dBCheck

https://github.com/HogIng/MobileSysteme.git/

Mobile Systeme SoSe 2014

Leitung: Prof. Dr. Andreas Plaß

von Inga Hogrefe und Madita Kirschbaum

27.06.2014

Gliederung

1. Konzept

1.1. Grundidee

Soundadaption soll die Umgebungslautstärke messen und anhand der Messdaten eine Anpassung der Lautstärke des Handys automatisch vornehmen.
So sollen dem Handy-User auch in geräuschvoller Umgebung Benachrichtigungen und Anrufe nicht entgehen, indem die Klingeltonlautstärke erhöht wird. In ruhiger Umgebung wird diese reduziert, sodass Anwesende nicht durch zu laute Handytöne gestört werden.
Bei Aktivierung der automatischen Anpassung läuft die Messung permanent im Hintergrund. Öffnet man "Soundadaption" wird die aktuelle Umgebungslautstärke angezeigt.

1.2. Problem

Das Mikrofon eines Smartphones ist im Allgemeinen auf die menschliche Stimme ausgerichtet, sodass der maximale Messwert begrenzt ist und nach eigenen Untersuchungen nur zuverlässige Werte zwischen 35 dB und 80 dB ausgegeben werden. Abweichungen können durch unterschiedliche Mikrofonsensibilität und Kapazität auftreten.

Ein Maximalwert ist beispielsweise beim Samsung GALAXY S I9000 bereits bei 80 dB erreicht.
Dieser Wert entspricht etwa der Lautstärke, die durch einen PKW im Abstand von 4 m bei einer Geschwindigkeit von 50 km/h erzeugt wird.
Aufgrund des begrenzten Messbereiches ist die Anpassung der Signaltöne an größere Lautstärken als dem modellabhängigen Maximalwert nicht

umsetzbar, sodass entschieden wurde, sich ausschließlich auf die Verwirklichung der Lautstärkemessung und einer ansprechenden Darstellung für den Nutzer zu konzentrieren.

Das Konzept wurde angepasst und der Name der App von Soundadaption zu dBCheck geändert.

1.3. Überarbeitetes Konzept

dBCheck dient zur Messung der Umgebungslautstärke des Users.
Dazu verwendet die Anwendung das Mikrofon des Smartphones und rechnet die ermittelten Werte mit Hilfe der angepassten Schalldruckpegelformel in Dezibel um.
Angezeigt werden die aktuelle sowie die durchschnittliche Lautstärke in Form eines Zahlenwertes. Zusätzlich ist die Darstellung durch einen Graphen und einen Farbbalken möglich.
Der Mittelwert kann vom User zurückgesetzt werden.

Entwickelt wird die Applikation für die Sofware-Plattform Android.

2. Umsetzung

2.1. Messung der Lautstärke

Um die Lautstärkemessung zu realisieren wurde sich zunächst mit dem Thema Lautstärke und ihrer Berechnung auseinander gesetzt.

2.1.1. Allgemeines

Ein Schallereignis, welches durch eine Änderung des Luftdrucks entsteht, bewegt unsere Trommelfelle und die Mikrofonmembranen und wird als Laut einer bestimmten Stärke empfunden.

Die Intensität eines Schallereignisses kann durch ein logarithmisches Maß, dem Schalldruckpegel, beschrieben werden. Die allgemeine Einheit des Schalldruckpegels ist Dezibel.

Allgemeine Formel zur Berechnung des Schalldruckpegels in Dezibel:

$$dB = 20 \log(p/p_0)$$

p: Schalldruck in Pascal, p_0: Bezugswert in Pascal

Der Bezugswert beschreibt den kleinsten wahrnehmbaren Schalldruckpegel – die Hörschwelle. Dieser beträgt für Luftschall $2 \cdot 10^{-5}$ Pa.

3.2. Formelanpassung

Über das Mikrofon des Smartphones ist es nicht möglich, Pascalwerte zu ermitteln, sodass eine Anpassung der Formel für die Dezibelberechnung vorgenommen werden muss.
Die Klasse MediaRecorder bietet die Möglichkeit unter Verwendung des Mikrofons Audio aufzunehmen und zu analysieren. So kann über getMaxAmplitude() der maximale Ausschlag, ein Wert des primitiven Datentyps int, seit letztem Methodenaufruf abgefragt werden.
Zur Ermittlung eines Bezugswertes für die in dBCheck verwendete Formel wurden Messungen mit den Modellen GALAXY Nexus und GALAXY S I9000 durchgeführt. Die ermittelten Werte wurden in Bezug gesetzt und so der Hörschwellenwert bestimmt.

