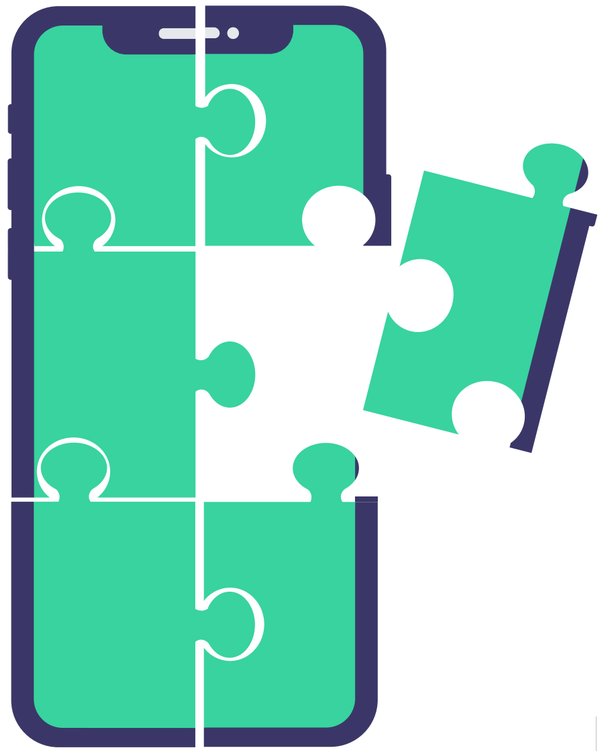
Android App Programming

04.LinearLayout & 뷰 속성

화면 구성하기 (LinearLayout)
뷰 속성 사용하기



android

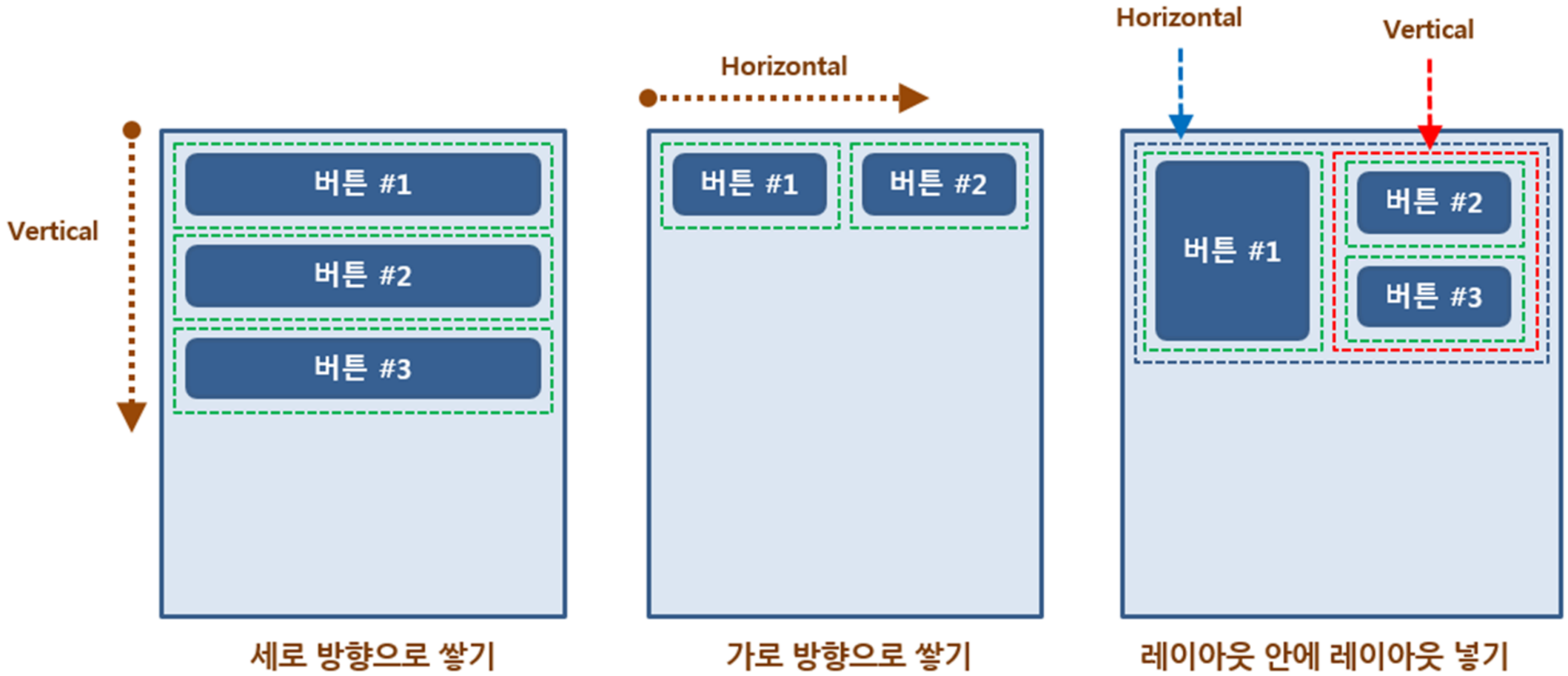
Kym

레이아웃 종류

레이아웃 이름	설 명
제약 레이아웃 (ConstraintLayout)	제약 조건(Constraint) 기반 모델 제약 조건을 사용해 화면을 구성하는 방법 안드로이드 스튜디오에서 자동으로 설정하는 디폴트 레이아웃
리니어 레이아웃 (LinearLayout)	박스(Box) 모델 한 쪽 방향으로 차례대로 뷰를 추가하며 화면을 구성하는 방법 뷰가 차지할 수 있는 사각형 영역을 할당
상대 레이아웃 (RelativeLayout)	규칙(Rule) 기반 모델 부모 컨테이너나 다른 뷰와의 상대적 위치로 화면을 구성하는 방법
프레임 레이아웃 (FrameLayout)	싱글(Single) 모델 가장 상위에 있는 하나의 뷰 또는 뷰그룹만 보여주는 방법 여러 개의 뷰가 들어가면 중첩하여 쌓게 됨. 가장 단순하지만 여러 개의 뷰를 중첩한 후 각 뷰를 전환하여 보여주는 방식으로 자주 사용함
테이블 레이아웃 (TableLayout)	격자(Grid) 모델 격자 모양의 배열을 사용하여 화면을 구성하는 방법 HTML에서 많이 사용하는 정렬 방식과 유사하지만 많이 사용하지는 않음

