



## 4장. 레이아웃을 활용한 다양한 뷰 배치

# 4.1 LinearLayout

## 4.1.1. LinearLayout 소개

- LinearLayout은 뷰를 순서대로 가로나 세로 방향으로 나열, 방향을 지정하는 orientation 속성을 제공

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

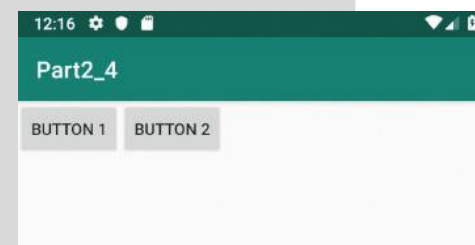
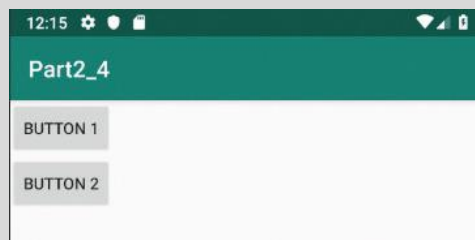
```
<Button
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Button 1"/>
```

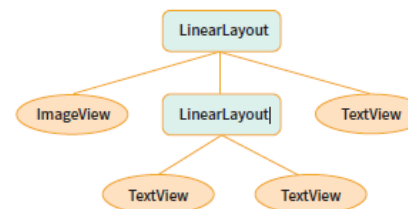
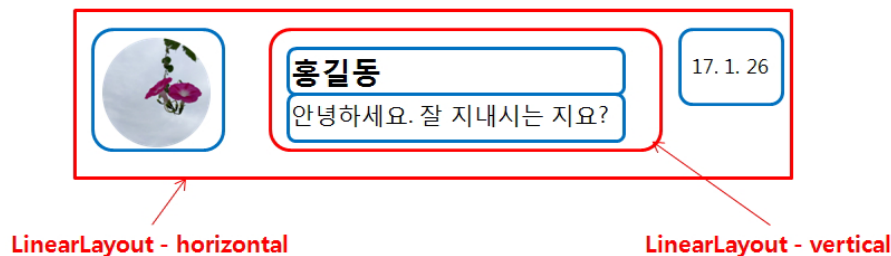
```
<Button
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Button 2"/>
```

```
</LinearLayout>
```



## 4.1.2. 레이아웃 중첩



# 4.1 LinearLayout

## 4.1.3. LinearLayout 속성

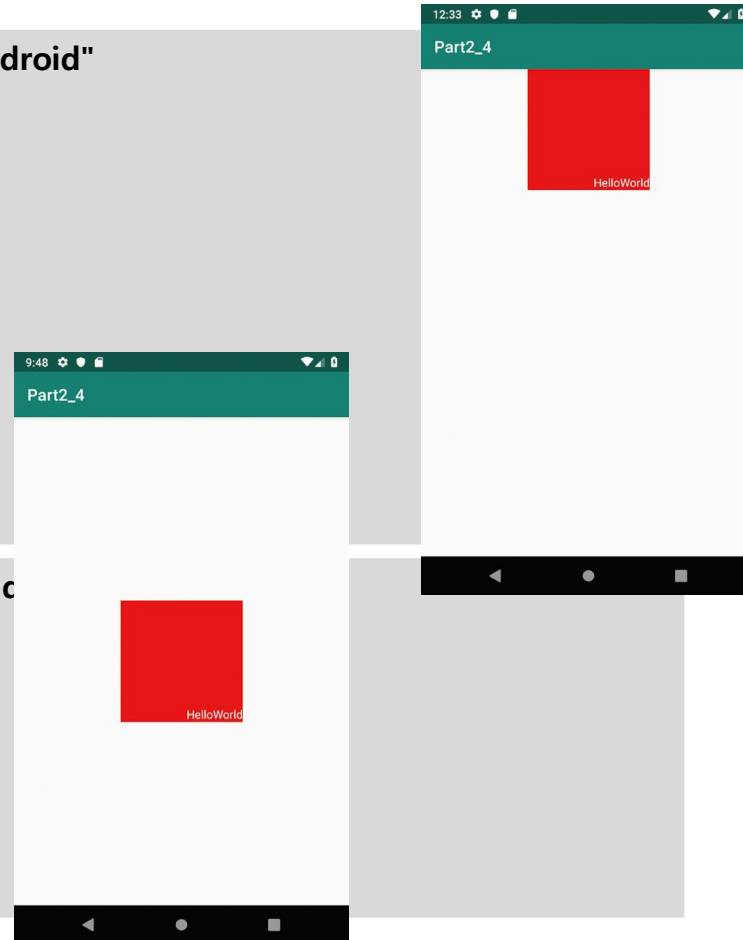
- gravity, layout\_gravity : gravity 속성은 뷰의 내용을 뷰 영역 내에서 어디에, layout\_gravity 속성은 뷰를 LinearLayout 영역 내에서 어디에

