

4. 레이아웃

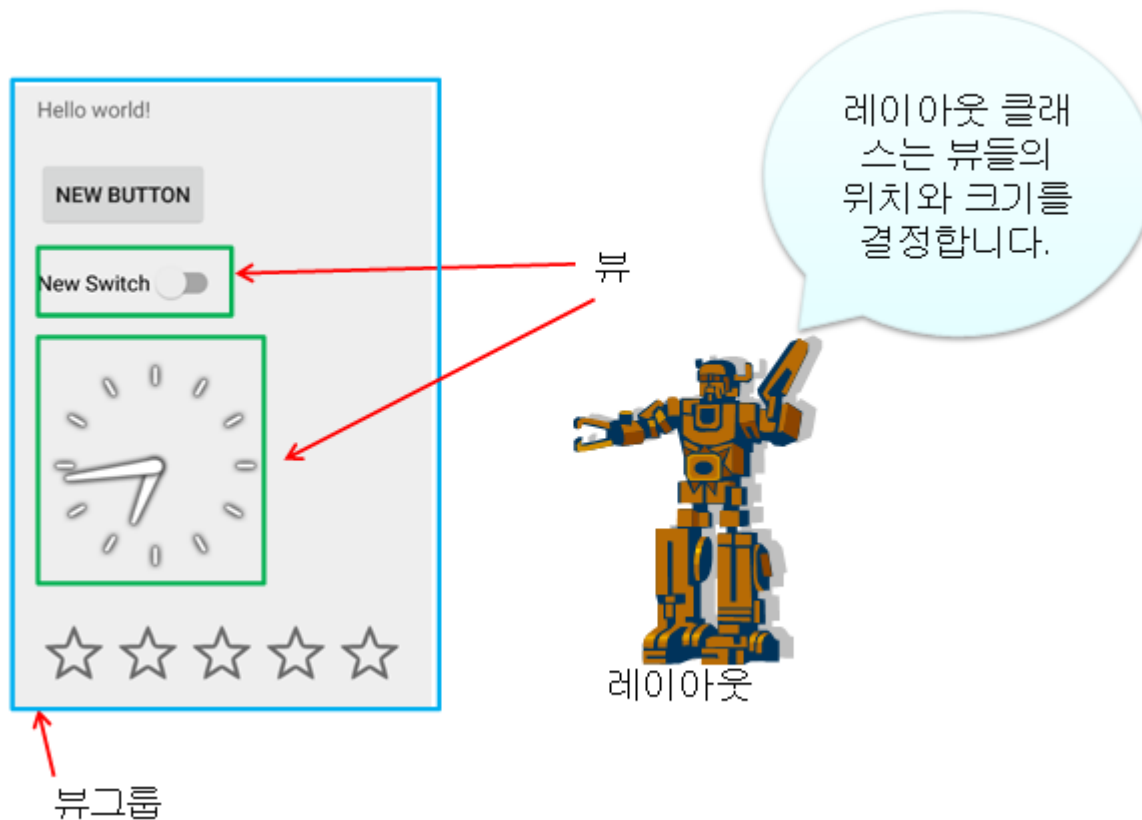
1. 레이아웃의 개념을 익힘
2. 화면을 다양한 레이아웃으로 구성
3. Java 코드만으로 화면을 작성



1. 레이아웃 개요

□ 레이아웃

- ▣ 뷰들을 화면에 배치하는 방법



1. 레이아웃 개요

□ 레이아웃 계층도

- ▣ ViewGroup 클래스로부터 상속

```
java.lang.Object
└ android.view.View
    └ android.widget.ViewGroup
        └ android.widget.LinearLayout
            └ android.widget.TableLayout
        └ android.widget.RelativeLayout
        └ android.widget.FrameLayout
        └ android.widget.GridLayout
```

1. 레이아웃 개요

□ 레이아웃 기본 개념[1/2]

■ 레이아웃에서 자주 사용되는 속성

- orientation : 레이아웃 안에 배치할 위젯의 수직 또는 수평 방향을 설정
- gravity : 레이아웃 안에 배치할 위젯의 정렬 방향을 좌측, 우측, 중앙으로 설정
- padding : 레이아웃 안에 배치할 위젯의 여백을 설정
- layout_weight : 레이아웃이 전체 화면에서 차지하는 공간의 가중 값을 설정, 여러 개의 레이아웃이 중복될 때 주로 사용
- baselineAligned : 레이아웃 안에 배치할 위젯들을 보기 좋게 정렬

1. 레이아웃 개요

▣ 레이아웃의 종류



(a) 리니어레이아웃



(b) 렐러티브레이아웃



(c) 테이블레이아웃



(d) 그리드레이아웃



(e) 프레임레이아웃

□ 레이아웃 종류

- 리니어레이아웃 : 왼쪽 위부터 아래쪽 또는 오른쪽으로 차례로 배치
- 렐러티브레이아웃 : 위젯 자신이 속한 레이아웃의 상하좌우의 위치를 지정하여 배치
- 테이블레이아웃 : 위젯을 행과 열의 개수를 지정한 테이블 형태로 배열
- 그리드레이아웃 : 테이블레이아웃과 비슷하지만, 행 또는 열을 확장하여 다양하게 배치할 때 더 편리
- 프레임레이아웃 : 위젯들을 왼쪽 위에 일률적으로 겹쳐서 배치하여 중복해서 보이는 효과를 냄

2. 리니어 레이아웃

□ 기본 리니어레이아웃 형태[1/4]

▣ orientation 속성

■ 리니어레이아웃의 가장 기본적인 속성

■ *Vertical* : 리니어레이아웃 안에 포함될 위젯의 배치를 수직방향으로 쌓음

■ *Horizontal* : 수평 방향으로 쌓겠다는 의미

■ orientation 속성이 vertical과 horizontal 값인 값인 경우



2. 리니어 레이아웃

□ 기본 리니어레이아웃 형태

▣ gravity 속성

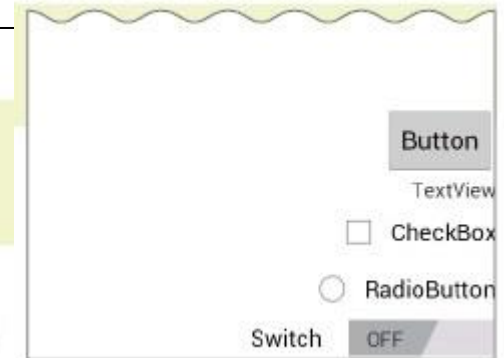
- gravity 속성은 레이아웃 안의 위젯들을 어디에 배치할 것인지를 결정

