onChange(int value);  
}
```

Step 5 _ MyPlusMinusView 작성





Create New Class

Name: MyPlusMinusView

Kind: Class

Superclass: android.view.View

Interface(s):

Package: com.example.user.lab11

Visibility: ☒ Public ☐ Package Private

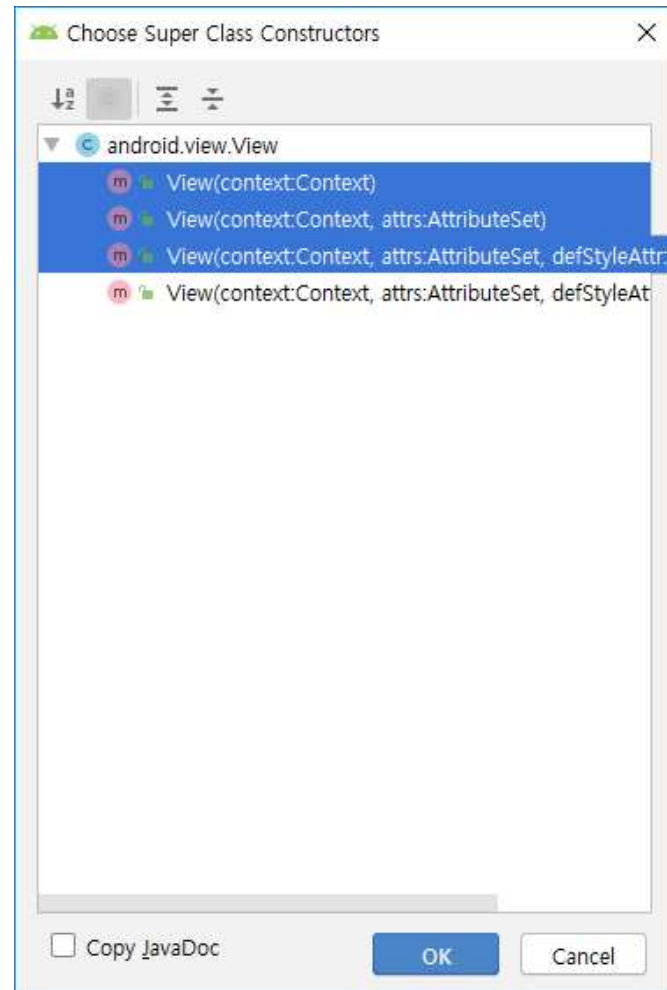
Modifiers: ☒ None ☐ Abstract ☐ Final

☐ Show Select Overrides Dialog

