Класс ColorMatrix и фильтр ColorMatrixColorFilter

[Фильтр ColorMatrixColorFilter](http://developer.alexanderklimov.ru/android/catshop/colormatrix.php#colormatrixcolorfilter)  
[Метод setScale()](http://developer.alexanderklimov.ru/android/catshop/colormatrix.php#setscale)  
[Метод setSaturation()](http://developer.alexanderklimov.ru/android/catshop/colormatrix.php#setsaturation)  
[Метод setRotate()](http://developer.alexanderklimov.ru/android/catshop/colormatrix.php#setrotate)  
[Метод setConcat()](http://developer.alexanderklimov.ru/android/catshop/colormatrix.php#setconcat)

Фильтр ColorMatrixColorFilter

Фильтр для изменения цвета, использующий значения 4х5 матрицы.

Матрица может иметь следующий вид: 4 строки, в каждой строке по 5 значений:

rR, rG, rB, rA, rT

gR, gG, gB, gA, gT

bR, bG, bB, bA, bT

aR, aG, aB, aA, aT

Допустим, у нас есть некий цвет ARGB. Применим к нему фильтр. Фильтр возьмёт текущее значение цвета и, используя матрицу, вычислит новые значения. Например, новое значение красного (Rn) он посчитает так:

New Red Value = R \* rR + G \* rG + B \* rB + A \* rA + rT

Т.е. значения исходного цвета (R,G,B,A) перемножаем на первые 4 значения (rR, rG, rB, rA) из первой строки матрицы и прибавляем пятое значение (rT) из этой же строки.

Самостоятельно считать ничего не нужно, фильтр сам всё рассчитает. Вы должны только эту матрицу предоставить.

Новое значение зелёного получается аналогично, используя исходные значения RGBA и вторую строку матрицы.

New Green Value = R \* gR + G \* gG + B \* gB + A \* gA + gT

Синий и прозрачность – третья и четвертая строки:

New Blue Value = R \* bR + G \* bG + B \* bB + A \* bA + bT

New Alpha Value = R \* aR + G \* aG + B \* aB + A \* aA + aT

Создать объект **ColorMatrix** можно, используя конструктор без аргументов или перегруженные версии с аргументами.

ColorMatrix colorMatrix = new ColorMatrix();

Созданный объект применяется к объекту **Paint** через фильтр **ColorMatrixColorFilter**:

paint.setColorFilter(new ColorMatrixColorFilter(colorMatrix));

Конструктору без аргументов соответствует матрица:

1 0 0 0 0

0 1 0 0 0

0 0 1 0 0

0 0 0 1 0

Если применить данную матрицу к цвету, то все цвета останутся без изменений.

Для наглядности нарисуем несколько разноцветных квадратиков и один значок, и будем применять к ним фильтры (идея взята с сайта, ссылка на которую приведена в конце статьи).

package ru.alexanderklimov.test;

import android.app.Activity;

import android.content.Context;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Canvas;

import android.graphics.Color;

import android.graphics.ColorFilter;

import android.graphics.ColorMatrix;

import android.graphics.ColorMatrixColorFilter;

import android.graphics.Paint;

import android.graphics.Rect;

import android.os.Bundle;

import android.view.View;

public class MainActivity extends Activity {

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(new DrawView(this));

}

class DrawView extends View {

private Paint mPaint;

private Rect mRect;

private Bitmap mIcon;

private ColorMatrix mColorMatrix;

private ColorFilter mFilter;

// Матрица

float[] cmData = new float[] {

1, 0, 0, 0, 0,

0, 1, 0, 0, 0,

0, 0, 1, 0, 0,

0, 0, 0, 1, 0 };

public DrawView(Context context) {

super(context);

mRect = new Rect(0, 0, 150, 150);

mPaint = new Paint(Paint.ANTI\_ALIAS\_FLAG);

mPaint.setStyle(Paint.Style.FILL\_AND\_STROKE);

mIcon = BitmapFactory.decodeResource(context.getResources(),

R.drawable.ic\_android\_cat);

mColorMatrix = new ColorMatrix(cmData);

mFilter = new ColorMatrixColorFilter(mColorMatrix);

}

@Override

protected void onDraw(Canvas canvas) {

canvas.drawARGB(80, 102, 204, 255);

canvas.translate(100, 100);

// Выводим объекты

drawObjects(canvas);

