|  |
| --- |
| Mobile Applikationen |
| Dokumentation |
| Bluetooth Chat |

|  |
| --- |
| Niklas Röhling, Niklas Timme, Nils Dralle  27.6.2016 |

Inhalt

[Bluetooth 2](#_Toc454872140)

[GUI 2](#_Toc454872141)

[Material Design 2](#_Toc454872142)

[Tabs 2](#_Toc454872143)

[NavigationView 3](#_Toc454872144)

[SwipeRefreshLayout 3](#_Toc454872145)

[Andere Geräte finden 3](#_Toc454872146)

[Bluetooth in Android 5](#_Toc454872147)

[Services anbieten 6](#_Toc454872148)

[Daten senden und empfangen 7](#_Toc454872149)

[Datenverarbeitung 7](#_Toc454872150)

Bluetooth

Bluetooth ist eine Funktechnologie, die den Datenaustausch mit anderen Geräten ermöglicht. Die Reichweite von Bluetooth beträgt ~10m. Eine Basis-Station (wie z.b. bei WLAN oft verwendet) ist nicht nötig, stattdessen stellen zwei Geräte eine direkte Verbindung miteinander her.

GUI

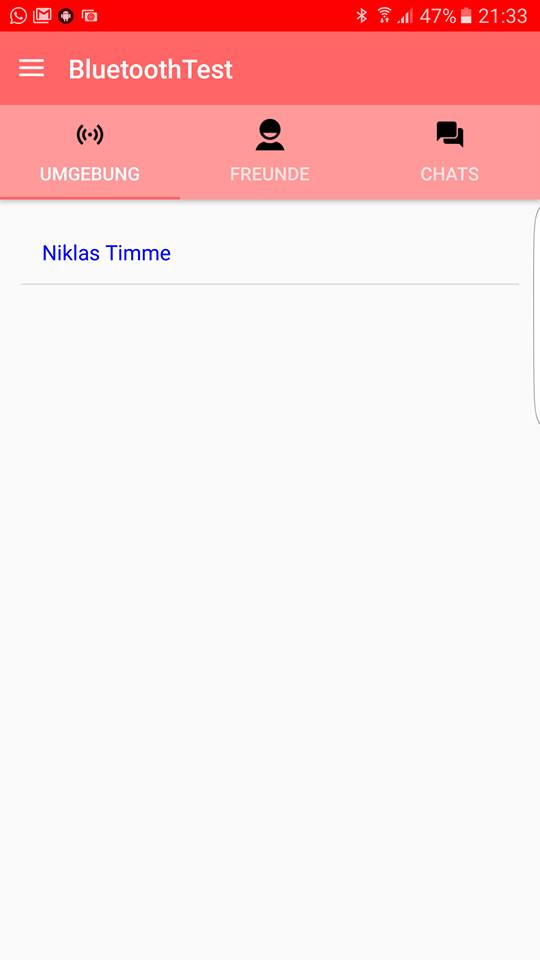
Die Grafische Benutzeroberfläche (GUI) ermöglicht dem Benutzer die Interaktion mit dem System. Sie bietet dem Benutzer die Möglichkeit andere Systeme, welche sich in der Reichweite des jeweiligen Bluetooth-Modul befinden und deren Bluetooth System aktiv ist, anzuzeigen und eine Kommunikation zu beginnen. Außerdem ist es möglich dem Benutzer die bereits bekannten Systeme, wie auch bereits geführte Unterhaltungen anzuzeigen. Zusätzlich wird dem Benutzer die Möglichkeit geboten sein Profil zu betrachten, zu verändern sowie andere Einstellungen vorzunehmen. Wie die die GUI im Detail mit dem Benutzer interagiert wird im folgendem Abschnitt erläutert.

