

안드로이드 튜토리얼

제 2장 레이아웃



동아리 ALOM

01/ Constraint
Layout

02/ Linear
Layout

03/ Relative
Layout

04/ 그 외
Layout

05/ 혼자
해보기





ConstraintLayout은 레이아웃에 배치되는 뷰들에 여러 제약 (Constraint)을 적용하여 각 뷰의 위치와 크기를 결정한다.

여기서 말하는 제약(Constraint)이란, 각 요소들의 최종 위치와 크기를 결정하게 될 조건이다. 각 제약조건은 다른 보기, 상위 레이아웃 또는 표시되지 않는 안내선을 기준으로 한 정렬 또는 연결을 나타낸다.

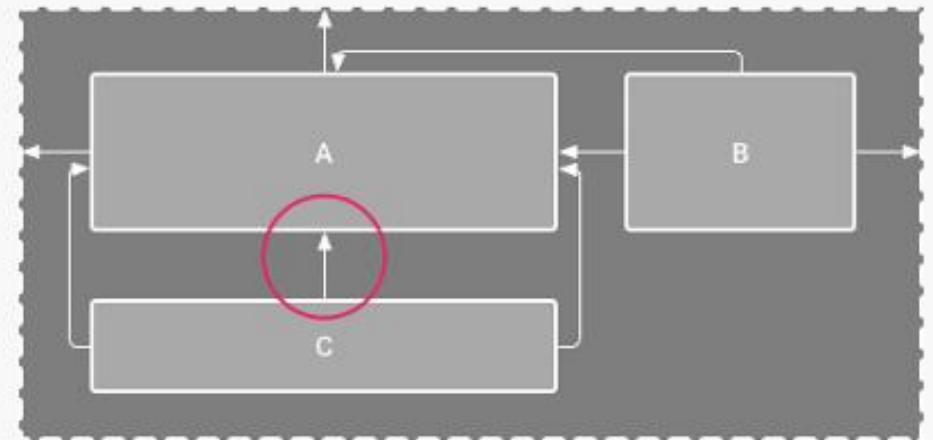
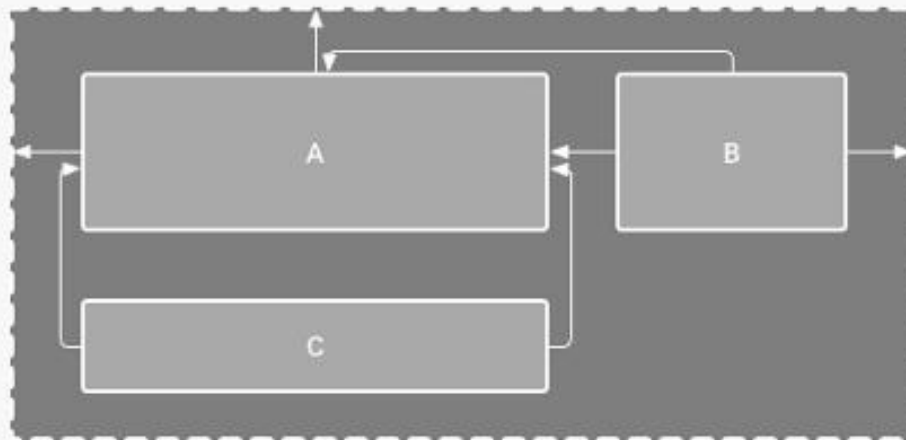
각 제약조건은 세로 또는 가로 축을 따라 보기의 위치를 정의하므로, 각 보기에는 축마다 하나 이상의 제약조건이 있어야 하며, 흔히 더 많이 필요하다.





1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



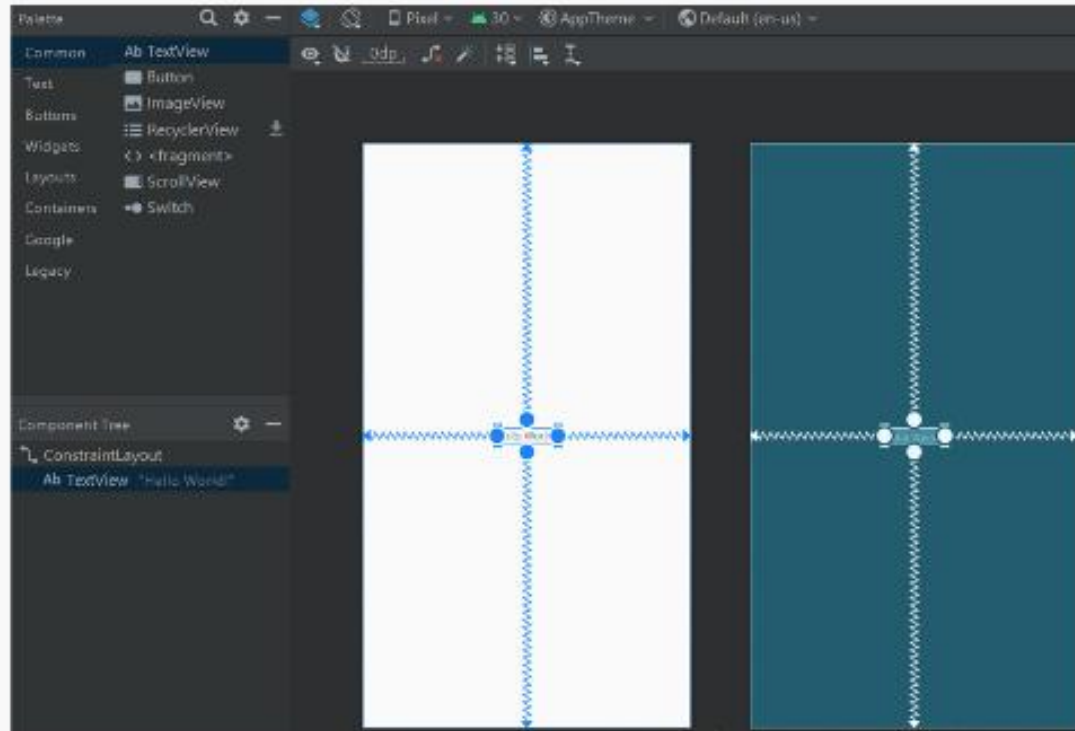
여기서 붉은 원으로 표시한 화살표가 제약조건이다.
보기 C는 보기 A 아래에 세로로 제약을 걸어 위치를 결정한다.





