TextView

[Общие сведения](http://developer.alexanderklimov.ru/android/views/textview.php#common)  
[Программная установка текста](http://developer.alexanderklimov.ru/android/views/textview.php#code)  
[Программная установка фона](http://developer.alexanderklimov.ru/android/views/textview.php#background)  
[Реагируем на событие onClick](http://developer.alexanderklimov.ru/android/views/textview.php#onclick)  
[Многострочный текст](http://developer.alexanderklimov.ru/android/views/textview.php#newline)  
[Увеличиваем интервалы между строками](http://developer.alexanderklimov.ru/android/views/textview.php#spacing)  
[Бой с тенью](http://developer.alexanderklimov.ru/android/views/textview.php#shadow)  
[Создание ссылок автоматом](http://developer.alexanderklimov.ru/android/views/textview.php#autolink)  
[Совет: Используйте полупрозрачность с умом](http://developer.alexanderklimov.ru/android/views/textview.php#semitrasparent)  
[Выделить текст для копирования](http://developer.alexanderklimov.ru/android/views/textview.php#selectable)

Компонент **TextView** предназначен для отображения текста без возможности редактирования его пользователем, что видно из его названия (Text - текст, view - просмотр).

Находится в разделе **Widgets** под четырьмя вариантами: **Plain TextView, Large Text, Medium Text, Small Text**. Отличаются размерами шрифта.

**TextView** - один из самых используемых компонентов. С его помощью пользователю удобнее ориентироваться в программе. По сути, это как таблички: *Руками не трогать, По газону не ходить, Вход с собаками воспрещен, Часы работы с 9.00 до 18.00* и т.д., и служит для представления пользователю описательного текста.

Для отображения текста в **TextView** в файле разметки используется атрибут **android:text**, например:

android:text="Погладь кота, ...!"

Такой подход является нежелательным. Рекомендуется всегда использовать текстовые ресурсы. В будущем эта привычка позволит вам обеспечить многоязыковую поддержку:

android:text="@string/hello"

Программная установка текста

Программно текст можно задать методом **setText()**:

// Инициализируем компонент

TextView textView = (TextView)findViewById(R.id.textView);

// задаём текст

textView.setText("Hello Kitty!");

// или с использованием текстовых ресурсов

textView.setText(R.string.hello);

Атрибуты

**android:textsize**

размер текста. При установке размера текста используется несколько единиц измерения: px (пиксели), dp, sp, in (дюймы), pt, mm. Для текстов рекомендуется использовать **sp**: android:textSize="48sp", аналог - метод **setTextSize()**

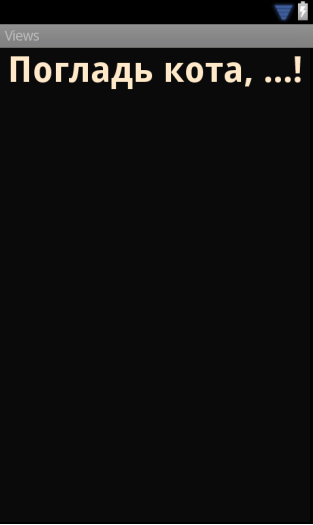
**android:textstyle**

стиль текста. Используются константы: **normal, bold, italic**. Например, **android:textStyle="bold"** выводит текст жирным

**android:textcolor**

цвет текста. Используются четыре формата в шестнадцатеричной кодировке: #RGB; #ARGB; #RRGGBB; #AARRGGBB, где R, G, B — соответствующий цвет, А — прозрачность (alpha-канал). Значение А, установленное в 0, означает прозрачность 100%.

Для всех вышеперечисленных атрибутов в классе **TextView** есть соответствующие методы для чтения или задания соответствующих свойств.



По умолчанию у компонентов **TextView** отсутствует фоновый цвет. Чтобы задать цвет, укажите значение **Drawable** для атрибута **android:background**. В качестве значения **Drawable** может использоваться изображение или XML-представление фигуры, включающий ресурс **Drawable** (поместить в папку **res/drawable**).

Программная установка фона

В некоторых случаях программисты из-за невнимательности неправильно меняют фон элемента программным способом и удивляются, почему ничего не работает.

Предположим, у вас определён в ресурсах зелёный цвет:

<color name="tvBackground">#337700</color>

Следующий код будет ошибочным:

tv.setBackgroundColor(R.color.tvBackground); // не работает

Нужно так (два варианта):

tv.setBackgroundResource(R.color.tvBackground); // первый вариант

tv.setBackgroundColor(getResources().getColor(R.color.tvBackground)); // второй вариант

Реагируем на событие onClick

Если вы хотите, чтобы **TextView** обрабатывал нажатия (атрибут **android:onClick**), то не забывайте также использовать в связке атрибут **android:clickable="true"**. Иначе работать не будет!