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 우단에 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 컨테이너 가득 채우도록 배치

2. 리니어 레이아웃

□ gravity 예

```
1 <LinearLayout
2     android:orientation="vertical"
3     android:gravity="right|bottom" >
4     <Button
5         android:layout_width="wrap_content"
6         android:layout_height="wrap_content"
7         android:text="Button" />
8     <TextView
9         android:text="TextView" />
10     ~~~ 중간 생략 ~~~
11 </LinearLayout>
```



□ 기본 리니어레이아웃 형태[3/4]

▣ layout_gravity 속성

- layout_gravity는 자신의 위치를 부모의 어디쯤에 위치시킬지를 결정

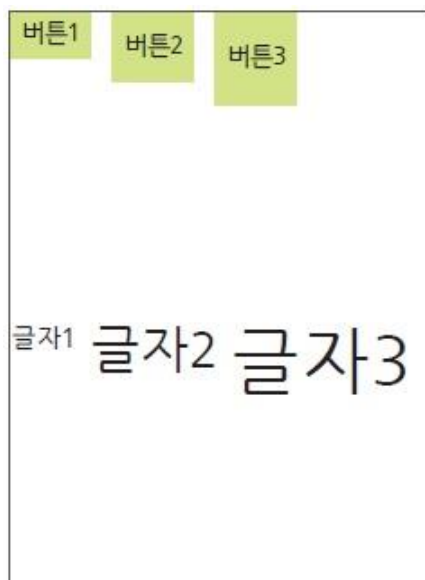
```
1 <LinearLayout
2     android:orientation="vertical" >
3     <Button
4         android:layout_gravity="right"
5         android:text="오른쪽" />
6     <Button
7         android:layout_gravity="center"
8         android:text="중앙" />
9     <Button
10        android:layout_gravity="left"
11        android:text="왼쪽" />
12 </LinearLayout>
```



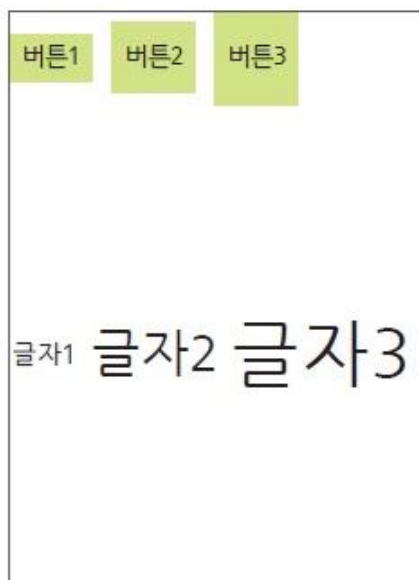
□ 기본 리니어레이아웃 형태[4/4]

▣ baselineAligned 속성

- baselineAligned 속성은 크기가 다른 위젯들을 보기 좋게 정렬함
- true와 false 값을 가질 수 있음



(a) false로 지정



(b) 생략하거나 true로 지정

그림 5-3 baselineAligned 속성

□ 중복 리니어레이아웃 형태[1/5]

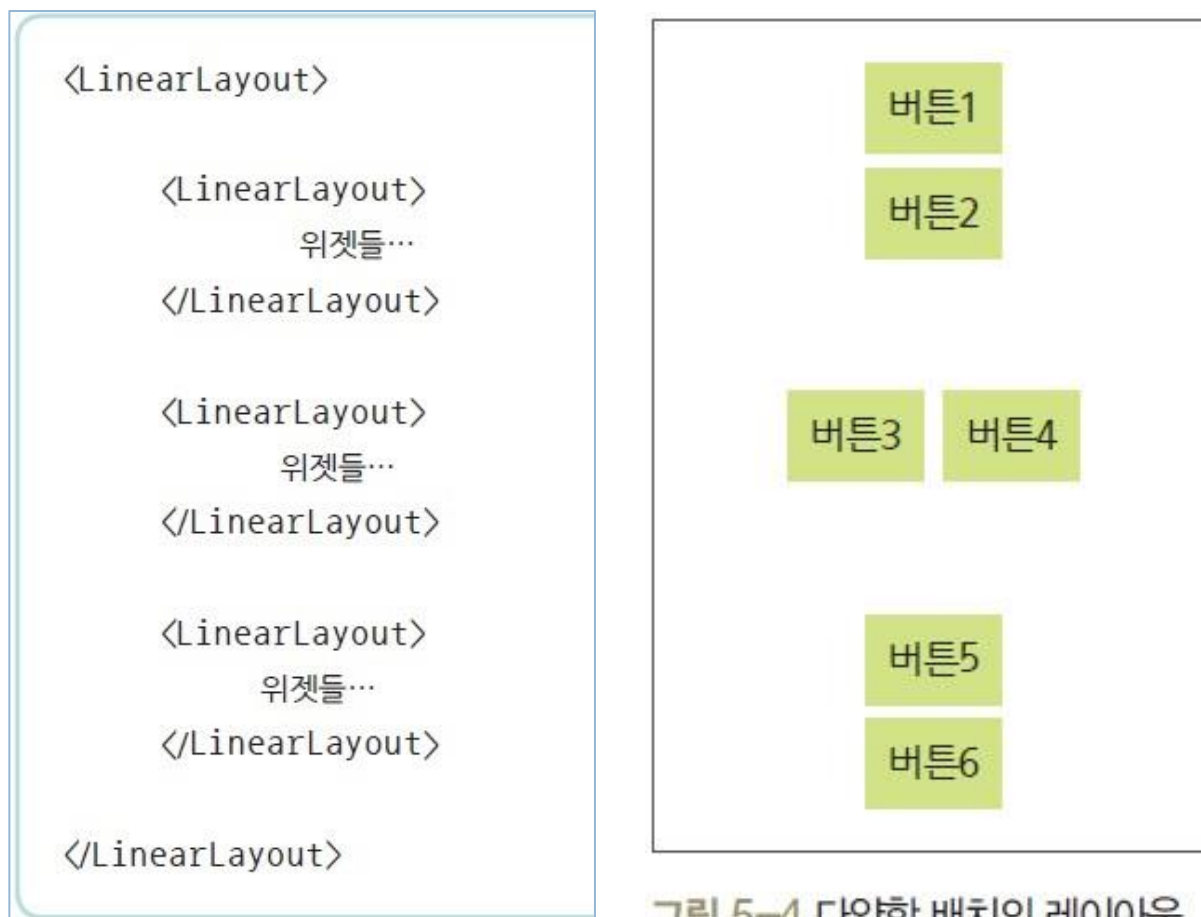
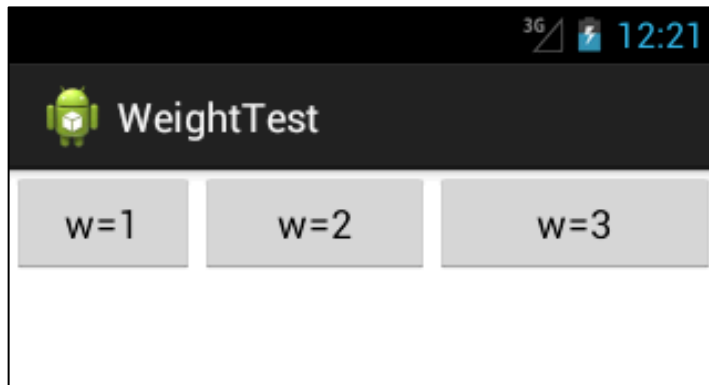


