Game Activity

```
import android.app.Activity;
import android.os.Bundle;

public class GameActivity extends Activity {
	GameView GV;

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		GV = new GameView(this);
		setContentView(GV);
	}
}
```

GameView

```
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.view.MotionEvent;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

public class GameView extends SurfaceView implements Runnable {
	private SurfaceHolder holder;   // variables needed to implement runnable
	Thread thread = null;
	volatile boolean running = false;
	static final long FPS = 10;

	private Bitmap ball1;    // variable needed for game
	private int x = 0, y = 0;
	private int xSpeed = 5, ySpeed = 5;

	public GameView(Context context) {
		super(context);
		thread = new Thread(this);
		holder = getHolder();
		holder.addCallback(new SurfaceHolder.Callback() {

		@Override
		public void surfaceDestroyed(SurfaceHolder holder) {
			boolean retry = true;
			running = false;
			while (retry) {
				try {
					thread.join();
					retry = false;
				}
```

```
                                catch (InterruptedException e) {}
                        }
                }

                @Override
                public void surfaceCreated(SurfaceHolder holder) {
                        running = true;
                        thread.start();
                }

                @Override
                public void surfaceChanged(SurfaceHolder holder,int format,int width,int height) {}
        });
        ball1 = BitmapFactory.decodeResource(getResources(), R.drawable.ball);
}
@Override
public void run() {
        long ticksPS = 1000 / FPS;
        long startTime;
        long sleepTime;

        while (running) {
                Canvas c = null;
                startTime = System.currentTimeMillis();

                try {
                        c = getHolder().lockCanvas();
                        synchronized (getHolder()) {
                        update();
                        onDraw(c);
                }
        }
        finally {
                if (c != null) {
                        getHolder().unlockCanvasAndPost(c);
                }
        }
        sleepTime = ticksPS-(System.currentTimeMillis() - startTime);
        try {
                if (sleepTime > 0)
                         thread.sleep(sleepTime);
                else
                        thread.sleep(10);
                }
                catch (Exception e) {}
        }
}
private void update(){
        x++;
}
```

```
@Override
protected void onDraw(Canvas canvas) {
        canvas.drawColor(Color.WHITE);
        x = x + xSpeed;

        if (x <0) {
                xSpeed = -xSpeed;
                x = 0;
        }
        else if (x + ball1.getWidth()  > getWidth()) {
                xSpeed = -xSpeed;
                x = getWidth()-ball1.getWidth();
        }
        y = y + ySpeed;

        if (y <0) {
                ySpeed = -ySpeed;
                y = 0;
        }
        else if (y + ball1.getHeight()  > getHeight()) {
                ySpeed = -ySpeed;
                y = getHeight()-ball1.getHeight();
        }
        canvas.drawBitmap(ball1, x , y , null);
}

        @Override
        public boolean onTouchEvent(MotionEvent event) {
                x = (int) event.getX();
                y = (int) event.getY();
                return super.onTouchEvent(event);
        }
}
```