1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



프로젝트가 생성되면
레이아웃은 기본적으로
ConstraintLayout이다.

가운데 textview를 눌러보면,
상하좌우로 부모 레이아웃에
제약이 걸려있는 것을 확인 할
수 있다.

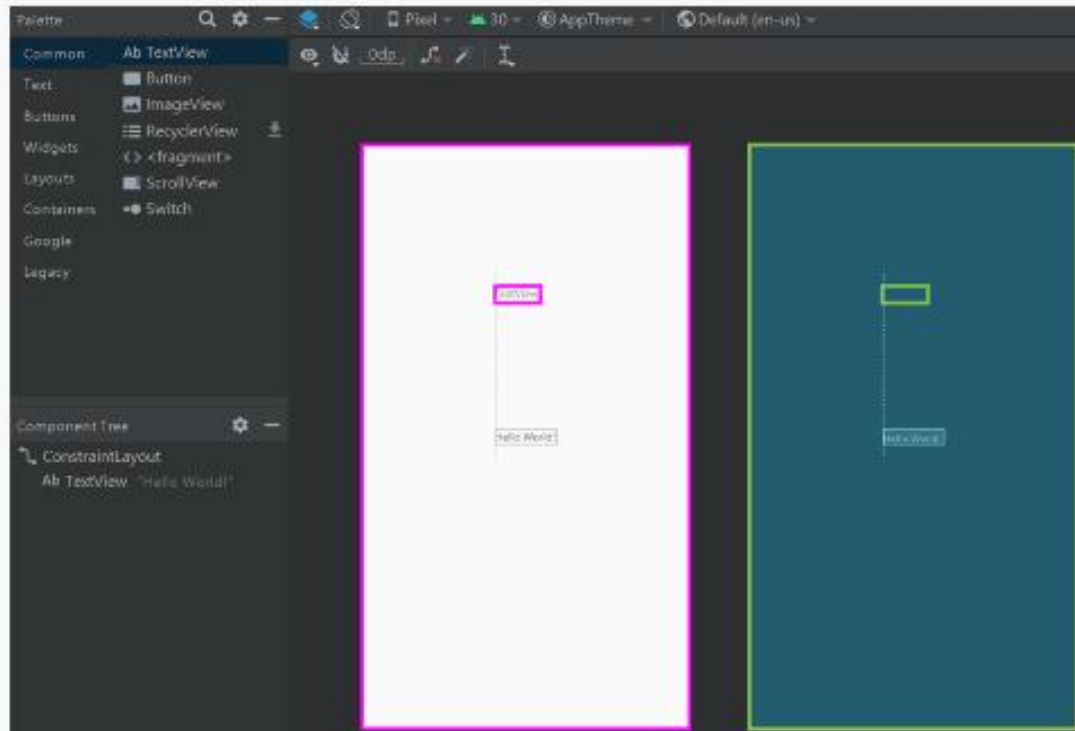
텍스트뷰가 부모 레이아웃의 면
에서 얼마큼 씩 떨어져 있음을
나타내 주는 제약조건이다.





1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



왼쪽 팔레트에서는 기본적으로 제공되는 위젯들을 사용할 수 있다. TextView를 드래그하여 화면 위에 놓으면, textview가 생성된다.

이제 제약을 걸어서 textview의 위치를 특정해주면 끝이다.

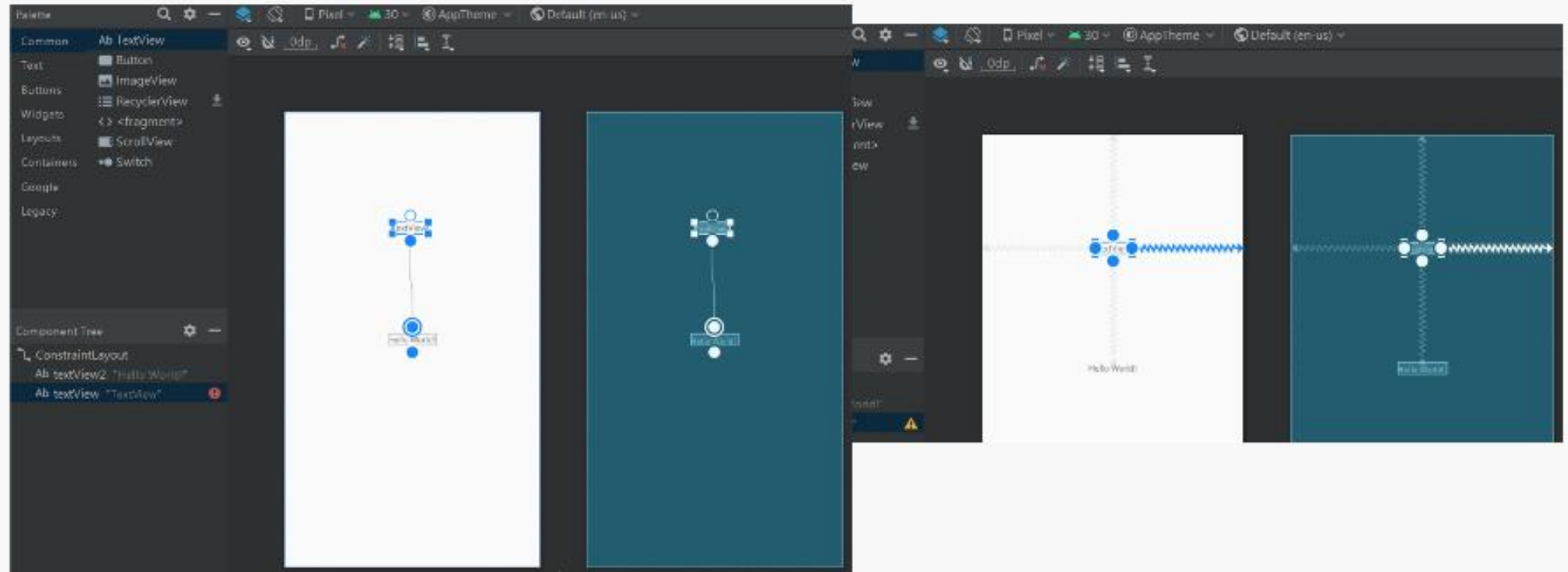
원래 존재하는 textview 위에, 옆에,아래 등 제약 조건에 따라 다양하게 위치시킬 수 있다.