그림 5-4 다양한 배치의 레이아웃

□ 가중치(weight)

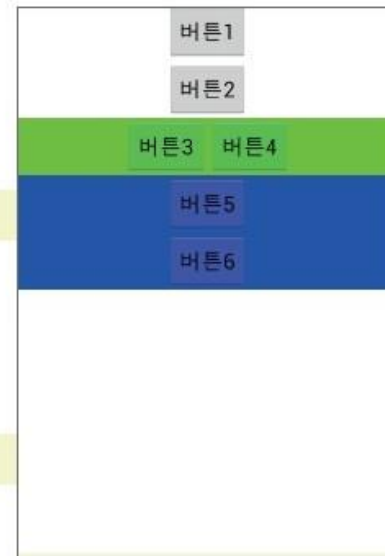
- 선형 레이아웃의 자식 뷰들의 가중치가 각각 1, 2, 3이면, 남아있는 공간의 $1/6$, $2/6$, $3/6$ 을 각각 할당받는다.



□ 중복 리니어레이아웃 형태

▣ layout_height를 wrap_content로 변경

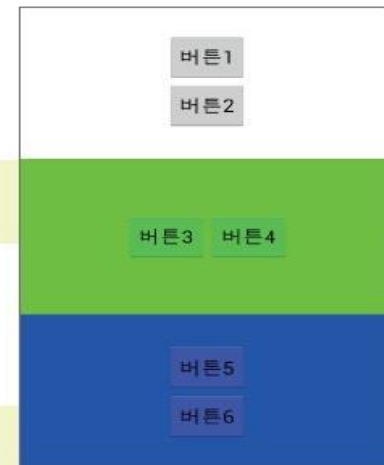
```
1 <LinearLayout
2     android:orientation="vertical" >
3     <LinearLayout
4         android:layout_width="match_parent"
5         android:layout_height="wrap_content"
6         ~~~~ 중간 생략 ~~~~
7     </LinearLayout>
8     <LinearLayout
9         android:layout_width="match_parent"
10        android:layout_height="wrap_content"
11        ~~~~ 중간 생략 ~~~~
12    </LinearLayout>
13    <LinearLayout
14        android:layout_width="match_parent"
15        android:layout_height="wrap_content"
16        ~~~~ 중간 생략 ~~~~
17    </LinearLayout>
18 </LinearLayout>
```



□ 중복 리니어레이아웃 형태

▣ layout_weight를 1로 지정

```
1 <LinearLayout
2     android:orientation="vertical" >
3     <LinearLayout
4         android:layout_width="match_parent"
5         android:layout_height="match_parent"
6         android:layout_weight="1"
7         ~~~~ 중간 생략 ~~~~
8     </LinearLayout>
9     <LinearLayout
10        android:layout_width="match_parent"
11        android:layout_height="match_parent"
12        android:layout_weight="1"
13        ~~~~ 중간 생략 ~~~~
14    </LinearLayout>
15    <LinearLayout
16        android:layout_width="match_parent"
17        android:layout_height="match_parent"
18        android:layout_weight="1"
19        ~~~~ 중간 생략 ~~~~
20    </LinearLayout>
21 </LinearLayout>
```



□ 중복 리니어레이아웃 형태[4/5]

▶ 직접 풀어보기 5-2

리니어레이아웃으로 다음 화면을 구성하는 XML을 작성하라. 단, 레이아웃은 구분되어 보이도록 서로 다른 색으로 지정한다.