Material Design

Tabs

Um dem Benutzer die Möglichkeit bieten zu können schnell zwischen verschiedenen Ansichten wechseln zu können wurden Tabs für das System verwendet. Der Benutzer kann durch Antippen der jeweiligen Registerkarte des Tabs schnell zu dessen Inhalt gelangen. Indem der Benutzer mit einem Finger von links nach rechts bzw. von rechts nach links wischt kann der Benutzer ebenfalls die einzelnen Tabs wechseln. Um die Hauptfunktionen der Anwendung dem Benutzer schnell visualisieren wird im ersten Tab eine Liste der in der Umgebung gefundenen Geräte angezeigt (Abbildung 1). Mit dem zweiten Tab werden dem Benutzer die Geräte bzw. Kontakte angezeigt mit denen sich das Gerät vorher bereits verbunden hat. Im dritten Tab werden dem Benutzer die zurzeit offenen Unterhaltungen angezeigt.

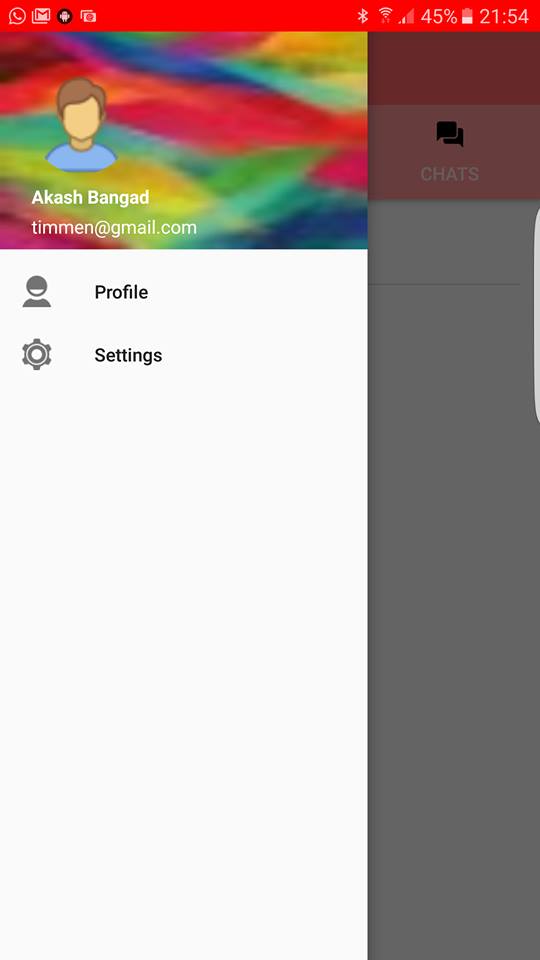
Abbildung 1: Tabs



NavigationView

Mit dem NavigationView Element wird Benutzer durch das Wischen mit einem Finger von links nach rechts oder durch Antippen auf die drei Balken (Abbildung 1) ein Navigationsmenü angezeigt. Dort kann der Benutzer sein Profilbild erkennen und zwischen 2 Menüpunkten wählen: Profil und Einstellungen (Abbildung 2).

