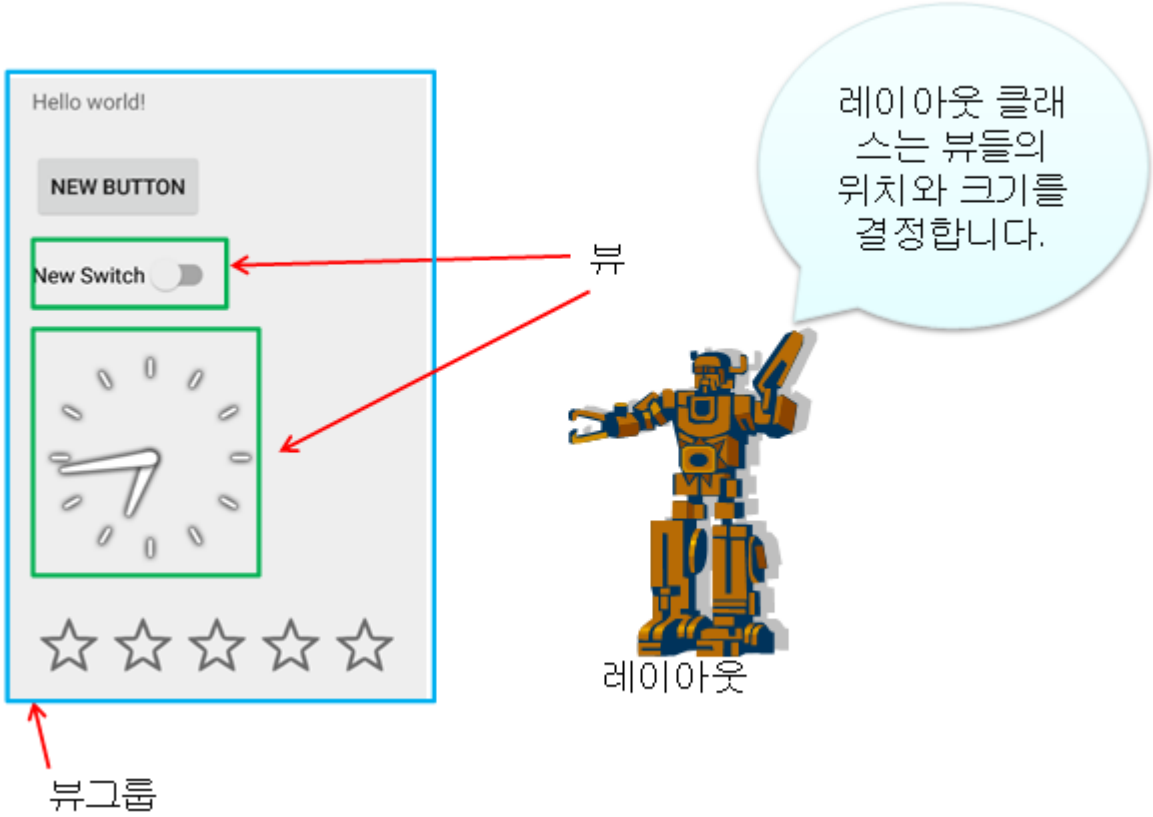
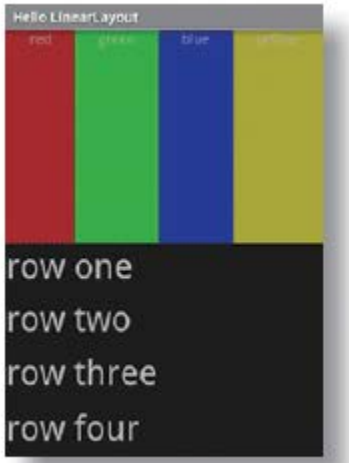
# CHAP 5. 레이아웃

