

Entwurfsbeschreibung_release2

hg17b - Android App für Weiterbildungsmanagement

Julian Dietz, Marcel Herhold, Nikolai Kortenbruck

12. März 2018

Inhaltsverzeichnis

1 Allgemeines	2
2 Produktübersicht	2
3 Grundsätzliche Struktur- und Entwurfsprinzipien	2
4 Struktur- und Entwurfsprinzipien einzelner Pakete	2
4.1 Server	2
4.1.1 Server(Klasse)	2
4.1.2 Commands	2
4.1.3 Handler	3
4.1.4 PupilDB	3
4.1.5 Mail	3
4.2 Applikation	3
4.2.1 Client	3
4.2.2 StartActivity	3
4.2.3 Main	3
4.2.4 Nextevents	3
4.2.5 Lastevents	4
4.2.6 Ranking	4
4.2.7 Setting	4
4.2.8 Logout	4
4.2.9 OrganizerLogin	4
4.2.10 OrganizerMain	4
4.2.11 OrganizerNextevents	4
4.2.12 OrganizerLastevents	4
5 Datenmodell	5
6 Glossar	6

1 Allgemeines

Wir entwickeln eine Android-App für Schüler und Veranstalter von Weiterbildungsevents. Schüler bekommen eine Übersicht über ihre bereits besuchten Veranstaltungen und haben die Möglichkeit, kommende Veranstaltungen zu finden und sich unverbindlich anzumelden. Veranstalter können in der App die tatsächlich anwesenden Schüler, mithilfe einer dem Schüler zugeteilten ID, erfassen.

2 Produktübersicht

Unser Produkt besteht aus einem Frontend und einem Backend.

Das Frontend ist die Android App. Beim Start der App wird unterschieden ob der Nutzer ein Schüler oder ein Veranstalter ist. Schüler geben dabei ihre ID ein und erhalten Einblick in ihre Eventdaten, zukünftige Events sowie ihren Platz in der Bestenliste. Veranstalter können sich nach einem Klick auf die untere Schaltfläche autorisieren und anschließend die Teilnehmer einer Veranstaltung erfassen. Das Frontend wurde mithilfe von Android Studio entwickelt.

Unser Backend besteht aus einem Server mit RDF-Triplestore in welchem die Eventdaten und die Daten der Schüler gespeichert werden. Der Server nutzt hierfür das Apache-Jena Framework.

3 Grundsätzliche Struktur- und Entwurfsprinzipien

Die Daten der Schüler (IDs, Score, Rang) liegen zentral auf einem Server. Die Kommunikation zwischen mobiler Android App und Server laufen über eine auf dem Server liegende Java Applikation. Diese stellt einen Server Socket zur Verfügung über den ein Client Socket der App mit dem Server kommuniziert.

4 Struktur- und Entwurfsprinzipien einzelner Pakete

4.1 Server

Die (Schüler-) Daten auf dem Server liegen im SPARQL-Format als Triples in einem Graphen vor und werden von der PupilDB Java-Klasse aus abgerufen und manipuliert. Diese definiert die dafür benötigten SPARQL- Query und Update Statements. Die Server Klasse der auf dem Server laufenden Java Applikation handelt den Verbindungsauflauf und die Abfragen/Antworten vom/an den Client. Nachfolgend noch eine Auflistung der Klassen welche zum Server gehören

4.1.1 Server(Klasse)

Diese Klasse ist die Main Klasse des Servers und erstellt die Datenbank und startet den Server. Meldet sich ein Client an so wird ein Objekt der Klasse Handler erstellt. Außerdem kann der Server eine Liste mit allen aktiven Clients anzeigen

4.1.2 Commands

Durch diese Klasse werden die Befehle für die Serveradministratoren zur Überwachung des Servers bereitgestellt. Im moment währen dies: "list"(eine liste aller verbundenen Clients #stop#um herunterfahren des Servers "help um alle Befehle aufzulisten

4.1.3 Handler

Handler ist die Klasse welche sich um die Kommunikation zwischen Client, Server und Datenbank kümmert dabei wartet sie auf Anfragen des Clients und sendet Daten aus PupilDB und wird beendet wenn der Client offline ist.

4.1.4 PupilDB

Die Klasse PupilDB stellt die Datenbank des Projekts und liefert Methoden zum manipulieren der Schüler sowie zum lesen der Schüler- und Veranstaltungsdaten. Die Schülerdaten werden dabei als Triple Graphen gespeichert. Die Methoden nutzen SPARQL-Querys und Update Statements.