LinearLayout

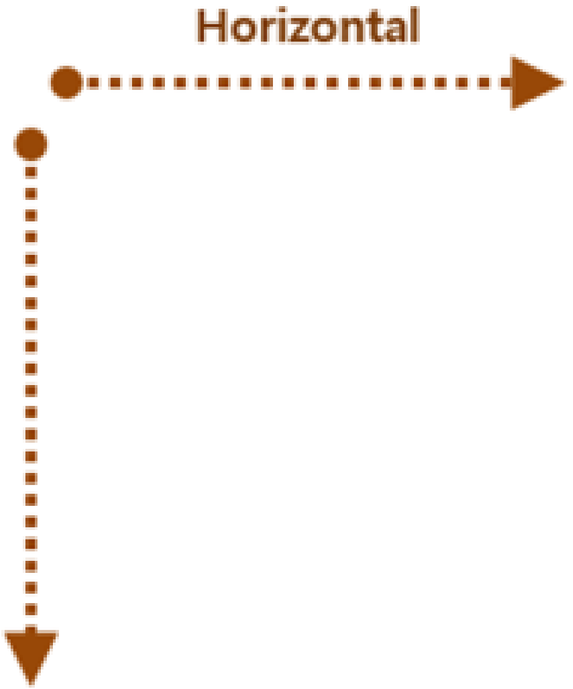
리니어레이아웃 (선형 레이아웃)
왼쪽 위부터 아래쪽 또는 오른쪽으로 차례로 배치



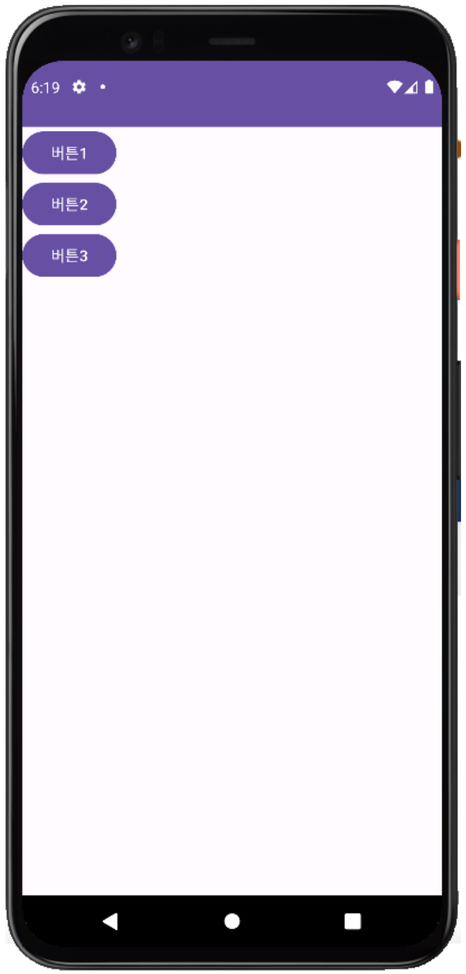
LinearLayout 속성

리니어 레이아웃은 orientation이라는 속성으로 방향성이 결정 되기 때문에 필수 속성
- 가로(horizontal), 세로(vertical)

```
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation= "vertical"
  android:layout_width=“match_parent"
  android:layout_height=“match_parent"
  >
```



LinearLayout 속성 (Orientation)