# 레이아웃

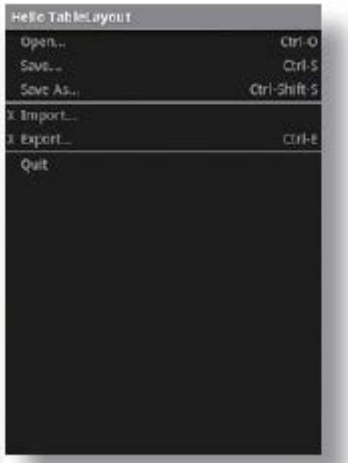
- 뷰들을 화면에 배치하는 방법



# 레이아웃의 종류



LinearLayout



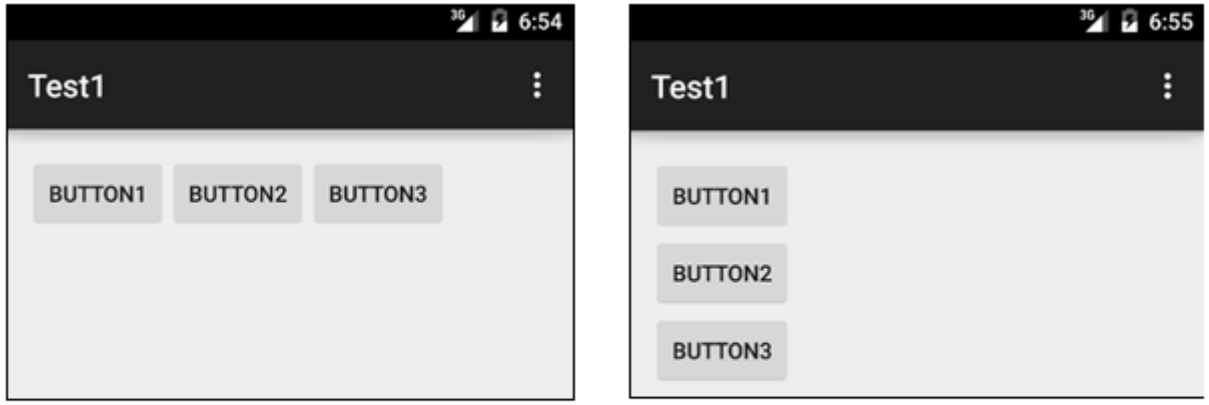
TableLayout

GridLayout

RelativeLayout

TabLayout

# 선형 레이아웃




(a) 수평 배치 (b) 수직 배치

그림 5.2 선형 레이아웃

# 선형 레이아웃 클래스의 속성

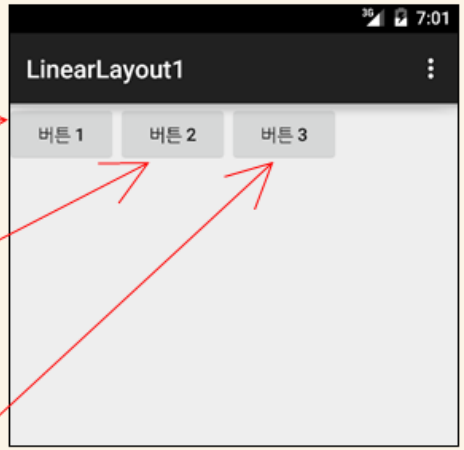
| 속성 | 관련 메소드 | 설명 |
|---|---|---|
| orientation | setOrientation(int) | “horizontal”은 수평으로, “vertical”은 수직으로 배치한다. |
| gravity | setGravity(int) | x축과 y축 상에 자식을 어떻게 배치할 것인지를 지정한다. |
| baselineAligned | setBaselineAligned (boolean) | false로 설정되면 자식뷰들의 기준선을 정렬하지 않는다. |

# 선형 레이아웃

*main.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
        <Button
                android:id="@+id/button01"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="버튼 1" />
        <Button
                android:id="@+id/button02"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="버튼 2" />
        <Button
                android:id="@+id/button03"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="버튼 3" />
</LinearLayout>
```

자식을 수평으로 배치



# GRAVITY 속성 값

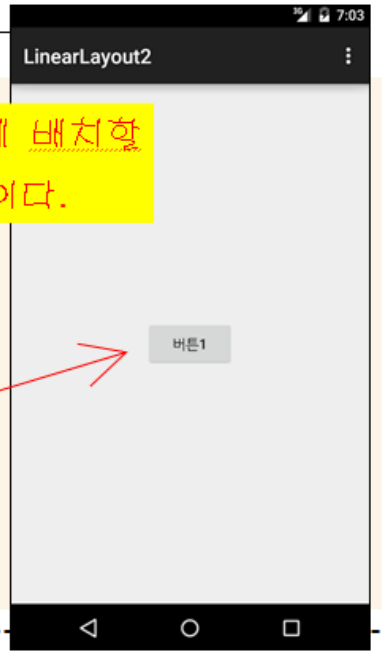
| 상수 | 값 | 설명 |
|---|---|---|
| top | 0x30 | 객체를 컨테이너의 상단에 배치, 크기를 변경하지 않음 |
| bottom | 0x50 | 객체를 컨테이너의 하단에 배치, 크기를 변경하지 않음 |
| left | 0x03 | 객체를 컨테이너의 좌측에 배치, 크기를 변경하지 않음 |
| right | 0x05 | 객체를 컨테이너의 우측에 배치, 크기를 변경하지 않음 |
| center_vertical | 0x10 | 객체를 컨테이너의 수직의 중앙에 배치, 크기를 변경하지 않음 |
| fill_vertical | 0x70 | 객체를 컨테이너의 수직을 채우도록 배치 |
| center_horizontal | 0x01 | 객체를 컨테이너의 수평의 중앙에 배치, 크기를 변경하지 않음 |
| fill_horizontal | 0x07 | 객체를 컨테이너의 수평을 채우도록 배치 |
| center | 0x11 | 객체를 컨테이너의 수평, 수직의 중앙에 배치 |
| fill | 0x77 | 객체가 컨테이너를 가득 채우도록 배치 |

# GRAVITY 속성

*main.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    >
<Button  android:id="@+id/button01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="버튼1"
    />
</LinearLayout>
```

"자식뷰를 중앙에 배치할 것!"이라는 의미이다.

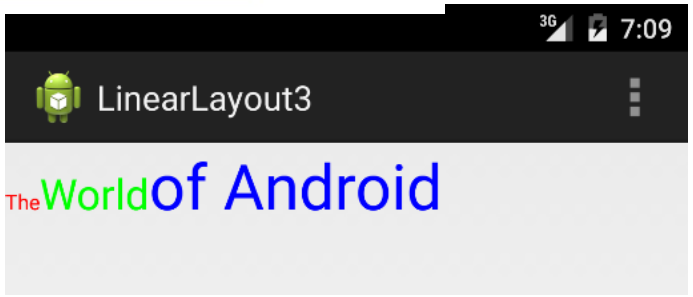


# 자식 뷰들의 베이스 라인 정렬

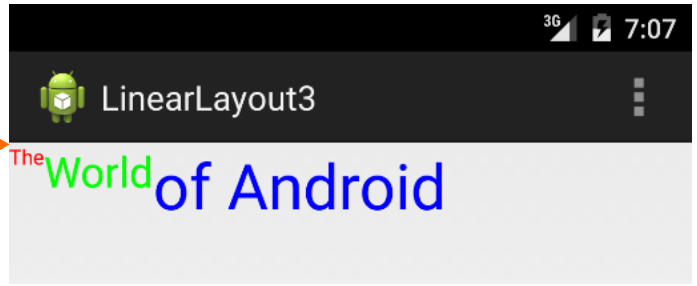
*main.xml*

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

자식들의 하단을 정렬한다.