Многострочный текст

Если вы хотите создать многострочный текст в **TextView**, то используйте символы **\n** для переноса строк.

Например, в ресурсах:

<string name="about\_text">

У лукоморья дуб зелёный;\n

Златая цепь на дубе том:\n

И днём и ночью <b>кот учёный</b>\n

Всё ходит по цепи кругом;\n

Идёт <b>направо</b> - песнь заводит,\n

<b>Налево</b> - сказку говорит.</string>

Обратите внимание, что в тексте также применяется простое форматирование.

Также перенос на новую строку можно задать в коде:

textView.setText("Первая строка \nВторая строка \nТретья строка");

Увеличиваем интервалы между строками

Вы можете управлять интервалом между соседними строчками текста через атрибут **android:lineSpacingMultiplier**, который является множителем. Установите дробное значение меньше единицы, чтобы сократить интервал или больше единицы, чтобы увеличить интервал между строками.

android:lineSpacingMultiplier="0.8"

Бой с тенью

Чтобы оживить текст, можно дополнительно задействовать атрибуты для создания эффектов тени: **shadowColor**, **shadowDx**, **shadowDy** и **shadowRadius**. С их помощью вы можете установить цвет тени и ее смещение. Во время установки значений вы не увидите изменений, необходимо запустить пример в эмуляторе или на устройстве. В следующем примере я создал тень красного цвета со смещением в 2 пикселя по вертикали и горизонтали. Учтите, что для смещения используются единицы **px** (пиксели), единицы **dp** не поддерживаются.

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:padding="12dp"

android:text="Бой с тенью"

android:textSize="80sp"

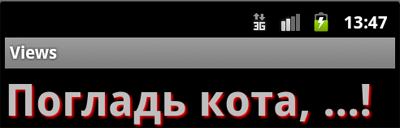
android:textStyle="bold"

android:shadowColor="#ff0000"

android:shadowDx="2"

android:shadowDy="2"

android:shadowRadius="5"/>



Программный эквивалент - метод **public void setShadowLayer (float radius, float dx, float dy, int color)**:

TextView textShadow = (TextView)findViewById(R.id.hello);

textShadow.setShadowLayer(

5f, //float radius

10f, //float dx

10f, //float dy

0xFFFFFFFF //int color

);

Создание ссылок автоматом

У **TextView** есть ещё два интересных свойства **Auto link** (атрибут **autoLink**) и **Links clickable** (атрибут **linksClickable**), которые позволяют автоматически создавать ссылки из текста.

Выглядит это следующим образом. Предположим, мы присвоим элементу **TextView** текст *Мой сайт: developer.alexanderklimov.ru* и применим к нему указанные свойства.

<TextView

android:id="@+id/textView"

android:layout\_width="match\_parent"

**android:autoLink="web"**

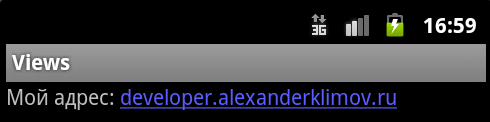
**android:linksClickable="true"**

android:text="Мой адрес: developer.alexanderklimov.ru" />

При этом уже на этапе разработки вы увидите, что строка адреса сайта после слов *Мой адрес:* стала ссылкой. Если вы запустите приложение и нажмете на ссылку, то откроется браузер с указанным адресом. Вам даже не придется писать дополнительный код. Аналогично, если указать номер телефона (параметр **phone**), то запустится звонилка.

У ссылки есть интересная особенность - при длительном нажатии на ссылку появляется диалоговое окно, позволяющее скопировать ссылку в буфер обмена.

Атрибут **autoLink** позволяет комбинировать различные виды ссылок для автоматического распознавания: веб-адрес, email, номер телефона.



Цвет ссылки можно поменять через свойство **Text color link** (XML-атрибут **textColorLink**), а программно через метод **setTextLinkColor()**.