1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



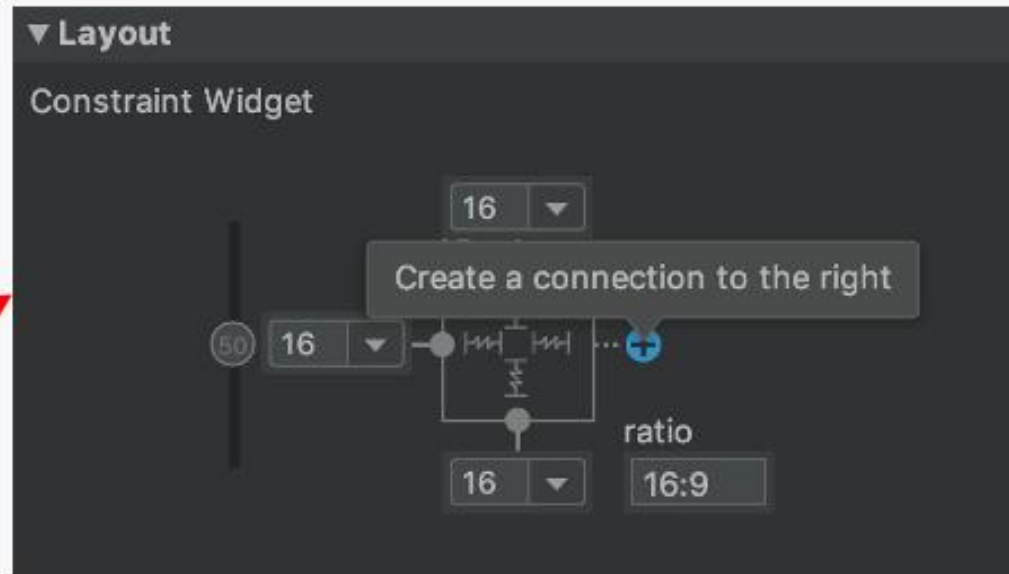
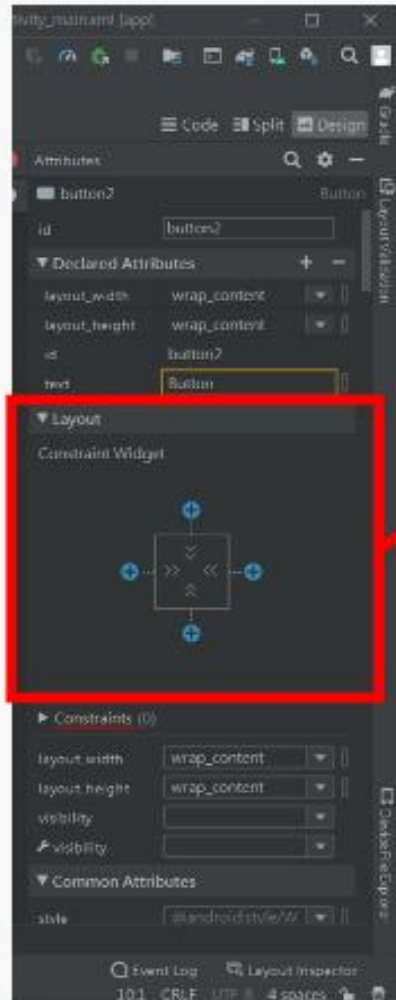
Textview를 눌렀을 때 나오는 파란 원을 선택하여
다른 위젯 혹은 부모 레이아웃까지 드래그하여 이어준다.
상하좌우 4방향을 다 이어주면 완료이다.






1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



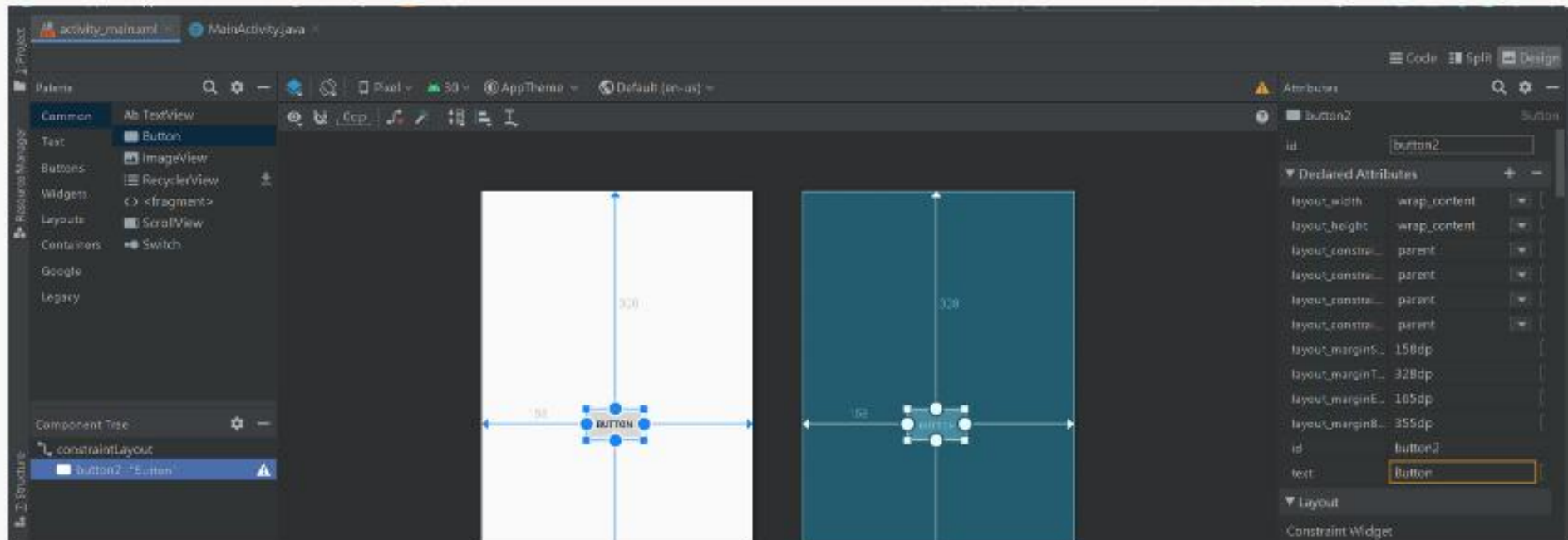
Attributes 창의 Layout을 사용하여 연결을 만들 수도 있다.
위 그림에 표시된 대로 Attributes 창의 Layout 섹션에서
Create a connection  버튼 중 하나를 클릭한다.





