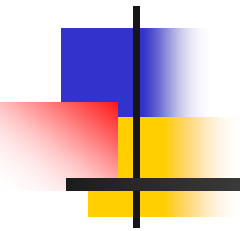
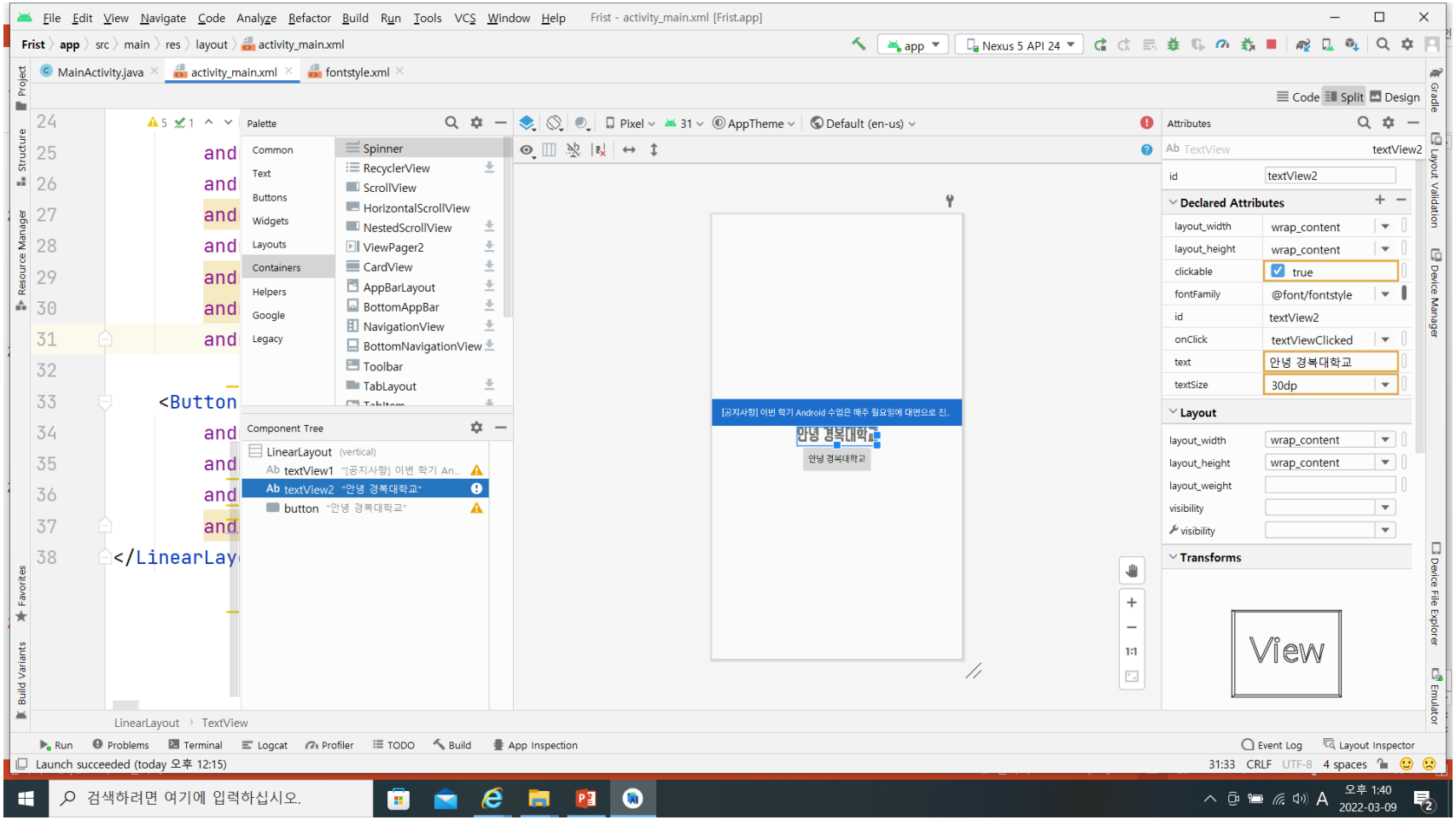


Android 사용자 인터페이스

배 희호 교수
경복대학교
스마트IT과

XML(Layout) 작성하기



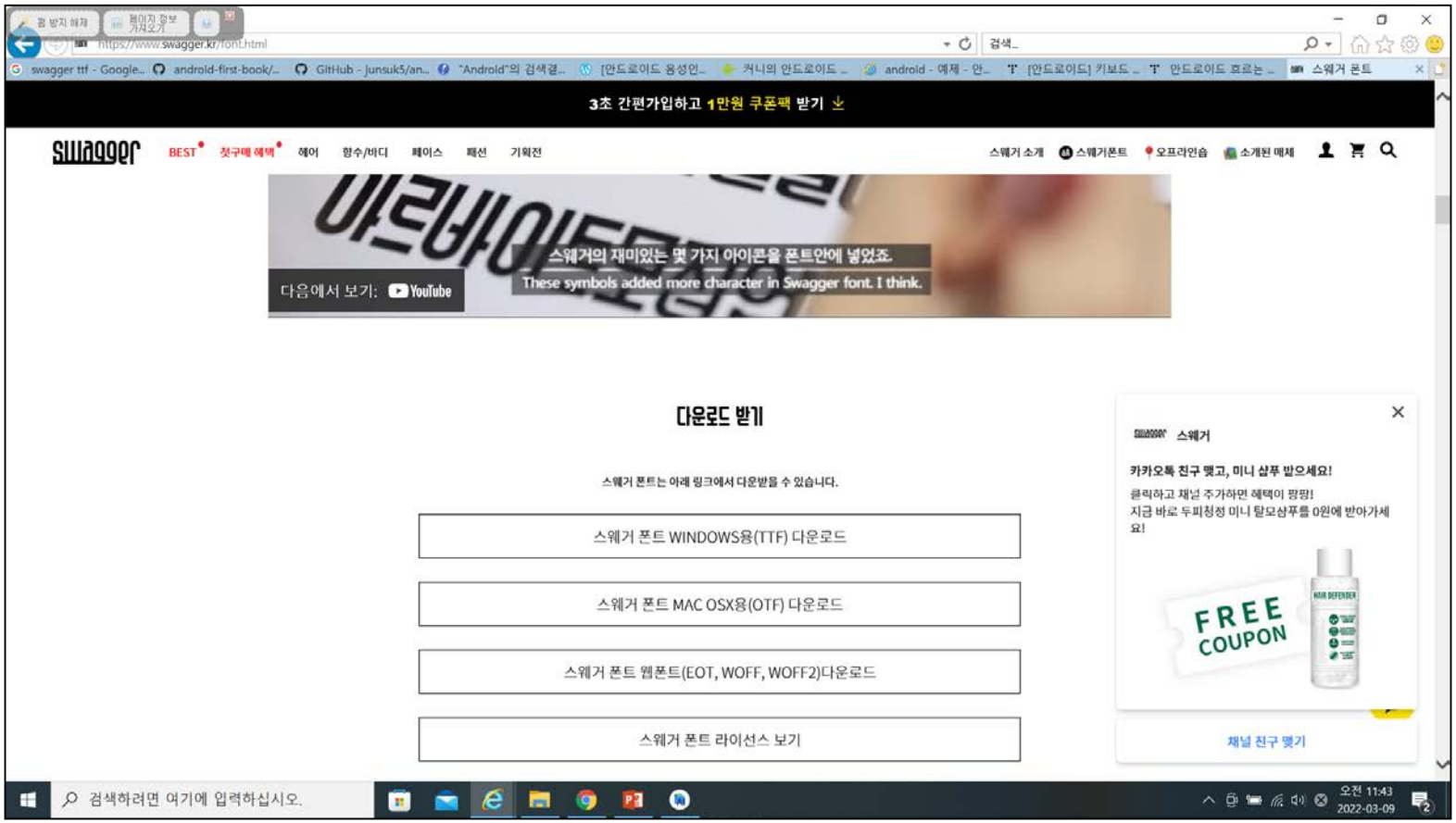
Futuristic Innovator
KYUNGBOK UNIVERSITY