4.1.5 Mail

Die Klasse Mail wird genutzt um automatisch E-Mails zu verschicken. Wenn ein Veranstalter sich zum ersten Mal in der App anmeldet, schickt der Server ihm über diese Klasse eine E-Mail mit einem Bestätigungslink. Dieser Link aktiviert das ssh Schlüsselpaar.

4.2 Applikation

Die Kommunikation mit dem Server wird in der Client Klasse der App gehandelt. In der StartActivity Klasse wird das Start Verhalten und der Login der Nutzer definiert. Die Klassen Logout, NextEvents, Ranking und Settings (zu finden in app/src/main/java/hg17b) sind jeweils für die Methoden der in der App vorhandenen Seiten Logout, Nächste Events, Ranking und Einstellungen zuständig. Die Layouts der Seiten werden in den dementsprechenden fragment xml Dateien definiert (zu finden in app/src/-main/res/layout). Das Layout des Hauptmenüs hingegen wird in der navigation_\menu unter /\verb /App/app/src/main/res/menu definiert. Hier noch eine Auflistung der Klassen der App

4.2.1 Client

Die Klasse Client erstellt einen Hintergrundthread welcher sich mit dem Server verbindet. Sie sendet und empfängt Daten von der Datenbank des Servers

4.2.2 StartActivity

Diese Klasse ist die Startseite der App und definiert beim starten das Layout und erstellt eine Instanz der Klasse Client. Nach dem klicken auf "Bestätigen" wird die eingegebene ID mit der Datenbank abgeglichen und wenn diese vorhanden ist wird zur Klasse Main weitergeleitet. Von hier aus kommt man auch zum Organizer login

4.2.3 Main

Die Klasse Main bestimmt das Design unserer Hauptseite. Das ist die Seite welche ein Schüler nach dem Einloggen als erstes sieht. Oben wird die eigene ID angezeigt, in der Mitte erfährt der Schüler wie viele Punkte er hat und unten werden bevorstehende Veranstaltungen angezeigt, für welche der Schüler sich angemeldet hat

4.2.4 Nextevents

Die Klasse NextEvents zeigt die bevorstehenden Veranstaltungen an. Dazu nutzt sie die Eventdatenbank, welche seit dem 2. Release integriert ist.

4.2.5 Lastevents

Die Klasse LastEvents zeigt die vom eingeloggten Schüler besuchten Veranstaltungen an. Auch diese Klasse nutzt dazu die Eventdatenbank.

4.2.6 Ranking

Die Klasse zeigt die Topliste, also die höchsten Punktzahlen in absteigender Reihenfolge, geliefert aus der Klasse PupilDB. Unter der Rangliste wird die Platzierung des angemeldeten Schülers gezeigt.

4.2.7 Setting

In dieser Klasse sollen dem Schüler Einstellungsmöglichkeiten zur Individualisierung der App geboten werden. Zurzeit ist die implementierung dieser Funktionen ein Kann-Ziel vom 2. Release.

4.2.8 Logout

Die Klasse LogOut dient zum trennen der Serververbindung, klickt man auf den Button abmelden, so wird der Hintegrundthread geschlossen und man landet auf der StartActivity Seite um sich erneut anzumelden.

4.2.9 OrganizerLogin

Mit dieser Klasse können sich Veranstalter anmelden. dazu benötigen sie eine verifizierte E-Mail Adresse. Mit klick auf bestätigen wird das Menü für Veranstalter initialisiert.

4.2.10 OrganizerMain

Die Klasse OrganizerMain bestimmt das Layout der Hauptseite für Veranstalter. Nach dem einloggen ist dies die erste Seite die ein Veranstalter sieht. Außerdem werden bevorstehende Veranstaltungen angezeigt.