1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



왼쪽 Component Tree에서 textView를 삭제하고
Palette에서 Button을 끌어와 레이아웃에 연결한다.
그 다음 우측 상단에 Code 탭으로 넘어가보자.





1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



The screenshot shows the XML code for an Android layout in Android Studio. The code defines a `ConstraintLayout` and a `Button` inside it. Annotations with arrows point to specific attributes in the code, explaining their functions:

- “match_parent”** : 부모가 가지는 길이를 전부 채운다.
This annotation points to `android:layout_width="match_parent"` and `android:layout_height="match_parent"` in the `ConstraintLayout` tag.
- “wrap_content”** : 해당 뷰가 그려질 수 있게 필요한 길이만큼 차지한다.
This annotation points to `android:layout_width="wrap_content"` and `android:layout_height="wrap_content"` in the `Button` tag.
- 마진** : 해당 뷰와 타겟 뷰 사이에 값만큼의 여백을 생성한다.
This annotation points to the margin attributes in the `Button` tag: `android:layout_marginStart="158dp"`, `android:layout_marginTop="328dp"`, `android:layout_marginEnd="165dp"`, and `android:layout_marginBottom="355dp"`.
- 이 버튼의 제약조건이 어디에 연결되어 있는지 보여준다.**
This annotation points to the `app:layout_constraint` attributes in the `Button` tag, which define the button's position relative to the parent layout.
- “parent”**는 부모 레이아웃을 뜻한다.
This annotation points to the value `"parent"` used in the `app:layout_constraint` attributes.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    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:id="@+id/constraintLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.buttonapp.MainActivity">
    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="158dp"
        android:layout_marginTop="328dp"
        android:layout_marginEnd="165dp"
        android:layout_marginBottom="355dp"
        android:text="button"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```





1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



The screenshot shows the Android Studio interface. On the left, the 'File' menu is open, and the path 'File > New > XML > Layout XML' is highlighted with a red box. On the right, the 'Configure Component' dialog is open, and the 'Layout File Name' field is highlighted with a red box, containing the text 'activity_constraint'. The 'Root Tag' field contains the text 'androidx.constraintlayout.widget.ConstraintLayout'. Overlaid on the right side of the screenshot is Korean text explaining the steps to create a new layout file.

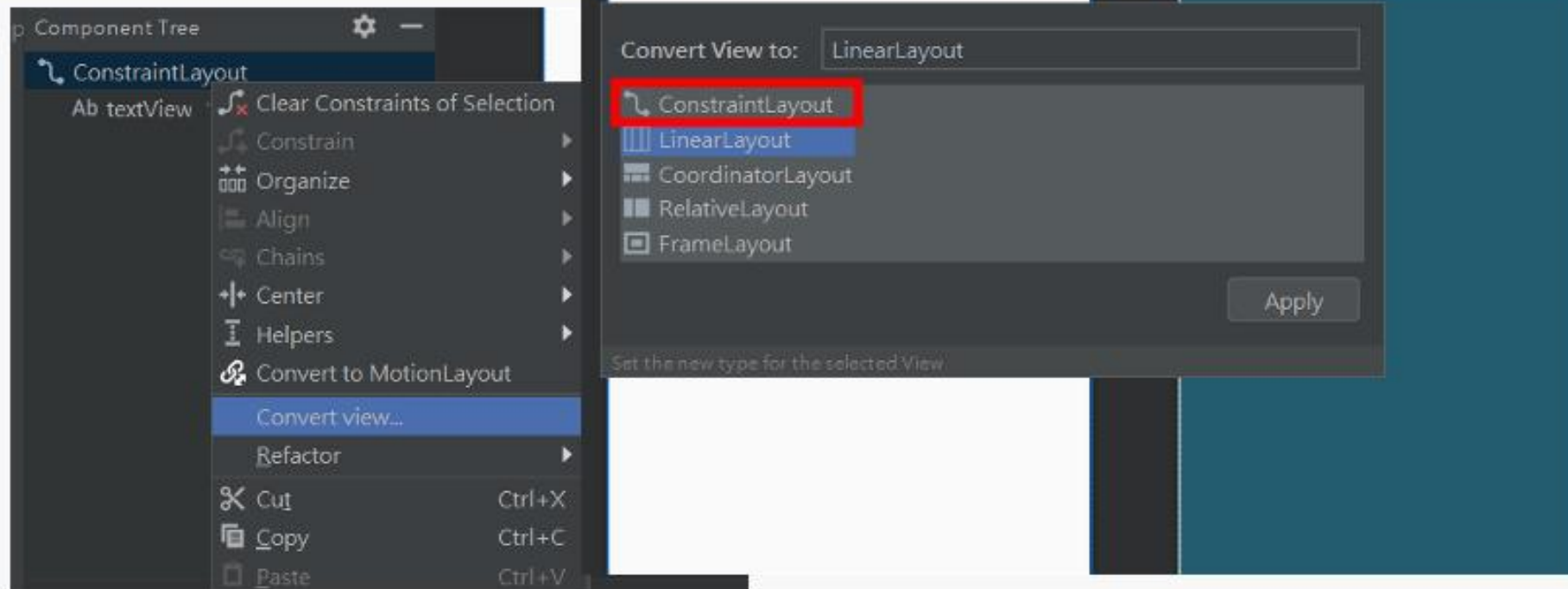
새로운 Layout 파일을 작성하는 법
Project 창에서 모듈 폴더를 클릭한 후
File > New > XML > Layout XML을 선택
레이아웃 파일의 이름을 입력하고 Root Tag
에 'androidx.constraintlayout.widget.ConstraintLayout' 입력





1. ConstraintLayout

TITLE | TITLE | TITLE | TITLE | TITLE



기존 레이아웃을 제약조건 레이아웃으로 변환하려면
Design 탭이 선택된 상태에서 왼쪽 하단
Component Tree 창에 레이아웃을 마우스 우클릭
Convert layout to ConstraintLayout클릭





<https://developer.android.com/training/constraint-layout?hl=ko>

공식 문서를 확인하여 더 자세한 사용방법을 확인 할 수 있다.





