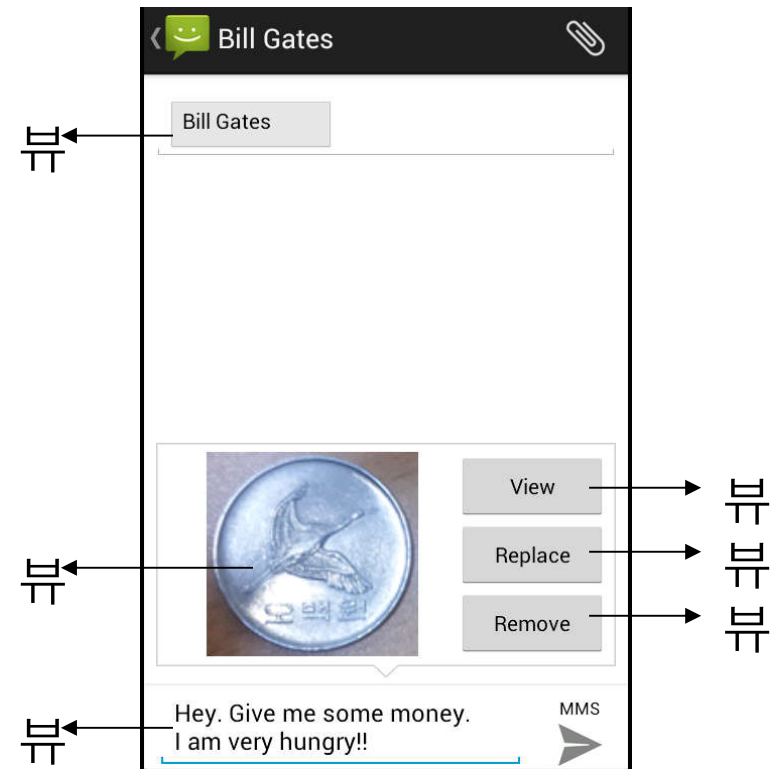




뷰와 뷰그룹

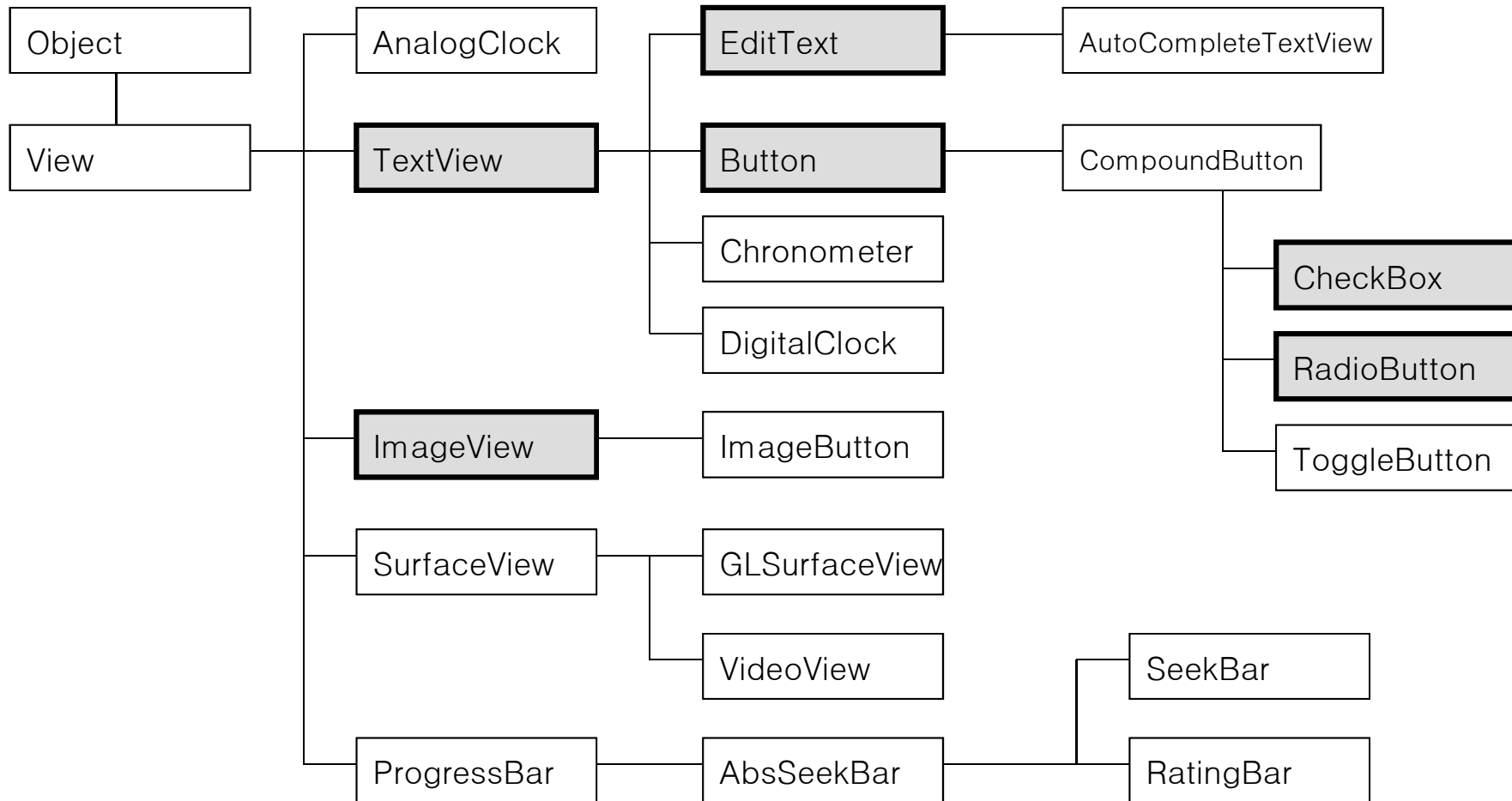
- 액티비티는 껍데기이며 사용자를 실제 대면하는 것은 뷰이다.
- 뷰는 다음 두 가지로 분류된다.
 - 위젯 : 직접 보이며 사용자와 상호 작용하는 UI를 구성한다. 컨트롤이라고도 부른다.
 - 뷰그룹 : 뷰를 담는 컨테이너이다. **모든 레이아웃은 뷰 그룹이다.**
- 뷰
 - View 클래스는 모든 뷰의 부모 클래스
 - View 클래스가 가지고 있는 필드나 메소드는 모든 뷰에서 공통적으로 사용 가능
- 뷰그룹
 - 뷰의 자식 클래스로 다른 뷰를 포함하며 정렬하는 기능을 가진다.