XML(Layout) 작성하기

- Palette(팔레트)
  - 우리가 그림을 그릴 때 Palette에 있는 물감을 이용해서 그림을 그리죠?
  - 똑같습니다. 여기에 물감 대신 여러 재료들이 있음 (버튼, 스크롤, 텍스트 등)
  - 여기에 있는 재료들을 이용해서 프로그램 작성
- Component Tree (구성 요소 관계도)
  - 탐색기처럼 어떤 요소안에 무엇이 있고, 그 안에 또 뭐가 있고를 보여주는 창 (Tree 형태)
  - 화면을 구성하는 요소들의 관계를 보여줌
- 작업 창
  - 사용자가 만들고 있는 App의 화면을 미리보기로 보여주는 창
- Attribute(속성)
  - 각각 요소들의 속성을 설정할 수 있는 창

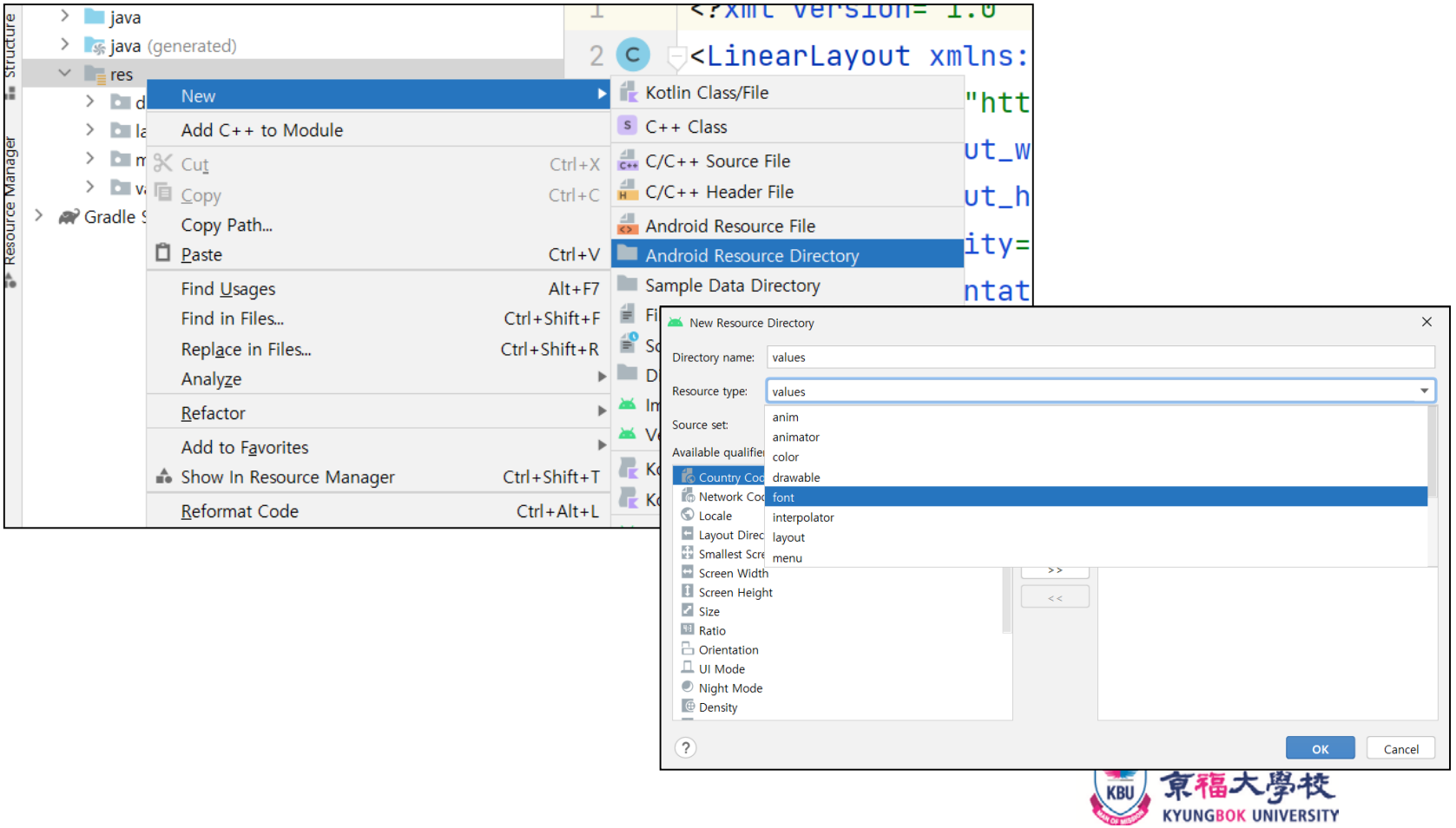
TextView font 변경하기

- Swagger 폰트 다운받기



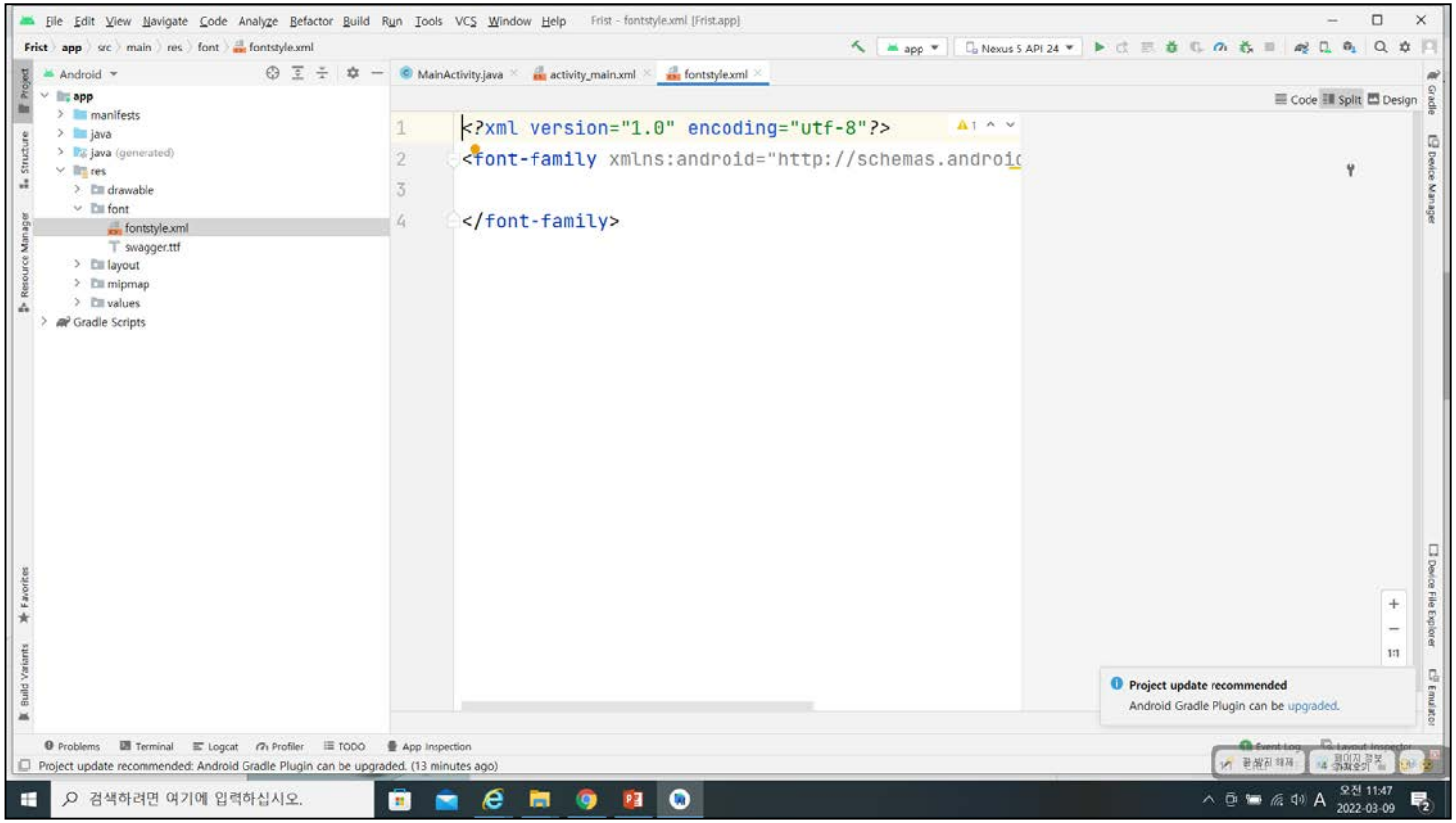
TextView font 변경하기

- font 폴더 생성
  - res 폴더 선택 – 마우스 오른쪽 버튼 클릭 – New – Android Resource Directory



TextView font 변경하기

- 원하는 font 파일을 font 폴더에 넣고 fontstyle.xml 파일을 생성
  - 파일 이름은 영문 소문자로 작성



TextView font 변경하기

- fontstyle.xml 코드 작성

```
<?xml version="1.0" encoding="utf-8"?>
<font-family xmlns:android="http://schemas.android.com/apk/res/android">
    <font
        android:font="@font/swagger"
        android:fontStyle="normal"
        android:fontWeight="400" />
</font-family>
```

- android:font="@font/swagger"