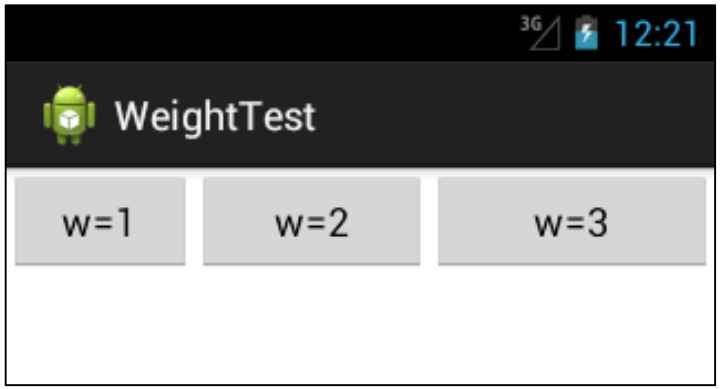



```
<LinearLayout
...
...
android:baselineAligned="false"
>
```



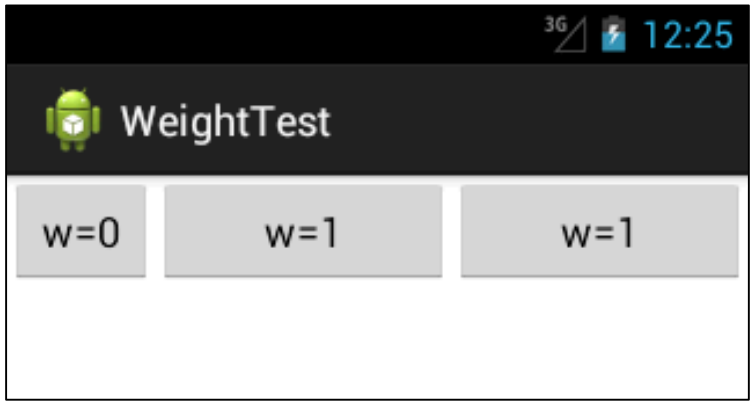
# 가중치(WEIGHT)

- 선형 레이아웃의 자식 뷰들의 가중치가 각각 1, 2, 3이면, 남아있는 공간의 1/6, 2/6, 3/6을 각각 할당받는다.



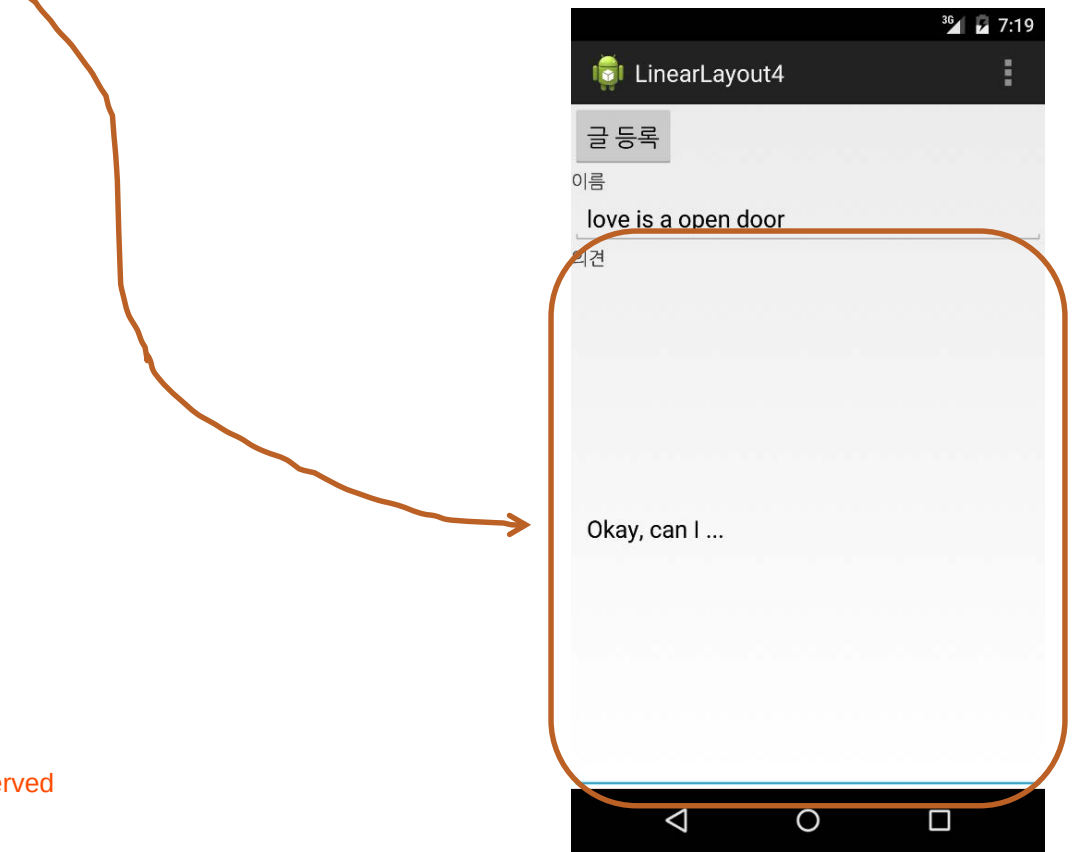
# 가중치(WEIGHT)

- 가중치를 1로 선언한 2개의 텍스트 뷰들은 남아있는 공간을 동일하게 차지할 것이다.



