

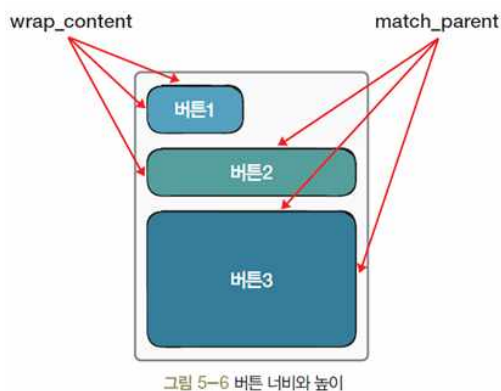
3 레이아웃

[1]. 레이아웃(Layout)

- ▶ 뷰 그룹으로부터 파생되는 클래스들로서 다른 뷰들을 차일드로 포함하는 뷰의 컨테이너
- ▶ 버튼, 에디트, 텍스트 같은 기본 위젯들이 일정한 규칙에 따라 모여서 하나의 레이아웃을 구성하고 레이아웃 여러 개가 모여 하나의 액티비티 화면을 완성함

(1) 너비와 높이 : layout_width, layout_height

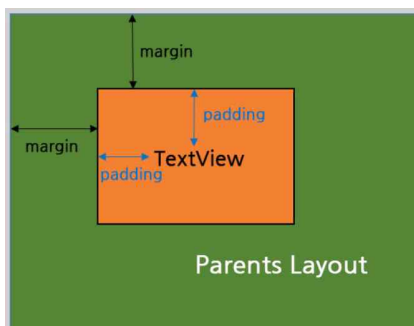
너비	android:layout_width="match_parent"	부모 영역의 너비만큼 채움
	android:layout_width="wrap_content"	내용물의 너비만큼 채움
높이	android:layout_height="match_parent"	부모 영역의 높이만큼 채움
	android:layout_height="wrap_content"	내용물의 높이만큼 채움



- ▶ layout_width나 layout_height 속성에는 크기를 바로 지정할 수 있음 (ex. layout_width = 100dp)
- ▶ dp : 'Density-independent Pixel'의 약자이며, 다양한 해상도에 맞게 크기를 자동으로 조절해주는 역할을 함

(2) 여백 : padding, layout_margin

padding	뷰 내부 여백 설정	paddingLeft, paddingTop, paddingRight, paddingBottom
layout_margin	뷰 바깥 여백 설정	layout_marginLeft, layout_marginTop, layout_marginRight, layout_marginBottom



(3) 정렬 : layout_gravity, gravity

layout_gravity	부모 컨테이너의 여유 공간 안에서 뷰의 정렬 방향을 지정하고자 할 때 사용	left, right, center, top, bottom, center_vertical, center_horizontal
gravity	객체 내에서 Content의 위치를 설정	left, right, center, top, bottom, center_vertical, center_horizontal

▶ 두개를 동시에 적용하기 위해서는 | (OR) 연산자 사용 (ex. bottom|right)

(4) 색깔 : background, backgroundTint, textColor 등

background	레이아웃에 색깔 지정
backgroundTint	텍스트뷰나 버튼에 색깔 지정
textColor	글자 색깔 지정

(5) 단축키

기능	단축키
기본 코드 완성	Ctrl + Space
자동 줄 맞춤	Ctrl + Alt + I (엘)
한줄 주석 달기/해제	Ctrl + /
여러줄 주석 달기/해제	Ctrl + Shift + /
스마트 코드 완성	Ctrl + Shift + Space
저장	Ctrl + S
실행	Shift + F10

[2]. 레이아웃의 종류

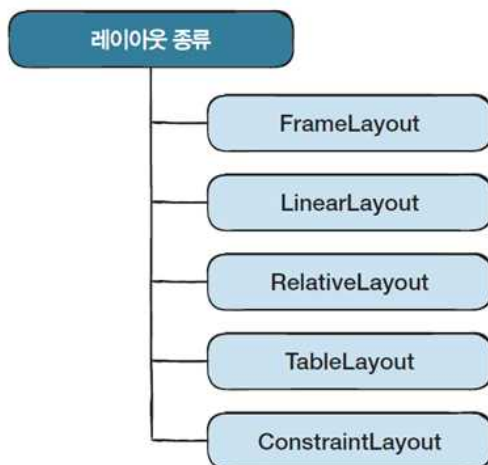


그림 6-1 안드로이드 레이아웃 종류

레이아웃 종류	설명
FrameLayout	좌측 상단을 기준으로 위젯을 차곡차곡 쌓기만 함
LinearLayout	선형. 위젯을 위에서 아래 방향으로 배치하거나, 왼쪽에서 오른쪽 방향으로 배치함
RelativeLayout	상대적. 배치된 위젯을 기반으로 상대적인 위치에 다른 위젯을 배치함
TableLayout	테이블(표) 형태로 위젯을 배치함
ConstraintLayout	[Design] 탭에서 마우스로 위젯을 배치하는 것에 용이함

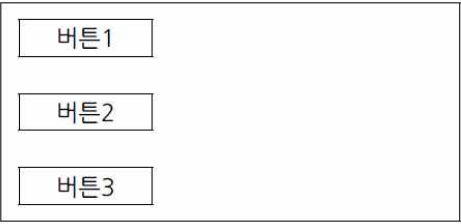
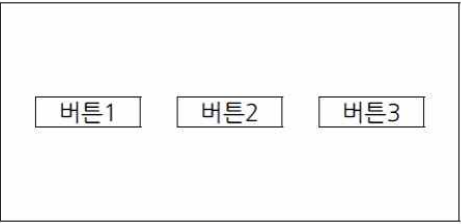
(1) LinearLayout (리니어, 선형 레이아웃)

- ▶ 위젯들을 일렬로 배치하는 레이아웃으로 가장 많이 사용하는 레이아웃 중 하나
- ▶ 위젯을 위에서 아래 방향(수직)으로 배치하거나, 왼쪽에서 오른쪽 방향(수평)으로 배치

▶ 속성

속성	설명
orientation	레이아웃 방향 지정, vertical(세로), horizontal(가로)
layout_gravity	여유 공간 안에서 뷰의 정렬 방향 지정
gravity	객체 내에서 content 정렬 방향 지정
padding	뷰 내부의 여백 설정
layout_margin	뷰 바깥의 여백 설정
layout_weight	가중치(비율) 지정

① orientation : 레이아웃 방향 지정

orientation = "vertical"	orientation = "horizontal"
 <p>버튼1 버튼2 버튼3</p> <p>< ① 수직(vertical) ></p>	 <p>버튼1 버튼2 버튼3</p> <p>< ② 수평(horizontal) ></p>

orientation	vertical (세로 배치)	horizontal (가로 배치)
가로 정렬 (left, center_horizontal, right)	Button에서 layout_gravity	Layout에서 gravity
세로 정렬 (top, center_vertical, bottom)	Layout에서 gravity	Button에서 layout_gravity

② layout_weight : 가중치 (가로, 세로 비율)

```

<?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="horizontal"
    android:weightSum="3">

    <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:text="버튼" />

    <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="버튼" />

</LinearLayout>

```

→ 너비는 **layout_weight** 속성에 따라 달라지므로 위젯의 너비는 **0dp**로 설정

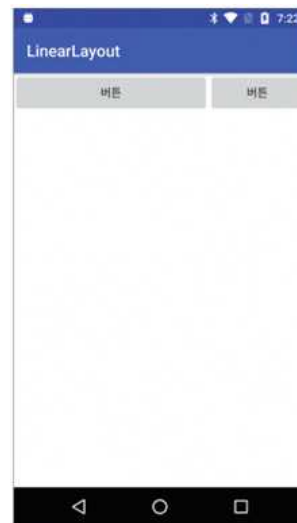
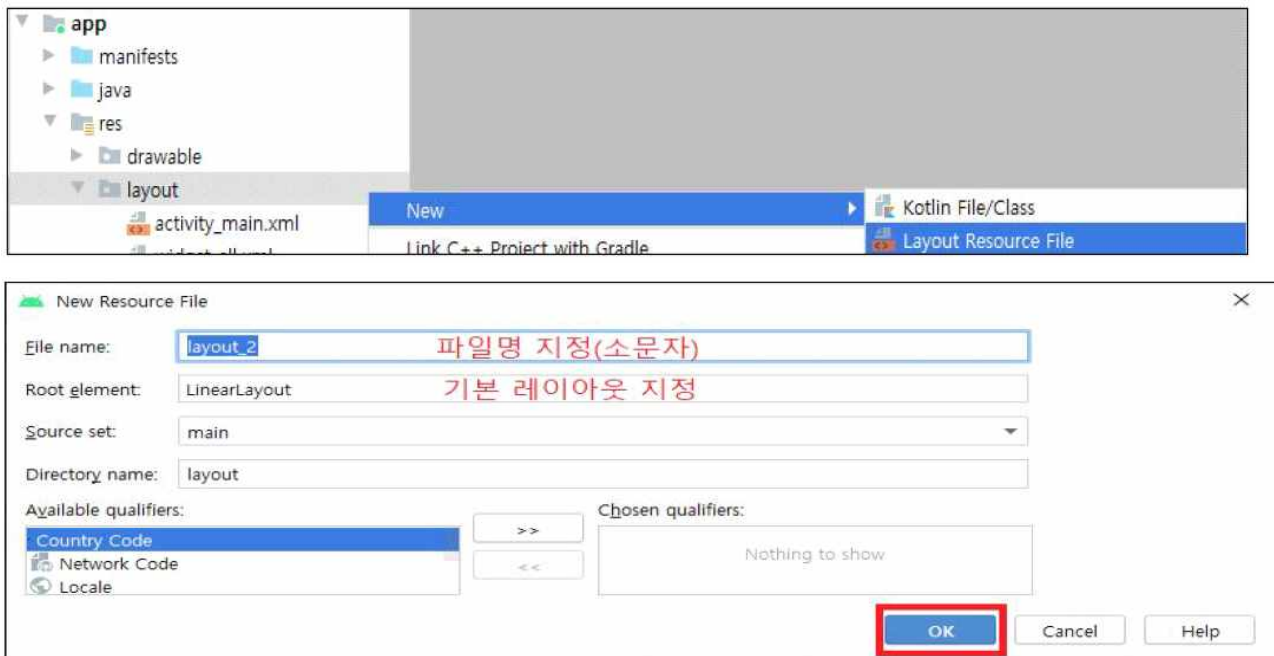


그림 6-6 수평 리니어 레이아웃 가중치

[새로운 레이아웃 추가 방법]



여기까지만 작업해도 새로운 파일에 레이아웃 디자인 작업을 할 수 있다.


만약 App을 실행하여 확인하고 싶다면 “MainActivity.java” 파일에서 다음의 내용을 수정한다.


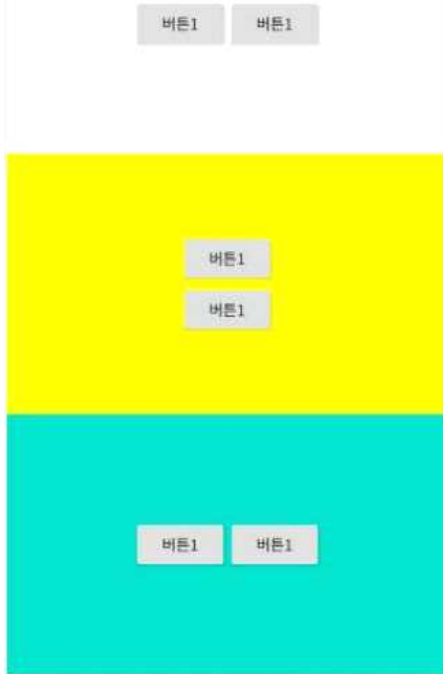
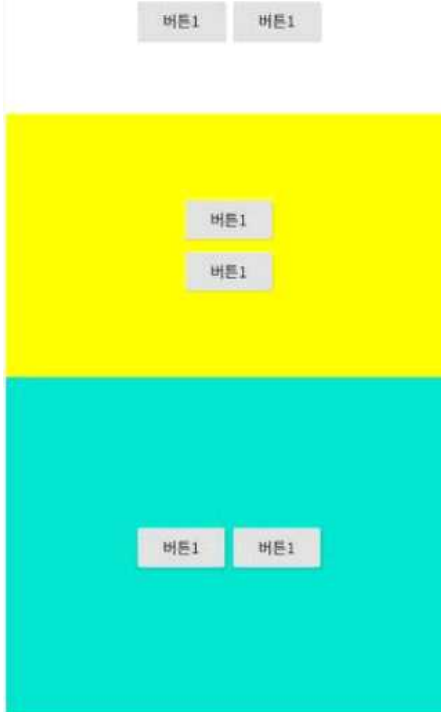
(Tip. 혹시 추가된 레이아웃이 뜨지 않는다면 프로젝트를 닫았다 다시 열면 보임)

