УДК 004

Создание обучающего веб-квеста с помощью JavaScript

Аристова Дарья Андреевна
Приамурский Государственный университет имени Шолом-Алейхема
Студент

Аннотация
Данная статья посвящена созданию обучающего веб-квеста с помощью JavaScript. Так как вопрос вовлечённости учащихся в процесс обучения остаётся актуальным в настоящее время, особенно это касается детей. Человеку легче воспринимать информацию в интерактивной форме. Традиционные методы проверки знания могут оказаться недостаточными для проверки знаний учащихся на фоне появляющихся средств организации учебного процесса.
Ключевые слова: веб-квест, JavaScript, HTML.

Creating a training web quest using JavaScript

Aristova Darya Andreevna
Sholom-Aleichem Priamursky State University
Student

Abstract
This article is devoted to creating a training web quest using JavaScript. Since the issue of student involvement in the learning process remains relevant at the present time, this is especially true for children. It is easier for a person to perceive information in an interactive form. Traditional methods of testing knowledge may not be sufficient to test students' knowledge against the background of emerging means of organizing the educational process.
Keywords: web-quest, JavaScript, HTML

Перед началом разработки веб-квеста, необходимо выбрать инструменты для реализации такой задачи.

Самым популярным решением являются Google Формы. Данный сервис позволяет создавать формы с неограниченным количеством вопросов.

Также подобное программное решение представлено Яндексом. Яндекс предоставляет схожий функционал и также позволяет создавать тесты, вопросы в которых предполагают различные формы ввода ответа.

Google Формы и конструктор от Яндекса способны создать опыт проверки знаний обучающихся схожий с тем, если бы преподаватель проводил тестирование традиционными методами. Но в данном случае необходима полная автоматизация. Поэтому будем создавать веб-квест с помощью JavaScript.

Создание веб-квеста можно разделить на несколько этапов:

1. Необходимо определить количество заданий и их содержание.

2. Если в задании есть вопросы, на которые требуются ответы со стороны пользователя, то нужно определить – какого типа должен быть формат выбора ответа (выбор одного ответа, выбор нескольких ответов, ввод ответа с клавиатуры).

3. Добавить в HTML страницы разметку для вопросов и других элементов веб-квеста.

4. Реализовать функции веб-квеста, применяя язык программирования JavaScript.

Перед началом разработки веб-квеста, необходимо определить ряд заданий и тестовых вопросов к ним.

В начале необходимо определить структуру проекта, то есть из каких файлов он будет состоять, как они взаимодействуют между собой и другие важные моменты на начальном этапе создания любого проекта.

Было определено 4 файла в проекте, которые нужно создать для обеспечения работоспособности проекта:

− index.html – главная страница, с которой начинается веб-квест;

− quest.html – страница с квестом, на которой реализован игровой процесс;

− task.html – страница с заданием квеста. Таких страниц может быть несколько;

− result.html – страница, на которой выводится результат прохождения квеста;

− game.js – JavaScript файл, в котором описана логика работы веб-квеста.

На главной странице выводится приветствие и кнопка для старта квеста (см. рисунок 1). Уже на этой странице задействован скрипт game.js, но только одна его функция – “movePage()”. Функция отвечает за перемещение на страницу квеста.

```
<body>
     <h1>Квест "Увлекательная информатика"</h1>
     <p>Добро пожаловать в веб-квест по информатике! Чтобы начать, нажмите кнопку ниже!</p>
     <button onclick="movePage()">Начать квест!</button>
     <script src="game.js"></script>
  </body>
```

Функция “movePage()” определена в game.js и состоит из одной строки, которая перенаправляет пользователя на другую страницу. Переход можно создать с помощью функции “href”, которая является частью модуля “location”. Функция принимает только одно значение – страницу, на которую необходимо перейти:

```
location.href = “page.html”
```

