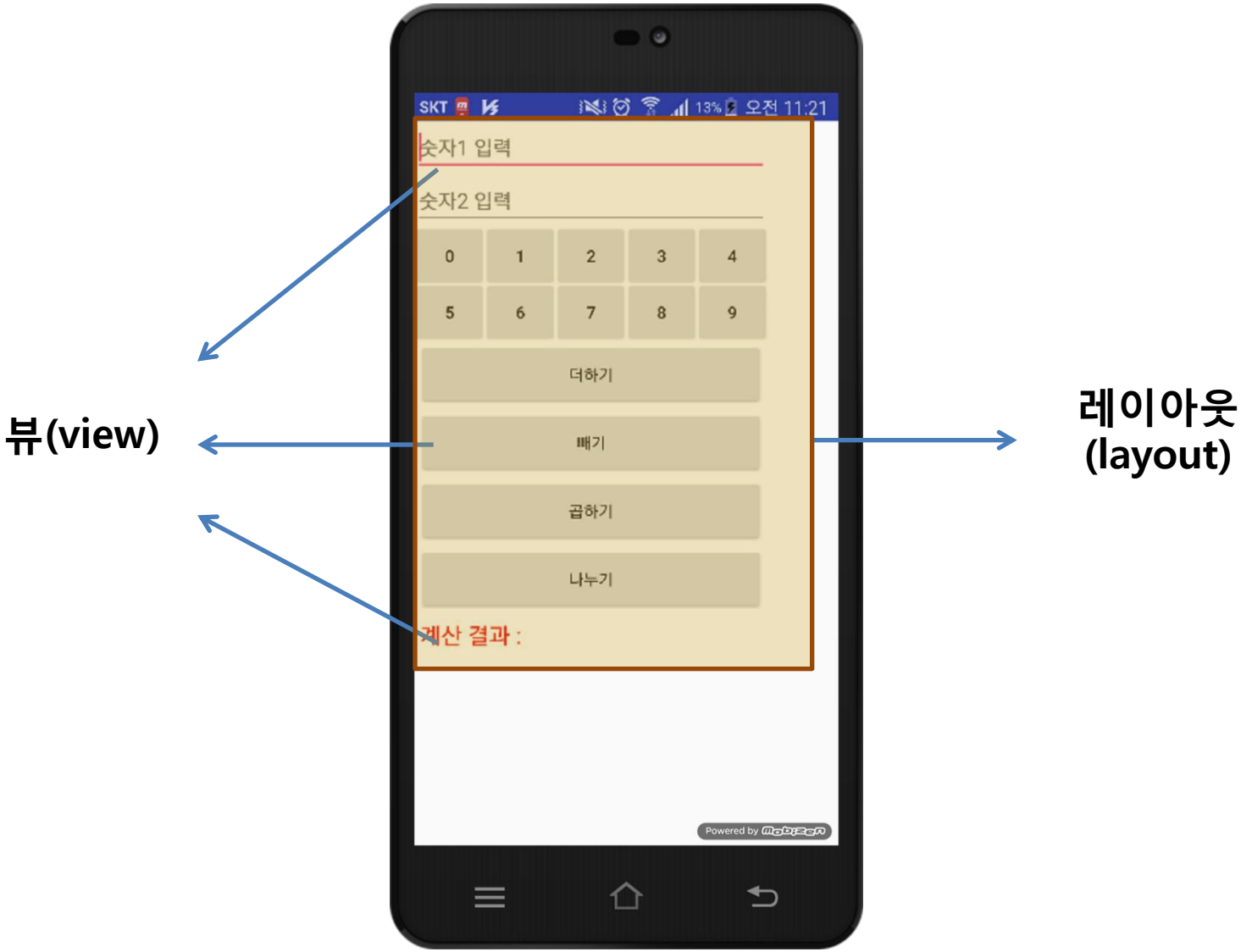
뷰 (VIEW)

뷰의 개념

- 뷰의 계층
  - ✓ 안드로이드 응용 프로그램의 화면을 구성하는 주요 단위인 액티비티는 화면에 직접적으로 보이지 않으며, 액티비티 안의 뷰가 사용자를 대면하는 실체임
  - ✓ 여러 개의 뷰가 모여 하나의 액티비티를 구성하고, 이러한 액티비티가 모여 하나의 응용 프로그램이 됨