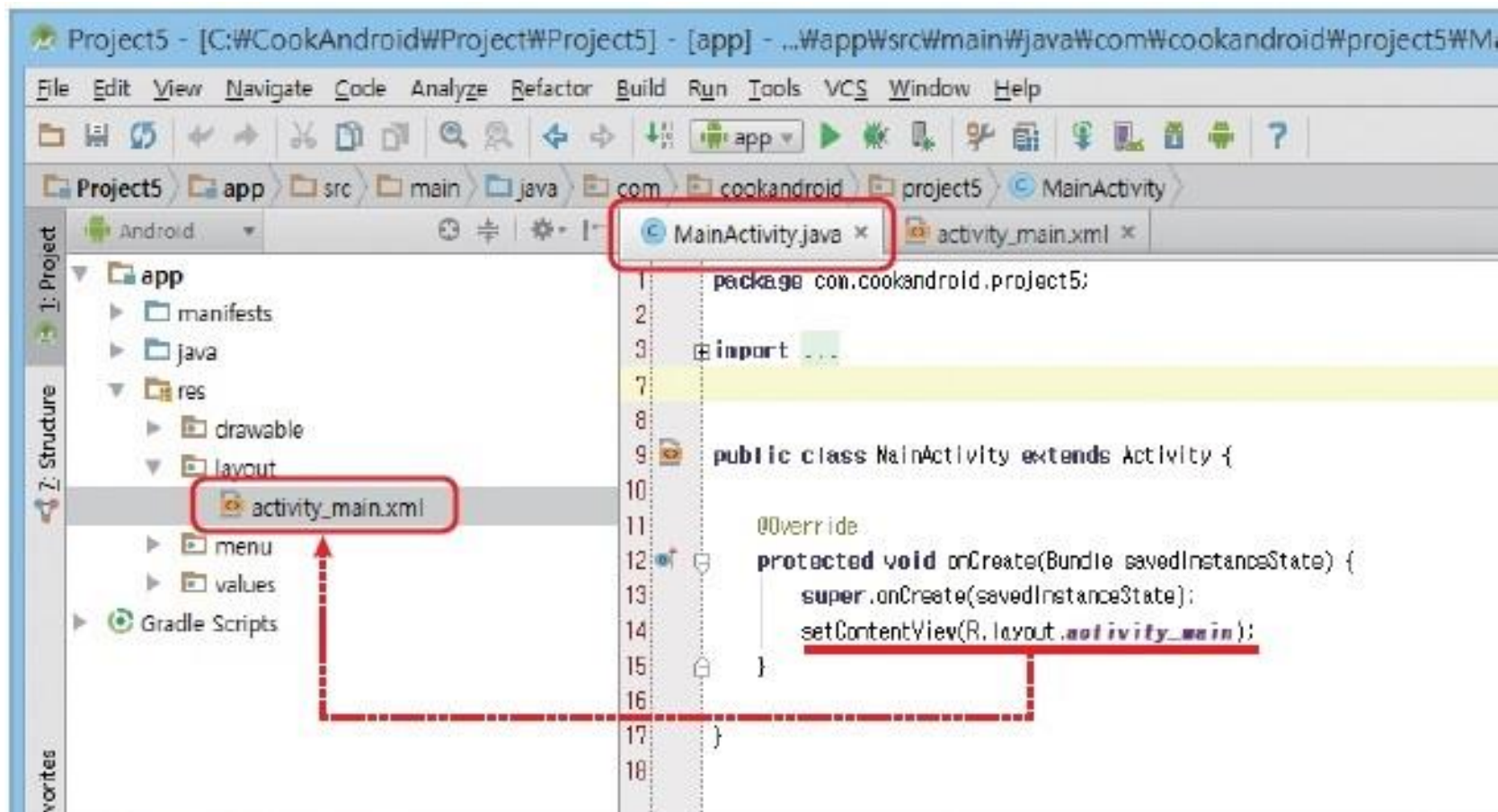
HINT 레이아웃 안에 다시 레이아웃을 여러 번 중첩해도 된다.



그림 5-5 중첩 리니어레이아웃

□ Java 코드로 화면 만들기[1/5]

▣ XML과 Java 코드 동작 방식



□ Java 코드로 화면 만들기[2/5]- [실습]

- 버튼을 클릭하면 토스트 메시지가 출력되는 화면을 Java로 코딩해보자.
- 안드로이드 프로젝트 생성
 - 프로젝트 이름 : Project5_1
 - 패키지 이름 : com.cookandroid.project5_1
- 화면 디자인 및 편집
 - main.xml 삭제
- Java 코드 작성 및 수정
 - main.xml 삭제했기 때문에 오류가 발생함
 - //를 붙여 주석으로 처리한 후 진행

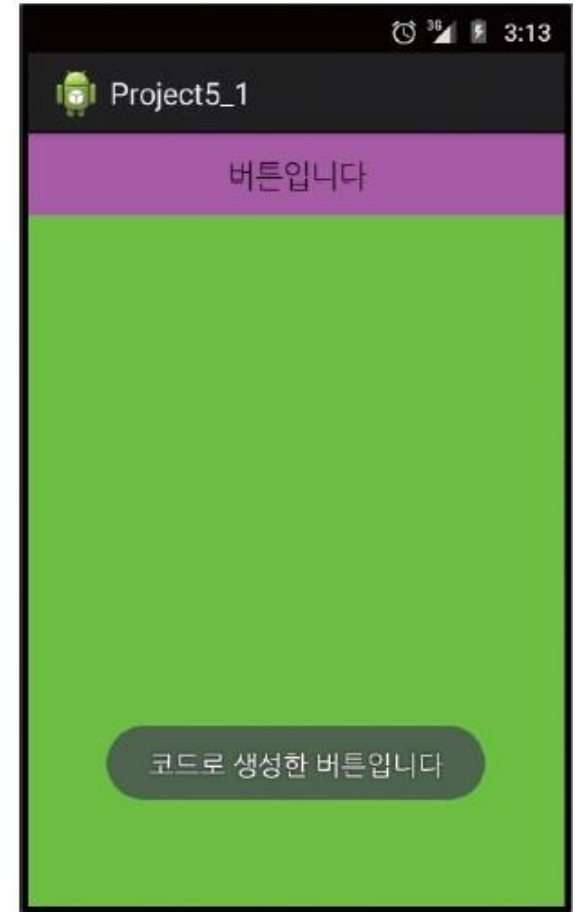
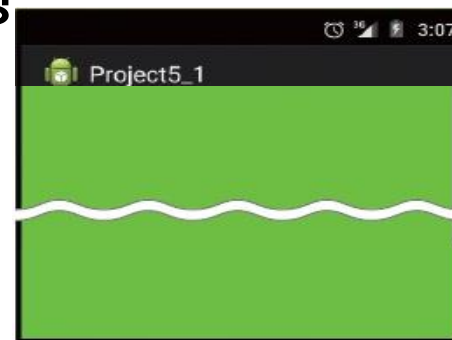


그림 5-7 Java 코드만 사용한 프로젝트

□ Java 코드로 화면 만들기[2/5]- [실습]

▣ 리니어레이아웃을 생성하는 코드를 작성하고 실행

```
1 public void onCreate(Bundle savedInstanceState) {  
2     super.onCreate(savedInstanceState);  
3     // setContentView(R.layout.activity_main);  
4  
5     LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(  
6         LinearLayout.LayoutParams.MATCH_PARENT,  
7         LinearLayout.LayoutParams.MATCH_PARENT);  
8  
9     LinearLayout baseLayout = new LinearLayout(this);  
10    baseLayout.setOrientation(LinearLayout.VERTICAL);  
11    baseLayout.setBackgroundColor(Color.rgb(0, 255, 0));  
12    setContentView(baseLayout,params);  
13 }
```



□ Java 코드로 화면 만들기[3/5]- [실습]

- 버튼을 만들고, 버튼을 클릭했을 때 토스트 메시지를 작성 (onCreate() 안에 이어서 코딩

```
1 Button btn = new Button(this);
2 btn.setText("버튼입니다");
3 btn.setBackgroundColor(Color.MAGENTA);
4 baseLayout.addView(btn);
5
6 btn.setOnClickListener(new View.OnClickListener() {
7     public void onClick(View arg0) {
8         Toast.makeText(getApplicationContext(),
9             "코드로 생성한 버튼입니다", Toast.LENGTH_SHORT).show();
10    }
11 });
```

□ Java 코드로 화면 만들기[5/5]

▶ 직접 풀어보기 5-3

다음 화면을 XML 파일 없이 Java 코드만 이용하여 완성하라.