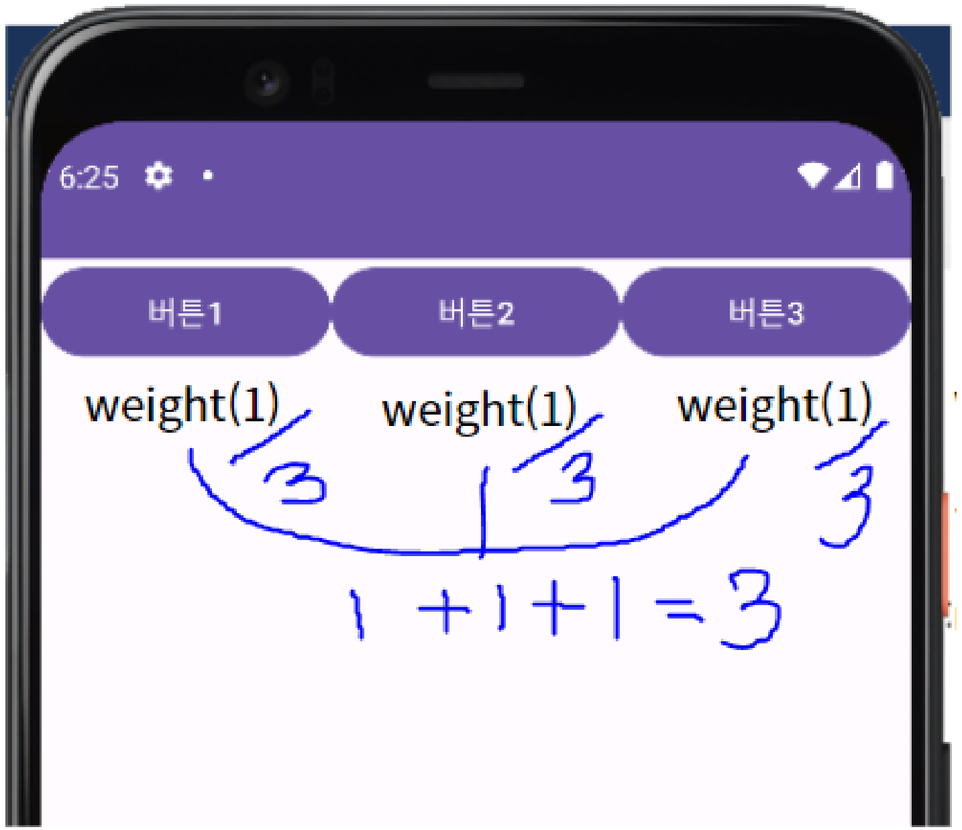


[Vertical 세로의 경우]

[Horizontal 가로의 경우]

LinearLayout 속성 (Weight)

layout_weight 속성은 같은 남아있는 여유공간을 얼마나 차지할 수 있는지를 비율로 지정하는 것
LinearLayout내부에 weight속성을 가진 모든 위젯의 합 (SUM) 과 위젯이 가진 weight를 나누어 계산함.
(처음에 어려움)



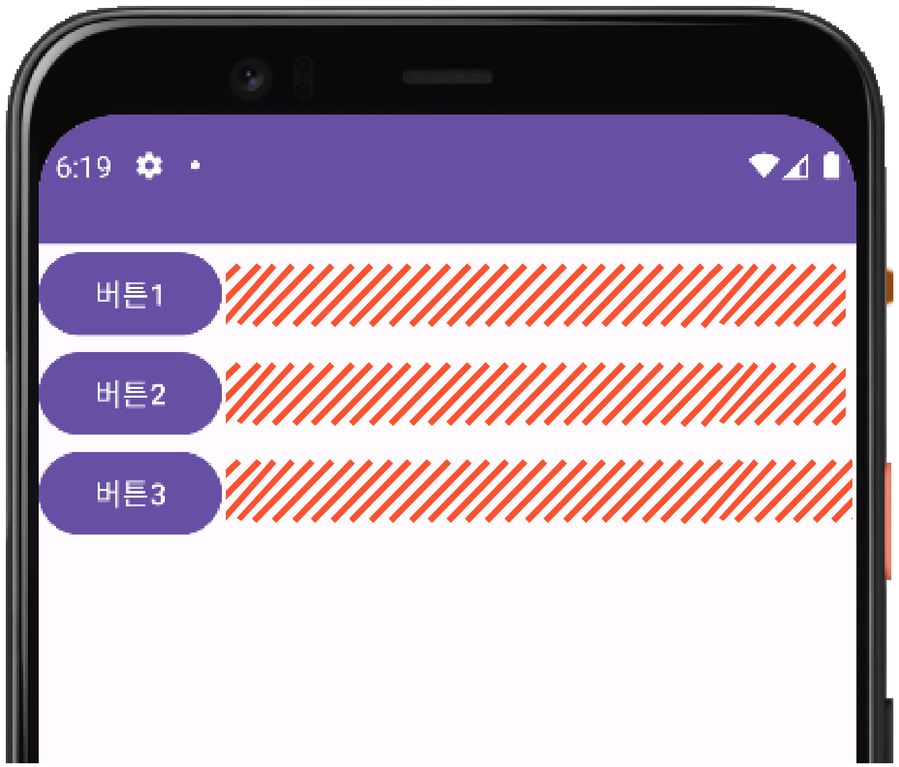
weight총합 : 3

각각의 위젯의 weight : 1

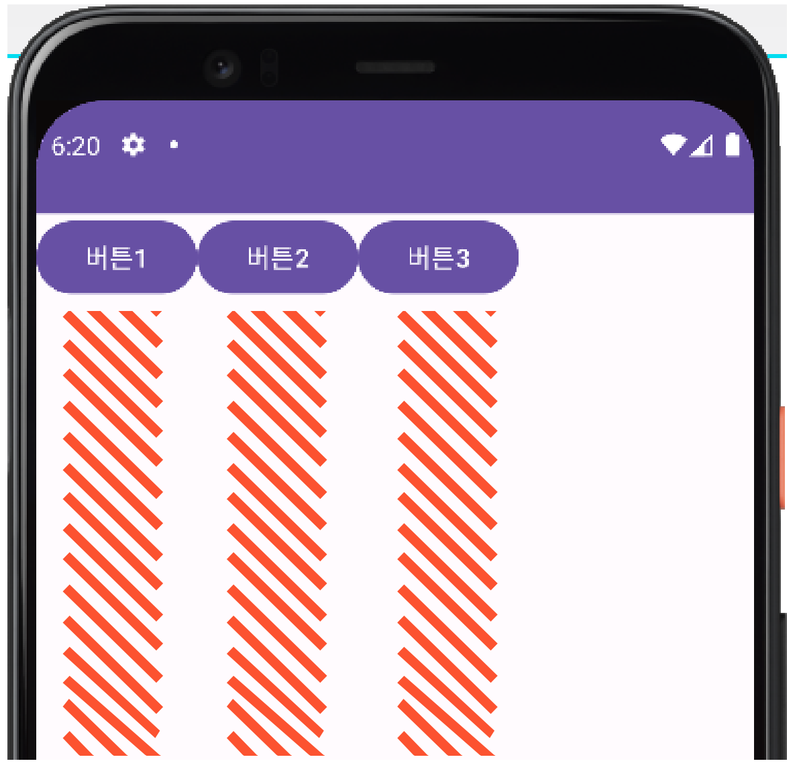
따라서 각각의 위젯은 1/3씩의 넓이를 가짐.

