Vorgehen Applikationsentwicklung

# Aufbau der Applikation

Nachfolgend wird der Aufbau von TI SensorTag zusammen mit unserem selbst entwickelten Board als Sensor benannt und die Android Applikation auf dem Smartphone wird einfach als Applikation benannt.

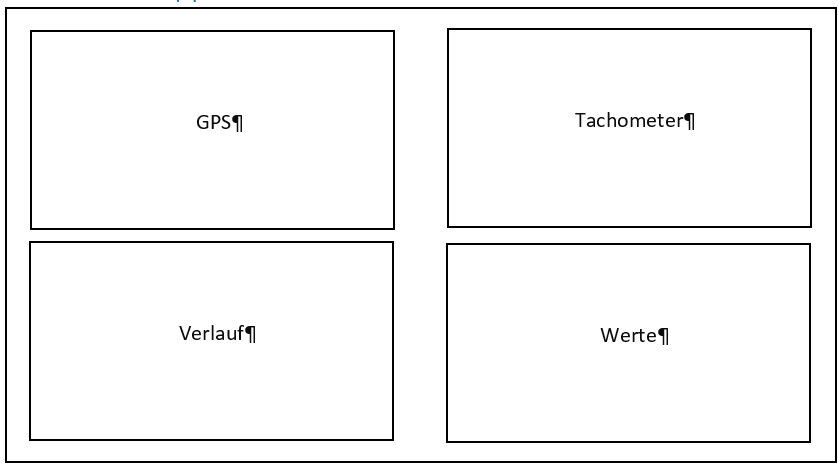


Abbildung : Aufteilung des Bildschirms

Der Aufbau der Applikation soll Anfangs in vier Teile des Bildschirms geteilt werden.

Im Bereich GSP soll eine Karte angezeigt werden, ausserdem soll es möglich sein die Route aufzuzeichnen und abzuspeichern.

Der Verlauf zeigt die Entwicklung der Geschwindigkeit über einen gewissen Zeitraum, wobei es wählbar sein soll welche Daten angezeigt werden.

Die Geschwindigkeit soll anschaulich in einem Tachometer angezeigt werden und die Tachonadel soll im besten Fall animiert sein.

Im Bereich der Werte sollen die aktuellen Werte in Zahlen und den entsprechenden Einheiten dargestellt werden.

Schnell wurde klar, dass dieser Aufbau wenig Sinn macht, da die Bereiche auf einem Smartphone viel zu klein wären und nicht mehr leserlich sein würden. Darum wurde ein neues Konzept erarbeitet, welches in der Abbildung x ersichtlich ist.

## Home-Screen

Als erstes soll der Benutzer den sogenannten Home-Screen sehen, hier werden der Tachometer, die Werte und ein paar Buttons angezeigt. Je nach Anzahl der Funktionen sollen neue Buttons implementiert werden. So soll beispielsweise für die Sensorwahl ein Button vorhanden sein. Während der Entwicklung der Applikation wurde die Idee von einer GPS-Karte und einem Verlauf ebenfalls verworfen, da diese Funktionen aus zeitlichen Gründen nicht mehr realisierbar waren. Jedoch wäre der Home-Screen in der Lage mehr Buttons aufzunehmen, es muss nur darauf geachtet werden, dass die Anzeige der Werte nicht zu gross wird.

## Sensorwahl

Schnell wurde klar, dass die Auswahl eines spezifischen Sensors wichtig werden würde, für den Fall, dass einmal mehrere Sensoren vorhanden wären. Ein Beispiel, würde unsere Applikation bei einem Fahrradrennen eingesetzt werden, so müsste die Applikation auf dem Smartphone mit dem Sensor am Fahrrad gepaart werden. Ansonsten würde man die Daten von allen Sensoren empfangen und die Anzeige würde nicht die Daten des eigenen Sensors anzeigen.

## Einheiten und Einstellungen

Es kam ebenfalls der Wunsch auf, die Einheiten der Werte, einstellbar zu machen, da die Applikation bestenfalls auch in anderen Ländern verwendet werden soll. Also wäre es optimal, wenn die Geschwindigkeit beispielsweise auch in Miles per Hour dargestellt werden könnte. Vielleicht müsste auch eine Einstellungsmöglichkeit bereitgestellt werden, damit die Temperatur kalibriert werden kann.

# BLE-Kommunikation

Der wichtigste Teil der Applikation ist die Bluetooth-Kommunikation, hier werden die Daten vom Sensor empfangen. Die Daten müssen nach dem Empfangen gefiltert werden, damit sichergestellt wird, das die empfangenen Daten von einem unserer Sensoren kommen. Sobald sichergestellt ist, das die empfangenen Daten zu unserem Sensor gehört, müssen diese umgerechnet werden.

