| | |
|---|---|
| **Project title** | **Quiz App** |
| **Author(s)** | **Gîrda Nora Marcela** |
| **Group** | **30424** |

## 1. Task Description

A quiz is a type of evaluation or assessment that usually consists of a series of questions or exercises designed to evaluate a person's knowledge in a specific field .

The goal of the project is to develop an application that lets you make and take quizzes. Its goal is to assess your knowledge across a variety of topics using questions that vary in difficulty. Since playing this with friends might make it enjoyable as well, maybe even fun, I would also classify it as a game. Therefore, this application may have an educational or recreational purpose. It might be used in the educational system, for example, by teachers testing students in a classroom, while it could also be played recreationally with friends as a worthwhile pastime that also teaches you something that maybe you didn’t know.

Therefore, the only data that is relevant to keep a track of and be able to access is the data that makes up a quiz. There will be no track of users and their previous scores, as it is a one time use app and it’s purpose is, above all others, testing the users’ knowledge.

This application, which is an ongoing project implemented using Java and JavaSwing, and will be improved further after also getting some user feedback, with the possibility of adding other features,  will allow you to create or take a quiz that has already been created by someone else, or a quiz that you had just created yourself, with a user-friendly and intuitive interface.

After the application is opened, there will be buttons and instructions displayed on screen to make the application easy to use and maintain its flow. When the user will be asked to sign in, the data will not be retained in memory. After you logged in, you can choose if you want to create a quiz or play one.

Only two types of questions are supported by this quiz application, multiple choice questions with four answering options from which a single one is correct and true/false statements.

If you choose to create a quiz, there is no going back, and you need to finish the creation. When creating a quiz, you first need to input the name and topic of the quiz,

these fields are required and they can't be left empty, and choose the difficulty of the quiz. The options for the difficulty are: easy, medium or difficult. After this step is done you need to input the questions. There is no limit to the number of questions that can be added. You need to select the "add question" button from the bottom of the screen to be able to add a question, each time the user wants to introduce a new question, this button needs to be pressed. Now, from a drop down list, the user can choose if the next question to be added will be a true/false or a multiple choice kind of question. If the next question will be a multiple choice, the user will be asked to input the question, and then the options one by one, and in the end to input the index of the correct answer. When a true/false is introduces, the user only needs to write the statement and choose if it is true or false. When the user is done adding questions, the "done" button needs to be pressed and the user can choose what he wants to do next, he can either play, create more or exit.

Each question has a weight of one in the score, and the purpose of the score is to keep track of the number of questions you correctly answered to, and be an ambition for the user to play the same quiz again util all the answers are answered correctly, this way he can be sure he had learned what he didn't know before.

When the user wants to play, a quiz from the available quiz list needs to be selected. After the quiz is selected the user will need to decide whether the quiz will be played in single mode, "One Man Show", or the user will be joined by some other friends and be played in multi-player mode "I'll have some company!". If there will be only one user taking the quiz, there will be some buttons that need to be pressed to get to actually answering the questions, all the instructions are displayed on the screen, and then the after the quiz starts each question that is contained in the chosen quiz will be displayed one by one, after submitting the answer, a frame will show up letting the user know if he got the correct answer or not. If the answer was not correct, the correct answer is also displayed. After all the questions have been submitted, a score will be displayed and then the user can choose what he wants to do next: play another quiz or exit the application by pressing specific buttons.

If the user chooses to play with some other friends it works like this: the main player( the one that started the game, the one that logged in first) needs to add the rest of the players, this happens by pressing the "add players" button from the bottom of the screen. Now the main user needs to introduce the names and the ages of the ones that will join him. After

inputting the name and age of the first player that joins, the button “submit+add more” needs to be pressed. Only after this button is pressed the data of the player is being processed. After all the players are added, the “done” button should be pressed, and then the quiz will start for the first player. If the name and age are entered and the “submit+add more” is not pressed, and instead the “done ” button is, then the player whose name and age were on the screen previously will not be retained and the player will not be able to play. When the game starts, the name of the player whose turn is will be displayed on the screen along with the quiz details. In order for the certain player to take the quiz, the device on which the application is opened should be in the possession of the player whose name is showed on the screen and the button “Take quiz” should be pressed to take the quiz. The quiz is displayed exactly as for the single player. After the player takes the quiz the button “ next player’s turn” should be pressed and the same will happen for each player until there are no more players and the leaderboard is displayed. If one player did not press “take quiz” button but pressed “ next player’s turn”, his score will be considered zero, and there is no other way of taking the quiz in the current session anymore. After the leaderboard is displayed, the user can take another quiz or exit the application.

