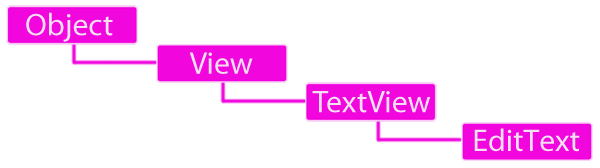
EditText

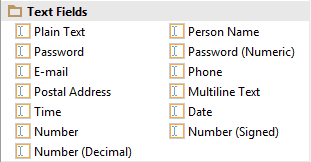
[EditText](http://developer.alexanderklimov.ru/android/views/edittext.php#edittext)  
[Текст-подсказка](http://developer.alexanderklimov.ru/android/views/edittext.php#hint)  
[Вызов нужной клавиатуры](http://developer.alexanderklimov.ru/android/views/edittext.php#keyboard)  
[Заблокировать текстовое поле](http://developer.alexanderklimov.ru/android/views/edittext.php#blocktext)  
[Другие свойства](http://developer.alexanderklimov.ru/android/views/edittext.php#other)  
[Методы](http://developer.alexanderklimov.ru/android/views/edittext.php#methods)  
[Выделение текста](http://developer.alexanderklimov.ru/android/views/edittext.php#select)  
[Обработка нажатий клавиш](http://developer.alexanderklimov.ru/android/views/edittext.php#keys)  
[Пустой ли EditText](http://developer.alexanderklimov.ru/android/views/edittext.php#empty)  
[Превращаем EditText в TextView](http://developer.alexanderklimov.ru/android/views/edittext.php#totextview)

EditText

Компонент **EditText** — это текстовое поле для пользовательского ввода, которое используется, если необходимо редактирование текста. Следует заметить, что **EditText** является наследником **TextView**.



В Android Studio на панели инструментов текстовые поля можно найти в категории **Text Fields** под разными именами.



Для быстрой разработки текстовые поля снабдили различными свойствами и дали разные имена: Plain Text, Person Name, Password, Password (Numeric), E-mail, Phone, Postal Address, Multiline Text, Time, Date, Number, Number (Signed), NumberDecimal.

Plain Text

**Plain Text** - самый простой вариант текстового поля без наворотов. При добавлении в разметку его XML-представление будет следующим:

<EditText

android:id="@+id/editText1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:ems="10" />

Person Name

При использовании элемента **Person Name** в XML добавляется атрибут **inputType**, который отвечает за вид клавиатуры (только буквы) при вводе текста.

<EditText

android:id="@+id/editText2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:ems="10"

android:inputType="textPersonName" />

Password и Password (Numeric)

При использовании **Password** в **inputType** используется значение **textPassword**. При вводе текста сначала показывается символ, который заменяется на звёздочку. Если используется элемент **Password (Numeric)**, то у атрибута **inputType** используется значение **numberPassword**. В этом случае на клавиатуре будут только цифры вместо букв. Вот и вся разница.

<EditText

android:id="@+id/editText2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:ems="10"

android:inputType="textPassword" />

E-mail

У элемента **E-mail** используется атрибут **android:inputType="textEmailAddress"**. В этом случае на клавиатуре появляется дополнительная клавиша с символом **@**, который обязательно используется в любом электронном адресе.

Кстати, финны называют символ @ кошачьим хвостом (взято из Википедии). А в России этот символ называют собакой. Где справедливость?

Phone

У элемента **Phone** используется атрибут **android:inputType="phone"**. Клавиатура похожа на клавиатуру из старого кнопочного сотового телефона с цифрами, а также с кнопками звёздочки и решётки.

Postal Address

Атрибут **android:inputType="textPostalAddress"**.

Multiline Text

У **Multiline Text** используется атрибут **android:inputType="textMultiLine"** позволяющий сделать текстовое поле многострочным. Дополнительно можете установить свойство **Lines** (атрибут**android:lines**), чтобы указать количество видимых строк на экране.

<EditText

android:id="@+id/editText1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:inputType="textMultiLine"

android:lines="5" >

Time и Date

Атрибут **android:inputType="time"** или **android:inputType="date"**. На клавиатуре цифры, точка, запятая, тире.

Number, Number (Signed), Number (Decimal)

Атрибут **android:inputType="number"** или **numberSigned** или **numberDecimal**. На клавиатуре только цифры и некоторые другие символы.

Текст-подсказка

Веб-мастера знают о таком атрибуте HTML5 как placeholder, когда в текстовом поле выводится строчка-подсказка приглушенным (обычно серым цветом). Живой пример приведен ниже.



Подсказка видна, если текстовый элемент не содержит пользовательского текста. Как только пользователь начинает вводить текст, то подсказка исчезает. Соответственно, если удалить пользовательский текст, то подсказка появляется снова. Это очень удобное решение во многих случаях, когда на экране мало места для элементов.

В Android у многих элементов есть свойство **Hint** (атрибут **hint**), который работает аналогичным образом. Установите у данного свойства нужный текст и у вас появится текстовое поле с подсказкой.



Запускаем приложение и видим подсказку, которая исчезает при попытке ввести текст.

Вызов нужной клавиатуры

Не во всех случаях нужна стандартная клавиатура с буковками и цифрами. Если вы пишете калькулятор, то проще показать пользователю цифровую клавиатуру. А если нужно ввести электронный адрес, то удобнее показать клавиатуру, где уже есть символ @. Ну а если ваше приложение пишется для котов, то достаточно вывести только те буквы, из которых можно составить слова **Мяу** и **Жрать давай** (к сожалению, такой клавиатуры ещё нет, но Google работает в этом направлении).