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <TextView
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:text="HelloWorld"
        android:background="#FF0000"
        android:textColor="#FFFFFF"
        android:layout_gravity="center_vertical|center_horizontal"
        android:gravity="bottom|right"/>
</LinearLayout>
```

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

    .....

</LinearLayout>
```



## 4.1 LinearLayout

weight : 여백확장, 값은 절대적 수치가 아닌 상대적으로 계산되는 값

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:background="#FF0000"/>
    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="2"
        android:background="#00FF00"/>
    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:background="#0000FF"/>
</LinearLayout>
```



# Step by Step 4-1 – LinearLayout

- 모듈 생성
- 이미지 리소스 파일 복사
- activity\_main.xml 파일 작성
- 실행





## 4.2 RelativeLayout

### 4.2.1. RelativeLayout 소개

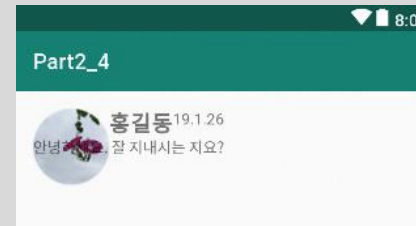
- 화면에 배치된 뷰를 기준으로 다른 뷰의 위치를 지정
- android:layout\_above: 기준 뷰의 윗부분에 배치
- android:layout\_below: 기준 뷰의 아랫부분에 배치
- android:layout\_toLeftOf: 기준 뷰의 왼쪽에 배치
- android:layout\_toRightOf: 기준 뷰의 오른쪽에 배치

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android">
    <ImageView
        android:id="@+id/icon"/>

    <TextView
        android:id="@+id/name"
        android:layout_toRightOf="@id/icon"/>

    <TextView
        android:id="@+id/content"
        android:layout_below="@id/name"/>

    <TextView
        android:id="@+id/date"
        android:layout_toRightOf="@id/name"/>
</RelativeLayout>
```



## 4.2 RelativeLayout

### 4.2.2. align 속성

- 기준이 되는 뷰와 왼쪽 변을 맞추거나 윗변을 맞추는 등의 작업
- android:layout\_alignTop: 기준 뷰와 윗부분을 정렬
- android:layout\_alignBottom: 기준 뷰와 아랫부분을 정렬
- android:layout\_alignLeft: 기준 뷰와 왼쪽을 정렬
- android:layout\_alignRight: 기준 뷰와 오른쪽을 정렬
- android:layout\_alignBaseline: 기준 뷰와 텍스트 기준선을 정렬

```
<TextView  
    android:id="@+id/content"  
    android:layout_below="@id/name"  
    android:layout_alignLeft="@id/name"/>
```

```
<TextView  
    android:id="@+id/date"  
    android:layout_toRightOf="@id/name"  
    android:layout_alignBaseline="@id/name"/>
```



## 4.2 RelativeLayout

### 4.2.3. alignParentXXX 속성

- RelativeLayout 영역의 상하좌우로 밀 수 있는 속성
- android:layout\_alignParentTop: 부모의 윗부분에 뷰의 상단을 위치
- android:layout\_alignParentBottom: 부모의 아랫부분에 뷰의 하단을 위치
- android:layout\_alignParentLeft: 부모의 왼쪽에 뷰의 왼쪽을 위치
- android:layout\_alignParentRight: 부모의 오른쪽에 뷰의 오른쪽을 위치
- android:layout\_centerHorizontal: 부모의 가로 방향 중앙에 뷰를 위치
- android:layout\_centerVertical: 부모의 세로 방향 중앙에 뷰를 위치
- android:layout\_centerInParent: 부모의 가로세로 중앙에 뷰를 위치

```
<TextView  
    android:id="@+id/date"  
    android:layout_alignBaseline="@id/name"  
    android:layout_alignParentRight="true"/>
```





