



EditText 속성(실습)

배 희호 교수
경북대학교
스마트IT과



EditText



- EditText는 TextView로부터 파생된 클래스로 TextView는 단순히 Text를 보여주는 역할을 한다면 EditText는 Text를 입력 및 수정까지 가능한 뷰(View) 위젯
- 값을 입력 받은 후 해당 값을 JAVA 코드에 가져와서 사용하는 용도로 많이 사용
- 대부분의 기능들이 TextView에 이미 구현되어 있음
 - 단지 TextView에서는 그것들이 활성화되지 않았을 뿐 임

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

EditText 계층도



EditText 속성



속성 이름	설명
autoLink	TextView를 링크 형태로 사용
autoText	텍스트 입력 시, 자동 오타 수정 기능 사용
breakStrategy	TextView의 텍스트 줄넘김 정책 지정
bufferType	getText() 함수로 리턴되는 버퍼 타입 지정
capitalize	알파벳 소문자 입력(표시) 시, 대문자로 자동 변환
cursorVisible	텍스트 입력 커서를 보일지 말지 여부 지정
digits	TextView에 입력 가능한 문자 제한
drawableBottom	텍스트를 기준으로 아래쪽에 이미지 출력
drawableEnd	텍스트를 기준으로 끝 위치에 이미지 출력
drawableLeft	텍스트를 기준으로 왼쪽에 이미지 출력
drawablePadding	텍스트와 이미지 사이의 간격 설정
drawableRight	텍스트를 기준으로 오른쪽에 이미지 출력
drawableStart	텍스트를 기준으로 시작 위치에 이미지 출력
drawableTint	drawable을 혼합하기 위한 색조(Tint) 지정



EditText 속성



속성 이름	설명
drawableTintMode	drawable tint 혼합(blending) 모드 설정
drawableTop	텍스트를 기준으로 위쪽에 이미지 출력
editable	입력 가능 여부 지정
editorExtras	텍스트 입력기에 추가 데이터 제공 (텍스트 입력기 구현에 한정)
elegantTextHeight	elegant height metrics 플래그 켜기
ellipsize	텍스트 생략기호(...) 또는 텍스트 흐르는 효과 주기
ems	EM 단위의 크기로 TextView의 고정 너비 설정
fontFamily	TextView의 텍스트 폰트 설정
fontFeatureSettings	Open Type 폰트(.otf)의 고급 설정 제어
freezesText	TextView의 상태(텍스트, 커서 등) 유지
gravity	TextView의 텍스트 정렬 방식 지정
height	TextView의 고정된 높이 지정
hint	TextView의 텍스트가 빈 상태일 때, 출력될 텍스트 설정



EditText 속성



속성 이름	설명
hyphenationFrequency	'-'(hyphen)이 추가되는 빈도 설정
imeActionId	텍스트 입력기(IME) 액션 버튼이 눌러졌을 때의 이벤트 ID 지정
imeActionLabel	텍스트 입력기(IME) 액션 버튼에 표시될 라벨 텍스트 지정
imeOptions	텍스트 입력기(IME)의 표시 옵션 지정
includeFontPadding	Font 위/아래 padding 사용 여부 설정
inputMethod	특정 유형의 입력 방법 지정
inputType	텍스트 입력기(IME)에서 입력 가능한 데이터 유형, 방법, 기능 지정
letterSpacing	글자 간격 조절
lineSpacingExtra	텍스트 줄 간격 조절 (텍스트 높이 상관 없이 지정 값 설정)
lineSpacingMultiplier	텍스트 줄 간격 조절 (텍스트 높이에 곱한 값 설정)



EditText 속성



속성 이름	설명
lines	정확히 텍스트 줄 단위로 TextView 높이 설정
linksClickable	링크 클릭 시, 링크 연결 프로그램 실행 여부 지정
marqueeRepeatLimit	marquee(텍스트 흐르는 효과) 애니메이션 반복 횟수 지정
maxEms	TextView의 EM 단위 최대 너비 설정
maxHeight	TextView의 최대 높이 지정
maxLength	TextView의 텍스트 최대 길이 제한
maxLines	TextView의 줄(line) 단위 최대 높이 지정
maxWidth	TextView의 최대 너비 지정
minEms	TextView의 EM 단위 최소 너비 설정
minHeight	TextView의 최소 높이 지정
minLines	TextView의 줄(line) 단위 최소 높이 지정
minWidth	TextView의 최소 너비 지정



EditText 속성



속성 이름	설명
numeric	숫자만 입력 가능하게 만들기
password	입력되는 텍스트 감추기 (대신 '.'(password dot) 표시)
phoneNumber	전화번호만 입력 가능하게 만들기
privateImeOptions	텍스트 입력기(IME)에 구현에 제한된(private) 옵션 지정
scrollHorizontally	TextView 가로 스크롤 가능하게 만들기
selectAllOnFocus	TextView가 Focus될 때, 자동으로 모든 텍스트 선택하기
shadowColor	텍스트 아래에 표시될 희미한 그림자 색상 지정
shadowDx	텍스트와 그림자 사이의 가로 방향(x 축) 간격 조절
shadowDy	텍스트와 그림자 사이의 세로 방향(y 축) 간격 조절
shadowRadius	그림자의 범위 조절
singleLine	텍스트가 한 줄로 출력되도록 강제
text	화면에 표시될 텍스트 지정



EditText 속성



속성 이름	설명
textAllCaps	텍스트를 모두 대문자로 출력
textAppearance	color, typeface, size, style을 한번에 설정
textColor	텍스트 색상 설정
textColorHighlight	선택 텍스트의 배경 색상 설정
textColorHint	hint 텍스트의 글자 색상 지정
textColorLink	링크로 사용되는 텍스트의 글자 색상 지정
textIsSelectable	TextView의 텍스트 선택 가능하도록 만들기
textScaleX	텍스트 글자 너비 조절(늘리기/줄이기)
textSize	텍스트 폰트 크기 설정
textStyle	텍스트 스타일(bold, italic) 지정
typeface	텍스트 폰트 typeface 설정
width	TextView의 고정된 너비 지정



EditText 속성



속성 값	설명
none	다른 xmr징이 없는 기본 EditText 입력 폼안에 줄바꿈이 가능
text	none과 같으나 줄 바꿈이 불가능
textCapCharacters	모든 입력된 영문이 대문자로 입력이 됨
textCapWords	단어의 첫번째 영문이 대문자로 입력 됨
textCapSentences	문장의 첫번째 영문이 대문자로 입력
textAutoCorrect	입력된 단어를 올바른 단어로 수정할 수 있음
textAutoComplete	단어를 입력중에 완성된 단어를 표시 할 수 있음
textMultiLine	입력 폼에 줄 바꿈이 가능하나 따로 설정하지 않으면 단일 줄의 텍스트로 제한됨
textImeMultiLine	여러줄의 텍스트 입력 가능 키보드에 줄바꿈 키가 표시됨



EditText 속성



속성 값	설명
textNoSuggestions	입력할때 사전에 등록되어있는 어떤 단어도 표시하지 않는다
textUri	URI를 입력
textEmailAddress	이메일 주소를 입력
textEmailSubject	이메일의 제목을 입력
textShortMessage	짧은 메시지를 입력
textLongMessage	긴 메시지를 입력
textPersonName	사람 이름을 입력
textPostalAddress	주소의 우편번호를 입력
textPassword	비밀번호를 입력 입력된 문자는 (*)로 표시



EditText 속성



속성 값	설명
textVisiblePassword	비밀번호를 입력 입력된 문자가 보임
textWebEditText	텍스트를 입력 웹 양식으로 제공
textFilter	다른 텍스트를 필터링 하기 위한 문자를 입력
textPhonetic	발음되는 발음 문자를 입력
textWebEmailAddress	이메일 주소를 입력 웹 양식으로 제공
textWebPassword	비밀번호를 입력 웹 양식으로 제공



EditText 속성



속성 값	설명
number	숫자만을 입력 받음
numberSigned	부호가 있는 숫자만을 입력 받음
numberDecimal	소숫점이 있는 소수를 입력 받음
numberPassword	숫자로 된 패스워드를 입력 받음
phone	전화번호를 입력 받음
datetime	날짜와 시간을 입력. 날짜는 -, 시간은 :로 구분
date	날짜를 입력
time	시간을 입력



EditText 속성



■ hint

- EditText가 비어 있고(empty) hint 텍스트가 표시되고 있는 상태에서, EditText에 텍스트가 입력되면 hint 속성에 지정된 텍스트는 사라지고, 사용자가 입력한 텍스트가 표시됨

