RYTLE GmbH
Konsul-Smidt-Straße 20
28217 Bremen

Coding Challenge

Fullstack Development

Imagine a small and a large cube. A small cube can be placed inside a large cube. Four small cubes fit in a large cube. Each cube has **an unique ID (string)**, an **optional owner ID (string or null)** as well as **a type (large or small)**. Please implement the following tasks in TypeScript without using additional packages (apart from Jest and TypeScript, see `./source/package.json`).

Think of the tasks as an API for other developers. Storage of the cubes and presentation are outside of the scope of the challenge.

1. Write an asynchronous stand-alone function that can create cubes. The first parameter should be the type of the cube (small or large). The second parameter is an optional owner ID (think of it as referring to an external entity, *not* the parent). The last parameter is an optional array with unique IDs for creating small cubes (recursive creation with a maximal depth of 1). Consider that a cube that is inside another cube inherits its owner.

   a. The function to be implemented should have this signature:

   ```
   async createCube(
       type: KindOfCube,
       id: string,
       ownerId?: string | null,
       childIds?: string[]
   ): Promise<Cube>
   ```

2. Write an asynchronous stand-alone function for deleting a cube. This function gets the list of all created cubes as the first parameter. The second paramter is the ID of the cube to be deleted. A third optional parameter indicates if cubes that are inside the to-be-deleted cube should also be deleted. This last parameter is false by default. If the contained cubes in the to-be-deleted cube should not be deleted themselves they should have an empty owner afterwards. An error should occur if the given cube ID does not exist.

   a. The function to be implemented should have this signature:

   ```
   async deleteCube(
       cubes: Cube[],
       id: string,
       deleteChilds?: boolean
   ): Promise<Cube[]>
   ```

   The first parameter should not be changed by the function. The return value should indicate which cubes are to be deleted.

3. Check your implemention with unit tests using Jest (https://www.npmjs.com/package/jest). Please test every kind of behaviour that you can derive from this task.

We provide you with a template for this coding challenge in the source folder.

Good luck and have fun - your RYTLE software team