LinearLayout 속성 (Weight)

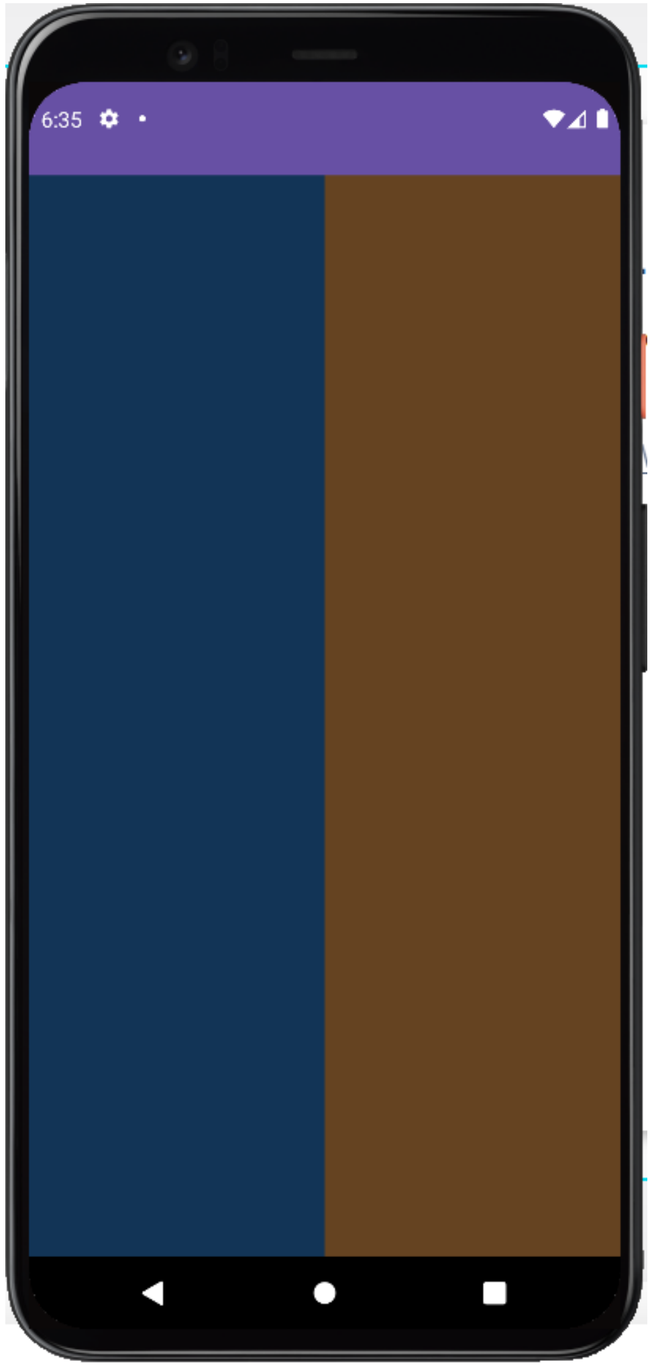
Orientation Vertical(세로)의 경우 좌측에 위젯을 배치할수없기때문에 weight속성으로 높이를 나눔.
Orientation Horizontal(가로)의 경우 밑에 위젯을 배치할수없기때문에 weight속성으로 넓이를 나눔.
정확한 비율을 맞추기위해 0dp속성을 같이 사용함.