# 가중치 예제

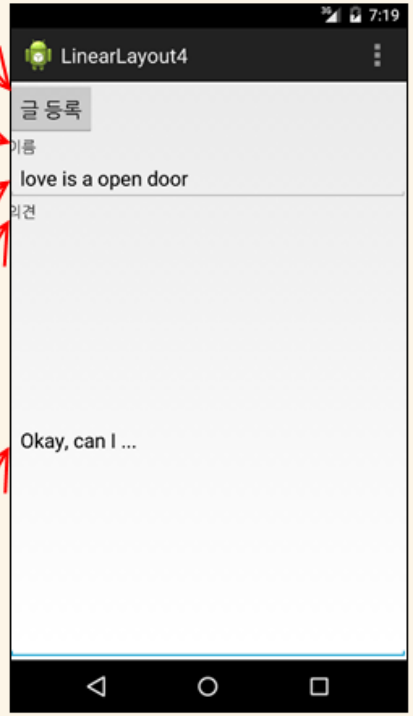
- 버튼, 텍스트 뷰, 에디트 텍스트 등의 뷰들을 가중치를 다르게 하여 배치한 예
- 에디트 텍스트만 가중치가 1이고 나머지는 전부 0



# 실습

```
<Button
    android:id="@+id/button01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="0"
    android:text="글 등록" />

<TextView
    android:id="@+id/textview01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="0"
    android:text="이름" />

<EditText
    android:id="@+id/edittext01"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="0" />

<TextView
    android:id="@+id/textview02"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="0"
    android:text="의견" />

<EditTextlove
    android:id="@+id/edittext02"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1" />
</LinearLayout>
```

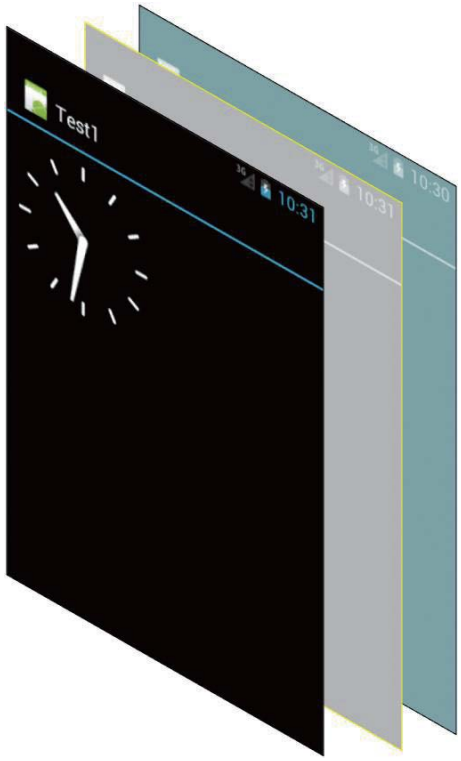
가중치를 0으로 설정




가중치를 1으로 설정

# 프레임 레이아웃

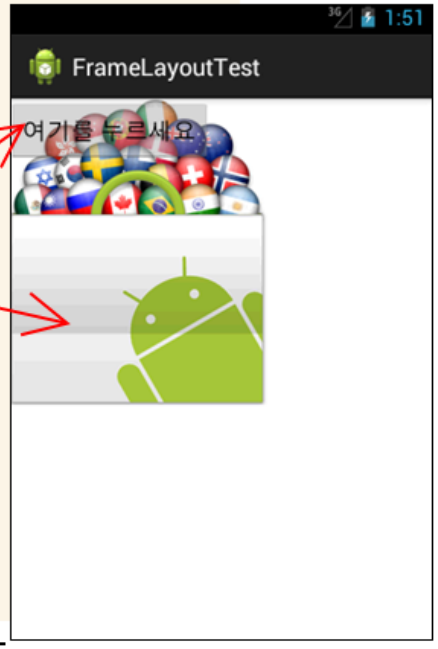
- 여러 자식 뷰들을 겹쳐서 배치
- 필요한 뷰의 가시성(visibility)을 true로 설정한다.



# 프레임 레이아웃 예제

*main.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
    <ImageView
        android:id="@+id/ImageView01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/marketbag"
    />
    <Button
        android:id="@+id/Button01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="여기를 누르세요"
    />
</FrameLayout>
```



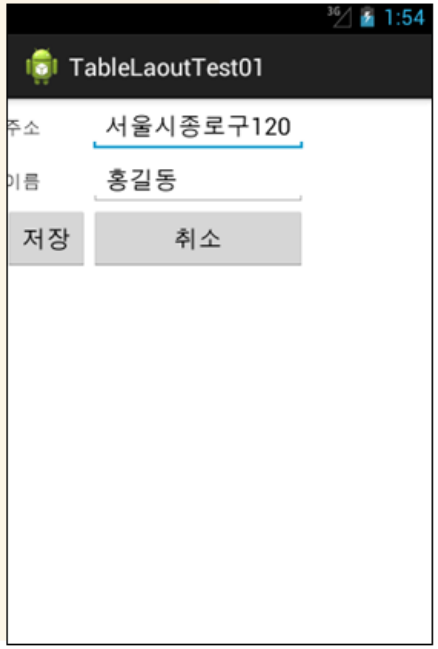

- android:visibility="visible"
- android:visibility="invisible"
- 코드로는 setVisibility(View.VISIBLE)로 설정을 변경할 수 있음

# 테이블 레이아웃

main.xml

테이블의 하나의 행

```
<TableLayout
xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout width="match parent"
        android:layout height="match parent">

