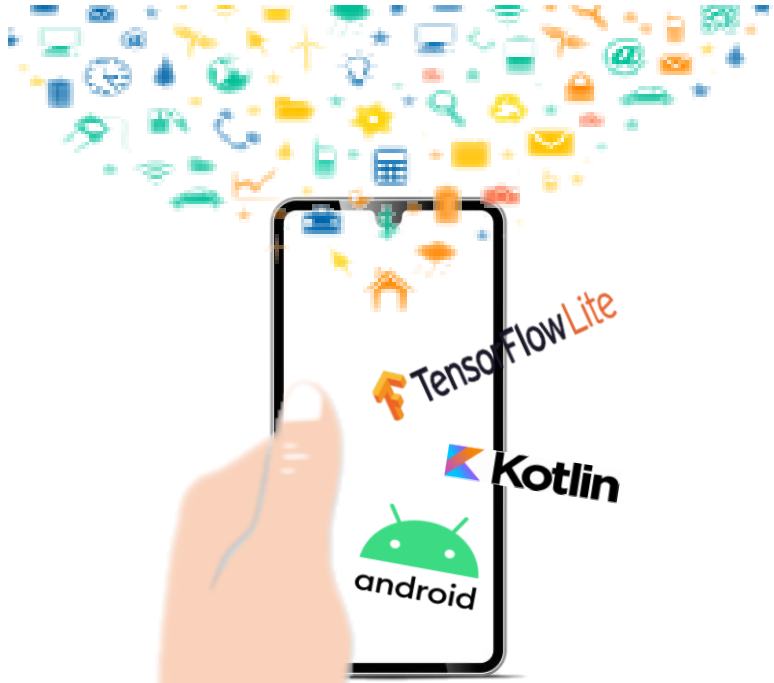
Android 애플리케이션 프로그래밍
- Layout

Layout 개요, Layout의 종류, Layout의 속성



부산대학교 정보·의생명공학대학
정보컴퓨터공학부

이론

강의 목표와 구성

❖ Layout

- Layout 개요
- Layout 종류와 속성
  - LinearLayout
  - RelativeLayout
  - FrameLayout
  - GridLayout
  - ConstraintLayout

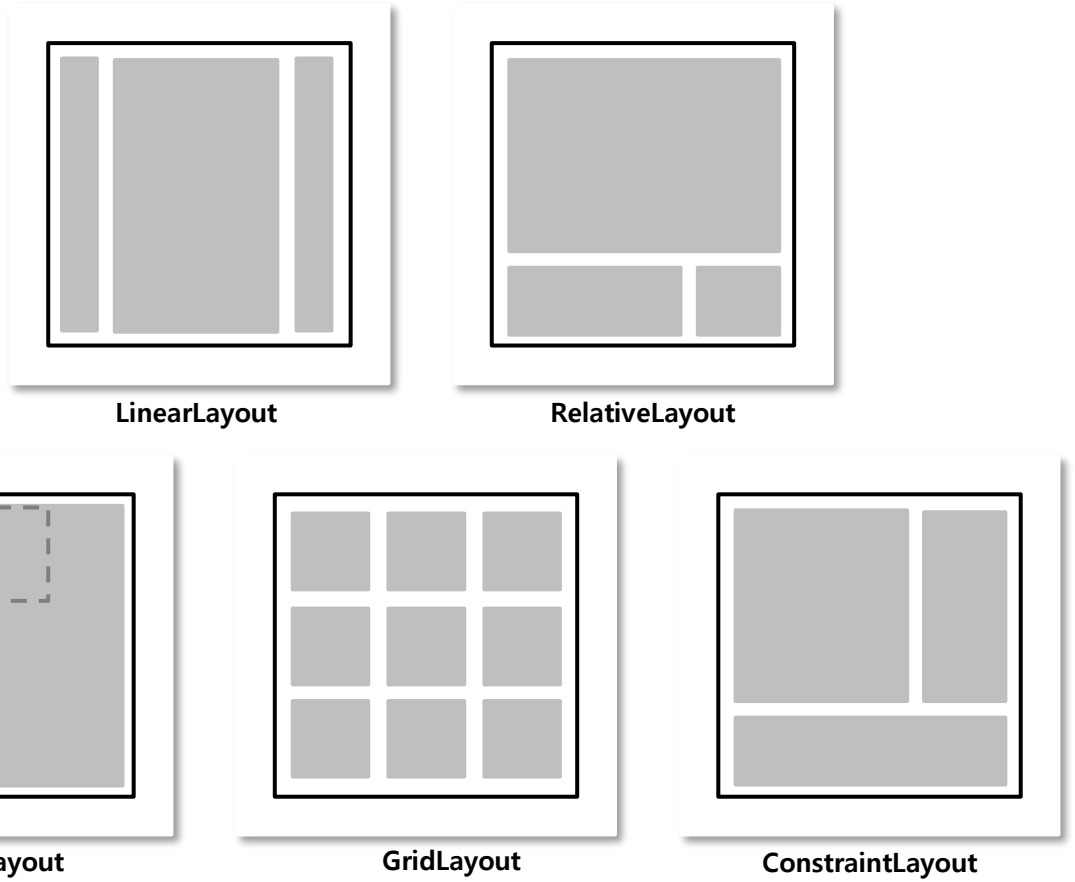
Layout 개요

❖ Layout 이란?

- **View를 배치**하는 **클래스**, **View를 배치**하는 **규칙**

❖ Layout의 종류

- **LinearLayout** - 선형으로 배치
- **RelativeLayout** – 상대 위치로 배치
- **FrameLayout** – 겹쳐서 배치
- **GridLayout** – 표 형태로 배치
- **ConstraintLayout** – 계층 구조로 배치



LinearLayout

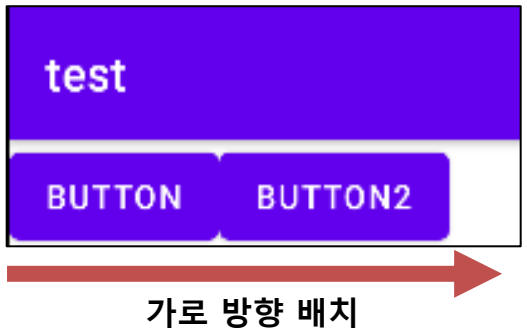
❖ LinearLayout 이란?