// Применяем фильтр

mPaint.setColorFilter(mFilter);

// Выводим эти же объекты чуть ниже с применённым фильтром

canvas.translate(0, 300);

drawObjects(canvas);

}

void drawObjects(Canvas canvas) {

canvas.save();

// Рисуем красный квадрат

mPaint.setColor(Color.RED);

canvas.drawRect(mRect, mPaint);

// Рисуем зелёный квадрат

mPaint.setColor(Color.GREEN);

canvas.translate(220, 0);

canvas.drawRect(mRect, mPaint);

// Рисуем синий квадрат

mPaint.setColor(Color.BLUE);

canvas.translate(220, 0);

canvas.drawRect(mRect, mPaint);

// Рисуем белый квадрат

mPaint.setColor(Color.WHITE);

canvas.translate(220, 0);

canvas.drawRect(mRect, mPaint);

// Выводим значок

canvas.translate(220, 0);

canvas.drawBitmap(mIcon, null, mRect, mPaint);

canvas.restore();

}

}

}

Массив **cmData** содержит значения для матрицы, который мы передаём экземпляру класса **ColorMatrix** - переменной **mColorMatrix**.

Далее подготовленную матрицу передаём фильтру **ColorMatrixColorFilter**. Для применения фильтра используется метод **setColorFilter()**.

В нашем примере фильтр применяется к кисти и мы можем наблюдать за изменениями цветов выводимых фигур.

Сейчас матрица выглядит следующим образом:

1, 0, 0, 0, 0,

0, 1, 0, 0, 0,

0, 0, 1, 0, 0,

0, 0, 0, 1, 0

Если мы возьмем RGBA и применим матрицу, получим следующие результаты:

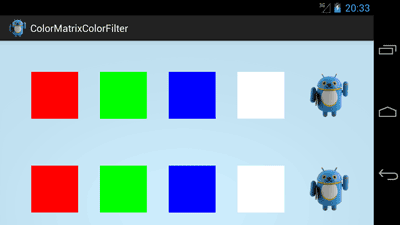
Rn = R \* 1 + G \* 0 + B \* 0 + A \* 0 + 0 = R

Gn = R \* 0 + G \* 1 + B \* 0 + A \* 0 + 0 = G

Bn = R \*0 + G \* 0 + B \* 1 + A \* 0 + 0 = B

An = R \* 0 + G \* 0 + B \* 0 + A \* 1 + 0 = A

Как видите, новые значения равны исходным. Матрица составлена таким образом, что RGBA-значения любого цвета не изменяются. Запустив пример, вы увидите, что первый и второй ряд фигур идентичны.



Поменяем матрицу:

float[] cmData = new float[]{

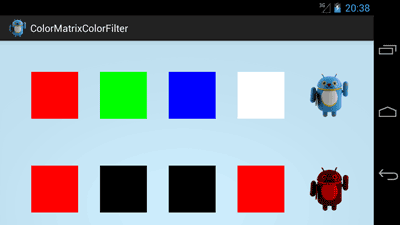
1, 0, 0, 0, 0,

0, 0, 0, 0, 0,

0, 0, 0, 0, 0,

0, 0, 0, 1, 0};

Новый результат.



Изучим, как происходят изменения цвета.

Был красный цвет c RGBA = (255,0,0,255). Применим фильтр:

Rn = 255 \* 1 + 0 \* 0 + 0 \* 0 + 255 \* 0 + 0 = 255

Gn = 255 \* 0 + 0 \* 0 + 0 \* 0 + 255 \* 0 + 0 = 0

Bn = 255 \* 0 + 0 \* 0 + 0 \* 0 + 255 \* 0 + 0 = 0

An = 255 \* 0 + 0 \* 0 + 0 \* 0 + 255 \* 1 + 0 = 255

Новые RGBA-значения получились такими (255,0,0,255) и они совпадают с исходным красным цветом. Поэтому красный квадрат остался без изменений.

Возьмём зелёный цвет. Его значение равно (0,255,0,255). Применяем фильтр:

Rn = 0 \* 1 + 255 \* 0 + 0 \* 0 + 255 \* 0 + 0 = 0

Gn = 0 \* 0 + 255 \* 0 + 0 \* 0 + 255 \* 0 + 0 = 0

Bn = 0 \* 0 + 255 \* 0 + 0 \* 0 + 255 \* 0 + 0 = 0

An = 0 \* 0 + 255 \* 0 + 0 \* 0 + 255 \* 1 + 0 = 255

Новые RGBA-значения зеленого = (0,0,0,255), а это чёрный цвет. Вот почему зелёный квадрат стал чёрным.