vertical의 경우 세로방향으로만 위젯을 쌓기 때문에 옆공간은 비율로 나눌필요가 없음



horizontal의 경우 가로방향으로만 위젯을 쌓기 때문에 아래 공간은 비율로 나눌필요가 없음



실습 **화면 구현 해보기**

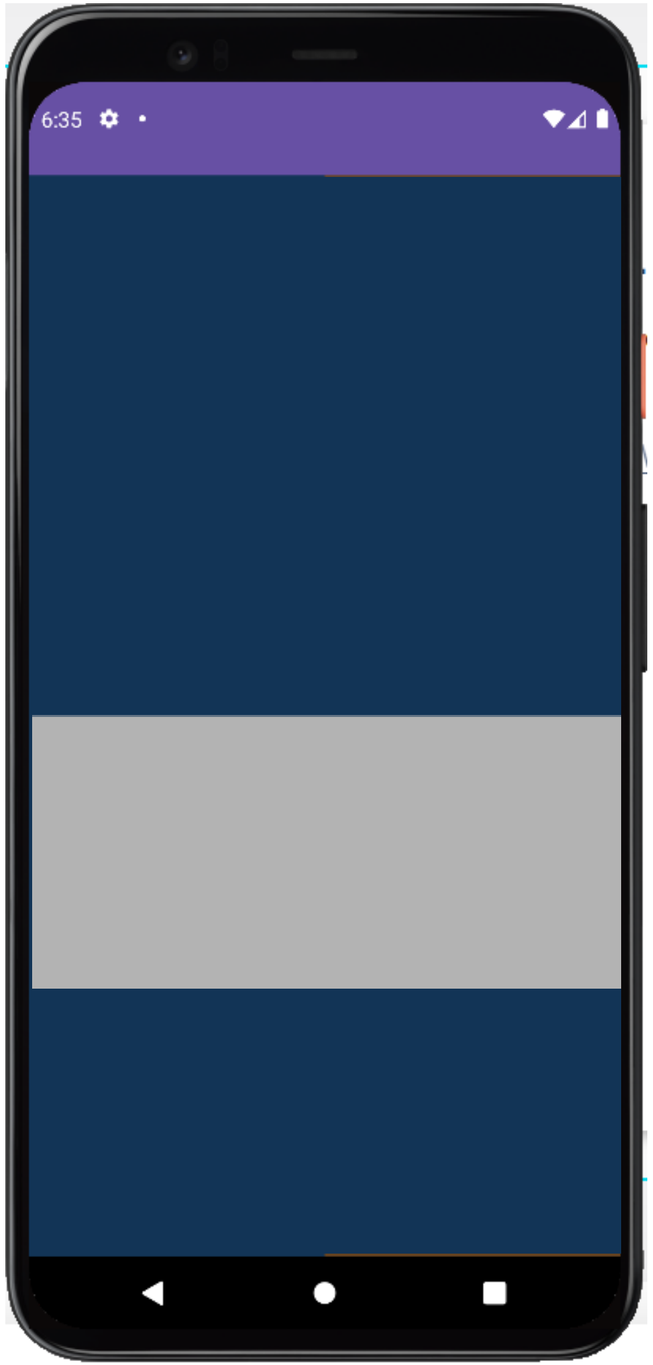
Linear Layout

1.Weight 속성을 이용하여 나누기1

2.Weight 속성을 이용하여 나누기2

3.Layout 중첩시키기

linear1 파일 추가 후 만들어보기
내부에는 <LinearLayout>위젯을 추가하고 backgroud속성을 바꾼다.
weight와 orientation 속성 주의



실습 **화면 구현 해보기**

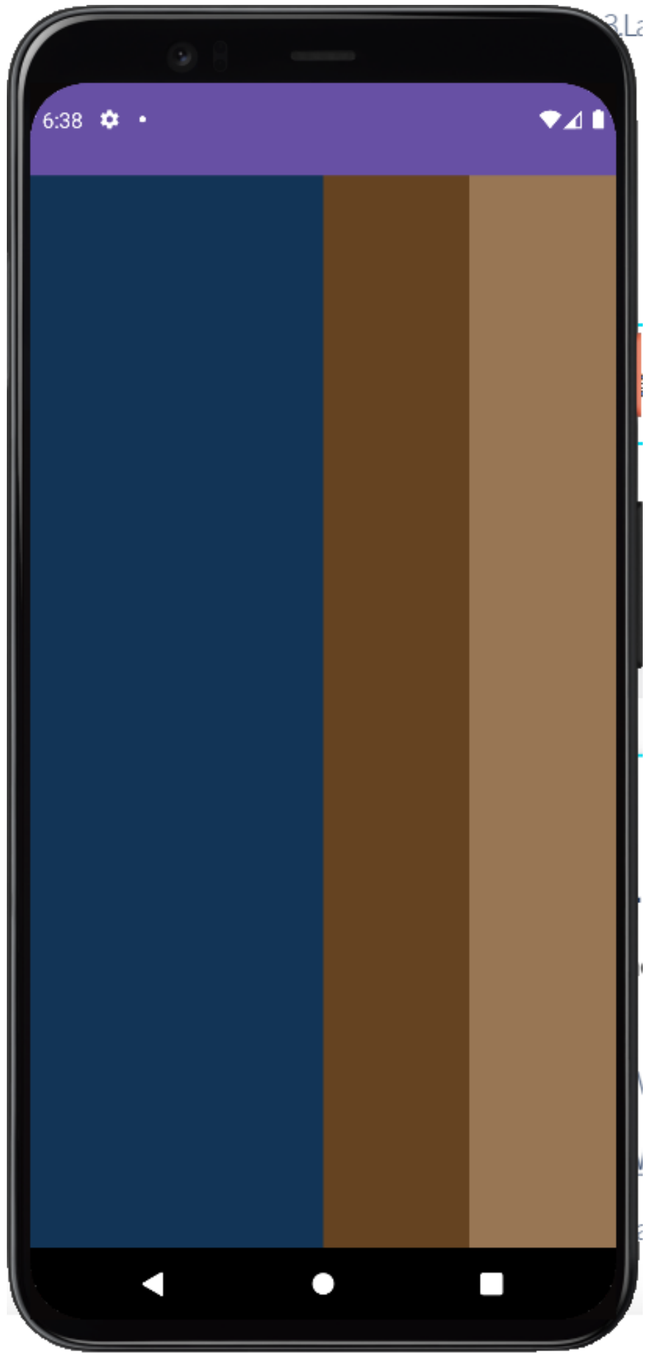
Linear Layout

1.Weight 속성을 이용하여 나누기1

2.Weight 속성을 이용하여 나누기2

3.Layout 중첩시키기

linear2 파일 추가 후 만들어보기
내부에는 <LinearLayout>위젯을 추가하고 backgroud속성을 바꾼다.
weight와 orientation 속성 주의