- View 를 **가로, 세로 방향으로 나열하는** Layout 클래스

❖ LinearLayout - Orientation 방향 속성

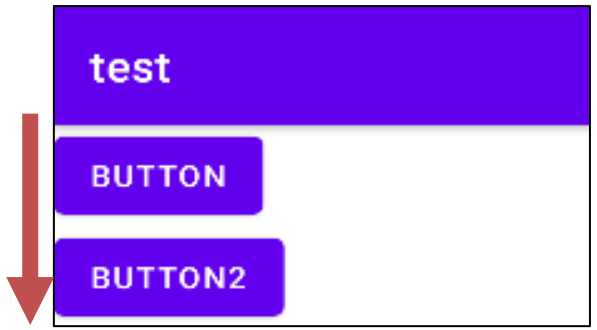
- **horizontal**: View를 **가로**로 나열한다.
- **vertical**: View를 **세로**로 나열한다.

android:orientation="horizontal"



가로 방향 배치

android:orientation="vertical"



세로 방향 배치

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        />
    <Button
        android:id="@+id/button2"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:text="Button2"
        />
</LinearLayout>
```

가로 배치

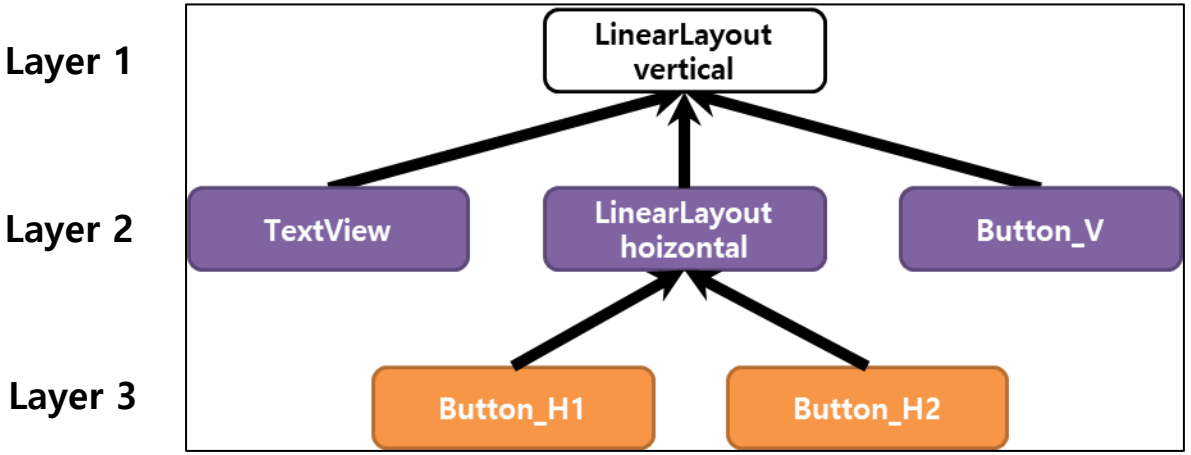
```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        />
    <Button
        android:id="@+id/button2"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:text="Button2"
        />
</LinearLayout>
```

세로 배치

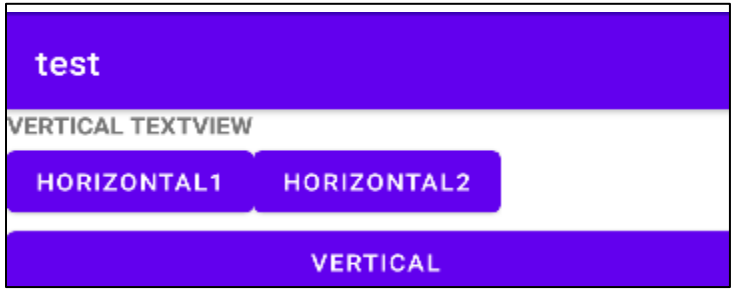
LinearLayout

❖ LinearLayout 에서의 중첩 구조

- **Layout 클래스**도 **View**이므로 **다른 Layout에 포함될 수 있다**.



- **Layout 클래스 안에 다른 Layout 클래스를 추가**함으로써 중첩 Layout 구조를 구현할 수 있다.




```
<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="wrap_content"
        android:text="VERTICAL TEXTVIEW"
        />
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
            <Button
                android:id="@+id/button"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="horizontal1" />

            <Button
                android:id="@+id/button2"
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:text="horizontal2"/>
    </LinearLayout>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="vertical"
        />
</LinearLayout>
```

Layer 1

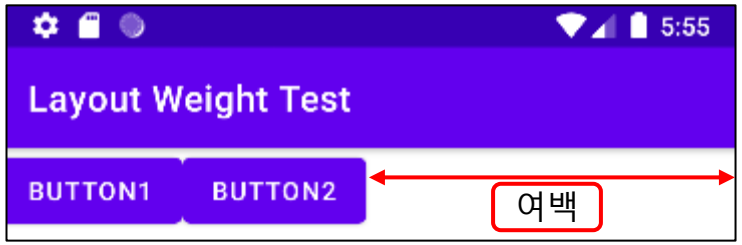
Layer 2

Layer 3

LinearLayout

❖ Layout의 속성 – layout_weight

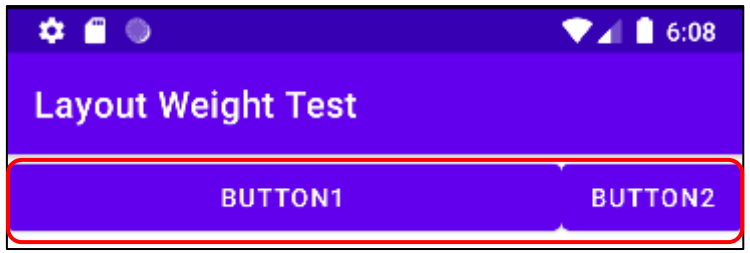
- View에 가중치를 주는 속성으로 **각각의 컴포넌트들의 크기를 조절**해준다.
- View를 배치하다 보면 가로나 세로방향으로 여백이 생길 수 있다.