■ textColorHint

- textColorHint는 hint의 색깔

■ lines

- 설정하지 않으면 자동으로 커지며, 설정하면 고정된 크기에서 내용이 많을 경우 스크롤(scroll)되면서 입력



EditText 속성

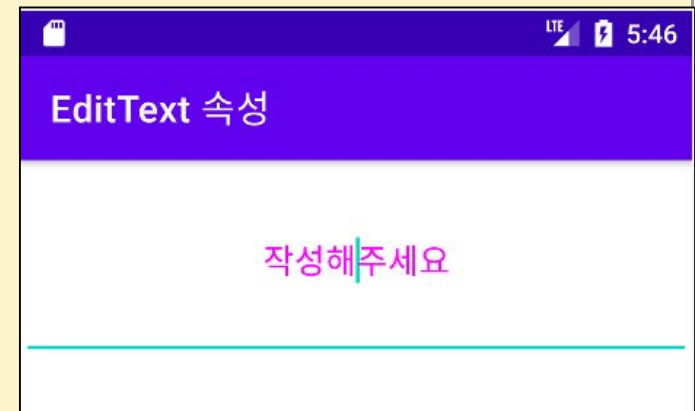
■ maxLength

■ maxLength는 최대 입력 가능한 길이

```
<?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"
    tools:context=".MainActivity">

    <EditText
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:hint="작성해주세요"
        android:gravity="center"
        android:textColorHint="#FF00FF"
        android:lines="4"
        android:maxLength="10"/>

</LinearLayout>
```





EditText 속성



- inputType

- 키보드(keyboard)에 표시되는 키의 종류가 달라짐

- 속성값

- Class

- 숫자, 문자, 비밀번호 등 입력 문자의 종류를 지정

- Variation

- Class의 세부적 속성이 들어감

- Flag

- 이보다 더 세부적인 속성으로 대문자 자동 변환, 여러 줄 입력 등의 동작을 수행

- 각 속성은 중복되어 사용가능 함

- 중복하여 사용하려면, |로 연결하여 사용



EditText 속성



inputType 속성 값	설명
none	편집이 불가능한 문자열
text	일반적 문자열
textMultiLine	여러 줄로 입력 가능
textPostalAddress	우편번호
textEmailAddress	이메일 주소
textPassword	패스워드
textVisiblePassword	패스워드 화면에 보임
number	숫자
numberSigned	부호가 붙은 숫자
numberDecimal	소수점이 있는 숫자
phone	전화번호
datetime	시간





EditText 속성



속성값	의미	적용 예
none	일반적인 입력기 모양	
text	<ul style="list-style-type: none">✓ 일반적인 입력기 모양에서 Next라는 기능이 추가됨✓ Next키를 누르면 현재 EditText에 있는 커서가 다음 EditText로 이동함	
phone phoneNumber 속성 대체	전화번호를 입력하기 편리한 입력기	






EditText 속성

<p>textNoSuggestions</p>	<ul style="list-style-type: none">✓ 특정 단어를 한 자씩 입력할 때마다 추천 단어가 상단에 표시됨✓ 하지만 textNoSuggestions 속성 값으로 설정하면 추천어를 표시하지 않음	
<p>number numeric 속성 대체</p>	<p>대부분 숫자 키로만 형성된 입력기가 나타남. 해당 입력기는 양수만 입력 가능(소수점, 음수 입력 불가). 숫자 입력 시에는 추천어가 필요치 않으므로 생략</p>	

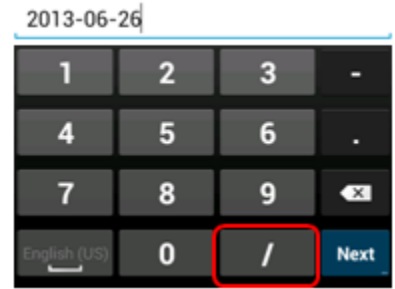

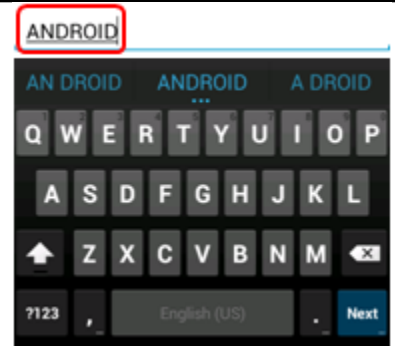


EditText 속성

<code>numbersigned</code> <code>numeric</code> 속성 대체	대부분 숫자 키로만 형성된 입력기가 나타남. 양수와 음수 입력이 가능(소수점 입력 불가). 숫자 입력에 시에는 추천어가 필요 없으므로 추천어는 제공되지 않음	
<code>numberDecimal</code> <code>numeric</code> 속성 대체	대부분 숫자 키로만 형성된 입력기가 나타남. 해당 입력기는 소수점 입력이 가능(음수 입력 불가). 숫자 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않음	
<code>time</code>	시간 정보를 입력하기에 적합한 입력기가 나타남. 특히 시분초 사이에 입력하는 기호 : 키가 존재. 시간 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않음	

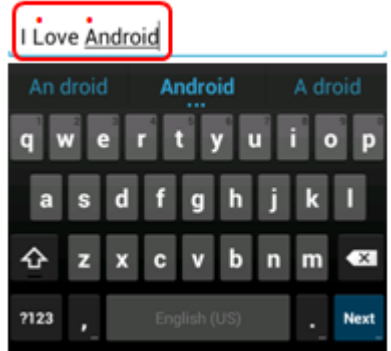
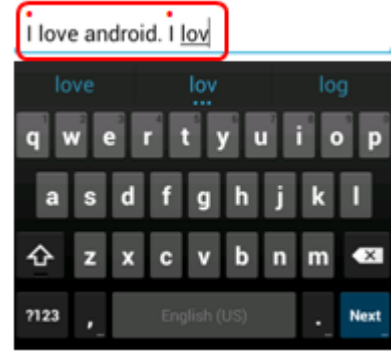
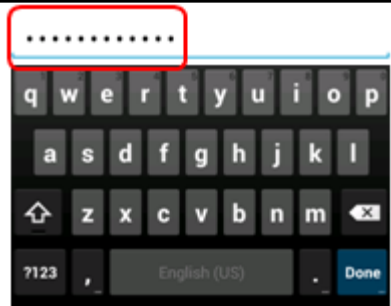


EditText 속성

date	날짜 정보를 입력하기에 적합한 입력기가 나타남. 특히 연월일 사이에 입력하는 기호 / 키가 존재. 날짜 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않음	
datetime	날짜와 시간 정보를 입력하기에 적합한 입력기가 나타남. 특히 연월일과 시분초 사이에 입력하는 기호 / : 키가 존재. 날짜와 시간 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않음	
textCapCharacters capitalize 속성 대체	모든 영문이 대문자로 입력. 입력 전에 Caps Lock 키를 해제하면 소문자도 입력 가능하나 한 글자를 입력하고 나면 다시 Caps Lock 키가 활성화되어 대문자 입력을 유도	






EditText 속성

<p>textCapWords capitalize 속성 대체</p>	<p>영문 한 단어 첫 자를 대문자로 입력. 입력 전에 Caps Lock 키를 해제하면 소문자도 입력 가능하나 한 단어를 입력하고 나면 다시 Caps Lock 키가 활성화되어 단어의 첫 글자 입력을 대문자로 입력하도록 유도</p>	
<p>textCapSentences capitalize 속성 대체</p>	<p>영문 한 문장의 첫 자를 대문자로 입력. 입력 전에 Caps Lock 키를 해제하면 소문자도 입력 가능하나, 한 문장을 입력하고 나면 다시 Caps Lock 키가 활성화되어 문장의 첫 글자 입력을 대문자로 입력하도록 유도</p>	
<p>textPassword password 속성을 대체</p>	<p>문자 기반의 입력기 형태이며, 입력되는 글자가 모두 • 기호로 표시</p>	



EditText 속성

<p>numberPassword</p> <p>주의 API 11부터 지원</p>	<p>숫자 기반의 입력기 형태이며, 입력되는 글자가 모두 • 기호로 표시</p>	
<p>textEmailAddress</p>	<p>이메일 주소를 입력하기 위한 입력기가 나타남. 특히 메일 주소에 꼭 들어가는 @와 . 기호 키가 추가됨. 이메일 주소 입력 시에는 추천어가 필요치 않으므로 생략</p>	
<p>textShortMessage</p>	<p>SMS 메시지를 입력할 때 많이 사용되는 이모티콘 모음 키가 추가</p>	