    <TableRow>
        <TextView  android:text="주소"/>
        <EditText   android:text="서울시종로구 120"/>
    </TableRow>
    <TableRow>
        <TextView android:text="이름"/>
        <EditText android:text="홍길동"/>
    </TableRow>
    <TableRow>
        <Button android:text="저장"/>
        <Button android:text="취소"/>
    </TableRow>
</TableLayout>
```



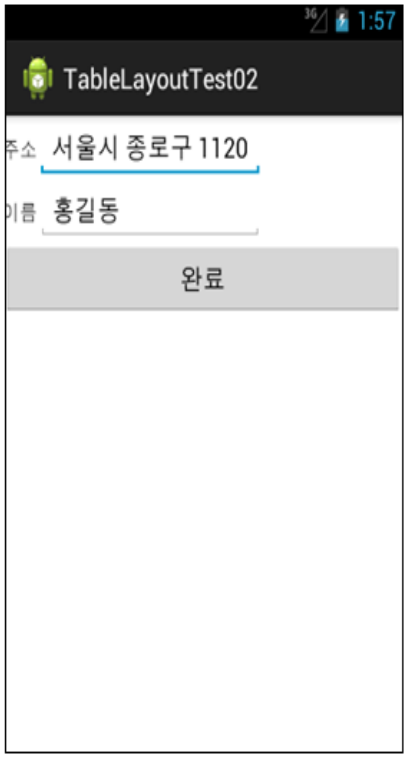
# 일반적인 뷰도 하나의 행이 될 수 있음

사용자 인터페이스 작성

*main.xml*

```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
    <TableRow>
        <TextView android:text="주소" />
        <EditText android:text="서울시 종로구 1120" />
    </TableRow>

    <TableRow>
        <TextView android:text="이름" />
        <EditText android:text="홍길동" />
    </TableRow>

    <Button android:text="완료" />

</TableLayout>
```

테이블의 하나의 행



# 상대적 레이아웃

| 속성 | 설명 |
|---|---|
| layout_above | 만약 true이면 현재 뷰의 하단을 기준 뷰의 위에 일치시킨다. |
| layout_below | 현재 뷰의 상단을 기준 뷰의 하단에 위치시킨다. |
| layout_centerHorizontal | 수평으로 현재 뷰의 중심을 부모와 일치시킨다. |
| layout_centerInParent | 부모의 중심점에 현재 뷰를 위치시킨다. |
| layout_centerVertical | 수직으로 현재 뷰의 중심을 부모와 일치시킨다. |
| layout_toLeftOf | 현재 뷰의 우측단을 기준 뷰의 좌측단에 위치시킨다. |
| layout_toRightOf | 현재 뷰의 좌측단을 기준 뷰의 우측단에 위치시킨다. |

# 상대적 레이아웃

*main.xml*

```
<?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">
    <TextView
            android:id="@+id/address"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_alignParentTop="true"
            android:text="주소를 입력하세요" />
    <EditText
            android:id="@+id/input"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@android:drawable/editbox_background"
            android:layout_below="@id/address" />
    <Button
            android:id="@+id/cancel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@id/input"
            android:layout_alignParentRight="true"
            android:layout_marginLeft="10dip"
            android:text="취소" />
    <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toLeftOf="@id/cancel"
            android:layout_alignTop="@id/cancel"
            android:text="확인" />
</RelativeLayout>
```

address 아래에 배치

input 아래에 배치

cancel의 왼쪽에 배치

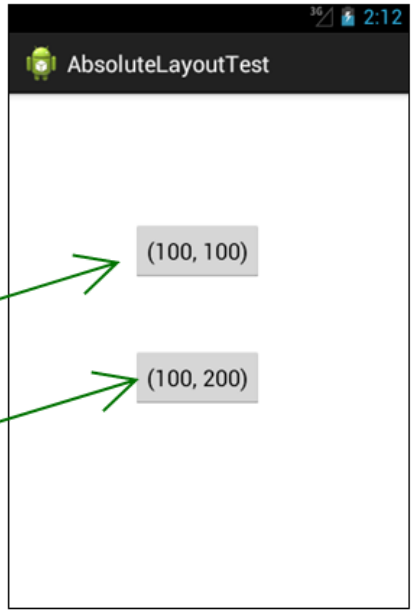
# 절대적 레이아웃

사용자 인터페이스 작성

*main.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="100px"
    android:layout_y="100px"
    android:text="(100, 100)"
    />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="100px"
    android:layout_y="200px"
    android:text="(100, 200)"
    />
</AbsoluteLayout>
```