  - font 폴더에 넣어뒀던 폰트 파일의 경로를 입력해주면 됨
  - 확장자명은 쓰지 않아도 됨
- android:fontStyle="normal"
  - normal = 기본 서체
  - italic = 옆으로 기울어진 서체
- android:fontWeight="400"
  - 폰트의 굵기 (400이 기본값)
  - 400보다 작으면 글씨체가 더 얇게 변하고 크게 설정하면 굵게 나옴

TextView font 변경하기

- xml에서 TextView에 fontFamily 적용
  - font 폴더의 fontstyle.xml 경로를 지정해주면 해당 fontstyle이 적용

```
<TextView
   android:fontFamily="@font/fontstyle“
   ……/>
```

띠 Banner(Ticker) 만들기

- Android의 TextView에서는 ellipsize라는 말 줄임 효과 속성으로 Text의 앞, 중간, 뒷 부분을 줄여주는 start, middle, end 값이 있지만 특이하게 marquee라는 속성값도 있음
- 이 값으로 TextView의 전체 글자를 흐르게 할 수 있음

```
<TextView
   android:id="@+id/textView"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:ellipsize="marquee"
   android:focusable="true"
   android:paddingHorizontal="16dp"
   android:paddingVertical="12dp"
   android:singleLine="true"
   android:textColor="@android:color/white"
   android:text="[공지사항] 이번 학기 Android 수업은 매주 월요일에
        대면으로 진행됩니다. 자세한 내용은 홈페이지를 참조해주세요."
   android:background="#1D6ECC"/>
```