# Step by Step 4-2 - RelativeLayout

- Activity 생성
- activity\_lab4\_2.xml 작성
- Lab4\_2Activity.java 실행



## 4.3 FrameLayout

### 4.3.1. FrameLayout 소개

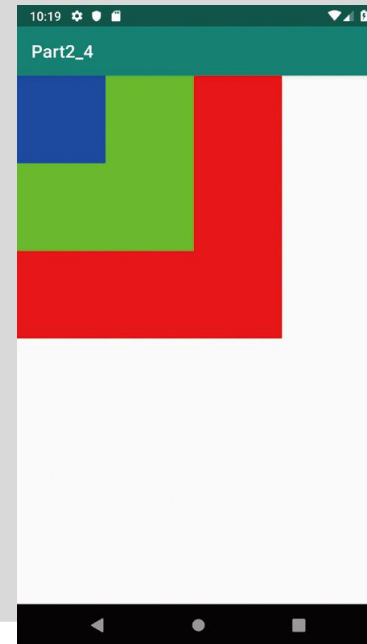
- 뷰들을 같은 영역에 겹쳐서 배치
- 뷰가 포함된 순서대로 배치되어 맨 마지막에 포함한 뷰가 가장 위에 위치
- visibility 속성으로 특정 순간에 뷰가 보이거나 안 보이게 제어

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

    <TextView
        android:layout_width="300dp"
        android:layout_height="300dp"
        android:background="#FF0000"/>

    <TextView
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:background="#00FF00"/>

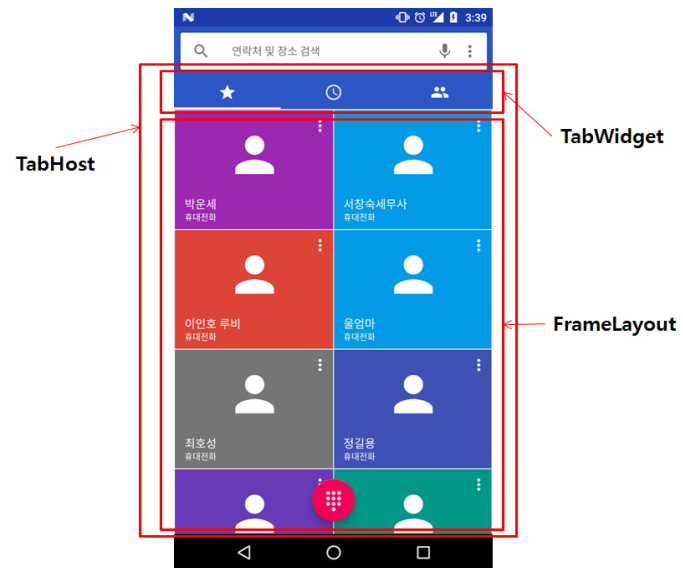
    <TextView
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:background="#0000FF"/>
</FrameLayout>
```



## 4.3 FrameLayout

### 4.3.2. 탭 화면 구현 : TabHost

- TabHost: 탭 전체 영역을 지칭
- TabWidget: 탭 버튼이 들어갈 영역을 지칭
- FrameLayout: 탭 버튼 클릭 시 나올 화면 영역을 지칭



## 4.3 FrameLayout