## 2. Class Discovery

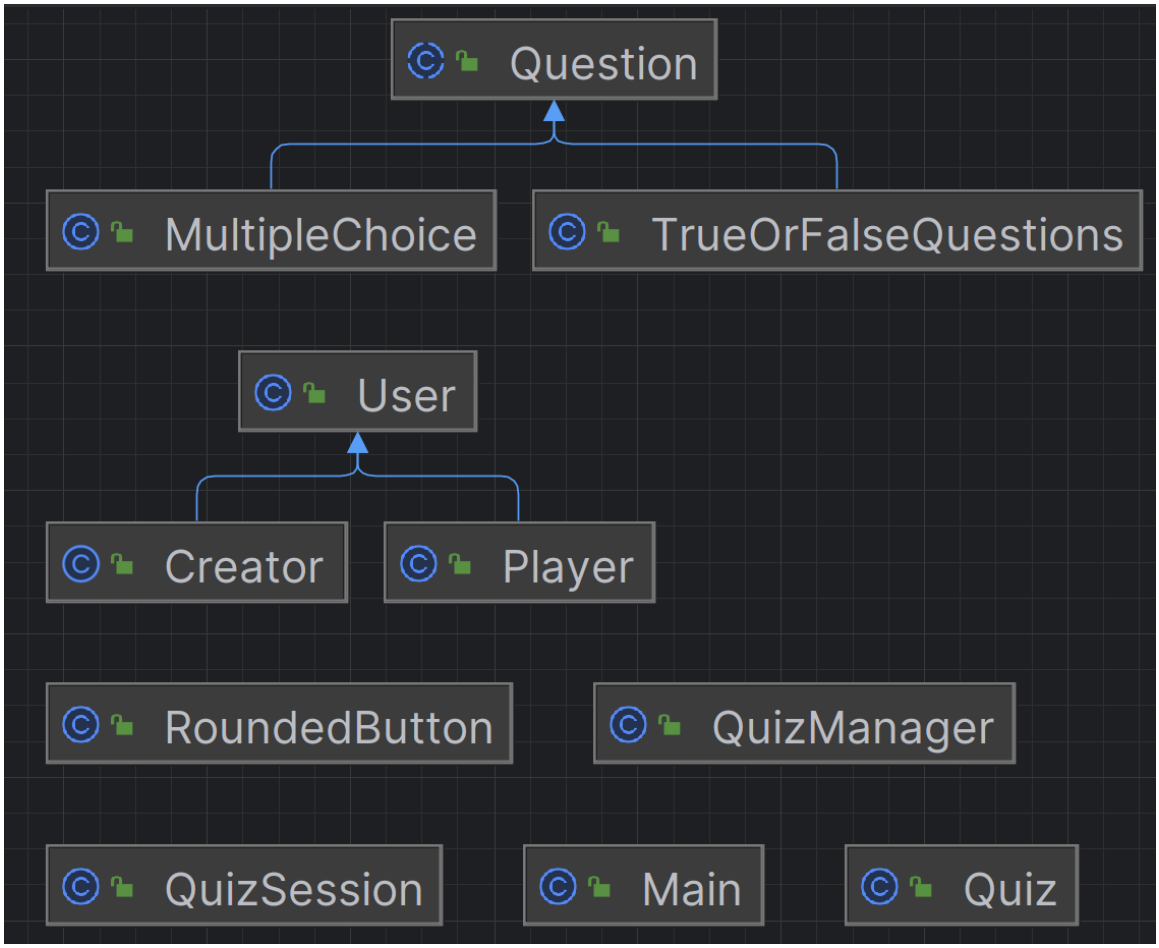
| **User** | |
|---|---|
| Manage user login information.<br>Direct users to either create or play a quiz. | QuizManager<br>Quiz<br>Player<br>Creator |
| **Player** | |
| Choose the quiz to play.<br>Manage player score.<br>Create a quiz session. | Quiz<br>QuizSession<br>QuizManager |
| **Creator** | |
| Create new quizzes.<br>Adds questions to a quiz. | Quiz<br>MultipleChoice<br>TrueOrFalseQuestions |
| **Quiz** | |
| Store quiz details (name, topic, difficulty).<br>Helps set the quiz details.<br>Hold a list of the questions that are in the quiz. | Question<br>MultipleChoice<br>TrueOrFalseQuestions<br>Creator<br>Player |