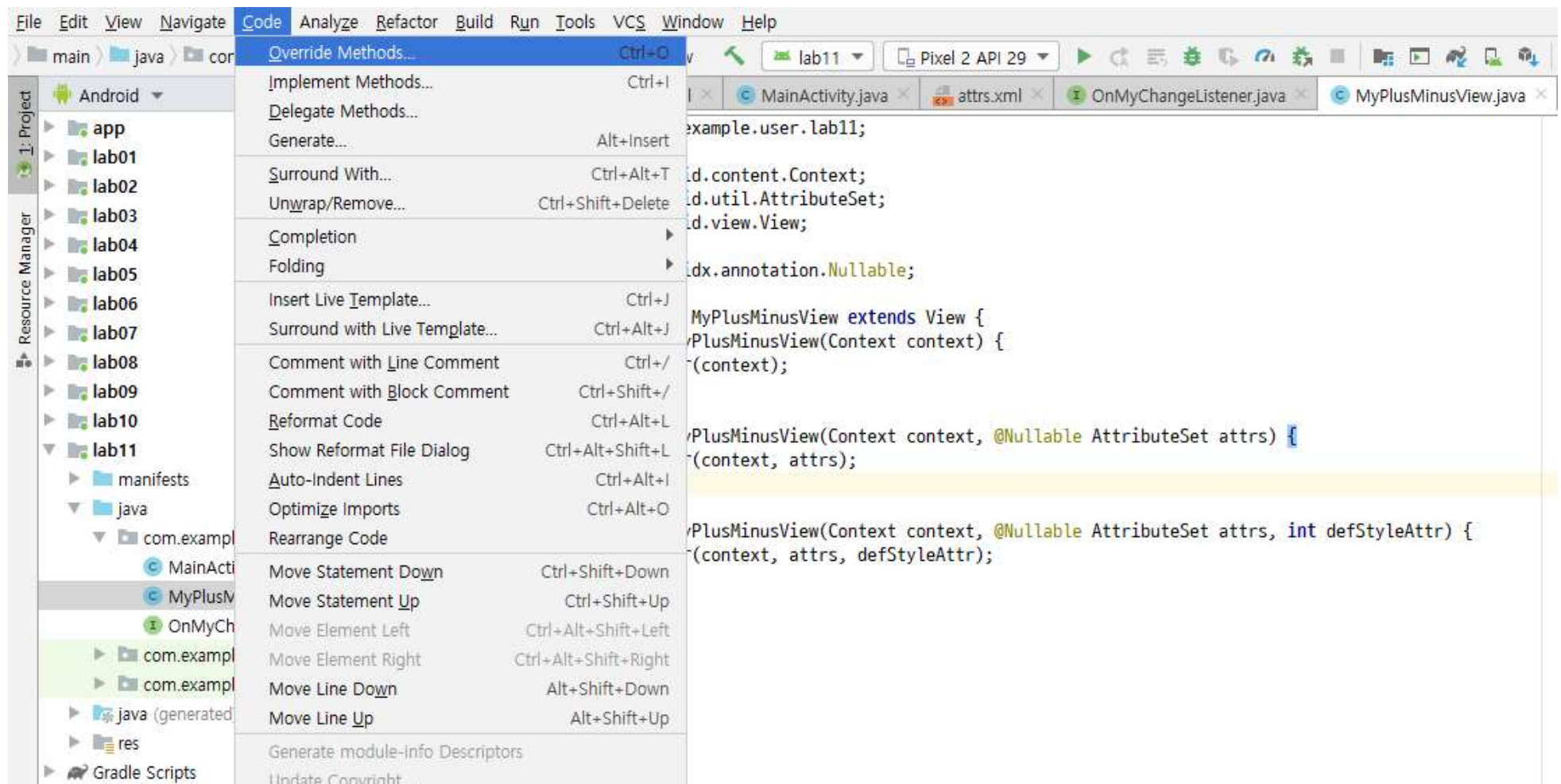
OK Cancel Help

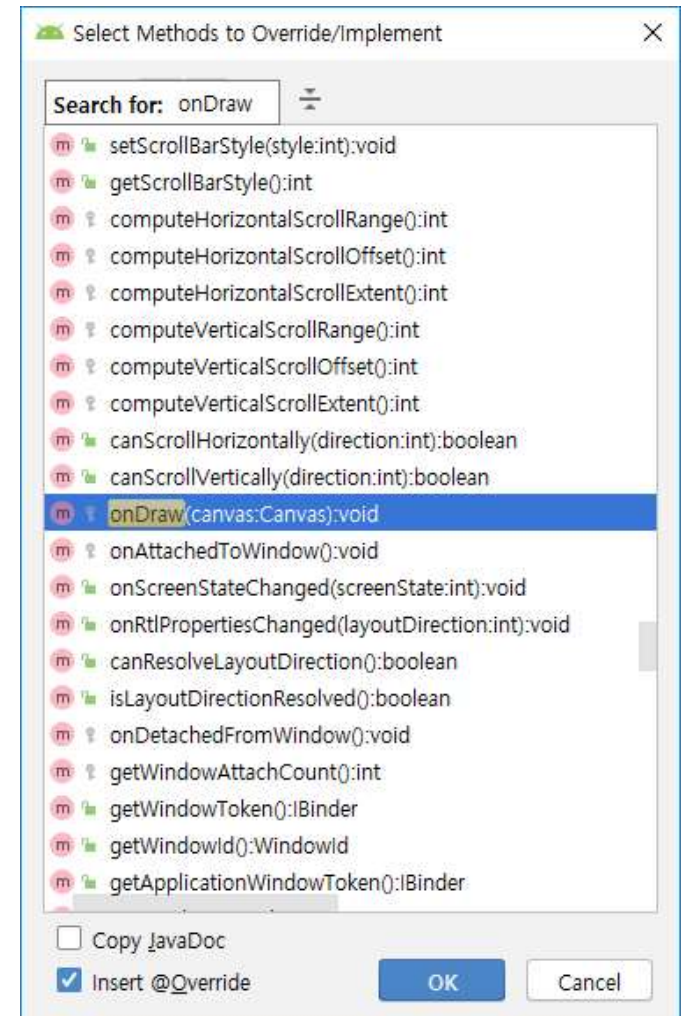
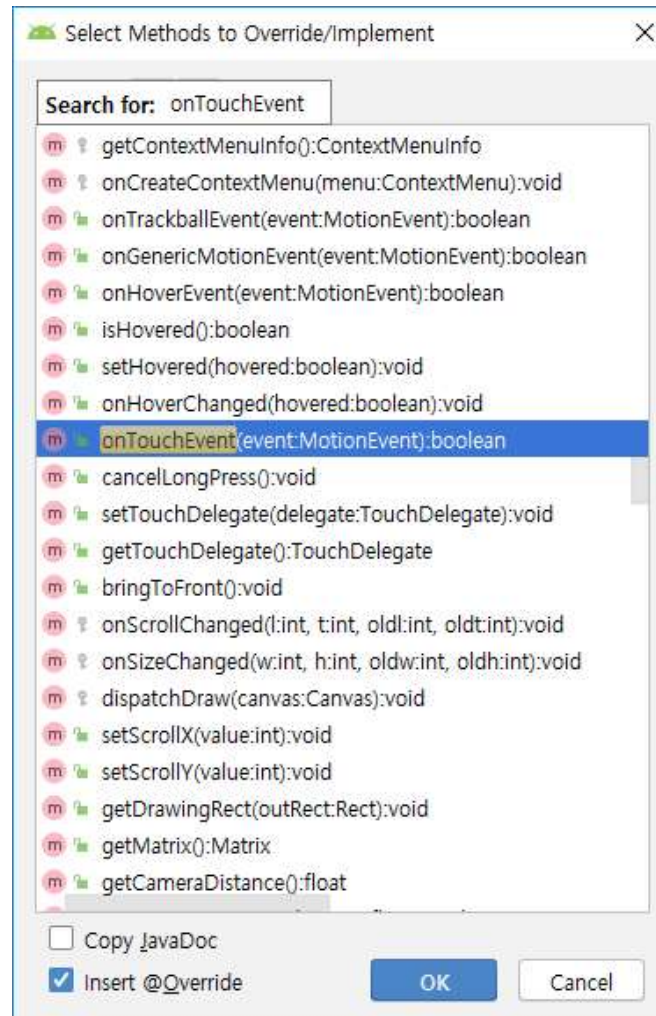
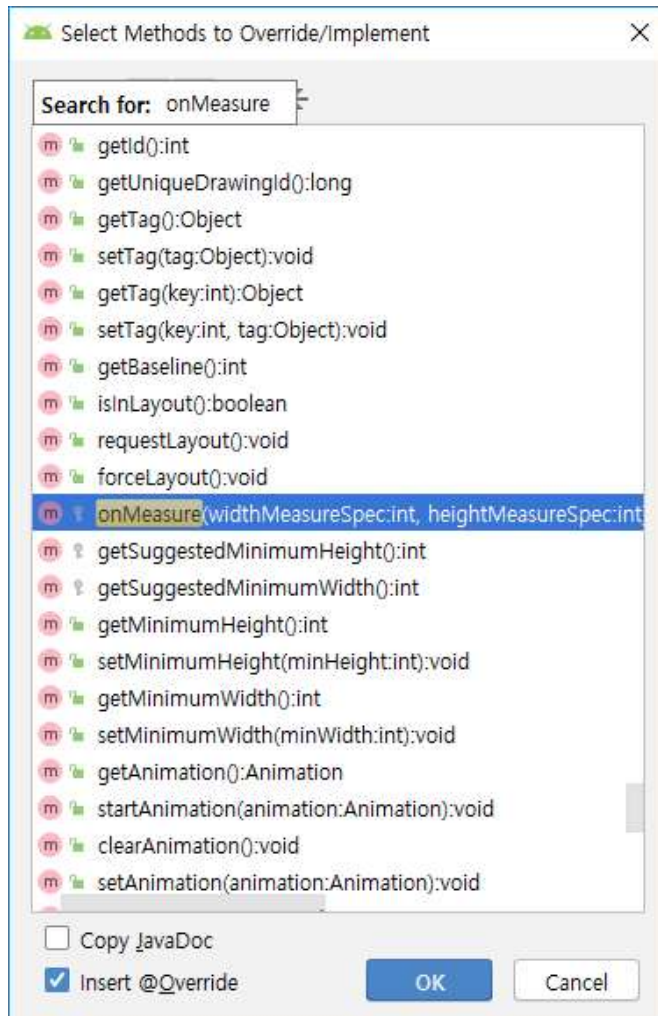