(100, 100)에 배치



# 코드로 속성 변경하기

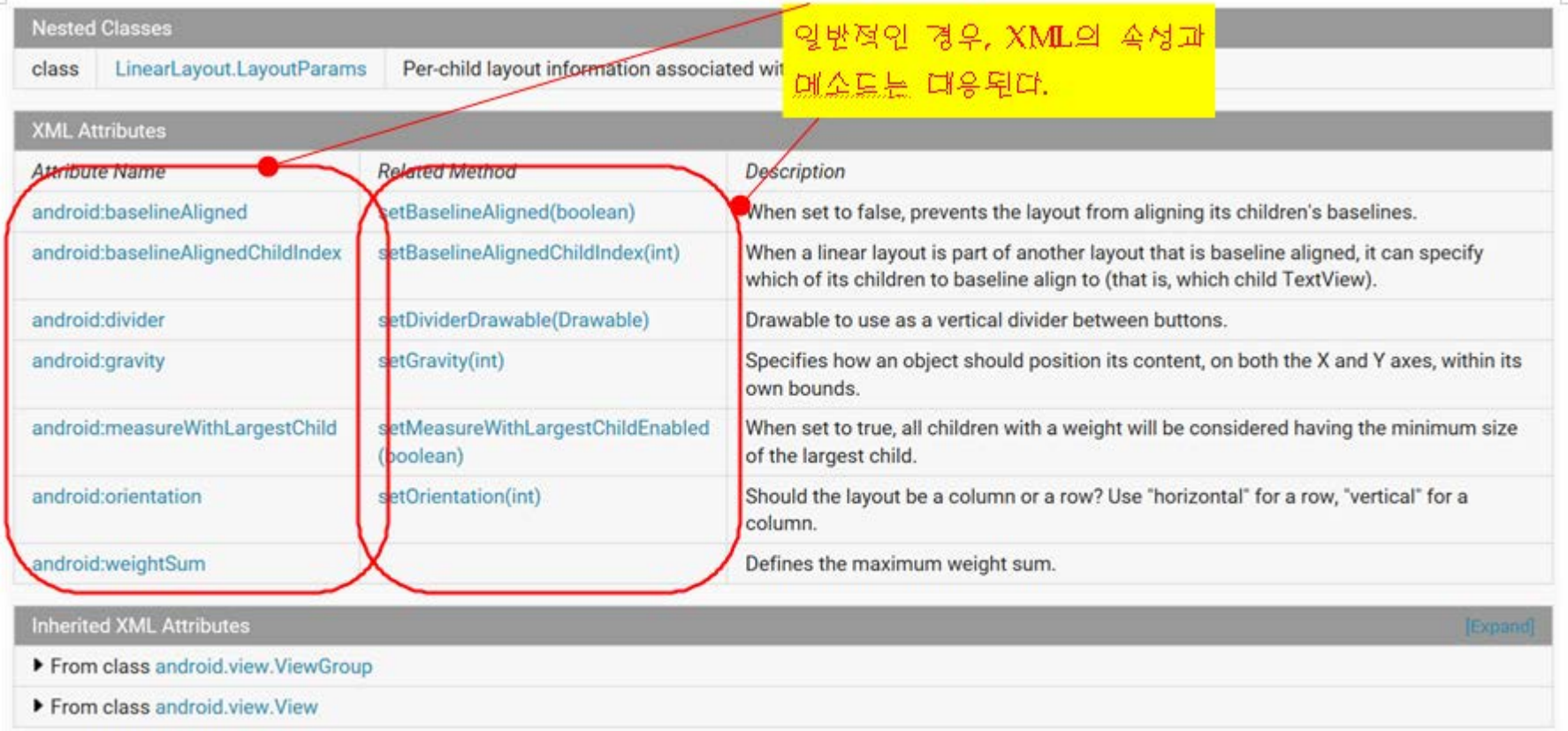



그림 5. 선형 레이아웃 클래스

# 코드로 속성 변경

*main.xml*

id를 부여한다.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/LayoutManager"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >
        <Button
              android:text="버튼"
              android:id="@+id/Button01"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content">
        </Button>
        <Button
              android:text="버튼"
              android:id="@+id/Button02"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content">
        </Button>
        <Button
              android:text="버튼"
              android:id="@+id/Button03"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content">
        </Button>
</LinearLayout>
```

# 코드로 속성 변경

*LayoutByCodeActivity.java*

```
package kr.co.company.layoutbycode;
// 소스만 입력하고 Alt+Enter 키를 눌러서 import 문장을 자동으로 생성한다.

public class LayoutByCodeActivity extends ActionBarActivity {

        @Override
        public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);

            LinearLayout manager =
                    (LinearLayout)findViewById(R.id.LayoutManager);
            manager.setOrientation(LinearLayout.HORIZONTAL);

            Button button = (Button)findViewById(R.id.Button01);
            button.setText("첫번째 버튼");
        }