LinearLayout이란?

LinearLayout은 세로 또는 가로의 단일 방향으로
모든 하위 요소를 정렬하는 뷰 그룹이다.

Android:orientation 특성을 사용하여
레이아웃 방향을 지정할 수 있다.

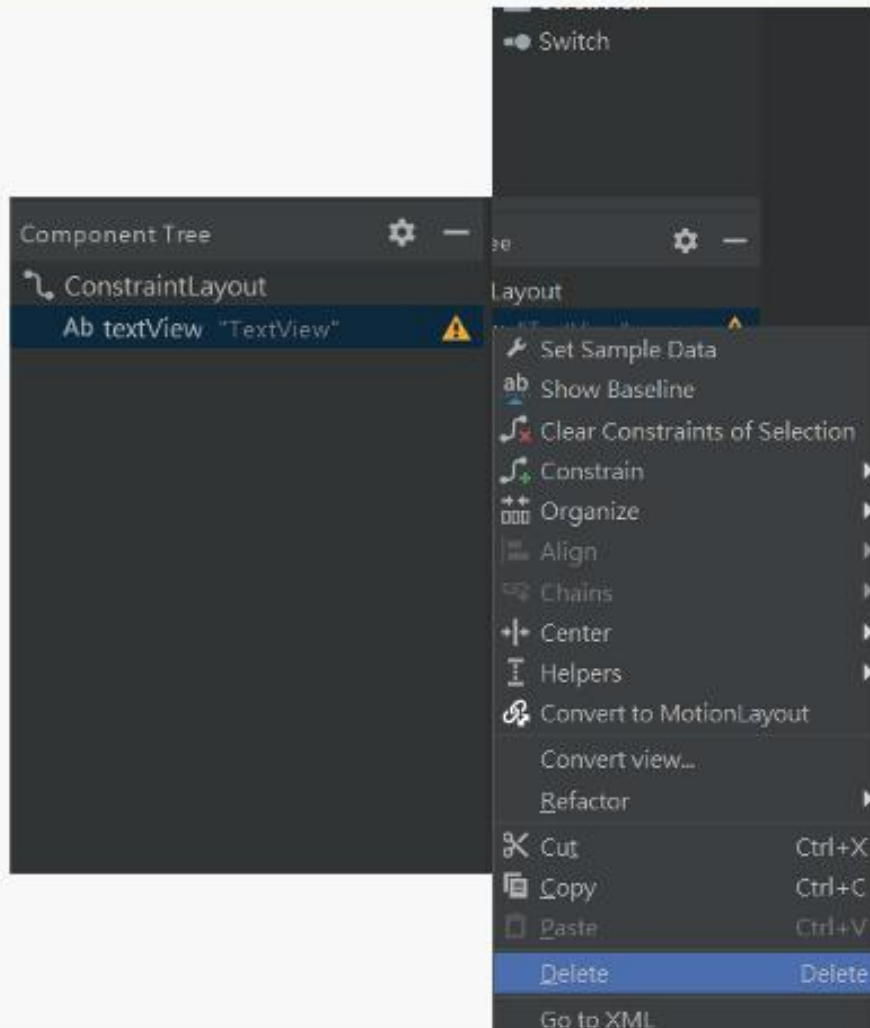
수직으로 정렬하는 LinearLayout(Vertical)과
수평으로 정렬하는 LinearLayout(Horizontal)
두가지가 존재한다.





2. LinearLayout

TITLE | TITLE | TITLE | TITLE | TITLE



LinearLayout 생성방법
기본 레이아웃인
ConstraintLayout일 때

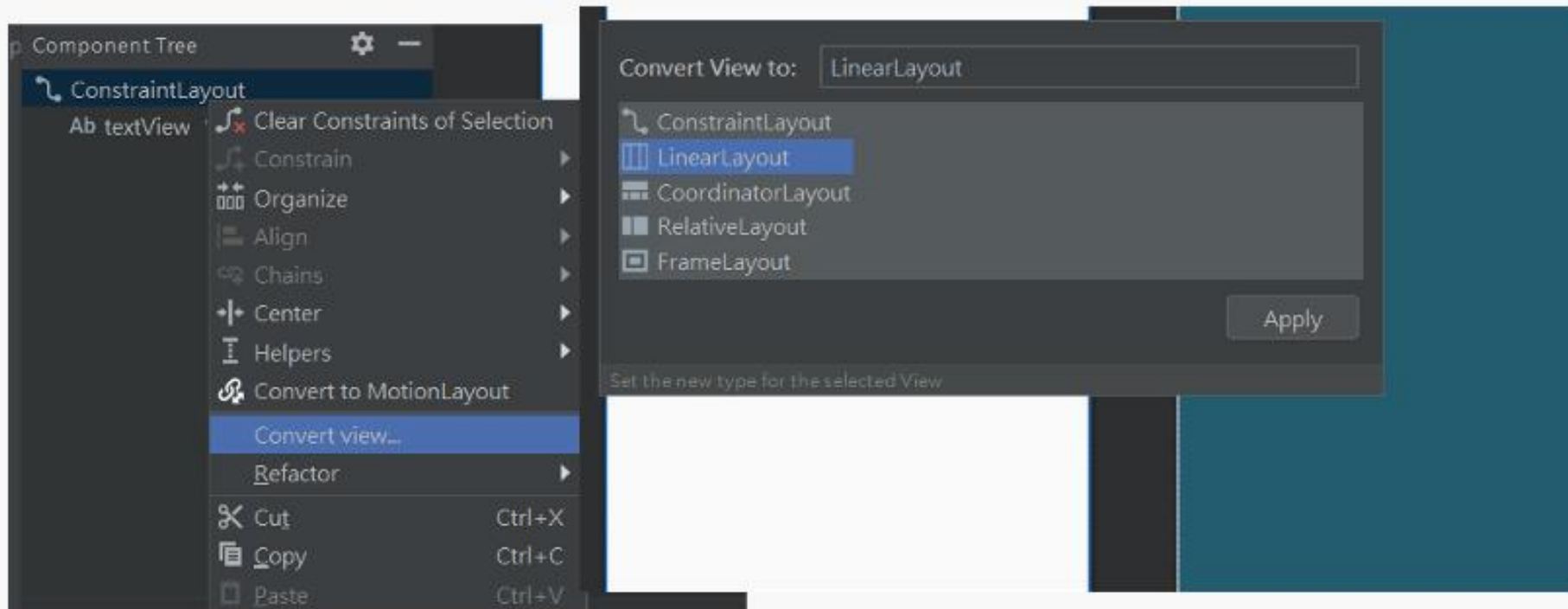
Component Tree에서
ConstraintLayout 아래
Textview 우클릭하여 Delete