띠 Banner(Ticker) 만들기

- MainActivity에서 TextView를 selected 처리

```
TextView textView = findViewById(R.id.textView);
textView.setSelected(true);
```

깜빡이는 Text Animation 만들기

- AlphaAnimation 클래스를 사용하면 간단하게 Text가 깜빡이는 Animation을 만들 수 있음

```
Animation anim = new AlphaAnimation(0.0f,1.0f);
anim.setDuration(100);
anim.setStartOffset(20);
anim.setRepeatMode(Animation.REVERSE);
anim.setRepeatCount(Animation.INFINITE);
```

깜빡이는 Text Animation 만들기

- AlphaAnimation은 투명도를 조절하는 클래스
  - new AlphaAnimation(0.0f,1.0f)
    - 매개 변수로 넣어야 하는 숫자는 투명도를 의미하는데 float형이고 범위는 0.0~1.0임
  - setDuration(100)
    - 지속시간을 의미하는데 AlphaAnimation을 생성하면서 지정해줬던 투명도를 몇 초 동안 실행하게 할 것이냐를 의미
    - 100 - 1초를 의미
  - setStartOffset(20)
    - 한번 Animation 끝난 뒤에 다음 Animation이 시작되기 위해 잠시 대기하는 시간이라고 생각하면 됨

깜빡이는 Text Animation 만들기

- AlphaAnimation은 투명도를 조절하는 클래스
  - setRepeatMode(Animation.REVERSE)
    - Animation이 반복되게 하려면 지정해주면 됨
    - 반복하고 싶지 않으면 해당 코드는 넣지 않아도 됨
  - setRepeatCount(Animation.INFINITE)
    - 몇 번 반복 할 것 인지를 정하는 함수
    - Animation.INFINITE를 넣으면 무한 반복을 할 수 있음
  - startAnimation(anim)
    - 원하는 Animation을 지정해주면 Animation이 시작됨
  - clearAnimation()
    - Animation을 종료시킬 수 있음

Button

Event

- 프로그램이 반응하도록 사용자가 만들어내는 동작이 발생하는 것을 말함
- Application과 사용자간의 상호작용
- Android에서는 프로그램이 시작부터 순차적으로 실행되는 것이 아니라 Event가 일어나기 전에는 다른 작업을 하고 있다가, Event가 발생하면 그 때 Application이 미리 지정해 둔 동작이 실행
- 이 처리 방식을 바로 이벤트 구동(Event-Driven) 방식
- 예) SmartPhone 화면을 Touch하거나 Button을 클릭하는 것

Event

- Event 처리 방법
  - View에서 발생하는 해당 Event 처리를 위해서는 각각의 Listener Interface를 구현하면서 Call Back 메소드에 처리하고자 하는 내용을 작성해야 함
  - Android에서 Event를 처리하는 방법
    - XML 파일에 Event 처리 메소드를 등록하는 방법
    - Event 처리 객체를 생성하는 방법
    - View 클래스의 Event 처리 메소드를 재정의하는 방법

Event

- Call Back 메소드
  - 특정 Event 발생시 시스템에 의해 자동으로 호출되는 메소드
  - 이 메소드에 코드를 작성함으로써 Event 발생시의 동작을 정의함
  - Event를 받는 가장 쉬운 방법은 해당 클래스를 상속받아 Call Back 메소드를 재정의하는 것

Event

- XML 파일에 Event 처리 메소드를 등록하는 방법
  - 유일하게 Click Event만 처리할 수 있는 방법
  - activity_main.xml 파일에서 TextView의 onClick 속성으로 textViewClicked 메소드를 지정해 준 것을 볼 수 있음
  - 사용자가 TextView를 Click하면 위의 레이아웃을 사용하는 Activity인 MainActivity.java에 정의된 textViewClicked라는 이름의 Click Event를 처리하는 메소드가 호출

Event

- Event 처리(Listener) 객체를 생성하는 방법
  - 가장 일반적으로 사용되는 방법으로 Event를 처리하는 객체를 별도로 생성하여 Widget에 등록하는 방법
  - Event를 처리하는 객체는 Event를 처리하는 메소드를 가지고 있어야 하며, 이는 이벤트 리스너(Event Listener)라는 인터페이스를 사용하면 됨
- Event Listener란?
  - Event를 처리하는 메소드들이 정의된 Interface를 말함
  - Interface스는 멤버변수가 필요 없이 꼭 필요한 메소드를 강제로 구현하게 하도록 하는 추상 메소드의 집합이라고 할 수 있음
  - 인터페이스는 멤버변수가 없어서 클래스(객체)로서의 역할은 할 수 없지만, 이는 리스너 객체를 모델링하기 위한 것이 아닌, 메소드 라이브러리를 만들어놓고 활용하기 위한 것

Event

- Event 처리(Listener) 객체를 생성하는 방법
- Interface란?
  - Interface스는 내용이 없는 메소드들의 형태만 구현해놓은 추상 메소드의 집합이라고 할 수 있음
  - 인터페이스는 일반 메소드나 일반 변수를 가질 수 없으며 변수의 형태는 static만 가능
  - 그리고 클래스와 가장 큰 차이점이 바로 생성자를 가질 수 없다는 것

Event

- Event 처리(Listener) 객체를 생성하는 방법
  - Listener의 종류

리스너	콜백 메소드	설명
OnClickListener	onClick()	사용자가 어떤 항목을 터치하거나 내비게이션 키나 트랙볼로 항목으로 이동한 후에 엔터키를 눌러서 선택하면 호출
OnLongClickListener	onLongClick()	사용자가 항목을 터치한 상태로 일정 시간동안 그대로 누르고 있으면 발생
OnFocusChangeListener	onFocusChange()	사용자가 하나의 항목에서 다른항목으로 이동할 때 호출
OnKeyListener	onKey()	포커스를 가지고 있는 항목 위에서 키를 눌렀다가 놓았을 때 호출
OnTouchListener	onTouch()	사용자가 터치 이벤트로 간주되는 동작을 한 경우에 호출