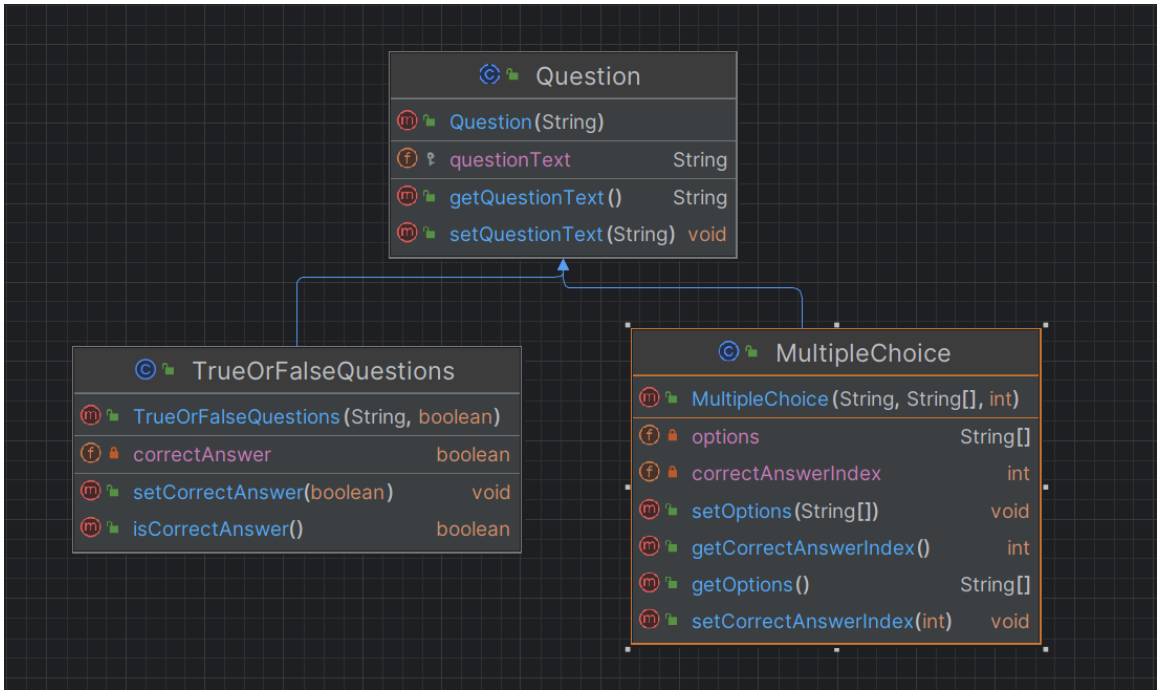
| **Question** | |
|---|---|
| Provide a template for different types of questions, keeps the question text. | MultipleChoice<br>TrueOrFalseQuestions<br>Quiz<br>QuizSession |
| **MultipleChoice** | |
| Store multiple-choice question information | Creator<br>Quiz<br>QuizSession |