```
3 import android.view.View;
4
5 public class MyPlusMinusView extends View {
6     1
```

There is no default constructor available in 'android.view.View'



```
1 package com.example.user.lab11;
2
3 import android.content.Context;
4 import android.util.AttributeSet;
5 import android.view.View;
6
7 import androidx.annotation.Nullable;
8
9 public class MyPlusMinusView extends View {
10     public MyPlusMinusView(Context context) {
11         super(context);
12     }
13
14     public MyPlusMinusView(Context context, @Nullable AttributeSet attrs) {
15         super(context, attrs);
16     }
17
18     public MyPlusMinusView(Context context, @Nullable AttributeSet attrs, int defStyleAttr) {
19         super(context, attrs, defStyleAttr);
20     }
21 }
```






```
MyPlusMinusView.java x
1 package com.example.user.lab11;
2
3 import android.content.Context;
4 import android.graphics.Canvas;
5 import android.util.AttributeSet;
6 import android.view.MotionEvent;
7 import android.view.View;
8
9 import androidx.annotation.Nullable;
10
11 public class MyPlusMinusView extends View {
12     public MyPlusMinusView(Context context) { super(context); }
13
14
15
16     public MyPlusMinusView(Context context, @Nullable AttributeSet attrs) { super(context, attrs); }
17
18
19
20     public MyPlusMinusView(Context context, @Nullable AttributeSet attrs, int defStyleAttr) {
21         super(context, attrs, defStyleAttr);
22     }
23
24
25     @Override
26     protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
27         super.onMeasure(widthMeasureSpec, heightMeasureSpec);
28     }
29
30
31     @Override
32     public boolean onTouchEvent(MotionEvent event) { return super.onTouchEvent(event); }
33
34
35     @Override
36     protected void onDraw(Canvas canvas) {
37         super.onDraw(canvas);
38     }
39 }
```

```
public class MyPlusMinusView extends View {
    Context context;
    // 증감 값
    int value;
    // 화면 출력 이미지
    Bitmap plusBitmap;
    Bitmap minusBitmap;
    // 이미지가 화면에 출력되는 좌표 정보
    Rect plusRectDst;
    Rect minusRectDst;

    // value 출력 문자열 색상
    int textColor;

    // Observer를 등록하기 위한 객체
    ArrayList<OnMyChangeListener> listeners;

    public MyPlusMinusView(Context context) {
        super(context);
        this.context = context;
        init(null);
    }

    public MyPlusMinusView(Context context, @Nullable AttributeSet attrs) {
        super(context, attrs);
        this.context = context;
        init(attrs);
    }

    public MyPlusMinusView(Context context, @Nullable AttributeSet attrs, int defStyleAttr) {
        super(context, attrs, defStyleAttr);
        this.context = context;
        init(attrs);
    }
}
```