Verwendete Formel in dBCheck:

$$dB = 20 \log(\text{max. Amplitude}/3)$$

2.2. Programmstruktur

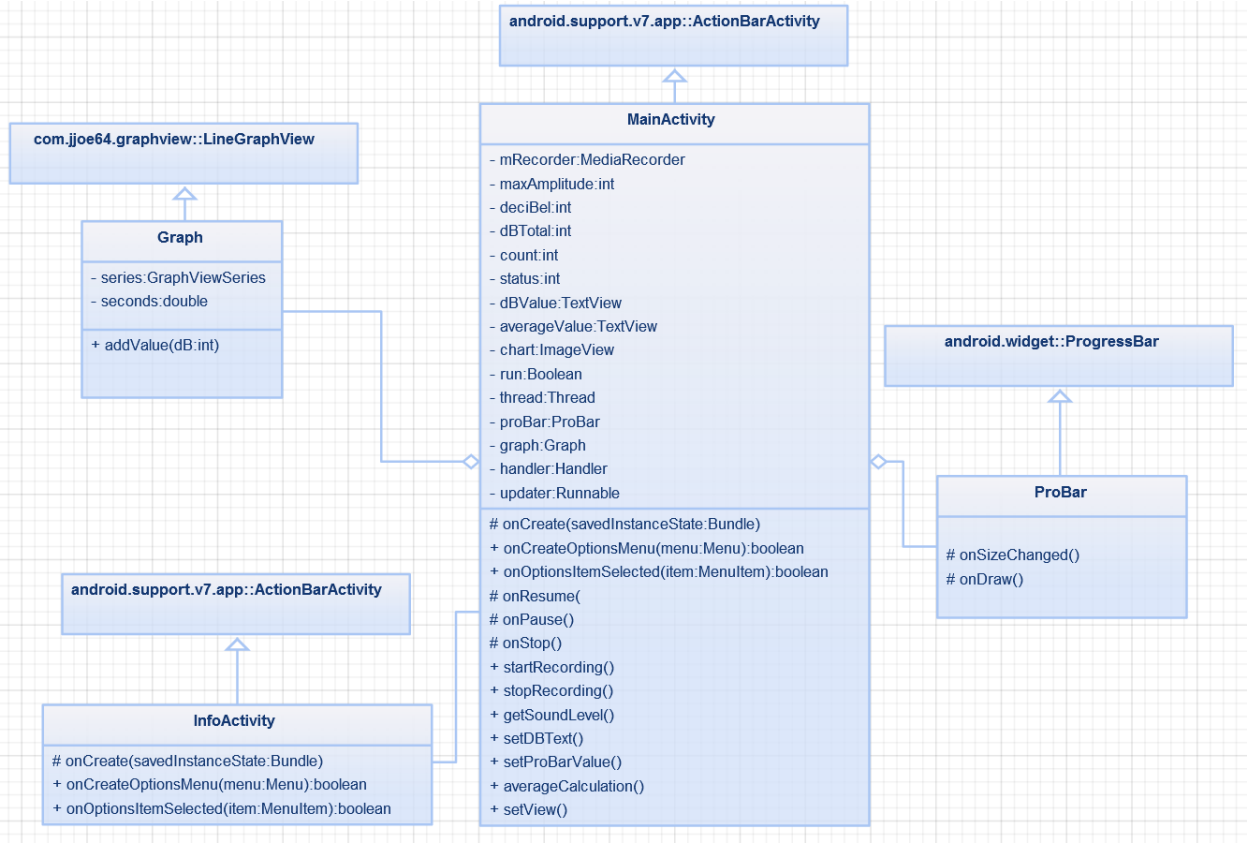

Abb. 1: UML-Klassendiagramm

2.2.1. MainActivity

Diese Klasse enthält Objekte der Klassen Graph und ProBar. Für den Zugriff auf das Mikrofon wird eine Instanz der Klasse MediaRecorder genutzt, um die maximale Amplitude des Audiostreams auszulesen und damit den Dezibel-Wert errechnen zu können. Ein Thread, dessen Methode run() sekündlich aufgerufen wird, dient zur Aktualisierung der gemessenen Dezibel-Werte.
Die Methode averageCalculation() berechnet einen Mittelwert aus den bisher gemessenen Werten. Dieser kann vom User zurückgesetzt werden.

Des Weiteren ist die Klasse für die Darstellung der vom Nutzer gewählten Ansicht zuständig.
Um das Verhalten der Anwendung, wenn sie geschlossen wird, eine andere Anwendung in den Vordergrund tritt oder der User wieder zu ihr zurückkehrt zu definieren, wurden die Methoden onStop(), onPause() und onResume() der Superklasse ActionBarActivity überschrieben.

