**Класс android.graphics.Color**

В графике важную роль играет цвет, который представлен классом **android.graphics.Color**. Класс **Color** содержит несколько констант и методов для конвертации и извлечения цветового компонента.

Цвета можно описывать четырьмя числами в формате ARGB, по одному для каждого канала(Alpha, Red, Green, Blue). Каждый из каналов является восьмибитным целым числом, т.е. может принимать значения от 0 до 256. Обычно цвет упаковывают в 32-битное целое число. Стоит отметить, что использовать целые числа для цветов эффективнее, чем экземпляры класса **Color**.

Если все значения RGB-каналов установить равными 0, то получим чёрный цвет, если присвоить значения 255, то получим белый цвет.

Иногда значения указываются не в десятичной форме, а в шестнадцатеричной от 00 до FF, вместо 0 и 255. В этом случае обычно пишут не ARGB, а AARRGGBB. Например, красный цвет в данном формате будет равен FFFF0000 (AA = FF, RR = FF, GG = 00, BB = 00).

Для создания цвета можно использовать статические константы класса **Color**, например:

int redcolor = Color.RED; // красный цвет

Таких констант немного, только для основных цветов.

Получить красную, зелёную и синюю составляющую цвета можно через методы **red()**, **green**, **blue()**. Получить нужный цвет из набора компонентов можно через методы **rgb()** и **argb()**, которые вернут вам int-значение цвета.

// Полупрозрачный красный

mycolor = Color.argb(127, 255, 0, 255);

Метод **parseColor()** позволяет получить int-значение из шестнадцатеричной формы.

int parsedColor = Color.parseColor("#7FFF0000");

Лучше задавать цвет в ресурсах для большей гибкости. Всегда можно будет поменять в случае необходимости без вмешательства в код программы:

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

<resources>

<color name="mycolor">#7fff0033</color>

</resources>

В коде можно обратиться к цвету следующим образом:

color = getResources().getColor(R.color.mycolor);

Создадим простейший пример для просмотра цветов.

<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" >

<Spinner

android:id="@+id/spinnerColor"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

</LinearLayout>

И код для программы:

package ru.alexanderklimov.colordemo;

// Если этот код работает, его написал Александр Климов,

// а если нет, то не знаю, кто его писал.

import android.app.Activity;

import android.graphics.Color;

import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.LinearLayout;

import android.widget.Spinner;

import android.widget.Toast;

public class MainActivity extends Activity {

private LinearLayout mBackgroundLinearLayout;

private static final String[] sColor = { "BLACK", "BLUE", "CYAN", "DKGRAY",

"GRAY", "GREEN", "LTGRAY", "MAGENTA", "RED", "TRANSPARENT",

"WHITE", "YELLOW" };

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

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mBackgroundLinearLayout = (LinearLayout) findViewById(R.id.linearLayout);

final Spinner colorSpinner = (Spinner) findViewById(R.id.spinnerColor);

ArrayAdapter<String> adapter = new ArrayAdapter<String>(getApplicationContext(),

android.R.layout.simple\_spinner\_item, sColor);

adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

colorSpinner.setAdapter(adapter);

colorSpinner.setSelection(0);

colorSpinner

.setOnItemSelectedListener(new Spinner.OnItemSelectedListener() {

@Override

public void onItemSelected(AdapterView<?> arg0, View arg1,

int arg2, long arg3) {

// TODO Auto-generated method stub

setBackgroundColor(colorSpinner.getSelectedItem()

.toString());

}

@Override

public void onNothingSelected(AdapterView<?> arg0) {

// TODO Auto-generated method stub

}

});

}

private void setBackgroundColor(String strColor) {

Toast.makeText(this, strColor, Toast.LENGTH\_SHORT).show();

if (strColor.equalsIgnoreCase("BLACK")) {

mBackgroundLinearLayout.setBackgroundColor(Color.BLACK);

} else if (strColor.equalsIgnoreCase("BLUE")) {

mBackgroundLinearLayout.setBackgroundColor(Color.BLUE);

} else if (strColor.equalsIgnoreCase("CYAN")) {

mBackgroundLinearLayout.setBackgroundColor(Color.CYAN);

} else if (strColor.equalsIgnoreCase("DKGRAY")) {

mBackgroundLinearLayout.setBackgroundColor(Color.DKGRAY);

} else if (strColor.equalsIgnoreCase("GRAY")) {

mBackgroundLinearLayout.setBackgroundColor(Color.GRAY);

} else if (strColor.equalsIgnoreCase("GREEN")) {

mBackgroundLinearLayout.setBackgroundColor(Color.GREEN);

} else if (strColor.equalsIgnoreCase("LTGRAY")) {

mBackgroundLinearLayout.setBackgroundColor(Color.LTGRAY);

} else if (strColor.equalsIgnoreCase("MAGENTA")) {

mBackgroundLinearLayout.setBackgroundColor(Color.MAGENTA);

} else if (strColor.equalsIgnoreCase("RED")) {

mBackgroundLinearLayout.setBackgroundColor(Color.RED);

} else if (strColor.equalsIgnoreCase("TRANSPARENT")) {

mBackgroundLinearLayout.setBackgroundColor(Color.TRANSPARENT);

} else if (strColor.equalsIgnoreCase("WHITE")) {

mBackgroundLinearLayout.setBackgroundColor(Color.WHITE);

} else if (strColor.equalsIgnoreCase("YELLOW")) {

mBackgroundLinearLayout.setBackgroundColor(Color.YELLOW);

}

}

}

Запустив проект, вы можете выбирать из выпадающего списка нужный цвет, чтобы закрасить им фон **LinearLayout**.



Кроме стандартной цветовой модели RGB используется также [Цветовая модель HSV](http://developer.alexanderklimov.ru/android/catshop/hsv.php)