EditText 속성

■ phoneNumber

- 전화번호 형식만 입력 가능하도록 제한하는 속성
- 속성값
 - true 또는 false 값 지정 (기본 값 false)
- numeric, password 속성과 마찬가지로 API Level 3에서 deprecated 됨
- 대신 "inputType" 속성 값에 "phone"을 지정하는 것으로 동일한 기능

```
inputType= "phone"
```

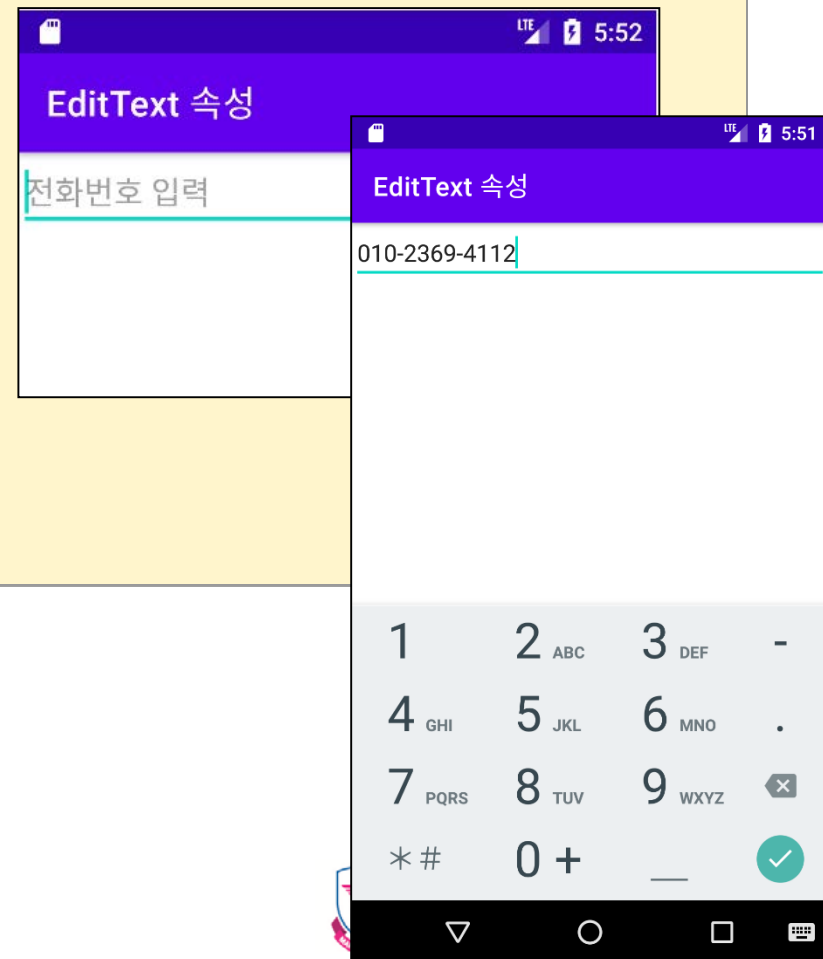


EditText 속성

```
<?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"
    tools:context=".MainActivity">

    <EditText
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:hint="전화번호 입력"
        android:inputType="phone"/>

</LinearLayout>
```





EditText 속성

■ password

- 입력되는 모든 문자열을 '.'(password dot)로 감춤
- 패스워드와 같은 보안값을 입력할 때 활용
- 속성값
 - true 또는 false 값 지정 (기본 값 false)
- password 속성은 API Level 3에서 deprecated되어 사용할 수 없음
- 대신 "inputType" 속성에 "textPassword" 값을 지정하는 것으로 대체할 수 있음

```
inputType="textPassword"
```

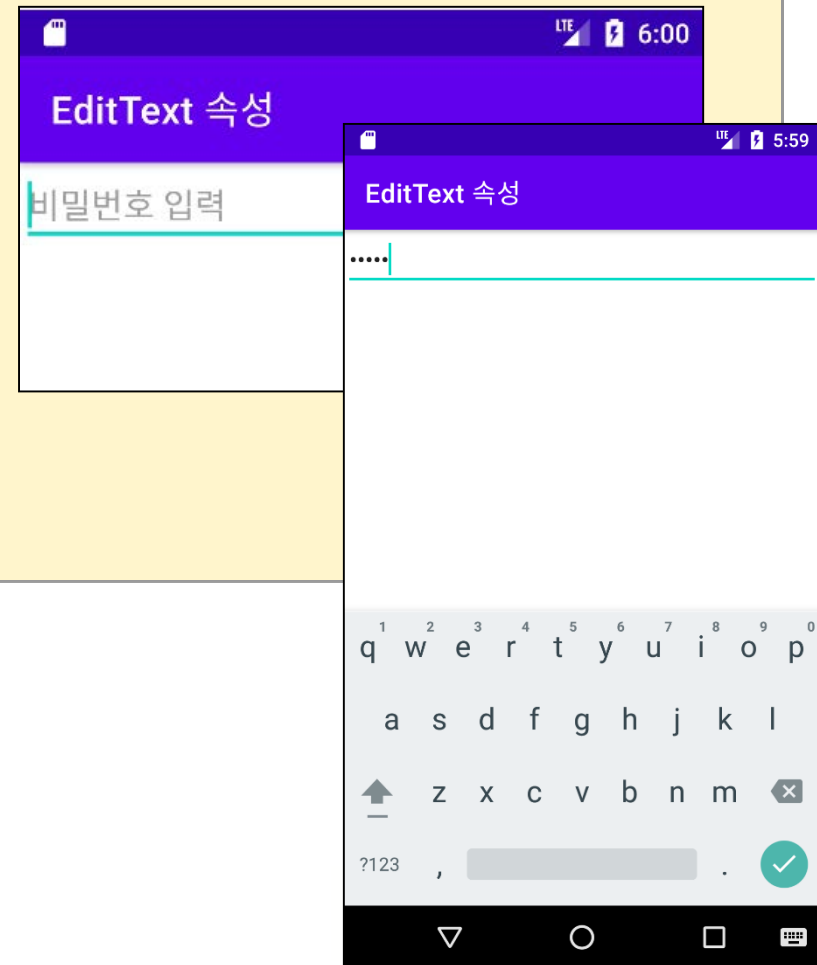


EditText 속성

```
<?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"
    tools:context=".MainActivity">

    <EditText
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:hint="비밀번호 입력"
        android:inputType="textPassword"/>

</LinearLayout>
```





EditText 속성

■ autoText

- 텍스트 입력 시, 자동 오타 수정 기능을 사용할 지 여부 지정
- 속성값
 - true 또는 false 지정 (기본 값 false)
- 일반적인 철자 범위 내에서 오타 수정 기능 제공
- 이 속성은 API level 3에서 deprecated 되었으며, 대신, 자동 오타 수정 기능은 "inputType" 속성을 통하여 제공

```
inputType="textAutoCorrect"
```



텍스트뷰(TextView) 속성

- 입력 문자 제한
 - digits, numeric 혼합 사용 가능
 - digits 속성
 - 지정해 놓은 문자열 내에 있는 문자들만 입력이 가능
 - numeric 속성
 - 숫자만 입력 가능

numeric 속성값	설명
integer	아라비아 숫자만 입력
signed	선두에 - 부호를 허용 숫자 중간에는 올 수 없음
decimal	소수점을 허용 소수점은 반드시 하나만 와야 함



EditText 속성



■ digits

- EditText에 입력될 수 있는 문자 지정
- 기본적으로 EditText에는 Android 시스템이 지원하는 모든 문자가 입력될 수 있음
- 하지만 여러가지 이유(미사용 문자 입력 방지, 오타 방지 등)로 인해 사용자가 입력할 수 있는 문자(digit)를 제한하도록 만들 수 있음

■ 속성값

- 문자열 값을 지정
- escape character는 '\x'를 사용 (예. '\n')
- unicode character는 '\uxxxx'를 사용