- 레이아웃에는 에디트텍스트 1개와 버튼 1개, 텍스트뷰 1개를 생성한다.
- 버튼을 클릭하면 에디트텍스트에 쓰인 문자열이 텍스트뷰에 나타나도록 한다.

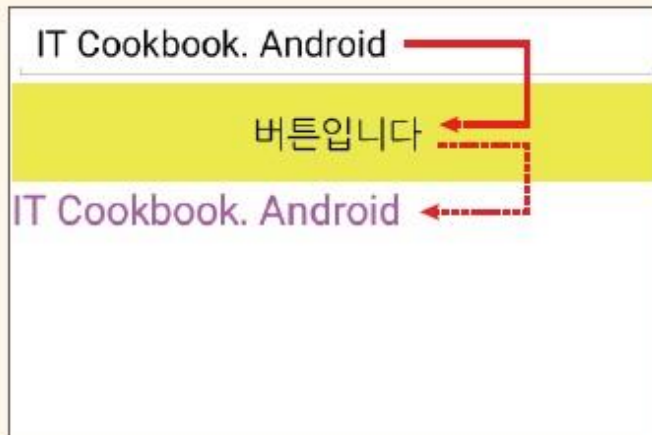


그림 5-8 XML 파일 없이 프로젝트 생성

3. 기타 레이아웃

□ 렐러티브레이아웃[1/6]

▣ 렐러티브레이아웃(상대레이아웃)

- 렐러티브레이아웃은 레이아웃 내부에 포함된 위젯들을 상대적인 위치로 배치

▣ 렐러티브레이아웃의 상하좌우에 배치

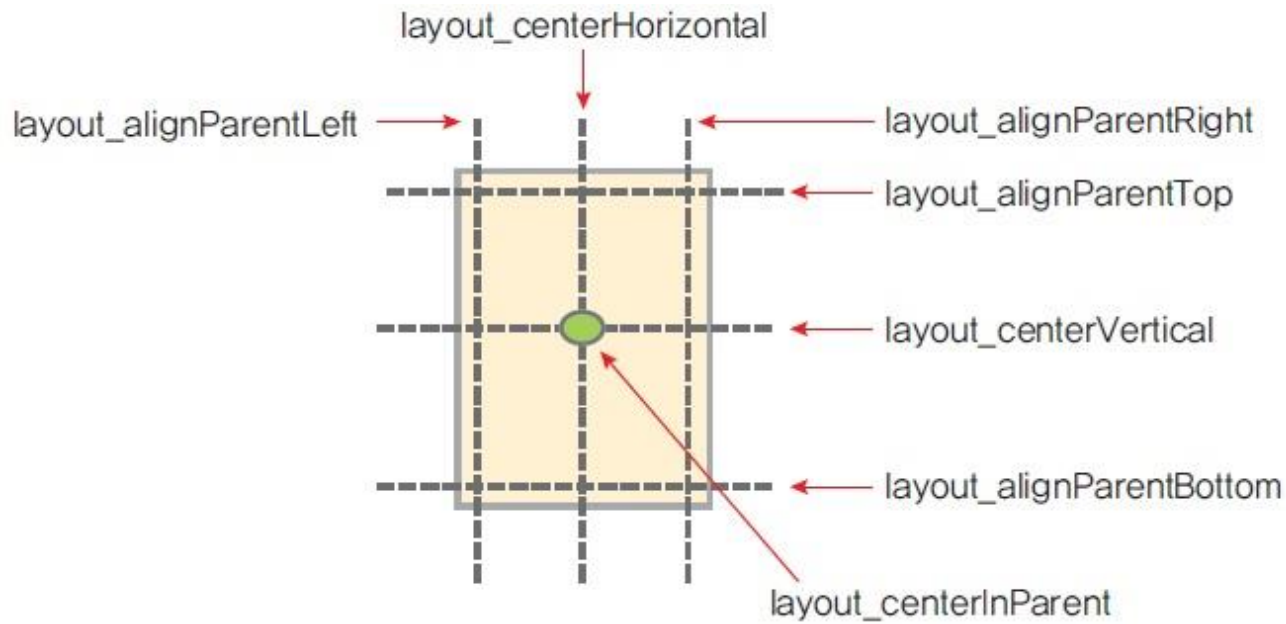


그림 5-9 부모(레이아웃)의 위치를 적용할 때 속성

□ 렐러티브레이아웃[2/6]- 예제

예제 5-10 렐러티브레이아웃 XML 코드 1

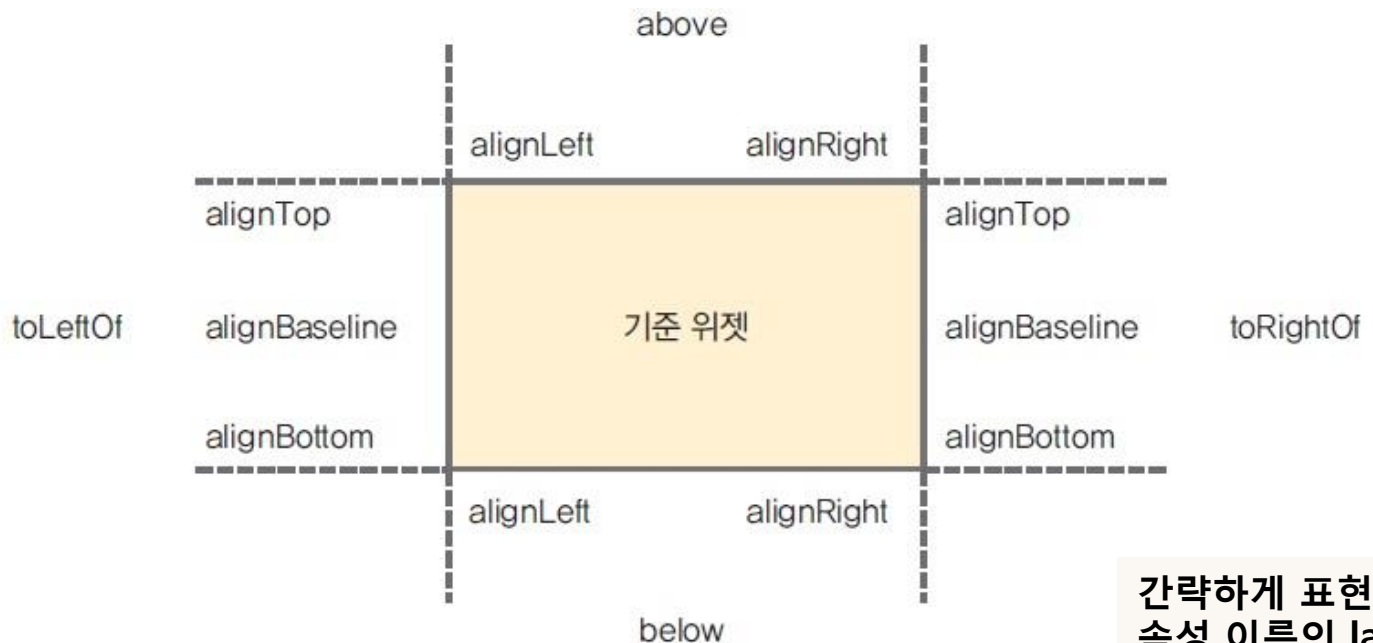
```
1 <RelativeLayout xmlns:android="http://~"
2     android:layout_width="match_parent"
3     android:layout_height="match_parent" >
4     <Button
5         android:layout_alignParentTop="true"
6         android:layout_centerHorizontal="true"
7         android:text="위쪽" />
8     <Button
9         android:layout_alignParentLeft="true"
10        android:layout_centerVertical="true"
11        android:text="좌측" />
12    <Button
13        android:layout_centerInParent="true"
14        android:text="중앙" />
15    <Button
16        android:layout_alignParentRight="true"
17        android:layout_centerVertical="true"
18        android:text="우측" />
19    <Button
20        android:layout_alignParentBottom="true"
21        android:layout_centerHorizontal="true"
22        android:text="아래" />
23 </RelativeLayout>
```



