EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Mathematisch-Naturwissenschaftliche Fakultät
Fachbereich Informatik
Informationsdienste – Prof. Dr. Thomas Walter

Einführung Internet-Technologien

Sommersemester 2019

Übungsblatt 10

Theorie

Die Lösungen zu den Theorieaufgaben werden in den Tutorien besprochen und im Moodle veröffentlicht. Sie brauchen *keine* Lösungen zu den Theorieaufgaben abgeben!

1. Wie lange darf die Bearbeitung einer Server-seitig ausgeführten Implementierung maximal dauern, bevor der Nutzer die Seite verlässt? Wie geht man bei länger dauernden Anfragen vor?
2. Erklären Sie die Funktionsweise und Anwendungsfälle von AJAX.
3. Worin unterscheidet sich AJAX im Vergleich zu konventionellen Webanwendungen?
4. Welche Vor- und Nachteile hat die Anwendung von AJAX?
5. Erklären Sie, wie Sie in JavaScript AJAX verwenden können.
6. Diskutieren Sie: AJAX ist nicht barrierefrei.
7. Wozu dient die PHP-Funktion `header()`?
8. Wie können Sie mit PHP Daten json-codiert ausgeben?

Praktische Übungen

Abgabe bis Montag, 15.07.2019, 14:00 Uhr. Erstellen Sie in Ihrem *public_html/*-Ordner auf dem Übungsserver einen Ordner *uebung10/*. Speichern Sie dort die in dieser Übung erstellten Dateien.

Geben Sie unter Checkpoint 10 im Moodle einen *anklickbaren* Link auf das von Ihnen auf dem Übungsserver erstellen Dateien *api.php* und *quiz.html* ab. Laden Sie zudem Ihren gesamten Quellcode als zip- oder tar.gz-Archiv im Moodle hoch!

In dieser Übung erstellen Sie ein Quiz-Spiel in JavaScript. Das Spiel ruft die Quizfragen per AJAX von Ihrer CRUD-Anwendung aus Übung 9 ab. Die hierfür notwendige API erstellen Sie ebenfalls in dieser Übung.

Falls Sie Übung 9 nicht bearbeitet haben, führen Sie die in Übung 9 unter „Vorbereitung“ beschriebenen Schritte aus. Den Rest von Übung 9 können Sie ignorieren.

