Android-Umsetzung der Spielebibliothek

Sven Eric Panitz
Hochschule Rhein-Main

Dieses Papier beschreibt die Android-Umsetzung des Spielebibliothek. Dieses ist die jüngste Umsetzung und auf Grund der unterschiedlichen Endgeräte die problematischste. Einige Ungereimtheiten finden sich noch in der Umsetzung, aber es sollte möglich sein, die Spiele auch für Android bereit zu stellen.

Ein Android Spieleprojekt Erzeugen Es gibt derzeit kein automatisches Skript, um ein Projekt für Android zu erzeugen. Aber es sollte reletiv einfach möglich sein, ein Projekt mit Hilfe der Entwicklungsumgebung Android-Studio mit leerer Activity zu erzeugen.

Hierzu ist in Android-Studio ein neues Projekt zu erzeugen. Als erstes ist die Entwicklungsumgebung für das Projekt so einzustellen, dass Java 8 Quelltext verarbeitet werden kann. Hierzu ist der Menupunkt Project-Structure auszuwählen (auch mit ›Strg-Alt-Umsch S‹ zu erreichen). Hier sind für Quell- und Zielkompatibilität die Java Version 1.8 zu wählen.

Anschließend die Quelltexte der Rahmenbibliothek und die Quelltexte der Android-Umsetzung in das Projekt in den Java-Quelltextorder kopieren.

Die Bilddateien für das Spiel landen in den Ordner `res/drawable` und die Klangdateien schließlich in den Ordner `res/raw`. Dieser ist im Projekt erst noch anzulegen.

Dann ist natürlich auch der Quelltext des eigenen Spiels in das Projekt zu kopieren. Als letzter Schritt ist die bei Projekterzeugung generierte Klasse `MainActivity` durch eine Implementierung entsprechend dem Beispiel am Ende dieses papiers abzuändern. Sie muss dann die Klasse `GameActivity` erweitern und im `super`-Konstruktoraufruf eine Instanz des eigenen Spiels übergeben.

Nun sollte sich die Android Applikation kompilieren und starten lassen.

1 Die Android-Umsetzung

Dieses kapitel beschäftigt sich mit der Umsetzung des Rahmenbibliothek auf Android. Dieses ist die jüngste Umsetzung, da es erst relativ spät möglich war, in der Entwicklung von Android-Applikationen auch Java 8 Quelltext zu verwenden.

1.1 Das Zeichenwerkzeug

Jede Umsetzung der Bibliothek brauch zunächst eine Implementierung der Schnittstelle `GraphicsTool` für die entsprechende Plattform. Hierbei wir der in der Rahmenbibliothek als generischer Typ offen gelassene Typ der Bilddateien konkretisiert. Im Android Fall entsteht eine Implementierung für Bilder des Android Typs `Bitmap`.

```
1 package name.panitz.game.framework.android;
2
3 import android.content.Context;
4 import android.content.res.Resources;
5 import android.graphics.Bitmap;
6 import android.graphics.BitmapFactory;
7 import android.graphics.Canvas;
```

```
8  import android.graphics.Paint;
9  import android.os.Build;
10 import android.support.annotation.RequiresApi;
11 import android.util.DisplayMetrics;
12
13 import name.panitz.game.framework.GameObject;
14 import name.panitz.game.framework.GraphicsTool;
15
16
17 public class AndroidGraphicsTool implements GraphicsTool<Bitmap> {
18   Canvas canvas;
19   Context view;
20   Paint paint;
21   float density;
22
23   private int byIdName(String name) {
24     Resources res = view.getResources();
25     return res.getIdentifier(name, "drawable", view.getPackageName());
26   }
27
28   public AndroidGraphicsTool(Paint paint,Context view, Canvas canvas){
29     this.canvas = canvas;
30     this.view = view;
31     this.paint = paint;
32     DisplayMetrics dm = view.getResources().getDisplayMetrics();
33     density = dm.density;
34   }
35
36
37   @Override
38   public void drawImage(Bitmap img, double x, double y) {
39     canvas.drawBitmap(img,(int)(density*x),(int)(density*y),null);
40   }
41
42   @Override
43   public void drawRect(double x, double y, double w, double h) {
44     paint.setStyle(Paint.Style.STROKE);
45     canvas.drawRect((float)(density*x),(float)(density*y)
46       ,(float)(w*density+(density*x))
47       ,(float)(h*density+(density*y)),paint);
48   }
49
50   @Override
51   public void drawLine(double x1, double y1, double x2, double y2) {
52     paint.setStyle(Paint.Style.STROKE);
53     canvas.drawLine((float)(density*x1),(float)(density*y1)
54       ,(float)(x2*density),(float)(y2*density),paint);
55   }
56
57
58   @Override
59   public void fillRect(double x, double y, double w, double h) {
```

```
60      paint.setStyle(Paint.Style.FILL);
61      canvas.drawRect((float)(density*x),(float)(density*y)
62          ,(float)(w*density+(density*x))
63          ,(float)(h*density+(density*y)),paint);
64
65    }
66
67    @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
68    @Override
69    public void drawOval(double x, double y, double w, double h) {
70      paint.setStyle(Paint.Style.STROKE);
71      canvas.drawOval((float)(density*x),(float)(density*y)
72          ,(float)(w*density+(density*x))
73          ,(float)(h*density+(density*y)),paint);
74    }
75
76    @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
77    @Override
78    public void fillOval(double x, double y, double w, double h) {
79      paint.setStyle(Paint.Style.FILL);
80      canvas.drawOval((float)(density*x),(float)(density*y)
81          ,(float)(w*density+(density*x))
82          ,(float)(h*density+(density*y)),paint);
83    }
84
85    @Override
86    public void setColor(double r, double g, double b) {
87      paint.setARGB(255,(int)(255*r),(int)(255*g),(int)(255*b));
88    }
89
90    @Override
91    public void drawString(double x, double y, int fntsize
92        , String fntName, String text) {
93      paint.setTextSize(fntsize);
94      canvas.drawText(text,(int)(density*x),(int)(density*y),paint);
95    }
96
97    @Override
98    public Bitmap generateImage(String name, GameObject<Bitmap> go) {
99      Bitmap bmp = BitmapFactory.decodeResource(view.getResources()
100         ,byIdName(name.substring(0,name.lastIndexOf('.'))));
101     go.setWidth(bmp.getWidth()/density);
102     go.setHeight(bmp.getHeight()/density);
103     return bmp;
104   }
105 }
```

Listing 1: app/src/main/java/name/panitz/game/framework/android/AndroidGraphicsTool.java

In dieser Klasse wird nicht nur das Zeichnen der Objekte umgesetzt, sondern auch das Laden der Resourcen. Es wird dabei davon ausgegangen, dass die Bilddateien Projekt-konkorm im Ordner `res/drawable` liegen. Leider muss man bei der Andoid-

Umsetzung auf die Animation von animierten gif-Dateien verzichten, da Android diese nicht unterstützt.

1.2 Klänge

Ebenso, wie das Zeichnen von Grafiken, ist das Abspielen (und Laden) von Klängen in jeder Umsetzung spezifisch durch Implementieren der entsprechenden Schnittstelle unzusetzen. Auch hier wird der in der Rahmenbibliothek durch eine Typvariable in generischen Typen offen gehaltene Typ der Klangdateien für die Umsetzung konkretisiert. Im Android Fall entsteht ein `SoundTool<MediaPlayer>`.

```
1  package name.panitz.game.framework.android;
2
3  import android.content.Context;
4  import android.content.res.Resources;
5  import android.media.MediaPlayer;
6
7  import name.panitz.game.R;
8  import name.panitz.game.framework.SoundTool;
9
10 import static android.media.MediaPlayer.*;
11
12 public class AndroidSoundTool implements SoundTool<MediaPlayer> {
13   Context view;
14
15   public AndroidSoundTool(Context view) {
16     this.view = view;
17   }
18
19   private int byIdName(String name) {
20     Resources res = view.getResources();
21     return res.getIdentifier(name, "raw", view.getPackageName());
22   }
23
24   @Override
25   public MediaPlayer loadSound(String name) {
26     return MediaPlayer.create(view
27         ,byIdName(name.substring(0,name.lastIndexOf('.'))));
28   }
29
30   @Override
31   public void playSound(MediaPlayer sound) {
32     if (sound.isPlaying()) {
33       sound.pause();
34       sound.seekTo(0);
35     }
36     sound.start();
37   }
38 }
```

Listing 2: app/src/main/java/name/panitz/game/framework/android/AndroidSoundTool.java

1.3 Spielansicht und Schleife

Gezeichnet wird auf einem `Canvas`-Objekt, das zu einem `SurfcaeView`-Objekt gehört. Die Hauptspielschleife wird als `Runnable` realisiert.

```
1  package name.panitz.game.framework.android;
2
3  import android.content.Context;
4  import android.graphics.Bitmap;
5  import android.graphics.Canvas;
6  import android.graphics.Color;
7  import android.graphics.Paint;
8  import android.media.MediaPlayer;
9  import android.util.Log;
10 import android.view.MotionEvent;
11 import android.view.SurfaceHolder;
12 import android.view.SurfaceView;
13
14 import name.panitz.game.framework.GameLogic;
15 import name.panitz.game.framework.SoundObject;
16
17 public class Screen extends SurfaceView implements Runnable {
18   GameLogic<Bitmap, MediaPlayer> game;
19   Thread gameThread = null;
20   SurfaceHolder ourHolder;
21
22   volatile boolean playing;
23   Canvas canvas;
24   Paint paint;
25
26   long fps;
27   private long timeThisFrame;
28
29   private AndroidGraphicsTool graphicsTool;
30   private AndroidSoundTool soundTool;
31
32   public Screen(Context context, GameLogic<Bitmap, MediaPlayer> game){
33     super(context);
34     this.game = game;
35     ourHolder = getHolder();
36     paint = new Paint();
37     this.setMinimumHeight((int) game.getHeight());
38     this.setMinimumWidth((int) game.getWidth());
39     soundTool = new AndroidSoundTool(context);
40   }
41
42
```

```
43    @Override
44    public void run() {
45      while (playing) {
46        // Capture the current time in milliseconds in startFrameTime
47        long startFrameTime = System.currentTimeMillis();
48
49        update();
50        draw();
51        game.playSounds(soundTool);
52
53        timeThisFrame = System.currentTimeMillis() - startFrameTime;
54        if (timeThisFrame >= 1) {
55          fps = 1000 / timeThisFrame;
56        }
57      }
58    }
59
60    @Override
61    public boolean onTouchEvent(MotionEvent me) {
62      return true;
63    }
64
65    public void update() {
66      if (!game.isStopped()) {
67        game.move();
68        game.doChecks();
69      }
70    }
71
72    public void draw() {
73      // Make sure our drawing surface is valid or we crash
74      if (ourHolder.getSurface().isValid()) {
75        // Lock the canvas ready to draw
76        canvas = ourHolder.lockCanvas();
77        canvas.drawColor(Color.argb(255, 26, 128, 182));
78        graphicsTool
79            = new AndroidGraphicsTool(paint, getContext(), canvas);
80        game.paintTo(graphicsTool);
81        graphicsTool.paint.setColor(Color.argb(255, 26, 128, 182));
82        graphicsTool.drawRect(game.getWidth(),0,800,800);
83        // Draw everything to the screen
84        ourHolder.unlockCanvasAndPost(canvas);
85      }
86
87    }
88
89    // If SimpleGameEngine Activity is paused/stopped
90    // shutdown our thread.
91    public void pause() {
92      playing = false;
93      try {
94        gameThread.join();
```

```
95      } catch (InterruptedException e) {
96        Log.e("Error:", "joining thread");
97      }
98    }
99
100   // If SimpleGameEngine Activity is started then
101   // start our thread.
102   public void resume() {
103     playing = true;
104     gameThread = new Thread(this);
105     gameThread.start();
106   }
107 }
```

Listing 3: app/src/main/java/name/panitz/game/framework/android/Screen.java

1.4 Die Hauptaktivität

Um die Spieleansicht in eine Spielaktivität zu kapseln, bedarf es noch einer Steuerung und auch die im Spiel vorhanden Knöpfen sind der Applikation hinzuzufügen.

```
1  package name.panitz.game.framework.android;
2
3  import android.graphics.Bitmap;
4  import android.media.MediaPlayer;
5  import android.os.Bundle;
6
7  import android.support.v7.app.AppCompatActivity;
8  import android.view.GestureDetector;
9  import android.view.MotionEvent;
10 import android.view.View;
11 import android.widget.LinearLayout;
12 import android.widget.RelativeLayout;
13
14 import name.panitz.game.framework.Button;
15 import name.panitz.game.framework.GameLogic;
16
17
18 public class GameActivity extends AppCompatActivity {
19   Screen gameView;
20   private GestureDetector mGestureDetector;
21
22   GameLogic<Bitmap,MediaPlayer> game;
23
24   public GameActivity(GameLogic<Bitmap,MediaPlayer> game) {
25     this.game = game;
26   }
27
28   @Override
29   protected void onCreate(Bundle savedInstanceState) {
30     super.onCreate(savedInstanceState);
```

```
31      gameView = new Screen(this, game);
32      gameView.setMinimumWidth((int) game.getWidth());
33      gameView.setMinimumHeight((int) game.getHeight());
34
35      LinearLayout rl = new LinearLayout(this);
36      // Create an object of our Custom Gesture Detector Class
37      CustomGestureDetector customGestureDetector
38        = new CustomGestureDetector(game);
39      // Create a GestureDetector
40      mGestureDetector=new GestureDetector(this,customGestureDetector);
41      mGestureDetector.setOnDoubleTapListener(customGestureDetector);
42
43      gameView.setOnTouchListener(new View.OnTouchListener() {
44        @Override
45        public boolean onTouch(View v, final MotionEvent event) {
46          mGestureDetector.onTouchEvent(event);
47          return true;
48        }
49      });
50
51
52      LinearLayout butView = new LinearLayout(this);
53      butView.setOrientation(LinearLayout.VERTICAL);
54
55      for (Button b:game.getButtons()){
56        android.widget.Button but = new android.widget.Button(this);
57        but.setText(b.name);
58        but.setOnClickListener(ev->b.action.run());
59        butView.addView(but);
60      }
61      rl.addView(butView);
62      rl.addView(gameView);
63
64      setContentView(rl);
65    }
66
67    @Override
68    protected void onResume() {
69      super.onResume();
70      // Tell the gameView resume method to execute
71      gameView.resume();
72    }
73
74    // This method executes when the player quits the game
75    @Override
76    protected void onPause() {
77      super.onPause();
78      // Tell the gameView pause method to execute
79      gameView.pause();
80    }
81  }
```

Listing 4: app/src/main/java/name/panitz/game/framework/android/GameActivity.java

1.5 Steuerung mit Fingergesten

Wir haben keine Tastatur und wollen auch nicht, dass das Spiel mit einer aufklappenden Tastatur auf dem Bildschirm gesteuert wird. Deshalb setzt die Android-Umsetzung einige wichtige Tasten durch Wischgesten auf dem Bildschirm um. Dieses sind vorerst die vier Pfeiltasten und die Leertaste.

```
1  package name.panitz.game.framework.android;
2
3  import android.graphics.Bitmap;
4  import android.media.MediaActionSound;
5  import android.media.MediaPlayer;
6  import android.view.GestureDetector;
7  import android.view.MotionEvent;
8
9  import name.panitz.game.R;
10 import name.panitz.game.framework.GameLogic;
11 import name.panitz.game.framework.KeyCode;
12
13 public class CustomGestureDetector
14   implements GestureDetector.OnGestureListener,
15         GestureDetector.OnDoubleTapListener {
16
17   GameLogic<Bitmap, MediaPlayer> game;
18
19   public CustomGestureDetector(GameLogic<Bitmap, MediaPlayer> game) {
20     this.game = game;
21   }
22
23   @Override
24   public boolean onSingleTapConfirmed(MotionEvent e) {
25     return true;
26   }
27
28   @Override
29   public boolean onDoubleTap(MotionEvent e) {
30     game.keyPressedReaction(KeyCode.VK_SPACE);
31     return true;
32   }
33
34   @Override
35   public boolean onDoubleTapEvent(MotionEvent e) {
36     return true;
37   }
38
39   @Override
```

```
40    public boolean onDown(MotionEvent e) {
41      return true;
42    }
43
44    @Override
45    public void onShowPress(MotionEvent e) {
46    }
47
48    @Override
49    public boolean onSingleTapUp(MotionEvent e) {
50      return true;
51    }
52
53    @Override
54    public boolean onScroll(MotionEvent e1, MotionEvent e2
55        , float distanceX, float distanceY) {
56      return true;
57    }
58
59    @Override
60    public void onLongPress(MotionEvent e) {
61    }
62
63    @Override
64    public boolean onFling(MotionEvent e1, MotionEvent e2
65        , float velocityX, float velocityY) {
66      if ((e1.getX() - e2.getX())<40) {
67        game.keyPressedReaction(KeyCode.RIGHT_ARROW);
68      }
69
70      if (e1.getX() - e2.getX() > 40) {
71        game.keyPressedReaction(KeyCode.LEFT_ARROW);
72      }
73
74      if (e2.getY() - e1.getY() >40) {
75        game.keyPressedReaction(KeyCode.DOWN_ARROW);
76      }
77
78      if (e1.getY() - e2.getY() > 40) {
79        game.keyPressedReaction(KeyCode.UP_ARROW);
80      }
81
82      return true;
83    }
84  }
```

Listing 5: app/src/main/java/name/panitz/game/framework/android/CustomGestureDetector.

1.6 MainActivity zum Starten eines Spiels

Hier ein kleines Beispiel, wie man sein Spiel in der Android-Umsetzung einbindet und gestartet bekommt-

```
1  package name.panitz.game;
2
3  import name.panitz.game.framework.android.GameActivity;
4  import name.panitz.game.example.simple.SimpleGame;
5
6  public class MainActivity extends GameActivity{
7    public MainActivity() {
8      super(new SimpleGame<>());
9    }
10 }
```

Listing 6: app/src/main/java/name/panitz/MainActivity.java