□ 렐러티브레이아웃[3/6]

▣ 다른 위젯의 상대 위치에 배치

- 각 속성의 값은 다른 위젯의 id를 지정하면 됨
- "@+id/기준 위젯의 아이디" 와 같은 형식으로 사용



간략하게 표현하기 위해
속성 이름의 layout_은 생략

그림 5-10 다른 위젯에서 상대적인 위치를 적용할 때의 속성

□ 렐러티브레이아웃[4/6]

▣ 렐러티브레이아웃 안에서 다른 위젯의 특정한 곳 배치 예제

```
1 <RelativeLayout xmlns:android="http://www."
2     android:layout_width="match_parent"
3     android:layout_height="match_parent" >
4     <Button
5         android:id="@+id/baseBtn"
6         android:layout_width="150dp"
7         android:layout_height="150dp"
8         android:layout_centerHorizontal="true"
9         android:layout_centerVertical="true"
10        android:text="기준 위젯" />
11    <Button
12        android:layout_alignTop="@+id/baseBtn"
13        android:layout_toLeftOf="@+id/baseBtn"
14        android:text="1번" />
15    ~~~~ 중간 생략 (버튼 2개) ~~~~
16    <Button
17        android:layout_above="@+id/baseBtn"
18        android:layout_alignLeft="@+id/baseBtn"
19        android:text="4번" />
20    <Button
21        android:layout_alignRight="@+id/baseBtn"
22        android:layout_below="@+id/baseBtn"
23        android:text="5번" />
24    <Button
25        android:layout_above="@+id/baseBtn"
26        android:layout_toRightOf="@+id/baseBtn"
27        android:text="6번" />
28 </RelativeLayout>
```



□ 렐러티브레이아웃[5/6]

예제 5-12 렐러티브레이아웃 속성을 조합한 XML 코드

```
1 <RelativeLayout xmlns:android="http://~"
2   android:layout_width="match_parent"
3   android:layout_height="match_parent" >
4   <Button
5     android:id="@+id/baseBtn1"
6     android:layout_alignParentLeft="true"
7     android:layout_alignParentTop="true"
8     android:text="기준1" />
9   <Button
10    android:id="@+id/baseBtn2"
11    android:layout_alignParentRight="true"
12    android:layout_centerVertical="true"
13    android:text="기준2" />
14   <Button
15     android:layout_above="@+id/baseBtn2"
16     android:layout_toRightOf="@+id/baseBtn1"
17     android:text="1번" />
18   <Button
19     android:layout_alignParentRight="true"
20     android:layout_below="@+id/baseBtn1"
21     android:text="2번" />
22 </RelativeLayout>
```



□ 렐러티브레이아웃[6/6]

▶ 직접 풀어보기 5-4

다음 화면의 XML 코드를 중복 리니어레이아웃과 렐러티브레이아웃으로 각각 작성해보자. 텍스트뷰 1개, 에디트텍스트 1개, 버튼 2개로 구성한다.

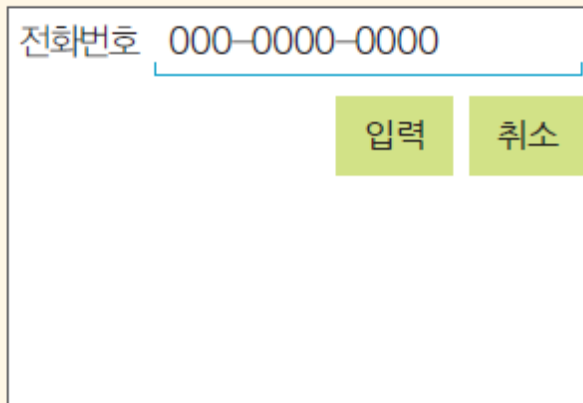


그림 5-11 레이아웃 비교

3. 기타 레이아웃 ▶ 테이블레이아웃[1/8]

□ 테이블레이아웃(TableLayout)

- ▣ 주로 위젯을 표 형태로 배치할 때 사용
- ▣ <TableRow>와 함께 사용되는데 <TableRow>의 개수가 바로 행의 개수가 됨
- ▣ 열의 개수는 <TableRow> 안에 포함된 위젯의 개수로 결정



그림 5-12 테이블레이아웃 개요

3. 기타 레이아웃 ▶ 테이블레이아웃[2/8]

□ 테이블레이아웃의 속성

- ▣ layout_column : 지정된 열에 현재 위젯을 표시하라는 의미
- ▣ stretchColumns : 지정된 열의 폭을 늘리라는 의미
- ▣ stretchColumns = "*" : 각 셀을 같은 크기로 확장, 전체 화면이 꽉 차는 효과