Программно можно установить ссылки на текст через класс **Linkify**:

TextView tvDisplay = (TextView)findViewById(R.id.tvDisplay);

String data = "" +

"Пример использования Linkify для создания ссылок в тексте.\n" +

"\n" +

"URL: http://developer.alexanderklimov.ru/ \n" +

"Email: barsik@yanesobaka.com \n" +

"Телефон: (495)-458-58-29 \n" +

"Адрес: 10110 ул.Котовского, г.Мышкин \n" +

"\n" +

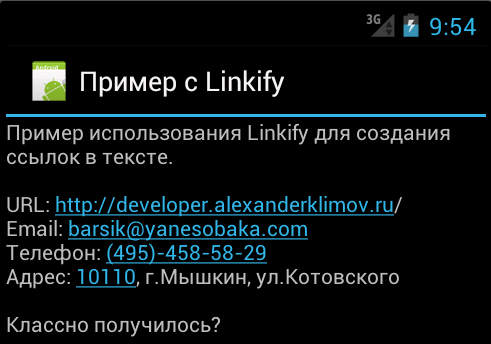
"Классно получилось?";

if(tvDisplay != null) {

tvDisplay.setText(data);

Linkify.addLinks(tvDisplay, Linkify.ALL);

}



Кроме константы **ALL**, можно также использовать **Linkify.EMAIL\_ADDRESSES**, **Linkify.MAP\_ADDRESSES**, **Linkify.PHONE\_NUMBERS**. К сожалению, русские адреса не распознаются. В моём случае индекс был распознан как телефонный номер, а город и улица не стали ссылкой.

В таких случаях придётся самостоятельно добавить ссылки в текстах. Например, определим ссылку в ресурсе:

<string name="my\_site"><a href="http://developer.alexanderklimov.ru/android">Самый лучший сайт про android</a></string>

Присвоим созданный ресурс тексту в **TextView** и запустим пример. Сам текст будет выглядеть как ссылка, но реагировать не будет. Чтобы исправить данную проблему, добавим код:

TextView textView = (TextView) findViewById(R.id.textView);

textView.setMovementMethod(LinkMovementMethod.getInstance());

Совет: Используйте полупрозрачность с умом

Если вам нужно установить текст полупрозрачным, то не используйте атрибут **android:alpha**:

<TextView

android:textColor="#fff"

android:alpha="0.5" />

Дело в том, что такой подход затрачивает много ресурсов при перерисовке.

Атрибут **textColor** позволяет установить полупрозрачность без потери производительности:

<TextView

android:textColor="80ffffff" />

Выделить текст для копирования

По умолчанию, текст в **TextView** нельзя выделить для копирования. Но в API 11 появилась такая возможность, которая может пригодиться. Делается либо при помощи XML-атрибута**android:textIsSelectable**, либо через метод **setTextIsSelectable()**.

Добавьте в разметку два компонента **TextView** и одно текстовое поле **EditText** для вставки скопированного текста. У первой текстовой метки установим возможность выделения текста декларативно.

<TextView

android:id="@+id/textView"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Выдели слово Кот для проверки"

**android:textIsSelectable="true"**

android:textSize="26sp"/>

Для второго компонента возможность выделения создадим программно.

TextView secondTextView = (TextView) findViewById(R.id.textView2);

secondTextView.setTextIsSelectable(true);

Сделайте долгий тап на тексте в любом **TextView**. Увидите стандартные ползунки для выбора длины текста. Скопируйте текст, сделайте длинный тап в **EditText** и вставьте текст.

Дополнительное чтение

[Используем собственные шрифты](http://developer.alexanderklimov.ru/android/theory/assets.php#customfont)

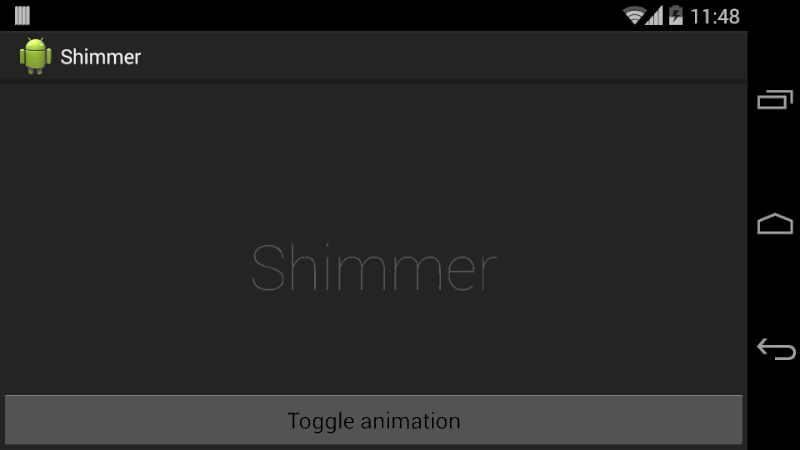
[Spannable](http://developer.alexanderklimov.ru/android/theory/spannable.php)

[TextUtils](http://developer.alexanderklimov.ru/android/theory/textutils.php)

[Продвинутые примеры с TextView](http://developer.alexanderklimov.ru/android/views/textview-advanced.php) (Закрытая зона/3-й месяц)

Библиотеки

[RomainPiel/Shimmer-android](https://github.com/RomainPiel/Shimmer-android) - мерцающий текст.



Реклама