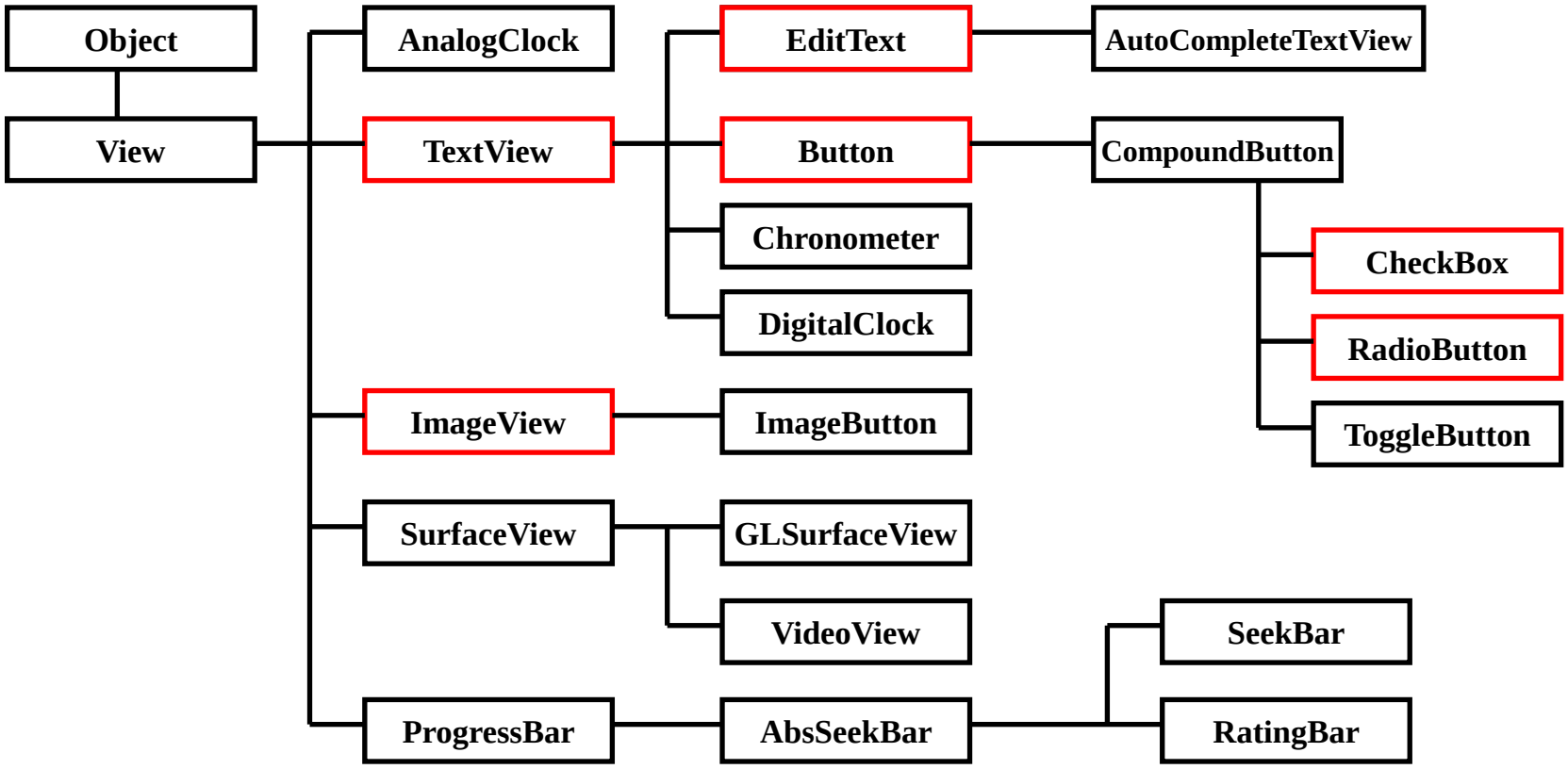
- 뷰
  - ✓ 위젯 : 직접적으로 보이며 사용자 인터페이스를 구성
  - ✓ 뷰 그룹 : 뷰를 담는 컨테이너 역할을 하며, 이 부류의 클래스들을 레이아웃이라 함