Event

- Listener 객체를 생성하는 방법
  - 내부 클래스
  - 익명 클래스
  - 액티비티 클래스 자체에 구현
  - 람다식

Event

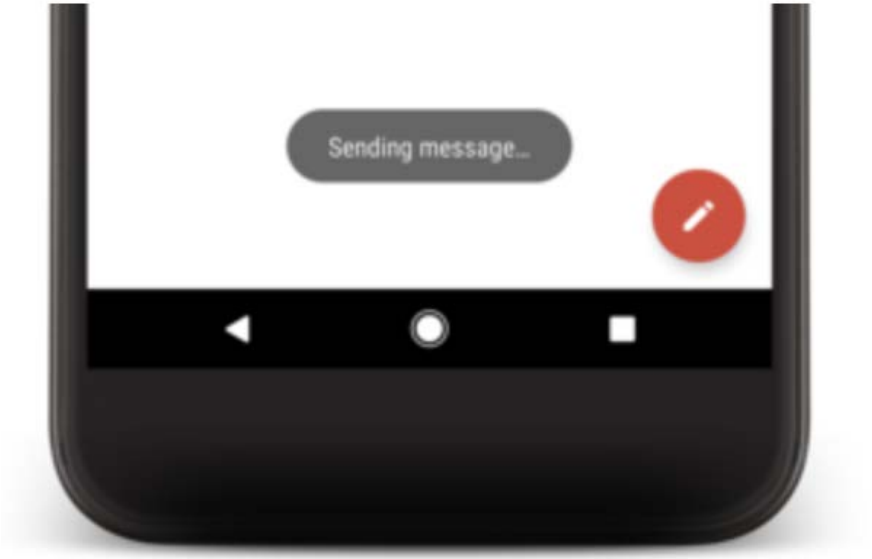
- View 클래스의 Event 처리 메소드를 재정의 하는 방법
  - Android의 Componenet들은 모두 View 클래스를 상속
  - Android 장치에서 사용자와 상호작용하는 객체는 View 클래스라는 것
  - 따라서 Event가 발생하면 View 클래스에 정의되어 있는 콜백 메소드가 호출. 따라서 View에서 발생한 이벤트를 처리하기 위해서는 뷰 클래스에 정의된 이벤트 메소드를 재정의하는 것이 가장 확실한 방법
  - 사용자 정의 클래스에서 이 방법을 사용하려면 반드시 View 클래스를 상속 받아야 함. 하지만 이벤트 처리만을 위해서 View 클래스를 통째로 상속받는 것은 효율적이지 못함
  - 이 방법은 사용자가 자신만의 커스텀 컴포넌트를 만들고자 View클래스를 상속 받았을 경우에 사용하면 동시에 이벤트 처리까지 할 수 있어서 좋음

Event

- Activity 자체에 Interface를 구현하는 방법
  - Interface를 구현한다는 말은 클래스를 정의할 때 인터페이스를 implements 한다는 것
  - Activity 자체에 리스너를 implements할 수 있음

Toast Message 사용하기

- Toast Message란 짧은 Message를 사용자에게 노출 시킨 후 일정 시간이 지나면 사라지는 팝업
- 우리가 앱에서 흔히 볼 수 있는 뒤로가기를 눌렀을 때 "뒤로가기를 한 번 더 누르면 앱이 종료됩니다."라고 뜨는 팝업이 토스트 메시지



Toast Message 사용하기

- 만드는 방법

```
Toast.makeText(getBaseContext(),  "Hello, World!",
                                          Toast.LENGTH_SHORT).show();
```

  - 첫 번째 인자는 컨텍스트
  - 두 번째는 노출할 메시지
  - 세 번째 인자는 노출 시간을 설정
    - Toast.LENGTH_SHORT는 약 4초(4000ms)
    - Toast.LENGTH_LONG는 약 7초(7000ms)

실습 예제

- 사용자 인터페이스

```
<?xml version="1.0" encoding="utf-8"?>
<font-family xmlns:android="http://schemas.android.com/apk/res/android">
    <font
        android:font="@font/swagger"
        android:fontStyle="normal"
        android:fontWeight="400" />
</font-family>
```

실습 예제

- 사용자 인터페이스

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".MainActivity">
```

실습 예제

- 사용자 인터페이스

```
<TextView
    android:id="@+id/textView1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ellipsize="marquee"
    android:focusable="true"
    android:paddingHorizontal="16dp"
    android:paddingVertical="12dp"
    android:singleLine="true"
    android:textColor="@android:color/white"
    android:text="[공지사항] 이번 학기 Android 수업은 매주 월요일에
                  대면으로 진행됩니다. 자세한 내용은 홈페이지를 참조해주세요."
    android:background="#1D6ECC"/>
```

실습 예제

- 사용자 인터페이스

```
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:clickable="true"
        android:fontFamily="@font/fontstyle"
        android:text="안녕 경복대학교"
        android:textSize="30dp"
        android:onClick="textViewClicked"/>

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="안녕 경복대학교" />
</LinearLayout>
```

실습 예제

- MainActivity.JAVA

```
public class MainActivity extends AppCompatActivity {

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

        Animation anim = new AlphaAnimation(0.0f, 1.0f);
        anim.setDuration(100);
        anim.setStartOffset(20);
        anim.setRepeatMode(Animation.REVERSE);
        anim.setRepeatCount(Animation.INFINITE);

        TextView textView1 = findViewById(R.id.textView1);
        textView1.setSelected(true);
```

실습 예제

- MainActivity.JAVA

```
        TextView textView2 = findViewById(R.id.textView2);
        textView2.startAnimation(anim);
        textView2.setTextColor(Color.RED);

        Button button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(getBaseContext(), "버튼이 눌렸습니다",
                                                    Toast.LENGTH_LONG).show();
            }
        });
    }
    public void textViewClicked(View view) {
        Toast.makeText(getBaseContext(), "TextView가 눌렸습니다",
                                                        Toast.LENGTH_LONG).show();
    }
}
```

EditText

- EditText는 TextView로부터 파생된 클래스로 TextView는 단순히 Text를 보여주는 역할을 한다면 EditText는 Text를 입력 및 수정까지 가능한 View Widget
- EditText를 화면에 배치하는 방법은 Layout 리소스 XML에서 <EditText></EditText> 사이에 EditText의 속성을 정의하면 됨

```
<EditText
   android:id="@+id/editText"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:ems="10"
   android:inputType="textPersonName"
   android:text="Sample EditText" />
```

EditText

- EditText 입력 및 수정 방지
  - 부모 클래스로부터 상속받은 속성 중 enabled 속성을 사용하면 EditText에 텍스트를 입력 및 수정이 불가능한 상태로 설정할 수 있음
  - default 속성값은 true로 사용 가능한 상태이며 false로 지정하게 되면 EditText를 사용할 수 없는 상태가 됨

