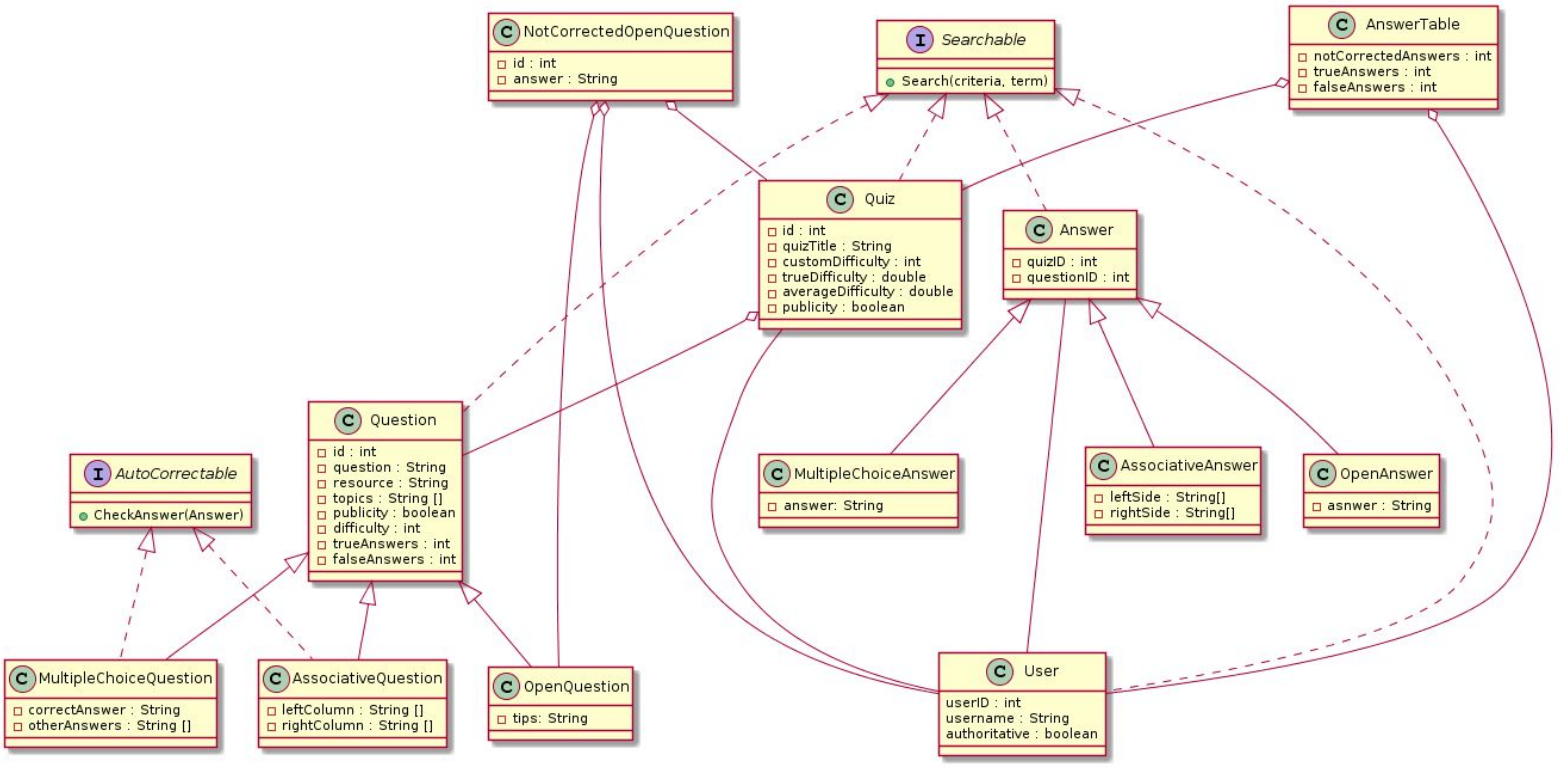
Quizmania

Quiz Manager Technical Details

Intention of this document is to give knowledge about the structure & design of the program and how the program has been implemented

Deniz Can Erdem YILMAZ
20.02.2019



Classes

1. Question

Question class is one of the main classes of the program. It is an abstract class used to derive other specific type questions. But it holds the most information (common attributes of questions) and most of the processes are done through this class. The common attributes includes the question text (body), question owner, topics, resource to show on GUI, question type, publicity, difficulty, correct answer count and false answer count.

- **Multiple Choice Question**
  This is the container for a question where there are multiple choices and one is correct. The user needs to pick one out of 4 choice. Correct choice hold as a single string while other false choices are hold as string array. This way correct and false choices are separated from each other. When solving quiz all 4 questions are randomized in order before shown to user

- **Associative Question**
  Associative question has two columns hold as string lists. The order of user inputs are considered as matched rows. Before showing the question to the user the lists are randomized.

- **Open Question**
  This question only holds a string field for tips and nothing else. There is no strict correct answer for this kind of question

2. User

User is one of the other main classes of the program. It is required to login to the system and use the functionalities. It holds the id, username and authority status. Instead of using the User object itself most of the time the id is used by other classes

3. Quiz

Quiz is the last main class of the program. It is constructed from a list of questions and a user (to indicate owner of quiz).