액티비티

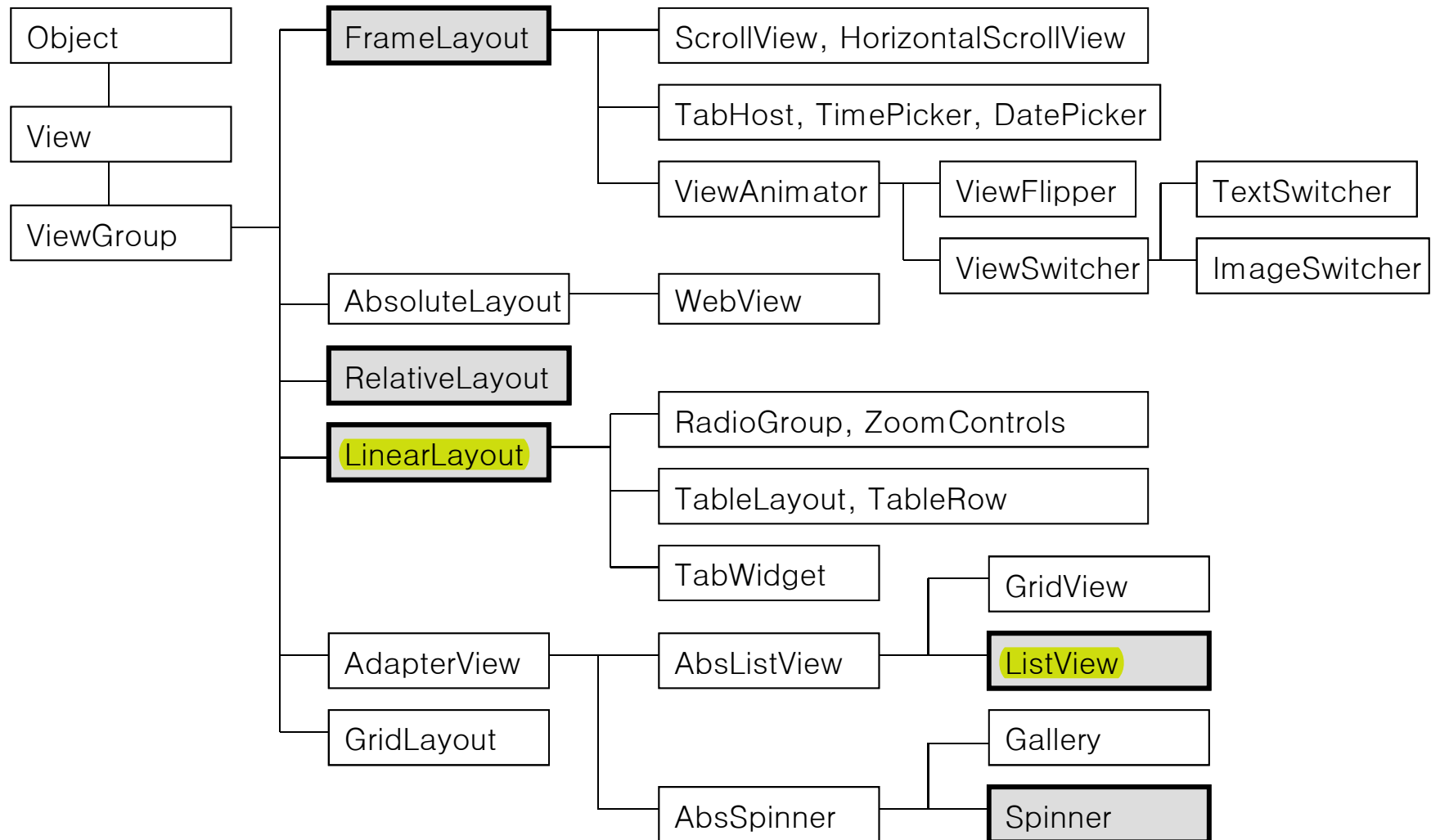
뷰와 뷰그룹

■ 뷰의 계층 구조 *object*



뷰와 뷰그룹

■ 뷰 그룹 계층 구조



뷰와 뷰그룹

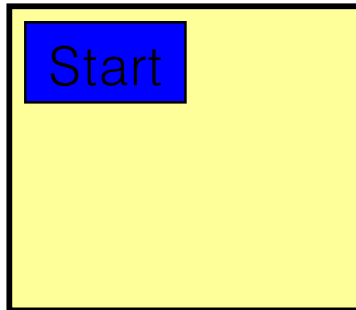
■ 뷰 속성

+

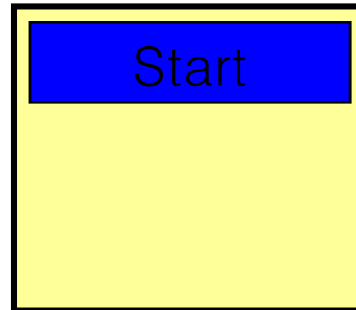
가

- id : 뷰를 칭하는 이름. @[+]id/ID 형식으로 붙인다. 코드에서 id로 참조
- layout_width, layout_height : 뷰의 크기를 지정한다.

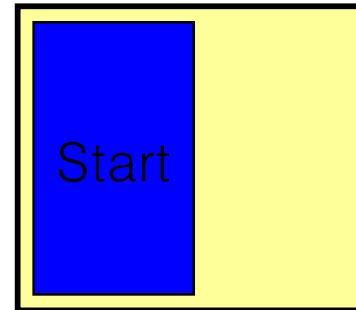
속성값	설명
match_parent(fill_parent)	부모의 주어진 크기를 다 채운다.
wrap_content	내용물의 크기만큼만 채운다.
상수 크기	지정한 크기에 맞춘다.