```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Button1" />

<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="button2" />
```

- layout_weight 속성을 통해 여백을 View로 채울 수 있다.



여백 자동 보정

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="Button1" />

<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="button2" />
```

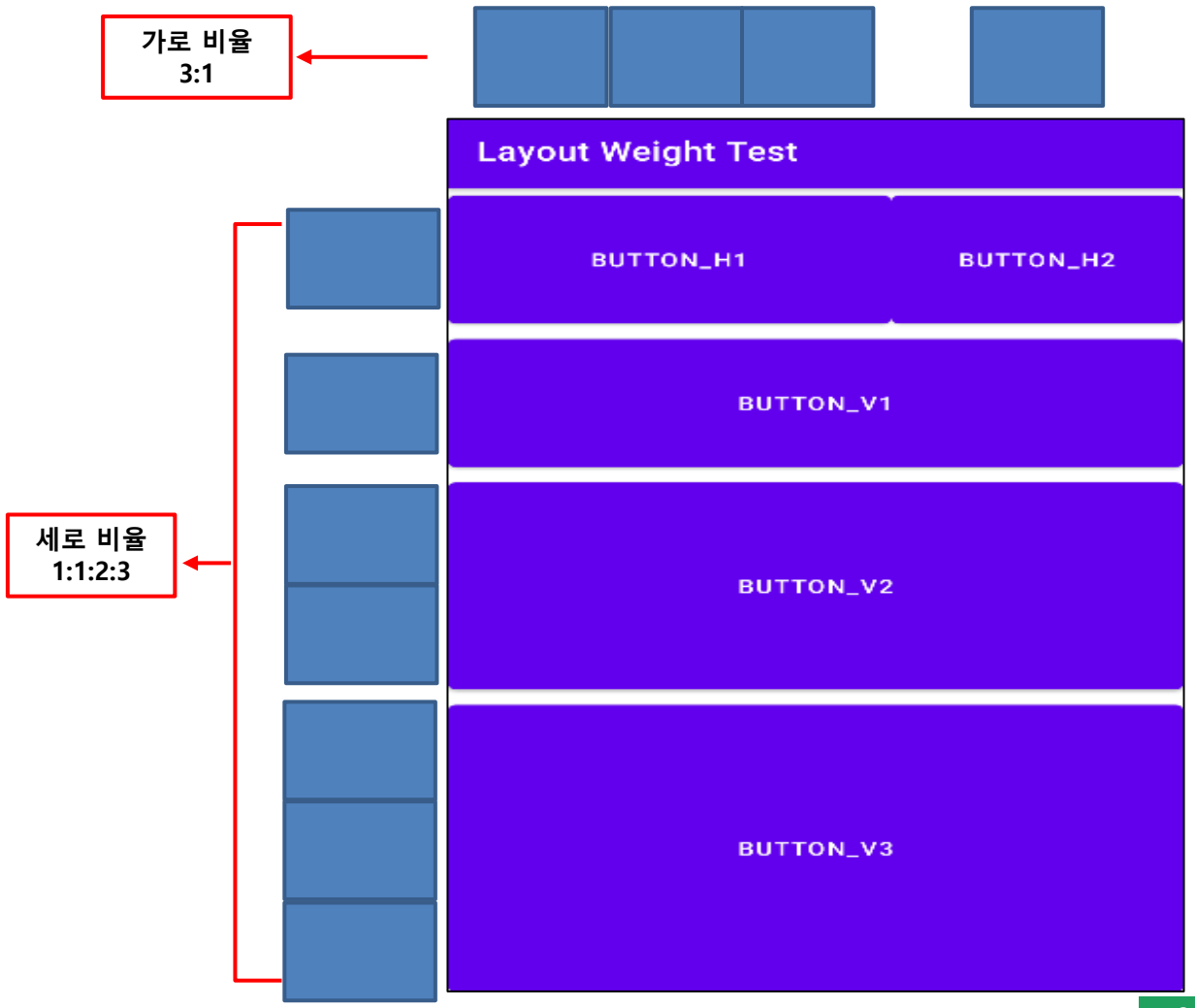
weight 명시

LinearLayout

❖ 중첩된 layout에서 여백 채우기

- 중첩된 layout에서는 **layout_weight**를 통해 **지정된 비율로 화면에 배치**할 수 있다.