Такой же результат получается с синим цветом.

А белый цвет (255,255,255,255) после преобразования получит следующие значения:

Rn = 255 \* 1 + 255 \* 0 + 255 \* 0 + 255 \* 0 + 0 = 255

Gn = 255 \* 0 + 255 \* 0 + 255 \* 0 + 255 \* 0 + 0 = 0

Bn = 255 \* 0 + 255 \* 0 + 255 \* 0 + 255 \* 0 + 0 = 0

An = 255 \* 0 + 255 \* 0 + 255 \* 0 + 255 \* 1 + 0 = 255

Таким образом, белый цвет (255,255,255,255) трансформируется в красный цвет (255,0,0,255).

Применив данный фильтр, мы для всех цветов «обнулили» значения синего (B) и зеленого (G) составляющих и оставили только красную (R) составляющую. Это видно на значке.

Но тут важно понимать одну вещь. Мы не выкрасили всё в красный цвет. Мы полностью убрали зеленый и синий, а красный оставили в том значении, в каком он был. В красном квадрате значение красного было 255. Таким и осталось. В синем и зелёном квадратах значение красного было 0. Таким и осталось.

Если рассматривать значок, видно, что красный неоднородный, где-то светлее, где-то темнее. Т.е. изначально значок была нарисован разными оттенками, в которых были использованы различные RGB-комбинации. Мы в этих комбинациях убрали G и B, оставили только R. Где R был, например 50, остался 50. Где был 150 – остался 150. А G и B теперь везде равен 0.

Давайте настроим матрицу так, чтобы красный везде стал максимальным, независимо от первоначального значения. А синий и зеленый снова обнулим.

float[] cmData = new float[]{

0, 0, 0, 0, 255,

0, 0, 0, 0, 0,

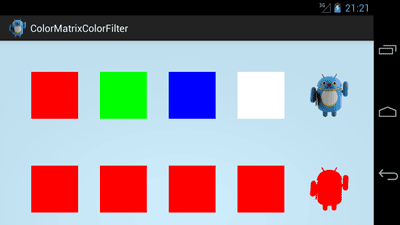
0, 0, 0, 0, 0,

0, 0, 0, 1, 0};

Мы убрали коэффициент 1 из первого числа первой строки. Теперь новое значение R уже не будет равно старому значение R, умноженному на 1. Теперь оно будет умножаться на 0, но последнее число первой строки равно 255. Оно будет прибавлено к нулю и мы получим полный красный цвет на замену первоначальным оттенкам красного.

Синий и зеленый также станут красными, так как G и B значения мы в них обнулим, а R будет равен 255, т.е. (255,0,0,255).

Результат



Теперь изменим матрицу так, чтобы обнулялся только синий компонент. Красный и зелёный компоненты останутся неизменными.

float[] cmData = new float[]{

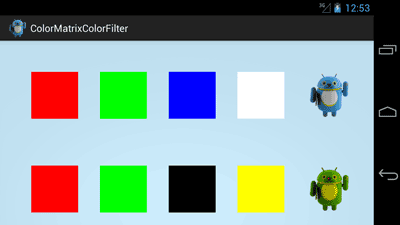
1, 0, 0, 0, 0,

0, 1, 0, 0, 0,

0, 0, 0, 0, 0,

0, 0, 0, 1, 0};

Получим следующий результат.



Убрав из синего цвета (0,0,255,255) синий компонент, мы получили чёрный цвет (0,0,0,255).

А убрав из белого цвета (255,255,255,255) синий компонент, мы получили жёлтый цвет (255,255,0,255).

Мы меняли компоненты цветов (RGB), теперь давайте попробуем поменять прозрачность (A). Напомню, что если A = 255, то цвет абсолютно непрозрачен. Если A = 0, то цвет совсем не виден.

float[] cmData = new float[]{

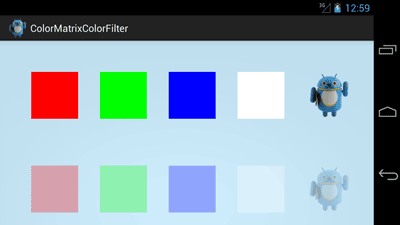
1, 0, 0, 0, 0,

0, 1, 0, 0, 0,

0, 0, 1, 0, 0,

0, 0, 0, 0.3f, 0};

Мы поставили коэффициент 0.3 для вычисления нового значения прозрачности (An = A \* 0.3). Теперь все цвета станут прозрачными на 30% от текущего уровня.