```
<TabHost xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/host"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <TabWidget
            android:id="@android:id/tabs"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/colorPrimary"
            />
        <FrameLayout
            android:id="@android:id/tabcontent"
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <TextView
                android:id="@+id/tab_content1"/>
            <TextView
                android:id="@+id/tab_content2"/>
            <TextView
                android:id="@+id/tab_content3"/>
        </FrameLayout>
    </LinearLayout>
</TabHost>
```

## 4.3 FrameLayout

- TabWidget의 id 값: android:id="@android:id/tabs"
- FrameLayout의 id 값: android:id="@android:id/tabcontent"
- 탭 버튼과 버튼에 따라 나타낼 화면을 TabSpec 클래스로 결합하여 추가
- TabSpec에는 버튼인 Indicator와 본문인 TabContent가 결합

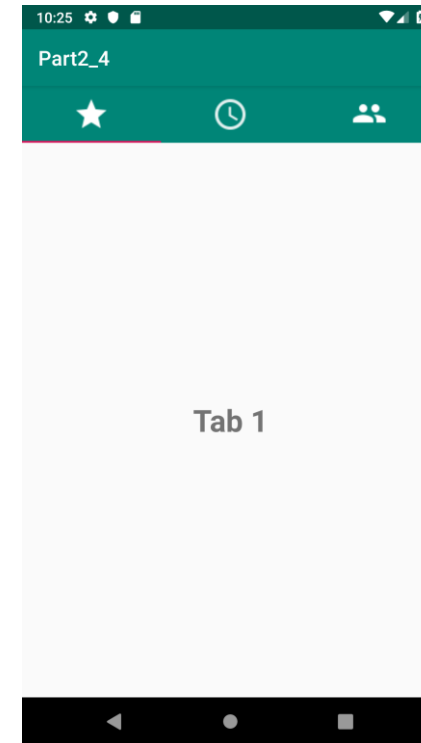
```
TabHost host=findViewById(R.id.host);
host.setup();

//각 TabSpec을 매번 TabSpec의
TabHost.TabSpec spec=host.newTabSpec("tab1");
//tab button 구성
spec.setIndicator(null, ResourcesCompat.getDrawable(getResources(),
    R.drawable.tab_icon1, null));
//tab 본 내용
spec.setContent(R.id.tab_content1);
host.addTab(spec);
```



# Step by Sep 4-3 – 탭 화면

- 액티비티 추가
- 이미지 리소스 복사
- activity\_lab4\_3.xml 작성
- Lab4\_3Activity.java 작성
- Lab4\_3Activity.java 실행

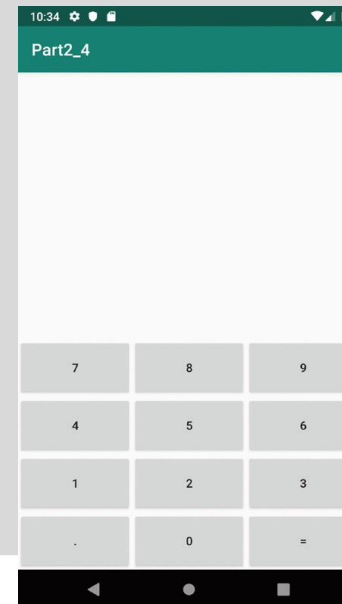
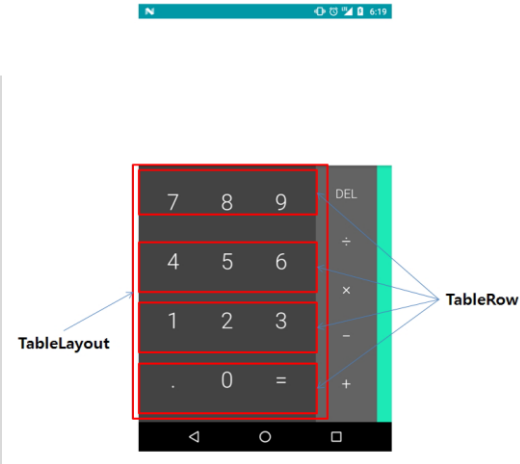




## 4.4 TableLayout

뷰를 테이블(table) 구조로 나열