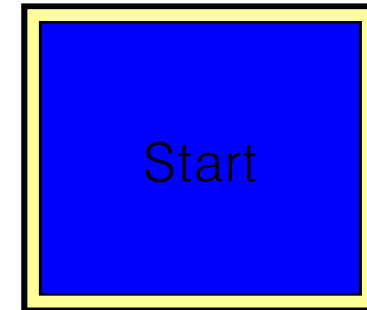
wrap_content
wrap_content



match_parent
wrap_content



wrap_content
match_parent



match_parent
match_parent

뷰와 뷰그룹

■ 뷰 속성

- 상수 크기를 지정할 때는 숫자 다음에 단위를 붙인다. 상수와 단위는 반드시 붙여 쓴다.
- 절대 길이보다는 가급적이면 dp나 sp 같은 논리 단위를 쓰는 것이 유리하다.
- 1dp는 160dpi일 때는 1픽셀로 정의되며 dp가 늘어나면 같이 늘어난다.

단위	설명
px	픽셀
in	인치
mm	밀리미터
pt	포인트
dp(또는 dip)	밀도에 독립적인 단위
sp(또는 sip)	폰트 가변 크기

뷰와 뷰그룹

■ 뷰 속성

- background : 뷰의 배경을 지정. 색상 또는 드로블 등으로 지정 가능.
 - #RGB
 - #ARGB
 - #RRGGBB
 - #AARRGGBB
- padding : 뷰와 내용 사이의 간격
- visibility : 뷰의 보임 여부를 지정

속성값	설명
visible	보이는 상태이다.
invisible	숨겨진 상태이되 자리는 차지한다.
gone	숨겨지며 자리도 차지하지 않는다.

- clickable, longClickable
- focusable

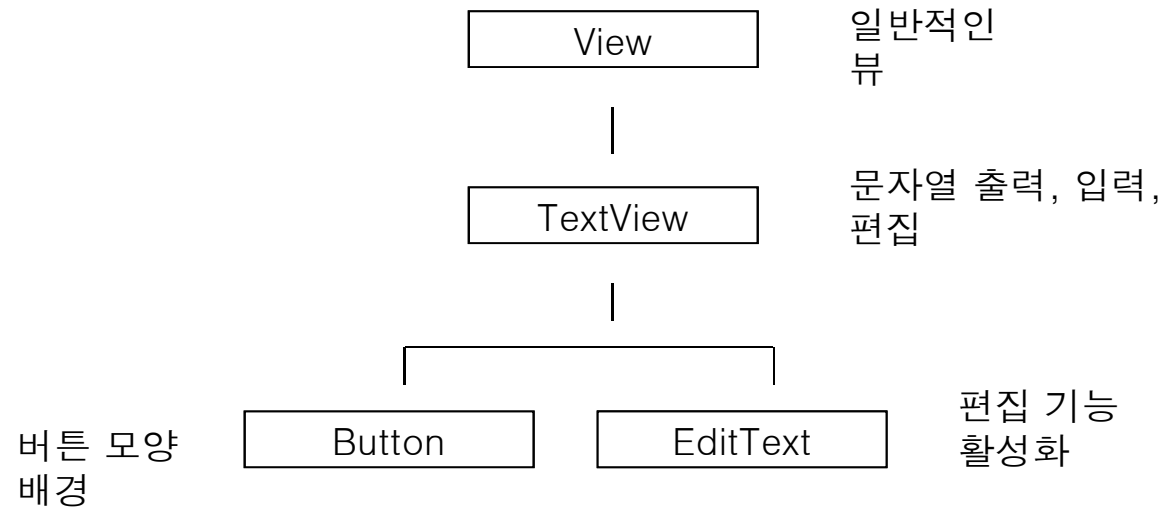
*

-

기본 위젯

■ 텍스트뷰

- 문자열 포매팅, 출력, 입력, 편집에 관련된 모든 기능을 제공하되 필요한 기능만 노출한다.
- 가장 기본 위젯이며 TextView로부터 버튼, 에디트텍스트가 파생된다.



기본 위젯

■ 텍스트뷰

- 문자열을 보여주는 위젯. 가장 흔하게 사용된다.
- text : 출력할 문자열. 디폴트는 빈 문자열이므로 반드시 지정해야 한다.

