### **Documentation for the "Quiz Application" Project**

#### **Project Description**

This project is a quiz application developed using the **Qt Framework**. It allows users to register, answer a series of questions within a time limit, and receive their final score based on the number of correct answers.

### **Main Components**

#### **1. question.h**

* Contains the Question class, which represents the structure of a question.
* Class fields:
  + QString questionText: the text of the question.
  + QStringList answerOptions: a list of answer options.
  + int correctAnswerIndex: the index of the correct answer.
* Constructor:
  + Initializes the fields of the class.

#### **2. mainwindow.h**

* Header file for the main application window.
* Key functions:
  + onStartQuiz(): starts the game and checks the user's name.
  + handleAnswer(int index): processes the user's selected answer.
  + onTimeout(): updates the timer and moves to the next question when time runs out.
  + setupQuestions(): loads the list of questions.
  + displayQuestion(): displays the current question and answer options.
  + finishQuiz(): ends the quiz and shows the results.

#### **3. mainwindow.cpp**

* Contains the logic of the quiz.
* Key stages:
  + **Application Initialization**:
    - Sets up the UI (setupUi).
    - Connects signals and slots (e.g., answer buttons and the timer).
  + **Loading Questions**:
    - Adds Question objects to the questions list.
  + **Starting the Game**:
    - Resets the score, timer, and question index.
    - Displays the first question.
  + **Handling Answers**:
    - Compares the selected answer's index with the correct one.
    - Increments the score for correct answers.
    - Moves to the next question.
  + **Timer**:
    - Updates the remaining time every second.
    - Advances to the next question if time runs out.
  + **Ending the Quiz**:
    - Stops the timer.
    - Displays the final result in a QMessageBox.

#### **4. main.cpp**

* Entry point of the application.
* Creates a QApplication object and displays the main window.

#### **5. mainwindow.ui**

* The graphical user interface, designed using Qt Designer.
* Key elements:
  + QLineEdit: a text field for the user's name (nameLineEdit).
  + QPushButton: a button to start the game (startButton).
  + QLabel: displays the question text (questionLabel).
  + QLabel: displays the remaining time (timeLabel).
  + QLabel: displays the score (scoreLabel).
  + Four QPushButton elements for the answer options (answerButton1, answerButton2, answerButton3, answerButton4).

### **Program Workflow**

1. The user enters their name and clicks the **"Start"** button.
2. If a name is provided:
   1. The timer starts.
   2. The first question and its answer options are displayed.
3. The user selects an answer:
   1. If the answer is correct, the score increases.
   2. The application moves to the next question.
4. The timer counts down:
   1. If time runs out, the application automatically moves to the next question.
5. After all questions are answered:
   1. The timer stops.
   2. The user sees their final result.

### **How to Use the Project**

#### **Build and Run**

1. Ensure that Qt is installed (e.g., Qt 5.15 or later).
2. Open the project in Qt Creator.
3. Build the project (Ctrl+B) and run it (Ctrl+R).

#### **Adding Questions**

* Questions are added in the setupQuestions() method in the mainwindow.cpp file.
* Format: questions.append(Question("Question text", {"Answer1", "Answer2", "Answer3", "Answer4"}, correct\_answer\_index));

#### **Interface**

* The required fields and buttons are already configured in the mainwindow.ui file.
* If necessary, you can modify them using **Qt Designer**.

### **Potential Improvements**

1. **Database Support**:
   1. Store questions and results in SQLite for dynamic loading.
2. **Difficulty Levels**:
   1. Divide questions into levels with varying time limits and scores.
3. **User Statistics**:
   1. Display a history of user results.
4. **Localization**:
   1. Add support for multiple languages using .ts files.

This project provides a foundation for further development and is ideal for beginner developers learning Qt.





















