**Drawable Фигуры и градиенты**

**Shape и ShapeDrawable**

Фигуры являются подмножеством [Drawable](http://developer.alexanderklimov.ru/android/theory/drawable.php)-ресурсов.

Данный вид ресурсов на основе класса **ShapeDrawable** позволяет описывать простые геометрические фигуры, указывая их размеры, фон и контур с помощью тега **<shape>**.

Можно создавать ресурсы фигур на основе стандартных фигур вроде прямоугольника, эллипса, линии. Для использования ресурсов фигур нужно создать в подкаталоге **res/drawable** XML-файл, в котором будет присутствовать тег <shape>, который в свою очередь может содержать дочерние элементы <corners>, <gradient>, <padding>, <size>, <solid>, <stroke>.

Имя файла без расширения будет служить идентификатором (ID): R.drawable.filename в Java-коде и @[package:]drawable/filename в XML-файлах.

**Элементы фигуры**

**<shape>**

Фигура. Является корневым элементом в XML.  
Атрибуты:

* **xmlns:android** - обязательный атрибут со строкой "http://schemas.android.com/apk/res/android"
* **android:shape** - задаёт тип фигуры: **rectangle** (прямоугольник, заполняющий элемент, является фигурой по умолчанию), **oval** (овал), **line** (линия, требуется также наличие элемента <stroke> для задания ширины линии), **ring** (окружность, для данной фигуры можно использовать атрибуты **android:innerRadius, android:innerRadiusRatio, android:thickness, android:thicknessRatio, android:useLevel**)
* **<corners>** - создаёт закругленные углы для фигуры. Только для прямоугольника. Возможные атрибуты: **android:radius, android:topLeftRadius, android:topRightRadius, android:bottomLeftRadius, android:bottomRightRadius**
* **<gradient>** - задаёт градиентную заливку для фигуры. Возможные атрибуты: **android:angle, android:centerX, android:centerY, android:centerColor, android:endColor, android:gradientRadius, android:startColor, android:type, android:useLevel**
* **<padding>** - отступы. Возможные атрибуты: **android:left, android:top, android:right, android:bottom**
* **<size>** - размеры фигуры. Возможные атрибуты: **android:height, android:width**
* **<solid>** - сплошной цвет для фигуры. Возможные атрибуты: **android:color**
* **<stroke>** - контур фигуры. Возможные атрибуты: **android:width**, **android:color**, **android:dashGap** (расстояние между черточками), **android:dashWidth** (длина пунктирной черточки)

**rectangle (Прямоугольник)**

rectangle

shape\_rect.xml - Атрибут **android:shape** здесь необязателен: **rectangle** — это значение по умолчанию.

<selector

xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape android:shape="rectangle">

<solid android:color="#0377BE"/>

</shape>

</item>

</selector>

Пример с градиентным прямоугольником в качестве разделителя

Создадим файл **separator.xml**:

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android"

android:shape="rectangle" >

<gradient

android:angle="0"

android:centerColor="#47CF4F"

android:endColor="#000000"

android:startColor="#000000" />

</shape>

В разметке приложения добавим код:

<TextView

android:id="@+id/tvSource"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="TextView" />

<View

android:layout\_width="wrap\_content"

android:background="@drawable/separator"

android:layout\_height="1dp" />

<TextView

android:id="@+id/tvResult"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="TextView" />

<View

android:layout\_width="wrap\_content"

android:background="@drawable/separator"

android:layout\_height="3dp" />

<Button

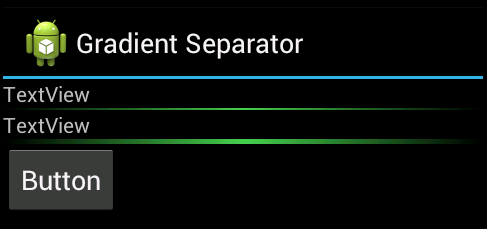
android:id="@+id/button1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button" />

У первого разделителя ширина 1dp, у второго - 3dp. Получили красивую полоску.