뷰의 개념


뷰(view)
레이아웃
(layout)
SKT
13%
오전 11:21
숫자1 입력
숫자2 입력
0
1
2
3
4
5
6
7
8
9
더하기
빼기
곱하기
나누기
계산 결과 :
Powered by

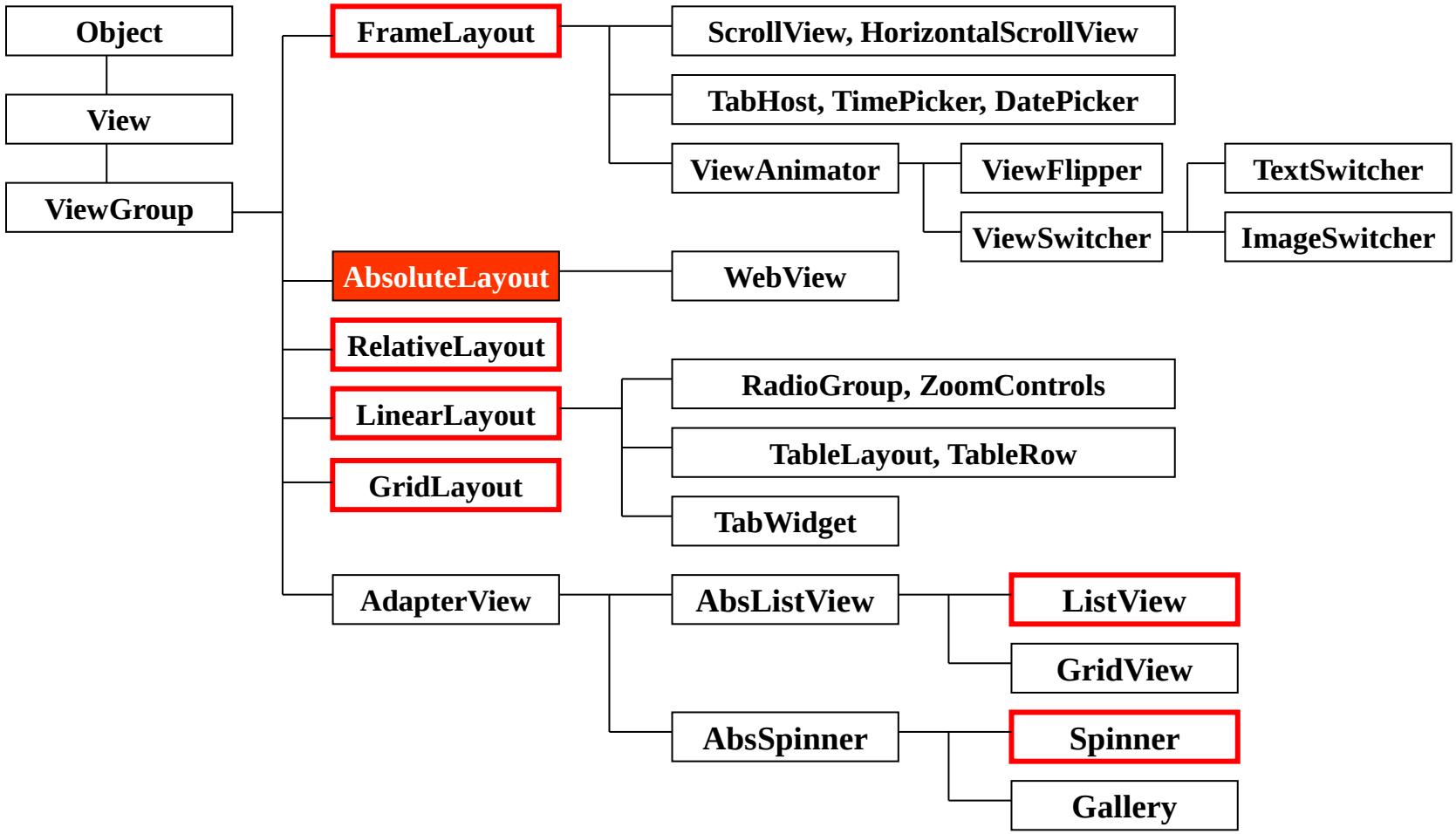
뷰의 상속관계

View로부터 직접 파생되는 모든 클래스가 바로 위젯이며 스스로를 그릴 수 있는 능력을 가짐



뷰의 상속관계

View로부터 파생된 **ViewGroup**의 서브 클래스
다른 뷰들을 자식으로 포함하며 포함된 뷰들을 정렬하는 기능을 가짐



뷰의 XML 속성

위젯과 레이아웃은 **view** 클래스의 속성과 메서드를 상속받는다.