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:orientation="horizontal">
        <Button
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_weight="3"
            android:text="Button_h1"/>
        <Button
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:text="Button_h2"/>
    </LinearLayout>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Button_V1"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:text="Button_V2"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="3"
        android:text="Button_V3"/>
</LinearLayout>
```



LinearLayout

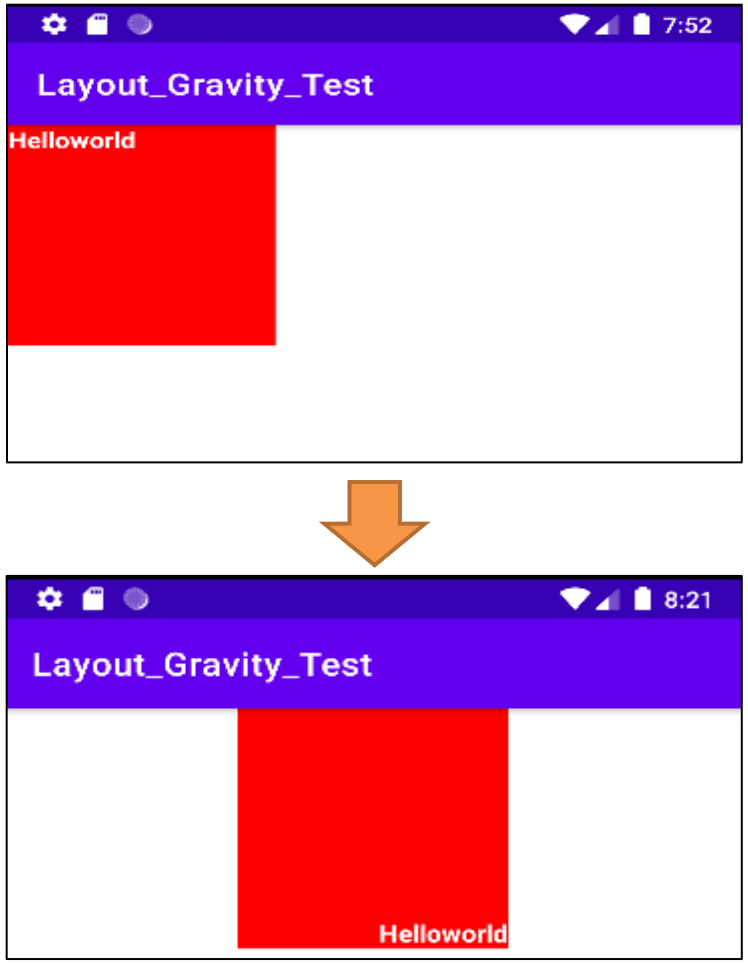
❖ Layout의 속성 – gravity, layout_gravity

- View를 정렬할 때 **gravity, layout_gravity 속성**을 사용한다.
- Gravity, layout_gravity 속성을 사용하지 않으면 **default 값은 left/top**. 즉, 왼쪽상단을 기준으로 정렬한다.
- **Layout_gravity** – 위젯 영역을 정렬한다.
- **Gravity** – View 내부의 콘텐츠(text, etc.)를 정렬한다.

```
<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:background="#FF0000"
        android:text="Helloworld"
        android:textColor="#FFFFFF"
        android:textSize="15dp"
        android:textStyle="bold"
        android:gravity="right|bottom"
        android:layout_gravity="center_horizontal"
        />
</LinearLayout>
```

내부 콘텐츠 정렬

위젯 영역 정렬



RelativeLayout

❖ RelativeLayout 이란?

- **상대 View의 위치를 기준으로 정렬**하는 Layout 클래스

❖ RelativeLayout의 속성

- **배치 속성과 정렬 속성(align)이 존재**함
  - 정렬 속성에는 **View를 기준으로 정렬하는 방법**, **부모 Layout을 기준으로 정렬하는 방법**이 있음

* 기준 View는 id 선언이 필수적이다.

속성	기능
layout_	기준 View 배치 (above, below, toLeftOf, toRightOf)
layout_align	기준 View 정렬 (Top, Bottom, Left, Right)
layout_alignParent	부모 View의 방향에 맞춤 (Top, Bottom, Left, Right)

```
android:layout_toRightOf="@id/image"
android:text="toRightOf"
```

```
android:layout_toLeftOf="@id/image"
android:text="toLeftOf"
```

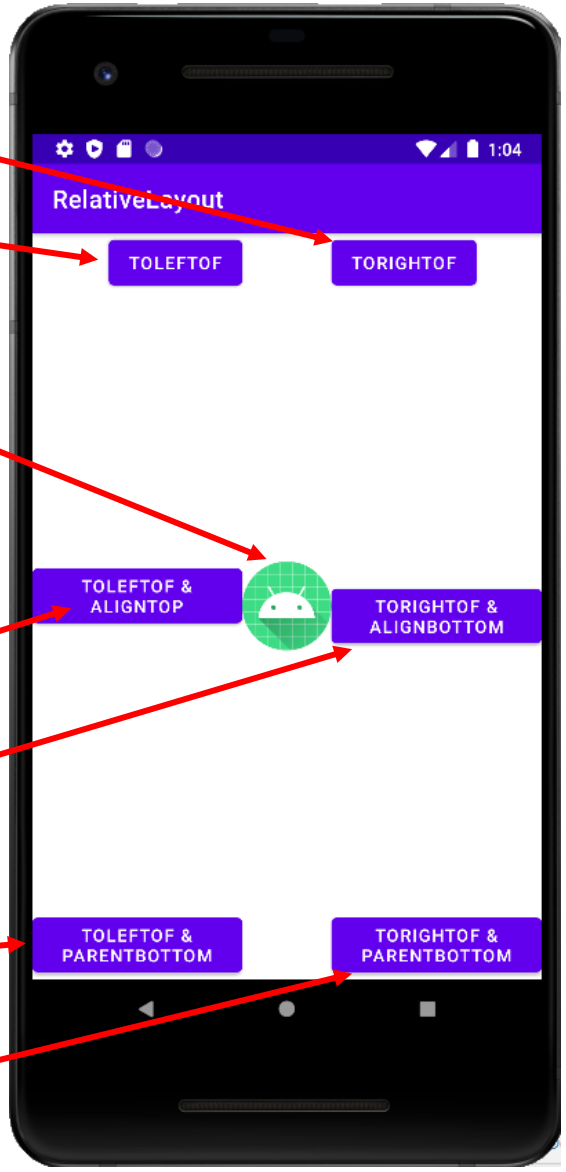
```
<ImageView android:id="@+id/image"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:src="@mipmap/ic_launcher" />
```

```
android:layout_alignTop="@id/image"
android:layout_toLeftOf="@+id/image"
android:text="toLeftOf &amp; alignTop"
```

```
android:layout_alignBottom="@id/image"
android:layout_toRightOf="@+id/image"
android:text="toRightOf &amp; alignBottom"
```

```
android:layout_alignParentBottom="true"
android:layout_toLeftOf="@id/image"
android:text="toLeftOf &amp; ParentBottom"
```