}
```

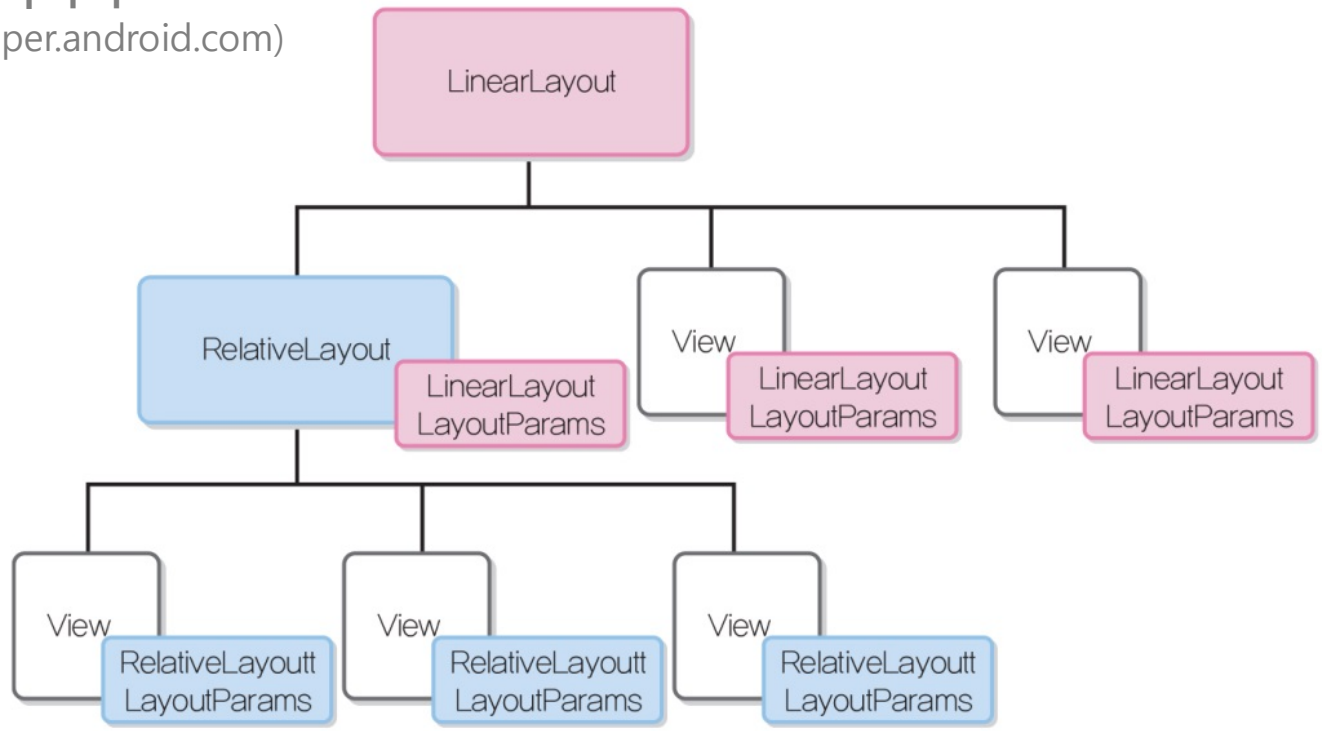
배치 방향을 수평으로 변경한다.

버튼의 텍스트를 변경한다.

# 레이아웃 객체를 코드로 생성하기

- 하나의 화면을 이루는 뷰들의 계층 구조

**레이아웃 파라미터**
(출처 : developer.android.com)



# 코드로 레이아웃 만들기


코드 작성

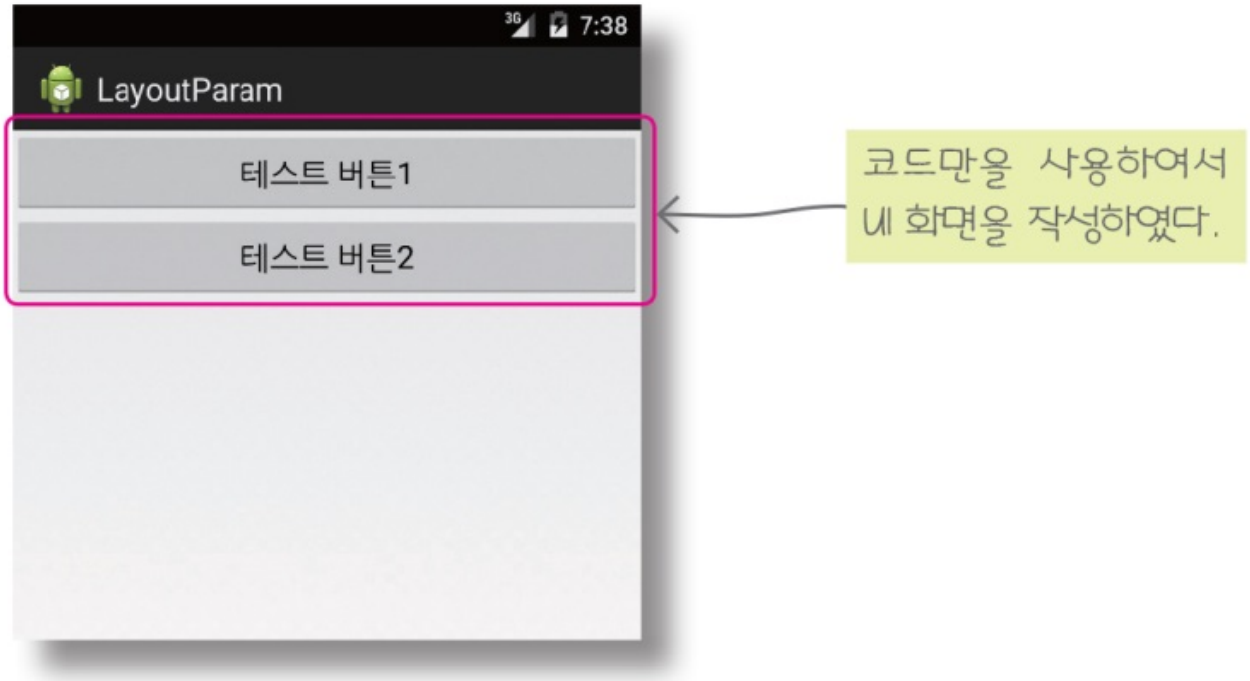
*LayoutParamActivity.java*

```
package kr.co.company.layoutparam;
// 소스만 입력하고 Alt+Enter를 눌러서 import 문장을 자동으로 생성한다.

public class LayoutParamActivity extends ActionBarActivity {

        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);

                LinearLayout manager = new LinearLayout(this);
                manager.setOrientation(LinearLayout.VERTICAL);

                Button button1 = new Button(this);
                button1.setText("테스트 버튼1");

                Button button2 = new Button(this);
                button2.setText("테스트 버튼2");

                LinearLayout.LayoutParams param = new
                        LinearLayout.LayoutParams(
                                LinearLayout.LayoutParams.MATCH_PARENT,
                                LinearLayout.LayoutParams.WRAP_CONTENT);
                manager.addView(button1, param);
                manager.addView(button2, param);
                setContentView(manager);
        }