4. Answer

Answer is a helper container that holds an answer for a specific question. It is an abstract class to derive question type specific answers out of it, since the correct answer containers for questions are different for different type of questions.

- **Multiple Choice Answer**
  Since there is only one correct answer for multiple choice question, Multiple Choice Answer holds only answer given by the user. To correct it, only checking if it matches with the correct answer of associated question will be enough.

- **Associative Answer**
  Since there are two columns of data on an associative question, the associative answer also holds 2 string lists. Lists are constructed from the user input from console or GUI. So there are matching rows created with user input. While correcting, corresponding index of left answer on question list must be same with corresponding index of right answer on question list

- **Open Answer**
  Open answer only holds the input given by user. Since there is no strict correct or false answer it will be required to manually evaluated by an authoritative.

5. AnswerTable

Answer table is a container that holds the result of a solved quiz by a single user. It is generally the statistics of a list of answers. Once answers are corrected either automatically or manually the answerTable entry will be updated on database to update statistics. It holds the quiz, user who solved the quiz and number of true, false and not corrected answers. Using the number of true and false answers the evaluation result can be calculated over 100 percent. Not corrected answers will not be included in percentage calculation. So if there are not corrected answers on a result the percentage will be misleading.

6. NotCorrectedOpenQuestion

This is another helper container to hold the answer of open question that is not evaluated yet. It holds the question that is answered, user who answered the question and the quiz which user was solving.

Once the open question answer is evaluated manually the object of NotCorrectedOpenQuestion will both be deleted from database and from the program itself. It will update the AnswerTable entry of given quiz result according to the correctness of the open question.

Interfaces

1. AutoCorrectable

This interface includes a single method named CheckAnswer. It takes a parameter of type Answer. According to the type of question that answered it is going to check the auto correction condition and return true or false according to the result. This interface is used by MultipleChoiceQuestion and AssociativeQuestion since they are auto correctable questions with specific and strict answers. Once a quiz has been solved and finished, Associative and Multiple Choices questions will be evaluated automatically thanks to this interface

2. Searchable

This interface includes a single method named Search. It takes 2 parameters of type String. One is to define the search criteria to be compared and the other is to define the search term to find out which attribute to compare with.

This is used with most of the classes who take place in a table for GUI and console and needs to be searched. Each class implements its own search method to have customized search results for different classes

Controllers

There are multiple controllers for GUI part. All of them are derived from a mother class named Controller to have some common helper methods tho be used while constructing the scene or processing a user input.

Controllers are the main structure of the program. They user the containers introduced on classes part to carry information over scenes and use services to create, update and/or delete from the database and show this changes or ask input for actions from user.

- **Controller**
  This is the main controller class. It has functional helpers such as ChangeScene, SearchHelper, AssociateTableWithClass etc. that each controller uses. This class implemented mostly to prevent code duplication

- **LoginMenuController**
  Login menu controller handles the login and register functionality. It gets inputs from user to either save new user to database or check if the user exists to log in

- **MainMenuController**
  This is the main area for users to pick an action. Instead of playing with data, this controller intended to change the scene to the desired scene of user

- **CreateQuizMenuController**
  This menu lets user create new questions or select existing questions to create new quiz. It gets the existing questions and lists to the user. Mixes selected questions with created questions to save new quiz to the database

- **EditDeleteQuizMenuController**
  This menu lets user to edit any quiz that created by that user. Questions can be removed from quiz, can be entirely removed from database or can be edited. Questions are not duplicated for each user or quiz. So any change done on the question or the quiz will be able seen by the all other users who use the same questions. Once every question of a quiz has been deleted the quiz also deleted from database. To prevent conflictions, the results and open answers waiting to be corrected which related to the question are also removed from database

- **CheckAnswersMenuController**
  This controller builds the menu to show the open question answers which needs manual correction. Once the correct or false button is clicked, it updates the results (AnswerTable) and the question & quiz statistics

- **UserPromotionMenuController**
  This controller lists all existing users to change their authority status. One can demote itself or everyone if it has the authority. This may cause to not having authoritative users without changing the values from database itself manually

- **SolveQuizMenuController**
  This controller handles the inputs given by user to solve the questions. For each question there will be a corresponding Answer created for it. Once each question is solved the results will be saved to the database. AutoCorrectable questions will be corrected and it will updated the statistics of questions and quizzes

- **SeeResultsMenuController**
  This menu shows the user the results of solved quizzes. If the user is authoritative it will show the results of the users who solves the quizzes created by that user. If user is normal user then it will show results of itself.

Services

There are 3 services used to keep the structure and program alive: DatabaseManager, QuizManager and CfgManager

- **QuizManager**
  This is a singleton used to achieve 2 things: controlling the console application navigation, registering users and holding the currently logged in user info to use during the program execution.

- **DatabaseManager**
  This is another singleton used to have predefined database methods to be used by other classes (used mostly by the controllers).

- **ConfigManager**
  This is another singleton to handle the program configuration file given as an argument to the program and store the config elements to be able to give whenever they are required

Launchers

There are 2 different launchers: Main and MainGUI. They mostly do the same things like initializing the singleton classes and setting up their parameters to make them ready for program execution. Main calls QuizManager method to initialize user interaction while MainGUI initializes a graphical interface.