2. LinearLayout

TITLE | TITLE | TITLE | TITLE | TITLE



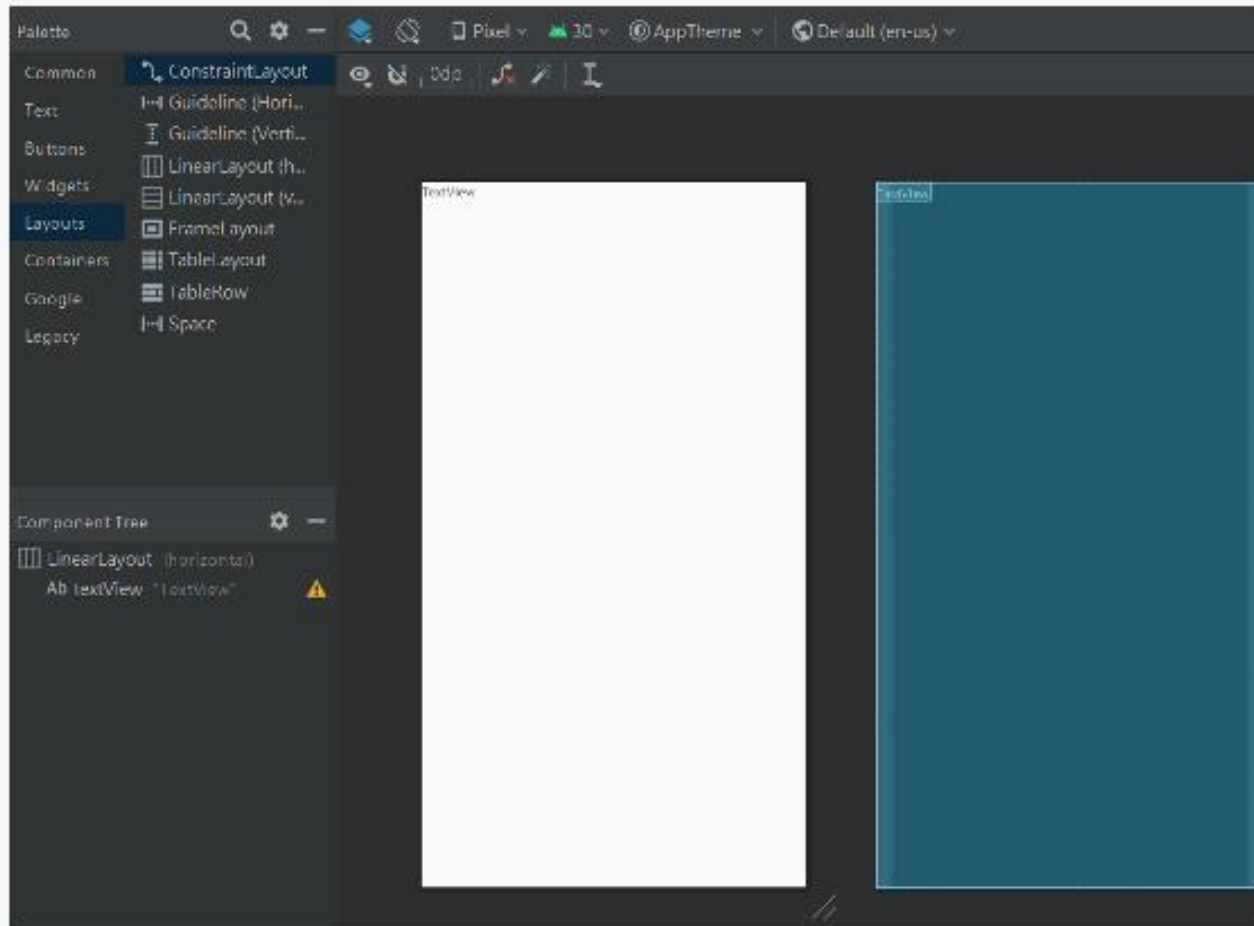
ConstraintLayout 우클릭 -> Convert view
Convert View 에서 LinearLayout 클릭





