Introduction

I am currently working on an Android game. It works like a former TV show where one game involved solving math challenges. In the game, the player gets 6 numbers and using these numbers, the player needs to construct a set goal by additions, subtractions, multiplications and divisions.

For example, when given [1, 2, 3, 4, 5, 6] as numbers and 49 as the goal, the puzzle can be solved by:

- Adding 4 and 6
- Multiplying the result, 10, and 5
- Subtracting 1 from the result, 50.

Of in other words: ((4 + 6) * 10) - 1.

Currently, I am working with a PostgreSQL database that both the backend is talking to (to update levels or add new levels) and the frontend (Android app) is talking to. For this code challenge, I developed a REST-API.

The complete code set (current state) is in multiple repositories available on my GitHub account at: www.github.com/mwbouwkamp

Project

The project consists of two main packages:

- nl.limakajo.numbers.RABO.API
- nl.limakajo.numbers.RABO.levelMaker.

API

The API is a Spring Boot JPA API and contains, a LevelRestControler that provides the endpoints of the API. It includes the basic CRUD operations as well as the option to retrieve a Level with an averageTime that matches the userAverageTime. This way, the user always gets presented a Level that matches its abilities. The averageTime that is saved in the database is updated once a user finishes the level. This way, the degree of complexity of the level stored remains representative for how hard a level is. This, however, is not used in the code provided. This would be something that is updated in the Android app (see my GitHub account).

In addition, the API has a Service level that uses a JPA DAO implementation for the actual interaction with the database.

levelMaker

The levelMaker package contains code for the generation of new levels. Levels are initially generated using random numbers. After that, a Solver will check whether or not a level is actually solvable. If so, the Level is retained. The Solver makes use of a breath-first-search algorithm to check the solvability of a Level. The Main class that acts as an entry point generates 10 solvable Levels and saves them to the database through the API.