id

layout_width, layout_height

background

padding

visibility

enabled

뷰의 XML 속성 – ID

- **id**

  뷰를 칭하는 이름을 정의하며, 코드나 **XML** 문서에서 뷰를 참조할 때 **id**를 사용하므로 직관적인 이름을 붙이는 것이 좋음

  형식 : **@[+]id/ID**

  - **@ : id**를 리소스**(R.java)**에 정의하거나 참조한다는 뜻이며, 무조건 붙여야 함
  - **+ : ID**를 새로 정의한다는 뜻이며, 참조 시는 생략 가능
  - **id :** 예약어
  - **/** : 뒤에 원하는 이름을 작성하되, **ID**는 고유한 명칭이므로 명령 규칙에 맞아야 하며, 뷰끼리 중복되어서는 안됨

  ex) android:id="@+id/name" : 텍스트 뷰에 **name**이라는 **id**를 부여함.

  XML 문서에 **ID**를 지정하면 이 이름이 **R.java**에 정수형 상수로 정의

  코드에서 뷰를 참조할 시 **findViewById** 메서드 호출, 인수로 참조할 뷰의 **id**를 전달

뷰의 XML 속성 – ID

- **id**

  모든 뷰에 **id**를 의무적으로 지정할 필요는 없으며, 코드에서 참조할 필요 없는 위젯은 보통 **id**를 생략

- **inflation**

  XML 레이아웃에 정의된 정보를 메모리 상에서 객체로 만드는 과정

  프로그램에서 메모리 상에 생성된 객체를 참조하기 위해서 **ID**를 지정

activity_main.xml

```
<Button
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:id="@+id/btn1"
   android:text="Click" />
```

MainActivity.java

```
setContentView(R.layout.activity_main);
Button btn1 =
(Button)findViewById(R.id.btn1);
```

뷰의 XML 속성 – layout_width, layout_height

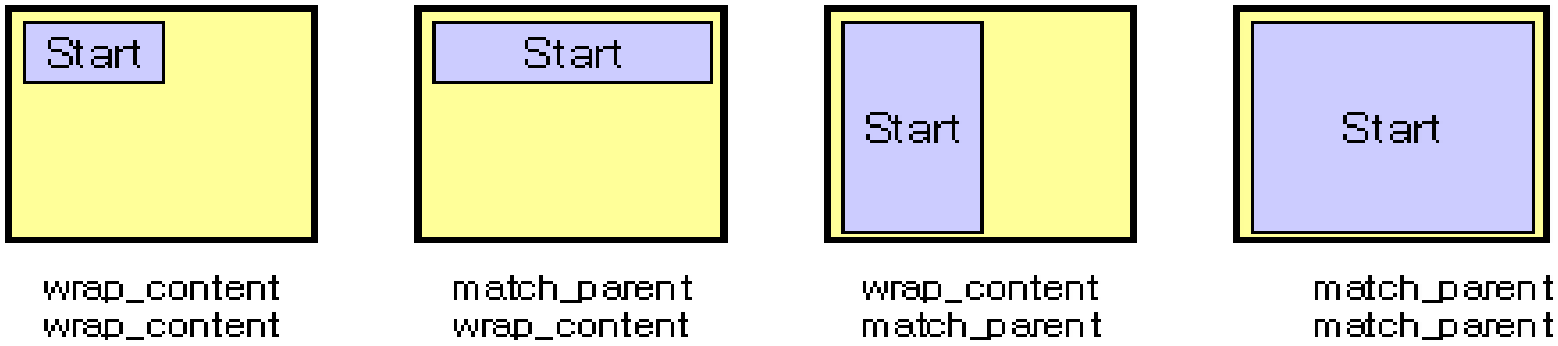
- **layout_width, layout_height**

  뷰의 폭과 높이를 지정하며, 수평, 수직 각 방향에 대해 크기를 지정 가능

  속성값으로 아래의 세 가지 중 하나의 값을 가짐

  - **match_parent(fill_parent) :** 부모의 주어진 크기를 다 채움
  - **wrap_content :** 내용물의 크기만큼만 채움.
  - 정수 크기 : 지정한 크기에 맞춤