2. LinearLayout

TITLE | TITLE | TITLE | TITLE | TITLE



LinearLayout으로
변경된 모습

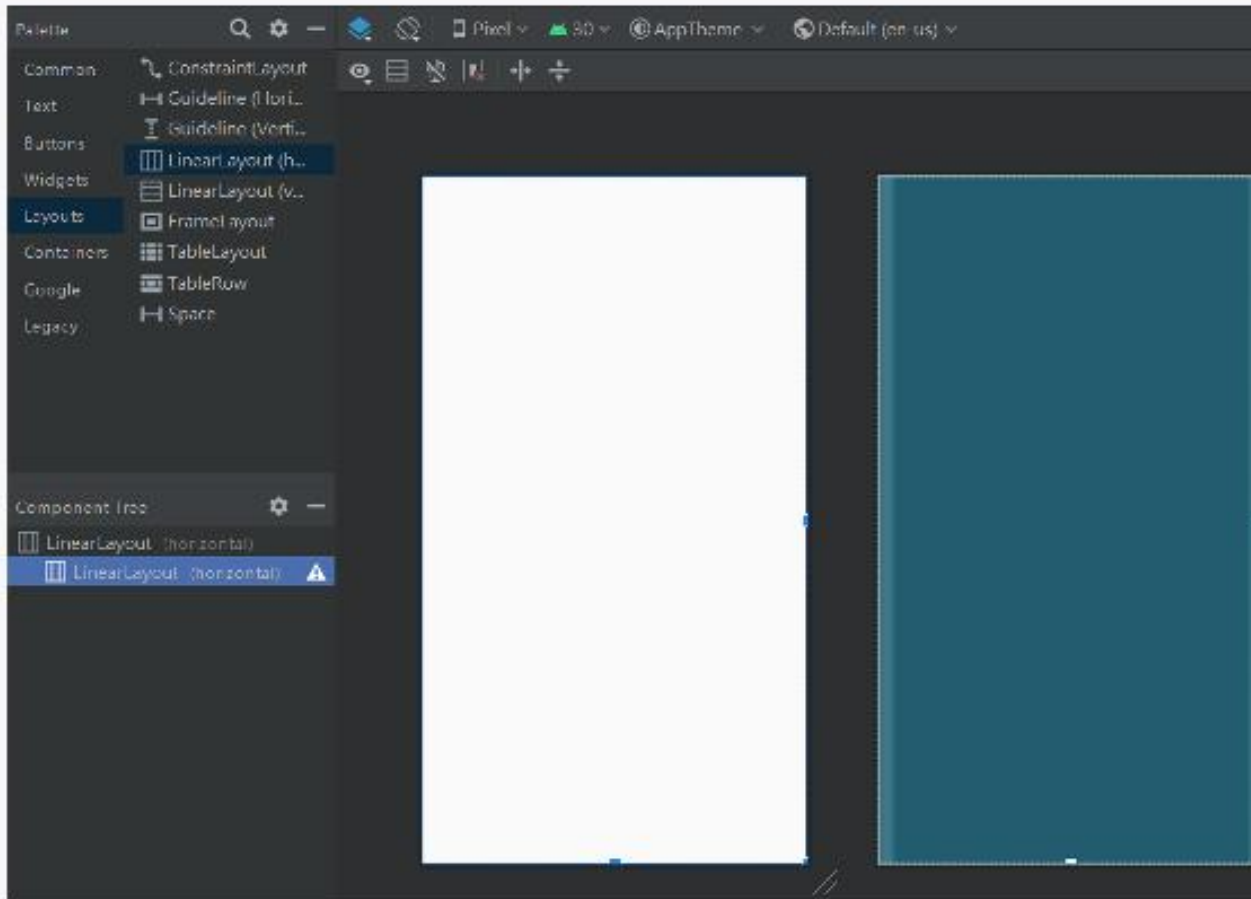
팔레트에 레이아웃 탭에서
해당하는 레이아웃을
끌어올 수 도 있다.





2. LinearLayout

TITLE | TITLE | TITLE | TITLE | TITLE



팔레트에 레이아웃 탭에서
리니어 레이아웃을
끌어넣었다.

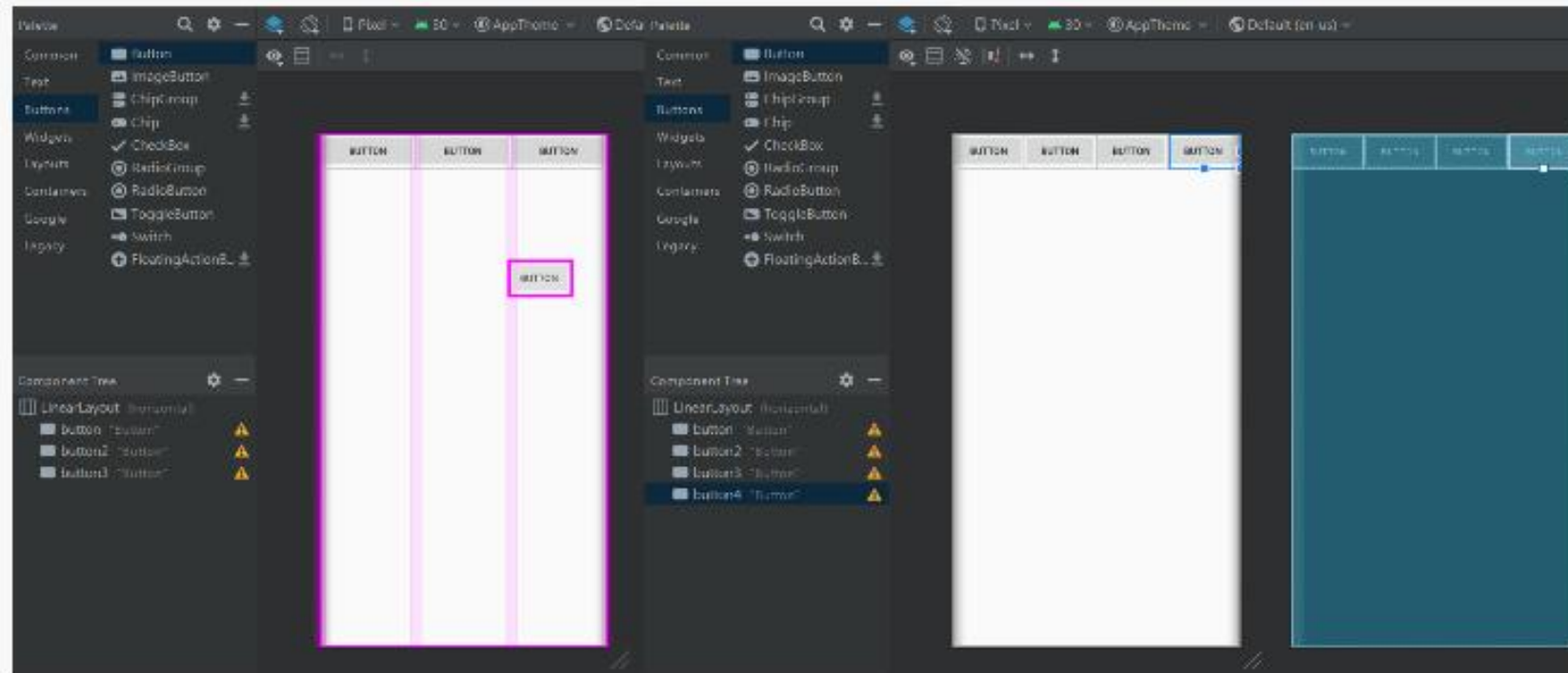
이처럼 레이아웃 안에
레이아웃을 넣을 수도 있다.
(부모 레이아웃과
자식 레이아웃)