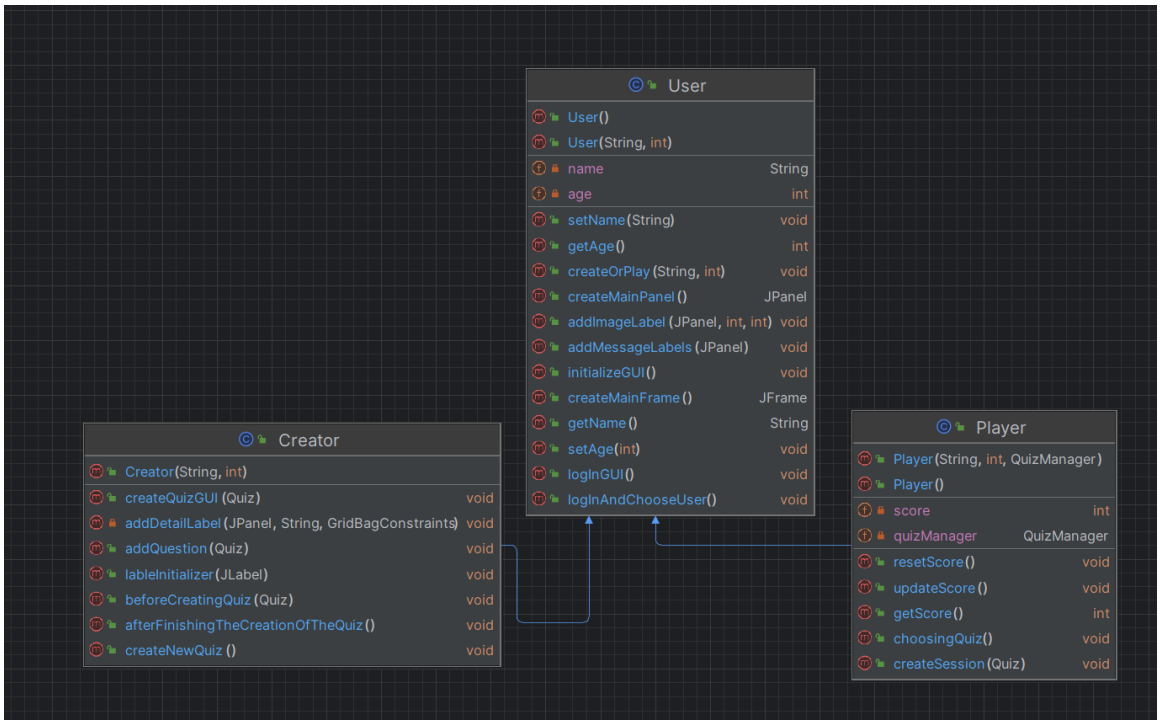
| **TrueOrFalseQuestions** | |
|---|---|
| Store true/false question information | Creator<br>Quiz<br>QuizSession |
| **QuizSession** | |
| Manage a quiz-taking session.<br>Handle multiplayer quiz flow. | Quiz<br>Player<br>Question<br>MultipleChoice<br>TrueOrFalseQuestions |
| **RoundedButton** | |
| Provide a custom UI for buttons. | User Interface Components |
| **QuizManager** | |
| Manage the collection of quizzes.<br>Load and save quizzes to file. | Quiz<br>Player |

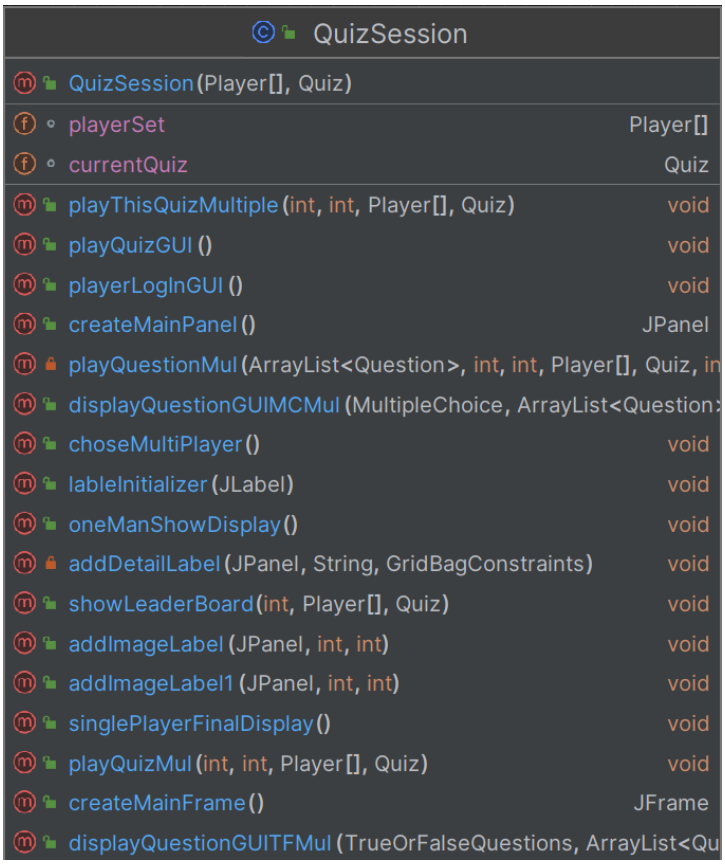
| **Main** | |
|---|---|
| Sets in motion the entire game | Creator |