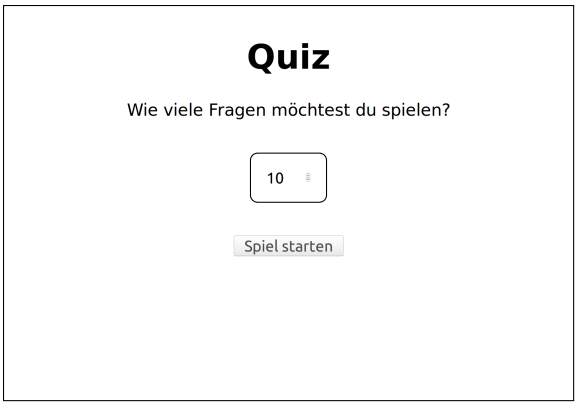

Abbildung 1: Beginn einer Quizrunde

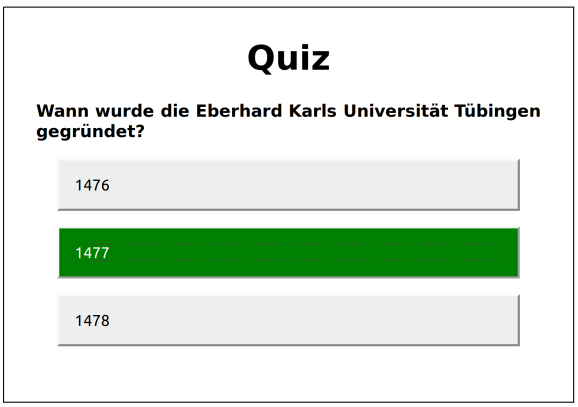

Abbildung 2: Frage richtig beantwortet

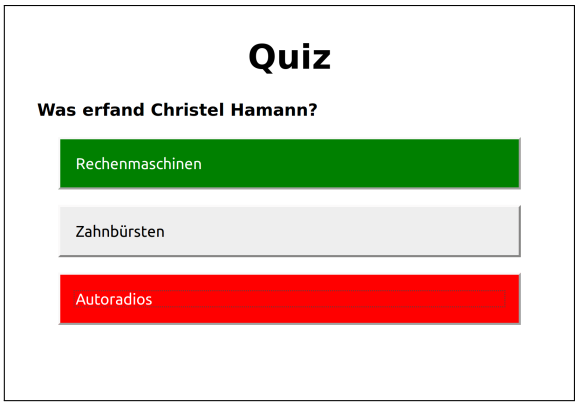

Abbildung 3: Frage falsch beantwortet

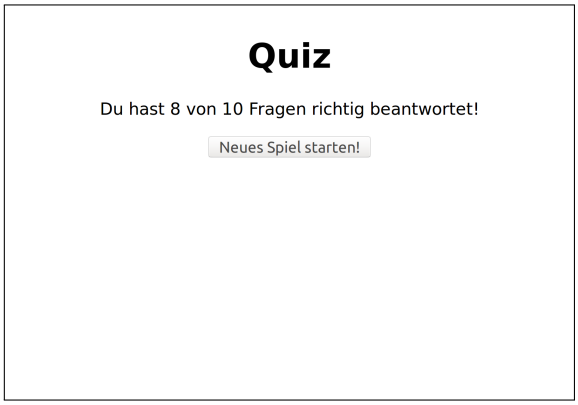

Abbildung 4: Ende einer Quizrunde

Aufgabe 1: *API erstellen (1 Punkt)*

Erstellen Sie ein PHP-Skript *api.php*, mit dem Quizfragen aus der Tabelle *questions* JSON-codiert abgerufen werden können. Das Skript soll folgende Anforderungen erfüllen:

- Das Skript akzeptiert einen optionalen HTTP-GET-Parameter „n“. Falls der Parameter nicht übermittelt wurde, wird n = 10 gesetzt.
- Per `SELECT`-Query werden n Quizfragen zufällig aus der Tabelle *questions* ausgewählt. Verwenden Sie *Prepared Statements*, um SQL-Injections zu verhindern.

  Tipp: Die Query, um 10 zufällige Fragen abzurufen lautet:

  ```
  SELECT * FROM questions ORDER BY RAND() LIMIT 10;
  ```

- Der Content-type wird auf application/json gesetzt. Verwenden Sie hierfür folgende Funktion: `header('Content-Type: application/json');` Vor dem Setzen des Headers darf keinerlei Ausgabe stattfinden!
- Die n zufällig gewählten Quizfragen werden als JSON-codiertes Array ausgegeben, wobei jede Frage in folgende Form gebracht wird:

  ```
  {
     "id":<id der Frage (Integer)>,
     "question":"<Frage>",
     "answers":"<Array der Antwortmöglichkeiten>",
     "solution":<Index der korrekten Antwort aus dem answers-Array (Integer)>
  }
  ```

  Beispiel:

  ```
  {
     "id":2,
     "question":"Wann wurde die Eberhard Karls Universität Tübingen gegründet?",
     "answers":[
        "1476",
        "1477",
        "1478"
     ],
     "solution":2
  }
  ```

  Tipp: Erstellen Sie ein PHP-Array mit der passenden Struktur und wenden Sie darauf die PHP-Funktion `json_encode($array)` an.

Aufgabe 2: *Quiz erstellen (2 Punkte)*

Erstellen Sie eine HTML-Datei *quiz.html*. Implementieren Sie darin in JavaScript folgendes Quiz-Spiel:

- Zu Beginn einer Quizrunde wird der Spieler gefragt, wie viele Quizfragen er spielen möchte. Hierfür wird ein Number-Input-Feld sowie ein Button „Spiel starten“ angezeigt. In das Feld kann der Spieler Zahlen zwischen 1 und 20 eingeben. Der voreingestellte Default-Wert soll 10 betragen. (Siehe Abb. 1)
- Wenn der Spieler auf „Spiel starten“ klickt, wird die entsprechende Anzahl Quizfragen per `XMLHttpRequest` von Ihrem *api.php*-Skript abgerufen.
- Danach wird die erste Quizfrage anzeigt: Hierfür wird die Frage sowie für jede Antwortmöglichkeit ein Button angezeigt. Wenn der Spieler auf einen Button klickt, wird überprüft, ob die Antwort richtig war. Falls die Antwort richtig war, wird der Button grün hinterlegt. Falls die Antwort falsch war, wird der Button rot hinterlegt und der Button der richtigen Antwort wird grün hinterlegt. (Siehe Abb. 2 und 3)
- Die Buttons bleiben nach dem Antworten für zwei Sekunden markiert, danach erscheint die nächste Frage.
- Wenn der Spieler alle Fragen beantwortet hat, wird angezeigt, wie viele Fragen richtig beantwortet wurden. Zusätzlich wird ein Button „Neues Spiel starten!“ angezeigt, mit dem eine neue Quizrunde gestartet werden kann. (Siehe Abb. 4)

Aufgabe 3: *Soundeffekte hinzufügen (optional 1 Bonuspunkt)*

Fügen Sie dem Spiel Soundeffekte hinzu: Wenn der Spieler eine Frage richtig oder falsch beantwortet hat, soll jeweils ein passender Soundeffekt abgespielt werden. Wählen Sie hierfür selbst passende Audio-Dateien aus.

Geben Sie bei der Textabgabe im Moodle an, wenn Sie die Bonusaufgabe bearbeitet haben!