У прямоугольников можно скруглить углы при помощи тега **corners**

Rounded rectangle

rectangle\_rounded\_all.xml

<?xml version="1.0" encoding="utf-8"?>

<selector

xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape>

<solid android:color="#B902B0"/>

<corners android:radius="10.0dip" />

</shape>

</item>

</selector>

Можно закруглить углы по отдельности:

rectangles

**rectangle\_rounded\_some.xml**

<?xml version="1.0" encoding="utf-8"?>

<selector

xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape>

<solid android:color="#EC6118"/>

<corners android:bottomRightRadius="0.1dp"

android:bottomLeftRadius="7dip"

android:topLeftRadius="7dip"

android:topRightRadius="0.1dp"/>

</shape>

</item>

</selector>

**oval (Эллипс)**

**shape\_oval.xml**

oval

<?xml version="1.0" encoding="utf-8"?>

<selector

xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape android:shape="oval">

<solid android:color="#FCD366"/>

</shape>

</item>

</selector>

Другой вариант с пунктиром:

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android"

android:shape="oval" >

<gradient

android:centerColor="#FFFF00"

android:centerX="0.5"

android:centerY="0.5"

android:endColor="#00FF00"

android:gradientRadius="100"

android:startColor="#FF0000"

android:type="radial" />

<stroke

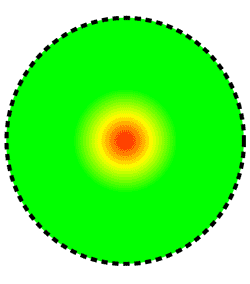
android:dashGap="6dip"

android:dashWidth="8dip"

android:width="5dip"

android:color="#000000" />

</shape>



**ring (Кольцо)**

ring

**shape\_ring.xml** - Для кольца имеются дополнительные атрибуты:

**innerRadius**

Внутренний радиус

**innerRadiusRatio**

Отношение между внешним и внутренним радиусами. По умолчанию равно 3

**thickness**

Толщина кольца (т.е. разница между внешним и внутренним радиусами)

**thicknessRatio**

Отношение ширины кольца к его толщине. По умолчанию равно 9

<?xml version="1.0" encoding="utf-8"?>

<selector

xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape android:shape="ring" android:innerRadiusRatio="3"

android:thicknessRatio="5.333">

<solid android:color="#7DBE15"/>

</shape>

</item>

</selector>

**line (Горизонтальная линия)**

line

**shape\_line.xml** - Линия может быть только горизонтальной

<?xml version="1.0" encoding="utf-8"?>

<selector xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape android:shape="line">

<stroke android:width="1dip" android:color="#F20107" />

</shape>

</item>

</selector>

**Градиенты: gradient и GradientDrawable**

Тег **gradient** (класс **GradientDrawable**) позволяет создавать сложные градиентные заливки. Каждый градиент описывает плавный переход между двумя или тремя цветами с помощью линейного/радиального алгоритма или же используя метод развертки.

Тег **gradient** внутри тега **shape**. Основные атрибуты: **type**, **startColor** (обязателен), **endColor** (обязателен) и **middleColor** (необязателен). Также иногда оказывается полезным атрибут**centerColor**.

Используя атрибут **type**, вы можете описать свой градиент:

**linear**



**gradient\_linear.xml**

* **android:type="linear"** можно опустить, он так и есть по умолчанию. Отображает прямой переход от цвета **startColor** к цвету **endColor** под углом, заданным в атрибуте **angle**.
* Атрибут **android:angle** используется только линейным градиентом и должен быть кратным значению 45.

<?xml version="1.0" encoding="utf-8"?>

<selector

xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape>

<gradient android:type="linear" android:endColor="#3C0000" android:startColor="#FF0202" android:angle="45.0" />

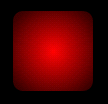
<corners android:radius="10.0dip" />

</shape>

</item>

</selector>

**radial**



* android:gradientRadius является обязательным для радиального градиента, а у остальных игнорируется. Рисует круговой градиент, начиная с цвета startColor и заканчивая startColor, от внешнего края фигуры до ее центра. Требует атрибут gradientRadius, который указывает радиус градиентного перехода в пикселях. Поддерживаются также необязательные атрибуты centerX и centerY, описывающие сдвиг центральной точки градиента. Поскольку радиус градиента указывается в пикселах, он не будет автоматически масштабироваться при разной плотности точек на экране. Чтобы минимизировать эффект ступенчатости, необходимо указывать разные значения радиуса для дисплеев с разным разрешением.
* Имеются также атрибуты android:centerX и android:centerY, в которых можно задавать относительное (0.0 – 1.0) расположение центра градиента.