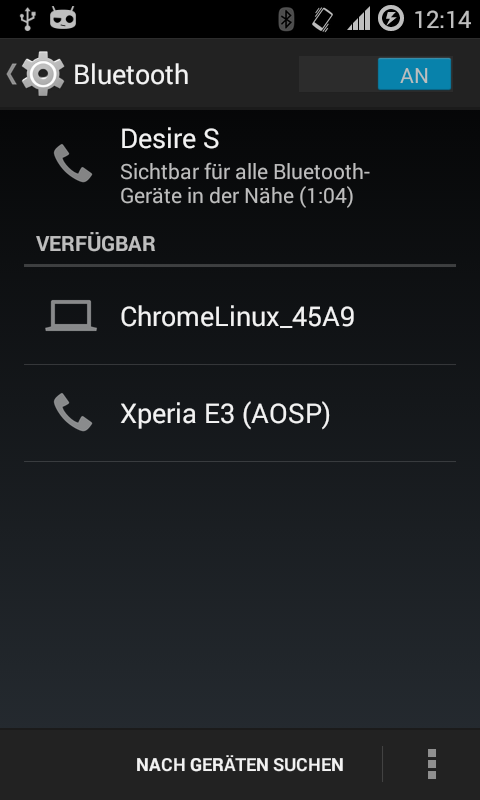
Abbildung 2: NavigationView



SwipeRefreshLayout

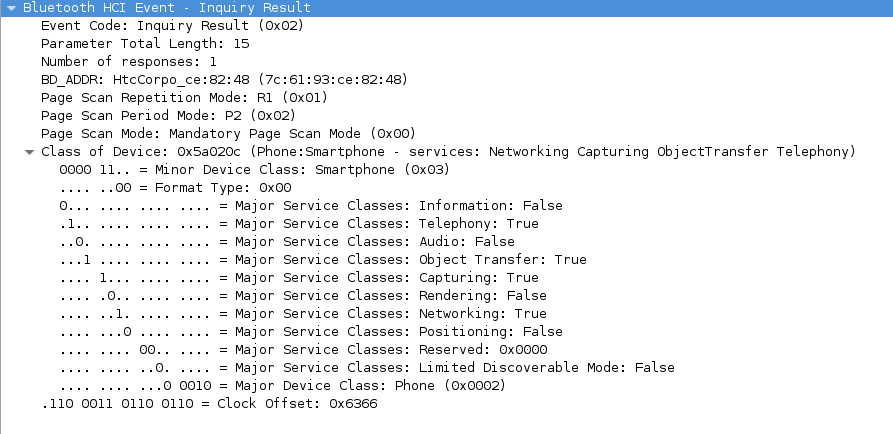
Andere Geräte finden

Um andere Bluetooth-Geräte in der Umgebung zu finden, muss die Umgebung nach anderen Geräten gescannt werden. Dabei sendet das scannende Gerät spezielle Datenpakete aus (Inquiry), die von anderen Bluetooth-Geräten beantwortet werden, vorausgesetzt, das andere Gerät ist „sichtbar“.[Cross und Hoeckle, 2007]

  
Abbildung 1: Bluetooth-Scan in Android

Zu den Geräten, die ein Bluetooth-Gerät über sich preisgibt, gehören: [Evans und Warren, 2009]

* Der Name des Gerätes. Dieser kann oftmals vom Benutzer festgelegt werden.
* Die MAC-Adresse des Geräts, welche das Gerät eindeutig identifiziert.
* Die „Device major class“ zur Bestimmung des Gerätetyps.
* Die „Device minor class“ zur Bestimmung der Unterklasse eines Gerätetyps.
* Die „Services“, die das Gerät aktuell anbietet.

  
Abbildung 2: Ein Inquiry Result in Wireshark

Die Services, die ein Gerät anbietet, werden über sog. UUIDs bekanntgegeben. UUIDs sind 128 Bit lange, eindeutige Nummern. Jeder Service bekommt eine UUID. Ein Service kann Beispielsweise ein offener RFCOMM-Socket sein.

Bluetooth in Android

Android bietet eine Programmierschnittstelle zur Steuerung der Bluetooth-Hardware eines Geräts. Dafür ist die Berechtigung „android.permission.BLUETOOTH“ nötig, für einige Funktionen allerdings auch „android.permission.BLUETOOTH\_ADMIN“. Seit Android 6 ist die Berechtigung „android.permission.ACCESS\_COARSE\_LOCATION“ notwendig, welche zusätzlich zur Laufzeit eingeholt werden muss.

  
Abbildung 3: Berechtigungen "AndroidManifest.xml"

Der Bluetooth-Scan wird mit der Methode „BluetoothAdapter.startDiscovery“ initiiert.

Um die Ergebnisse zu erhalten, muss ein Broadcast-Receiver für „ACTION\_FOUND“ registriert.

  
Abbildung 4: Registrierung des Broadcast-Receivers

Der Broadcast-Receiver muss nach Benutzung am System abgemeldet werden.

Die Ergebnisse werden im Broadcast-Receiver gesammelt und in eine lokale Liste gespeichert. Um die Services zu erhalten, die ein Gerät anbietet, muss auf für jedes Gerät die Methode „fetchUuidsWithSdp“ aufgerufen werden. Dies muss nach dem Ende des Scans geschehen, und die Geräte müssen nacheinander gescannt werden. Um die Ergebnisse des Service-Scans zu erhalten, muss der Broadcast-Receiver zusätzlich für „ACTION\_UUID“ registriert werden.