  ex) "Start"라는 캡션을 가지는 버튼 배치



뷰의 XML 속성 – background

- **background**
  - ✓ 뷰의 배경을 지정하며, 색상 및 이미지 등의 여러 가지 객체로 지정 가능
  - ✓ 색상 지정 시 네 가지 형식이 적용되며, 배경뿐만 아니라 색상을 지정하는 모든 속성에 적용

    - #RGB

    - #ARGB

    - #RRGGBB

    - #AARRGGBB

    ex) #ff0000 (#f00) : 빨간색, #0000ff (#00f) : 파란색

뷰의 XML 속성 – padding

- **padding**
  - ✓ 뷰와 내용물과의 간격을 지정
  - ✓ **padding** 속성 값을 지정하여 **4**방향에 대한 여백을 조절
  - ✓ 속성값 : **paddingLeft, paddingTop, paddingRight, paddingBottom**

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#00ff00"
    android:padding="30dp"
    android:text="텍스트1" />
```

뷰의 XML 속성 – layout_margin

- layout_margin
  - ✓ 뷰와 뷰 사이에 간격을 두고 싶다면 layout_margin 속성을 사용
  - ✓ 속성값 : layout_marginLeft, layout_marginTop, layout_marginRight, layout_marginBottom

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#00ff00"
    android:layout_margin="30dp"
    android:text="텍스트1" />
```

뷰의 XML 속성 – visibility

- **visibility**
  - ✓ 뷰의 표시 여부를 지정
  - ✓ 속성값
    - **visible :** 보이는 상태임.
    - **invisible :** 숨겨진 상태, 자리는 차지
    - **gone :** 숨겨진 상태, 자리도 차지하지 않음

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#00ff00"
    android:layout_margin="30dp"
    android:text="텍스트1" />
```

뷰의 XML 속성 – enabled

- **enabled**
  - ✓ 뷰의 동작 여부를 지정
  - ✓ 속성값

    - true, false

```
<Button
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:id="@+id/btn2"
   android:enabled="false"
   android:text="버튼2" />
```

Widget – TextView

- ✓ **text** : 텍스트 뷰의 가장 중요한 속성으로 출력할 문자열을 지정
  : 리터널 및 리소스로 대입
  : 리소스에 대한 레퍼런스로 지정한다. 보통 **string.xml**에 문자열을 정의해 놓고 **@string/id** 식으로 지정한다.
- ✓ **textColor** : 글자의 색상을 지정, #RRGGBB나 #AARRGGBB 형식
- ✓ **textSize** : 글자의 크기를 dp, px, in, mm, sp 단위로 지정
- ✓ **typeface** : 글자의 글꼴을 지정
  : normal(기본), sans, serif, monospace
- ✓ **textStyle** : 글자의 스타일을 지정
  : normal(기본), bold, italic, bold|italic, normal|italic
- ✓ **singleLine** : 글이 길어 줄이 넘어갈 경우 강제로 한 줄만 출력하고 문자열의 맨 뒤에 "..."을 표시
  : true, false(기본)

Widget – EditText

View ← TextView 서브 클래스, 고유의 속성은 따로 가지지 않음

값을 입력받은 후 해당 값을 Java 코드에 가져와서 사용하는 용도로 많이 사용

XML

```
<EditText
    android:id="@+id/msgInput"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

JAVA

```
1. 변수 선언
private EditText msgInput;
2. 변수에 에디트텍스트 참조 얻기
msgInput   = (EditText)findViewById(R.id.msgInput);
3. 에디트텍스트에 입력된 값 가져오기
String msg = msgInput.getText().toString();
```