3. 기타 레이아웃 ▶ 테이블레이아웃[3/8]

예제 5-13 테이블레이아웃 XML 코드

```
1 <TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
2     >
3     <TableRow>
4         <Button
5             android:text="1" />
6         <Button
7             android:layout_span="2"
8             android:text="2" />
9         <Button
10            android:text="3" />
11     </TableRow>
12     <TableRow>
13         <Button
14             android:layout_column="1"
15             android:text="4" />
16         <Button
17             android:text="5" />
18         <Button
19             android:text="6" />
20     </TableRow>
21 </TableLayout>
```



3. 기타 레이아웃 ▶ 테이블레이아웃[4/8]

□ 실습

- 테이블레이아웃을 활용하여 숫자 버튼까지 있는 계산기를 만들어보자.
- 안드로이드 프로젝트 생성
 - 프로젝트 이름 : Project5_2
 - 패키지 이름 : com.cookandroid.project5_2
- 화면 디자인 및 편집
 - TableLayout 1개와 TableRow 9개로 구성
 - 에디트텍스트 2개, 숫자 버튼 10개, 연산 버튼 4개, 텍스트뷰 1개를 생성
 - 연산 버튼 위젯에는 layout_margin을 적절히 지정
 - 결과를 보여줄 TextView는 색상을 빨간색, 글자 크기는 20dp
 - 각 위젯의 id는 위에서부터 Edit1, Edit2, BtnNum0~9, BtnAdd, BtnSub, BtnMul, BtnDiv, TextResult



3. 기타 레이아웃 ▶ 테이블레이아웃[5/8]

□ 화면 디자인 및 편집

예제 5-14 activity_main.xml

```
1 <TableLayout xmlns:android="http://www."
2     android:layout_width="match_parent"
3     android:layout_height="match_parent" >
4
5     <TableRow>
6         <EditText
7             android:id="@+id/Edit1"
8             android:layout_span="5"
9             android:hint="숫자1 입력" />
10    </TableRow>
11
12    ~~~~ 중간 생략 (TableRow 1개, EditText 1개) ~~~~
13
14    <TableRow>
15        <Button
16            android:id="@+id/BtnNum0"
17            android:text="0" />
18        ~~~~ 중간 생략 (숫자 Button 4개) ~~~~
19    </TableRow>
20
21    <TableRow>
22        ~~~~ 중간 생략 (숫자 Button 5개) ~~~~
```

```
23 </TableRow>
24
25 <TableRow>
26     <Button
27         android:id="@+id/BtnAdd"
28         android:layout_margin="5dp"
29         android:layout_span="5"
30         android:text="더하기" />
31 </TableRow>
32
33 ~~~~ 중간 생략 (TableRow 3개, 연산 Button 3개) ~~~~
34
35 <TableRow>
36     <TextView
37         android:id="@+id/TextResult"
38         android:layout_margin="5dp"
39         android:layout_span="5"
40         android:text="계산 결과 : "
41         android:textColor="#FF0000"
42         android:textSize="20dp" />
43 </TableRow>
44 </TableLayout>
```


3. 기타 레이아웃 ► 테이블레이아웃[6/8]

□ Java 코드 작성 및 수정

▣ 전역변수 선언

- 숫자 버튼을 제외한 activity_main.xml의
- 7개 위젯에 대응할 위젯변수 7개
- 입력될 2개 문자열 저장할 문자열 변수 2개
- 계산 결과를 저장할 정수 변수 1개
- 10개 숫자 버튼을 저장할 버튼 배열
- 10개 버튼의 id를 저장할 정수형 배열
- 증가 값으로 사용할 정수 변수

예제 5-15 Java 코드 1

```
1  ~~~~ 중간 생략 ~~~~
2  public class MainActivity extends Activity {
3      EditText edit1, edit2;
4      Button btnAdd, btnSub, btnMul, btnDiv;
5      TextView textResult;
6      String num1, num2;
7      Integer result;
8      Button[] numButtons = new Button[10];
9      Integer[] numBtnIDs = { R.id.BtnNum0, R.id.BtnNum1, R.id.BtnNum2, R.id.BtnNum3,
10         R.id.BtnNum4, R.id.BtnNum5, R.id.BtnNum6, R.id.BtnNum7,
11         R.id.BtnNum8, R.id.BtnNum9};
12     int i;
13
14
15     @Override
16     public void onCreate(Bundle savedInstanceState) {
17     ~~~~ 중간 생략 ~~~~
```

3. 기타 레이아웃 ► 테이블레이아웃[7/8]

□ Java 코드 작성 및 수정

- 숫자 버튼이 없다고 가정하고 연산 버튼을 터치했을 때의 내용을 코딩

```
1  ~~~~ 중간 생략 ~~~~
2  public void onCreate(Bundle savedInstanceState) {
3      super.onCreate(savedInstanceState);
4      setContentView(R.layout.activity_main);
5
6      setTitle("테이블레이아웃 계산기");
7
8      edit1 = (EditText) findViewById(R.id.Edit1);
9      edit2 = (EditText) findViewById(R.id.Edit2);
10     btnAdd = (Button) findViewById(R.id.BtnAdd);
11     ~~~~ 중간 생략 (연산 버튼 3개 대입) ~~~~
12     textResult = (TextView) findViewById(R.id.TextResult);
13
14     btnAdd.setOnTouchListener(new View.OnTouchListener() {
15         public boolean onTouch(View arg0, MotionEvent arg1) {
16             num1 = edit1.getText().toString();
17             num2 = edit2.getText().toString();
18             result = Integer.parseInt(num1) + Integer.parseInt(num2);
19             textResult.setText("계산 결과 : " + result.toString());
20             return false;
21         }
22     });
23     ~~~~ 중간 생략 (연산 버튼 3개 터치 이벤트 리스너) ~~~~
24 }
```

3. 기타 레이아웃 ► 테이블레이아웃[8/8]

□ Java 코드 작성 및 수정

- 숫자 버튼 10개를 배열 변수에 대입한 후에 각 버튼의 클릭 이벤트 리스너를 만듦

```
1  for (i = 0; i < numBtnIDs.length; i++) {
2      numButtons[i] = (Button) findViewById(numBtnIDs[i]);
3  }
4
5  for (i = 0; i < numBtnIDs.length; i++) {
6      final int index; // 주의! 꼭 필요함..
7      index = i;
8
9      numButtons[index].setOnClickListener(new View.OnClickListener() {
10         public void onClick(View v) {
11
12             if (edit1.isFocused() == true) {
13                 num1 = edit1.getText().toString()
14                     + numButtons[index].getText().toString();
15                 edit1.setText(num1);
16             } else if (edit2.isFocused() == true) {
17                 num2 = edit2.getText().toString()
18                     + numButtons[index].getText().toString();
19                 edit2.setText(num2);
20             } else {
21                 Toast.makeText(getApplicationContext(),
22                     "먼저 에디트텍스트를 선택하세요", Toast.LENGTH_SHORT).show();
23             }
24         }
25     });
26
27 }
```