```
<EditText
   android:id="@+id/Sample1"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:ems="10"
   android:inputType="textPersonName"
   android:text="editable = true" />
<EditText
   android:id="@+id/Sample2"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:ems="10"
   android:inputType="textPersonName"
   android:text="editable = false" />
```

EditText

- EditText 속성(hint / textColorHint)
  - hint 속성은 EditText 배경에 속성값으로 지정한 문자열을 나타나게 함
  - Text 영역에 어떠한 내용을 입력하라는 간단한 안내 문구나 텍스트 입력 예시를 표시할 때 사용
  - textColorHint 속성은 hint 속성에 의해 지정된 문자열의 색상을 지정할 때 사용

```
<EditText
   android:id="@+id/Sample1"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:ems="10"
   android:hint="이메일을 입력하세요."
   android:inputType="textPersonName"
   android:textColorHint="@color/colorAccent" />
```

EditText

- EditText 속성(selectAllOnFocus / textColorHighlight)
  - selectAllOnFocus 속성은 EditText를 클릭하였을 때 텍스트 영역을 전체 선택된 상태를 만들고자 할 때 사용
  - EditText를 클릭하고 텍스트를 입력하게 되면 기존에 있는 텍스트 내용은 일괄적으로 삭제되면서 새로 입력된 텍스트가 입력
  - textColorHighlight 속성은 EditText에서 Text에 해당하는 사각형 영역을 표현할 때 사용

```
<EditText
  android:id="@+id/Sample1"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:ems="10"
  android:inputType="textPersonName"
  android:selectAllOnFocus="true"
  android:text="Sample"
  android:textColorHighlight="@color/colorPrimary" />
```

EditText

- EditText 속성(maxLength)
  - maxLength 속성은 EditText에 입력 가능한 텍스트의 수를 지정할 수 있음

```
<EditText
    android:id="@+id/editText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="textPersonName"
    android:maxLength="5"
    android:text="Name" />
```

텍스트 길이는 **5**글자로 설정하였기 때문에 위 결과화면에서 더 이상의 텍스트 입력은 불가능

EditText

- EditText 속성(background 값에 @null)
  - EditText 밑줄은 default 값으로 존재
  - 디자인 상으로 밑줄을 없애고 싶을 때, xml 파일에서 간단히 제거할 수 있음

```
<EditText
   android:id="@+id/et_title"
   android:hint="제목"
   android:gravity="left|center"
   android:textSize="16dp"
   android:paddingLeft="10dp" />
```

```
<EditText
   android:id="@+id/et_title"
   android:hint="제목"
   android:gravity="left|center"
   android:textSize="16dp"
   android:paddingLeft="10dp"
   android:background="@null" />
```

EditText

- EditText 속성 (imeOptions)
  - 입력 후 엔터 키 -> 키보드 사라지게

```
android:singleLine="true"
android:imeOptions="actionDone"
```

EditText

- Keyboard의 <Enter> 버튼을 눌렀을 경우

```
input.setOnEditorActionListener(new OnEditorActionListener() {
    @Override
    public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
            if (actionId == EditorInfo.IME_ACTION_DONE ||
                    event.getKeyCode() == KeyEvent.KEYCODE_ENTER) {
                return true;
            }
            return false;
    }
});
```

문자열 리소스(strings.xml)

- Android Project에서 /res/value 경로 아래에 strings.xml이라는 문자열 리소스 파일이 있음
- Project를 처음 생성하면 아래와 같이 기본으로 app_name이라는 이름을 가진 문자열 리소스가 있음
- 사용할 문자열 추가하기
  - app_name과 같은 형태로 태그로 작성하려는 문자열을 감싸주면 됨
  - 반드시 name="" 속성을 사용해서 해당 문자열 리소스의 이름을 지정해 주어야 하며, 지정한 이름으로 코드에서 사용할 수 있음

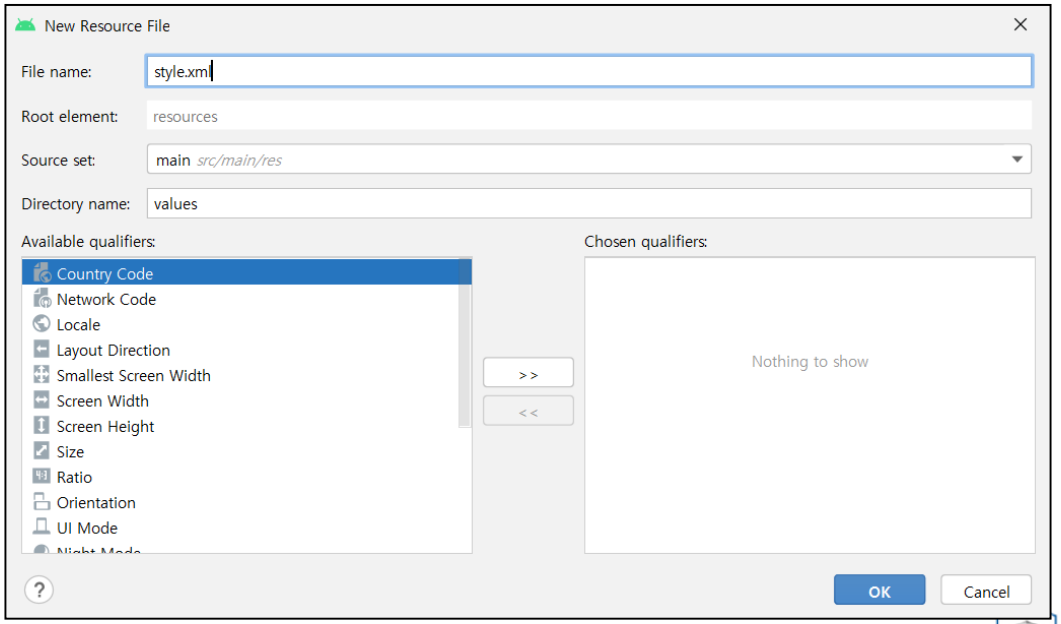
```
<resources>
   <string name="app_name">Sample</string>
   <string name="name">홍길동</string>
   <string name="nickname">별명이 무엇입니까?</string>
 </resources>
```

문자열 리소스(strings.xml)

- XML 파일에서 사용하기
  - 속성 중 android:text=""에 <string> 태그의 name 속성 값을 넣어줌
  - 예) android:text="@string/name"
  - 예) android:text="@string/nickname”

Style 적용하기

- TextView와 EditText에서 속성 값으로 {fontFamily, textColor, textSize}가 중복으로 인해 코드가 불필요하게 많아짐 -> 이를 해결하기 위해서 style.xml을 사용
- Style.xml 파일 추가
  - /res/values 폴더를 우 클릭
  - New/Value Resource File을 클릭



Style 적용하기

- style.xml에 style 추가하기