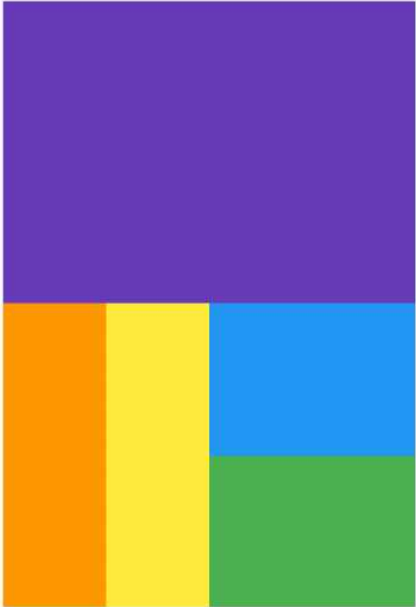
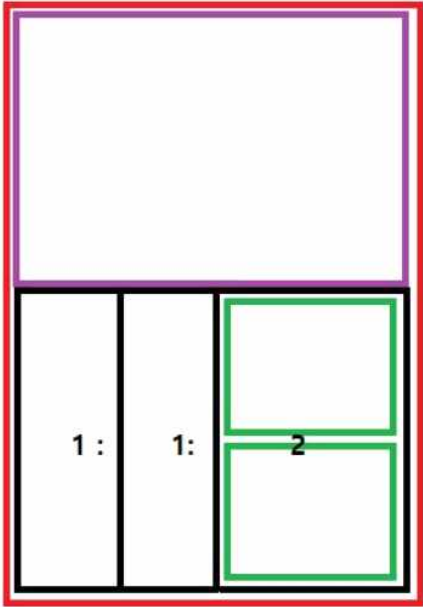
```
public class MainActivity extends AppCompatActivity {

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

기본 레이아웃인 activity_main 대신
새 레이아웃 이름으로 수정하면 됨

수행 과제	linear_1.xml 레이아웃 파일을 추가하여 3개의 버튼을 추가해보자
수행 방법	<p>LinearLayout에 Button 세 개를 추가하고, orientation을 설정해보자</p> <p>☞(Tip 1) 안드로이드 스튜디오의 장점 중 하나는 “자동 채우기” 기능이 제공된다는 것이다. “L” 까지만 입력하고 “Ctrl + space” 키를 누르면 L로 시작하는 코드가 추천된다. “Ctrl + Space”를 꼭 기억하자.</p> <p>☞(Tip 2) xml파일은 각 태그마다 “<” 로 시작하여 “/>” 코드로 끝맺는다. 태그 내부에 다른 태그를 내포할 때는 <태그이름> ...내용... </태그이름>의 형태를 갖는다.</p>
<pre> 1 <?xml version="1.0" encoding="utf-8"?> 2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android" 3 android:layout_width="match_parent" 4 android:layout_height="match_parent" 5 android:orientation="vertical" > 6 7 <Button 8 android:layout_width="100dp" 9 android:layout_height="100dp" 10 android:text="버튼1" /> 11 <Button 12 android:layout_width="100dp" 13 android:layout_height="100dp" 14 android:text="버튼2" /> 15 <Button 16 android:layout_width="100dp" 17 android:layout_height="100dp" 18 android:text="버튼3" /> 19 20 </LinearLayout> </pre> 	
<p>1) “버튼4”라는 버튼을 “버튼3” 아래에 더 추가해보세요.</p> <p>2) 현재 모든 버튼은 가로, 세로 100dp 사이즈입니다. dp가 무엇인지 인터넷을 검색해보세요.</p> <p>3) “버튼3” 버튼의 세로 사이즈를 200dp로 바꾸어봅시다.</p> <p>4) "orientation" 속성을 이용하여 버튼이 가로 방향으로 배치되도록 수정해보세요.</p> <p>* 나머지 버튼도 가로, 세로 사이즈와 orientation을 바꾸면서 자유롭게 바꾸어보면서, 속성 변화에 대해 확인해보세요.</p>	

수행 과제	linear_2_1.xml, linear_2_2.xml, linear_2_3.xml 레이아웃 파일을 추가하여 중복된 레이아웃을 표현해보자.	
수행 방법	안드로이드 화면을 구성하다 보면 하나의 화면 안에서 수평, 수직을 다양하게 배치해야 할 필요가 있다. 이런 경우 레이아웃 내부에 레이아웃을 다시 생성하는 방법을 이용하여 원하는 모양을 구성할 수 있다.	
<p>◆ Quiz1) linear_2_1.xml 파일에 그림과 같은 형태의 레이아웃을 작성해 보시오. 각 영역의 바탕색을 자유롭게 지정하여 영역이 표시되도록 하시오.</p> <p>◆ Quiz2) linear_2_2.xml 파일에 그림과 같은 형태의 레이아웃을 작성해 보시오. 각 영역은 1:1:1의 비율을 가지며 부모 영역을 채우는 레이아웃으로 작성하시오.</p> <p>◆ Quiz3) linear_2_3.xml 파일에 그림과 같은 형태의 레이아웃을 작성해 보시오. 각 영역은 1:2:3의 비율을 가지며 부모 영역을 채우는 레이아웃으로 작성하시오.</p>		
Quiz1 완성	Quiz2 완성	Quiz3 완성
		

수행 과제	linear_3.xml 레이아웃 파일을 추가하고, LinearLayout을 이용하여 중복된 레이아웃을 표현해보자.	
수행 방법	레이아웃 내부에 또 다른 레이아웃이 내포된 상태이다. LinearLayout의 orientation , layout_weight, background 속성을 이용한다.	
완성 디자인		레이아웃 구조
		

4 위젯(Widget)

[1]. 위젯소개 및 종류

- ▶ 위젯(widget) : 안드로이드에서 화면을 구성하는데 사용하는 요소
- ▶ 위젯 상속 구조

① 뷰(View)

View 클래스를 상속해서 만들어진 위젯

② 뷰그룹(ViewGroup)

ViewGroup 클래스를 상속한 위젯

③ 레이아웃(Layout)

뷰그룹에 속한 위젯 중에서 뷰를 배치하는 용도로 사용하는 배치 관리자

④ 뷰 컨테이너(View Container)

뷰그룹의 위젯 중에서 레이아웃을 제외한 나머지 위젯

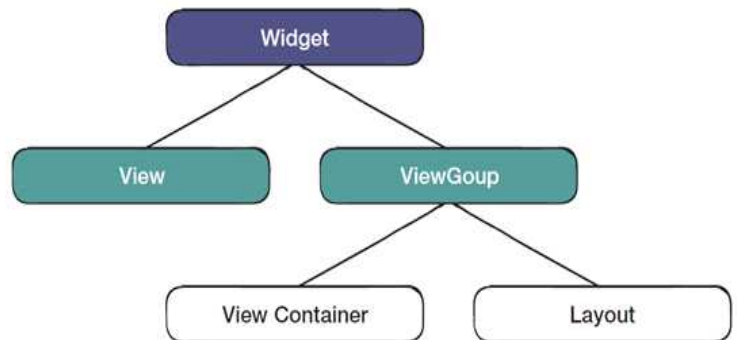


그림 5-1 안드로이드 위젯

- ▶ [Design]탭 - [Pallete]창

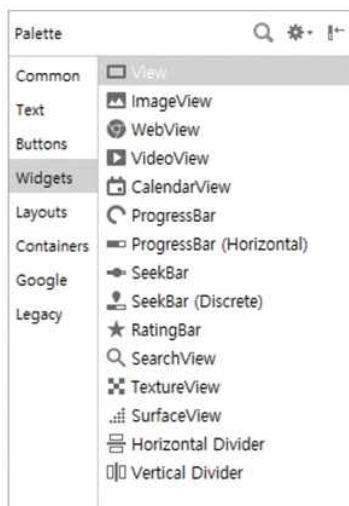


그림 5-2 [Pallete] 창

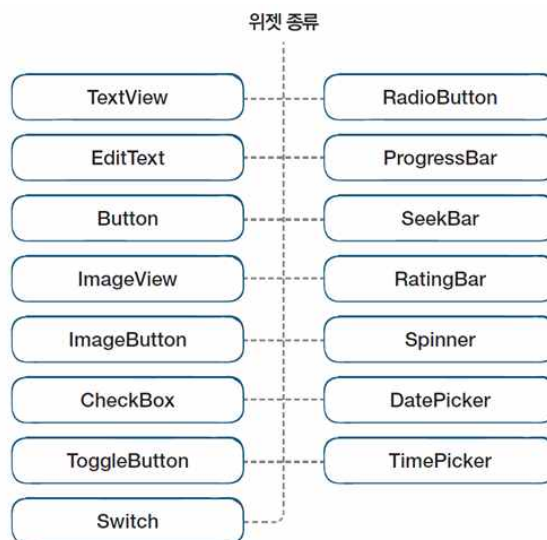


그림 5-3 위젯의 종류

[2]. Button (버튼)

- ▶ 사용자의 클릭 입력을 받아 클릭 이벤트 처리함

(1) 클래스 계층구조

```
java.lang.Object
└─ android.view.View
   └─ android.widget.TextView
      └─ android.widget.Button
```

(2) 주요 속성

속성	설명	
android:id="@+id/아이디"	버튼 아이디 지정	
android:layout_width	버튼 너비	
android:layout_height	버튼 높이	
android:backgroundTint	배경색(그림자 표현)	
android:textAllCaps	버튼의 영어 텍스트를 대문자로 지정	
	true	대문자로 보임(기본)
	false	개발자가 지정한 텍스트 그대로 보임
android:text	버튼에 보이는 텍스트 설정	

[3]. TextView (텍스트뷰)

- ▶ 화면에서 텍스트를 출력하는 가장 기본적인 위젯
- ▶ 단순히 화면에 문자를 보여 주기만 할 뿐이며, 사용자로부터 입력을 받지 못함
⇒ 고정된 텍스트를 보여 주거나 다른 위젯의 제목을 표시할 때 주로 사용

(1) 클래스 계층 구조




```
java.lang.Object
└─ android.view.View
   └─ android.widget.TextView
```

(2) View 클래스 공통 속성

속성	설명						
android:alpha	투명도를 나타내는 것으로 0~1 사이의 값을 가진다. 0일 경우 완전 투명, 1인 경우 완전 불투명						
android:background	배경색 또는 배경이미지						
android:id	해당 뷰에 ID를 지정 android:id = "@+id/textView1"						
android:visibility	<table border="1"> <tr> <td>visible</td><td>화면에 보임</td></tr> <tr> <td>invisible</td><td>화면에 숨기고 영역은 그대로 유지</td></tr> <tr> <td>gone</td><td>화면에서 숨기고 영역도 차지하지 않음</td></tr> </table> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>리니어 레이아웃(수평)</p> <p>리니어 레이아웃(수평)</p> </div> <div style="text-align: center;"> <p>리니어 레이아웃(수평)</p> <p>리니어 레이아웃(수평)</p> </div> </div> <p style="text-align: center;">그림 5-7 리니어 레이아웃에 버튼 visibility 속성</p>	visible	화면에 보임	invisible	화면에 숨기고 영역은 그대로 유지	gone	화면에서 숨기고 영역도 차지하지 않음
visible	화면에 보임						
invisible	화면에 숨기고 영역은 그대로 유지						
gone	화면에서 숨기고 영역도 차지하지 않음						

(3) TextView 주요 속성

속성	설명										
android:autoLink	<p>자동으로 링크를 걸어주는 속성으로 전화번호나 이메일 주소를 설정해 놓으면 사용자가 클릭했을 때 자동으로 전화 앱이나 이메일 앱 실행</p> <table border="1"> <tr> <td>phone</td><td>전화번호에 링크 걸어줌</td></tr> <tr> <td>email</td><td>이메일 주소에 링크 걸어줌</td></tr> <tr> <td>web</td><td>웹페이지 주소에 링크 걸어줌</td></tr> <tr> <td>all</td><td>자동으로 링크를 모두 걸고 싶을 때 사용</td></tr> <tr> <td>none</td><td>링크를 걸지 않음</td></tr> </table>	phone	전화번호에 링크 걸어줌	email	이메일 주소에 링크 걸어줌	web	웹페이지 주소에 링크 걸어줌	all	자동으로 링크를 모두 걸고 싶을 때 사용	none	링크를 걸지 않음
phone	전화번호에 링크 걸어줌										
email	이메일 주소에 링크 걸어줌										
web	웹페이지 주소에 링크 걸어줌										
all	자동으로 링크를 모두 걸고 싶을 때 사용										
none	링크를 걸지 않음										
android:lines	TextView가 여러 줄 입력이 가능하도록 라인 수 지정										
android:text	화면에 텍스트 표시										
android:textColor	텍스트 색상 지정										
android:textSize	글자의 크기 지정, 단위는 주로 sp 사용										
android:textStyle	글자의 스타일 지정 굵게(bold), 기울임(italic), 굵게기울임(bold italic)										
android:typeface	글꼴 지정 (normal, sans, serif, monospace)										

수행 과제	school_info.xml 레이아웃 파일을 추가하고, LinearLayout 과 TextView 위젯을 이용하여 다음 화면을 디자인해보자.
수행 방법	조건에 맞게 작성하되, 조건에 없는 부분은 보이는 대로 작성한다.
완성 디자인	조건
	<ul style="list-style-type: none"> - 전체 레이아웃 : 안쪽 여백 20dp ① 바깥아래 여백 30dp, 글자크기 40dp, 글자 색상 ② 글자크기 20dp, 굵게 지정, 글자 색상 ③ 글자크기 16dp, 글자 색상 ④ 바깥위 여백 20dp ⑤ 전화번호 클릭하면 전화 앱 실행 ⑥ 바깥위 여백 20dp ⑦ 이메일 주소 클릭하면 이메일 앱 실행 ⑧ 레이아웃 : 바깥위 여백 20dp 이전 버튼 : 색상 다음 버튼 : 색상, visibility 지정 종료 버튼 : 색상
<p>1) 각종 텍스트의 크기와, 색상, 위치를 바꾸어 보자.</p> <p>2) “다음” 버튼의 visible 속성을 변경하여 아래와 같은 형태로 만드시오.</p>  <p>3) “다음” 버튼의 visible 속성을 변경하여 아래와 같은 형태로 만드시오.</p> 	

[4]. EditText (에디트텍스트)

- ▶ 텍스트뷰(TextView)가 사용자에게 텍스트를 보여 줄 수 있는 위젯이라면, 에디트텍스트(EditText)는 사용자에게 텍스트를 입력 받을 수 있는 위젯
- ▶ 에디트 텍스트는 글자 입력을 위한 다양한 속성을 제공하며, 이를 이용해 입력 내용을 제한할 수 있음

(1) 클래스 계층구조

```
java.lang.Object
└─ android.view.View
    └─ android.widget.TextView
        └─ android.widget.EditText
```


(2) 주요 속성

- ▶ 에디트텍스트는 텍스트뷰를 상속받아 만든 위젯이기 때문에 텍스트뷰의 속성도 사용 가능

속성	값	설명
inputType	"phone"	전화번호 형식
	"number"	숫자 형식
	"textPasswd"	비밀번호형식
	"date"	날짜 형식 입력에 편리
hint	텍스트	힌트 메시지 출력

- ▶ 사용자가 에디트 텍스트를 길게 누르면 복사, 잘라내기, 붙여넣기와 같은 메뉴를 제공한다.
이 기능은 운영 체제가 기본 제공하는 것으로 따로 코드를 작성할 필요가 없다.



수행 과제	지난 시간에 만들었던 school_info 레이아웃에 EditText 위젯을 추가하여 다음의 화면을 구성해보자.
완성 디자인	조건
	<p>① ID 지정 : editText1</p> <p>② ID 지정 : editText2 입력형식 : 비밀번호</p>
<p>1) 위에서 작성한 아이디, 비밀번호 아래에 “학번”을 입력하는 칸을 추가해보자.</p> <p>2) “학번”은 숫자로만 이루어져 있고 최대길이는 5자이다. 사용자가 5글자까지만 입력할 수 있도록 하기 위해 어떤 속성을 이용하면 좋을지 찾아보자.</p>	

5 이벤트 처리와 인텐트

[1]. 버튼 이벤트

1) 화면 초기화 : onCreate()

액티비티가 생성될 때 호출되며, 화면이 초기화된다.

자동 생성되는 코드의 형태는 다음과 같다.

```
@Override
protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //setContentView가 실행된 뒤에, 다음 작업을 시작해야한다.
}
```

2) 자바와 XML의 Button 연결 : findViewById() ;

```
Button button1;
button1 = findViewById(R.id.XXX);
```

XXX 부분에는 XML에서 지정한 id를 입력한다.

해당 아이디를 가진 위젯을 찾아서 자바에서 사용할 수 있는 객체로 반환해준다.

3) 버튼 클릭 이벤트 감지자 등록 : button1.setOnClickListener(new View.OnClickListener(){})

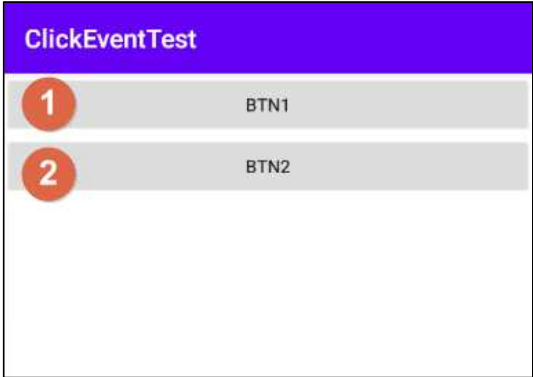
button1을 클릭하면 호출될 콜백 리스너 등록한다.

버튼 클릭이 감지되면 등록되었던 OnClickListener가 실행되고, 그 내부에서 우리가 원하는 동작을 코딩할 수 있다.

자동 생성되는 코드의 형태는 다음과 같다.