## Inbetriebnahme Code PA

Als erster Schritt wurde der Code der vorangegangenen Arbeit analysiert und die wichtigen Teile zur Bluetoothkommunikation übernommen. Es wurden folgende Funktionen aus dem Code übernommen: onStart, onActivityResult, scanLeDevice, BluetoothAdapter.LeScanCallback.

Die Funktion onStart wurde in BLE\_init umbenannt und der Mechanismus zum Wachhalten des Geräts wurde entfernt. Der Mechanismus wurde entfernt, da dieser nicht mehr funktionierte. Ebenfalls wurde die Funktion in die Funktion onCreate eingefügt, um den Code übersichtlicher zu gestalten und da diese Funktion nur einmalig aufgerufen werden musste.

Die Methoden onActivityResult und scanLeDevice wurden komplett übernommen.

Die Callback-Funktion des Bluetoothadapters BluetoothAdapter.LeScanCallback wurde strukturell übernommen, jedoch wurde die Codierung zur Verarbeitung der Daten entfernt.

## Filter einbauen

Der nächste logische Schritt war, dass die empfangenen Daten gefiltert werden mussten. Dafür wurde folgende Struktur definiert, damit festgestellt werden kann, ob die Daten von einem unserer Sensoren versendet wurden.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Length | Type | UUID | Paket-ID | Speed | Pressure | Temperature | Huminity | Checksumme |
| 23 | 0x03 | 0xDEBA | 2 Byte | 4 Byte | 4 Byte | 4 Byte | 4 Byte | 2 Byte |

Aufgrund dieser Struktur können alle empfangenen Daten, welche nicht die definierte Länge, Typ und UUID besitzen, ignoriert werden.

Wenn sichergestellt ist, dass die Daten von unserem Sensor stammen, musste noch die Adresse vom Sender gespeichert werden. Alle Adressen werden in einer Liste gespeichert, vor dem Speichern wird jedoch noch geprüft, ob die Adresse bereits in der Liste ist, falls ja wird die Adresse nicht noch einmal gespeichert.

## Berechnung der Daten

Der Sensor sendet Werte, wie den Zeitunterschied zwischen zwei Magnetdurchläufen, dem Luftdruck, der Temperatur und die relative Luftfeuchtigkeit. Diese Daten müssen erst noch in die realen Werte umgerechnet werden, da alle Werte in mehreren Bytes verteilt vorliegen.

Der Zeitunterschied zwischen zwei Magnetdurchläufen wird in vier Bytes dargestellt. Die ersten beiden Bytes enthalten die Anzahl Sekunden, es muss also das erste Byte um acht Bit geschoben werden und anschliessen muss das zweite Byte noch addiert werden. Folgende Formel zeigt die Umrechnung von zwei Bytewerten in eine reelle Zahl:

Die erhaltene Zahl muss noch mit einem Typecast in den Datentyp float umgewandelt werden.

Das dritte und vierte Byte des übermittelten Zeitunterschieds enthält die Sekundenbruchteile. Folgende Formel zeigt, wie die Bytes verrechnet werden müssen, um die korrekten Werte zu erhalten:

Der erhaltene Sekundenbruchteil kann nun zu den ganzen Sekunden dazu addiert werden, um die ganze Zeit zwischen zwei Magnetdurchläufen zu erhalten.

Ein Beispiel, wird also der hexadezimale Wert 0x00014000 empfangen sind das 1.25 Sekunden zwischen zwei Magnetdurchläufen.

Um die Geschwindigkeit zu erhalten muss der Radumfang durch die erhaltene Zeit geteilt werden. Jedoch befinden sich an unserem Rad zwei Magnete, deswegen darf nur der halbe Radumfang verrechnet werden. Ausserdem ist das Resultat in m/s und muss deshalb noch in km/h umgerechnet werden, was nichts anderes als eine Multiplikation mit dem Faktor 3.6 ist.

Der Druck wird in Pa übertragen, hier müssen die verschiedenen Bytes miteinander addiert werden und in hPa umgerechnet werden. Es werden jedoch nur die letzten drei Bytes mit Werten befüllt sein, das erste Byte kann ignoriert werden. Folgende Formel zeigt die Umrechnung der Bytes in eine float Zahl.

Aus dem Druck lässt sich mittels barometrischer Höhenformel die Höhe über Meer berechnen.

# Einstellungsmöglichkeiten

## Adressauswahl

## Einheiten

## Kalibrierung

# Animierter Tachometer

## Funktionsweise

## Mathematik der Tachonadel