```
<?xml version="1.0" encoding="utf-8" ?>
<resources>
    <style name="MyStyle">
        <item name="android:fontFamily">@font/fontstyle</item>
        <item name="android:textColor">@color/black</item>
        <item name="android:textSize">20sp</item>
    </style>
</resources>
```

Style 적용하기

- 중복을 줄여보자

```
<TextView
   android:id="@+id/name_text"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:fontFamily="@font/fontstyle"
   android:text="name"
   android:textAlignment="center"
   android:textColor="@color/black"
   android:textSize="20sp" />
<EditText
  android:id="@+id/editText_name"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:fontFamily="@font/fontstyle"
  android:hint="what_is_your_nickname"
  android:textAlignment="center"
  android:textColor="@color/black"
  android:textSize="20sp" />
```

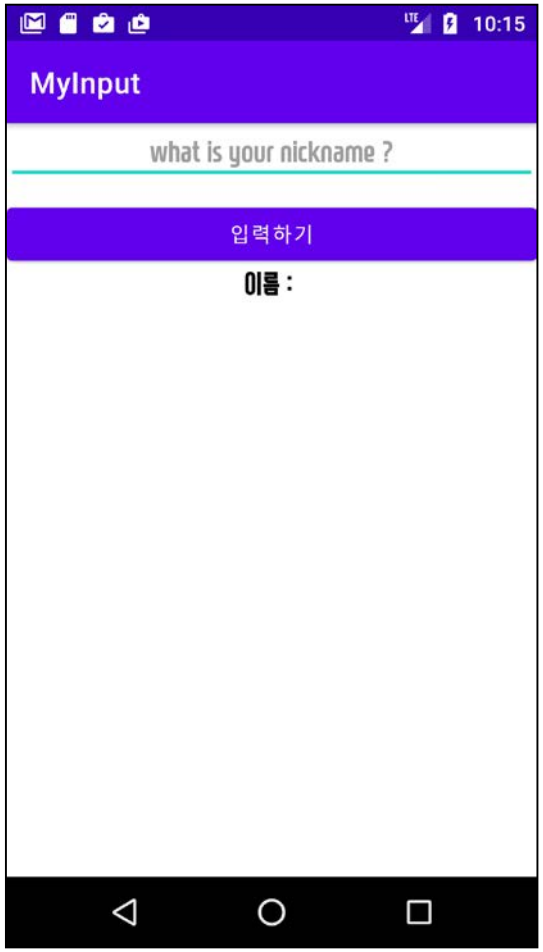
Style 적용하기

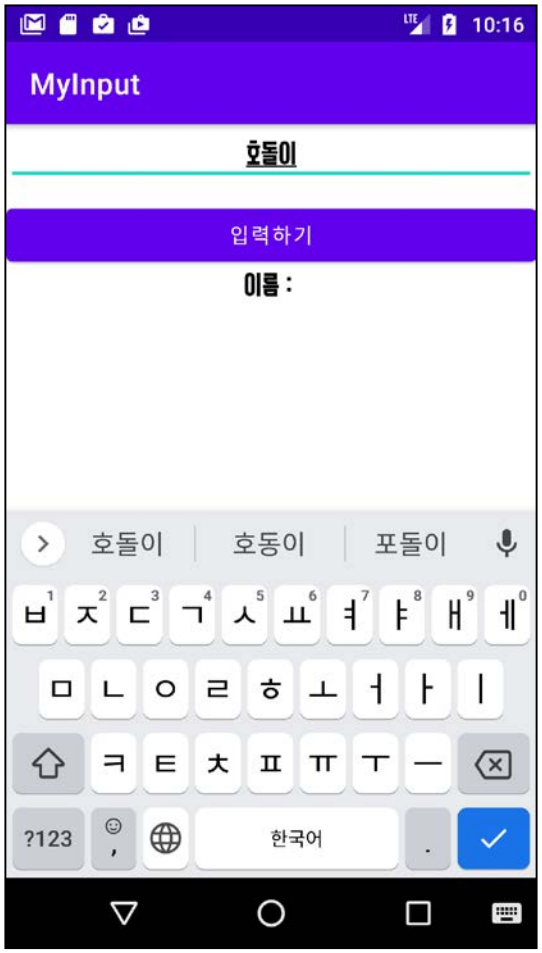
- 중복을 줄여보자

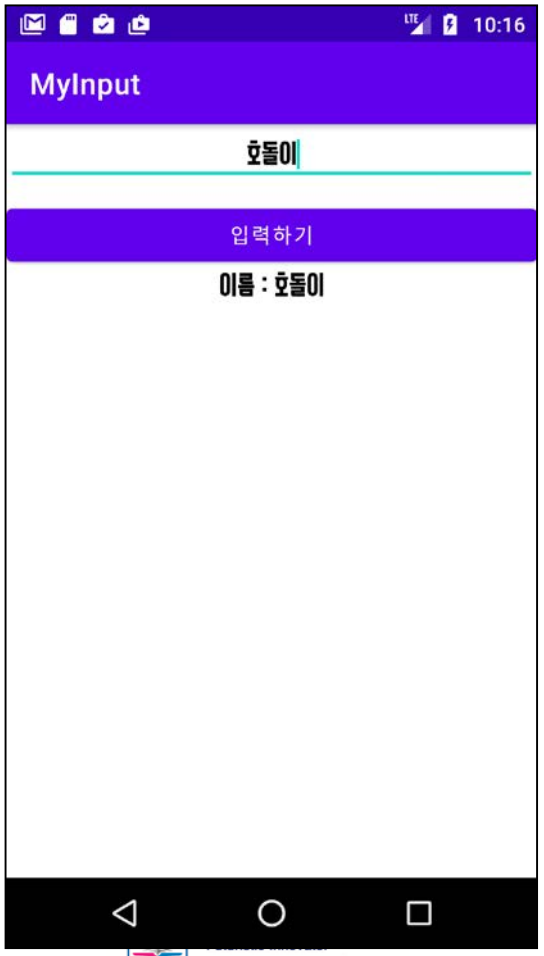
```
<TextView
   android:id="@+id/name_text"
   style="@style/MyStyle"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:text="name"
   android:textAlignment="center"/>

