Основы работы с 2D графикой на Android

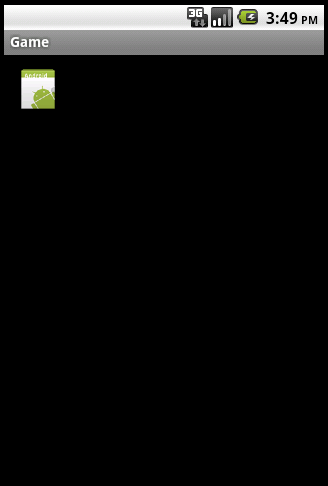
Совсем недавно я начал изучать программирование под android и конечно сразу поставил перед собой интересную цель — сделать небольшую игру.  
  
Ознакомившись за пару дней с Java (отличный курс: intuit.ru/department/pl/javapl/) и почитав developer.android.com, перешел к самому главному пункту — к графике. И теперь предлагаю вам разобрать работу с ней на простом примере.  
  
Итак, приступим.

Теория

Для вывода 2D графики android предоставляет два пути:  
  
а) Выводить графику в объекте View, который находится в вашем layout.  
В этом случае вся графика/анимация управляется самим андроидом и вы, грубо говоря, только определяете какую картинку показать.  
Этот способ подходит, если вы хотите вывести простую статичную графику, которая не должна динамично изменяться.  
  
б) Рисовать графику напрямую на канве (Canvas). В этом случае вы вручную вызываете методы канвы для рисования картинок, геометрических объектов и текста.  
Вы должны использовать этот способ, если графика в вашей программе должна часто обновляться/перерисовываться. Это как раз то, что нам нужно для игры.  
  
Вариант «б» можно реализовать двумя способами:  
  
1) В том же Activity, в котором находится наш класс View для вывода графики, мы зызываем метод invalidate(), который обновляет содержимое канвы.  
2) Создаем отдельный поток, который обновляет содержимое канвы настолько быстро, насколько может (в этом случае не нужно вызывать invalidate()).  
  
В первом случае изображение будет обновляться только по нашему требованию, что подходит для простых игр типа шахмат, которые не требует большого fps. Его мы и будем использовать в примере.

Код

Сначала импортируем нужные пакеты.  
  
import android.app.Activity;  
import android.os.Bundle;  
import android.view.View;  
import android.view.MotionEvent;  
import android.content.Context;  
import android.graphics.Canvas;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
  
В нашем Activity создадим класс GraphicsView, расширяющий View и определим внутри него метод onDraw(), в котором будут находиться команды рисования.  
  
public class Game extends Activity   
{  
@Override  
public void onCreate(Bundle savedInstanceState)  
{  
super.onCreate(savedInstanceState);  
GraphicsView myview=new GraphicsView(this); // создаем объект myview класса GraphicsView  
setContentView(myview); // отображаем его в Activity  
}  
  
public class GraphicsView extends View  
{  
public GraphicsView(Context context) { super(context); }  
  
@Override  
protected void onDraw(Canvas canvas)  
{  
// здесь будут находиться код, рисующий нашу графику  
}  
  
}  
  
}  
  
Для примера будем выводить картинку, а именно, стандартную иконку приложения.  
  
protected void onDraw(Canvas canvas)  
{  
// загружаем иконку из ресурсов в объект myBitmap  
Bitmap myBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.icon);  
// рисуем myBitmap на канве в координатах 10, 10  
canvas.drawBitmap(myBitmap, 10, 10, null);  
}  
  
Можете запустить программу и вы увидите следующее:



Каждый раз, когда нам необходимо обновить содержимое GraphicsView (например при изменении координат выводимой картинки), мы должны вызвать invalidate(), который говорит андроиду, что мы хотели бы перерисовать содержимое GraphicsView, и он вызовет наш метод onDraw(). Давайте будем вызывать его при касании экрана.  
  
Определим у нашего класса GraphicsView метод onTouchEvent().  
  
public boolean onTouchEvent(MotionEvent event)  
{  
if(event.getAction() == MotionEvent.ACTION\_DOWN) { invalidate() }  
return true;  
}  
  
Теперь, запустив программу и коснувшись экрана, мы будем вызывать invalidate(). Но, так как код внутри onDraw() у нас не меняется, то и никаких изменений на экране мы не увидим.

Давайте менять координаты картинки на те, в которых произошло прикосновение.  
  
Дополним код командами event.getX() и event.getX(). При прикосновении к экрану они получат координаты касания. Сохраним их в переменных touchX и touchY.  
  
public boolean onTouchEvent(MotionEvent event)  
{  
if(event.getAction() == MotionEvent.ACTION\_DOWN)  
{  
touchX = event.getX();  
touchY = event.getY();  
invalidate();  
}  
return true;  
}  
  
Вернемся к onDraw. Зададим переменные типа float с начальными нулевыми значениями и затем впишем их вместо фиксированных координат рисунка.  
  
float touchX = 0;  
float touchY = 0;  
  
@Override  
protected void onDraw(Canvas canvas)  
{  
// загружаем иконку из ресурсов в объект myBitmap  
Bitmap myBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.icon);  
// рисуем myBitmap на канве в координатах касания  
canvas.drawBitmap(myBitmap, touchX, touchY, null);  
}  
  
Можете запустить программу и коснуться экрана — картинка переместится в точку касания.  
  
Надеюсь, что этот пример послужит вам толчком для дальнейшего изучения программирования для Android и возможно создания вашей первой игры.