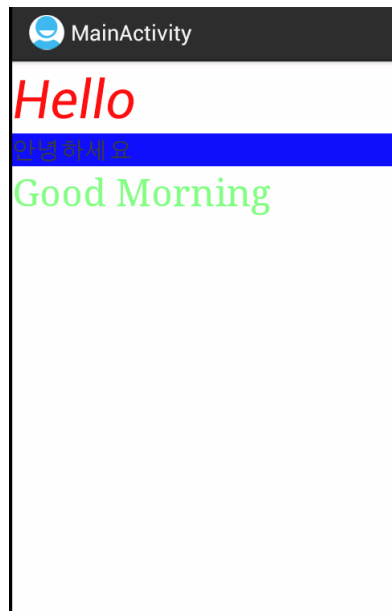
형식	설명
"문자열"	겹 따옴표로 문자열을 싸서 바로 대입한다. \ 문자가 들어가면 이스케이프된다. \n은 개행이며 \uxxxxx는 유니코드 문자이다.
@[패키지:]type:name	리소스에 대한 레퍼런스로 지정한다. 보통 strings.xml에 문자열을 정의해 놓고 @string/id 식으로 지정한다.
?[패키지:]type:name	테마 속성으로 지정한다.

- textColor : 문자열의 색상
- textSize : 문자열의 크기. sp 단위 사용
- textStyle : 모양. normal, bold, italic의 조합
- typeface : 글꼴의 모양.
- singleLine : 자동 개행 금지.
- width, height : 텍스트뷰가 대부분 레이아웃의 자식으로 배치되기 때문에 거의 사용되지 않음.
대신 layout_??? 속성으로 사용

기본 위젯

■ 텍스트뷰

- 예제: 레이아웃, [res/values/strings.xml](#), [액티비티](#)



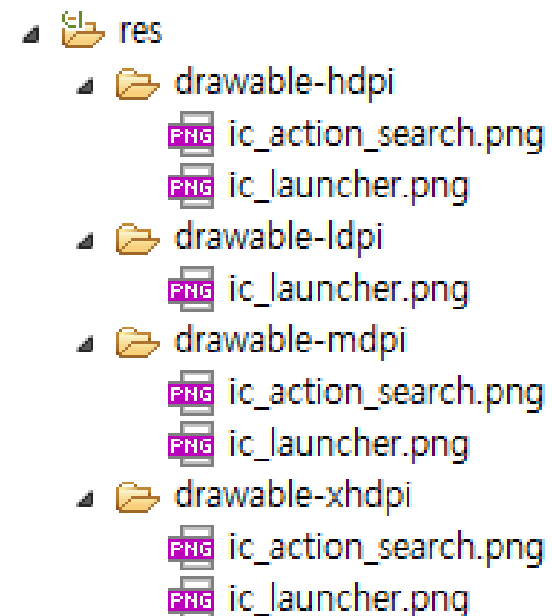
```
21 <TextView
22     android:layout_width="match_parent"
23     android:layout_height="wrap_content"
24     android:text="Good Morning"
25     android:textColor="#8000ff00"
26     android:textSize="5mm"
27     android:typeface="serif"
28 />
29 </LinearLayout>
```

```
1 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2     android:orientation="vertical"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5 >
6 <TextView
7     android:layout_width="match_parent"
8     android:layout_height="wrap_content"
9     android:text="@string/insa"
10    android:textColor="#ff0000"
11    android:textSize="20pt"
12    android:textStyle="italic"
13 />
14 <TextView
15     android:layout_width="match_parent"
16     android:layout_height="wrap_content"
17     android:text="@string/anyoung"
18     android:textSize="20sp"
19     android:background="#0000ff"
20 />
```

기본 위젯

■ 이미지뷰

- 아이콘이나 비트맵을 출력하는 위젯
- 속성
 - src : 출력할 비트맵. @drawable/ID 형식. 반드시 지정해야 한다.
 - maxHeight, maxWidth : 최대 크기
 - adjustViewBounds : 종횡비 유지를 위해 크기 조정
 - cropToPadding : 여백을 위해 이미지 절단
 - tint : 위쪽에 덮히는 색조
 - scaleType : 확대, 축소 방식
- drawable 폴더에 밀도별로 이미지 준비. 장치 독립성 확보를 위해 여러 밀도의 이미지가 필요하다.
- mipmap 폴더는 프로그램의 아이콘을 저장하는 폴더이며 밀도별로 나누어져 있다.



기본 위젯

■ 이미지뷰

- 예제 : [레이아웃](#), [액티비티](#)
- 이미지를 res/drawable 폴더에 복사
- 레이아웃에 ImageView 배치하고 src에 출력할 리소스 지정
- 이미지 리소스의 조건
 - 명칭 규칙에 적합해야 한다.
 - 소문자만 가능하다.
 - 같은 이름의 다른 확장자 불가
- 파일 형태의 모든 리소스에 적용된다.