```
android:layout_alignParentBottom="true"
android:layout_toRightOf="@id/image"
android:text="toRightOf &amp; ParentBottom"
```

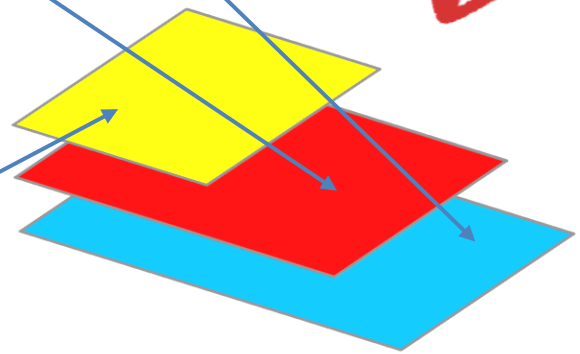


FrameLayout

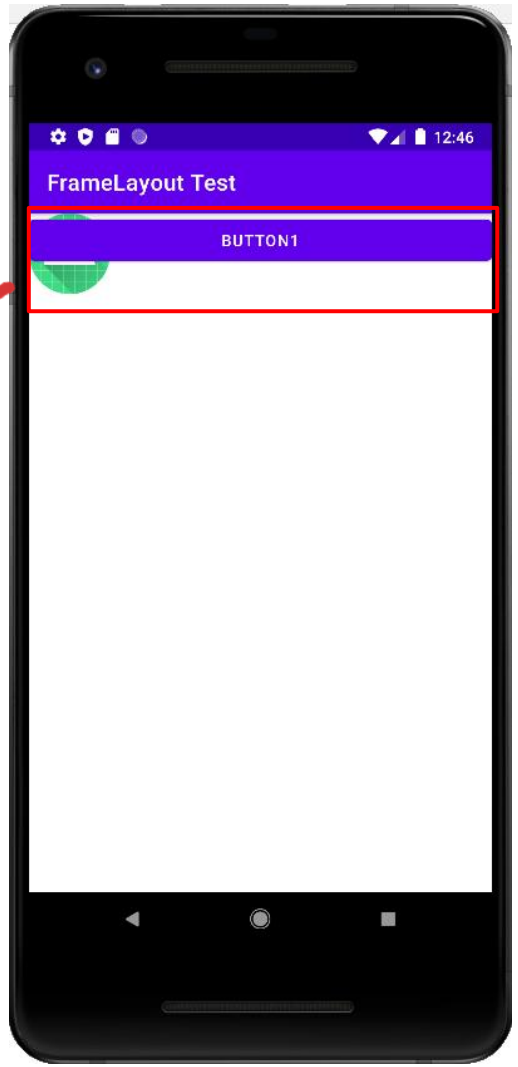
❖ FrameLayout 이란?

- **View를 겹쳐서 출력**하는 Layout 클래스이다.
- View를 추가한 **순서대로 위에 겹쳐서** 계속 출력하는 Layout이다.
- 겹쳐진 View 중에서 **특정 순간 하나의 View**만 출력할 때 사용한다.
- 액티비티 코드에서 원하는 순간에 View의 visibility 속성값을 바꾸어 조절한다.

```
<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <ImageView
        android:id="@+id/imV"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@mipmap/ic_launcher"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Button1" />
</FrameLayout>
```



Widget 중첩



GridLayout

❖ GridLayout이란?

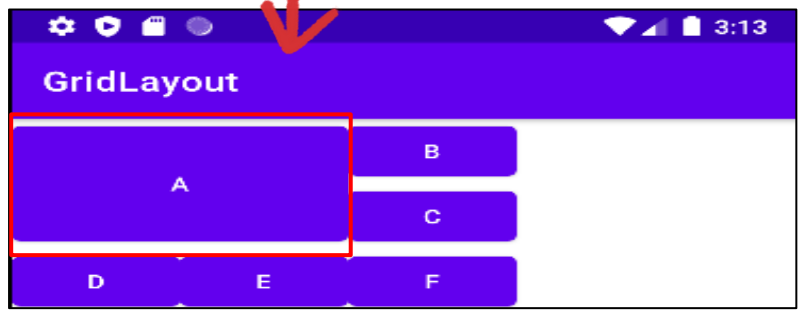
- **행과 열로 구성된 테이블 화면**을 만드는 Layout

❖ GridLayout 의 설명

- GridLayout의 배치 규칙
  - **Orientation** – View를 나열할 방향을 설정한다.
  - **rowCount, columnCount** – 가로·세로로 나열할 View의 개수를 지정한다.
- GridLayout의 특정 View의 위치 조정
  - **layout_row, layout_column** – View가 위치할 가로, 세로 방향 인덱스를 지정
- GridLayout의 특정 View의 크기 확장 및 병합
  - **layout_gravity** – 특정 View를 확장해서 출력한다.
    - fill, fill_horizontal, fill_vertical 등 옵션이 존재한다.
  - **Layout_columnSpan, layout_rowSpan**: 가로 또는 세로로 병합한다.