2. LinearLayout

TITLE | TITLE | TITLE | TITLE | TITLE



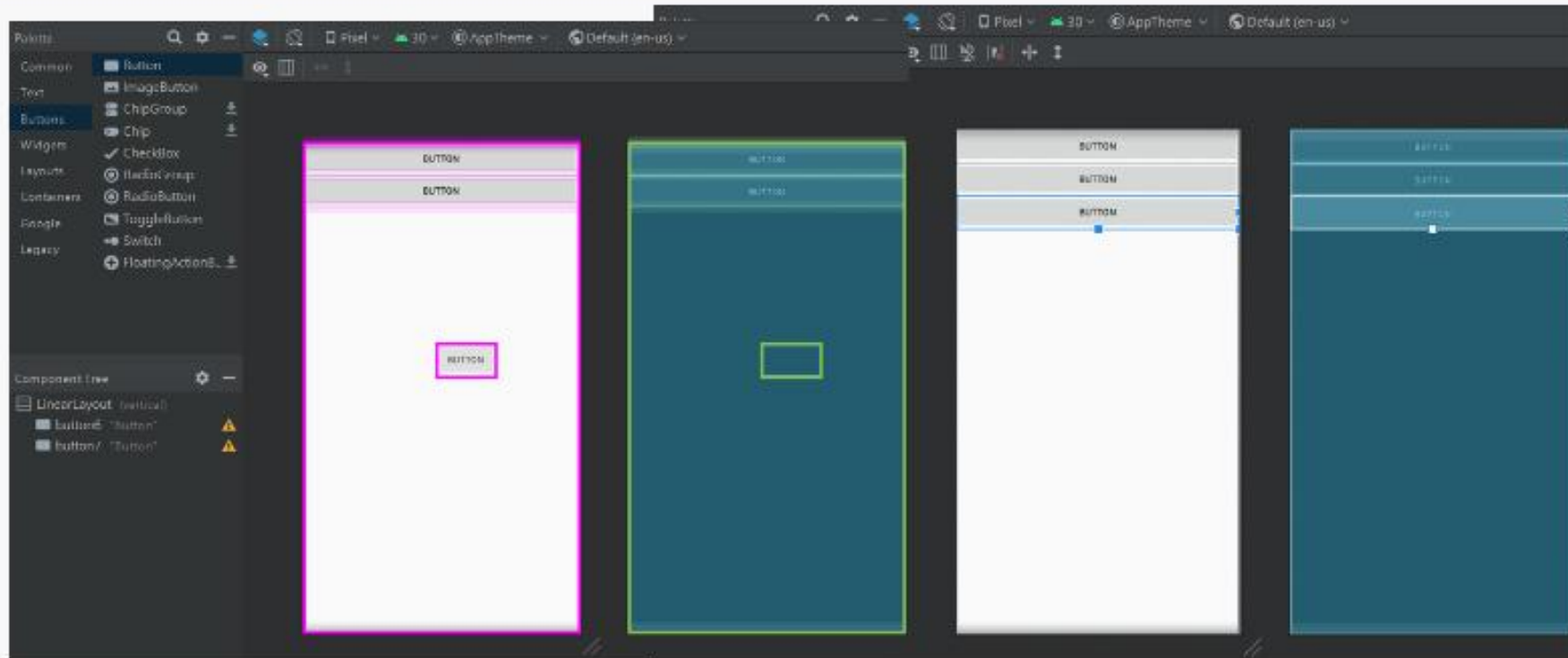
LinearLayout(horizontal)의 특징
레이아웃 안에 수평으로 위젯들이 쌓인다





2. LinearLayout

TITLE | TITLE | TITLE | TITLE | TITLE



LinearLayout(vertical)의 특징
레이아웃 안에 수직으로 위젯들이 쌓인다.





LinearLayout을 이해하려면
레이아웃이 가지는 특징이 잘 나타나는
화면을 직접 구성해보면 된다.

<https://blog.naver.com/eominsuk55/220222898504>

좋은 예시 이다. 처음엔 바로 보고 만들기 어려우니
7~ 13까지의 강좌를 읽고 따라한 후, 만들어본다.
이후 모범 답안을 확인하고 똑같이 만든 후
레이아웃을 직접 눈으로 확인하고 이해한다.





<https://developer.android.com/guide/topics/ui/layout/linear.html>

공식 문서를 확인하여 더 자세한 사용방법을 확인 할 수 있다.





상대 레이아웃(RelativeLayout)

제약 레이아웃(ConstraintLayout)이 등장하
기 전까지 활발하게 쓰이던 레이아웃이었으나
제약 레이아웃의 등장으로 대체되었다.
이제는 잘 사용되지 않는다.





프레임 레이아웃(FrameLayout)
테이블 레이아웃(TableLayout)
그리드 레이아웃(GridLayout)

등등의 레이아웃들은 사용 빈도도 적고
지금은 잘 사용하지 않는다.
만약 필요해지면, 그때 검색해서 학습하면 된다.



android



DO IT YOURSELF

〈계산기 화면 구성하기〉

입력창 2개 (EditText)
연산 버튼 4개 (Button)
결과창 1개 (TextView)