```
//button1의 클릭을 감지하기 위하여 클릭 리스너를 지정한다.
button1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // 여기에 원하는 동작을 입력합니다.
    }
});
```

4) 버튼 클릭되었을 때 하고자 하는 작업 : onClick()

수행 과제	ClickEventTest 프로젝트를 만들고 버튼 이벤트를 작성해보자.	
수행 방법	<ul style="list-style-type: none"> - activity_main.xml 에 버튼 두개를 추가해보자. - MainActivity.java 에 버튼을 클릭했을 때 Toast 메시지를 보이도록 작성해보자. 	
activity_main.xml		조건
		<ul style="list-style-type: none"> ① id지정 : btn1 버튼을 클릭했을 때 "BTN1 !!" 메시지 출력 ② id지정 : btn2 버튼을 클릭했을 때 "BTN2 !!" 메시지 출력
MainActivity.java		방법1, 방법2
<pre> 10 public class MainActivity extends AppCompatActivity { 11 12 Button btn1, btn2; // 버튼 변수(객체) 선언 13 14 @Override 15 protected void onCreate(Bundle savedInstanceState) { 16 super.onCreate(savedInstanceState); 17 setContentView(R.layout.activity_main); 18 19 // 버튼 연결 20 btn1 = findViewById(R.id.btn1); 21 btn2 = findViewById(R.id.btn2); 22 23 // 버튼에 클릭 이벤트 감지자 등록 24 btn1.setOnClickListener(click); // 방법 1. 인터페이스를 객체화시켜서 따로 만들기 25 btn2.setOnClickListener(new View.OnClickListener() { // 방법 2. 익명 내부 클래스로 한꺼번에 만들기 26 @Override 27 public void onClick(View v) { 28 // 버튼의 이벤트가 감지되었을 때 호출되는 메서드 29 // 버튼이 클릭되었을 때 하고자하는 작업을 이곳에서 한다. 30 Toast.makeText(getApplicationContext(), text: "BTN2 !!", Toast.LENGTH_SHORT).show(); 31 } 32 }); 33 34 } // onCreate() 35 36 View.OnClickListener click = new View.OnClickListener() { // 방법 1. 인터페이스를 객체화시켜서 따로 만들기 37 @Override 38 public void onClick(View v) { 39 Toast.makeText(getApplicationContext(), text: "BTN1 !!", Toast.LENGTH_SHORT).show(); 40 } 41 }; 42 </pre>		

[2]. EditText 값 처리하기

1) EditText 변수(객체) 선언

```
EditText edit_user ;
```

2) 자바와 XML의 EditText 연결 : findViewById() ;


```
edit_user = findViewById(R.id.userid) ;
```

3) EditText에 입력된 문자를 읽어서 String으로 저장

```
String str = edit_user.getText().toString() ;
```

EditText형의 edit_user에서 getText() 하게 되면 리턴되는 타입은 문자열이 아니고 Editable 이라는 형태가 반환됨 ⇒ String으로 다시 한번 변환하기 위해 toString() 메소드를 사용

수행 과제	ClickEventTest 프로젝트에서 EditText에 입력된 값을 처리해보자.
수행 방법	<ul style="list-style-type: none"> - activity_main.xml 에 EditText를 추가해보자. - MainActivity.java 에 EditText에 입력된 값에 따라 다른 Toast를 보이도록 작성해보자.

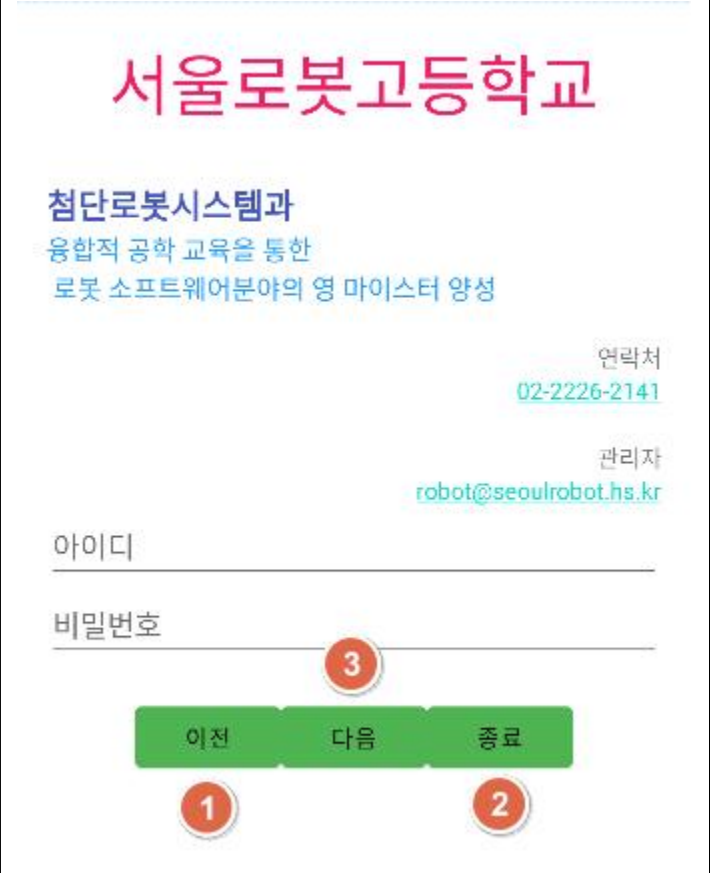
activity_main.xml	조건
	<p>① EditText id지정 : name</p> <p>② id지정 : btn3 버튼을 클릭했을 때 EditText(name)에 입력된 값의 길이가 5 미만이면 "이름을 다시 입력하세요" Toast 출력 5 이상이면 "*** 환영합니다" Toast 출력 (입력값이 GilDong이라면 "GilDong 환영합니다" 출력)</p>

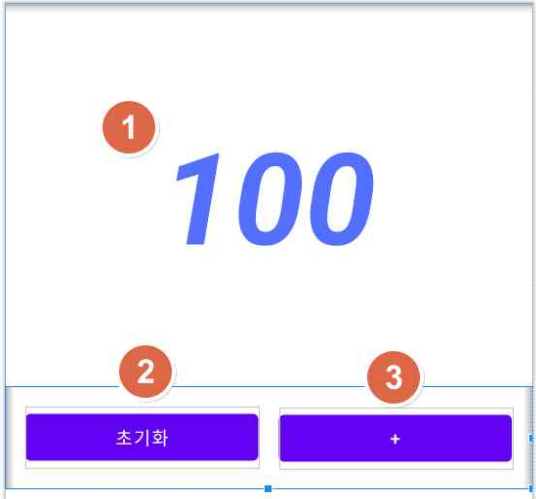

MainActivity.java

```


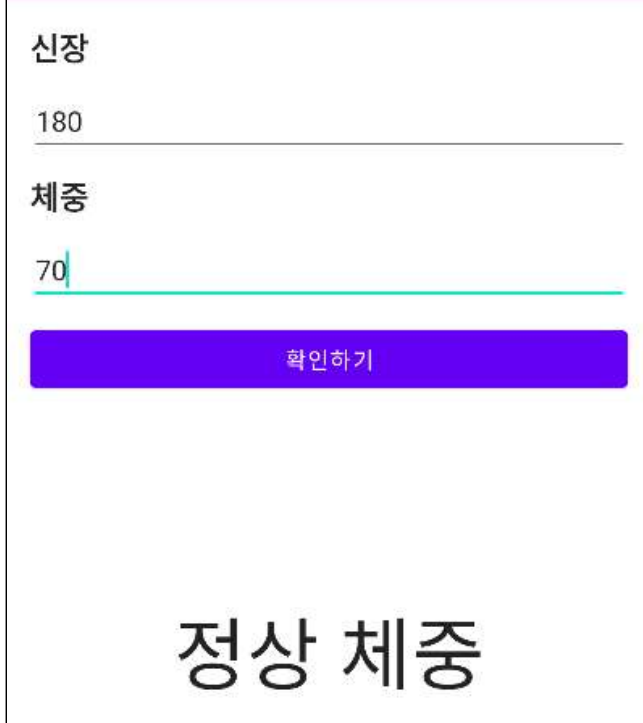
11 public class MainActivity extends AppCompatActivity implements View.OnClickListener {
12     EditText name; // 1. EditText 변수(객체) 선언
13
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.activity_main);
18         name = findViewById(R.id.name); // 2. EditText 연결
19
20         findViewById(R.id.btn1).setOnClickListener(this);
21         findViewById(R.id.btn2).setOnClickListener(this);
22         findViewById(R.id.btn3).setOnClickListener(this);
23     }
24
25     @Override
26     public void onClick(View v) {
27         switch (v.getId()) {
28             case R.id.btn1:
29                 Toast.makeText(getApplicationContext(), text: "BTN1 !!", Toast.LENGTH_SHORT).show();
30                 break;
31             case R.id.btn2:
32                 Toast.makeText(getApplicationContext(), text: "BTN2 !!", Toast.LENGTH_SHORT).show();
33                 break;
34             case R.id.btn3:
35                 String str_name = name.getText().toString(); // 3. String으로 만들기
36                 if (str_name.length() < 5)
37                     Toast.makeText(context: this, text: "이름을 다시 입력하세요", Toast.LENGTH_SHORT).show();
38                 else
39                     Toast.makeText(getApplicationContext(), text: name + " 환영해요 !!", Toast.LENGTH_SHORT).show();
40                 break;
41         }
42     }
43 }

```

수행 과제	지난 시간에 만들었던 school_info 레이아웃에 버튼 이벤트를 처리해보자.
완성 디자인	조건
	<ol style="list-style-type: none"> ① 이전 버튼을 클릭하면 Toast 메시지 "이전 버튼 클릭!!" ② 종료 버튼을 클릭하면 Toast 메시지 "종료 버튼 클릭!!" ③ 아이디와 비밀번호를 입력하고 다음 버튼을 클릭하면 Toast 메시지 <ul style="list-style-type: none"> - "아이디는 000, 비번은 000입니다" - 아이디나 비번이 입력되지 않았다면 "아이디와 비번을 확인해주세요"

수행 과제	NumberCounter 프로젝트에서 LinearLayout을 이용하여 숫자세기 앱을 작성해 보자.		
완성 디자인		화면 디자인 조건	
		① TextView - id : numberTextView - 높이 : 300dp - 글자크기 : 100sp - 굵게, 기울임 ② Button - id : resetButton - 여백 : 16dp ③ Button - id : plusButton - 여백 : 16dp	
		이벤트 처리	
		- [초기화] 버튼을 클릭하면 숫자가 0으로 초기화 - [+] 버튼을 클릭하면 숫자가 1씩 증가	
색상 등록하기		로그 기록	
		1. Logcat 이란? - 안드로이드 앱에서 발생하는 로그를 모니터링하고 출력하는 도구. - 앱의 동작, 에러, 경고, 정보 등을 추적하고 기록하여 디버깅 과정을 보다 효과적으로 도와줌. - Logcat은 안드로이드 스튜디오의 일부로 제공되며, 앱을 실행하거나 디바이스에 연결했을 때 로그를 실시간으로 확인할 수 있음. 2. Logcat 기능 - Verbose - 상세한 로그 메시지 - Debug - 디버깅 정보 - Info - 일반적인 정보 메시지 - Warning - 경고 메시지 - Error - 오류 메시지 3. 로그 사용법 <pre>Log.i(tag: "onClick", msg: "플러스 된 숫자는" + number);</pre> <pre>Log.d(tag: "onClick", msg: "리셋 된 숫자는" + number);</pre>	

```
31      @Override
32      public void onClick(View view) {
33          switch (view.getId()) {
34              case R.id.plusButton:
35                  number += 1;
36                  numberTextView.setText(Integer.toString(number));
37                  Log.i( tag: "onClick", msg: "플러스 된 숫자는" + number);
38                  break;
39              case R.id.resetButton:
40                  number = 0;
41                  numberTextView.setText(Integer.toString(number));
42                  Log.d( tag: "onClick", msg: "리셋 된 숫자는" + number);
43                  break;
44          }
45      }
46  }
```

수행 과제	BMI_Calculator 프로젝트에서 LinearLayout을 이용하여 BMI를 계산하는 앱을 작성해보자.														
완성 디자인	화면 디자인 조건														
 <p>BMI_Calculator</p> <p>신장 ①</p> <p>②</p> <p>체중 ①</p> <p>②</p> <p>확인하기 ③</p> <p>결과값 ④</p>	<p>[공통조건]</p> <ul style="list-style-type: none"> - 바깥 위 여백 : 10dp - strings.xml과 colors.xml 이용 <p>① TextView</p> <ul style="list-style-type: none"> - 크기 : 20sp - 굵게 <p>② EditText</p> <ul style="list-style-type: none"> - id : heightEditText, weightEditText - 숫자만 입력 <p>③ Button</p> <ul style="list-style-type: none"> - id : resultButton <p>④ TextView</p> <ul style="list-style-type: none"> - id : resultText - 높이 : 300dp - 크기 : 50dp 														
BMI_Calculator	이벤트 처리														
 <p>BMI_Calculator</p> <p>신장</p> <p>180</p> <p>체중</p> <p>70</p> <p>확인하기</p> <p>정상 체중</p>	<ul style="list-style-type: none"> - 입력이 하나라도 없는 경우 "빈 값이 있습니다" Toast 메시지 출력 - BMI 계산법 : 체중(kg)/신장²(m) (입력은 cm, 계산은 m 단위가 필요해서 나누기 100) - BMI 지수에 따라 결과값에 출력 <table border="1" data-bbox="770 1648 1484 1942"> <thead> <tr> <th>분류</th><th>BMI 지수</th></tr> </thead> <tbody> <tr> <td>고도비만</td><td>35 이상</td></tr> <tr> <td>중정도비만</td><td>30이상 ~ 34.9이하</td></tr> <tr> <td>경도비만</td><td>25이상 ~ 29.9이하</td></tr> <tr> <td>과체중</td><td>23이상 ~ 24.9이하</td></tr> <tr> <td>정상 체중</td><td>18.5이상 ~ 22.9이하</td></tr> <tr> <td>저체중</td><td>18.5미만</td></tr> </tbody> </table>	분류	BMI 지수	고도비만	35 이상	중정도비만	30이상 ~ 34.9이하	경도비만	25이상 ~ 29.9이하	과체중	23이상 ~ 24.9이하	정상 체중	18.5이상 ~ 22.9이하	저체중	18.5미만
분류	BMI 지수														
고도비만	35 이상														
중정도비만	30이상 ~ 34.9이하														
경도비만	25이상 ~ 29.9이하														
과체중	23이상 ~ 24.9이하														
정상 체중	18.5이상 ~ 22.9이하														
저체중	18.5미만														

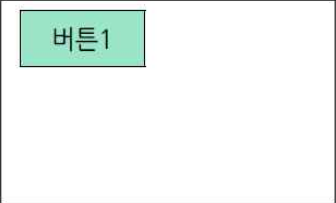
(2) Relative Layout (상대적, 렐러티브 레이아웃)

- ▶ 상대적 위치에 기반하여 위젯들을 배치하는 레이아웃
- ▶ 누구 밑에 누구, 누구 옆에 누구 식으로 서로 간의 관계를 지정하는 레이아웃
- ▶ 종류
 - 위젯과 상위 위젯과의 위치 관계 지정
 - 위젯끼리의 관계 지정

(가) 부모 레이아웃을 기준으로 하위 버튼을 배치하기 위한 속성

- ▶ default 속성 : layout_alignParentLeft(왼쪽), layout_alignParentTop(위),
⇒ 별도의 속성을 지정하지 않으면 부모영역 기준으로 “왼쪽 위”에 생성된다.

	속성 의미	속성 이름
가로	왼쪽	layout_alignParentLeft = “true”
	가로 중앙	layout_centerHorizontal = “true”
	오른쪽	layout_alignParentRight = “true”
중앙	정중앙	layout_centerInParent = “true”
세로	위	layout_alignParentTop = “true”
	세로 중앙	layout_centerVertical = “true”
	아래	layout_alignParentBottom = “true”

 <p>layout_alignParentLeft (기본) layout_alignParentTop (기본)</p>	 <p>layout_centerHorizontal layout_alignParentTop (기본)</p>	 <p>layout_alignParentRight layout_alignParentTop (기본)</p>
 <p>layout_alignParentLeft (기본) layout_centerVertical</p>	 <p>layout_centerInParent</p>	 <p>layout_alignParentRight layout_centerVertical</p>
 <p>layout_alignParentLeft (기본) layout_alignParentBottom</p>	 <p>layout_centerHorizontal layout_alignParentBottom</p>	 <p>layout_alignParentRight layout_alignParentBottom</p>

(나) 위젯끼리의 배치 방법

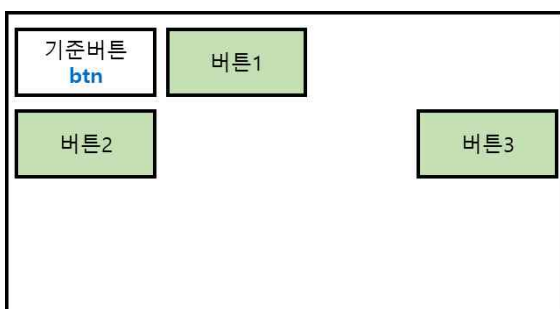
- ▶ 위젯끼리 상대적인 위치 관계를 지정하려면 “누구의 위” 처럼 “누구”를 지칭하기 위한 고유 식별 ID가 필요함
- ▶ id
 - 모든 뷰(view)는 이름을 가질 수 있다.
 - id는 “호출”의 개념으로 어떤 Event에 반응하는 대상이 된다.
 - id는 프로젝트 내에서 유일한 이름이어야 한다.

아이디 새로 생성 (+ 기호가 붙는다)	android:id="@+id/이름"
xml에서 아이디 호출	"@id/이름"
java에서 아이디 호출	R.id.이름

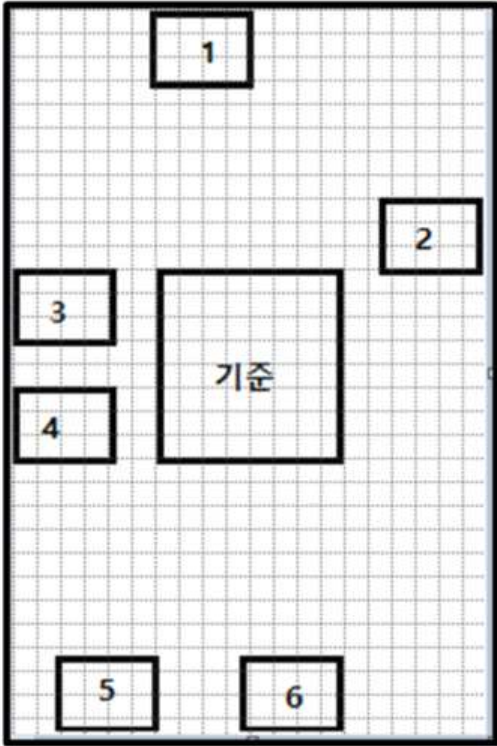
▶ 주요 속성

속성 이름	속성 의미
layout_above = "@id/기준ID"	기준ID의 위쪽에 배치한다.
layout_below = "@id/기준ID"	기준ID의 아래쪽에 배치한다.
layout_toLeftOf = "@id/기준ID"	기준ID의 왼쪽에 배치한다.
layout_toRightOf = "@id/기준ID"	기준ID의 오른쪽에 배치한다.
layout_alignLeft = "@id/기준ID"	기준ID와 왼쪽 선을 맞춘다.
layout_alignRight = "@id/기준ID"	기준ID와 오른쪽 선을 맞춘다.
layout_alignTop = "@id/기준ID"	기준ID와 위쪽 선을 맞춘다.
layout_alignBottom = "@id/기준ID"	기준ID와 아래쪽 선을 맞춘다.

▶ 예제



- 버튼1 : 기준버튼의 오른쪽에 배치
layout_toRightOf = "@id/btn"
- 버튼2 : 기준버튼의 아래쪽에 배치
layout_below = "@id/btn"
- 버튼3 : 기준버튼의 아래 오른쪽에 배치
layout_below = "@id/btn"
layout_alignParentRight = "true"

수행 과제	relative_1.xml 레이아웃 파일을 추가하고, RelativeLayout을 이용하여 다음 화면을 디자인 해보자.
수행 방법	중앙의 base 버튼을 기준으로 정하고, 상대적인 위치로 다른 위젯을 배치한다. 지금까지 배운 RelativeLayout의 속성을 이용한다.
완성 디자인	조건
	<p>① 기준 버튼</p> <ul style="list-style-type: none"> - 가로/세로 : 150dp - id : "base" <p>② 1~6번 버튼</p> <ul style="list-style-type: none"> - 가로 : 100dp - 세로 : 50dp

(3) Table Layout (표, 테이블 레이아웃)

- ▶ 테이블 형태로 위젯을 배치하는 레이아웃
- ▶ 테이블은 1개 이상의 행(table row) 객체로 구성
- ▶ 각 행(table row) 안에는 동일한 개수의 열(column)이 배치

▶ 테이블 레이아웃 구조

	0열	1열	2열	3열
0행->	성명	국어	수학	총점
1행->	홍길동	90	80	170
2행->	이순신	80	100	180
3행->	합계			370

① 0행 : 버튼뷰 4개