```
<TableLayout>
  <TableRow>
    <Button android:text="7" />
    <Button android:text="8" />
    <Button android:text="9" />
  </TableRow>
  <TableRow>
    <Button android:text="4" />
    <Button android:text="5" />
    <Button android:text="6" />
  </TableRow>
  <TableRow>
    <Button android:text="1" />
    <Button android:text="2" />
    <Button android:text="3" />
  </TableRow>
  <TableRow>
    <Button android:text="." />
    <Button android:text="0" />
    <Button android:text="=" />
  </TableRow>
</TableLayout>
```



## 4.4 TableLayout

- `android:shrinkColumns="0,1"`: 화면 크기를 벗어나는 경우 인덱스 0, 1의 열 크기를 줄임
- `android:stretchColumns="1"`: 화면 여백이 발생하는 경우 인덱스 1의 열 크기를 늘림
- `android:layout_column="1"`: 뷰의 위치 지정, 인덱스 1의 위치에 뷰가 위치
- `android:layout_span="2"`: 두 개의 열을 하나의 뷰가 차지

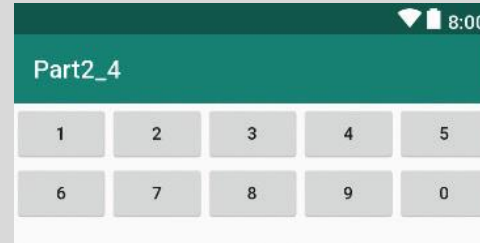
# 4.5 GridLayout

## 4.5.1. GridLayout 소개

- 뷰를 테이블 구조로 나열
- 테이블 구조로 나열된다는 점에서 TableLayout과 유사
- 뷰가 레이아웃에 포함된 순서 대로 가로나 세로 방향으로 나열된다는 점에서 LinearLayout과도 유사

```
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:columnCount="5">
    <Button android:text="1"/>
    <Button android:text="2"/>
    <Button android:text="3"/>
    <Button android:text="4"/>
    <Button android:text="5"/>
    <Button android:text="6"/>
    <Button android:text="7"/>
    <Button android:text="8"/>
    <Button android:text="9"/>
    <Button android:text="0"/>

</GridLayout>
```



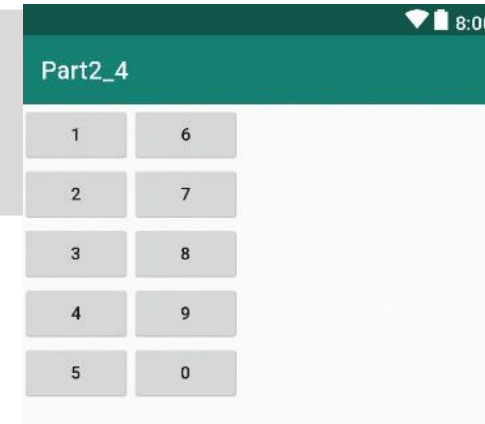
## 4.5 GridLayout

```
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:rowCount="5">
```

### 4.5.2. GridLayout 속성

- orientation: 뷰의 배치 방향을 지정. 기본값은 가로 방향
- columnCount: 가로 방향일 때 한 줄에 몇 개의 뷰를 나열할 것인지 지정
- rowCount : 세로 방향일 때 한 줄에 몇 개의 뷰를 나열할 것인지 지정
- layout\_column: 뷰가 위치할 열 인덱스 지정
- layout\_row: 뷰가 위치할 행 인덱스 지정
- layout\_columnSpan: 가로 방향으로 여러 열을 하나의 뷰가 차지하고자 할 때
- layout\_rowSpan : 세로 방향으로 여러 행을 하나의 뷰가 차지하고자 할 때
- layout\_gravity : 하나의 열 내에서 뷰의 정렬 위치 지정

```
<Button android:text="5"
    android:layout_row="1"
    android:layout_column="1"/>
<Button android:text="6"/>
<Button android:text="7"
    android:layout_row="2"
    android:layout_column="0"/>
```

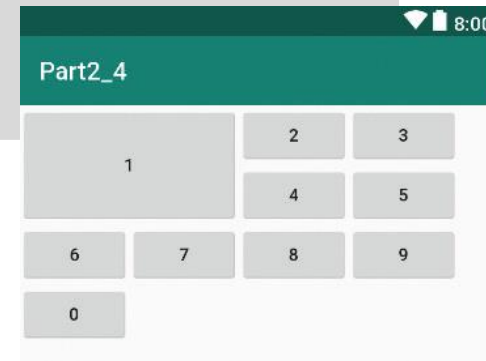


## 4.5 GridLayout