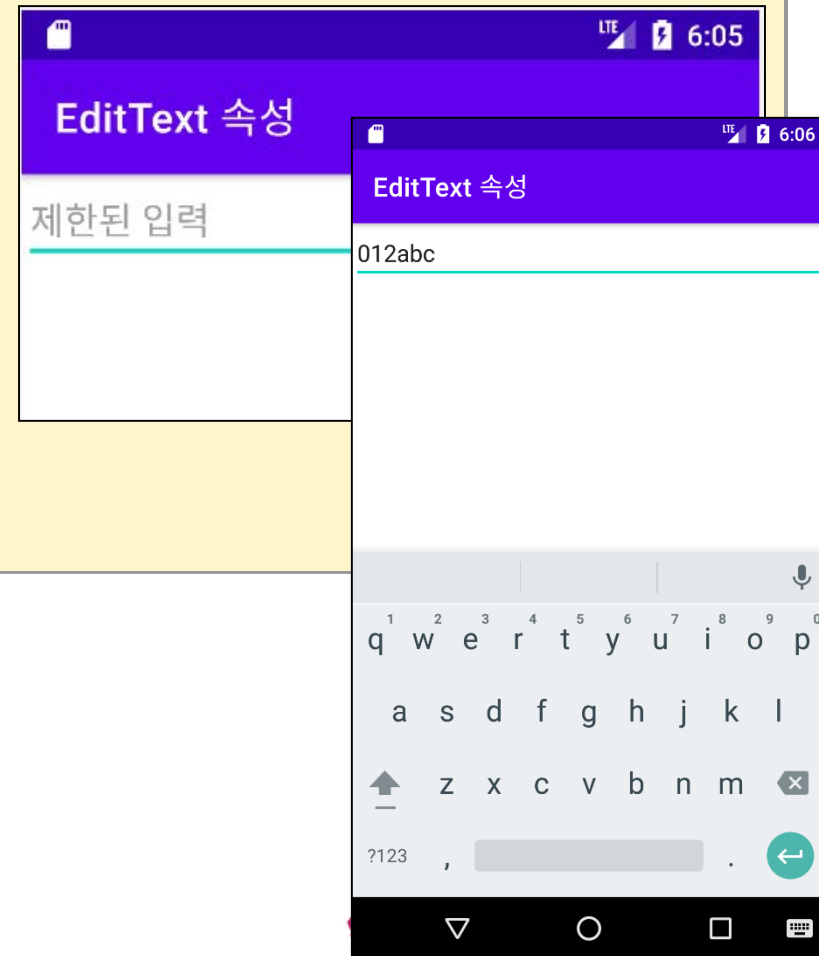


EditText 속성

```
<?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"
    tools:context=".MainActivity">

    <EditText
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:hint="제한된 입력"
        android:digits="012aAbBcC"/>

</LinearLayout>
```





EditText 속성



■ numeric

- 숫자만 입력되도록 만들 때 사용
- numeric 속성은 API Level 3에서 deprecated 되었으며, 대신 "inputType" 속성에 "number" 값을 사용하는 것으로 대체
- integer, signed, decimal 값 사용 가능
- 세 가지 중 하나 또는 |로 조합해서 사용 가능
- 예) "signed | decimal"
- 속성값
 - integer (0x01) : 숫자 입력 (정수만 가능)
 - signed (0x03) : 숫자 입력 (부호("-") 사용 가능)
 - decimal (0x05) : 숫자 입력 (소수(".") 입력 가능)



EditText 속성

■ numeric

number 속성 값	대체된 inputType 속성 값
integer (0x01)	inputType="number"
signed (0x03)	inputType="numberSigned"
decimal (0x05)	inputType="numberDecimal"



EditText 속성



```
<?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:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="정수 입력"
    android:inputType="number" />
```

```
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="정수 입력(음수 입력 가능)"
    android:inputType="numberSigned" />
```



EditText 속성

<EditText

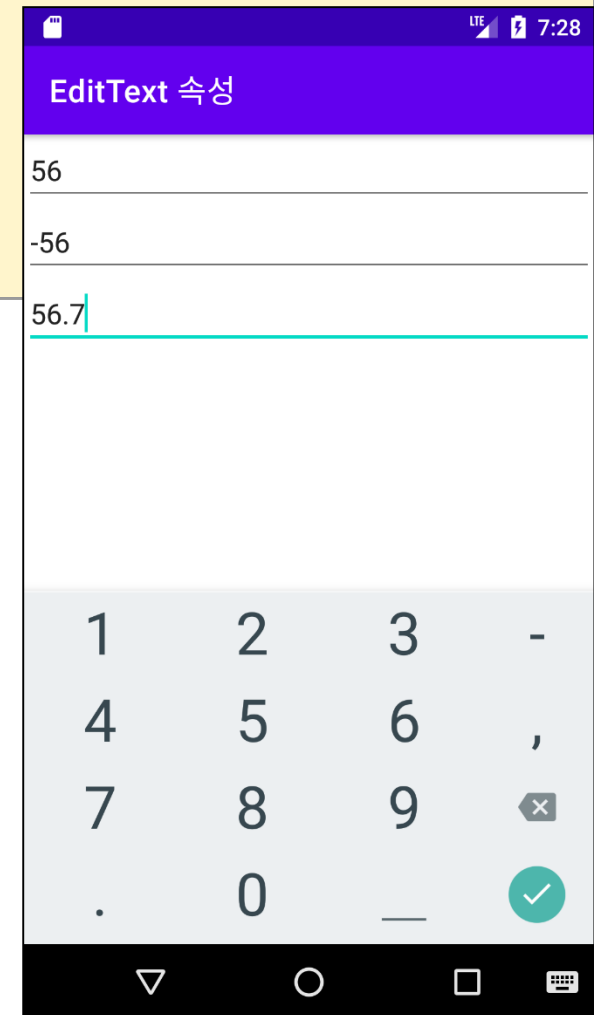
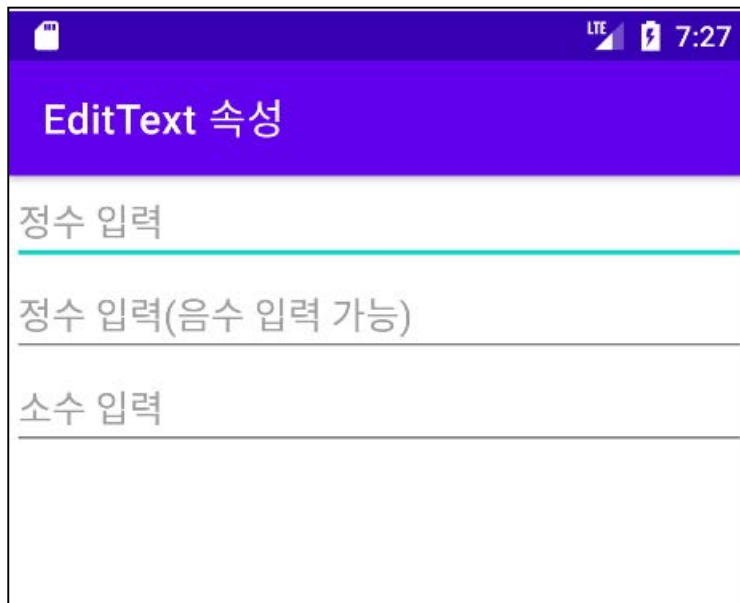
android:layout_width="match_parent"

android:layout_height="wrap_content"

android:hint="소수 입력"

android:inputType="numberDecimal" />

</LinearLayout>





EditText 속성

■ 문자열 입력 제한

- EditText의 입력 길이를 제한하는 정석적인 방법은 입력 필터를 사용

- `void setFilters (InputFilter[] filters)`

- 길이 제한 필터는 다음 클래스로 지정하며 생성자의 인수로 제한할 길이를 지정

- `InputFilter.LengthFilter (int max)`

■ 필터 적용 예

```
editText.setFilters(new InputFilter[]  
                    { filterKoEnNum });
```