```
<TableRow>
```

```
버튼, 버튼, 버튼, 버튼
```

```
</TableRow>
```

② 1행 : 텍스트뷰 4개

```
<TableRow>
```

```
텍스트뷰, 텍스트뷰, 텍스트뷰, 텍스트뷰
```

```
</TableRow>
```

③ 2행 : 텍스트뷰 4개

```
<TableRow>
```

```
텍스트뷰, 텍스트뷰, 텍스트뷰, 텍스트뷰
```

```
</TableRow>
```

④ 3행 : 텍스트뷰 2개

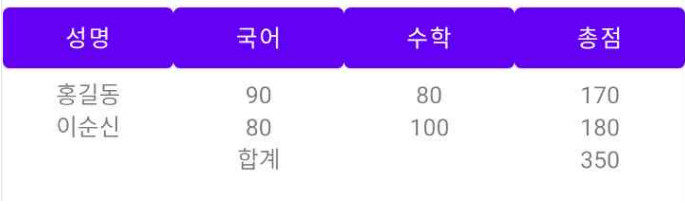
```
<TableRow>
```

```
텍스트뷰(layout_span=3), 텍스트뷰
```

```
</TableRow>
```

▶ 태그별 주요 속성

태그	속성	설명
TableLayout	<code>android:stretchColumns="*"</code>	테이블의 모든 열을 화면에 가득 차도록 늘려줌
	<code>android:stretchColumns="0,1"</code>	열 범위를 지정하면 특정 열만 늘어남 (열 번호는 0부터 시작) ⇒ 나머지 열에 배치된 위젯 먼저 표시하고, 남은 영역 전체를 0과 1열에 배치된 위젯으로 채움
TableRow	<code>android:layout_span="3"</code>	열 합치기 ⇒ 3개의 열을 합쳐서 위젯을 배치
	<code>android:layout_column="3"</code>	특정 열 위치에 위젯을 배치하라 ⇒ 3열에 위젯 배치

수행 과제	tablelayout_1_1.xml, tablelayout_1_2.xml 레이아웃 파일을 추가하고, TableLayout을 이용하여 다음 화면을 디자인 해보자.																																		
수행 방법	<p>모든 열을 화면에 가득 차도록 늘려준다.</p> <p>0열은 Button, 나머지는 TextView로 디자인한다.</p> <p>Quiz2는 바탕색(backgroundTint), 여백(margin), 글자색(textColor)등의 속성을 활용한다.</p>																																		
Quiz1 완성 디자인		Quiz2 완성 디자인																																	
 <table border="1"> <thead> <tr> <th>성명</th><th>국어</th><th>수학</th><th>총점</th></tr> </thead> <tbody> <tr> <td>홍길동</td><td>90</td><td>80</td><td>170</td></tr> <tr> <td>이순신</td><td>80</td><td>100</td><td>180</td></tr> <tr> <td>합계</td><td></td><td></td><td>350</td></tr> </tbody> </table>		성명	국어	수학	총점	홍길동	90	80	170	이순신	80	100	180	합계			350	 <table border="1"> <thead> <tr> <th>성명</th><th>국어</th><th>수학</th><th>총점</th></tr> </thead> <tbody> <tr> <td>홍길동</td><td>90</td><td>80</td><td>170</td></tr> <tr> <td>이순신</td><td>80</td><td>100</td><td>180</td></tr> <tr> <td>합계</td><td></td><td></td><td>350</td></tr> </tbody> </table>		성명	국어	수학	총점	홍길동	90	80	170	이순신	80	100	180	합계			350
성명	국어	수학	총점																																
홍길동	90	80	170																																
이순신	80	100	180																																
합계			350																																
성명	국어	수학	총점																																
홍길동	90	80	170																																
이순신	80	100	180																																
합계			350																																

(4) ConstraintLayout (컨스트레인트 레이아웃)

- ▶ 중첩된 뷰 그룹이 없는 단일 뷰 계층 구조로 크고 복잡한 레이아웃을 만들 수 있음.
- ▶ Relative 레이아웃과 비슷하지만 더 유연하고 사용하기 편리

(가) 제약조건

▶ 제약조건 개요

- 가로 및 세로축에 최소 하나의 제약조건이 있어야함.
- 그림1에서 뷰C에는 세로 제약조건이 없음. 제약 조건의 누락으로 컴파일 오류가 발생하지는 않지만 Layout Editor에서는 누락된 제약 조건을 툴바에 오류로 표시(Show Warnings and errors !)

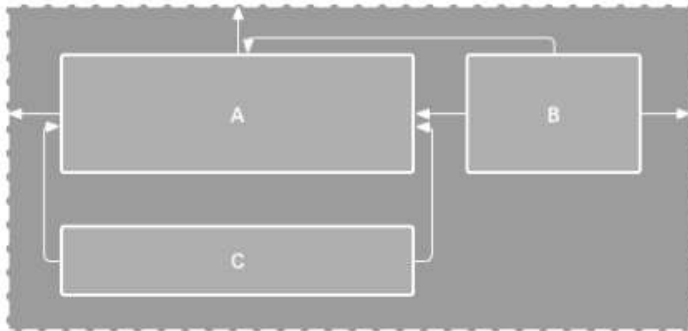


그림 1. 편집기에 뷰 C가 A 아래에 표시되지만 세로 제약조건이 없습니다.

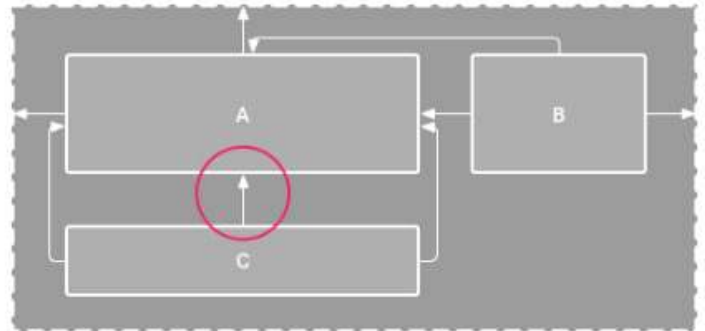


그림 2. 이제 보기 C는 보기 A 아래에 세로로 제한됩니다.

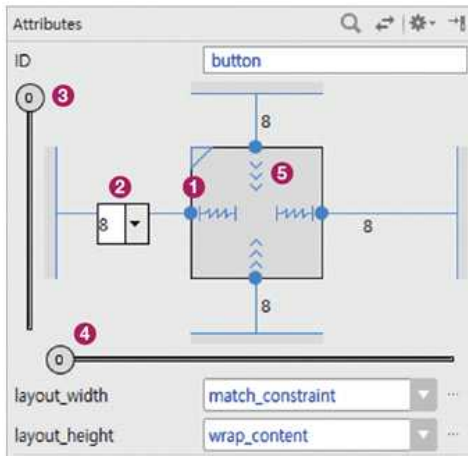
- ▶ 앵커포인트 : 위젯의 상, 하, 좌, 우에 찍혀있는 동그란 점으로, 이 점을 레이아웃이나 다른 위젯으로 드래그하면 해당 위젯과 드래그한 대상과의 관계가 형성됨

▶ 제약조건 추가 또는 삭제

- 제약조건 핸들을 클릭하여 사용가능한 앵커 포인트로 드래그.
- Attributes 창의 Layout 섹션에서 **Create a connection** + 버튼 중 하나를 클릭
- 제약조건을 클릭하여 선택한 다음 삭제를 클릭하거나 **ctrl 키**를 누른 상태에서 제약조건이 빨간색으로 바뀌면 클릭하여 삭제할 수도 있음.

<p>추가1. 뷰를 직접 드래그</p>	<p>추가2. Attribute 창의 버튼을 클릭</p>	<p>삭제. ctrl 키를 누른 상태에서 클릭하여 삭제</p>

▶ 위젯 기본 설정 [Attributes] 창



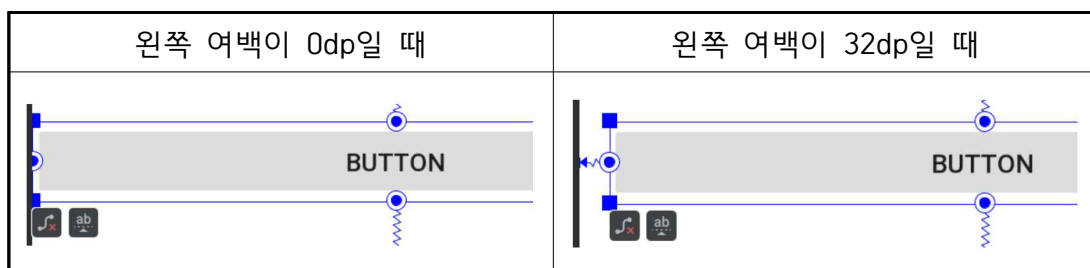
① 관계

- 사각형 테두리에 보이는 앵커포인트를 사용하여 부모 레이아웃이나 다른 위젯과의 관계를 설정
 - 반대 방향으로 작용하는 두 개의 제약을 동시에 적용하면 두 제약의 가운데에 배치.
- 다른 뷰를 기준으로 제약을 주기 위해서는 반드시 id를 설정해야함.



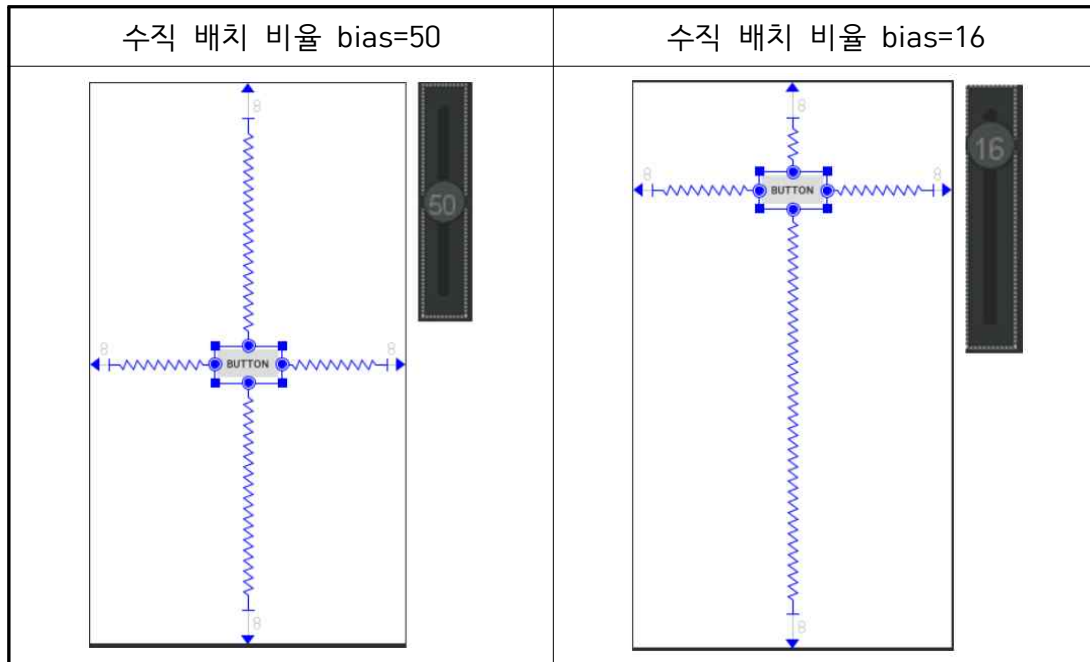
② 여백

- 위젯의 여백 설정
- 해당 방향으로 여백을 두기 위해서는 반드시 그쪽 방향에 제약이 있어야 함.






③④ 수직/수평 배치 비율

- bias라는 속성 값에 따라 위젯의 배치가 달라짐
- 전체 100으로 생각할 때, 수직은 위쪽의 공간, 수평은 왼쪽의 공간을 기준으로 표시됨

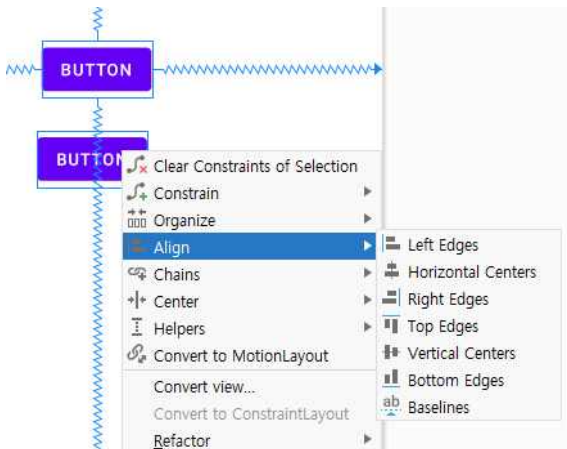


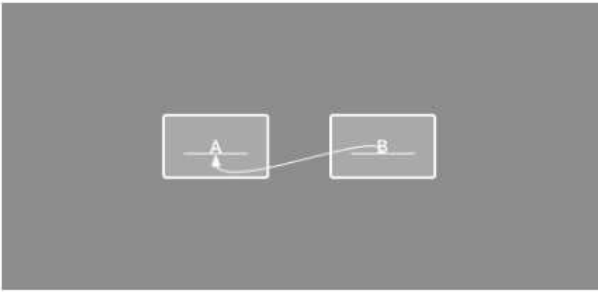
⑤ 너비, 높이 모드

아이콘	설명
Wrap Content 	위젯의 너비나 높이가 wrap_content로 설정
Fixed 	위젯의 너비나 높이가 고정 (ex. layout_width=100dp)
Match Constraints 	위젯의 너비나 높이가 0dp로 설정 (Constraint Layout에서는 match_parent를 사용할 수 없음. 제약조건과 일치(0dp)를 사용. 제약조건이 걸린 상위뷰와 동일한 너비를 가짐을 의미)

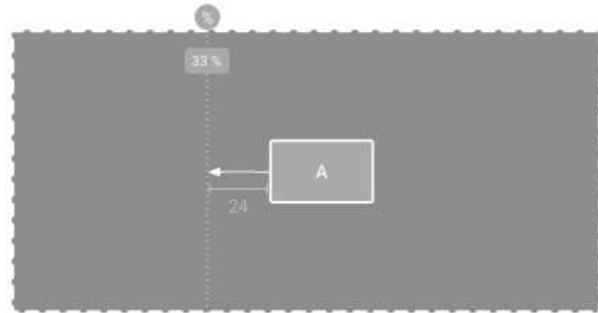
▶ 위젯 정렬

- 여러 개의 위젯을 선택한 후 마우스 오른쪽 버튼을 클릭하면 위젯을 정렬할 수 있음

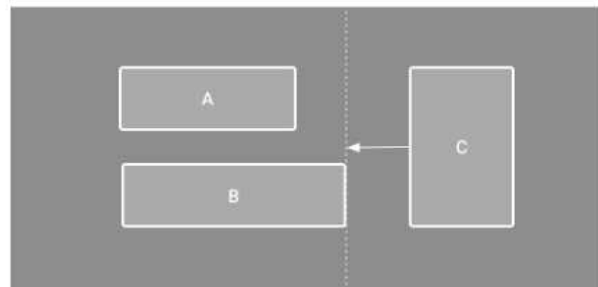
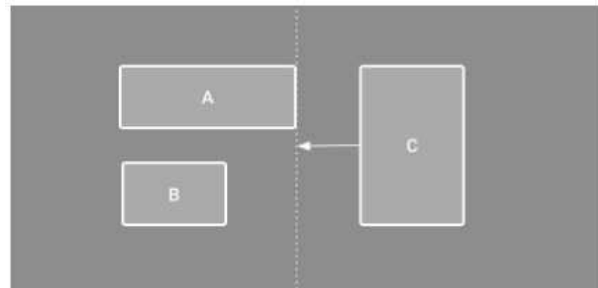




- 기준선 정렬 : 해당 뷰의 텍스트 기준선을 다른 뷰의 텍스트 기준선에 맞춤



- 가이드라인으로 제한 : 가이드라인을 만들기 위해서는 **Guidelines** 을 클릭하여 Add Vertical Guideline 또는 Add Horizontal Guideline으로 생성



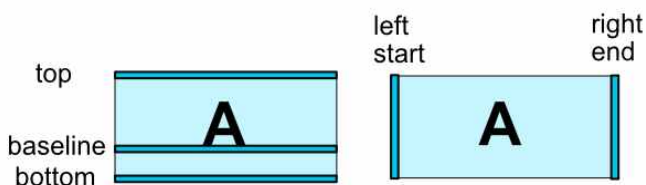
- 경계선으로 제한 : **Guidelines** 을 클릭하여 Add Vertical Barrier 또는 Add Horizontal Barrier을 클릭

① 상대적 배치

▶ 형식

`layout_constraint[위치1]_to[위치2]Of="[대상]"`

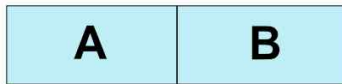
- 지정할 뷰의 [위치1]을 [대상]의 [위치2]에 배치
- [위치] : Right, Left, Top, Bottom, Baseline, Start, End
- [대상] : @+id/지정한ID (특정 위젯의 ID), parent (부모 뷰인 ConstraintLayout에 관련하여 배치)



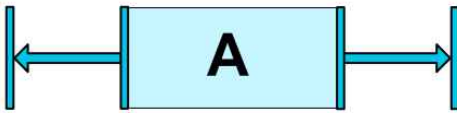
▶ 예