```
<Button android:text="1"  
    android:layout_rowSpan="2"  
    android:layout_columnSpan="2"/>
```

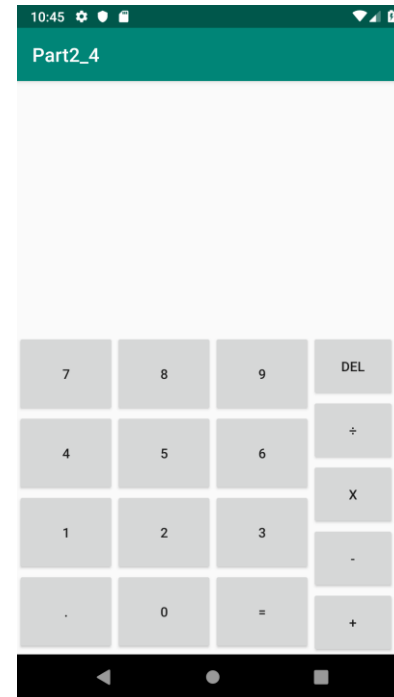


```
<Button android:text="1"  
    android:layout_rowSpan="2"  
    android:layout_columnSpan="2"  
    android:layout_gravity="fill"/>
```



# Step by Step 4-4 – TableLayout, GridLayout 활용

- 액티비티 추가
- activity\_lab4\_4.xml 작성
- Lab4\_4Activity.java 실행





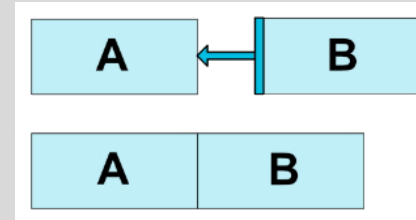
## 4.6 ConstraintLayout

### 4.6.1. 상대적 위치 지정

- 2016년 Google IO 행사에서 발표
- RelativeLayout과 마찬가지로 상대 위치에 따라 뷰의 배치
- Horizontal Axis: Left, Right, Start and End sides
- Vertical Axis: top, bottom sides and text baseline

```
<Button  
    android:id="@+id/btn1"  
    .../>
```

```
<Button  
    ...  
    app:layout_constraintLeft_toRightOf="@+id/btn1"/>
```

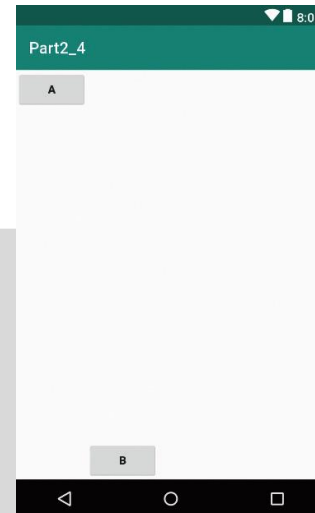


## 4.6 ConstraintLayout

- layout\_constraintLeft\_toLeftOf
- layout\_constraintLeft\_toRightOf
- layout\_constraintRight\_toLeftOf
- layout\_constraintRight\_toRightOf
- layout\_constraintTop\_toTopOf
- layout\_constraintTop\_toBottomOf
- layout\_constraintBottom\_toTopOf
- layout\_constraintBottom\_toBottomOf
- layout\_constraintBaseline\_toBaselineOf
- layout\_constraintStart\_toEndOf
- layout\_constraintStart\_toStartOf
- layout\_constraintEnd\_toStartOf
- layout\_constraintEnd\_toEndOf

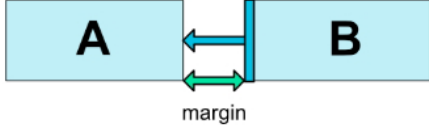
```
<Button
    android:id="@+id/btn1"
    ...../>