```

// 생성자의 공통 코드
private void init(AttributeSet attrs) {
    // 이미지 획득
    plusBitmap = BitmapFactory.decodeResource(context.getResources(), R.drawable.plus);
    minusBitmap = BitmapFactory.decodeResource(context.getResources(), R.drawable.minus);

    // 이미지 출력 사각형 좌표 정보 설정
    plusRectDst = new Rect(10, 10, 210, 210);
    minusRectDst = new Rect(400, 10, 600, 210);

    // custom 속성값 획득
    if (attrs != null) {
        TypedArray a = context.obtainStyledAttributes(attrs, R.styleable.MyView);
        textColor = a.getColor(R.styleable.MyView_customTextColor, Color.RED);
    }
    listeners = new ArrayList<>();
}

```

defValue

int: Value to return if the attribute is not defined or not a resource.

// Observer 등록을 위한 함수

```
public void setOnMyChangeListener(OnMyChangeListener listener) {  
    listeners.add(listener);  
}
```

// 크기 결정

@Override

```
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {  
    int widthMode = MeasureSpec.getMode(widthMeasureSpec);  
    int widthSize = MeasureSpec.getSize(widthMeasureSpec);  
    int heightMode = MeasureSpec.getMode(heightMeasureSpec);  
    int heightSize = MeasureSpec.getSize(heightMeasureSpec);  
  
    int width = 0;  
    int height = 0;  
  
    if (widthMode == MeasureSpec.AT_MOST) {  
        width = 700;  
    } else if (widthMode == MeasureSpec.EXACTLY) {  
        width = widthSize;  
    }  
    if (heightMode == MeasureSpec.AT_MOST) {  
        height = 250;  
    } else if (heightMode == MeasureSpec.EXACTLY) {  
        height = heightSize;  
    }  
    setMeasuredDimension(width, height);  
}
```

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    int x = (int)event.getX();
    int y = (int)event.getY();
    // 플러스 아이콘이 터치된 거라면...
    if (plusRectDst.contains(x, y) && event.getAction() == MotionEvent.ACTION_DOWN) {
        // 데이터 변경
        value++;
        // 화면 갱신
        invalidate();
        for (OnChangeListener listener: listeners) {
            // observer에 데이터 전달
            listener.onChange(value);
        }
        return true;
    } else if (minusRectDst.contains(x, y) && event.getAction() == MotionEvent.ACTION_DOWN) {
        value--;
        invalidate();
        for (OnChangeListener listener: listeners) {
            // observer에 데이터 전달
            listener.onChange(value);
        }
        return true;
    }
    return false;
}
```

invalidate() 함수만 호출하면 내부적으로 onDraw() 함수가 다시 호출되어 변경된 값으로 다시 그리게 됨

@Override

protected void onDraw(Canvas canvas) {

// 화면 지우기

canvas.drawColor(Color.alpha(Color.CYAN));

Return the alpha component of a color int.

Fill the entire canvas' bitmap (restricted to the current clip) with the specified color, using srcover porterduff mode.

// 이미지의 사각형 정보

Rect plusRectSource = new Rect(0, 0, plusBitmap.getWidth(), plusBitmap.getHeight());

Rect minusRectSource = new Rect(0, 0, minusBitmap.getWidth(), minusBitmap.getHeight());

Paint paint=new Paint();

public void drawBitmap (Bitmap bitmap, Rect src, Rect dst, Paint paint)

// plus 이미지 그리기

canvas.drawBitmap(plusBitmap, plusRectSource, plusRectDst, null);

// value 문자열 그리기

paint.setTextSize(80);

paint.setColor(textColor);

canvas.drawText(String.valueOf(value), 260, 150, paint);

// minus 이미지 그리기

canvas.drawBitmap(minusBitmap, minusRectSource, minusRectDst, null);

}

}

Step 6 _ activity_main.xml 작성

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

    <com.example.user.lab11.MyPlusMinusView
        android:id="@+id/customView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:customTextColor="#0000ff" />

    <View
        android:id="@+id/barView"
        android:layout_width="match_parent"
        android:layout_height="30dp"
        android:background="@android:color/darker_gray" />