실습 **화면 구현 해보기**

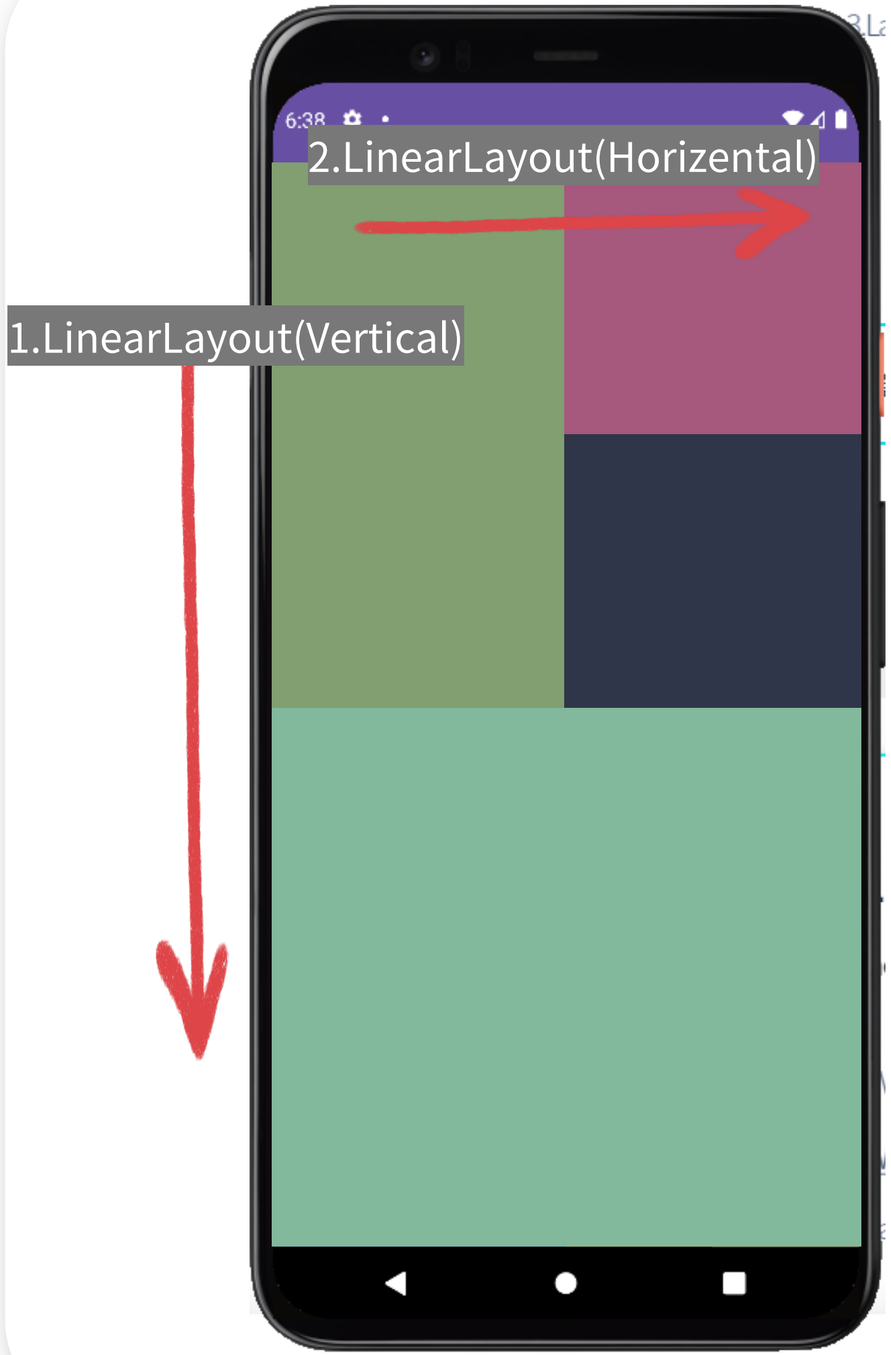
Linear Layout

1.Weight 속성을 이용하여 나누기1

2.Weight 속성을 이용하여 나누기2

3.Layout 중첩시키기

linear3 파일 추가 후 만들어보기



실습

화면 구현 해보기

Linear Layout

1.Weight 속성을 이용하여 나누기1

2.Weight 속성을 이용하여 나누기2

3.Layout 중첩시키기

linear4 파일 추가 후 만들어보기

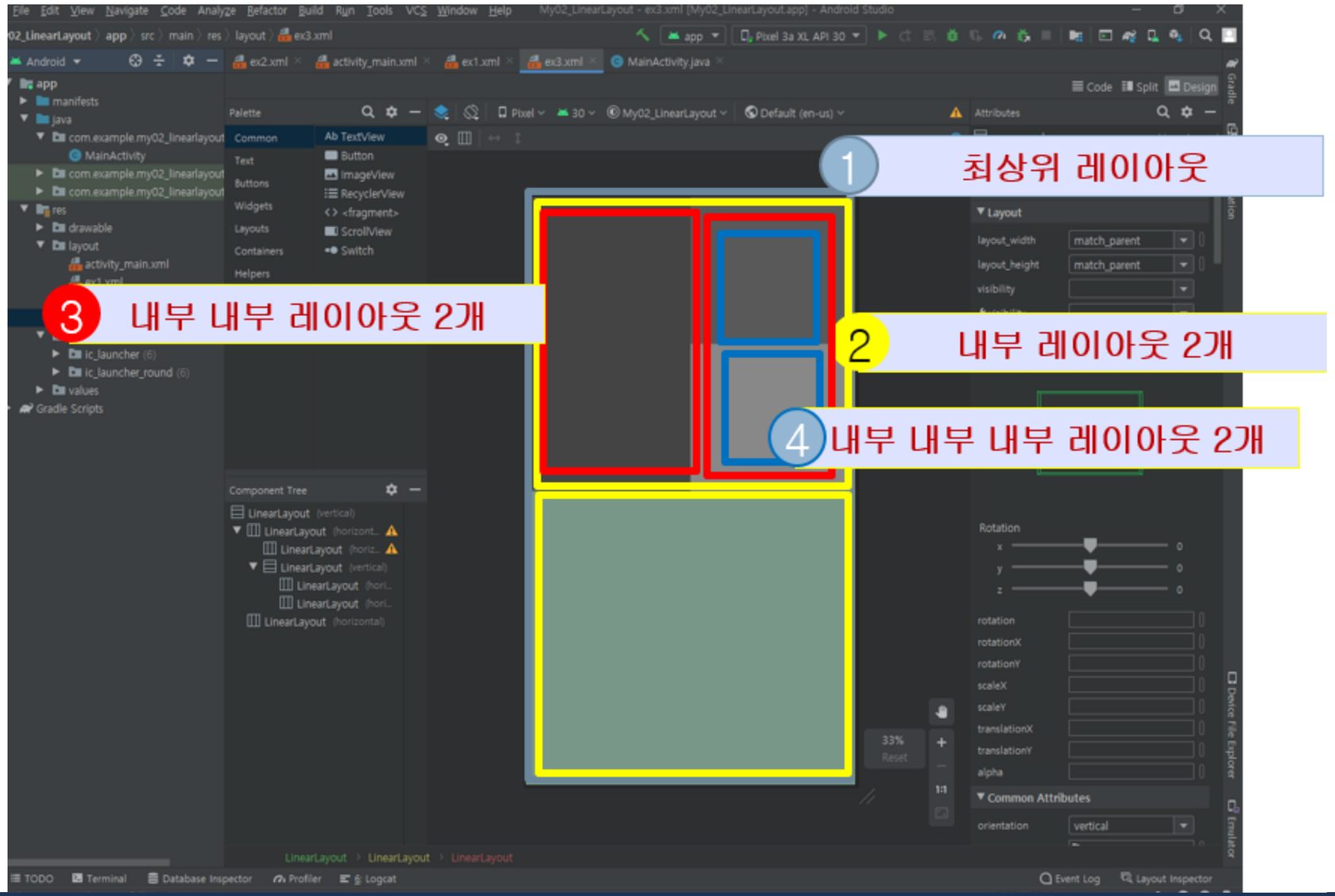
LinearLayout 중첩시키기

세로 방향의 리니어 레이아웃 안에 가로 방향의 리니어 레이아웃을 넣을 수 있음
레이아웃 안에 레이아웃을 계속 넣을 수 있으나 레벨이 너무 깊으면 시스템 부하 증가

```
<LinearLayout android:orientation="horizontal" >
    <ImageView />
    <LinearLayout android:orientation="vertical" >
        <TextView />
        <TextView />