<Button
    .....
    app:layout_constraintLeft_toRightOf="@+id/btn1"
    app:layout_constraintBottom_toBottomOf="parent"/>
```




## 4.6 ConstraintLayout

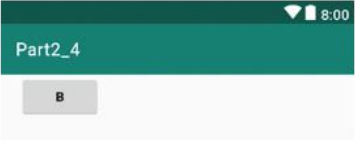
### 4.6.2. 여백(margin)

- 뷰와 뷰 사이의 간격을 표현하기 위해서 margin 설정
  - `android:layout_marginStart`
  - `android:layout_marginEnd`
  - `android:layout_marginLeft`
  - `android:layout_marginTop`
  - `android:layout_marginRight`
  - `android:layout_marginBottom`
- 
- 또한 상대 뷰가 `View.GONE` 상태일 때의 margin 값을 따로 설정가능 `layout_goneMarginStart`
  - `layout_goneMarginEnd`
  - `layout_goneMarginLeft`
  - `layout_goneMarginTop`
  - `layout_goneMarginRight`
  - `layout_goneMarginBottom`


## 4.6 ConstraintLayout



```
<Button
    android:id="@+id/btn1"
    android:text="A" />
<Button
    android:text="B"
    app:layout_constraintLeft_toRightOf="@+id/btn1"
    android:layout_marginLeft="20dp" />
```



```
<Button
    android:id="@+id/btn1"
    android:text="A"
    android:visibility="gone" />
<Button
    android:text="B"
    app:layout_constraintLeft_toRightOf="@+id/btn1"
    android:layout_marginLeft="20dp" />
```

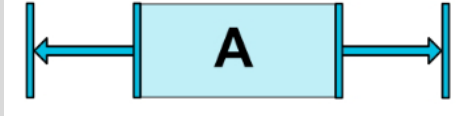


```
<Button
    android:id="@+id/btn1"
    android:text="A"
    android:visibility="gone" />
<Button
    android:text="B"
    app:layout_constraintLeft_toRightOf="@+id/btn1"
    android:layout_marginLeft="20dp"
    app:layout_goneMarginLeft="0dp" />
```

## 4.6 ConstraintLayout

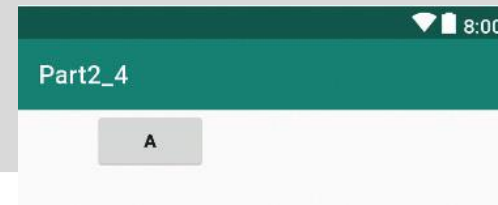
### 4.6.3. 가운데 맞춤과 치우침(bias)

```
<Button  
...  
app:layout_constraintLeft_toLeftOf="parent"  
app:layout_constraintRight_toRightOf="parent"/>
```



- layout\_constraintHorizontal\_bias: 가로 치우침 조절
- layout\_constraintVertical\_bias: 세로 치우침 조절

```
<Button  
...  
app:layout_constraintLeft_toLeftOf="parent"  
app:layout_constraintRight_toRightOf="parent"  
app:layout_constraintHorizontal_bias="0.2"/>
```



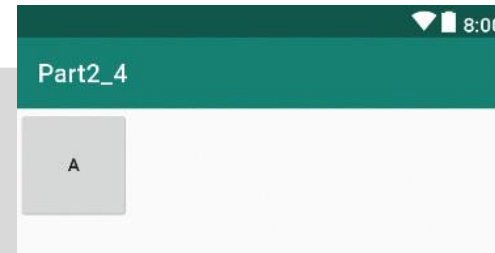
값을 0.2로 지정하면 20%의 의미로 왼쪽에서 20% 위치

## 4.6 ConstraintLayout

### 4.6.4. 비율(Ratio)

- 뷰의 크기를 지정할 때 가로세로 비율에 의한 크기를 지정
- 크기 값이 0dp로 지정되어야 가능

```
<Button  
    android:id="@+id/btn1"  
    android:layout_width="wrap_content"  
    android:layout_height="0dp"  
    app:layout_constraintDimensionRatio="1:1"  
    app:layout_constraintLeft_toLeftOf="parent"/>
```



```
<Button  
    android:id="@+id/btn2"  
    android:layout_width="0dp"  
    android:layout_height="0dp"  
    .....  
    app:layout_constraintLeft_toLeftOf="parent"  
    app:layout_constraintRight_toRightOf="parent"  
    app:layout_constraintDimensionRatio="H,2:1"/>
```