```
<androidx.constraintlayout.widget.ConstraintLayout ...>
    <Button android:id="@+id/buttonA" ... />
    <Button android:id="@+id/buttonB" ...
        app:layout_constraintLeft_toRightOf="@+id/buttonA" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

⇒ buttonB의 왼쪽편을 buttonA의 오른쪽에 배치하라



② 중앙 배치



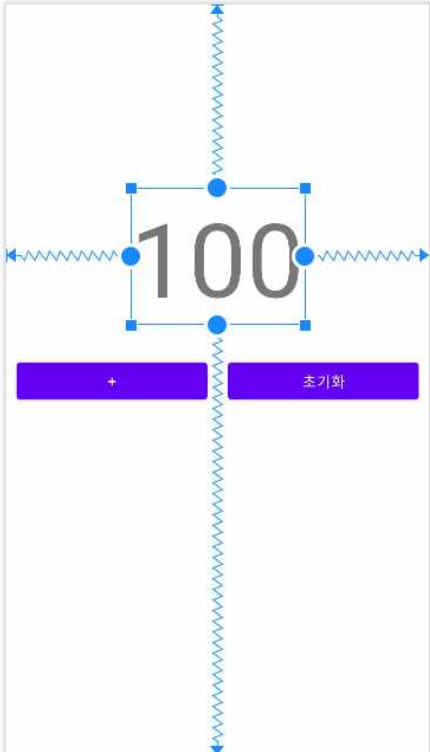
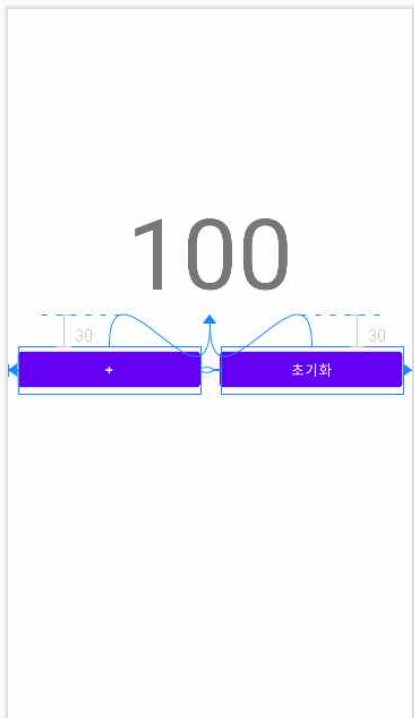
```
<androidx.constraintlayout.widget.ConstraintLayout ...>
    <Button android:id="@+id/button" ...
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"/>
</androidx.constraintlayout.widget.ConstraintLayout>
```

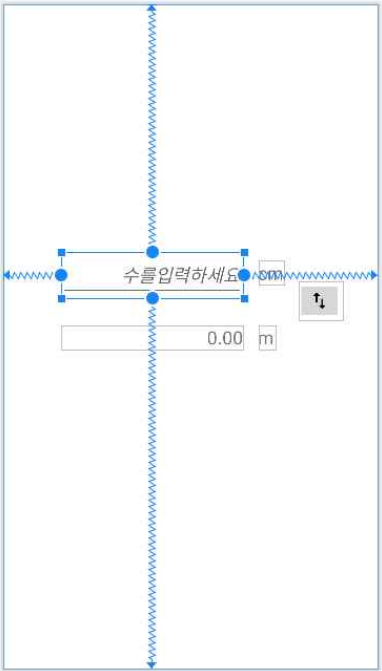


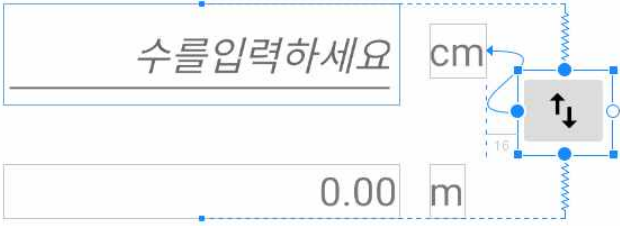
▶ bias 속성 : 이미 정렬된 View를 한쪽으로 치우치게 만듦

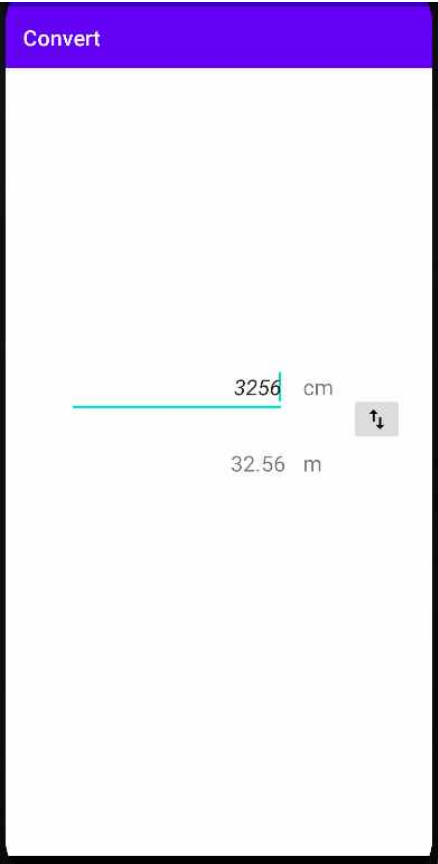
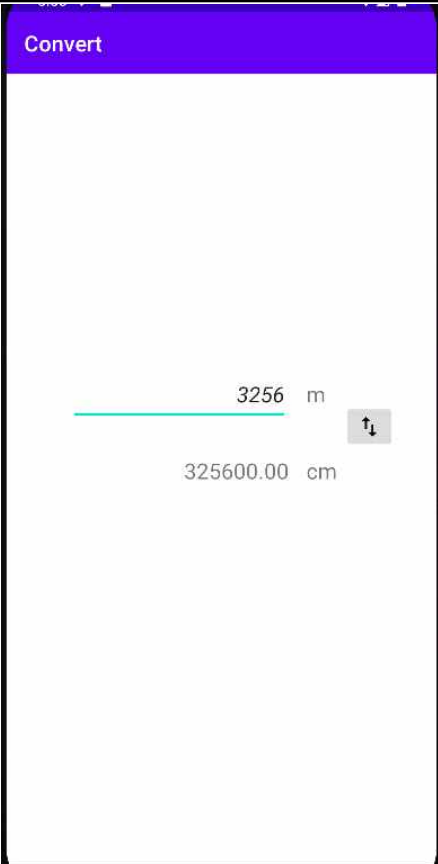
```
<androidx.constraintlayout.widget.ConstraintLayout ...>
    <Button android:id="@+id/button" ...
        app:layout_constraintHorizontal_bias="0.3"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"/>
</androidx.constraintlayout.widget.ConstraintLayout>
```

⇒ 중앙으로 배치된 A를 왼쪽으로 30% 치우치게 만듦



수행 과제	Counter 프로젝트에서 ConstraintLayout을 이용하여 화면을 구성해보자
화면 디자인	상세 조건
 <p>The diagram shows a UI design for a counter application. At the top, the number '100' is displayed in a large font. Below it are two buttons: a blue button with a '+' sign on the left and a blue button with the text '초기화' (Reset) on the right. Blue dashed lines with arrows indicate constraints: the number '100' is centered horizontally and vertically, and the buttons are positioned below it, also centered horizontally.</p>	<p>① TextView</p> <ul style="list-style-type: none"> - id : countTextView - 글자크기 : 100sp - 제약조건 <ul style="list-style-type: none"> * 상하좌우 제약조건은 그림참고 * 수직 기울임 : 30%
 <p>The diagram shows a detailed view of the two buttons. The left button is blue with a '+' sign, and the right button is blue with the text '초기화'. Blue dashed lines with arrows indicate constraints: the buttons are positioned below the number '100', and their horizontal positions are defined by constraints relative to the center and each other. The text '30' is shown near the buttons, likely indicating a margin or padding value.</p>	<p>② Button</p> <ul style="list-style-type: none"> - id : plusButton, clearButton - 가로 : 제약조건과 일치 - 바깥 위 여백 : 30dp - 수평방향 여백 : 10dp - 제약조건 <ul style="list-style-type: none"> * 상하좌우 제약조건은 그림참고

수행 과제	Convert 프로젝트에서 ConstraintLayout을 이용하여 단위를 변환하는 앱을 작성해보자.
화면 디자인	상세 조건
	<p>① EditText</p> <ul style="list-style-type: none"> - id : inputEditText - 가로 : 200dp - 오른쪽 정렬 - 소수점까지 입력가능 - 최대길이 20 - 글자크기 : 20sp - 글자 기울임 - 제약조건 <ul style="list-style-type: none"> * 상하좌우 제약조건은 그림참고 * 수직 기울임 : 40% * 수평 기울임 : 30%
	<p>② TextView</p> <ul style="list-style-type: none"> - id : outputTextView - 가로 : 제약조건과 일치 - 바깥 위 여백 : 30dp - 오른쪽 정렬 - 글자크기 : 20sp - 제약조건 <ul style="list-style-type: none"> * 상하좌우 제약조건은 그림참고
	<p>③④ TextView</p> <ul style="list-style-type: none"> - id : inputUnitTextView / outputUnitTextView - 바깥 왼쪽 여백 : 30dp - 글자크기 : 20sp - 제약조건 <ul style="list-style-type: none"> * 상하좌우 제약조건은 그림참고
	<p>⑤ ImageButton</p> <ul style="list-style-type: none"> - id : swapImageButton - 바깥 왼쪽 여백 : 16dp - vector asset을 활용해 swap 이미지를 추가하여 넣기 - 제약조건 <ul style="list-style-type: none"> * 상하좌우 제약조건은 그림참고

수행 과제	Convert 프로젝트에서 ConstraintLayout을 이용하여 단위를 변환하는 앱을 작성해보자.	
기능구현		상세 조건
		<p>① EditText에 값을 입력함과 동시에 변환된 값이 TextView에 표시.</p> <ul style="list-style-type: none"> - addTextChangedListener 활용 <p>② 결과값은 소수점 둘째 자리까지 표시</p> <ul style="list-style-type: none"> - String.format("%.2f", num) 활용
		<p>③ 이미지버튼을 클릭하면 m에서 cm로 단위를 변환할 수 있음.</p> <ul style="list-style-type: none"> - setOnClickListener 활용

[3] 액티비티(Activity)

▶ 액티비티의 정의

- 안드로이드 앱의 중요한 구성요소.
- 일반적인 프로그램의 경우 main()메서드에서 실행되는 패러다임을 가지나 안드로이드의 경우 특정 콜백 메서드를 호출하여 액티비티 코드를 실행시켜 시작됨.

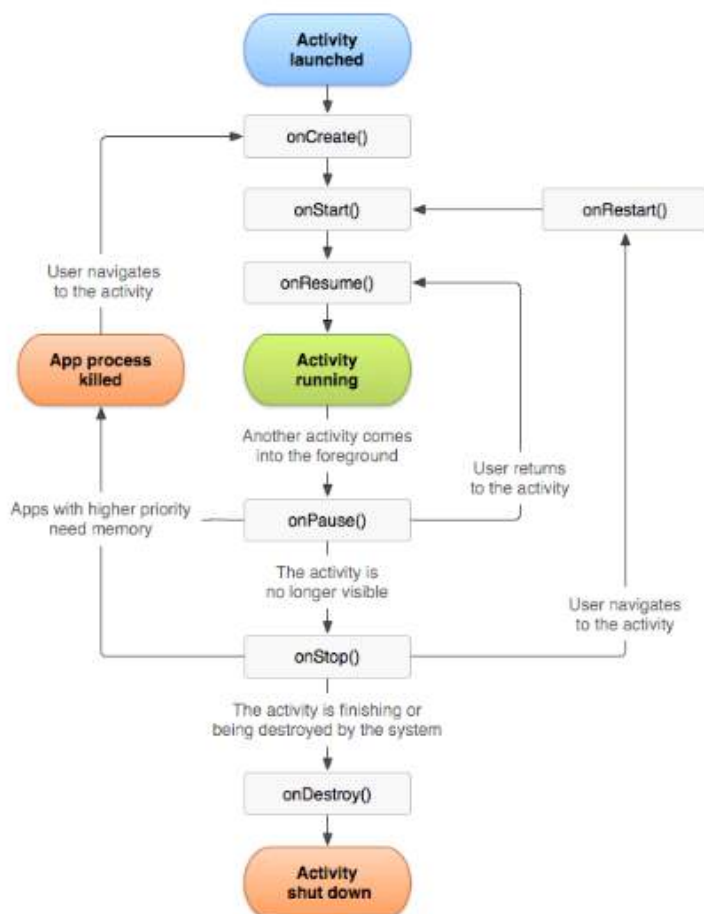
▶ manifest 구성

- 액티비티를 사용하기 위해서는 메니페스트파일을 열고 <activity> 요소를 <application>요소의 하위 요소로 추가해야함.

```
<manifest ... >
  <application ... >
    <activity android:name=".ExampleActivity" />
    ...
  </application ... >
  ...
</manifest >
```

▶ 액티비티 생명 주기 관리(Activity Life Cycle)

- 액티비티는 여러단계의 생명 주기 상태를 거침
- 단계별 생명주기 상태를 전환하는데 콜백 메소드를 사용.