Можно самостоятельно придумывать собственные комбинации для матрицы. Существуют уже готовые решения. Например, преобразование в чёрно-белый цвет.

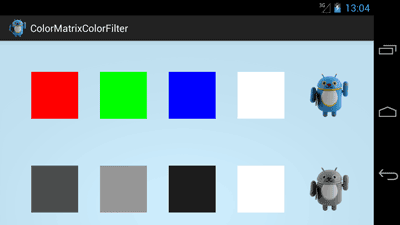
float[] cmData = new float[]{

0.3f, 0.59f, 0.11f, 0, 0,

0.3f, 0.59f, 0.11f, 0, 0,

0.3f, 0.59f, 0.11f, 0, 0,

0, 0, 0, 1, 0,};



Вот ещё одна комбинация, дающая серый цвет.

float[] cmData = new float[]{

0.213f, 0.715f, 0.072f, 0.0f, 0.0f,

0.213f, 0.715f, 0.072f, 0.0f, 0.0f,

0.213f, 0.715f, 0.072f, 0.0f, 0.0f,

0.0f, 0.0f, 0.0f, 1.0f, 0.0f,

};

Как видите, можно играться настройками и добиваться нужных результатов.

Инвертирование цветов

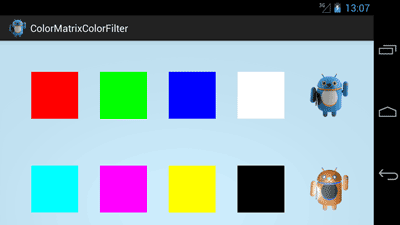
float[] cmData = new float[]{

-1, 0, 0, 0, 255,

0, -1, 0, 0, 255,

0, 0, -1, 0, 255,

0, 0, 0, 1, 0,};



Метод setScale()

У **ColorMatrix** есть специальный метод **setScale()**, который позволяет указать на какие значения необходимо умножать RGBA значения цвета. В этом случае вам не нужно составлять свою матрицу, а просто указать в параметрах метода коэффициенты для R, G, B и A компонентов. Сама матрица будет иметь вид из первого примера, когда после её применения ничего не изменилось.

Перепишем конструктор

public DrawView(Context context) {

super(context);

mRect = new Rect(0, 0, 150, 150);

mPaint = new Paint(Paint.ANTI\_ALIAS\_FLAG);

mPaint.setStyle(Paint.Style.FILL\_AND\_STROKE);

mIcon = BitmapFactory.decodeResource(context.getResources(),

R.drawable.ic\_android\_cat);

mColorMatrix = new ColorMatrix();

mColorMatrix.setScale(1, 0, 0, 1);

mFilter = new ColorMatrixColorFilter(mColorMatrix);

}

Вместо **new ColorMatrix(cmData)** используем пустой конструктор без указания матрицы **new ColorMatrix()**. А далее вызываем метод, в котором указываем нужные коэффициенты. Параметры (1, 0, 0, 1) соответствуют матрице из второго примера:

float[] cmData = new float[]{

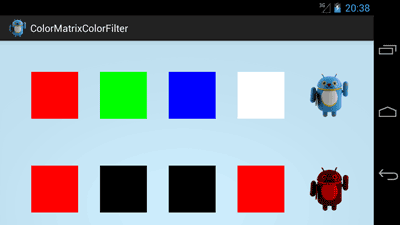
1, 0, 0, 0, 0,

0, 0, 0, 0, 0,

0, 0, 0, 0, 0,

0, 0, 0, 1, 0};

И мы видим тот же результат.



Если нужно сделать цвета полупрозрачными для нашего примера, то код был бы:

mColorMatrix.setScale(1, 0, 0, 0.5f);

Проверьте самостоятельно.

Напишем отдельный пример для загрузки изображения из Галереи и применим к нему метод **setScale()**.