    </LinearLayout>
</LinearLayout>
```

LinearLayout 중첩시키기

Layout종류는 모두 중첩이 가능함.
LinearLayout(최상위 부모) 내부에 LinearLayout을 추가 후 orientation속성을 이용하여 뷰가 쌓이는 방향을 바꿀 수 있음



Android 뷰의 속성들

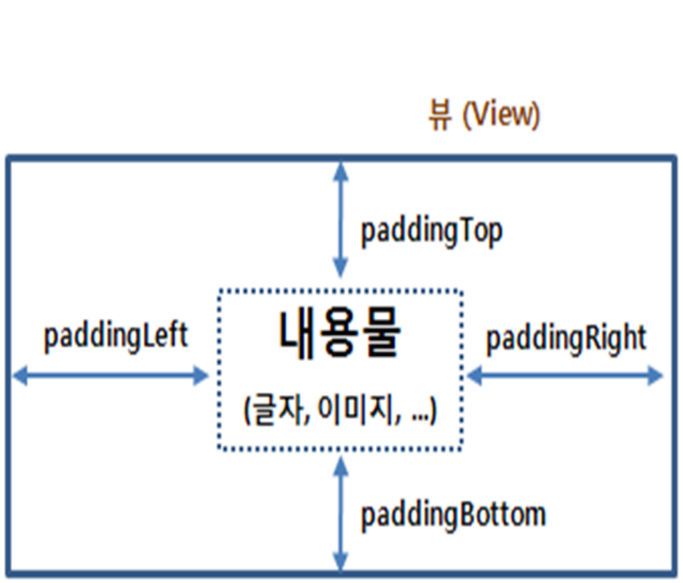
xml의 id : Java코드에서 setContentView메소드가 실행되고 나서 R클래스로 부터 id로 위젯에 접근하는 용도
Layout파일 내부에서는 key값이기때문에 중복 불가

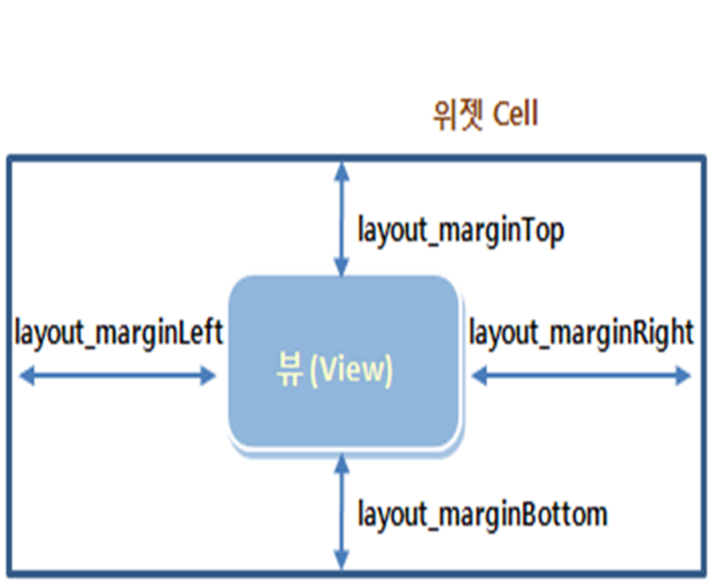
```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp"
    android:orientation="vertical">

    <TextView
        android:id="@+id/name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="텍스트뷰" />
</LinearLayout>
```