</LinearLayout>
```

Step 7 _ MainActivity 작성

```
public class MainActivity extends AppCompatActivity implements OnMyChangeListener {

    View barView;

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

        MyPlusMinusView plusMinusView = findViewById(R.id.customView);
        barView = findViewById(R.id.barView);

        // 인터페이스를 구현한 객체를 View에 등록
        plusMinusView.setOnMyChangeListener(this);
    }

    @Override
    public void onChange(int value) {
        if (value < 0) {
            barView.setBackgroundColor(Color.RED);
        } else if (value < 30) {
            barView.setBackgroundColor(Color.YELLOW);
        } else if (value < 60) {
            barView.setBackgroundColor(Color.BLACK);
        } else {
            barView.setBackgroundColor(Color.GREEN);
        }
    }
}
```

Step 8 _ 실행



커스텀 뷰 작성 방법 - 기본 작성 방법

- API에서 제공하는 뷰를 그대로 이용하면서 약간 변형시킨 뷰

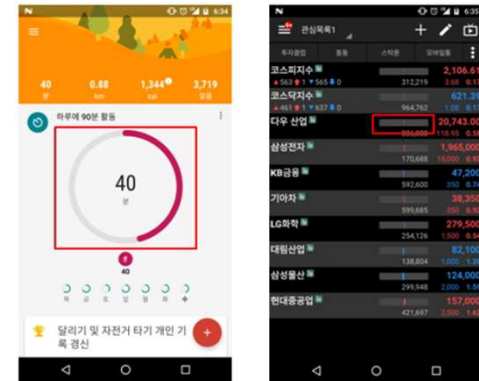
```
public class MyView extends TextView {  
  
}
```

- 여러 뷰를 합쳐서 한번에 출력하기 위한 뷰

```
public class MyView extends ViewGroup {  
  
}
```

- 기존 API에 전혀 존재하지 않는 뷰

```
public class MyView extends View {  
  
}
```



커스텀 뷰 작성 방법 - 기본 작성 방법

```
public class MyView extends View {
```

```
    public MyView(Context context) {  
        super(context);  
    }
```

} when creating a view from code

```
    public MyView(Context context, AttributeSet attrs) {  
        super(context, attrs);  
    }
```

} when inflating a view from XML

```
    public MyView(Context context, AttributeSet attrs, int defStyleAttr) {  
        super(context, attrs, defStyleAttr);  
    }
```

} when inflating a view from XML and
applying a class-specific base style
from a theme attribute

커스텀 뷰 작성 방법 - 기본 작성 방법

- onDraw() 함수
 - 뷰가 화면에 출력될 때 자동으로 호출
 - 이 함수에서 그릴 내용이 뷰 영역에 출력



- 커스텀 뷰를 레이아웃 XML에 등록할 때는 클래스명만 등록하면 안되고, 전체 패키지명으로 등록해야 함

```
<com.example.user.lab11.MyPlusMinusView
    android:id="@+id/customView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:customTextColor="#0000ff" />
```

커스텀 뷰 작성 방법 - 커스텀 속성 이용

- res/values 폴더 하위에 attrs.xml 파일을 이용하며, <declare-styleable> 태그로 속성을 등록해 주어야 함

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <declare-styleable name="MyView">
        <attr name="customTextColor" format="color" />
    </declare-styleable>
</resources>
```

- attrs.xml 파일에 등록된 속성을 레이아웃 XML에서 사용

```
<?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"
    ...>

    <com.example.user.lab11.MyPlusMinusView
        android:id="@+id/customView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:customTextColor="#0000ff" />

</LinearLayout>
```

커스텀 뷰 작성 방법 - 커스텀 속성 이용

- 생성자 매개변수 중 AttributeSet을 이용하여 속성값을 추출할 수 있음

```
public MyView(Context context, AttributeSet attrs) {  
    super(context, attrs);  
    this.context = context;  
  
    if (attrs != null) {  
        TypedArray a = context.obtainStyledAttributes(attrs, R.styleable.MyView);  
        color = a.getColor(R.styleable.MyView_customTextColor, Color.RED);  
    }  
}
```