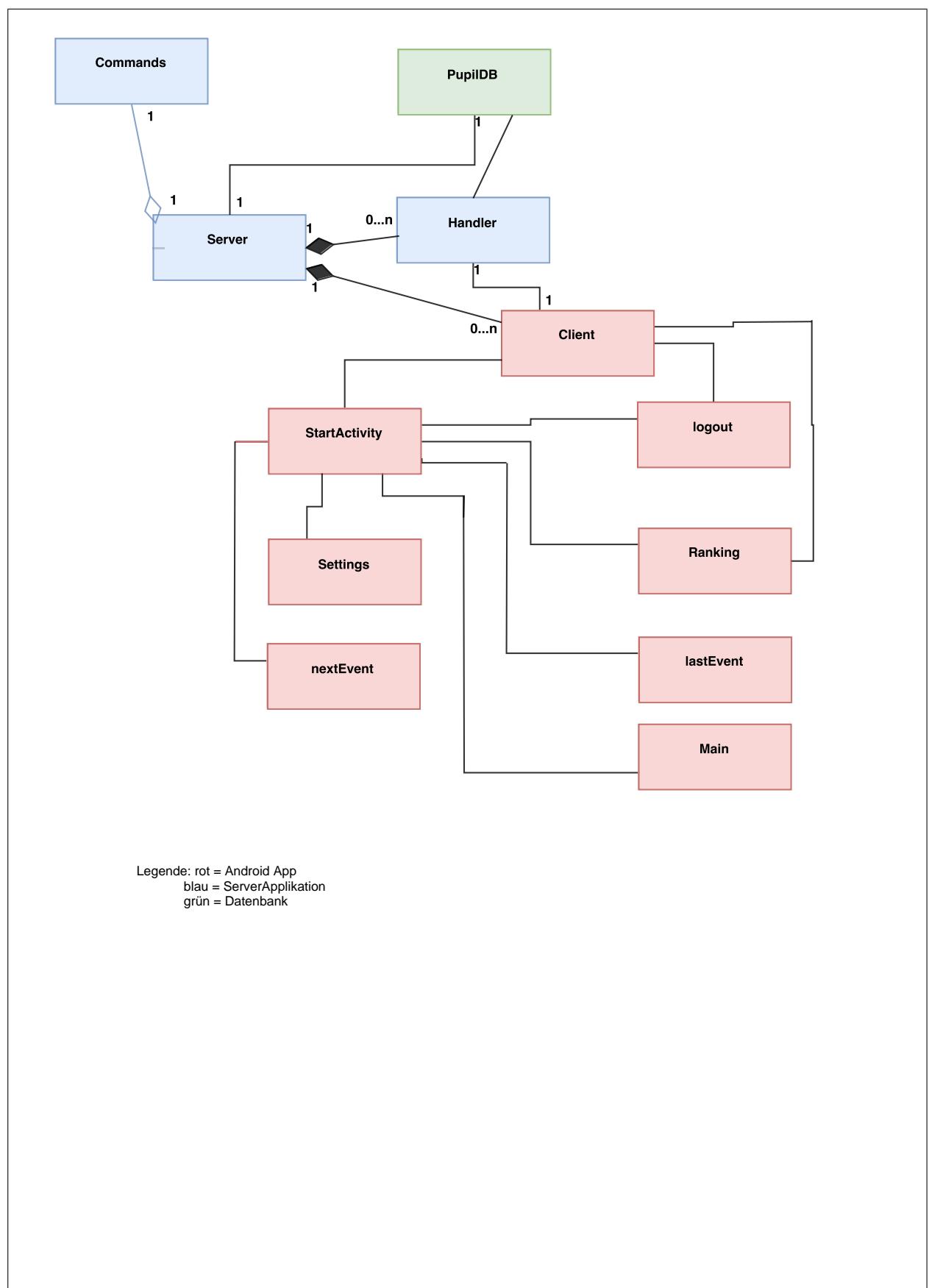
4.2.11 OrganizerNextevents

Hier werden dem eingeloggten Veranstalter kommende Events angezeigt.

4.2.12 OrganizerLastevents

In dieser Klasse kann ein angemeldeter Veranstalter Informationen zu seinen vergangenen Veranstaltungen einsehen.

5 Datenmodell



6 Glossar

Backend Backend beschreibt das auf dem Server laufende Programm, ein SPARQL Endpunkt (Apache Jena), sowie Java und Java Applikationen zum bearbeiten diverser Anfragen.

Frontend Frontend beschreibt das auf dem Endgerät des Nutzers laufende Programm.

ID (Identifikationsnummer) Im Zusammenhang dieses Projektes ist die Identifikationsnummer auf den Pässen der Schüler gemeint.

RDF RDF steht für Resource Description Framework. Es wurde von der W3C als Standard zur Beschreibung von Metadaten konzipiert. Es gilt als ein grundlegender Baustein des Semantischen Webs. RDF ähnelt den klassischen Methoden zur Modellierung von Konzepten wie UML-Klassendiagramme und ER-Modell.

Schüler Der Schüler nimmt am Ferienpass teil. Er soll die App leicht bedienen können.

Veranstalter Der Veranstalter organisiert Freizeitaktivitäten für die Schüler. Er soll mit der App den Code des Schülers einscannen.