Button (addroid.widget.Button)

● 버든이 눌렸을 때 일어나는 이벤트 처리: 익명 클래스를 사용하는 방법

```
public class MyActivity extends Activity {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.content_layout_id);

        // 익명 클래스 외부에 있는 변수를 사용하려면, 변수를 선언할 때 반드시 final 키워드를
        final Button button = findViewById(R.id.button1);

        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Code here executes on main thread after user presses button
            }
        });
    }
}
```

위의 코드는 이벤트 발생 시 호출할 메서드를 구현한 리스너 객체를 생성하여 Button에 지정해 주어 결과적으로 시스템이 사용자가 버든을 누르면 이벤트를 감지하고 onClick(View)에 작성한 코드를 실행한다.
→ 리스너(Listener): 이벤트 발생 여부를 감지하는 객체로 View 클래스에 정의돼 있는 인터페이스로 이벤트 발생 시에 호출될 추상 메서드도 이 인터페이스 안에 정의돼 있다.

1) 사용자와 상호 작용을 하게 될 Button 객체: final Button button = findViewById(R.id.button1);
2) Button의 클릭 이벤트를 감지하는 Listener 객체: new View.OnClickListener() - 이벤트 핸들러라고도 부른다.
3) 이벤트 감지 이후 로직의 처리 등을 위해 실행되는 Callback 함수: public void onClick(View v) {}
4) 버든과 리스너 연결: button.setOnClickListener(new View.OnClickListener())

· View.onClickLinstener 인터페이스를 구현한 인스턴스를 생성 시에, 콜백으로 처리할 인터페이스의 추상 메서드 onClick()도 구현한다.
→ 콜백 메서드: 개발자가 해당 함수의 사용을 명령하는 것이 아닌, 이벤트/메시지/알람을 받았을 때 자동으로 실행되는 메서드로 대부분 on으로 시작하는 함수들이 콜백 메서드다.
· 버든 클래스의 setOnClickListener()함수는 OnClickLinstener를 구현한 인스턴스를 매개값으로 받아 이벤트를 처리할 버든에 리스너를 연결해 준다. 그 결과, 시스템은 사용자가 버든을 누르면 이벤트를 감지해 onClick(View)에 작성한 코드를 실행한다.

EditText

● getText(): public Editable getText (). 텍스트뷰가 보여주는 텍스트를 리턴한다. 여기에 .toString()을 해 주면 String으로 리턴한다.

Toast

작은 팝업으로 작업에 관한 간단한 피드백을 제공한다. 메시지에 필요한 공간만 차지하며, 진행 중인 활동은 그대로 표시되고 상호작용도 유지된다. 시간이 초과하면 자동으로 사라진다. 앱이 Android 12(API 수준 31) 이상을 타겟팅한다면 토스트 메시지는 텍스트 두 줄로 제한되고 텍스트 옆에 애플리케이션 아이콘이 표시된다. 이 텍스트의 줄 길이는 화면 크기에 따라 다르므로 텍스트를 최대한 짧게 만드는 것이 좋다.

● makeText(android.content.Context, java.lang.CharSequence, int): 토스트를 만드는 메서드. 매개변수로 애플리케이션 Context, 사용자에게 표시되어야 하는 텍스트, 그리고 토스트 메시지가 화면에 남아 있어야 하는 시간을 받으며, Toast객체를 반환한다. 끝에 .show() 메서드를 호출하면토스트 메시지를 표시한다.

- getApplicationContext(): 에플리케이션의 콘텍스트를 반환한다.
- Toast.LENGTH_LONG: 비교적 길게 토스트를 화면에 남아 있게 한다. (짧게 - LENGTH_SHORT)

Activity

● close(): 엑티비티를 닫는다

Interface

● TextWatcher 인터페이스: 텍스트의 변화를 감지하는 인터페이스로, 인스턴스로 생성 (구현) 시에 onTextChanged(), beforeTextChanged(), 그리고 afterTextChanged()메서드를 모두 구현해 주어야 한다.

▶ beforeTextChanged(CharSequence s, int start, int count, int after): 문자열(CharSequence) 안에서 start에서 시작하는 count만큼의 글자들이 새로운 길이(after)의 텍스트로 변경되려고 한다는 내용을 전달하기 위해 호출되는 메서드. 입력하기 전에 조치할 내용이 있으면 사용한다.

▶ onTextChanged(CharSequence s, int start, int before, int count): 문자열(CharSequence)안에서 start에서 시작하는 count만큼의 글자들이 before길이를 가진 예전 글자들을 대체했다는 내용을 전달하기 위해 호출된 메서드. 텍스트가 변화할 때마다 count의 숫자가 바로 변한다. 입력란에 변화가 있을 때 사용한다 (ex. 문자를 입력할 때마다 바이트수를 표시하는 텍스트에 변화주기).

▶ afterTextChanged(Editable s): s 내의 어딘가에서 텍스트가 변경됐음을 알리기 위해 호출된다. 입력이 끝났을 때 조치할 내용이 있으면 사용한다(ex. 특정 길이를 넘어가면 문자를 지워주기).

TextView

● addTextChangedListener(TextWatcher watcher): 텍스트뷰에 이벤트 리스터를 달아준다. 매개 변수로는 TextWatcher의 인스턴스가 들어간다.

● setText(): 뷰가 보여줄 텍스트를 지정해 준다.