Quiz Application

Tuesday, September 9, 2025 1:17 PM

Documentation: Java Quiz Application

◇ Question Class

```java
// question class
class Question{  10 usages  new *

    // variables
    int id ;  6 usages
    String q ;  6 usages
    String opt1 ;  6 usages
    String opt2 ;  6 usages
    String opt3 ;  6 usages
    String opt4 ;  6 usages
    String ans ;  6 usages

    // methods
    @Override  new *
    public String toString() {
        return id + "> " + q + "\na. " + opt1 + "\nb. " + opt2 + "\nc. " + opt3 + "\nd. " + opt4 ;
    }
}
```

Purpose

- Represents a single **quiz question**.
- Stores the question text, four options (a, b, c, d), and the correct answer (ans).

Key Features

- **Fields**:
  - id → unique identifier for the question.
  - q → the question text.
  - opt1–opt4 → multiple-choice options.
  - ans → stores the correct answer as "a", "b", "c", or "d".
- **toString() method**:
  - Overrides the default Object.toString().
  - Formats the question neatly when printed.

◇ Quiz Service Class

```java
class QuizService{  2 usages  new *

    // variables
    final private Question[] questions ;  2 usages
    private int score = 0 ;  2 usages
    Scanner input = new Scanner(System.in) ;  1 usage

    // constructor
    public QuizService(Question[] questions) {  1 usage  new *
        this.questions = questions ;
    }

    // methods
    public void playQuiz(){  1 usage  new *
        for(Question q : questions){
            System.out.println(q);



            String choice ;
            System.out.println("(a/b/c/d): ");
            choice = input.nextLine() ;
            if (choice.equals(q.ans)){
                System.out.println("correct!!");
                score++ ;
            }
            else {
                System.out.println("wrong..");
            }
        }
    }
    public void showScore(){  1 usage  new *
        System.out.println("your score : " + score);
    }
}
```

Purpose

- Encapsulates the **quiz logic**: asking questions, evaluating answers, and calculating the score.

Key Features

- **Fields**:
  - questions → array of Question objects.
  - score → stores the player's score.
  - input → scanner for user input.
- **Constructor**:
  - Initializes the quiz with a set of questions.
- **playQuiz()**:
  - Iterates through each Question.
  - Displays the question using toString().
  - Accepts user input (a/b/c/d).
  - Checks correctness using .equals().
  - Updates the score.
- **showScore()**:
  - Prints the total score after the quiz ends.

Purpose

- Acts as the **driver class** (contains the main() method).
- Creates an array of 5 Question objects with Java-related MCQs.
- Passes the questions to QuizService.
- Starts the quiz and shows the final result.

◇ Program Flow

1. The main() method creates 5 Java questions.
2. A QuizService object is initialized with these questions.
3. The playQuiz() method:
    - Prints each question.
    - Prompts the user for input (a/b/c/d).
    - Validates the answer.
    - Updates the score.
4. After all questions, showScore() displays the total marks.

◇ Sample Run (Console Output)

```
d. execute()
(a/b/c/d):
b
correct!!
3> Which of these is not a Java access modifier?
a. public
b. protected
c. private
d. friendly
(a/b/c/d):
d
correct!!
4> Which of these is used to handle exceptions in Java?
a. goto
b. try-catch
c. finalize
d. throwable
(a/b/c/d):
b
correct!!
5> Which collection class allows duplicate elements?
a. HashSet
b. TreeSet
c. ArrayList
d. None of the above
(a/b/c/d):
c
correct!!
your score : 4
```

☑ This structure demonstrates:

- **OOP concepts** (encapsulation, object arrays, method overriding).
- **User input handling** with Scanner.

- **MCQ quiz system logic**.