3. 기타 레이아웃 ▶ 그리드레이아웃[1/4]

□ 그리드레이아웃(GridLayout)

- 테이블레이아웃처럼 위젯을 표 형태로 배치할 때 사용하는데 좀 더 직관적임
- Android 4.0(아이스크림 샌드위치, API 14)부터 지원

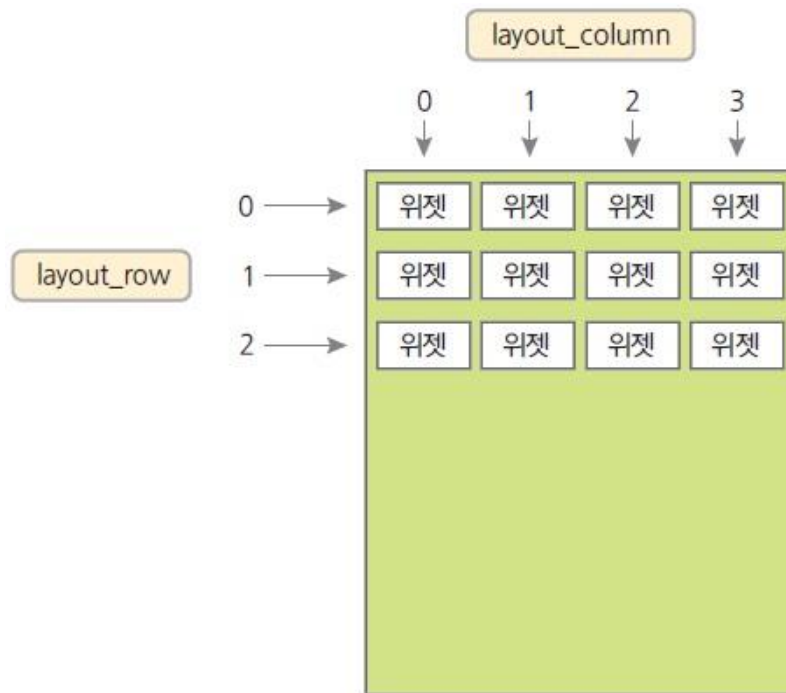


그림 5-15 그리드레이아웃 개요

3. 기타 레이아웃 ▶ 그리드레이아웃[2/4]

□ 그리드레이아웃 속성

■ <GridLayout> 자체에 자주 사용되는 속성

- rowCount : 행 개수
- columnCount : 열 개수
- orientation : 그리드를 수평 방향을 우선할지, 수직 방향을 우선할지를 결정

■ 그리드레이아웃 안에 포함될 위젯에서 자주 사용되는 속성

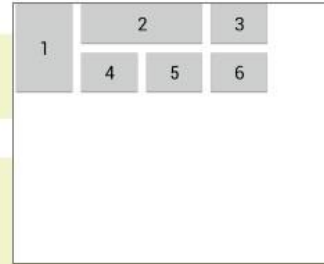
- layout_row : 자신이 위치할 행 번호(0번부터 시작)
- layout_column : 자신이 위치할 열 번호(0번부터 시작)
- layout_rowSpan : 행을 지정된 개수만큼 확장
- layout_columnSpan : 열을 지정된 개수만큼 확장
- layout_gravity : 주로 fill, fill_vertical, fill_horizontal 등으로 지정, 행 또는 열 확장시, 위젯을 확장된 셀에 꽉 채우는 효과를 냄

. 기타 레이아웃 ▶ 그리드레이아웃[3/4]

□ 그리드레이아웃 예제

예제 5-18 그리드레이아웃 XML 코드

```
1 <GridLayout xmlns:android="http://~"
2     android:columnCount="4"
3     android:rowCount="2" >
4     <Button
5         android:layout_column="0"
6         android:layout_row="0"
7         android:layout_rowSpan="2"
8         android:layout_gravity="fill_vertical"
9         android:text="1" />
10    <Button
11        android:layout_column="1"
12        android:layout_row="0"
13        android:layout_columnSpan="2"
14        android:layout_gravity="fill_horizontal"
15        android:text="2" />
16    <Button
17        android:layout_column="3"
18        android:layout_row="0"
19        android:text="3" />
20    <Button
21        android:layout_column="1"
22        android:layout_row="1"
23        android:text="4" />
24    ~~~~ 중간 생략 ~~~~
25 </GridLayout>
```



3. 기타 레이아웃 ▶ 그리드레이아웃[4/4]

▶ 직접 풀어보기 5-5

[실습 5-2]를 그리드레이아웃으로 변경해서 실행해보자.

HINT Java 코드는 변경할 필요가 없고, XML만 변경하면 된다. XML 위젯의 id도 동일하게 사용한다.



그림 5-16 그리드레이아웃 계산기

3. 기타 레이아웃 ▶ 프레임레이아웃[1/3]

□ 프레임레이아웃(FrameLayout)

- 단순히 레이아웃 내의 위젯들은 왼쪽 상단부터 겹쳐서 출력
- 프레임레이아웃 자체로 사용하기보다는 탭 위젯 등과 혼용해서 사용할 때 유용



그림 5-17 프레임레이아웃 개요

3. 기타 레이아웃 ▶ 프레임레이아웃[2/3]

□ 프레임레이아웃의 속성

- ▣ foreground : 프레임레이아웃의 전경 이미지를 지정
- ▣ foregroundGravity : 전경 이미지의 위치를 지정

3. 기타 레이아웃 ▶ 프레임레이아웃[3/3]

□ 프레임레이아웃 예제

예제 5-19 프레임레이아웃 XML 코드

```
1 <FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
2     android:layout_width="match_parent"
3     android:layout_height="match_parent"
4     android:foreground="@drawable/dog"
5     android:foregroundGravity="fill_horizontal" >
6     <RatingBar
7         android:id="@+id/ratingBar1" >
8     <ImageView
9         android:src="@drawable/ic_launcher" />
10    <CheckBox
11        android:text="CheckBox" />
12 </FrameLayout>
```