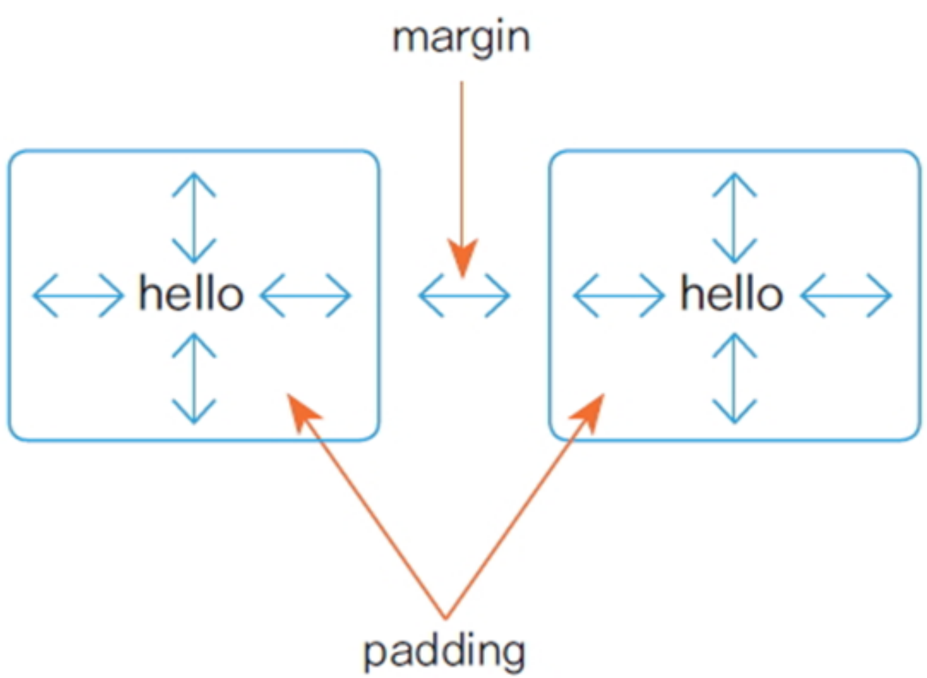
```java
package kr.co.hanbit.firstapp;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.widget.TextView;

public class NewActivity extends AppCompatActivity {

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

        TextView nameText = (TextView) findViewById(R.id.name);
    }
}
```

Android 뷰의 속성들

뷰의 간격 설정
Html과 마찬가지로 간격(여백)은 margin 과 padding을 이용
margin, padding 속성을 이용하면 간격이 네 방향 모두 같은 크기로 설정
paddingLeft, paddingRight, padding Top, paddingBottom와 layout_marginLeft, layout_margin Right, layout_marginTop, layout_marginBottom 속성을 이용 가능







Android 뷰의 속성들

visibility 속성은 뷰가 화면에 출력되어야 하는지를 설정
visible, invisible, gone으로 설정
invisible은 뷰가 화면에 보이지 않지만 자리는 차지
gone으로 설정하면 자리조차 차지하지 않음

Android 뷰의 속성들

Gravity 속성

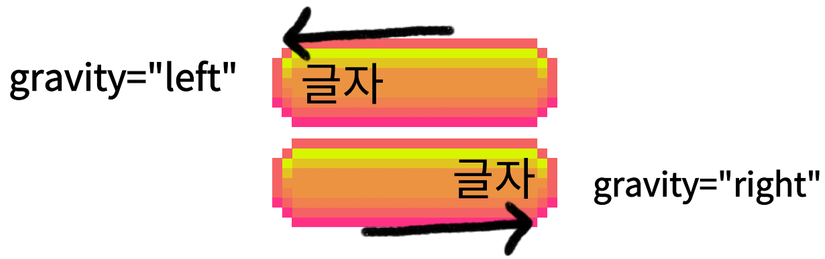
layout_gravity: 부모 컨테이너의 여유 공간에 뷰가 모두 채워지지 않아 여유 공간 안에서 뷰를 정렬할 때

gravity : [내부] 뷰에서 화면에 표시하는 내용물을 정렬할 때

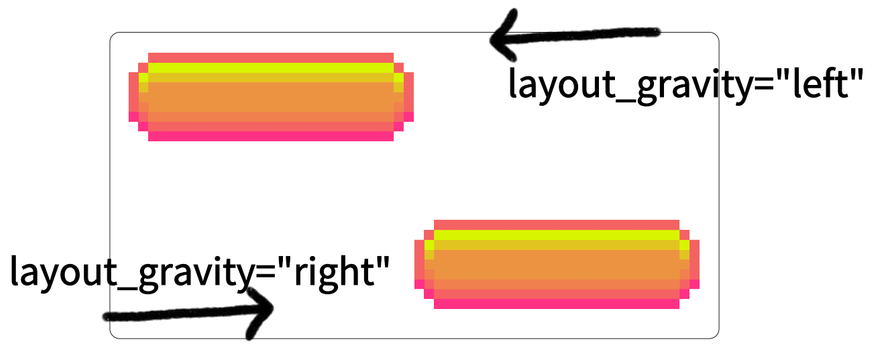
gravity속성은 or (|) 로 연결해서 사용가능 (top|right)

정렬 속성값	설 명
top	대상 객체를 위쪽 끝에 배치하기
bottom	대상 객체를 아래쪽 끝에 배치하기
left	대상 객체를 왼쪽 끝에 배치하기
right	대상 객체를 오른쪽 끝에 배치하기
center_vertical	대상 객체를 수직 방향의 중앙에 배치하기
center_horizontal	대상 객체를 수평 방향의 중앙에 배치하기
fill_vertical	대상 객체를 수직 방향으로 여유 공간만큼 확대하여 채우기
fill_horizontal	대상 객체를 수평 방향으로 여유 공간만큼 확대하여 채우기
center	대상 객체를 수직 방향과 수평 방향의 중앙에 배치하기
fill	대상 객체를 수직 방향과 수평 방향으로 여유 공간만큼 확대하여 채우기

버튼에 gravity 속성을 준경우 내부에 있는 요소(텍스트)가 정렬



버튼에 layout_gravity 속성을 준경우 버튼 자체가 정렬 (부모 레이아웃 내에서)



혼자 만들어보기

lineartest파일 추가 후 작업

정리

Linear Layout

Linear Layout은 선형 레이아웃으로 내부 요소가 orientation이라는 속성에 따라 가로 또는 세로로 계속 쌓이는 형태임.

뷰의 겹침이 불가능하며 , weight속성과 0dp,wrap_content를 이용하여 정확한 비율로 분할 시 많이 사용 된다.

Linear Layout을 좀 더 연습하거나 마스터 하고 싶다면 몬드리안의 작품을 만들어 보는것을 추천.ㅎㅎ