**gradient\_radial.xml**

<?xml version="1.0" encoding="utf-8"?>

<selector

xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape>

<gradient android:type="radial" android:endColor="#3C0000" android:startColor="#FF0202" android:gradientRadius="50"/>

<corners android:radius="10.0dip" />

</shape>

</item>

</selector>

**sweep**

Рисует развёрточный градиент с помощью перехода между цветами **startColor** и **endColor** вдоль внешнего края фигуры (как правило, кольца).



Можно использовать атрибуты **android:centerX** и **android:centerY**.

**gradient\_sweep.xml**

<?xml version="1.0" encoding="utf-8"?>

<selector

xmlns:android="http://schemas.android.com/apk/res/android">

<item>

<shape>

<gradient android:type="sweep" android:endColor="#3C0000" android:startColor="#FF0202" />

<corners android:radius="10.0dip" />

</shape>

</item>

</selector>

**gradient\_circle.xml**

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android"

android:shape="oval" >

<gradient

android:centerColor="#0f0"

android:endColor="#f00"

android:startColor="#f00"

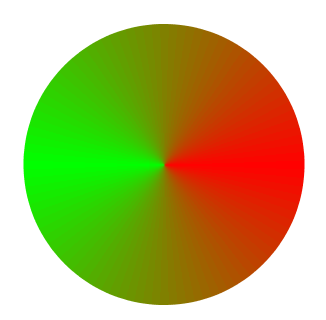
android:type="sweep" />

<size

android:height="400dp"

android:width="400dp" />

</shape>



**Примеры с shape**

Закругляем уголки у компонентов

Создадим отдельный файл **res/drawable/roundrect.xml** и с его помощью скруглим уголки у LinearLayout, ImageView, TextView, EditText:

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android"

android:shape="rectangle" >

<corners android:radius="15dp" />

<gradient

android:angle="90"

android:endColor="#FFFFFFFF"

android:startColor="#FF000000"

android:type="linear" />

<stroke

android:width="1dp"

android:color="#FF000000" />

<padding

android:bottom="15dp"

android:left="15dp"

android:right="15dp"

android:top="15dp" />

</shape>

В разметке активности пишем следующее:

<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:layout\_margin="10dp"

android:background="@drawable/roundrect"

android:orientation="vertical"

tools:context=".MainActivity" >

<ImageView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:background="@drawable/roundrect"

android:src="@drawable/ic\_launcher"

android:contentDescription="Round Image View"

tools:ignore="HardcodedText" />

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_margin="2dp"

android:background="@drawable/roundrect"

android:text="Я - TextView"

tools:ignore="HardcodedText" />

<EditText

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_margin="2dp"

android:background="@drawable/roundrect"

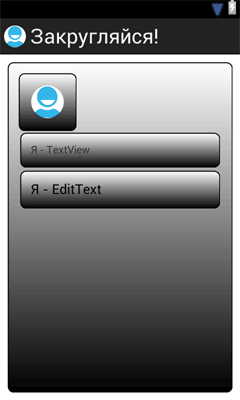
android:hint="EditText with roundrect"

android:text="Я - EditText"

tools:ignore="HardcodedText" />

</LinearLayout>

Результат:



**Овальный кабинет**

В Белом доме есть Овальный кабинет. Если вам придётся писать приложение для администрации президента США, то все элементы нужно сделать овальными. Создадим файл**res/drawable/oval.xml**:

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android"

android:shape="oval" >

<gradient

android:angle="90"

android:endColor="#FFFFFFFF"

android:startColor="#FF000000"

android:type="linear" />

<stroke

android:width="1dp"

android:color="#FF000000" />

<padding

android:bottom="15dp"

android:left="15dp"

android:right="15dp"

android:top="15dp" />

</shape>

Заменим в предыдущем примере **android:background="@drawable/roundrect"** на **android:background="@drawable/oval"**.

Получим овальный интерфейс:

