FlipPhone

Projekt Dokumentation

Duc Tan Dang
2076602
Sven Schröder
2058820

Hochschule für Angewandte Wissenschaften Hamburg

Fakultät Design, Medien und Information

Department Medientechnik

Prüfer: Herr A. Plaß

Hamburg, 23. Juni 2014

Einleitung

Im Rahmen des Studiengangs Media Systems im Kurs Mobile Systeme hatten wir vor eine App zu entwickeln die Messwerte mit Hilfe der Smartphone Sensoren wieder gibt. Dabei hatten wir die Idee eine Art Spaßapp zu entwickeln. Das Smartphone soll dabei in die Luft geworfen werden, und zwar so, dass es sich um die eigenen Achsen dreht. Die Umdrehungen um alle drei Achsen werden gezählt beziehungsweise "gemessen". In Wirklichkeit reicht es, wenn man es in der Hand dreht, weil bei solchen Würfen doch ein Risiko besteht, dass das Smartphone zu Bruch geht. Als Sensor soll das Gyroskop oder auch der Orientierungssensor dienen.

Marktforschung

Bei der Recherche über ähnliche Apps haben wir folgende Apps entdeckt:

- Throw Your Phone - https://play.google.com/store/apps/details?id=com.bytemods.throw_phone
- S.M.T.H - https://play.google.com/store/apps/details?id=com.carrotpop.www.smth

Bei der App Throw Your Phone bekommt man Punkte, umso höher man wirft, desto mehr Punkte bekommt man. Jedoch soll die App laut Rezensionen oft abstürzen oder erst gar nicht starten.
Sent Me To Heaven (S.M.T.H.) hat gute Rezensionen und soll die Höhe in Metern wiedergeben, die das Smartphone in der Luft zurück legt. Außerdem sieht sie optisch professioneller aus als Throw Your Phone. Diese App wurde jedoch aus dem Apple Store entfernt, da die Benutzung der App zu Hardwareschäden führen kann.
Beide Apps messen jedoch nur die Höhe und nicht die Umdrehungen, wie bei unserer App. Somit gibt es zur Zeit keine uns bekannte Android App die wie FlipPhone ist.

Umsetzung

Bei der Umsetzung hatten wir die Idee, dass wir den Orientierungssensor benutzen. Der Sensor gibt die Gradzahlen in Abhängigkeit von der Neigung des Smartphones wieder. Wenn ein Wert doppelt durchlaufen wurde, zählt eine Zählervariable um Eins hoch.

Klassen und Objekte

Die App besteht aus drei verschiedenen Activities mit deren jeweiligen Views als xml-Datei. Die erste Activity ist die Klasse MainActivity.java. Diese Klasse ist der Startbildschirm wenn man die App startet. Sie beinhaltet nur den App Namen, das Logo und einen Button. Dieser Button führt zur zweiten Actitvity, die Klasse ReadActivity.java. Das ist die Klasse mit den eigentlichen Funktionen. Sie benutzt den Sensor und zählt die Umdrehungen. Der Button dieser Acitivity übergibt außerdem die Anzahl der Umdrehungen auf die nächste Activity. Die dritte und letzte Activity heißt ResultActivity.java und erhält die übergegebenen Werte von der ReadActivity. Von der ResultActivity gelangt man durch den Button wieder zum Anfang (MainActivity) der App.

Controller

Sensor

```
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_ORIENTATION);
```

Die Variable “sensor” wird als TYPE_ORIENTATION Sensor definiert.
Dieser Sensor gibt die Winkel zu den Achsen X, Y und Z an, zu denen sich das Smartphone befindet.
Dieser Sensor scheint veraltet zu sein, weshalb dieser Teil des Codes durchgestrichen ist.
Er reicht für unsere Bedürfnisse dennoch aus, da wir nur diese drei Achsen benötigen.
Wir haben außerdem keine groben Fehler die unsere Berechnung stören fest gestellt.

```
if (sensor != null) {
        sensorManager.registerListener(mySensorEventListener, sensor,
SensorManager.SENSOR_DELAY_FASTEST);
} else {
        Toast.makeText(this, "ORIENTATION Sensor not found", Toast.LENGTH_LONG).show();
        finish();
}
```

Die Geschwindigkeit mit der sich die Werte des Sensors ändern können, wird auf "SENSOR_DELAY_FASTEST" gestellt. Dies ist die schnellste Einstellung und ermöglicht es auch bei schnellen Rotationen des Android Devices die Umdrehungen zu erfassen. Allerdings ist selbst diese Einstellung nicht uneingeschränkt schnell.

Sensor Event Listener

```
public void onSensorChanged(SensorEvent event) {
```

Die nachfolgenden Zeilen werden durch das Ändern des zuvor festgelgten Sensors ausgeführt.

```
rotationsX = (TextView) findViewById(R.id.textView5);
rotationsY = (TextView) findViewById(R.id.textView6);
rotationsZ = (TextView) findViewById(R.id.textView7);
```

Hier werden die textViews mit den IDs aus der entsprechenden xml Datei definiert.

```
if ((axisX > 30 && axisX < 120) && rotationConditionX == false){
        rotationConditionX = true;
}
if ((axisX > 270 && axisX < 360) && rotationConditionX == true){
        rotationConditionX = false;
        numberOfRotationsX += 1;
}
```

Der X-Achsen Wert des Orientation Sensors zählt von 0 bis 359.

Wenn der Wert einen bestimmten Wertebereich erreicht, dann ist die Bedingung gegeben das wenn der Wert danach einen zweiten Bereich erreicht die Anzahl der Umdrehungen um eins erhöht wird.

Die Wertebereiche sollen eine Umdrehung darstellen, daher startet der erste Bereich bei 0° und der zweite endet bei 360°. Um Fehler beim Start zu verringern beginnt der erste Bereich allerdings bei 30°.

```
// Y Achsen Zähler
if ((axisY > 30 && axisY < 120) && rotationConditionY == false){
        rotationConditionY = true;
}
if ((axisY > -150 && axisY < -60) && rotationConditionY == true){
        numberOfRotationsY += 1;
        rotationConditionY = false;
}
// Z Achsen Zähler
if ((axisZ > 30 && axisZ < 120) && rotationConditionZ == false){
        rotationConditionZ = true;
}
if ((axisZ > -150 && axisZ < -60) && rotationConditionZ == true){
        numberOfRotationsZ += 1;
        rotationConditionZ = false;
}
```

Bei den Bedingungen von der Y- und Z-Achse passiert das selbe, wie bereits bei der X-Achse. Allerdings zählt der Orientation Sensor hier nur von 0 bis 180 sowie von -180 bis 0. Daher mussten wir die Wertebereiche entsprechend anpassen.

Activities

Um die Anzahl der Drehungen von der zweiten Activity (ReadActivity.java) in die dritte Activity (ResultActivity.java) zu übergeben, benutzen wir einen Intent. Dabei speichern wir zu allererst die Anzahl der Umdrehung in einem String.

```
String rollValue = rotationsX.getText().toString();
String pitchValue = rotationsY.getText().toString();
String yawValue = rotationsZ.getText().toString();
```

Danach erstellen wir einen Intent:

```
Intent intent = new Intent(this, ResultActivity.class);
```

Und diesem Intent geben wir den String:

```
intent.putExtra(EXTRA_PITCH, pitchValue);
intent.putExtra(EXTRA_ROLL, rollValue);
intent.putExtra(EXTRA_YAW, yawValue);
```

Um die Anzahl der Drehungen in der dritten Activity zu bekommen, holen wir uns den zuvor erstellten Intent und speichern die Werte wieder in einem anderen String ab:

```
Intent intent = getIntent();
String roll = intent.getStringExtra(ReadActivity.EXTRA_ROLL);
String pitch = intent.getStringExtra(ReadActivity.EXTRA_PITCH);
```

```
String yaw = intent.getStringExtra(ReadActivity.EXTRA_YAW);
```
Jetzt muss man einfach nur die Strings in die Textfelder einfügen.

View

Layout/Design

Die Benutzeroberfläche wurde sehr einfach gehalten, da wir uns mehr auf die Funktionstüchtigkeit konzentriert haben.

Alle drei Activities sind in Graustufen dargestellt und besitzen einen schwarzen, runden Button.

Der Button ist in einer xml-Datei abgespeichert und wurde als <shape> definiert.

Code:

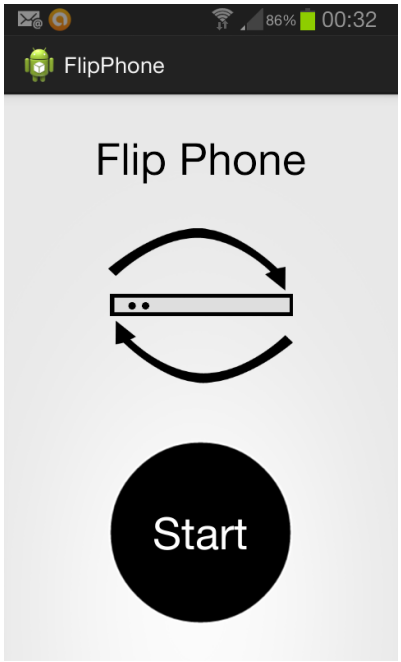
```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
   android:shape="oval">
   <solid android:color="#000000"/>
   <stroke android:width="2sp" android:color="#fff" />
</shape>
```

In der ersten Activity sind außerdem noch ein TextView mit dem App Namen und ein ImageView mit dem Logo.

Die zweite Activitiy besitzt sieben verschiedene TextViews, jeweils drei für die Namen der Achsen und drei für die Darstellungen der Zahlen der Umdrehungen. Die übrige TextView dient als Überschrift.

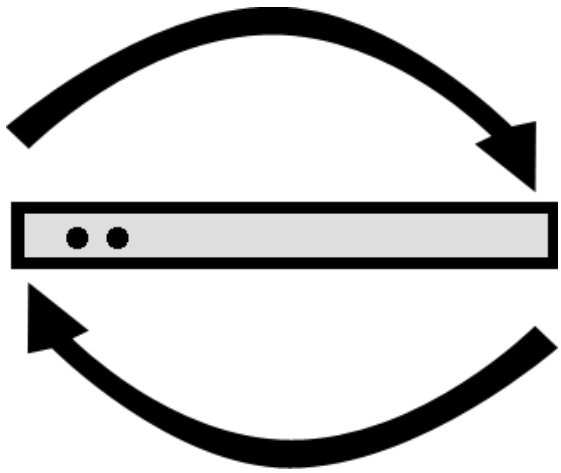
Die dritte Activity ist der zweiten Activity ähnlich, nur der Überschriftentext ist anders.

Alle Activities haben ein RelativeLayout zur einfachen Verschiebung der View Elemente.



Logo

Das Logo stellt eine von der Seite, liegendes Smartphone dar. Es ähnelt dem iPhone. Drum herum befinden sich zwei abgerundete Pfeile die eine Drehbewegung darstellen. Dadurch spiegelt das Logo auch den Namen “FlipPhone” wieder, da sich das Smartphone in der Darstellung um sich selbst dreht. Es wurde komplet eigenständig in Photoshop erstellt.



Probleme

Die größten Probleme hatten wir bei der Entscheidung welchen Sensor wir benutzen wollen. Wir hatten zuerst probiert den Gyroskop Sensor zu benutzen. Allerdings war es uns nicht gelungen, die Werte des Gyroskops in Umdrehungen umzurechnen.
Daher entschieden wir uns den “Orientation” Sensor zu benutzen. Dieser ist zwar nicht mehr aktuell war aber der einzige Sensor der zuverlässig Werte ausgegeben hat die wir direkt benutzen konnten.
Allerdings scheint es so das der “Orientation” Sensor manchmal sehr schwerfällig reagiert, so dass bei sehr schnellen Umdrehungen des Android Devices keine Umdrehung gezählt wird. Außerdem gibt es hin und wieder Probleme mit der Ausrichtung des Devices, da die Achsen sich nicht nach dem Device richten.

Quellen

- http://www.basicthinking.de/blog/2013/08/06/send-me-to-heaven-apple-verbietet-wirf-dein-iphone-so-hoch-du-kannst-app/
- https://developer.android.com/index.html
- Skript von der Vorlesung Mobile Systeme von Herrn Plaß