EditText 속성

■ 3개의 대문자만 입력하도록 입력 제한 (내장 필터 이용)

```
<?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=".MainActivity7">

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="무제한 입력 가능" />

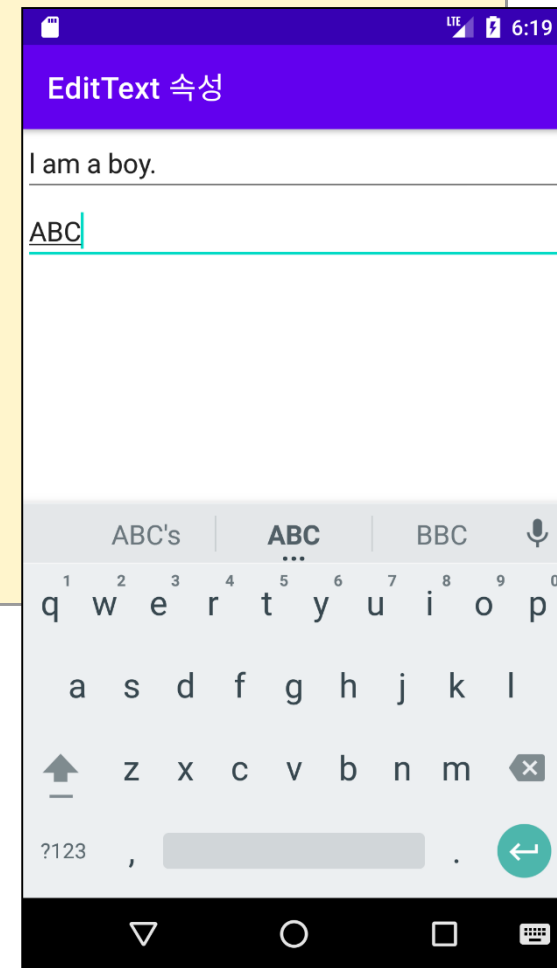
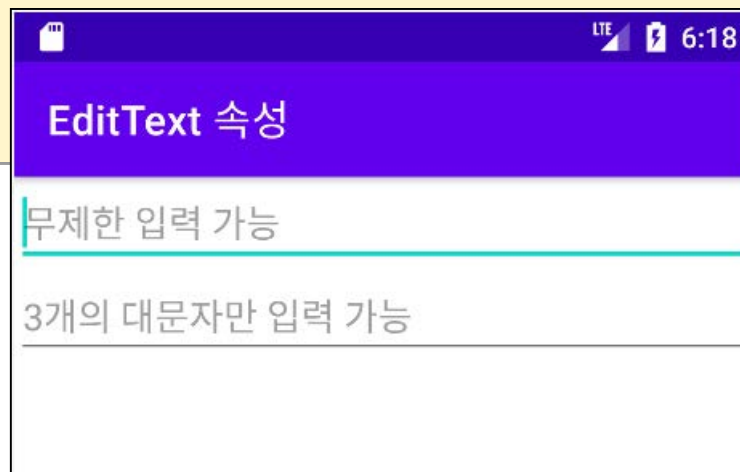
    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="3개의 대문자만 입력 가능" />

</LinearLayout>
```



EditText 속성

```
public class MainActivity7 extends AppCompatActivity {  
  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main7);  
  
        EditText editText = findViewById(R.id.editText);  
        editText.setFilters(new InputFilter[] {  
            new InputFilter.LengthFilter(3),  
            new InputFilter.AllCaps()  
        });  
    }  
}
```





EditText 속성

■ capitalize

- 텍스트 입력 시, 알파벳이 소문자로 입력되는 경우 대문자로 자동 변경하는 옵션을 제공할 수 있음
- 비록 대소문자가 없는 한국어에서는 무의미하나, 영어에서는 유용하게 사용될 수 있음
- capitalize 속성은 API level 3에서 deprecated 됨
- 속성값
 - none(0) : 자동 대문자 변환 사용 안 함(기본값)
 - sentences(1) : 각 문장의 첫 번째 단어를 대문자로 변환
 - words(2) : 각 단어의 첫 번째 글자를 대문자로 변환
 - characters(3) : 모든 글자를 대문자로 변환



EditText 속성

■ capitalize

capitalize 속성 값	대체된 inputType 속성 값
none(0)	inputType 사용 안함 또는 inputType="text"
sentences(1)	inputType="textCapSentences"
Words(2)	inputType="textCapWords"
characters(3)	inputType="textCapCharacters"



EditText 속성

```
<?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:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="대문자 입력"
        android:inputType="textCapCharacters" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="단어단위 대문자 입력"
        android:inputType="textCapWords" />
```

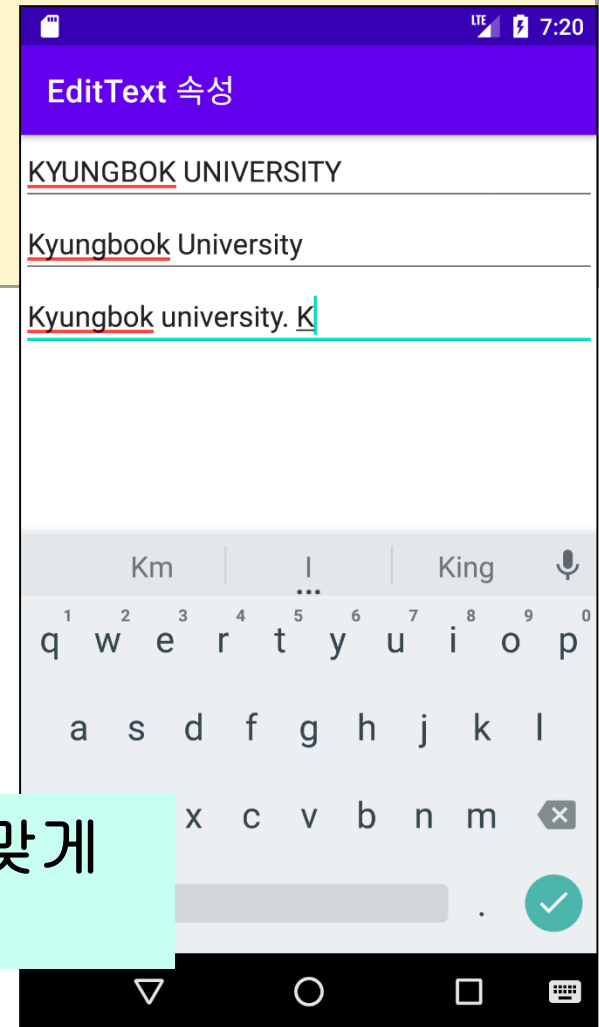
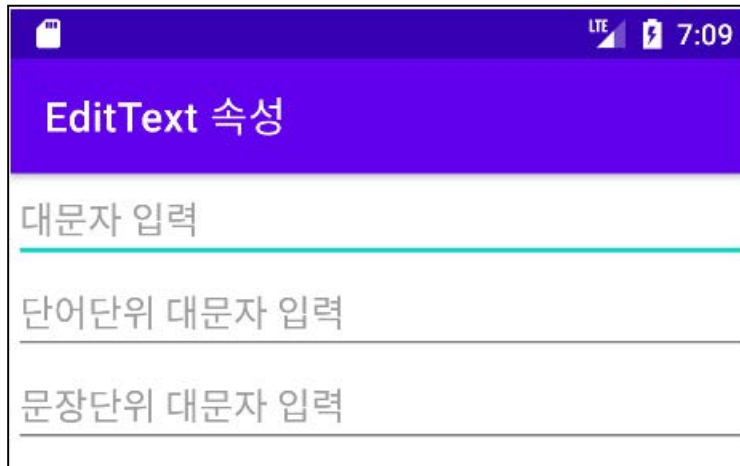



EditText 속성

<EditText

```
android:layout_width="match_parent"  
android:layout_height="wrap_content"  
android:hint="문장단위 대문자 입력"  
android:inputType="textCapSentences" />
```

</LinearLayout>



Caps Lock 키 상태가 속성 값에 맞게
변경되는 것임



EditText 속성

■ selectAllOnFocus

- EditText는 자신이 Focus되는 순간, 자신의 모든 텍스트를 선택 상태로 만드는 기능을 제공
- 다른 위젯에 Focus가 되어 있는 상태에서 EditText로 Focus가 이동되면, Focus가 이동된 EditText의 모든 텍스트가 자동으로 선택되도록 만드는 것
- 속성값
 - true 또는 false 값 지정 (기본 값 false)



EditText 속성



■ textColorHighlight

- EditText에 표시되는 텍스트 기본 색상은 "textColor" 속성으로 지정할 수 있음
- 그렇다면 텍스트를 선택했을 때의 배경 색상은 어떻게 지정할 수 있을까요?
 - "textColorHighlight" 속성을 사용하는 것
- 배경 컬러 값 지정
- textColorHighlight 속성을 사용하지 않았을 때, 선택 텍스트의 배경 색상은 기기의 Android 버전 및 테마에 따라 다르게 출력될 수 있음