}
```

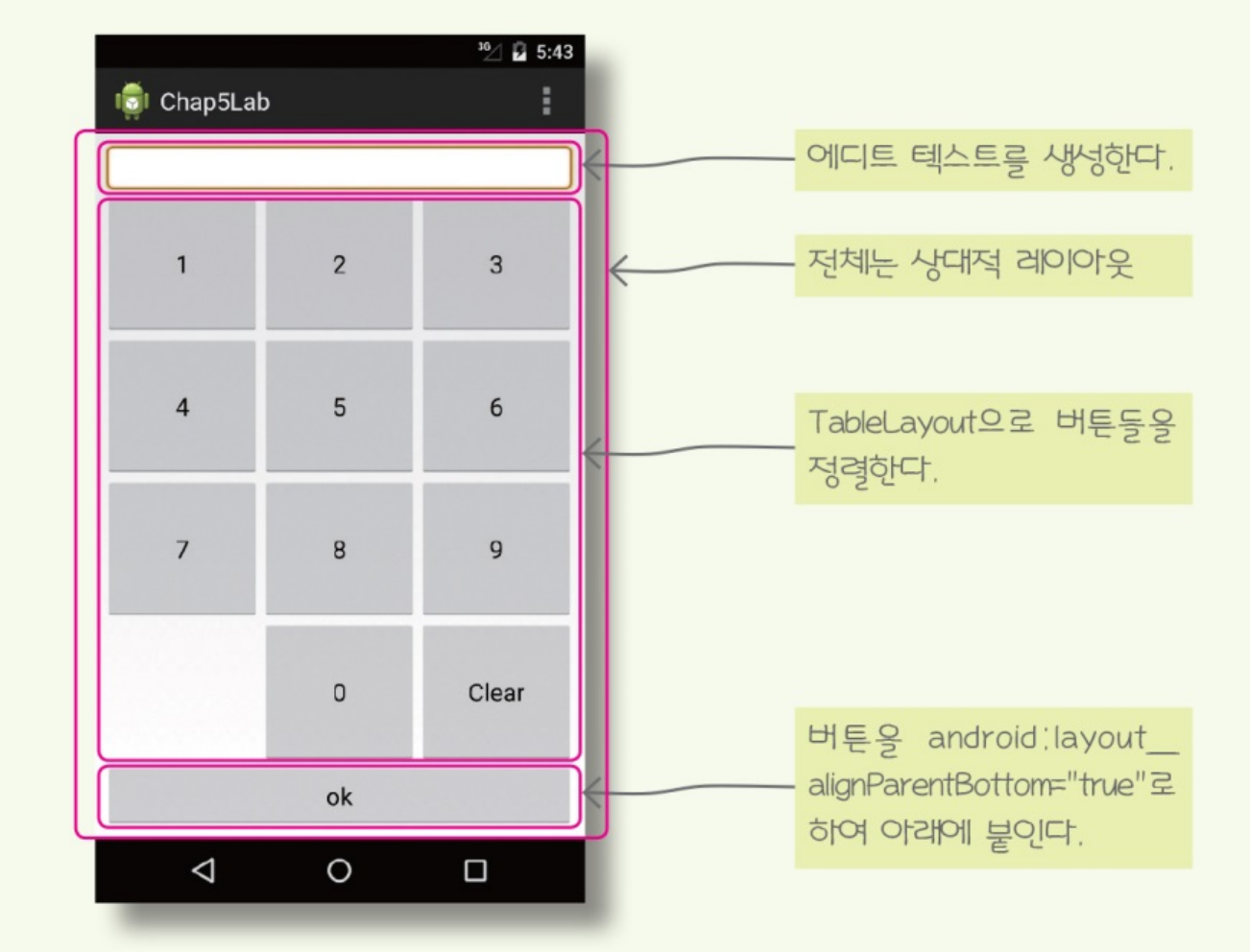
LayoutParams 객체를 생성한다.

자식뷰를 추가할 때마다 LayoutParams 객체를 전달한다.

# 실행 결과



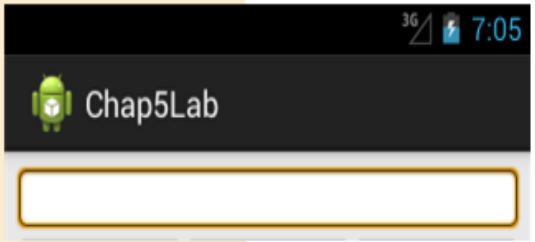
# LAB: 계산기 앱 작성



# 상단 구현

*main.xml*

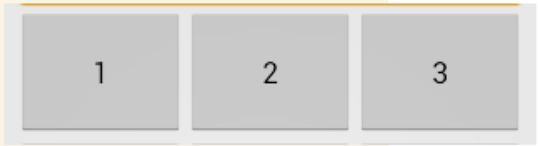
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:padding="7dip">
  <EditText android:id="@+id/number"
        android:background="@android:drawable/editbox_background"
        android:cursorVisible="false"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"  />
        ...
</RelativeLayout>
```



# 버튼 구현

*main.xml*

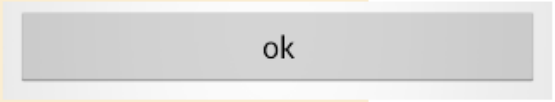
```
<TableLayout  android:id="@+id/row1"
              android:layout_width="fill_parent"
              android:layout_height="wrap_content"
          android:layout_below="@id/numberSeparator2"
          android:layout_above="@id/ok"
          android:layout_weight="1">
    <TableRow   android:layout_weight="1">
      <Button  android:id="@+id/n1"
           android:layout_width="0dip"
           android:layout_height="fill_parent"
           android:text="1"
           android:layout_weight="1" />
      <Button  android:id="@+id/n2"
           android:layout_width="0dip"
           android:layout_height="fill_parent"
           android:text="2"
           android:layout_weight="1" />
      <Button  android:id="@+id/n3"
           android:layout_width="0dip"
           android:layout_height="fill_parent"
           android:text="3"
           android:layout_weight="1" />
    </TableRow>
```



# 하단 구현

*main.xml*

```
<Button  android:id="@+id/ok"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="ok"
        android:layout_alignParentBottom="true" />
```



# 실습