2.2.2. InfoActivity

Wird der Info-Button der MainActivity geklickt, öffnet sich eine zweite Activity, die Informationen über die Bedienung der Anwendung gibt, Beispiele von allgemeinen Dezibel-Werten zeigt und über die Kapazität einzelner Handymodelle aufklärt.
Über den Pfeil-Button gelangt der User zurück zur MainActivity.

2.2.3. Graph

In der Klasse Graph wird eine Unterklasse von LineGraphView deklariert. Diese wurde über eine externe Library dem Projekt hinzugefügt.
Ein Objekt der Klasse GraphViewSeries wird zur Verwaltung der Dezibel-Werte genutzt.
Mit addValue() wird ein neuer Wert übergeben und ein Neuzeichnen des Graphen bewirkt.

2.2.4. ProBar

Dies ist eine erbende Klasse der ProgressBar. Durch eine Translation und Rotation der ProgressBar wird eine vertikale Darstellung möglich gemacht.

2.3. Design

2.3.1. MainActivity

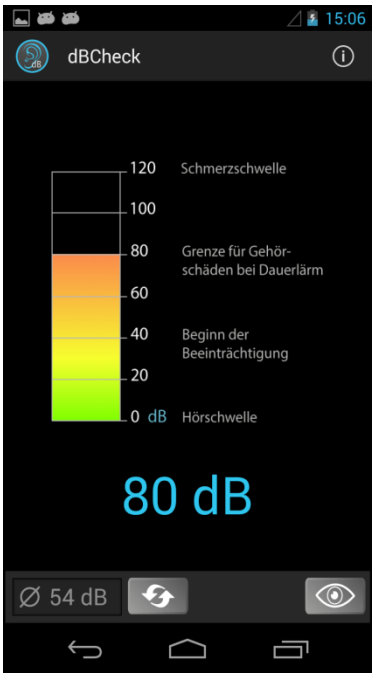

Abb. 2: MainActivity/Progressbar

ProgressBar:

Eine mögliche Darstellung der Umgebungslautstärke ist die vertikale Progressbar, dessen Höhe und Farbe vom aktuellen Dezibel-Wert abhängig ist.
Zusätzlich werden Angaben zum menschlichen Empfinden bzw. zur Auswirkung auf den Menschen gemacht.

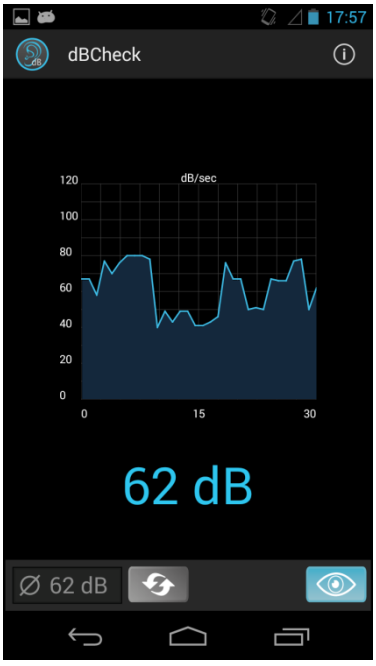

Abb. 3: MainActivity/Graph

LineGraphView:

Als zweite Darstellungsform kann ein Graph gewählt werden. Dieser zeigt die Dezibel-Werte der letzten 30 Sekunden an.

Zwischen beiden Darstellungsarten kann mit dem Auge-Button gewechselt werden.
Der Durchschnittswert kann über den Pfeil-Button zurückgesetzt werden.
Wird einer der Button ausgewählt, leuchtet dieser blau auf (s. Abb. 3).

Wird der Button in der Actionbar geklickt, gelangt der User zur Info der Anwendung.

2.3.2. InfoActivity

Abb. 4: InfoActivity

Die Infoseite klärt über die Bedienung von dBCheck auf. Des Weiteren erfährt der Nutzer Allgemeines über den Schalldruckpegel sowie die Mikrofonabhängigkeit und kann sich eine Tabelle mit Beispiellautstärken angucken.