```
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:columnCount="3"
    android:orientation="horizontal">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_rowSpan="2"
        android:layout_columnSpan="2"
        android:layout_gravity="fill"
        android:text="A" />
    <Button...>
    <Button...>
    <Button...>
    <Button...>
    <Button...>
</GridLayout>
```

column(2) + row(2) = 4



실습

실습 목표와 구성

1. 기초(따라하기)

- LinearLayout 기반 View 배치 1
- LinearLayout 기반 View 배치 2
- RelativeLayout 기반 View 배치
- FrameLayout 기반 View 배치
- GridLayout 기반 View 배치

2. 심화(완성하기)

- 전화 앱의 키패드 화면 만들기

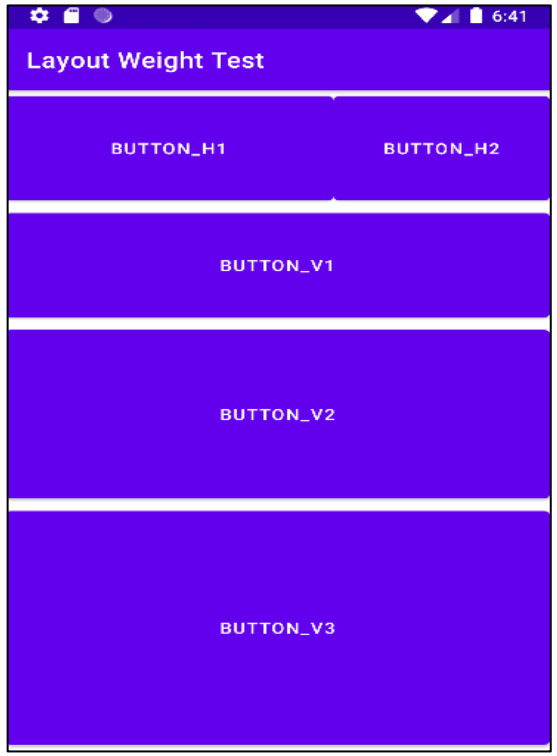
3. 심화(과제물)

- 카카오톡 채팅방 만들기

기초(따라하기) – 예제 1

❖ LinearLayout 기반 View 배치 1

- (10, 15, 20, 26, 31, 36 lines) **layout_weight** : Layout을 특정 비율로 정렬



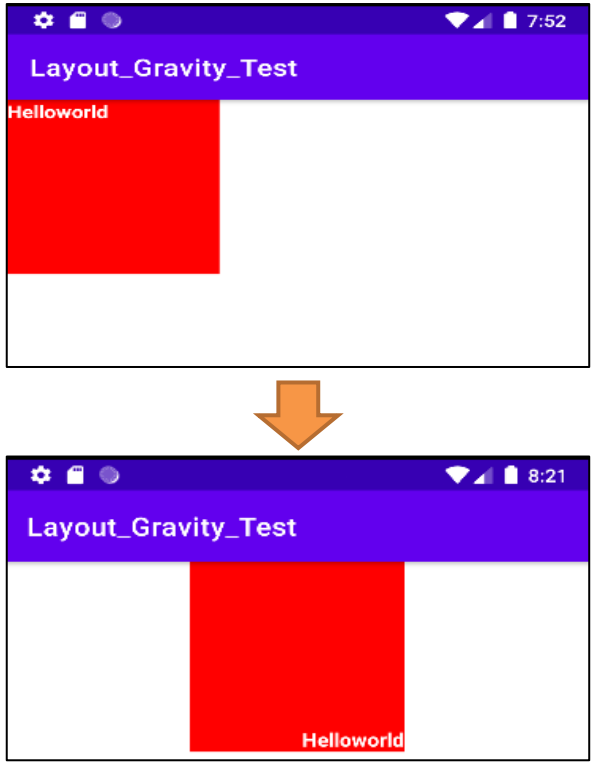
layout_main.xml

```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:orientation="vertical">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:orientation="horizontal">
        <Button
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_weight="3"
            android:text="Button_h1"/>
        <Button
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:text="Button_h2"/>
    </LinearLayout>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Button_V1"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:text="Button_V2"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="3"
        android:text="Button_V3"/>
</LinearLayout>
```

기초(따라하기) – 예제 2

❖ LinearLayout 기반 View 배치 2

- (11 line) **layout_gravity**: 부모 Layout을 기준으로 옵션값에 따라 정렬
- (13 line) **gravity**: 해당 View 내부의 데이터를 옵션에 따라 정렬



layout_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:orientation="vertical">
7
8      <TextView
9          android:layout_width="150dp"
10         android:layout_height="150dp"
11         android:layout_gravity="center_horizontal"
12         android:background="#FF0000"
13         android:gravity="right|bottom"
14         android:text="Helloworld"
15         android:textColor="#FFFFFF"
16         android:textSize="15dp"
17         android:textStyle="bold" />
18 </LinearLayout>
```

기초(따라하기) – 예제 3

❖ RelativeLayout 기반 View 배치

- (12 line) **src**: 이미지 View에 표시될 drawable 설정

- 참고사항
  - **layout_ 속성** : 기준 View의 배치
    - "above" : 위, "below" : 아래, "toLeftOf" : 좌, "toRightOf" : 우
  - **Layout_align 속성** : 기준 View의 맞춤
    - "Top" : 위, "Bottom" : 아래, "Left" : 좌, "Right" : 우
  - **Layout_alignParent 속성** : 부모 View의 방향에 맞춤
    - "Top" : 위, "Bottom" : 아래, "Left" : 좌, "Right" : 우

layout_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent">
6
7      <ImageView
8          android:id="@+id/image"
9          android:layout_width="wrap_content"
10         android:layout_height="wrap_content"
11         android:layout_centerInParent="true"
12         android:src="@mipmap/ic_launcher" />
13
14     <Button
15         android:layout_width="wrap_content"
16         android:layout_height="wrap_content"
17         android:layout_toRightOf="@id/image"
18         android:text="toRightOf" />
19
20     <Button
21         android:layout_width="wrap_content"
22         android:layout_height="wrap_content"
23         android:layout_toLeftOf="@id/image"
24         android:text="toLeftOf" />
25
26     <Button
27         android:layout_width="wrap_content"
28         android:layout_height="wrap_content"
29         android:layout_toLeftOf="@id/image"
30         android:text="toLeftOf" />
31
32     <Button
33         android:layout_width="wrap_content"
34         android:layout_height="wrap_content"
35         android:layout_alignBottom="@id/image"
36         android:layout_toRightOf="@+id/image"
37         android:text="toRightOf &amp;amp; alignBottom" />
```

기초(따라하기) – 예제 3

❖ **RelativeLayout 기반 View 배치**

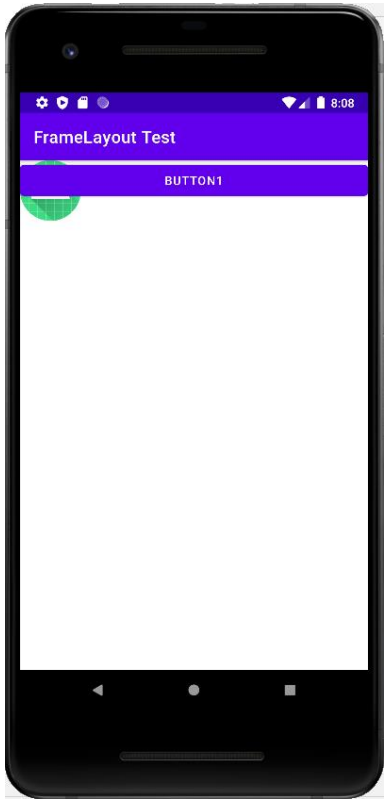
layout_main.xml