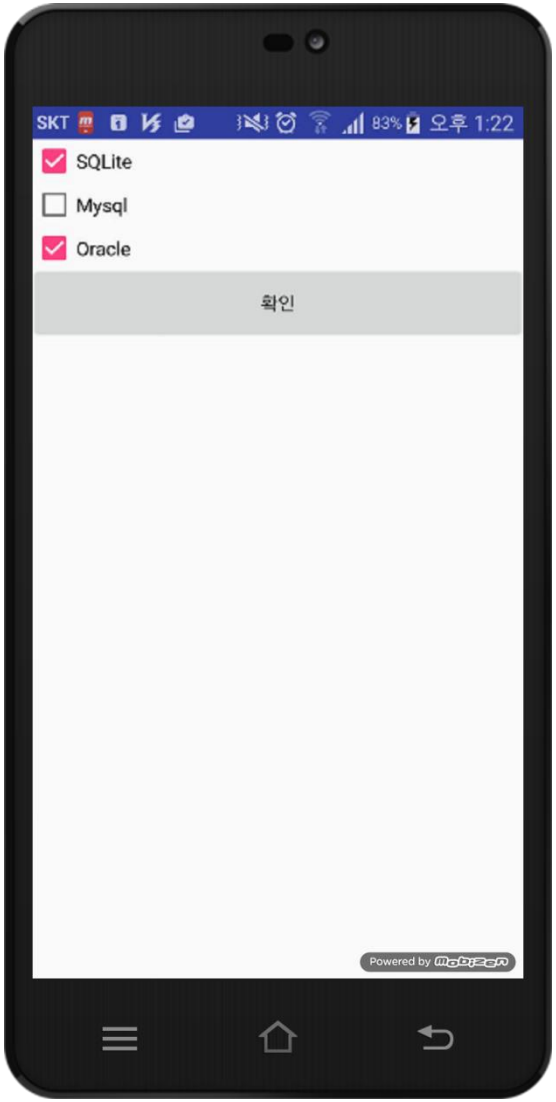
Widget – CompundButton

- ✓ Button 클래스의 하위 클래스
- ✓ 체크박스(CheckBox), 라디오버튼(RadioButton), 스위치(Switch), 토글버튼(ToggleButton)의 상위 클래스

CompundButton 계층도

```
java.lang.Object
    └ android.view.View
        └ android.widget.TextView
            └ android.widget.Button
                └ android.widget.CompundButton
                    └ android.widget.CheckBox
                    └ android.widget.RadioButton
                    └ android.widget.Switch
                    └ android.widget.ToggleButton
```

Widget – CheckBox

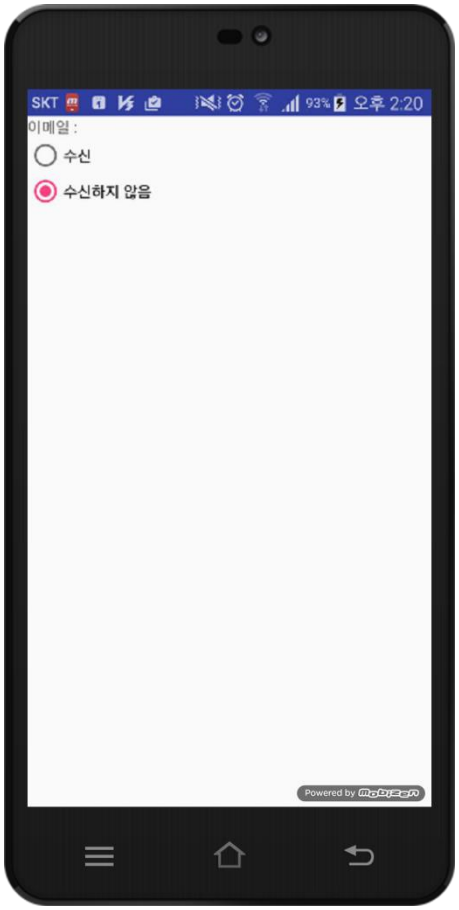



```
<CheckBox  android:id="@+id/db1"
           ......
           android:checked="true"
           android:text="SQLite" />
```

```
CheckBox ck1 = (CheckBox)findViewById(R.id.db1);
ck1.setOnCheckedChangeListener(new
                CompoundButton.OnCheckedChangeListener() {
   @Override
   public void onCheckedChanged(
                CompoundButton compoundButton, boolean b)  {
      String msg = compoundButton.getText() + " " +
                (b ? "체크됨" : "체크해제됨");
      Toast.makeText(
           getApplicationContext(), msg, Toast.LENGTH_SHORT
      ).show();
   }
});
```