Die erhaltene Liste von UUIDs wird mit den in der App gespeicherten UUIDs verglichen, welche zum Anbieten von Verbindungen genutzt werden.

Services anbieten

Der Aufbau von Bluetooth-Verbindungen in Android funktioniert ähnlich wie der Aufbau von TCP-Verbindungen. Auch für Bluetooth-Verbindungen gibt es einen Server und einen Client, die dafür zuständigen Klassen heißen „BluetoothServerSocket“ und „BluetoothSocket“. Eine Server-Socket wird mit der Methode „listenUsingRfcommWithServiceRecord“ unter Angabe eine 128-Bit UUID erstellt. Ein Client kann sich dann einen Bluetooth-Socket erzeugen und sich mithilfe der Methode „createRfcommSocketToServiceRecord“ unter Angabe der selben UUID verbinden. Auf dem Server wird für die Kommunikation über die neue Verbindung ein neuer Socket erstellt. Zu beachten ist hierbei, das dies nur dann funktioniert, wenn die Geräte vorher gekoppelt wurden.

Seit API-Level 10 stellt Android die Methoden „listenUsingInsecureRfcommToServiceRecord“ und „createInsecureRfcommSocketToServiceRecord“ bereit, die eine Verbindung auch zwischen nicht gekoppelten Geräten ermöglichen. In der App werden je nach Kopplungsstatus der Geräte beide Varianten verwendet.

Der Socket stellt einen Input- und einen Outputstream bereit, über die Daten ausgetauscht werden können. Es ist zu empfehlen, diese Streams in ihren jeweiligen Buffered- Versionen zu kapseln, da gesendete und empfangene Nachrichten sonst oftmals nur in Teilen gesendet werden.

Daten senden und empfangen

Um abzufragen, wie viele Daten an einem Inputstream bereitstehen, kann die Methode „Inputstream.available()“ genutzt werden. Diese Methode gibt allerdings je nach Gerätehersteller und Android-Version nicht die tatsächliche Anzahl an Bytes zurück, die bereit steht, sondern nur, ob Daten bereitstehen.

Um die bereitstehenden Daten zu erhalten, wird die Methode „Inputstream.read()“ verwendet. Mithilfe dieser Methode werden die Daten in ein Byte-Array geschrieben und können anschließend in einen String umgewandelt werden.

Der umgekehrte Weg, das Senden von Daten, geht etwas einfacher: Der zu sendende String wird in ein Byte-Array umgewandelt und mithilfe von „OutputStream.write()“ in den Stream geschrieben. „OutputStream.flush()“ sorgt dann dafür, dass die Daten gesendet werden.

Datenverarbeitung

Um die Verarbeitung der empfangenen Daten zu vereinfachen, werden alle übertragenen Nachrichten mit JSON formatiert. Android stellt dazu im Package „org.json“ einige Klassen zur Verarbeitung von JSON bereit.

  
Abbildung 5: Beispiel einer mit JSON formatierten Nachricht

Abbildungsverzeichnis

[Abbildung 1: Bluetooth-Scan in Android 2](#Abbildung!0|sequence)

[Abbildung 2: Ein Inquiry Result in Wireshark 3](#Abbildung!3|sequence)

[Abbildung 3: Berechtigungen "AndroidManifest.xml" 3](#Abbildung!1|sequence)

[Abbildung 4: Registrierung des Broadcast-Receivers 3](#Abbildung!2|sequence)

[Abbildung 5: Beispiel einer mit JSON formatierten Nachricht 4](#Abbildung!4|sequence)

Literaturverzeichnis

Cross und Hoeckle, 2007: Daniel Cross, Justin Hoeckle, Michael Lavine, Jason Rubin and Kevin Snow, DETECTING NON-DISCOVERABLEBLUETOOTH DEVICES, 2007

Evans und Warren, 2009: David Evans, Robert H. Warren, Anonymity properties of stored or transmitted datataken from Bluetooth scans, 2009