Dokumentation für das Fachinformatiker Quiz-Skript

1. Einleitung

Dieses Skript implementiert eine einfache GUI-Anwendung für ein Quiz, das mit Python und Tkinter erstellt wurde. Das Quiz zieht seine Fragen und Antwortmöglichkeiten aus einer JSON-Datei und präsentiert sie in einer grafischen Benutzeroberfläche. Der Benutzer kann durch das Quiz navigieren, Antworten auswählen und erhält am Ende eine Auswertung seiner Leistung.

2. Abhängigkeiten

- **Python 3.x**
- **Tkinter**: Eine Standard-GUI-Bibliothek für Python.
- **JSON**: Für das Laden von Fragen und Antworten aus einer externen Datei.
- **Random**: Zum Mischen von Fragen und Antworten.

3. Aufbau der JSON-Datei

Die JSON-Datei enthält die Quiz-Daten und hat folgendes Format:

json

Code kopieren

```
[
  {
    "frage": "Was ist Python?",
    "optionen": ["Programmiersprache", "Schlange", "Musikinstrument", "Tanzstil"],
    "antwort": "Programmiersprache"
  },
  ...
]
```

Jedes Element in der Liste repräsentiert eine Frage, die drei Eigenschaften hat:

- **frage**: Der Fragetext.
- **optionen**: Eine Liste von möglichen Antworten.
- **antwort**: Die korrekte Antwort.

4. Funktionsübersicht

4.1 lade_quiz_daten(datei)

Diese Funktion lädt die Quiz-Daten aus der angegebenen JSON-Datei.

- **Parameter**:
    - datei (str): Der Pfad zur JSON-Datei.
- **Rückgabewert**:
    - Eine Liste von Fragen und Antworten, wie in der JSON-Datei definiert.

4.2 QuizApp.__init__(self, root)

Der Konstruktor für die Klasse QuizApp, die die Hauptanwendung darstellt.

- **Parameter**:
    - root (tk.Tk): Das Hauptfenster der Tkinter-Anwendung.
- **Beschreibung**:
    - Initialisiert das Fenster, erstellt alle GUI-Elemente und startet das Quiz.

4.3 QuizApp.shuffle_questions(self, questions, limit=None)

Mischt die Reihenfolge der Fragen und begrenzt die Anzahl der angezeigten Fragen.

- **Parameter**:
    - questions (list): Die Liste der Fragen, die gemischt werden sollen.
    - limit (int, optional): Die maximale Anzahl der Fragen, die im Quiz verwendet werden sollen.
- **Rückgabewert**:
    - Eine gemischte Liste von Fragen, möglicherweise auf eine bestimmte Anzahl beschränkt.

4.4 QuizApp.shuffle_options(self, question_data)

Mischt die Reihenfolge der Antwortmöglichkeiten für eine gegebene Frage.

- **Parameter**:
    - question_data (dict): Ein Dictionary, das eine Frage und ihre Antwortmöglichkeiten enthält.
- **Rückgabewert**:
    - Eine Liste von gemischten Antwortmöglichkeiten und den Index der richtigen Antwort.

4.5 QuizApp.show_question(self)

Zeigt die aktuelle Frage und die zugehörigen Antwortmöglichkeiten an.

- **Beschreibung**:
    - Aktualisiert das GUI-Element, um die nächste Frage anzuzeigen, wenn eine vorhanden ist. Andernfalls wird die Auswertung angezeigt.

4.6 QuizApp.check_answer(self)

Überprüft die vom Benutzer gewählte Antwort und gibt Feedback.

- **Beschreibung**:
    - Vergleicht die gewählte Antwort mit der richtigen Antwort und aktualisiert das Feedback-Label entsprechend. Wechselt nach 2 Sekunden zur nächsten Frage.

4.7 QuizApp.show_evaluation(self)

Zeigt die Auswertung des Quiz an, nachdem alle Fragen beantwortet wurden.

- **Beschreibung**:
    - Entfernt die bisherigen GUI-Elemente und zeigt eine Zusammenfassung der Ergebnisse an. Gibt dem Benutzer die Möglichkeit, das Quiz zu schließen.

5. GUI-Elemente

- **root**: Das Hauptfenster der Anwendung.
- **frame**: Ein Rahmen, der alle anderen GUI-Elemente enthält.
- **frage_label**: Zeigt die aktuelle Frage an.
- **option_buttons**: Radiobuttons für die Antwortmöglichkeiten.
- **check_button**: Eine Schaltfläche zum Überprüfen der gewählten Antwort.
- **feedback_label**: Ein Label, das Feedback zur gewählten Antwort gibt.
- **auswertung_label**: Zeigt die Auswertung nach Abschluss des Quiz an.
- **close_button**: Eine Schaltfläche zum Schließen der Anwendung.

6. Ablauf des Programms

1. **Start**: Das Skript lädt die Fragen aus der JSON-Datei und mischt sie.
2. **Frageanzeige**: Die erste Frage wird angezeigt, und der Benutzer wählt eine Antwort aus.
3. **Antwortüberprüfung**: Der Benutzer klickt auf „Überprüfen“, um die Antwort zu bestätigen.
4. **Feedback**: Das Programm gibt Feedback zur Richtigkeit der Antwort.
5. **Nächste Frage**: Nach einer kurzen Verzögerung wird die nächste Frage angezeigt.

6. **Auswertung**: Nachdem alle Fragen beantwortet wurden, wird eine Auswertung angezeigt.
7. **Beenden**: Der Benutzer kann das Quiz über die „Schließen“-Schaltfläche beenden.

7. Anpassungsmöglichkeiten

- **Fragenanzahl**: Die Anzahl der gestellten Fragen kann über den limit-Parameter in shuffle_questions angepasst werden.
- **Fragen und Antworten**: Diese können durch Bearbeiten der JSON-Datei verändert werden.
- **Layout und Stil**: Farben, Schriftarten und Größen der GUI-Elemente können durch Anpassung der Tkinter-Widgets verändert werden.

8. Fazit

Dieses Skript bietet eine einfache Möglichkeit, ein Quiz mit einer grafischen Benutzeroberfläche in Python zu erstellen. Es ist leicht anpassbar und erweiterbar, sodass es für verschiedene Themen und Zwecke verwendet werden kann.