Widget – Radio, RadioGroup

- 라디오버튼 : 여러 개 중 하나만 선택해야 하는 경우에 사용
- 라디오그룹 : 라디오버튼만 여러 개 나열하면 클릭하는 것마다 모두 중복 선택이 되므로 라디오그룹과 함께 사용해야 함



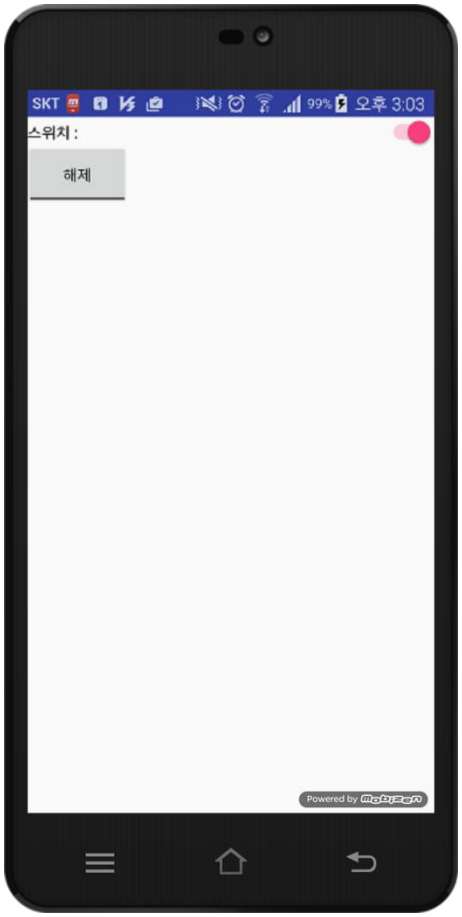

```
<RadioGroup   android:id="@+id/emailGroup"
              생략….
              android:orientation="vertical">
   <RadioButton
      android1:id="@+id/email1"
      생략…
      android:checked="true"
      android:text="수신" />
   <RadioButton
      android1:id="@+id/email2"
      생략…
      android:text="수신하지  않음" />
</RadioGroup>
```

Widget – Radio, RadioGroup

```
private RadioGroup emailGroup;
emailGroup = (RadioGroup)findViewById(R.id.emailGroup);
emailGroup.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
    public void onCheckedChanged(RadioGroup group, int checkedId) {
        String msg = "";
        switch (checkedId) {
            case R.id.email1:
                    msg = "수신  선택";
                    break;
            case R.id.email2:
                    msg = "수신하지  않음  선택";
                    break;
        }
        Toast.makeText( getApplicationContext(), msg, Toast.LENGTH_SHORT ).show();
    }
});
```

Widget – Switch, ToggleButton

➢ 설정시 많이 사용되는 버튼 형태 : on, off 형태의 선택만 가능




```
<Switch
    android:id="@+id/switchBtn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:checked="true"
    android:text="스위치 : " />

<ToggleButton
    android:id="@+id/toggleBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:checked="false"
    android:text="토글버튼 : " />
```

Widget – Switch, ToggleButton

```
private Switch switchBtn;

switchBtn = (Switch)findViewById(R.id.switchBtn);

switchBtn.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener()

{

    @Override

    public void onCheckedChanged(CompoundButton compoundButton, boolean isCheck) {

        Toast.makeText(

            getApplicationContext(),    isCheck ? "체크  상태" : "체크해제  상태",

            Toast.LENGTH_SHORT

        ).show();

    }

});
```

Widget – ImageView

- 아이콘이나 비트맵을 출력하는 위젯

src

출력할 이미지를 지정하는 가장 중요한 속성
주로 리소스에 이미지를 복사해 두고 **@drawable/ID** 형식으로 이미지를 출력하는 방법을 사용

maxHeight, maxWidth

이미지가 출력될 최대 크기를 지정

adjustViewbounds

이미지의 종횡비를 맞추기 위해 이미지 뷰의 크기를 적당히 조정할 것인가를 지정
true : 비율 유지, **false :** 비율 무시

tint

이미지에 색조를 입힌다. 지정한 색상이 이미지 위에 살짝 덮여 출력

scaleType

이미지의 원래 크기와 다르게 출력할 때 적용할 확대, 축소 방식을 지정