```
39      <Button
40          android:layout_width="wrap_content"
41          android:layout_height="wrap_content"
42          android:layout_alignTop="@id/image"
43          android:layout_toLeftOf="@+id/image"
44          android:text="toLeftOf &amp; alignTop" />
45
46      <Button
47          android:layout_width="wrap_content"
48          android:layout_height="wrap_content"
49          android:layout_alignParentBottom="true"
50          android:layout_toLeftOf="@id/image"
51          android:text="toLeftOf &amp; ParentBottom" />
52
53      <Button
54          android:layout_width="wrap_content"
55          android:layout_height="wrap_content"
56          android:layout_alignParentBottom="true"
57          android:layout_toRightOf="@id/image"
58          android:text="toRightOf &amp; ParentBottom" />
59  </RelativeLayout>
```

기초(따라하기) – 예제 4

❖ FrameLayout 기반 View 배치

- (3 line) LinearLayout 선언부
- (4, 5, 9, 32 line) **match_parent** : 부모 Layout 크기에 맞춤
- (10, 15, 20, 21, 26, 27, 33 line) **wrap_content**: 내용물 크기에 따라 사이즈 지정



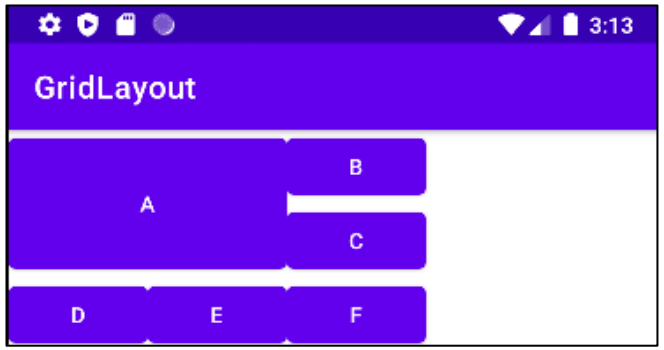
layout_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <layout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools">
4
5
6      <FrameLayout
7          android:layout_width="match_parent"
8          android:layout_height="match_parent"
9          tools:context=".MainActivity">
10
11         <ImageView
12             android:id="@+id/imV"
13             android:layout_width="wrap_content"
14             android:layout_height="wrap_content"
15             android:src="@mipmap/ic_launcher" />
16
17         <Button
18             android:layout_width="match_parent"
19             android:layout_height="wrap_content"
20             android:text="Button1" />
21     </FrameLayout>
22
23
24 </layout>
```

기초(따라하기) – 예제 5

❖ GridLayout 기반 View 배치

- (5 line) **columnCount**: GridLayout의 열 개수 지정
  - rowCount: GridLayout의 행 개수 지정
- (11 line) **rowSpan**: 해당 View를 속성의 수치만큼 행 방향으로 병합
- (12 line) **columnSpan**: 해당 View를 속성의 수치만큼 열방향으로 병합
- (13 line) layout_gravity= “fill”: Span옵션으로 병합한 만큼 View를 확장