Разметка

<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"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

tools:context=".MainActivity" >

<Button

android:id="@+id/buttonLoadimage"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Load Image" />

<ScrollView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" >

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="vertical" >

<ImageView

android:id="@+id/imageView"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:adjustViewBounds="true"

android:background="@drawable/ic\_android\_cat"

android:scaleType="centerInside" />

<TextView

android:id="@+id/textViewCMScale"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="ColorMatrix Scale" />

<SeekBar

android:id="@+id/seekBarRedScale"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:max="200"

android:progress="100" />

<SeekBar

android:id="@+id/seekBarGreenScale"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:max="200"

android:progress="100" />

<SeekBar

android:id="@+id/seekBarBlueScale"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:max="200"

android:progress="100" />

<SeekBar

android:id="@+id/seekBarAlphaScale"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:max="200"

android:progress="100" />

</LinearLayout>

</ScrollView>

</LinearLayout>

У компонента **ImageView** задан в качестве фона один из рисунков. При загрузке из Галереи новая картинка закроет фоновый рисунок. Но поменяв прозрачность, вы сможете увидеть фон снова.

Код для активности

package ru.alexanderklimov.test;

import java.io.FileNotFoundException;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Canvas;

import android.graphics.ColorMatrix;

import android.graphics.ColorMatrixColorFilter;

import android.graphics.Paint;

import android.net.Uri;

import android.os.Bundle;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.SeekBar;

import android.widget.SeekBar.OnSeekBarChangeListener;

import android.widget.TextView;

public class MainActivity extends Activity {

final int RQS\_IMAGE = 1;

private Button mLoadImageButton;

private ImageView mGaleryImageView;

private SeekBar mRedSeekbar, mGreenSeekbar, mBlueSeekbar, mAlphaSeekbar;

private TextView mScaleTextView;

private Uri mSource;

private Bitmap mBitmap;

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mLoadImageButton = (Button) findViewById(R.id.buttonLoadimage);

mGaleryImageView = (ImageView) findViewById(R.id.imageView);

mLoadImageButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(

Intent.ACTION\_PICK,

android.provider.MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI);

startActivityForResult(intent, RQS\_IMAGE);

}

});

mScaleTextView = (TextView) findViewById(R.id.textViewCMScale);

mRedSeekbar = (SeekBar) findViewById(R.id.seekBarRedScale);

mGreenSeekbar = (SeekBar) findViewById(R.id.seekBarGreenScale);

mBlueSeekbar = (SeekBar) findViewById(R.id.seekBarBlueScale);

mAlphaSeekbar = (SeekBar) findViewById(R.id.seekBarAlphaScale);

mRedSeekbar.setOnSeekBarChangeListener(seekBarChangeListener);

mGreenSeekbar.setOnSeekBarChangeListener(seekBarChangeListener);

mBlueSeekbar.setOnSeekBarChangeListener(seekBarChangeListener);

mAlphaSeekbar.setOnSeekBarChangeListener(seekBarChangeListener);

}

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent data) {

super.onActivityResult(requestCode, resultCode, data);

if (resultCode == RESULT\_OK) {

switch (requestCode) {

case RQS\_IMAGE:

mSource = data.getData();

try {

mBitmap = BitmapFactory

.decodeStream(getContentResolver().openInputStream(

mSource));

mRedSeekbar.setProgress(100);

mGreenSeekbar.setProgress(100);

mBlueSeekbar.setProgress(100);

mAlphaSeekbar.setProgress(100);

loadBitmapScaleColor();

} catch (FileNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

}

OnSeekBarChangeListener seekBarChangeListener = new OnSeekBarChangeListener() {

@Override

public void onProgressChanged(SeekBar seekBar, int progress,

boolean fromUser) {

// TODO Auto-generated method stub

}

@Override

public void onStartTrackingTouch(SeekBar seekBar) {

// TODO Auto-generated method stub

}

@Override

public void onStopTrackingTouch(SeekBar seekBar) {

loadBitmapScaleColor();

}

};

private void loadBitmapScaleColor() {

if (mBitmap != null) {

int progressScaleRed = mRedSeekbar.getProgress();

int progressScaleGreen = mGreenSeekbar.getProgress();

int progressScaleBlue = mBlueSeekbar.getProgress();

int progressScaleAlpha = mAlphaSeekbar.getProgress();

float scaleRed = (float) progressScaleRed / 100;

float scaleGreen = (float) progressScaleGreen / 100;

float scaleBlue = (float) progressScaleBlue / 100;

float scaleAlpha = (float) progressScaleAlpha / 100;

mScaleTextView.setText("setScale: " + String.valueOf(scaleRed) + ", "