- AttributeSet 객체로 속성값을 획득하는 함수
 - int getAttributeCount(): 속성 개수
 - String getAttributeName(int index): 속성명 획득
 - String getAttributeValue(int index): 속성값 획득
 - int getAttributeIntValue(int index, int defaultValue): 속성값 획득
 - boolean getAttributeBooleanValue(int index, boolean defaultValue): 속성값 획득
 - float getAttributeFloatValue(int index, float defaultValue): 속성값 획득

```
for (int i = 0; i < attrs.getAttributeCount(); i++) {  
    attributes[i] = attrs.getAttributeName(i) + "=" + attrs.getAttributeValue(i);  
}
```

커스텀 뷰 작성 방법 - 크기 결정

- onMeasure() 함수
 - 뷰 내부에서 크기 결정을 위해 호출
 - onDraw() 함수 호출 전에 자동으로 호출

```
@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
    super.onMeasure(widthMeasureSpec, heightMeasureSpec);
    setMeasuredDimension(500, 500);
}
```

setMeasuredDimension() 함수를 이용해 지정한 값이 이 뷰의 크기가 됨.
액티비티 레이아웃 XML에서 "wrap_content", "match_parent" 혹은 수치로
직접 주든 전혀 적용되지 않고 500픽셀로만 나옴.

- 레이아웃 XML 파일의 크기 설정 정보는 onMeasure() 함수의 매개변수
 - 모드와 크기를 획득할 수 있음

```
@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
    int widthMode = MeasureSpec.getMode(widthMeasureSpec);
    int widthSize = MeasureSpec.getSize(widthMeasureSpec);

    int heightMode = MeasureSpec.getMode(heightMeasureSpec);
    int heightSize = MeasureSpec.getSize(heightMeasureSpec);
    //.....
}
```

커스텀 뷰 작성 방법 - 크기 결정

- 모드

- MeasureSpec.AT_MOST

- 뷰 내부에서 지정하라는 의미. 레이아웃 XML에서 wrap_content로 선언한 경우

- MeasureSpec.EXACTLY

- 뷰를 이용하는 액티비티 쪽에서 크기를 결정한 경우. 레이아웃 XML에서 match_parent, 100px 등으로 선언한 경우

- MeasureSpec.UNSPECIFIED

- 모드가 설정되지 않았을 경우

- 모드를 참조해서 적절한 알고리즘으로 뷰 자신의 크기를 결정해 주면 됨

```
if (heightMode == MeasureSpec.AT_MOST) {  
    height = 200;  
} else if (heightMode == MeasureSpec.EXACTLY) {  
    height = heightSize;  
}  
  
setMeasuredDimension(width, height);
```

커스텀 뷰 작성 방법 - 이벤트 추가

① 인터페이스 정의

```
public interface OnMyChangeListener {  
    void onChange(int value);  
}
```

② 인터페이스 구현

```
public class MainActivity extends  
    AppCompatActivity implements  
    OnMyChangeListener{  
    @Override  
    protected void onCreate(Bundle  
        savedInstanceState) {  
        // 인터페이스를 구현한 객체를 View에  
        // 등록  
        plusMinusView.setOnMyChangeListener(this);  
    }  
    @Override  
    public void onChange(int value) {  
    }  
}
```

③ 객체 등록

```
public class MyPlusMinusView extends View {  
    //Observer를 등록하기 위한 객체  
    ArrayList<OnMyChangeListener> listeners;  
  
    //observer 등록을 위한 함수  
    public void setOnMyChangeListener(OnMyChangeListener  
        listener){  
        listeners.add(listener);  
    }  
    //사이즈 결정  
    @Override  
    public boolean onTouchEvent(MotionEvent event) {  
        //데이터 변경  
        value++;  
        //화면 갱신  
        invalidate();  
    }  
    ④ 이벤트 발생  
    for(OnMyChangeListener listener : listeners){  
        //observer에게 데이터 전달  
        listener.onChange(value);  
    }  
    ⑤ 등록 객체의 함수 호출  
    @Override  
    protected void onDraw(Canvas canvas) {  
    }  
}
```