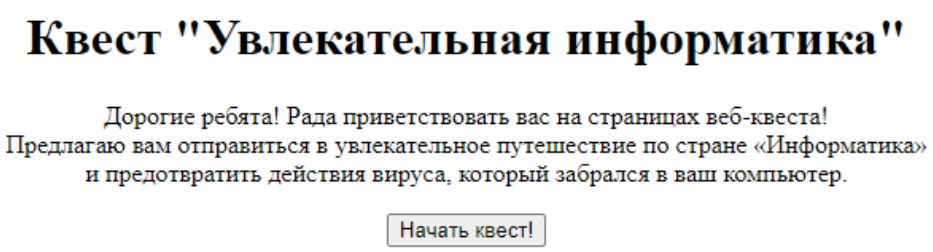

Рисунок 1 – Содержание главной страницы

На странице quest.html представлен квест. На странице представлено 5 кнопок, каждая из которых ведёт на страницу с заданием (то есть на task.html). Если задание пользователем было выполнено, то рядом с кнопкой отображается текст “(Выполнено)”. Если выполнены все задания, то пользователя переводит на страницу с результатом. Выглядит HTML код страницы следующим образом:

```
<body onload="main()">
     <p>Вам необходимо выполнить 5 заданий, чтобы завершить
квест!</p>
   <button           onclick="location.href='task1.html'">Задание
1</button><p id="win1"></p>
   <button           onclick="location.href='task2.html'">Задание
2</button><p id="win2"></p>
   <button           onclick="location.href='task3.html'">Задание
3</button><p id="win3"></p>
   <button           onclick="location.href='task4.html'">Задание
4</button><p id="win4"></p>
   <button           onclick="location.href='task5.html'">Задание
5</button><p id="win5"></p>
          <script src="game.js"></script>
</body>
```

Первое задание состоит из 5-и вопросов (см. рисунок 2). После вопросов есть кнопка, которая выполняет JavaScript код, проверяющий правильность написанных ответов. Код получает доступ к данным, которые написал пользователь через функцию “getElementById” и проверяет значение с правильными ответом. Если все ответы правильные, то решение засчитывается и сохраняется с помощью функции “localStorage.setItem”:

```
function task1() {
let score = 0;
var question1 = document.getElementById('question1').value;
if (question1 == "ОЗУ") score++;
var question2 = document.getElementById('question2').value;
if (question2 == "видеокарта") score++;
var question3 = document.getElementById('question3').value;
if (question3 == "кулер") score++;
var question4 = document.getElementById('question4').value;
if (question4 == "SSD") score++;
```

```
var question5 = document.getElementById('question5').value;
if (question5 == "процессор") score++;
if (score == 5) {
     localStorage.setItem('task1', 1);
}
location.href = "quest.html";
}
```

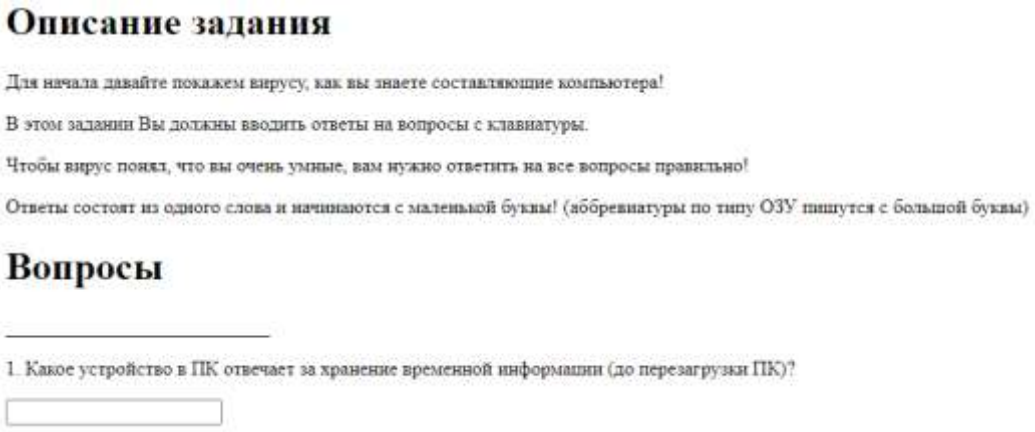

Рисунок 2 – Вопросы из первого задания

Но можно создавать и задания, на которые ответ можно выбирать, а не вносить с клавиатуры. Выбор ответов можно реализовать через элемент radio button. Второе задание также состоит из 5-и вопросов (см. рисунок 3). Чтобы сделать проверку правильности выбранных ответов, можно создать форму, в которую будут записываться элементы выбора ответов. Из этой формы можно получить значение выбранного элемента. Форма первого вопроса выглядит так:

```
<p>1. В каком из вариантов записан полный путь к файлу?</p>
<form id="question1">
     <input      type="radio"      name="answer1"      value="1">
C:\FRST\Logs\folder:file.txt<br>
     <input      type="radio"      name="answer1"      value="2">
\FRST\Logs\folder\file.txt<br>
     <input      type="radio"      name="answer1"      value="3">
C:\FRST\Logs\folder\file.txt<br>
     <input      type="radio"      name="answer1"      value="4">
\folder:file.txt<br>
</form>
```