+ String.valueOf(scaleGreen) + ", "

+ String.valueOf(scaleBlue) + ", "

+ String.valueOf(scaleAlpha));

Bitmap bitmapColorScaled = updateScale(mBitmap, scaleRed,

scaleGreen, scaleBlue, scaleAlpha);

mGaleryImageView.setImageBitmap(bitmapColorScaled);

}

}

private Bitmap updateScale(Bitmap src, float rScale, float gScale,

float bScale, float aScale) {

int width = src.getWidth();

int height = src.getHeight();

Bitmap bitmapResult = Bitmap

.createBitmap(width, height, Bitmap.Config.ARGB\_8888);

Canvas canvasResult = new Canvas(bitmapResult);

Paint paint = new Paint();

ColorMatrix colorMatrix = new ColorMatrix();

colorMatrix.setScale(rScale, gScale, bScale, aScale);

ColorMatrixColorFilter filter = new ColorMatrixColorFilter(colorMatrix);

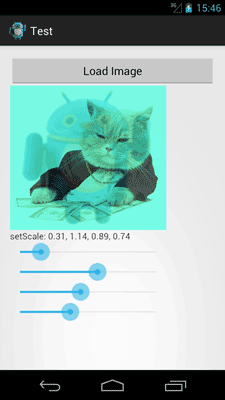
paint.setColorFilter(filter);

canvasResult.drawBitmap(src, 0, 0, paint);

return bitmapResult;

}

}



Метод setSaturation()

Метод **setSaturation()** позволяет управлять насыщенностью цвета. Принимает на вход значения от 0 до 1. Если задать 0, то получим чёрно-белую картинку.

Разметка

<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"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

tools:context=".MainActivity" >

<Button

android:id="@+id/buttonLoadimage"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Load Image" />

<ImageView

android:id="@+id/imageView"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:adjustViewBounds="true"

android:background="@android:color/background\_dark"

android:scaleType="centerInside" />

<TextView

android:id="@+id/textViewSaturation"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Saturation" />

<SeekBar

android:id="@+id/seekBarSaturation"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:max="512"

android:progress="256" />

</LinearLayout>

Загружаем из галереи рыжего бандита по кличке Рыжик и превращаем его в серого гангстера по кличке Васька.

package ru.alexanderklimov.test;

import java.io.FileNotFoundException;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Canvas;

import android.graphics.ColorMatrix;

import android.graphics.ColorMatrixColorFilter;

import android.graphics.Paint;

import android.net.Uri;

import android.os.Bundle;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.SeekBar;

import android.widget.SeekBar.OnSeekBarChangeListener;

import android.widget.TextView;

public class MainActivity extends Activity {

final int RQS\_IMAGE = 1;

private Button mLoadImageButton;

private ImageView mGaleryImageView;

private SeekBar mSaturationSeekbar;

private TextView mSaturationTextView;

private Uri mSource;

private Bitmap mBitmap;

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mLoadImageButton = (Button) findViewById(R.id.buttonLoadimage);

mSaturationTextView = (TextView) findViewById(R.id.textViewSaturation);

mGaleryImageView = (ImageView) findViewById(R.id.imageView);

mLoadImageButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(

Intent.ACTION\_PICK,

android.provider.MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI);

startActivityForResult(intent, RQS\_IMAGE);

}

});

mSaturationSeekbar = (SeekBar) findViewById(R.id.seekBarSaturation);

mSaturationSeekbar.setOnSeekBarChangeListener(seekBarChangeListener);

}

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent data) {

super.onActivityResult(requestCode, resultCode, data);

if (resultCode == RESULT\_OK) {

switch (requestCode) {

case RQS\_IMAGE:

mSource = data.getData();

try {

mBitmap = BitmapFactory.decodeStream(getContentResolver()

.openInputStream(mSource));

mSaturationSeekbar.setProgress(256);

loadSaturationBitmap();