2.3.3. Verworfene Layouts

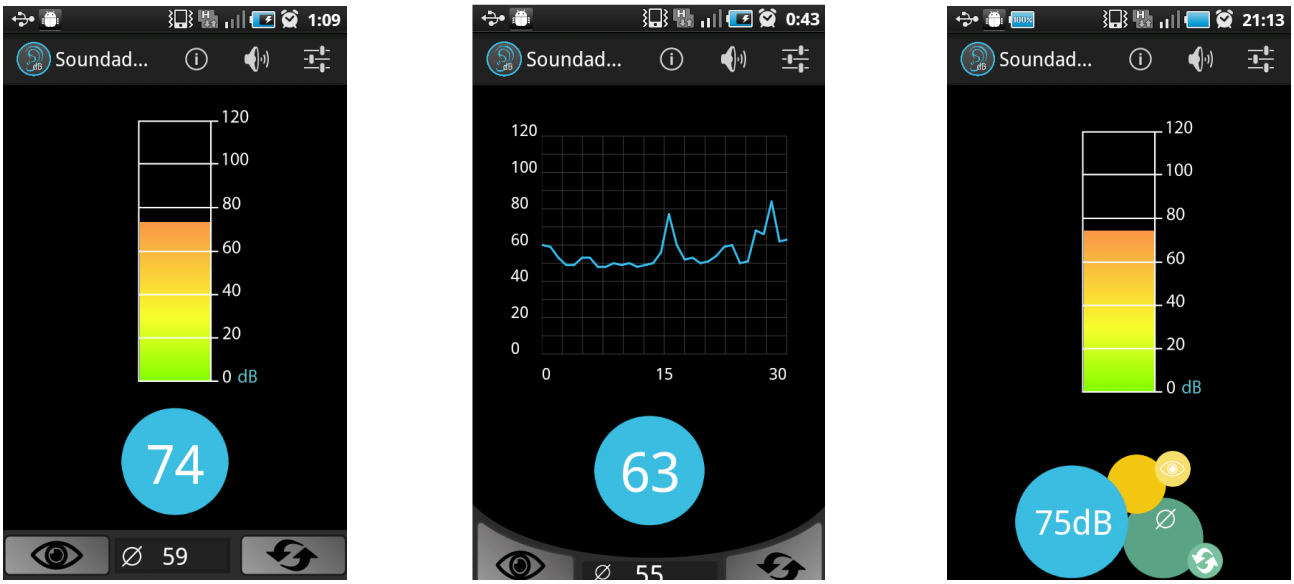

Abb. 5: verschiedene Entwicklungsstände des Designs

3. Projektverlauf

Zur Umsetzung unserer Android-Anwendung wurde mit einem lokalen Repository gearbeitet.
Die Endfassung des Projekts wurde zur Einsicht zu GitHub (https://github.com/HogIng/MobileSysteme.git/) hochgeladen.

Anlässlich der ersten Erfahrung mit der App-Entwicklung von dBCheck, strebten wir den größtmöglichen Wissensaustausch an, der durch ein gemeinsames Arbeiten in mehreren Sitzungen erreicht wurde. Auf diese Weise konnten Probleme und Ideen umgehend besprochen und umgesetzt werden.

Abb. 6 zeigt sowohl die Dauer als auch den Inhalt aller Sitzungen.

Datum	**Dauer**	**Thema**
18.05.2014	14:00Uhr - 17.00Uhr	Ideenfindung
19.05.2014	18:00Uhr - 22:00Uhr	Konzeptausarbeitung
21.05.2014	10.00Uhr - 14:30 Uhr	Recherche zur Lautstärkemessung (Schalldruckpegel), UML-Diagramm, Projekterstellung
25.05.2014	12:00 Uhr- 21:00Uhr	Einbindung MediaRecorder, Überlegung zur Schalldruckformel, Wertausgabe
29.05.2014	18:30 Uhr - 22:00 Uhr	Bezugswert für Formel ermitteln, Recherche zu Mikrofonkapazität,
1.06.2014	11:00 Uhr - 19:20 Uhr	ProgressBar hinzufügen, Darstellung anpassen
2.06.2014	13:00 Uhr- 17:00Uhr	Thread einbinden, Aktualisierung der ProgressBar
4.06.2014	10:45 Uhr - 18:00Uhr	Layout ProgressBar überarbeiten, UML-

		Diagramm anpassen
5.06.2014	10:30Uhr-20:20 Uhr	Durchschnittsfunktion hinzufügen, Darstellung, Reset-Button, Funktionalität testen, Fehlerbehebung
10.06.2014	14:30Uhr- 19:00Uhr	Grafiken erstellen, Recherche zu GraphView
12.06.2014	11:00Uhr- 18:00Uhr	GraphView einbinden, Methoden hinzufügen
15.06.2014	10:30Uhr-18:00 Uhr	Verbesserung d. Darstellung des Graphen, Ansichtswechsel
16.06.2014	20:00Uhr-22.30Uhr	Design überarbeiten & verbessern
17.06.2014	14:00Uhr- 24.00Uhr	Actionbar, InfoActivity, Farbwechsel Button
19.06.2014	9:30Uhr- 21:00Uhr	Activity Statusanpassungen , Testdurchläufe , allgemeine Überarbeitung
20.06.2014	11:30Uhr- 24:00Uhr	letzte Testläufe, Plakaterstellung, Doku
23.06.2014	9:00Uhr- 16:30Uhr	Doku fertigstellen

Abb. 6: tabellarischer Projektverlauf