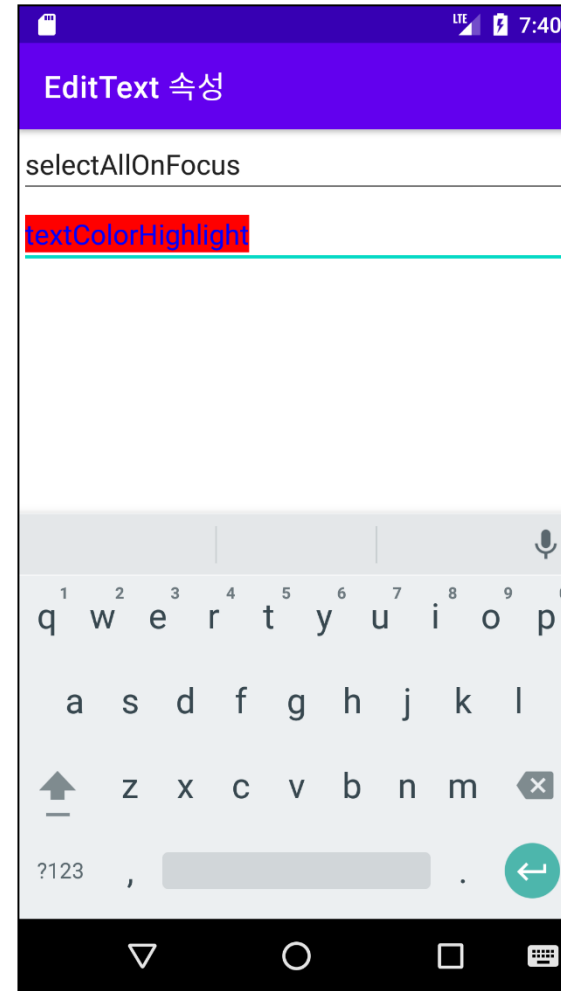


EditText 속성

```
<?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:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:selectAllOnFocus="true"
        android:text="selectAllOnFocus" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:selectAllOnFocus="true"
        android:text="textColorHighlight"
        android:textColor="#0000FF"
        android:textColorHighlight="#FF0000" />
</LinearLayout>
```



EditText 속성





EditText 속성

■ bufferType

- getText() 함수가 반환하는 최소(minimum) 버퍼 타입 지정

■ 속성값

- normal, spannable, editable값 사용 가능

- TextView의 기본 값 normal

- EditText 경우, 속성 설정과 관계없이 editable로 적용

- normal (0) : CharSequence 리턴 할 수 있음

- spannable (1) : 오직 Spannable만 리턴 할 수 있음

- editable (2) : 오직 Spannable 또는 Editable만 리턴 할 수 있음



EditText 속성



- **bufferType**
 - **CharSequence**
 - 바꿀 수 없는(immutable) 문자열(contents) 처리에 사용. (내용 불변)
 - **Spannable**
 - 문자열(contents)은 바꿀 수 없으나 markup 객체를 적용(attach) 또는 해제(detach) 가능
- **Editable**
 - 문자열(contents) 및 markup 객체를 모두 변경 가능



EditText 속성

- bufferType 값을 "editable"로 지정하면 셋 중 아무거나 타입 변환(cast)하여 사용할 수 있음

```
<TextView  
    android:id="@+id/textView1"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:bufferType="editable"  
    android:text="ABC" />
```

```
TextView textView = findViewById(R.id.textView1) ;  
CharSequence csStr = (CharSequence)textView.getText() ;
```




EditText 속성



- enable

- enable은 말 그대로 EditText의 사용여부 지정

- 속성값

- true, false로 제어가 가능



EditText 속성



```
<?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:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="editable " />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="enabled = false "
        android:enabled="false"/>
```



EditText 속성

```
<EditText  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="enable = true "  
    android:enabled="true"/>  
</LinearLayout>
```





EditText 속성

■ cursorVisible

- cursorVisible 속성을 사용하면 입력 커서를 보이거나 감출 수 있음
- 커서를 보일지 말지 여부 지정
- 속성값
 - true 또는 false 값 지정 (기본 값 true)



EditText 속성

- EditText에 자동 Focus 제거
 - EditText는 기본적으로 자동으로 포커스가 주어짐
 - 그래서 EditText가 있는 창이 열리면 키보드가 자동으로 올라오게 됨
 - EditText는 포커스를 받으면 커서가 문자열의 제일 끝으로 이동함
 - Activity가 실행되면서 자동으로 EditText에 포커스가 가는데 이를 제거하고 싶다면 EditText의 상위 Layout에 다음과 같은 속성을 추가

```
android:focusable="true"  
android:focusableInTouchMode="true"
```



EditText 속성

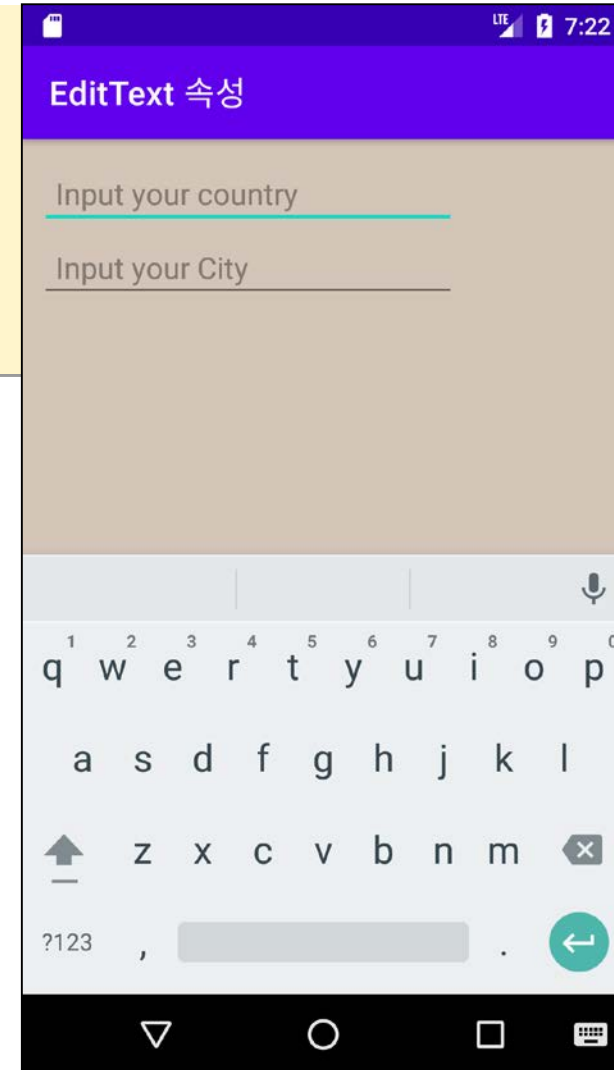
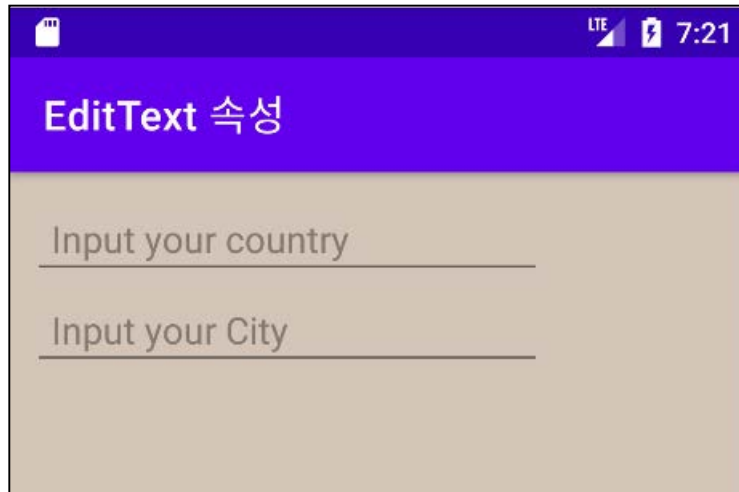
```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#d4c5b8"
    android:focusable="true"
    android:focusableInTouchMode="true"
    android:orientation="vertical"
    android:padding="10dp"
    tools:context=".MainActivity12">

    <EditText
        android:layout_width="250dp"
        android:layout_height="wrap_content"
        android:hint="Input your country"
        android:padding="10dp" />
```



EditText 속성

```
<EditText  
    android:layout_width="250dp"  
    android:layout_height="wrap_content"  
    android:hint="Input your City"  
    android:padding="10dp" />  
</LinearLayout>
```





EditText 속성

- EditText 밑줄(Underline) 제거 방법
 - EditText라는 것을 나타내주는 요소들이 커서와 밑줄이기도 하겠지만, 밑줄을 미관상 표시하고 싶지 않은 경우가 있음
 - xml 파일 상에서 EditText내의 background 옵션에 '투명' 색상 값, 혹은 'null'을 적용시켜주면, 밑줄을 나타내지 않는 것이 가능