① onCreate() : 액티비티가 활동을 생성할 때 실행되는 것으로 필수적으로 구현해야함. 액티비티 생명 주기동안 한번만 발생하는 시작 로직을 실행.

② onStart() : UI를 유지하는 코드가 초기화됨. 빠르게 완료된후 onResume 메소드를 호출

③ onResume() : 앱이 사용자와 상호작용시작. 전화가 걸려오거나 사용자가 다른 액티비티로 이동하여 기기 화면이 꺼지는 일이 발생하기 전까지 이 상태를 유지.

④ onPause() : 사용자가 액티비티를 떠나는 것을 나타내는 첫 번째 신호.

⑤ onStop() : 액티비티가 사용자가에서 표시가 되지 않는 상태

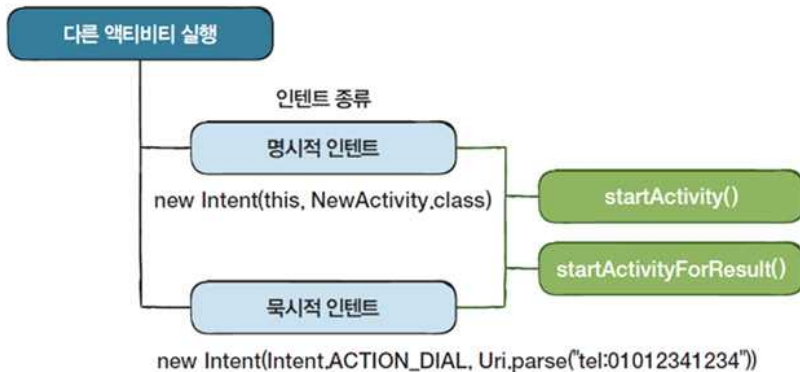
⑥ onRestart()

⑦ onDestroy() : 액티비티가 소멸되 전에 호출.

수행 과제	앱을 실행시키면서 Activity Life Cycle을 확인해보자.
수행 방법	아래의 코드를 작성하고 실행해서 로그를 확인하면서 Activity Life Cycle을 확인해보자.
MainActivity.java	
<pre> 8 public class MainActivity extends AppCompatActivity { 9 10 @Override 11 protected void onCreate(Bundle savedInstanceState) { 12 super.onCreate(savedInstanceState); 13 setContentView(R.layout.activity_main); 14 15 Log.d(tag: "Life Cycle", msg: "onCreate"); 16 } 17 18 @Override 19 protected void onStart() { 20 super.onStart(); 21 22 Log.d(tag: "Life Cycle", msg: "onStart"); 23 } 24 25 @Override 26 protected void onResume() { 27 super.onResume(); 28 Log.d(tag: "Life Cycle", msg: "onResume"); 29 } 30 31 @Override 32 protected void onStop() { 33 super.onStop(); 34 Log.d(tag: "Life Cycle", msg: "onStop"); 35 } 36 37 @Override 38 protected void onDestroy() { 39 super.onDestroy(); 40 41 Log.d(tag: "Life Cycle", msg: "onDestroy"); 42 } 43 44 @Override 45 protected void onPause() { 46 super.onPause(); 47 Log.d(tag: "Life Cycle", msg: "onPause"); 48 } 49 50 1 usage 51 @Override 52 protected void onRestart() { 53 super.onRestart(); 54 55 Log.d(tag: "Life Cycle", msg: "onRestart"); 56 } 57 } </pre>	<p><Q1> 앱을 처음 실행시키면 Activity Life Cycle 의 어떤 단계까지 실행되나? <u>onCreate -> onStart -> onResume</u></p> <p><Q2> Home 버튼을 클릭하면 Activity Life Cycle 의 어떤 단계까지 실행되나? <u>onPause -> onStop</u></p> <p><Q3> OverView 버튼을 클릭해서 재실행하면 Activity Life Cycle 의 어떤 단계까지 실행되나? <u>onRestart -> onStart -> onResume</u></p> <p><Q4> Intent 를 활용해 다른 Activity를 실행하면 Activity Life Cycle 의 어떤 단계까지 실행되나? <u>onPause -> onStop</u></p> <p>다시 원래 Activity 돌아왔을 때 호출되는 단계는? <u>onRestart -> onStart -> onResume</u></p> <p><Q5> Activity를 종료하면 Activity Life Cycle 의 어떤 단계까지 실행되나? <u>onPause -> onStop->onDestroy</u></p>

[4]. 인텐트 (Intent)

- ▶ 인텐트 : 다른 액티비티를 실행하거나 데이터를 전달할 수 있는 안드로이드 구성 요소



실행하는 방법	명시적 인텐트	실행하고자 하는 액티비티를 명시적으로 지정하는 방식
	묵시적 인텐트	지정한 액션과 데이터를 통해 실행할 수 있는 액티비티를 안드로이드가 찾아서 실행해주는 방식
실행하는 목적	startActivity()	단순 호출
	startActivityResult()	호출한 액티비티에서 결과를 돌려받고자 함

(1) 다른 액티비티 실행



그림 10-19 다른 액티비티 실행


- ① Intent 객체 생성하여 호출될 액티비티 지정

```
Intent intent = new Intent(MainActivity.this, SubActivity.class) ;
```

현재 액티비티
호출될 액티비티

- ② 객체 실행 메소드

```
startActivity(intent) ;
```

수행 과제	IntentTest 프로젝트에서 Intent를 활용하여 버튼을 클릭했을 때 다른 화면을 실행해보자.	
수행 방법	<ul style="list-style-type: none"> - MainActivity와 SubActivity를 생성하고 레이아웃에 버튼을 추가해보자. - MainActivity.java와 SubActivity.java 파일에 버튼을 클릭했을 때 다른 액티비티가 실행되도록 코드를 추가해보자. 	
activity_main.xml		activity_sub.xml
		
① id지정 : btn_next 버튼을 클릭했을 때 sub화면으로 전환		② id지정 : btn_prev 버튼을 클릭했을 때 main화면으로 전환
MainActivity.java		
<pre> 10 public class MainActivity extends AppCompatActivity implements View.OnClickListener { 11 12 @Override 13 protected void onCreate(Bundle savedInstanceState) { 14 super.onCreate(savedInstanceState); 15 setContentView(R.layout.activity_main); 16 findViewById(R.id.btn_next).setOnClickListener(this); 17 } 18 19 @Override 20 public void onClick(View v) { 21 22 // 다음 페이지로 화면 전환 23 // 화면을 전환할 때 사용하는 클래스 24 Intent intent = new Intent(packageContext: MainActivity.this, SubActivity.class); 25 26 // 화면 전환하기 27 startActivity(intent); 28 } 29 } </pre>		
SubActivity.java		
<pre> 9 public class SubActivity extends AppCompatActivity implements View.OnClickListener { 10 11 @Override 12 protected void onCreate(Bundle savedInstanceState) { 13 super.onCreate(savedInstanceState); 14 setContentView(R.layout.activity_sub); 15 16 findViewById(R.id.btn_prev).setOnClickListener(this); 17 } 18 19 @Override 20 public void onClick(View v) { 21 // 이전 페이지로 화면전환 22 // Intent intent = new Intent(SubActivity.this, MainActivity.class); 23 // startActivity(intent); 24 finish(); // 현재 액티비티 종료 25 } 26 } </pre>		

(2) 다른 액티비티에 데이터 전달



그림 10-20 다른 액티비티에 데이터 전달

MainActivity

- ① Intent 객체 생성하여 호출될 액티비티 지정

```
Intent intent = new Intent(MainActivity.this, SubActivity.class) ;
```

현재 액티비티호출될 액티비티

- ② intent에 값 저장

```
intent.putExtra("n", "gildong") ;
```

키 전달하는 값

- ③ 객체 실행 메소드

```
startActivity(intent) ;
```

OtherActivity

- ① MainActivity에서 보내준 intent 받기

```
Intent intent = getIntent() ;
```



- ② 저장된 데이터를 "키"로 꺼냄 (주의!! 저장된 데이터 형식에 맞게 메소드 호출)

```
String str = intent.getStringExtra("n") ;
```

키

- ③ 꺼낸 데이터 활용 (예: 텍스트뷰에 전달받은 문자열 보여주기)

```
txt.setText(str) ;
```


수행 과제	IntentTest 프로젝트에서 Intent를 활용하여 버튼을 클릭했을 때 EditText에 입력받은 값을 전달하면서 화면을 전환해보자.	
수행 방법	- activity_main.xml에 EditText를, sub_main.xml에 TextView를 추가해보자. - MainActivity.java에서 SubActivity.java로 화면 전환될 때 EditText에 입력된 값을 함께 전달하여 TextView에 표현해보자.	
activity_main.xml		activity_sub.xml
		
① EditText id지정 : edit ② ①에 값 입력 후 버튼을 클릭했을 때 입력 받은 값을 전달하면서 sub화면으로 전환		③ TextView id지정 : txt 전달받은 데이터를 이곳에 설정(setText())
MainActivity.java		
<pre> 10 public class MainActivity extends AppCompatActivity implements View.OnClickListener { 11 EditText edit; 12 @Override 13 protected void onCreate(Bundle savedInstanceState) { 14 super.onCreate(savedInstanceState); 15 setContentView(R.layout.activity_main); 16 17 edit = findViewById(R.id.edit); 18 findViewById(R.id.btn_next).setOnClickListener(this); 19 } 20 @Override 21 public void onClick(View v) { 22 Intent intent = new Intent(packageContext: this, SubActivity.class); 23 // intent에 값 저장 24 intent.putExtra(name: "t", edit.getText().toString()); 25 startActivity(intent); 26 } 27 } </pre>		
SubActivity.java		
<pre> 10 public class SubActivity extends AppCompatActivity implements View.OnClickListener { 11 TextView txt; 12 @Override 13 protected void onCreate(Bundle savedInstanceState) { 14 super.onCreate(savedInstanceState); 15 setContentView(R.layout.activity_sub); 16 17 txt = findViewById(R.id.txt); 18 // MainActivity에서 보내준 intent를 받는다 19 Intent intent = getIntent(); 20 // 키 "t"를 이용해 문자열 값을 꺼내어 변수에 저장한다 21 String str = intent.getStringExtra(name: "t"); 22 // 텍스트뷰에 값을 표현한다 23 txt.setText(str); 24 25 findViewById(R.id.btn_prev).setOnClickListener(this); 26 } 27 @Override 28 public void onClick(View v) { finish(); } 29 } 30 31 </pre>		

(3) 호출한 액티비티에서 결과를 돌려받음



MainActivity

① Intent 객체 생성하여 호출될 액티비티 지정

```
Intent intent = new Intent(MainActivity.this, SubActivity.class) ;
```

현재 액티비티 호출될 액티비티

② onActivityResult 생성하기

```
ActivityResultLauncher<Intent> startActivityResult = registerForActivityResult(
    new ActivityResultContracts.StartActivityForResult(),
    new ActivityResultCallback<ActivityResult>() {

        @Override
        public void onActivityResult(ActivityResult result) {

            if(result.getResultCode() == RESULT_OK){

                //resultCode 값이 OK 일때 동작할 내용 구현

            }

        }

    });
```

③ 객체 실행 메소드

```
startActivityResult(intent) ;
```

OtherActivity

① MainActivity에서 보내준 intent 받기


```
Intent intent = getIntent() ;
```

② 보내고 싶은 메시지를 입력하고, ResultCode 입력

```
intent.putExtra( name: "result", value: "success");
setResult(RESULT_OK, intent);
```

③ OtherActivity 종료

```
finish();
```

수행 과제	IntentTest 프로젝트에서 Intent를 활용하여 버튼을 클릭하며 화면을 전환하고 Prev 버튼을 클릭했을 때 Success 라는 메시지를 보낸뒤 화면에 출력하기	
수행 방법	<ul style="list-style-type: none"> - activity_main.xml에 TextView를 추가해보자. - SubActivity.java에서 다시 MainActivity.java로 화면 전환될 때 TextView에 Success 라는 메시지를 표현해보자. 	
activity_main.xml		activity_sub.xml
		
<p>① Button id지정 : next</p> <p>② TextView id지정 : resultTextView</p> <p>next 버튼을 클릭하면 화면이 전환되고, 다시 돌아왔을 때 받은 메시지가 화면에 출력</p>		<p>③ Button id지정 : prevButton</p> <p>prev 버튼을 클릭했을 때 해당 Success 라는 메시지를 이전 액티비티로 전달하며 해당 액티비티는 종료</p>
MainActivity.java		
<pre> 18 TextView resultTextView; 19 20 @Override 21 protected void onCreate(Bundle savedInstanceState) { 22 super.onCreate(savedInstanceState); 23 setContentView(R.layout.activity_main); 24 25 Button next = findViewById(R.id.next); 26 resultTextView = findViewById(R.id.resultTextView); 27 28 next.setOnClickListener(new View.OnClickListener() { 29 @Override 30 public void onClick(View view) { 31 Intent intent = new Intent(packageContext: MainActivity.this, SubActivity.class); 32 startActivityResult.launch(intent); 33 } 34 }); 35 } 36 37 1 usage 38 ActivityResultLauncher<Intent> startActivityResult = registerForActivityResult(39 new ActivityResultContracts.StartActivityForResult(), 40 new ActivityResultCallback<ActivityResult>() { 41 @Override 42 public void onActivityResult(ActivityResult result) { 43 if(result.getResultCode() == RESULT_OK){ 44 //resultCode 값이 OK 일때 동작할 내용 구현 45 String returnData = result.getData().getStringExtra(name: "result"); 46 resultTextView.setText(returnData); 47 } 48 } 49 }); </pre>		

SubActivity.java

```

11 public class SubActivity extends AppCompatActivity {
12
13     String str;
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.activity_sub);
18
19         Button prevButton = findViewById(R.id.prevButton);
20
21         prevButton.setOnClickListener(new View.OnClickListener() {
22             @Override
23             public void onClick(View view) {
24
25                 Intent intent = getIntent();
26
27                 intent.putExtra( name: "result", value: "success");
28                 setResult(RESULT_OK, intent);
29                 finish();
30             }
31         });
32     }
33 }

```

- ▶ finish()를 호출했을 때 이전 액티비티로 돌아가는 이유? Activity 가 실행되면 Stack 에 쌓이기 때문

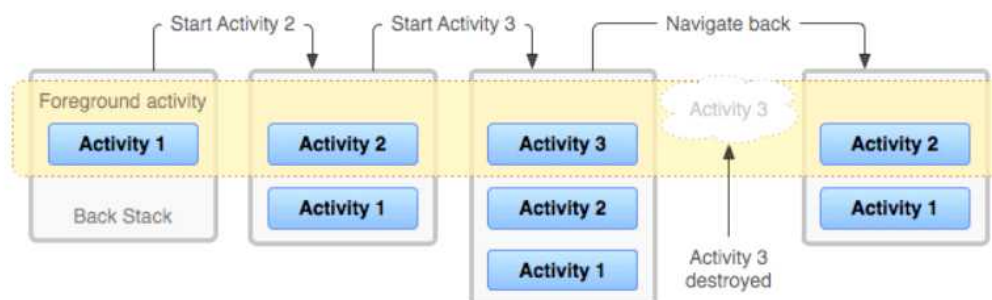


그림 1. 작업의 새로운 각 활동이 백 스택에 항목을 추가하는 방식을 나타냅니다. 사용자가 뒤로를 탭하거나 뒤로 동작을 취하면 현재 활동이 소멸되고 이전 활동이 다시 시작됩니다.