기본 위젯

■ 버튼 및 에디트텍스트

- Button : 클릭으로 명령을 입력받는 위젯
- EditText : 문자열을 입력받는 위젯
- 모든 속성은 TextView로부터 상속받는다.
- 텍스트 뷰, 이미지 뷰와는 달리 상호작용을 하므로 이벤트를 처리해야 한다.

■ 에디트텍스트

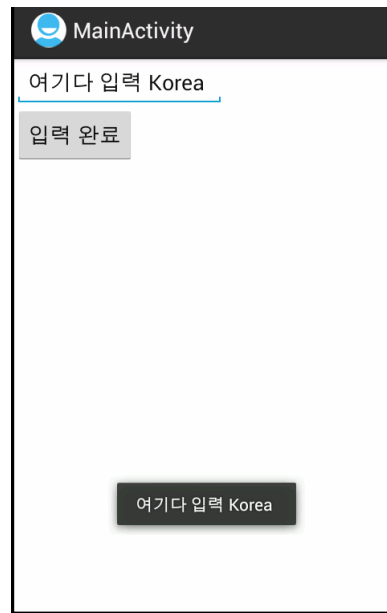
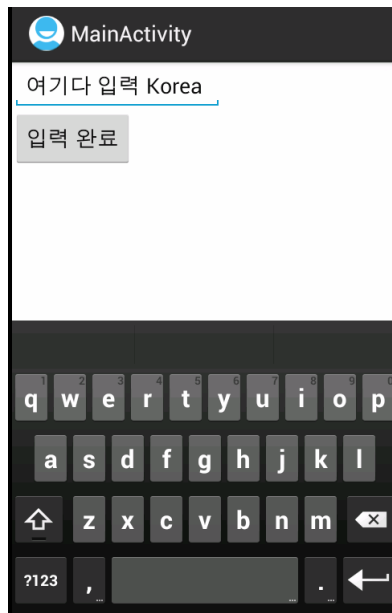
- inputType 속성 : 입력 종류 선택

inputType	설명
none	편집할 수 없는 문자열
text	단순 문자열
textMultiLine	여러 줄 입력 가능한 문자열
textEmailAddress	email 주소
textPassword	비밀 번호
number	숫자
numberSigned	숫자와 부호
numberDecimal	숫자와 부호와 소수점
phone	전화번호
datetime	날짜와 시간

기본 위젯

■ 버튼 및 에디트텍스트

- 예제 : [레이아웃](#), [액티비티](#)



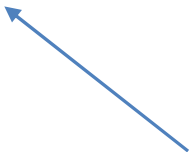
기본 위젯

■ 버튼

- XML의 onClick 속성
- 클릭은 매우 자주 사용되는 이벤트
- SDK 1.6부터 액티비티에 클릭을 위한 리스너를 포함시켜 사용 방법을 단순화
- 안드로이드는 XML 문서에 android.onClick 속성을 사용하여 클릭에 대응하는 핸들러 메소드를 지정하고, 코드 내부에 핸들러 메소드를 구현하는 특별한 방법을 제공
- 클릭을 위한 핸들러 메소드는 public으로 선언되어야 하며, 또한 클릭될 위젯의 레퍼런스인 하나의 변수 View를 포함

```
public void _____ (View v) {  
  
}
```

onClick 속성 값으로 제공한 메서드 이름



기본 위젯

■ 토스트

● 의미

- 설정 변경 혹은 이벤트 등이 발생했을 때 유용
- 활성 액티비티의 포커스를 뺏지 않고, 잠시 표시되었다가 사라짐
- 다른 방법인 Notification에 비하여 매우 간단

● 사용 방법

- `makeText()` 메소드로 인스턴스를 생성 후 `show()` 메소드를 사용하여 화면에 표시
- API

`public static Toast makeText (Context context, int resId, int duration)`

`public static Toast makeText (Context context, CharSequence text, int duration)`

- 토스트를 표시할 시간

`Toast.LENGTH_SHORT, Toast.LENGTH_LONG`

기본 위젯

■ 토스트 + onClick 속성

- 예제 : [레이아웃](#), [액티비티](#)