} catch (FileNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

}

OnSeekBarChangeListener seekBarChangeListener = new OnSeekBarChangeListener() {

@Override

public void onProgressChanged(SeekBar seekBar, int progress,

boolean fromUser) {

// TODO Auto-generated method stub

}

@Override

public void onStartTrackingTouch(SeekBar seekBar) {

// TODO Auto-generated method stub

}

@Override

public void onStopTrackingTouch(SeekBar seekBar) {

loadSaturationBitmap();

}

};

private void loadSaturationBitmap() {

// TODO Auto-generated method stub

if (mBitmap != null) {

int progressSat = mSaturationSeekbar.getProgress();

// Saturation, 0=gray-scale. 1=identity

float saturation = (float) progressSat / 256;

mSaturationTextView.setText("Saturation: "

+ String.valueOf(saturation));

mGaleryImageView.setImageBitmap(updateSaturation(mBitmap,

saturation));

}

}

private Bitmap updateSaturation(Bitmap src, float settingSat) {

int width = src.getWidth();

int height = src.getHeight();

Bitmap bitmapResult = Bitmap

.createBitmap(width, height, Bitmap.Config.ARGB\_8888);

Canvas canvasResult = new Canvas(bitmapResult);

Paint paint = new Paint();

ColorMatrix colorMatrix = new ColorMatrix();

colorMatrix.setSaturation(settingSat);

ColorMatrixColorFilter filter = new ColorMatrixColorFilter(colorMatrix);

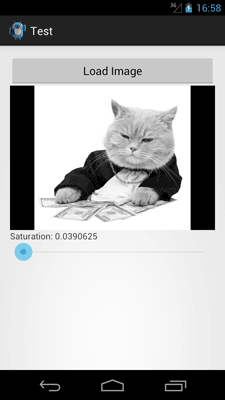
paint.setColorFilter(filter);

canvasResult.drawBitmap(src, 0, 0, paint);

return bitmapResult;

}

}



Метод setRotate()

Пример применения метода **setRotate()**.

Разметка

<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"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

tools:context=".MainActivity" >

<Button

android:id="@+id/buttonLoadimage"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Load Image" />

<ScrollView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" >

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="vertical" >

<ImageView

android:id="@+id/imageView"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:adjustViewBounds="true"

android:background="@drawable/ic\_android\_cat"

android:scaleType="centerInside" />

<TextView

android:id="@+id/textViewRotate"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Rotate Axis" />

<RadioGroup

android:id="@+id/axisgroup"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal" >

<RadioButton

android:id="@+id/radioAxisRed"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:checked="true"

android:text="Red" />

<RadioButton

android:id="@+id/radioAxisGreen"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Green" />

<RadioButton

android:id="@+id/radioAxisBlue"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Blue" />

</RadioGroup>

<SeekBar

android:id="@+id/seekBarRotate"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:max="360"

android:progress="0" />

</LinearLayout>

</ScrollView>

</LinearLayout>

Код

package ru.alexanderklimov.test;

import java.io.FileNotFoundException;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Canvas;

import android.graphics.ColorMatrix;

import android.graphics.ColorMatrixColorFilter;

import android.graphics.Paint;

import android.net.Uri;

import android.os.Bundle;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.RadioButton;

import android.widget.RadioGroup;

import android.widget.RadioGroup.OnCheckedChangeListener;

import android.widget.SeekBar;

import android.widget.SeekBar.OnSeekBarChangeListener;

import android.widget.TextView;

public class MainActivity extends Activity {

final int RQS\_IMAGE = 1;

private Button mLoadImageButton;

private ImageView mGaleryImageView;

private SeekBar mRotateSeekbar;

private TextView mRotateTextView;

private RadioGroup mAxisRadioGroup;

private RadioButton mAxisRedRadioButton, mAxisGreenRadioButton,

mAxisBlueRadioButton;

private Uri mSource;

private Bitmap mBitmap;

final int RQS\_IMAGE1 = 1;

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

setTitle("ColorMatrixColorFilter");

mLoadImageButton = (Button) findViewById(R.id.buttonLoadimage);

mGaleryImageView = (ImageView) findViewById(R.id.imageView);

mLoadImageButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(

Intent.ACTION\_PICK,

android.provider.MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI);

startActivityForResult(intent, RQS\_IMAGE);