У элемента **EditText** на этот случай есть атрибут **inputType**:

<EditText android:id="@+id/etWidth1"

android:hint="@string/catname"

**android:inputType="textCapWords"**

android:layout\_height="wrap\_content"

android:layout\_width="wrap\_content">

</EditText>

В данном случае с атрибутом **inputType="textCapWords"** каждый первый символ каждого слова при вводе текста автоматически будет преобразовываться в прописную. Удобно, не так ли?

Значение **textCapSentences** делает прописным каждый первый символ предложения.

Если вам нужен режим CapsLock, то используйте значение **textCapCharacters** и все буквы сразу будут большими при наборе.

Для набора телефонного номера используйте **phone**, и тогда вам будут доступны только цифры, звездочка (\*), решетка (#).

Для ввода веб-адресов удобно использовать значение **textUri**. В этом случае у вас появится дополнительная кнопочка **.com** (при долгом нажатии на нее появятся альтернативные варианты .net, .org и др.).

Вот вам целый список доступных значений (иногда различия очень трудно различимы)

text

textCapCharacters (клавиатура с символами в верхнем регистре)

textCapWords

textCapSentences

textAutoCorrect

textAutoComplete

textMultiLine

textImeMultiLine

textNoSuggestions (без подсказок при вводе текста)

textUri

textEmailAddress

textEmailSubject

textShortMessage

textLongMessage

textPersonName

textPostalAddress

textPassword

textVisiblePassword (без автокоррекции)

textWebEditText

textFilter

textPhonetic

number

numberSigned

numberDecimal

phone

datetime

date

time

Заблокировать текстовое поле

Для блокировки текстового поля присвойте значения **false** свойствам **Focusable**, **Long clickable** и **Cursor visible**.

Другие свойства

**minLines и maxLines**

Позволяют ограничить количество строк текста, которое можно ввести в текстовом поле

**maxLength**

Позволяет задать максимальное количество символов для ввода

Методы

Основной метод класса **EditText** — **getText()**, который возвращает текст, содержащийся в текстовом поле. Возвращаемое значение имеет специальный тип **Editable**, а не **String**.

String strCatName = nickNameEditText.getText().toString(); // приводим к типу String

Соответственно, для установки текста используется метод **setText()**.

Большинство методов для работы с текстом унаследованы от базового класса **TextView**: **setTypeface(null, Typeface)**, **setTextSize(int textSize)**, **SetTextColor(int Color)**.

Выделение текста

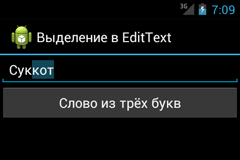
У **EditText** есть специальные методы для выделения текста:

* **selectAll()** — выделяет весь текст;
* **setSelection(int start, int stop)** — выделяет участок текста с позиции start до позиции stop;
* **setSelection(int index)** — перемещает курсор на позицию index;

Предположим, нам нужно выделить популярное слово из трёх букв в большом слове (это слово "кот", а вы что подумали?).

// выделяем 4, 5, 6 символы

edit.setSelection(3, 6);



Обработка нажатий клавиш

Для обработки нажатий клавиш необходимо зарегистрировать обработчик **View.OnKeyListener**, используя метод **setOnKeyListener()** элемента **EditText**. Например, для прослушивания события нажатия клавиши Enter во время ввода текста пользователем (или котом), используйте следующий код:

final EditText editText = (EditText)findViewById(R.id.EditText1);

editText.setOnKeyListener(new View.OnKeyListener()

{

public boolean onKey(View v, int keyCode, KeyEvent event)

{

if(event.getAction() == KeyEvent.ACTION\_DOWN &&

(keyCode == KeyEvent.KEYCODE\_ENTER))

{

// сохраняем текст, введенный до нажатия Enter в переменную

String strCatName = editText.getText().toString();

return true;

}

return false;

}

}

);

Пустой ли EditText

Чтобы проверить, пустой ли **EditText**, можно воспользоваться кодом:

if (editText.getText().toString().equals(""))

{

// Здесь код, если EditText пуст

}

else

{

// если есть текст, то здесь другой код

}

Также можно проверять длину текста, если она равно 0, значит текст пуст.

if (editText.getText().length() == 0)

Превращаем EditText в TextView

Практического смысла тут нет, но для общего развития превратим текстовое поле в текстовую метку. Для этого нужно сделать компонент недоступным, убрать курсор, установить прозрачный фон и отключить слушатель нажатий клавиш.

EditText editText = (EditText)findViewById(R.id.editText);

editText.setEnabled(false);

editText.setCursorVisible(false);

editText.setBackgroundColor(Color.TRANSPARENT);

editText.setKeyListener(null);

Также можно установить эти свойства через XML, кроме последнего пункта.

<EditText

android:id="@+id/editText"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

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

android:cursorVisible="false"

android:editable="false"

android:ems="10"

android:text="Text" />

Дополнительное чтение

[Переопределяем кнопку Enter](http://developer.alexanderklimov.ru/android/keyboard.php#enter)

[Собственный компонент - быстрая очистка текста](http://developer.alexanderklimov.ru/android/views/quickclearedittext.php)