Тогда код для определения выбранного ответа для первого вопроса выглядит следующим образом (данный код работает и для остальных вопросов в задании):

```
var question1 = document.getElementById('question1');
if (question1.elements['answer1'].value == 3) score++;
```

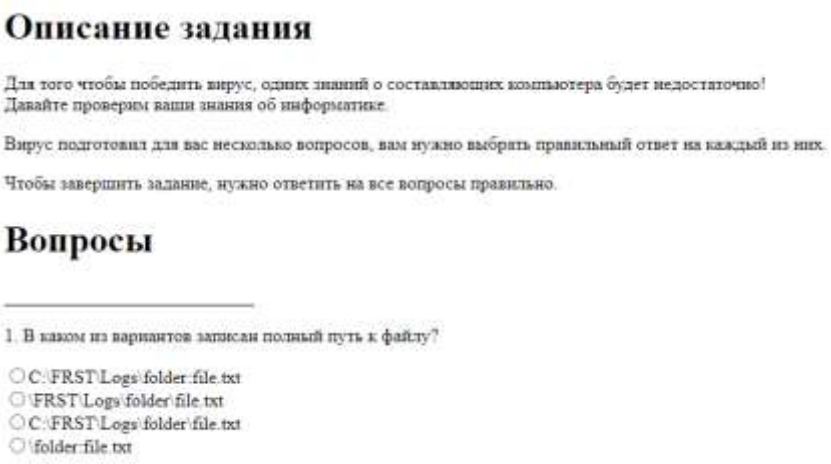

Рисунок 3 – Второе задание веб-квеста

Ранее также было описано, что при выполнении задании, на странице с квестом возле кнопки появляется текст “(Выполнено)” (см. рисунок 4). Нужно это для того, чтобы у пользователя не возникала путаница, пока он проходит квест.

Рисунок 4 – Страница квеста, если выполнить первые 3 задания

Если выполнены все задания, то JavaScript код перенаправляет пользователя на страницу result.html, где ему предлагается пройти веб-квест ещё раз (см. рисунок 5). На этом работа веб-квеста завершается.

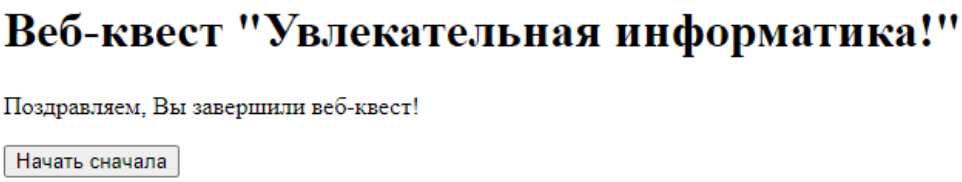

Рисунок 5 – Страница result.html

В данной статье описаны процесс разработки веб-квестам и сама работа веб-квеста. Работу приложения можно представить следующим образом:

1. Пользователь заходит в приложение и ему предлагается начать прохождение квеста.
2. Во время прохождения квеста, пользователь выбирает или вводит ответы на вопросы и при совпадении с правильным ответом засчитываются баллы.
3. После выполнения всех заданий пользователя переводит на страницу с результатами, где выводится количество набранных им баллов.

Библиографический список

1. Крокфорд Д. Как устроен JavaScript. СПб.: Питер, 2019. 304 с.
2. Минник К. JavaScript для чайников. М.: Диалектика, 2019. 320 с.
3. Фримен Э. Изучаем программирование на JavaScript. СПб.: Питер, 2015. 635 с.
4. Google Формы - бесплатное создание форм онлайн URL: https://www.google.com/intl/ru_ru/forms/about/