}

});

mRotateTextView = (TextView) findViewById(R.id.textViewRotate);

mRotateSeekbar = (SeekBar) findViewById(R.id.seekBarRotate);

mRotateSeekbar.setOnSeekBarChangeListener(seekBarChangeListener);

mAxisRadioGroup = (RadioGroup) findViewById(R.id.axisgroup);

mAxisRedRadioButton = (RadioButton) findViewById(R.id.radioAxisRed);

mAxisGreenRadioButton = (RadioButton) findViewById(R.id.radioAxisGreen);

mAxisBlueRadioButton = (RadioButton) findViewById(R.id.radioAxisBlue);

mAxisRadioGroup

.setOnCheckedChangeListener(groupOnCheckedChangeListener);

}

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent data) {

super.onActivityResult(requestCode, resultCode, data);

if (resultCode == RESULT\_OK) {

switch (requestCode) {

case RQS\_IMAGE:

mSource = data.getData();

try {

mBitmap = BitmapFactory.decodeStream(getContentResolver()

.openInputStream(mSource));

mAxisRedRadioButton.setChecked(true);

mRotateSeekbar.setProgress(0);

loadRotatedBitmap();

} catch (FileNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

}

OnSeekBarChangeListener seekBarChangeListener = new OnSeekBarChangeListener() {

@Override

public void onProgressChanged(SeekBar seekBar, int progress,

boolean fromUser) {

// TODO Auto-generated method stub

}

@Override

public void onStartTrackingTouch(SeekBar seekBar) {

// TODO Auto-generated method stub

}

@Override

public void onStopTrackingTouch(SeekBar seekBar) {

loadRotatedBitmap();

}

};

OnCheckedChangeListener groupOnCheckedChangeListener = new OnCheckedChangeListener() {

@Override

public void onCheckedChanged(RadioGroup group, int checkedId) {

loadRotatedBitmap();

}

};

private void loadRotatedBitmap() {

if (mBitmap != null) {

int progressRotation = mRotateSeekbar.getProgress();

float rotationDegree = (float) progressRotation;

if (mAxisRedRadioButton.isChecked()) {

mRotateTextView.setText("setRotate: " + "Red: "

+ String.valueOf(rotationDegree));

mGaleryImageView.setImageBitmap(updateRotation(mBitmap, 0,

rotationDegree));

} else if (mAxisGreenRadioButton.isChecked()) {

mRotateTextView.setText("setRotate: " + "Green: "

+ String.valueOf(rotationDegree));

mGaleryImageView.setImageBitmap(updateRotation(mBitmap, 1,

rotationDegree));

} else if (mAxisBlueRadioButton.isChecked()) {

mRotateTextView.setText("setRotate: " + "Blue: "

+ String.valueOf(rotationDegree));

mGaleryImageView.setImageBitmap(updateRotation(mBitmap, 2,

rotationDegree));

}

}

}

private Bitmap updateRotation(Bitmap src, int axis, float degrees) {

int width = src.getWidth();

int height = src.getHeight();

Bitmap bitmapResult = Bitmap.createBitmap(width, height,

Bitmap.Config.ARGB\_8888);

Canvas canvasResult = new Canvas(bitmapResult);

Paint paint = new Paint();

ColorMatrix colorMatrix = new ColorMatrix();

colorMatrix.setRotate(axis, degrees);

ColorMatrixColorFilter filter = new ColorMatrixColorFilter(colorMatrix);

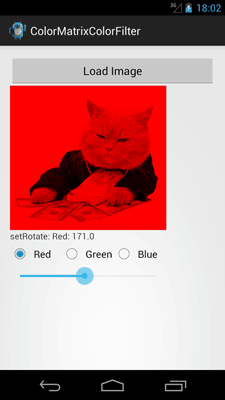
paint.setColorFilter(filter);

canvasResult.drawBitmap(src, 0, 0, paint);

return bitmapResult;

}

}



Метод setConcat()

Метод **setConcat()** использует две матрицы. Пример можно найти [здесь](http://android-er.blogspot.ru/2013/09/colormatrixsetconcat-to-combine.html).

Дополнительные материалы

[Android-er: Create Sepia bitmap using ColorMatrix](http://android-er.blogspot.ru/2015/02/create-sepia-bitmap-using-colormatrix.html)

[Android-er: Android example code using ColorFilter](http://android-er.blogspot.ru/2015/10/android-example-code-using-colorfilter.html)

[Android-er: Convert ImageView to black and white, and set brightness, using ColorFilter](http://android-er.blogspot.ru/2015/10/convert-imageview-to-black-and-white.html)