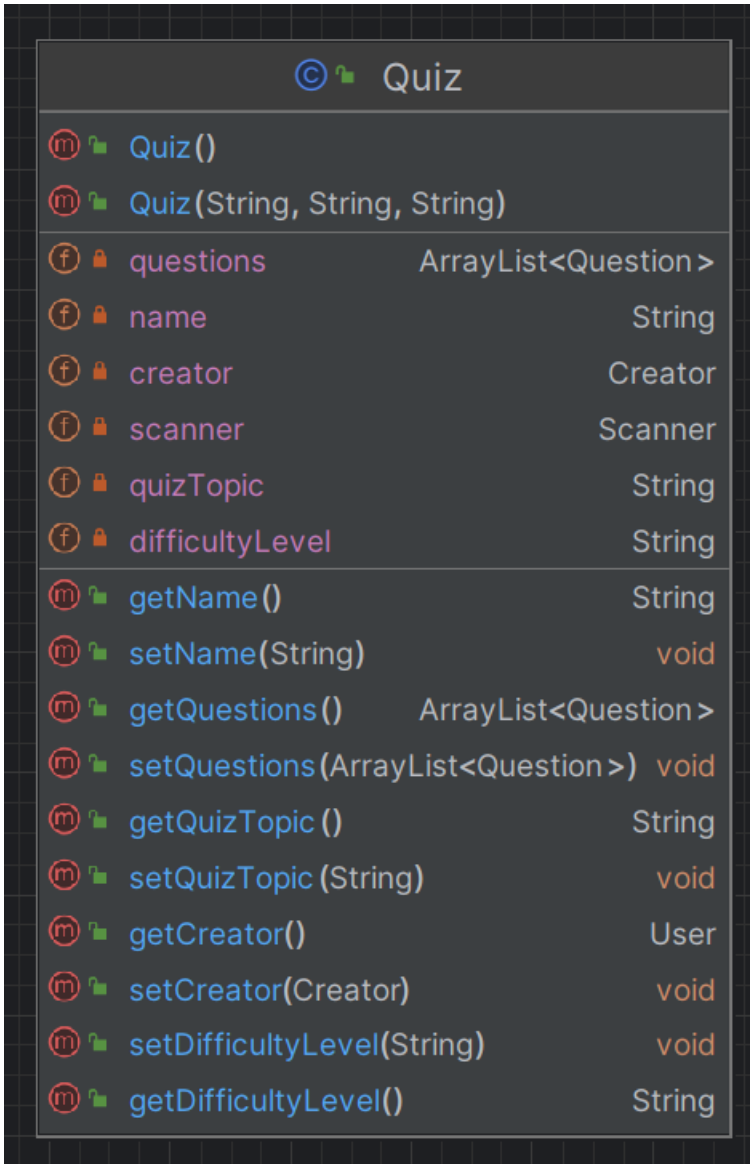
## 3. Class Diagram

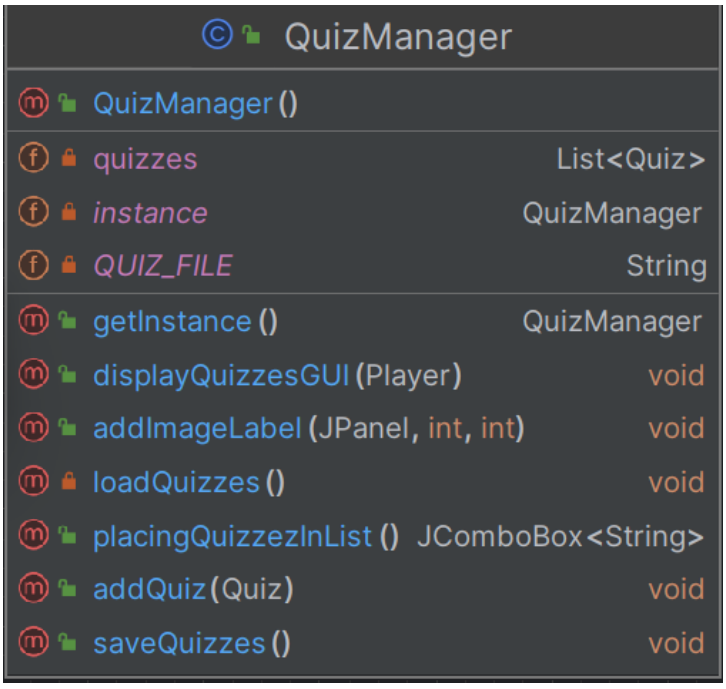


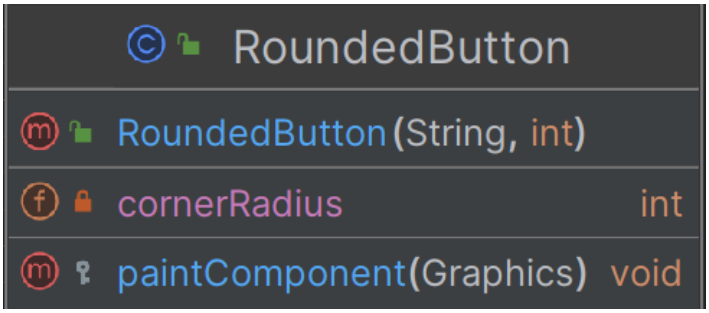

Question
Question(String)
questionText String
getQuestionText() String
setQuestionText(String) void
TrueOrFalseQuestions
TrueOrFalseQuestions(String, boolean)
correctAnswer boolean
setCorrectAnswer(boolean) void
isCorrectAnswer() boolean
MultipleChoice
MultipleChoice(String, String[], int)
options String[]
correctAnswerIndex int
setOptions(String[]) void
getCorrectAnswerIndex() int
getOptions() String[]
setCorrectAnswerIndex(int) void

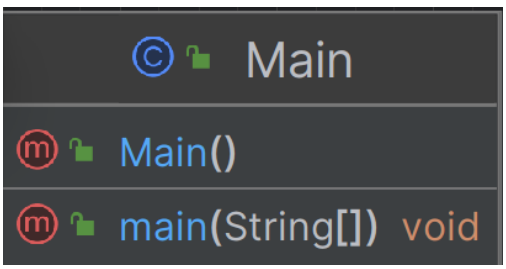

User
User()
User(String, int)
name String
age int
setName(String) void
getAge() int
createOrPlay(String, int) void
createMainPanel() JPanel
addImageLabel(JPanel, int, int) void
addMessageLabels(JPanel) void
initializeGUI() void
createMainFrame() JFrame
getName() String
setAge(int) void
logInGUI() void
logInAndChooseUser() void
Creator
Creator(String, int)
createQuizGUI(Quiz) void
addDetailLabel(JPanel, String, GridBagConstraints) void
addQuestion(Quiz) void
lableInitializer(JLabel) void
beforeCreatingQuiz(Quiz) void
afterFinishingTheCreationOfTheQuiz() void
createNewQuiz() void
Player
Player(String, int, QuizManager)
Player()
score int
quizManager QuizManager
resetScore() void
updateScore() void
getScore() int
choosingQuiz() void
createSession(Quiz) void