(4) 묵시적 인텐트

- ▶ 지정한 액션과 데이터를 통해 실행할 수 있는 액티비티를 안드로이드가 찾아서 실행해 주는 방식
- ▶ Intent를 생성하고 액션을 지정한 다음 반드시 startActivity()를 호출해야 실행됨

Action	설명
ACTION_DIAL	<ul style="list-style-type: none"> ▶ 전화 다이얼 액티비티 호출 <pre>Intent intent = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:01012345678")) ;</pre>
ACTION_CALL	<ul style="list-style-type: none"> ▶ 전화 액티비티 호출 <pre>Intent intent = new Intent(Intent.ACTION_DIAL , Uri.parse("tel:01012345678")) ;</pre> <ul style="list-style-type: none"> - AndroidManifest.xml 파일 내에 전화걸기 권한 추가가 필요함 <code><uses-permission android:name="android.permission.CALL_PHONE" /></code> - 타켓 sdk 버전이 23 이상이면 런타임 권한 문제를 코드에 명시해야함
ACTION_WEB_SEARCH	<ul style="list-style-type: none"> ▶ 웹 검색 액티비티 호출 <pre>Intent intent = new Intent(Intent.ACTION_WEB_SEARCH) ; Intent.putExtra(SearchManager.QUERY, "안드로이드") ;</pre>
ACTION_VIEW	<ul style="list-style-type: none"> ▶ 브라우저 실행 <pre>Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.naver.com")) ;</pre> <ul style="list-style-type: none"> ▶ 구글맵 <pre>Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("geo:0,0?q=서울")) ;</pre> <ul style="list-style-type: none"> ▶ 구글맵(위도,경도,확대/축소) <pre>Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("geo:37.4806003,127.0838545?z=15")) ;</pre> <p>// z 값은 zoom을 의미하며 1~21까지 지정가능</p> <ul style="list-style-type: none"> ▶ SMS 전송 <pre>Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("smsto:01012345678")) ; intent.putExtra("sms_body", "안녕하세요!!") ;</pre>
ACTION_SENDTO	<ul style="list-style-type: none"> ▶ 이메일 전송을 지정하는 액션 <pre>Intent intent = new Intent(Intent.ACTION_SENDTO, Uri.parse("mailto:seooulrobot@gmail.com")) ;</pre>

▶ 용어

- URI 통합 자원 식별자(Uniform Resource Identifier: 인터넷에 있는 자원을 나타내는 유일한 주소
- URL(Uniform Resource Locator) : URI의 하위개념으로, 인터넷 도메인 이름이나 IP 주소는 물론, 이메일, 파일 전송과 같이 컴퓨터 네트워크 정보 자원을 이용하는 모든 형태의 자원의 위치

수행 과제	IntentTest 프로젝트에서 묵시적 Intent를 활용하여 버튼을 클릭했을 때 다른 앱을 실행해보자.	
수행 방법	- activity_main.xml에 Button 세개를 추가해보자. - MainActivity.java에서 묵시적 Intent를 활용하여 각각 다른 앱을 실행해보자.	
activity_main.xml		조건
		① id 지정 : btn_dial 버튼 클릭했을 때 전화 다이얼 실행 ② id 지정 : btn_map 버튼 클릭했을 때 구글맵에서 “서울” 검색 ③ id 지정 : btn_sms 버튼 클릭했을 때 sms 실행
MainActivity.java		
<pre> 11 public class MainActivity extends AppCompatActivity implements View.OnClickListener { 12 EditText edit; 13 14 @Override 15 protected void onCreate(Bundle savedInstanceState) { 16 super.onCreate(savedInstanceState); 17 setContentView(R.layout.activity_main); 18 19 edit = findViewById(R.id.edit); 20 findViewById(R.id.btn_next).setOnClickListener(this); 21 findViewById(R.id.btn_dial).setOnClickListener(this); 22 findViewById(R.id.btn_map).setOnClickListener(this); 23 findViewById(R.id.btn_sms).setOnClickListener(this); 24 } 25 26 @Override 27 public void onClick(View v) { 28 switch (v.getId()) { 29 case R.id.btn_next: 30 Intent intent = new Intent(packageContext: this, SubActivity.class); 31 intent.putExtra(name: "t", edit.getText().toString()); 32 startActivity(intent); 33 break; 34 case R.id.btn_dial: 35 Intent dialIntent = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:01012345678")); 36 startActivity(dialIntent); 37 break; 38 case R.id.btn_map: 39 Intent mapIntent = new Intent(Intent.ACTION_VIEW, Uri.parse("geo:0,0?q=서울")); 40 startActivity(mapIntent); 41 break; 42 case R.id.btn_sms: 43 Intent smsIntent = new Intent(Intent.ACTION_VIEW, Uri.parse("smsto:01012345678")); 44 smsIntent.putExtra(name: "sms_body", value: "안녕하세요!! 홍길동입니다!!"); 45 startActivity(smsIntent); 46 break; 47 } </pre>		

[5]. Image View

- ▶ 이미지를 화면에 보여주기 위한 위젯

(1) 클래스 계층구조

```
java.lang.Object
└─ android.view.View
    └─ android.widget.ImageView
```

(2) 출력할 이미지 지정하기

- ▶ 이미지뷰의 src 속성으로 출력할 이미지를 지정
- ▶ 프로젝트의 res > drawable 폴더에 그래픽 파일을 복사해 두고 이 이미지를 '@drawable/id' 형식으로 레이아웃이나 코드에서 사용함
- ▶ 그래픽 파일 이름은 반드시 소문자 영어로 지정하고, 형식은 png 또는 jpg로 지정

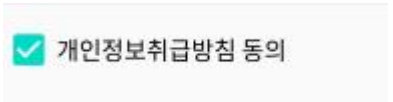
(3) 주요 속성

속성	속성값	설명
scaleType (스케일 타입)		이미지뷰와 이미지의 크기가 일치하지 않을 때 스케일 타입(scaleType) 속성을 이용하여 이미지를 어떤 방식으로 확대, 축소할 것인지를 지정할 수 있다.
	fitXY	가로, 세로 모두 확장하여 뷰의 크기를 다 채우며, 가로세로비를 유지하지 않는다.
	fitStart	가로세로비를 유지하여 스케일링하며, 왼쪽 상단에 놓는다.
	fitCenter	가로세로비를 유지하여 스케일링하며, 중앙에 놓는다.
	center	이미지를 중앙에 놓으며, 스케일링하지 않는다.
	centerCrop	가로세로비를 유지하여 스케일링하며, 뷰의 크기 이상으로 채운다. 따라서 이미지 일부가 잘릴 수 있다.
	centerInside	가로세로비를 유지하여 스케일링하며, 뷰의 크기 이하로 채운다. 뷰가 더 작으면 이미지 축소가 발생한다.
src	android:src=" @drawable/파일명 "	ImageView의 내용으로 drawable 폴더 내의 임의의 이미지를 지정한다. - 파일명은 반드시 영어 소문자, 숫자로 지정 (숫자로 시작X) - 한글 안됨, 공백 안 됨, 특수문자는 _(언더바)만 가능
void setImageResource(int resId)		ImageView에 출력할 이미지를 지정할 때 사용한다.









[6]. CheckBox

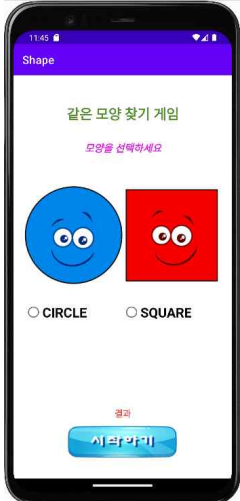


- ▶ 사용자가 특정 항목을 선택할 수 있게 해주는 위젯

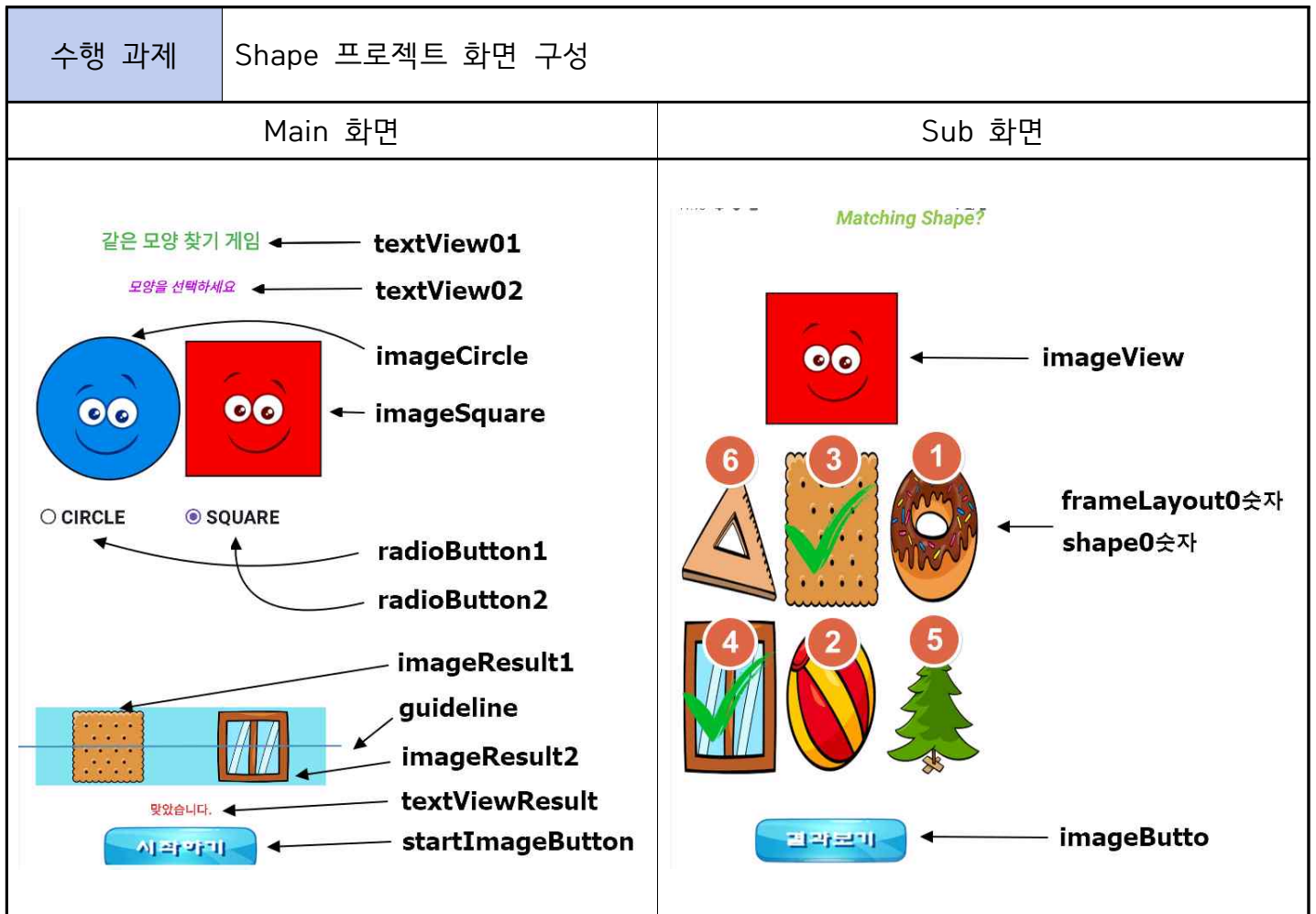
<pre><CheckBox android:layout_width="wrap_content" android:layout_height="wrap_content" android:checked="true" android:clickable="true" android:text="개인정보취급방침 동의"/></pre>	
<p>[속성]</p> <p>checked : 체크박스에 기본 체크 표시 할 것인지 말 것 인지를 지정</p> <p>clickable : 사용자가 체크박스를 클릭할 수 있는지 설정 (false로 설정하면 클릭할 수 없음)</p>	
<p>[메소드]</p> <p>boolean isChecked() - 체크여부를 true/false로 return</p> <p>void setChecked(boolean checked) - 체크상태 설정</p> <p>void setOnCheckedChangeListener() - 체크상태 변경을 감지하는 리스너 등록</p> <p>void toggle() - 체크상태 반대로 변경</p>	


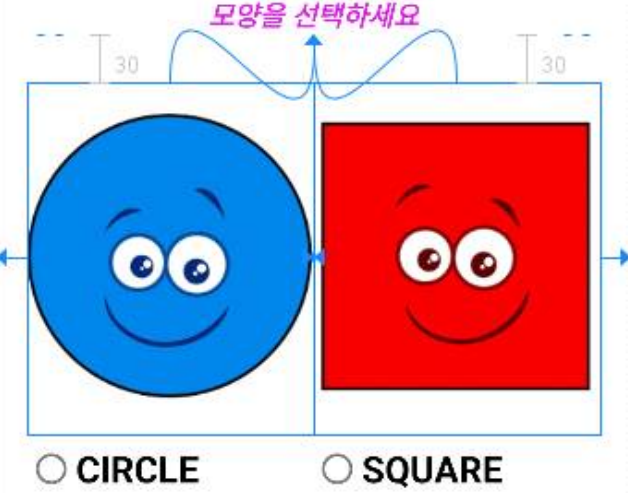
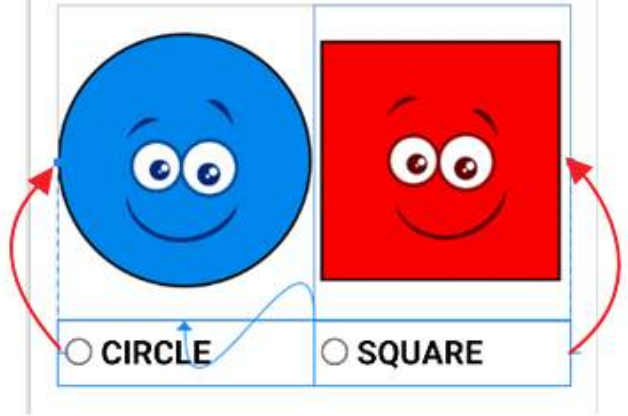
[7]. RadioButton

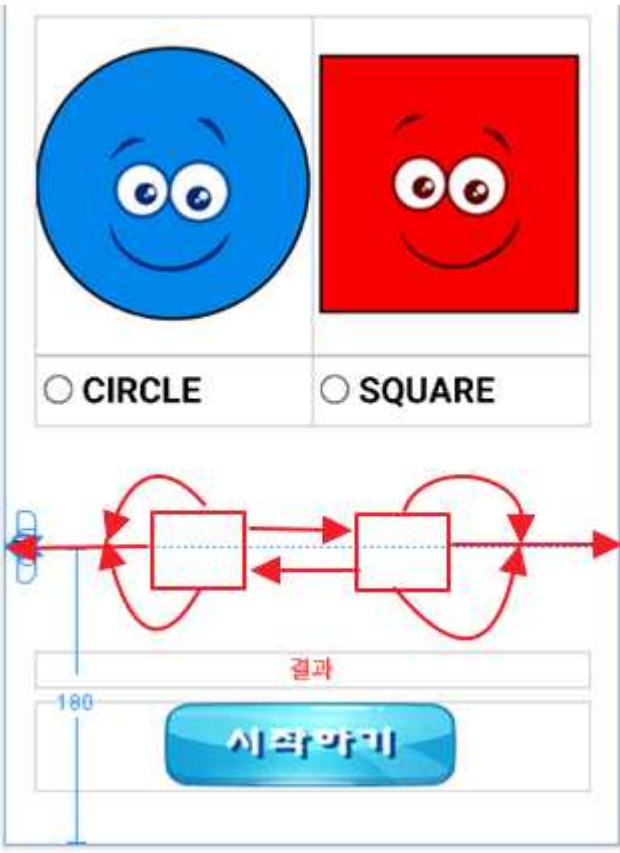
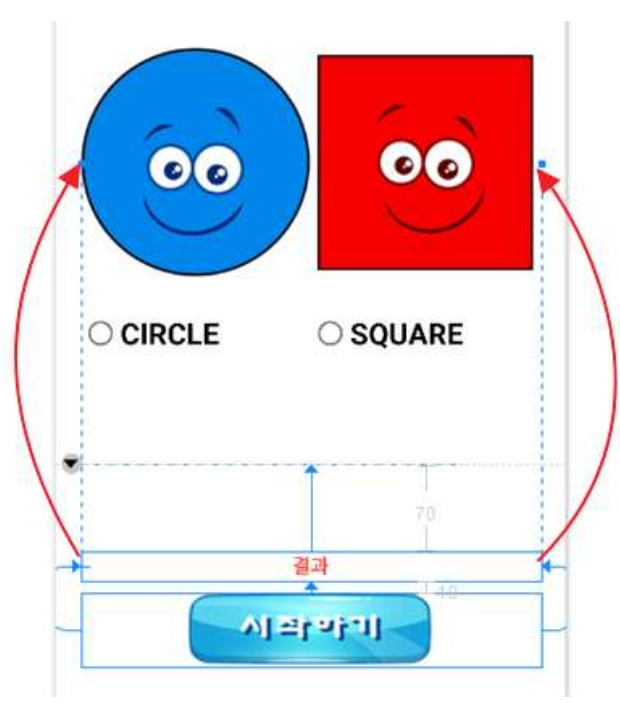
- ▶ CheckBox와 유사하게 체크하는 방식
- ▶ 반드시 RadioGroup과 함께 사용해야 여러 개 중 한 개만 선택할 수 있음
- ▶ RadioGroup과 같이 다른 뷰를 포함할 수 있는 위젯을 "뷰 컨테이너"라고 함