<EditText
  android:id="@+id/editText_name"
  style="@style/MyStyle"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:hint="what_is_your_nickname"
  android:textAlignment="center" />
```

실습 예제

- EditText에 입력된 Text를 가져와 TextView의 Text에 출력하는 프로그램을 작성하여라.







실습 예제

- 사용자 인터페이스

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/editText"
        style="@style/MyStyle"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="what is your nickname ?"
        android:textAlignment="center"
        android:singleLine="true"
        android:imeOptions="actionDone"/>
```

실습 예제

- 사용자 인터페이스

```
    <Button
        android:id="@+id/show"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:text="입력하기" />

    <TextView
        android:id="@+id/textView"
        style="@style/MyStyle"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="이름 : "
        android:textAlignment="center" />
</LinearLayout>
```

실습 예제

- MainActivity.JAVA

```
public class MainActivity extends AppCompatActivity {
    InputMethodManager manager;

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

        manager = (InputMethodManager) getSystemService(
                                                Context.INPUT_METHOD_SERVICE);
        EditText input = findViewById(R.id.editText);

        TextView textView = findViewById(R.id.textView);
        Button show = findViewById(R.id.show);
```

실습 예제

- MainActivity.JAVA

```
show.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String msg = input.getText().toString();
        if (msg.equals("")) {
            Toast.makeText(getBaseContext(), "입력해주세요",
                                                Toast.LENGTH_SHORT).show();
        } else {
            textView.setText("이름 : " + msg);
        }
    }
});
```

실습 예제

- MainActivity.JAVA

```
        input.setOnEditorActionListener(new TextView.OnEditorActionListener() {
            @Override
            public boolean onEditorAction(TextView textView, int i,
                                          KeyEvent keyEvent) {
                if (i == EditorInfo.IME_ACTION_DONE) {
                    manager.hideSoftInputFromWindow(input.getWindowToken(), 0);
                    return true;
                }
                return false;
            }
        });
    }
}
```

Soft Keyboard

- Android KeyBoard 입력 창 올리기/ 내리기
  - InputMethodManager 객체를 선언

```
InputMethodManager manager = (InputMethodManager)
                              getSystemService(INPUT_METHOD_SERVICE);
EditText input =  findViewById(R.id.editText);
```

  - KeyBoard 보이기/올리기

```
manager.showSoftInput(input, 0);
```

  - 키보드 숨기기/내리기

```
manager.hideSoftInputFromWindow(input.getWindowToken(), 0);
```

Soft Keyboard

- AndroidManifest.xml에서의 KeyBoard 제어하기
  - AndroidManifest.xml에 설정값을 넣어줌으로써, Activity 시작 시에 자동으로 KeyBoard를 보이거나 숨길 수 있음
  - <activity> 설정에서 android:windowSoftInputmode의 value에 "stateAlwaysVisible"과 "stateAlwaysHidden"를 넣어주면 됨
  - KeyBoard 보이기

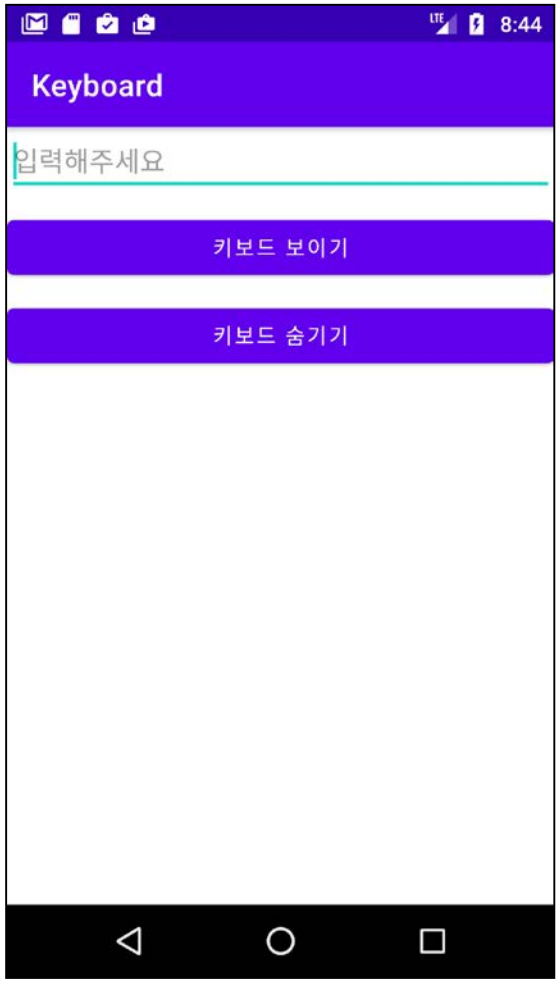
```
<activity android:name=".MainActivity"
          android:windowSoftInputMode="stateAlwaysVisible"/>
```

  - KeyBoard 숨기기

```
<activity android:name=".MainActivity"
          android:windowSoftInputMode="stateAlwaysHidden"/>
```

실습 예제

- Keyboard를 제어해보자



실습 예제

- 사용자 인터페이스

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="입력해주세요"
        android:textColor="#000000" />
```

실습 예제

- 사용자 인터페이스

```
    <Button
        android:id="@+id/show"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:text="키보드 보이기" />

    <Button
        android:id="@+id/hide"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:text="키보드 숨기기" />
</LinearLayout>
```

실습 예제

- 사용자 인터페이스

```
public class MainActivity extends AppCompatActivity
                                                    implements View.OnClickListener {
    InputMethodManager manager;
    EditText input;

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

        input = findViewById(R.id.editText);
        Button show = findViewById(R.id.show);
        Button hide = findViewById(R.id.hide);
        manager = (InputMethodManager) getSystemService(
                                                    INPUT_METHOD_SERVICE);
        show.setOnClickListener(this);
        hide.setOnClickListener(this);
    }
```

실습 예제

- 사용자 인터페이스

```
    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.show:
                manager.showSoftInput(input, 0);
                break;
            case R.id.hide:
                manager.hideSoftInputFromWindow(input.getWindowToken(), 0);
                break;
        }
    }
}
```