방법 1.

```
android:background="@android:color/transparent"
```

방법 2.

```
android:background="@null"
```




EditText 속성

- EditText에 '숫자'와 '.'(점) 입력 키보드 띄우기

```
edittext.setInputType(InputType.TYPE_CLASS_NUMBER  
| InputType.TYPE_NUMBER_FLAG_DECIMAL)
```

- EditText에 '숫자'와 '.'(점) '-'(마이너스) 입력 키보드 띄우기

```
edittext.setInputType(InputType.TYPE_CLASS_NUMBER  
| InputType.TYPE_NUMBER_FLAG_DECIMAL  
| InputType.TYPE_NUMBER_FLAG_SIGNED)
```



EditText 속성



■ EditText 키보드 관리

- 키보드를 화면에 정렬하는 방법에 다음 속성들을 사용하면 변화를 최대한 부드럽게 처리할 수 있음
- 매니페스트의 <activity> 태그 속성으로 windowSoftInputMode의 값 지정

속성값	설명
adjustPan	<ul style="list-style-type: none">✓ 포커스를 받은 뷰가 보이도록 스크롤하여 이동✓ 입력을 받는 뷰는 보이지만 다른 뷰들이 키보드에 가려지는 문제가 있음. 다른 뷰를 보이게 하려면 입력 후 Back키를 눌러 키보드를 닫아야 함
adjustResize	<ul style="list-style-type: none">✓ 윈도우의 크기를 강제로 조정✓ 뷰의 크기가 줄어들기는 하지만 모든 뷰가 다 보이는 상태이므로 키보드가 열린 채로 다른 뷰를 사용할 수 있음
adjustUnspecified	<ul style="list-style-type: none">✓ 시스템이 Pan, Resize 중 하나를 자동으로 선택✓ 시스템은 스크롤이 가능한 뷰인지, 레이아웃에 여백이 얼마나 있는지 등을 고려하여 두 방법 중 하나를 선택



EditText 속성

■ EditText의 키보드 보이기

```
InputMethodManager imm = (InputMethodManager)
    context.getSystemService(
        Context.INPUT_METHOD_SERVICE);

imm.showSoftInput(editText,
    InputMethodManager.SHOW_IMPLICIT);
```



EditText 속성



■ EditText 키보드 숨기기

```
InputMethodManager imm = (InputMethodManager)
    activity.getSystemService(
        Context.INPUT_METHOD_SERVICE);
imm.hideSoftInputFromWindow(
    editText.getWindowToken(), 0);
```

■ EditText 키보드 toggle로 만들기

```
InputMethodManager imm = (InputMethodManager)
    activity.getSystemService(
        Context.INPUT_METHOD_SERVICE);
imm.toggleSoftInput(
    InputMethodManager.SHOW_FORCED, 0);
```



EditText 속성

■ 커서 및 선택 관리

- EditText는 현재 편집 위치를 표시하기 위해 커서를 표시하며 선택 영역이 있을 때는 선택 블록을 표시함
- 코드에서 커서의 위치나 선택 블록에 간섭이 필요할 경우 다음 메소드로 선택 영역을 변경하거나 조사할 수 있음

```
int getSelectionStart()  
int getSelectionEnd()  
void setSelection (int start, int stop)  
void setSelection (int index)  
void selectAll ()  
void extendSelection (int index)
```



EditText 속성

```
<?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:text="Edit Selection" />
```



EditText 속성

<Button

```
    android:id="@+id/button1"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Home"/>
```

<Button

```
    android:id="@+id/button2"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="End"/>
```

<Button

```
    android:id="@+id/button3"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Select Block"/>
```



EditText 속성



<Button

```
android:id="@+id/button4"  
android:layout_width="240dp"  
android:layout_height="wrap_content"  
android:text="Select All" />
```

<Button

```
android:id="@+id/button5"  
android:layout_width="240dp"  
android:layout_height="wrap_content"  
android:text="Get Selected Block" />
```

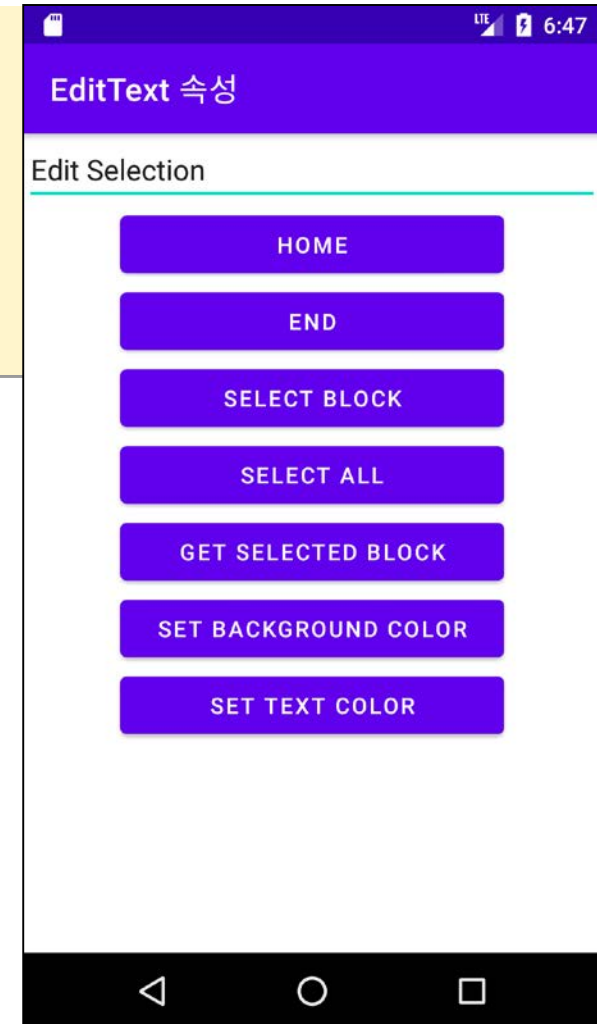
<Button

```
android:id="@+id/button6"  
android:layout_width="240dp"  
android:layout_height="wrap_content"  
android:text="SET BackGround Color" />
```




EditText 속성

```
<Button  
    android:id="@+id/button7"  
    android:layout_width="240dp"  
    android:layout_height="wrap_content"  
    android:text="SET Text Color" />  
</LinearLayout>
```





EditText 속성

```
public class MainActivity extends AppCompatActivity {  
    EditText editText;  
    boolean toggle = false;  
    boolean color = false;  
  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        editText = findViewById(R.id.editText);  
  
        Button button1 = findViewById(R.id.button1);  
        Button button2 = findViewById(R.id.button2);  
        Button button3 = findViewById(R.id.button3);  
        Button button4 = findViewById(R.id.button4);  
        Button button5 = findViewById(R.id.button5);  
        Button button6 = findViewById(R.id.button6);  
        Button button7 = findViewById(R.id.button7);  
    }  
}
```



EditText 속성

```
button1.setOnClickListener(listener);  
button2.setOnClickListener(listener);  
button3.setOnClickListener(listener);  
button4.setOnClickListener(listener);  
button5.setOnClickListener(listener);  
button6.setOnClickListener(listener);  
button7.setOnClickListener(listener);  
}
```

```
Button.OnClickListener listener = new Button.OnClickListener() {  
    public void onClick(View v) {  
        switch (v.getId()) {  
            case R.id.button1:  
                editText.setSelection(0);  
                break;  
            case R.id.button2:  
                editText.setSelection(editText.getText().length());  
                break;  
        }  
    }  
}
```



EditText 속성

```
case R.id.button3:
    editText.setSelection(3, 10);
    break;
case R.id.button4:
    editText.selectAll();
    break;
case R.id.button5:
    int start = editText.getSelectionStart();
    int end = editText.getSelectionEnd();
    Toast.makeText(getBaseContext(),
        "start = " + start + ",end = " + end,
        Toast.LENGTH_LONG).show();
    break;
```



EditText 속성

```
case R.id.button6:
    if (!toggle) {
        editText.setBackgroundColor(Color.RED);
        toggle = true;
    } else {
        editText.setBackgroundColor(Color.WHITE);
        toggle = false;
    }
    break;
case R.id.button7:
    if (!color) {
        editText.setTextColor(Color.BLUE);
        color = true;
    } else {
        editText.setTextColor(Color.BLACK);
        color = false;
    }
}
}
};
}
```



EditText 속성

■ 문자열 변경 Listener

- EditText는 편집 관련 기능들이 모두 활성화되어 있어 Layout에 배치해 놓기만 해도 즉시 편집이 가능함
- 텍스트가 변경되는 시점에 특정 작업을 하고 싶다면 다음 메소드로 Listener를 등록

```
void addTextChangedListener(TextWatcher watcher)
```