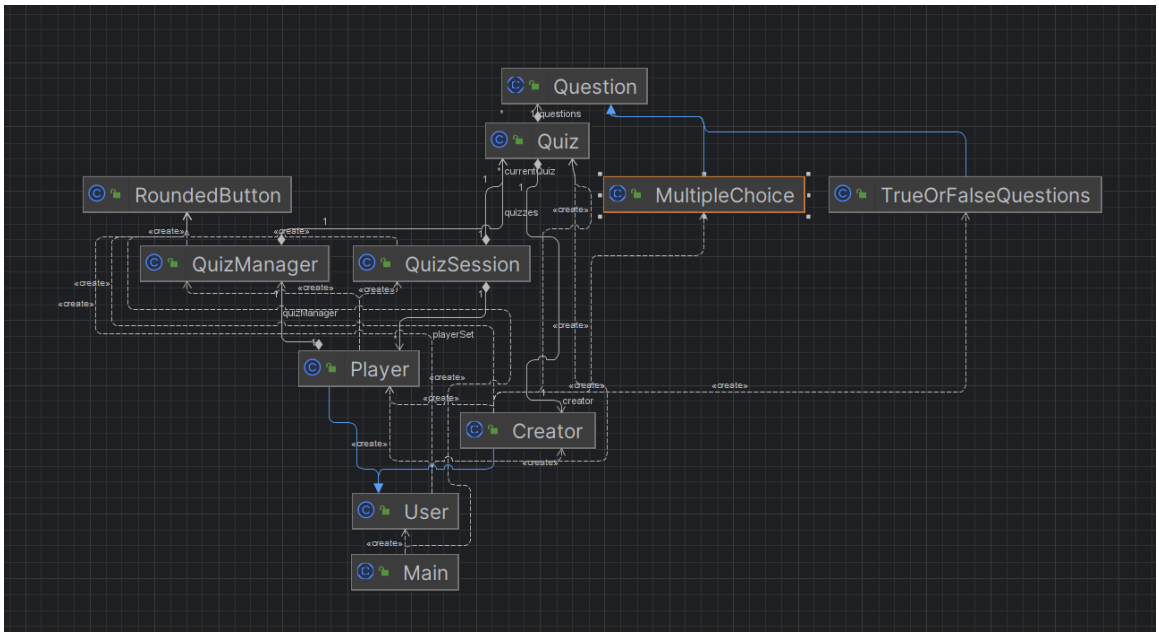

QuizSession
QuizSession(Player[], Quiz)
playerSet Player[]
currentQuiz Quiz
playThisQuizMultiple(int, int, Player[], Quiz) void
playQuizGUI() void
playerLogInGUI() void
createMainPanel() JPanel
playQuestionMul(ArrayList<Question>, int, int, Player[], Quiz, in
displayQuestionGUIMCMul(MultipleChoice, ArrayList<Question>
choseMultiPlayer() void
lableInitializer(JLabel) void
oneManShowDisplay() void
addDetailLabel(JPanel, String, GridBagConstraints) void
showLeaderBoard(int, Player[], Quiz) void
addImageLabel(JPanel, int, int) void
addImageLabel1(JPanel, int, int) void
singlePlayerFinalDisplay() void
playQuizMul(int, int, Player[], Quiz) void
createMainFrame() JFrame
displayQuestionGUITFMul(TrueOrFalseQuestions, ArrayList<Qu


Quiz
Quiz()
Quiz(String, String, String)
questions ArrayList<Question>
name String
creator Creator
scanner Scanner
quizTopic String
difficultyLevel String
getName() String
setName(String) void
getQuestions() ArrayList<Question>
setQuestions(ArrayList<Question>) void
getQuizTopic() String
setQuizTopic(String) void
getCreator() User
setCreator(Creator) void
setDifficultyLevel(String) void
getDifficultyLevel() String


QuizManager
QuizManager()
quizzes List<Quiz>
instance QuizManager
QUIZ_FILE String
getInstance() QuizManager
displayQuizzesGUI(Player) void
addImageLabel(JPanel, int, int) void
loadQuizzes() void
placingQuizzezInList() JComboBox<String>
addQuiz(Quiz) void
saveQuizzes() void


RoundedButton
RoundedButton(String, int)
cornerRadius int
paintComponent(Graphics) void


Main
Main()
main(String[]) void


Question
questions
Quiz
currentQuiz
quizzes
RoundedButton
MultipleChoice
TrueOrFalseQuestions
QuizManager
QuizSession
quizManager
playerSet
Player
creator
Creator
User
Main
«create»