layout_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent"
5      android:columnCount="3"
6      android:orientation="horizontal">
7
8      <Button
9          android:layout_width="wrap_content"
10         android:layout_height="wrap_content"
11         android:layout_rowSpan="2"
12         android:layout_columnSpan="2"
13         android:layout_gravity="fill"
14         android:text="A" />
15
16     <Button
17         android:layout_width="wrap_content"
18         android:layout_height="wrap_content"
19         android:text="B" />
20
21     <Button
22         android:layout_width="wrap_content"
23         android:layout_height="wrap_content"
24         android:text="C" />
25
26     <Button
27         android:layout_width="wrap_content"
28         android:layout_height="wrap_content"
29         android:text="D" />
30
31     <Button
32         android:layout_width="wrap_content"
33         android:layout_height="wrap_content"
34         android:text="E" />
35
36     <Button
37         android:layout_width="wrap_content"
38         android:layout_height="wrap_content"
39         android:text="F" />
40
41 </GridLayout>
```

심화(완성하기) – 예제 6

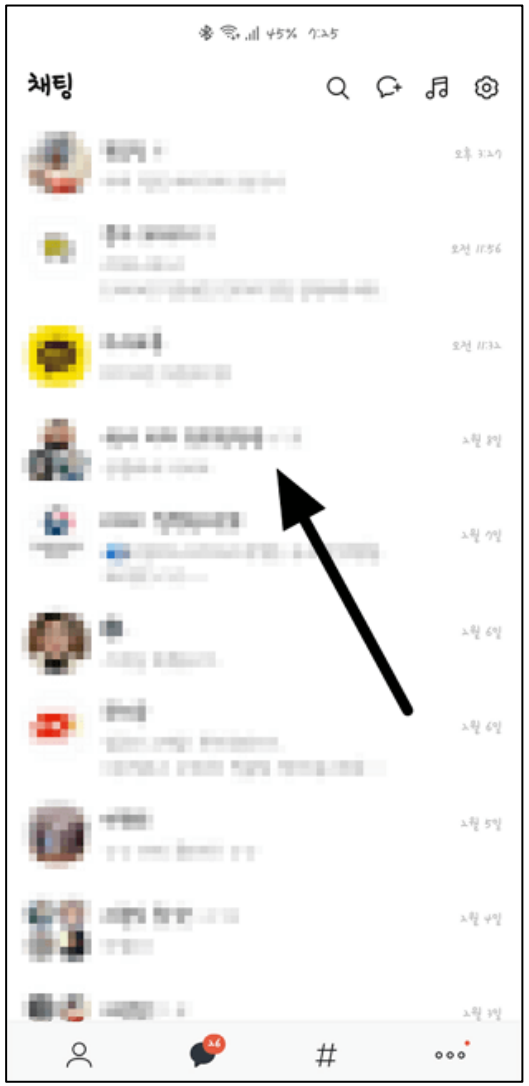
❖ 전화 앱의 키패드 화면 만들기

- 조건
  - 중첩 Layout 사용할 것
    - 단, LinearLayout 만 사용
  - EditText 배치할 것
    - 기능 x
  - 제공하는 이미지 사용할 것
    - 영상통화버튼, 전화버튼, 삭제버튼 이미지
    - 해당 버튼은 ImageView를 사용할 것

심화(과제물) – 예제 7

❖ 카카오톡 채팅 목록구현

- 조건
  - 카카오톡 채팅방 목록 레이아웃을 완성할 것
    - 기능 동작 x
  - 이미지를 사용할 것
    - app/res/drawble 에서 이미지 로드 할 것
  - 전체 레이아웃이 가지런히 정렬되어야 함
    - Layout_weight, padding 등 속성을 사용할 것
    - Layout 관련 속성 수치로 정렬 금지
      - » ex. 10dp

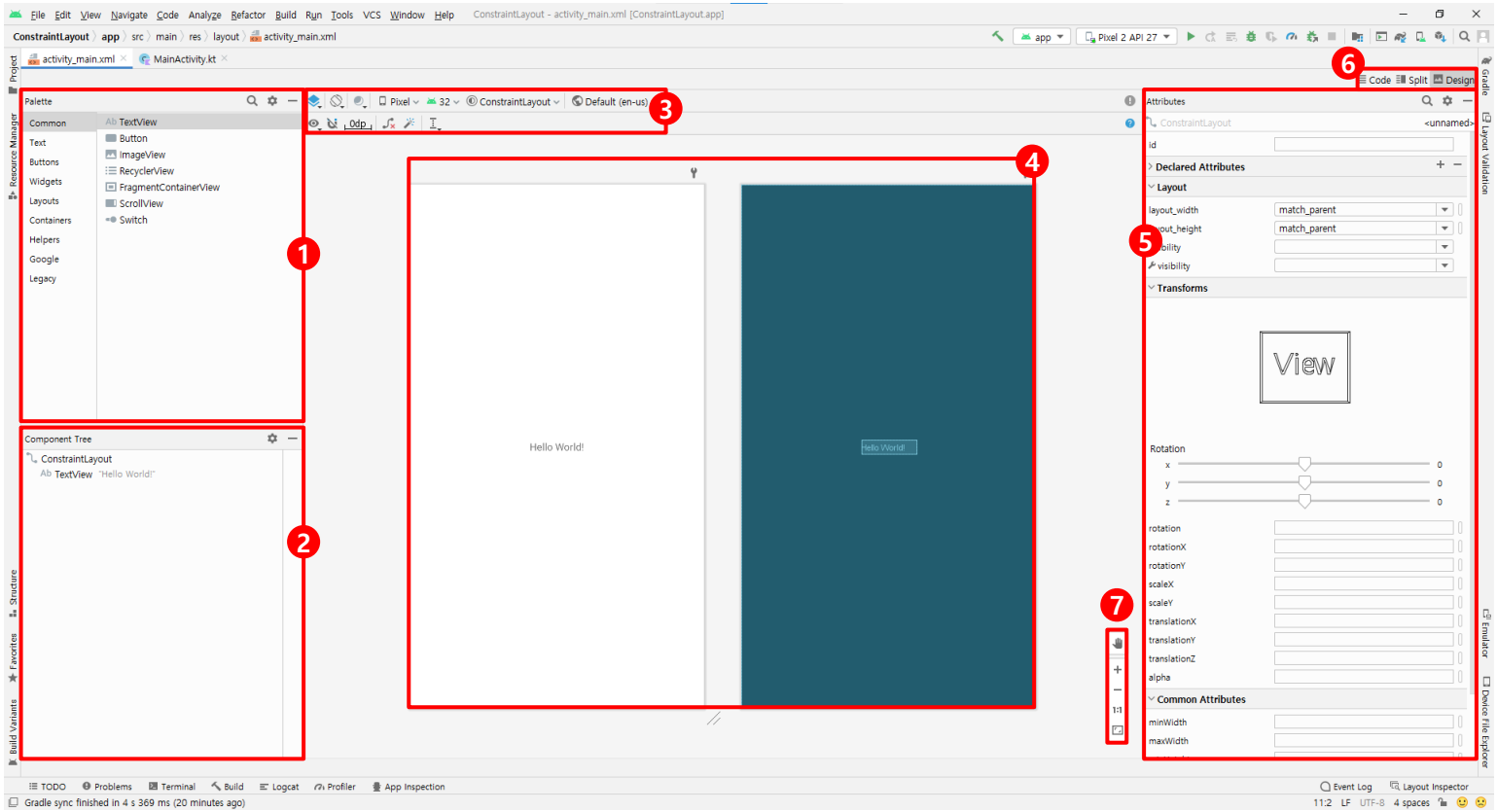


별첨 - ConstraintLayout

❖ ConstraintLayout이란?

- RelativeLayout과 비슷하게 **View를 상대위치로 배치**하지만 더 많은 속성을 제공
- 대량의 속성으로 인해 XML파일에 직접 작업하기는 부담이기 때문에 **Layout 편집기를 사용**한다.

❖ Layout 편집기



1. **팔레트(Palette):** Layout을 구성할 수 있는 다양한 View와 View 그룹
2. **컴포넌트 트리(Component Tree):** Layout에서 구성요소의 계층구조 표시
3. **툴바(Toolbar):** Layout 속성 설정
4. **디자인 편집기(Design editor):** 디자인 View나 청사진 View 작업 영역
5. **속성(Attributes):** 선택한 View의 속성을 지정
6. **보기 모드(View mode):** Layout 모드 선택. 'Code(코드)', 'Split(분할)', 'Design(디자인)' 중 선택
7. **화면 조절(Zoom and pan control):** 확대/축소 및 화면 이동 조절