- 편집 이벤트를 처리하는 TextWatcher 객체를 생성한 후 Listener로 등록해 놓으면 사용자가 문자열을 편집할 때마다 TextWatcher 인터페이스의 다음 메소드가 호출

```
void beforeTextChanged (CharSequence s,  
                        int start, int count, int after)  
void afterTextChanged (Editable s)  
void onTextChanged (CharSequence s, int start,  
                   int before, int count)
```



EditText 속성

```
<?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=".MainActivity2">

    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="여기에 입력하면 입력내용을 보여줍니다 " />

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```



EditText 속성

```
<?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=".MainActivity2">

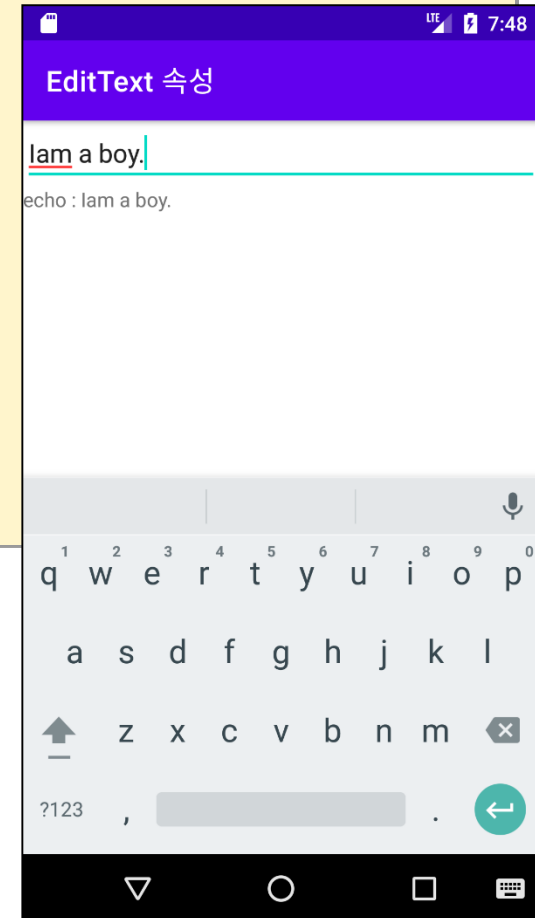
    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="여기에 입력하면 입력내용을 보여줍니다 " />

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```




EditText 속성

```
public class MainActivity2 extends AppCompatActivity {  
    TextView textView;  
  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main2);  
  
        textView = findViewById(R.id.textView);  
        EditText editText = findViewById(R.id.editText);  
        editText.addTextChangedListener(watcher);  
    }  
}
```





EditText 속성

```
TextWatcher watcher = new TextWatcher() {  
    public void afterTextChanged(Editable s) {  
    }  
    public void beforeTextChanged(CharSequence s, int start, int count,  
                                   int after) {  
    }  
    public void onTextChanged(CharSequence s, int start, int before,  
                               int count) {  
        textView.setText("echo : " + s);  
    }  
};  
}
```



EditText 속성

■ requestFocus

- 실행 시켰을 때 첫 포커스를 강제 설정

<EditText

android:id="@+id/main_userId"

android:layout_width="match_parent"

android:layout_height="wrap_content"

android:inputType="textVisiblePassword"

android:singleLine="true"

android:textSize="20dp" >

<requestFocus />

</EditText>



EditText 속성(키보드 제어)

■ 키보드 제어하기

```
<?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="키보드 제어하기 " />
```



EditText 속성(키보드 제어)

■ 사용자 인터페이스

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <Button
        android:id="@+id/button1"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="키보드 보이기"/>

    <Button
        android:id="@+id/button2"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="키보드 숨기기"/>
</LinearLayout>
</LinearLayout>
```



EditText 속성(키보드 제어)

■ MainActivity.JAVA

```
public class MainActivity extends AppCompatActivity {  
    EditText editText;  
    InputMethodManager manager;  
  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        manager = (InputMethodManager)  
            getSystemService(INPUT_METHOD_SERVICE);  
        editText = findViewById(R.id.editText);  
        Button button1 = findViewById(R.id.button1);  
        Button button2 = findViewById(R.id.button2);  
        button1.setOnClickListener(clickListener);  
        button2.setOnClickListener(clickListener);  
    }  
}
```



EditText 속성(키보드 제어)

■ MainActivity.JAVA

```
Button.OnClickListener clickListener = new Button.OnClickListener() {  
    public void onClick(View v) {  
        switch (v.getId()) {  
            case R.id.button1:  
                manager.showSoftInput(editText, 0);  
                break;  
            case R.id.button2:  
                manager.hideSoftInputFromWindow(  
                    editText.getWindowToken(), 0);  
                break;  
        }  
    }  
};  
}
```



EditText 속성

- EditText 글을 입력 후 다음 버튼을 눌렀을 때 포커스가 가는 순서 조정

```
EditText lotId_input = findViewById(R.id.lotStatus_lotId);  
lotId_input.setNextFocusDownId(R.id.lotStatus_okBtn);
```

// 버튼이든 EditText던 상관없이 포커스를 보낼 수 있음



EditText 속성

- 더 이상 포커스 이동을 중지 시키려면

<Button

```
    android:id="@+id/main_okBtn"  
    android:layout_width="fill_parent"  
    android:layout_weight="1.0"  
    android:imeOptions="actionDone"  
    android:text="ok" />
```



EditText 속성

■ 영문만 허용

```
protected InputFilter filterAlpha = new InputFilter() {  
    public CharSequence filter(CharSequence source, int start,  
        int end, Spanned dest, int dstart, int dend) {  
        Pattern ps = Pattern.compile("^[a-zA-Z]+$");  
        if (!ps.matcher(source).matches()) {  
            return "";  
        }  
        return null;  
    }  
};
```



EditText 속성

■ 영문만 허용 (숫자 포함)

```
protected InputFilter filterAlphaNum = new InputFilter() {  
    public CharSequence filter(CharSequence source, int start,  
                                int end, Spanned dest, int dstart, int dend) {  
        Pattern ps = Pattern.compile("^[a-zA-Z0-9]+$");  
        if (!ps.matcher(source).matches()) {  
            return "";  
        }  
        return null;  
    }  
};
```



EditText 속성

■ 한글만 허용

```
public InputFilter filterKor = new InputFilter() {  
    public CharSequence filter(CharSequence source, int start,  
        int end, Spanned dest, int dstart, int dend) {  
        Pattern ps = Pattern.compile("^[ㄱ-ㅎ가-힣]+$");  
        if (!ps.matcher(source).matches()) {  
            return "";  
        }  
        return null;  
    }  
};
```



EditText 속성

■ 한글, 숫자, 영어 허용

```
protected InputFilter filterKoEnNum = new InputFilter() {  
    public CharSequence filter(CharSequence source, int start,  
                                int end, Spanned dest, int dstart, int dend) {  
        Pattern ps = Pattern.compile("^[a-zA-Z0-9ㄱ-ㅎ가-힣]+$");  
        if (!ps.matcher(source).matches()) {  
            return "";  
        }  
        return null;  
    }  
};
```



EditText 속성

■ 한글, 숫자, 영어소문자, 띄어쓰기 허용

```
protected InputFilter filterKoEnNum2 = new InputFilter() {  
    public CharSequence filter(CharSequence source, int start,  
        int end, Spanned dest, int dstart, int dend) {  
        if(source.equals("")){ // for backspace  
            return source;  
        }  
        if(source.toString().matches("[a-z0-9ㄱ-ㅎ가-힣 ]+")){  
            return source;  
        }  
        Toast.makeText(getBaseContext(), "특수문자 및 영문대문자는  
            입력할 수 없습니다.").show();  
        return "";  
    }  
};
```



EditText 속성

■ 파일명에서 특수문자 사용 체크

```
InputFilter specialCharacterInFileNameFilter = new InputFilter() {  
  
    @Override  
    public CharSequence filter(CharSequence source, int start,  
        int end, Spanned dest, int dstart, int dend) {  
        boolean isOK = false;  
        String sText = new StringBuilder(source).toString();  
        try{  
            isOK = validateFileName(sText);  
        } catch (Exception e) {  
        }  
        if (!isOK) {  
            Toast t = Toast.makeText(getBaseContext(), "Please try  
                without special characters.", Toast.LENGTH_SHORT);  
            t.show();  
            return "";  
        }  
        return source;  
    }  
};
```