<pre><RadioGroup android:layout_width="match_parent" android:layout_height="wrap_content" > <RadioButton android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="선택1"/> <RadioButton android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="선택2"/> <RadioButton android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="선택3"/> </RadioGroup></pre>	<table border="1"> <thead> <tr> <th data-bbox="906 1339 1187 1420">라디오그룹 미적용</th><th data-bbox="1187 1339 1468 1420">라디오그룹 적용</th></tr> </thead> <tbody> <tr> <td data-bbox="906 1420 1187 1666">  </td><td data-bbox="1187 1420 1468 1666">  </td></tr> </tbody> </table>	라디오그룹 미적용	라디오그룹 적용		
라디오그룹 미적용	라디오그룹 적용				
					
<p>[속성]</p> <p>checkBox와 동일함</p>					
<p>[메소드]</p> <p>checkBox와 동일함</p>					

수행 과제	Shape 프로젝트 소개 - 같은 모양 찾기 게임	
화면 디자인	상세 조건	
	<p><UI></p> <ul style="list-style-type: none"> - constraint layout 으로 화면구현 <p><기능></p> <ul style="list-style-type: none"> -Circle, Square 중에서 선택하기 -시작하기 버튼을 클릭하면 화면전환 	
	<p><UI></p> <ul style="list-style-type: none"> - Linear layout, Table layout, Frame layout 으로 화면구현 (**수업시간에 구현하지는 않음) <p><기능></p> <ul style="list-style-type: none"> -동그란 모양을 선택하면 화면에 체크표시가됨 -두번체크 할수 있음 -결과보기 버튼을 클릭하면 이전화면으로 돌아감 	
	<p>처음화면으로 돌아와서 같은모양을 찾았는지 확인해서 결과를 출력</p>	



수행 과제	Shape 프로젝트에서 ConstraintLayout을 이용하여 첫페이지 UI를 구현
화면 디자인	상세 조건
① SubActivity 와 activity_sub.xml 하나의 액티비티 더 추가하기 ② drawable 폴더아래에 앱만들 때 사용할 이미지파일 복사해 넣기 ③ Constraint Layout 에 안쪽 여백 20dp	
	④ TextVeiw - id : textView01 , textView02 - 가로 : 제약조건만큼 - 가운데 정렬 - 위쪽여백 30dp - 글자크기는 25sp, 18sp - 텍스트컬러는 화면참고 - 텍스트스타일은 화면참고 - 제약조건 * 상하좌우 제약조건은 그림참고
	⑤ ImageView - id :imageCircle / imageSquare - 가로 : 제약조건만큼 - 이미지자르기 : 안쪽중앙축소 - 위쪽여백 30dp - 제약조건 * 상하좌우 제약조건은 그림참고
	⑥ RadioGroup - orientation : 수평방향 - 가로 : 제약조건만큼 - 제약조건 * 상하좌우 제약조건은 그림참고 ⑦ RadioGroup안에 두 개의 RadioButton 추가 - id : radioButton1 / radioButton2 - 가로 : 제약조건만큼, 가중치 1:1 - 글자크기 24sp - 굵게

수행 과제	Shape 프로젝트에서 ConstraintLayout을 이용하여 첫페이지 UI를 구현
화면 디자인	상세 조건
	<p>⑧ Guideline 추가</p> <ul style="list-style-type: none"> - 결과를 출력하는 ImageView 가 화면에 표시될 때도 있고 안될 때도 있으므로 가이드라인을 추가해서 이를 기준으로 레이아웃시킴 - id : guideline - 수평가이드라인 - bottom에서 180dp 만큼 위쪽에 위치 <p>⑨ ImageView</p> <ul style="list-style-type: none"> - id : imageResult01 / imageResult02 - 가로 : 제약조건만큼 - 높이 : 100dp - 배경색깔 : 민트색 - visibility를 gone 으로 설정 - 제약조건 <ul style="list-style-type: none"> * top과 bottom은 가이드라인에 제약 * start와 end 는 부모, imageView 에 제약
	<p>⑩ TextView</p> <ul style="list-style-type: none"> - id : textViewResult - 가로 : 제약조건만큼 - 글자크기 : 16sp - 굵게 - 위쪽여백 70dp - 글자 색상, 정렬 그림 참조 - 제약조건은 그림참고 <p>⑪ ImageButton</p> <ul style="list-style-type: none"> - id: startImageButton - 가로 : 제약조건만큼 - 위쪽여백 10dp - 배경으로 투명하게(#00000000) - 제약조건은 그림 참조

[5] 뷰바인딩

- ▶ 뷰와 상호작용하는 코드를 쉽게 작성
- ▶ 바인딩 클래스의 인스턴스에는 상응하는 레이아웃 ID가 있는 모든 뷰의 직접 참조가 가능
- ▶ 대부분의 경우 뷰 결합이 findViewById를 대체
- ▶ 바인딩되는 결합클래스의 이름은 XML 파일의 이름을 카멜표기법으로 변환하고 끝에 Binding을 추가하여 생성됩니다.

ex) XML 파일이 activity_lotto.xml 인 경우 바인딩 클래스의 이름은 ActivityLottoBinding 이 됩니다.

build.gradle파일에 아래의 내용 추가

```

5  android {
6      namespace 'com.example.games_2023'
7      compileSdk 33
8
9      defaultConfig {
10         applicationId "com.example.games_2023"
11         minSdk 21
12         targetSdk 33
13         versionCode 1
14         versionName "1.0"
15
16         testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
17     }
18
19     buildTypes {
20         release {
21             minifyEnabled false
22             proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
23         }
24     }
25     compileOptions {
26         sourceCompatibility JavaVersion.VERSION_1_8
27         targetCompatibility JavaVersion.VERSION_1_8
28     }
29
30     viewBinding{
31         enabled = true
32     }
33 }

```

- ▶ viewBinding 요소를 build.gradle 파일에 넣고 싱크를 맞추기

MainActivity.java

```

18 public class MainActivity extends AppCompatActivity {
    14 usages
19     ActivityMainBinding binding;
20
21     @Override
22     protected void onCreate(Bundle savedInstanceState) {
23         super.onCreate(savedInstanceState);
24         binding = ActivityMainBinding.inflate(getLayoutInflater());
25         setContentView(binding.getRoot());
26     }
27 }

```

- ▶ onCreate 메소드 안에서 위와 같이 설정
- ActivityMainBinding 클래스에 포함된 inflate() 메소드를 호출하여 객체(인스턴스) 생성
- getRoot() 메서드를 활용하여 루트뷰 참조를 가져옴 (constraint layout 이 루트뷰)
- 루트뷰를 setContentView()에 전달하여 화면상의 활성 뷰로 만들어줌

SubActivity.java

```

13 public class SubActivity extends AppCompatActivity {
14
15     17 usages
16     ActivitySubBinding binding;
17
18     @Override
19     protected void onCreate(Bundle savedInstanceState) {
20         super.onCreate(savedInstanceState);
21         binding = ActivitySubBinding.inflate(getLayoutInflater());
22         setContentView(binding.getRoot());
23     }
24 }

```

- ▶ 위와 동일, Activity_sub.xml의 루트뷰는 LinearLayout.

수행 과제	Shape 프로젝트에서 기능구현 1단계 - <MainActivity.java> 시작하기 버튼을 클릭했을 때	
화면 디자인		상세 조건
<ul style="list-style-type: none"> - 라디오버튼 클릭 여부에 따라서 동작 구현. 라디오버튼이 클릭되지 않았다면 “찾고 싶은 모양을 클릭하세요” 라는 토스트 메시지 띄우기 - 어떤 라디오버튼이 클릭되었는지 확인 후, 선택된 모양정보를 포함하여 인텐트 실행 - 호출한 서브액티비티에서 다시 결과값을 돌려받을 수 있는 인텐트를 생성하기 		
<pre> 13 public class MainActivity extends AppCompatActivity { 14 15 5 usages 16 ActivityMainBinding binding; 17 2 usages 18 ActivityResultLauncher<Intent> launcher; 19 20 @Override 21 protected void onCreate(Bundle savedInstanceState) { 22 super.onCreate(savedInstanceState); 23 binding = ActivityMainBinding.inflate(getLayoutInflater()); 24 setContentView(binding.getRoot()); 25 26 binding.startImageButton.setOnClickListener(new View.OnClickListener() { 27 @Override 28 public void onClick(View view) { 29 30 Intent intent = 31 32 if(binding.radioButton1.isChecked()){ //Circle 선택시 33 <input type="text"/> 34 } 35 else if(binding.radioButton2.isChecked()){ //Square 선택시 36 <input type="text"/> 37 } 38 else{ 39 <input type="text"/> 40 } 41 } 42 }); 43 } 44 } 45 46 47 48 </pre>		

수행 과제	Shape 프로젝트에서 기능구현 2단계 - <SubActivity.java> 로 화면이 전환되었을 때
화면 디자인	상세 조건
- 메인액티비티에서 선택받은 모양을 확인하여 화면에 이미지를 표시하기	
<pre> 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 </pre>	<pre> public class SubActivity extends AppCompatActivity { 4 usages ActivitySubBinding binding; @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); binding = ActivitySubBinding.inflate(getLayoutInflater()); setContentView(binding.getRoot()); Intent intent = getIntent(); int shapeNum = if(shapeNum == 0){ } else if(shapeNum == 1){ } } } </pre>

수행 과제	Shape 프로젝트에서 기능구현 3단계 - <SubActivity.java> 로 화면이 전환되었을 때
화면 디자인	상세 조건
<ul style="list-style-type: none"> - 어떤 이미지가 체크되었는지 확인하기 위해서 길이 6개짜리 배열 생성 - 2개까지 선택가능하므로 변수를 생성 - 체크표시를 할지, 체크표시를 취소할지, 아무동작도 하지 않을지를 결정하기 위해서 상수 생성 	
19 20 21 22 23	<pre> 5 usages int countShape = 0; 7 usages int[] selectShape = new int[6]; 7 usages private static final int CHECK = 1; 8 usages private static final int UNCHECK = 2; 2 usages private static final int NONE = 3; </pre>
<ul style="list-style-type: none"> - 선택한 이미지에 체크표시를 할지, 취소할지, 아무동작도 하지 않을지를 결정하는 메소드를 추가 • 선택 할 때 : countShape 의 값을 1 증가, 배열 selectShape 값을 1로 변경 • 선택을 취소할 때 : countShape 의 값을 1 감소, 배열 selectShape 값을 0로 변경 • 선택하거나 취소할 때, countShape 의 값이 2인경우를 고려해야 함 : 취소는 가능하지만 선택은 불가능 	
112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135	<pre> private int selectUpdate(int i) { if (countShape == 2 && selectShape[i] == 0) { return NONE; } else if (countShape == 2 && selectShape[i] == 1) { return UNCHECK; } else if (selectShape[i] == 0) { return CHECK; } else if (selectShape[i] == 1) { return UNCHECK; } return NONE; } </pre>

수행 과제	Shape 프로젝트에서 기능구현 3단계 - <SubActivity.java>	
	화면 디자인	상세 조건
<ul style="list-style-type: none"> - 6개의 보기 이미지에 각각 이벤트 리스너 달기 - 어떤 이미지를 클릭했는지에 따라서 switch 문으로 분기 - selectUpdate 메소드의 결과에 따라서 <ul style="list-style-type: none"> • CHECK : CHECK 이미지 표시하기 • UNCHECK : 마지막에 추가된 뷰를 삭제하기 • NONE : 아무것도 하지 않기 		
<pre> 52 @Override 53 public void onClick(View view) { 54 55 int returnValue; 56 57 switch (view.getId()) { 58 case R.id.shape01: 59 returnValue = selectUpdate(0); 60 61 62 63 64 65 break; 66 case R.id.shape02: 67 returnValue = selectUpdate(1); 68 69 70 71 72 73 break; 74 case R.id.shape03: 75 returnValue = selectUpdate(2); 76 77 78 79 80 81 break; 82 case R.id.shape04: 83 returnValue = selectUpdate(3); 84 85 86 87 88 89 break; 90 case R.id.shape05: 91 returnValue = selectUpdate(4); 92 93 94 95 96 97 break; 98 case R.id.shape06: 99 returnValue = selectUpdate(5); 100 101 102 103 104 105 break; 106 107 } 108 } </pre>		

수행 과제	Shape 프로젝트에서 기능구현 4단계 - <SubActivity.java> 결과보기 버튼이 클릭이 되었을때
화면 디자인	상세 조건
- 결과보기 버튼이 클릭되었을 때, 결과 배열을 호출한 액티비티로 돌려준다	
<pre> 52 53 54 55 56 57 58 59 60 61 </pre>	<pre> binding.imageButton.setOnClickListener(new View.OnClickListener() { @Override public void onClick(View view) { finish(); } }); </pre>

수행 과제	Shape 프로젝트에서 기능구현 5단계 - <MainActivity.java> 로 다시 돌아왔을때
화면 디자인	상세 조건
- 정답을 표시하기 위한 이미지뷰 객체를 배열로 저장하기 위해 resultImage 생성 - 호출한 액티비티에서 돌려받을 결과값을 저장하기 위해서 selectResult 생성 - 이미지 6개의 배열을 생성 drawableId	
<pre> 22 23 24 25 26 27 28 </pre>	<pre> 4 usages ImageView[] resultImage = new ImageView[2]; 7 usages int[] selectResult = new int[6]; 1 usage int drawableId[] = { R.drawable.shape01, R.drawable.shape02, R.drawable.shape03, R.drawable.shape04, R.drawable.shape05, R.drawable.shape06 }; </pre>

- resultImage 배열에 결과값을 그려줄 ImageView 객체를 저장하기
- launcher 에 registerForActivityResult 구현하기

<pre> 59 60 61 62 63 64 65 66 67 68 69 70 71 </pre>	<pre> resultImage[0] = binding.imageResult01; resultImage[1] = binding.imageResult02; launcher = registerForActivityResult(new ActivityResultContracts.StartActivityForResult(), new ActivityResultCallback<ActivityResult>() { @Override public void onActivityResult(ActivityResult o) { } }); </pre>
---	--

수행 과제	Shape 프로젝트에서 기능구현 6단계 - <MainActivity.java> 로 다시 돌아왔을때	
화면 디자인	상세 조건	
- onActivityResult 콜백메소드 내부 구현하기		
<ul style="list-style-type: none">• 되돌려받은 결과값 저장하기• 해당 이미지를 화면에 그려주기• 정답여부를 표시하기		
62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92	<pre>launcher = registerForActivityResult(new ActivityResultContracts.StartActivityForResult(), new ActivityResultCallback<ActivityResult>() { @Override public void onActivityResult(ActivityResult o) { int num = 0; // 되돌려받은 결과값 저장 <div></div> // 해당 이미지를 화면에 그려주기 for(int i=0; i<selectResult.length; i++){ if(selectResult[i]==1){ <div></div> } } // 정답여부 표시하기 <div></div> } });</pre>	

수행 과제	Shape 프로젝트에서 기능구현 단계 - 코드 리팩토링		
화면 디자인		상세 조건	
<ul style="list-style-type: none">- 게임을 여러번 반복하다보면 이전게임에서 선택한 이미지가 그대로 남아 있어서 한 개만 선택했을 때 이전에 선택했던 이미지가 남아 있는 문제가 발생- resultImage의 Visibility를 Gone 으로 초기화해주는 코드가 필요함. 어디에다 추가할까?			
61			//결과값을 표시하는 이미지부의 Visibility 를 Gone 으로 초기화
62			for(int i=0; i<resultImage.length; i++){
63			resultImage[i].setVisibility(View.GONE);
64			}