Setup QuizApp

Verwendete/Benötigte Ressourcen

Server (für Endbenutzer und Entwickler)

- Java (JRE & JDK), Version >= 7
- Servlet-Container (Tomcat)
- Relationale Datenbank (MySQL)
- Java Entwicklungsumgebung (Eclipse)
- VersionControlSystem (git bzw. Github)
- Google Firebase Account und Autorisierungsdaten
- Google API Libraries
- GSON Library
- EclipseLink
- Lokales Netzwerk, dem der Server sowie Android-Geräte beitreten können

App (für Entwickler)

- Java Entwicklungsumgebung (Android Studio)
- Testgerät(e) (Emulator bzw. Smartphone) mit Android, Version >= 5
- VersionControlSystem (git bzw. Github)
- GSON Library

App (für Endbenutzer)

- Android Gerät(e) mit Android Version >= 5
- QuizApp *.apk* Datei

Hinweis: Der benötigte Sourcecode und weitere Hilfedokumente sind auf Github verfügbar.
https://github.com/OpticWafare/Quiz_App

Server (für Endbenutzer & Entwickler)

Aufsetzen der Datenbank

Nach erfolgreicher Installation von MySQL ist eine neue QuizApp-Datenbank darin zu erstellen. Die Tabellen in der QuizApp-Datenbank werden automatisch von JPA erstellt und müssen daher nicht manuell eingegeben werden.

Server

Ein neues Projekt in Eclipse oder einer anderen Java-Entwicklungsumgebung erstellen und den Code des Servers von Github implementieren.

➔ https://github.com/OpticWafare/Quiz_App/tree/master/SourceCode/Server/Quiz_App

Implementieren Sie die Google Libaries durch:
Rechtsklick auf Projektordner → BuildPath → Configure BuildPath → Libaries → Add JARS → Entsprechende Google JARS im Projekt implementieren → Apply and Close

Stellen Sie in der JPA Persistence Unit (*src/META-INF/persistence.xml*) die richtigen Einstellungen ein:

- Name der Persistence Unit
- Name der Datenbank
- Port der Datenbank
- Name des Users
- Passwort des Users

Stellen Sie im DBManager den richtigen Namen der JPA Persistence Unit ein (Variabel *PERSISTENCE_UNIT_NAME* in *model.DBManager*).

Konnektivität App <-> Server sicherstellen

- Das Smartphone mit der App muss sich im gleichen Netzwerk befinden wie der Server.
- Wenn die App den Server trotzdem nicht erreichen kann, dann kann das an einem Routing-Fehler liegen.
  Beheben kann man das Problem indem man von dem Server aus einen **Ping** an die IP-Adresse des Smartphones sendet.

App (für Entwickler)

- Ein neues Projekt in Android Studio erstellen und den Code von Github implementieren.
  → https://github.com/OpticWafare/Quiz_App/tree/master/SourceCode/Android

- Vergewissern Sie sich, dass in der *build.gradle* Datei die importierten Versionen der Libraries noch aktuell sind, um zu verhindern, dass der Code obsolet ist und daher nicht mehr funktioniert.

- Passen Sie die voreingestellte IP Adresse des Servers, den Port des Servers und den Servletnamen im *SendToServletTask* an, damit Sie beim Starten der App diese nicht immer neu eingeben müssen.

Beim Verwenden eines Emulators

Vergewissern Sie sich, dass bei Verwenden des Android Emulators die richtige Zeitzone eingestellt ist, damit die Pushmitteilungen zum richtigen Zeitpunkt ankommen können.

Google Firebase

Ein Google Konto wird für die Verwendung des Google Firebase Service wird benötigt

Für App-Entwickler

Auf https://firebase.google.com/ → *Get Started* →Projektname und Git Account verlinken →Nutzungsbedingungen akzeptieren → Google Firebase Library in Android Studio implementieren

Hilfe bietet ein internes Tutorial in Android Studio unter dem Reiter Tools → Firebase

Für Server-Entwickler

- Die „*authorization.json*“ Datei von dem erstellten Firebase Projekt herunterladen und in *src/model/* einfügen.

- Einfügen der eigenen API-URL bzw. des eigenen Firebase Projektnamens in die Variable *API_URL* der Klasse *control.NotificationSender*.