УДК 004

Создание обучающей веб-игры на развитие памяти с помощью JavaScript

Болтовский Гавриил Александрович
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
Целью данной статьи является создание обучающего веб-игры. Программа реализована с языка программирования JavaScript, а также HTML. Результатом исследования является готовая программа с подробным описанием принципов её реализации.
Ключевые слова: JavaScript, HTML, веб-игра

Creating an educational web game for memory development using JavaScript

Boltovsky Gavriil Alexandrovich
Sholom-Aleichem Priamursky State University
Student

Abstract
The purpose of this article is to create an educational web game. The program is implemented with the programming language JavaScript, as well as HTML. The result of the research is a ready-made program with a detailed description of the principles of its implementation.
Keywords: JavaScript, HTML, WebGame

1 Введение

1.1 Актуальность исследования

Игры можно разрабатывать и запускать в веб-среде (в браузерах). Для этого существуют отдельные языки программирования, позволяющие создать не только дизайн и визуальную составляющую проекта, но и его функционал. Развитие памяти является постоянным процессом в жизни человека и существует много способов улучшить работу памяти. Исключением не стали интерактивные виды развлечений. В настоящее время существует много различных игр, которые тем или иным образом помогают игроку развить свою память. Чаще всего игры задействуют функцию запоминания (например, в игре необходимо запомнить расположение объекта или его отдельные свойства).

1.2 Обзор исследований

Геймификация является одним из популярных направлений в разработке современных методик обучения. Например, в исследовании [1]

рассматривается геймификация в образовательном процессе на примере соревновательных практик у студентов гуманитарного колледжа. В работе Б.А. Байболотова [2] геймификационные практики используются для развития математической логики среди учеников средней школы; автором рассмотрен опыт внедрения, а также возникшие трудности.

Язык программирования JavaScript активно используется для создания игр. Существует множество статей по этой теме, например, Р. В. Семченко и П.А. Еровлев описывают создание игры «Змейка» [3]. Игра «Тетрис» была создана Е.Н. Поварницыным, процесс её разработки описан в исследовании [4].

1.3 Цель исследования

Цель исследования является – разработка игры на развитие памяти.

1.4 Постановка задачи

Задачи исследования направлены на создание обучающей веб-игры, целью которой является развитие памяти у пользователей. В рамках данного проекта необходимо разработать сценарий работы игры и описать взаимодействие пользователя с игровой системой. В первую очередь требуется создать разметку HTML-страниц, включающую игровое поле в виде таблицы 4x4, где будут располагаться пары изображений. Затем следует реализовать функцию случайной генерации расположения изображений на игровом поле и разработать алгоритм, проверяющий правильность выбранных пар картинок, а также подсчитывающий количество допущенных ошибок. Важной задачей является также создание логики взаимодействия ячеек таблицы.

2 Методы исследования

Методы исследования, примененные в данном проекте, основываются на использовании технологий CSS, JavaScript и HTML для создания обучающей веб-игры. В ходе разработки не применялись специфические интегрированные среды разработки; работа велась исключительно с использованием текстового редактора и браузера. В процессе реализации игрового проекта особое внимание уделялось созданию удобного и интуитивно понятного интерфейса, а также разработке алгоритмов, обеспечивающих корректное функционирование игры. HTML использовался для структуры веб-страниц и создания игрового поля в виде таблицы 4x4. CSS применялся для стилизации элементов интерфейса. JavaScript был задействован для реализации игровой логики, включая генерацию случайного расположения пар изображений, проверку правильности выбора игрока, подсчет ошибок и управление состоянием игры.

3 Результаты и обсуждения

В качестве проекта для разработки была выбрана игра, в которой нужно запомнить расположение пар изображений и выбрать их, не допустив ошибок.

Перед началом разработки также нужно описать сценарий работы игры и взаимодействия игрока с ней. Описать ход работы игры можно следующим образом:

1. Пользователь заходит на главную страницу, где объясняются правила игры. На странице также есть кнопка, которая начинает игру;

2. При нажатии на кнопку, пользователя перенаправляет на страницу с игровым процессом, где начинает работать JS код с игровой логикой;

3. Пользователь изучает расположение изображений. После изображения скрываются, и игрок начинает выбирать пары картинок;

4. Если пользователь выбрал все пары и не допустил больше 3-х ошибок, то его переводит на финальную страницу, в которой его поздравляют с победой и предлагают пройти игру ещё раз;

5. Если пользователь допустил больше 3-х ошибок, то появляется сообщение о поражении, и игрок возвращается на главную страницу.

Разработку игры можно разделить на несколько этапов:

1. Задать разметку на HTML страницах;

2. Написать функцию случайной генерации игрового поля и заполнения изображениями;

3. Написать функцию, которая проверяет правильность выбранных пар изображений. В случае если выбраны несовпадающие изображения, то увеличить счётчик ошибок на 1;

4. Написать функцию, которая проверяет количество ошибок (если их 3, то игра завершается) и количество открытых изображений (если их столько же, сколько было закрыто изображений в начале игры, то игрок побеждает).

Первым делом необходимо создать игровое поле. Это будет таблица 4*4, в которую будут записываться числа случайным образом. Чтобы создать таблицу, в HTML есть тег «table». Внутри этого тега задаются строки и столбцы таблицы с помощью тегов «tr» и «td». Ячейки могут принимать значения, но в рамках проекта они будут при инициализации пустыми. Также ячейки могут принимать JS-функции, которые будут срабатывать при нажатии. Для этого есть свойство «onclick» для тега «td». Функция, отвечающая за открытие ячейки (показ числа/изображения внутри неё) называется «flipCard». Таким образом код таблицы можно записать как 4 тега «tr», внутри которых будет ещё 4 тега «td» с параметром «onclick» (рис. 1).

```
<table>
  <tr>
    <td onclick="flipCard(0, 0)"></td>
    <td onclick="flipCard(0, 1)"></td>
    <td onclick="flipCard(0, 2)"></td>
    <td onclick="flipCard(0, 3)"></td>
  </tr>
  <tr>
    <td onclick="flipCard(1, 0)"></td>
    <td onclick="flipCard(1, 1)"></td>
    <td onclick="flipCard(1, 2)"></td>
    <td onclick="flipCard(1, 3)"></td>
  </tr>
  <tr>
    <td onclick="flipCard(2, 0)"></td>
    <td onclick="flipCard(2, 1)"></td>
    <td onclick="flipCard(2, 2)"></td>
    <td onclick="flipCard(2, 3)"></td>
  </tr>
  <tr>
    <td onclick="flipCard(3, 0)"></td>
    <td onclick="flipCard(3, 1)"></td>
    <td onclick="flipCard(3, 2)"></td>
    <td onclick="flipCard(3, 3)"></td>
  </tr>
</table>
```

Рисунок 1 – Код игрового поля внутри проекта

Следует задать CSS стиль для тега «td», чтобы ячейки были отделены друг от друга. Это сделает игру приятной и простой для пользователя (рис. 2).

Рисунок 2 – Игровое поле на странице в браузере

Но поле на данный момент не заполнено и нажатие на ячейки ничего не даёт. Для реализации функциональной части игры нужно использовать JavaScript. Описать работу JS кода можно следующим образом:

1. Создаётся массив значений (или изображений), которые будут находиться в ячейках игрового поля. Они должны попарно повторяться, то есть всего уникальных элементов 8, но с учётом их пар длина массива будет составлять 16 элементов.

2. Массив случайным образом перемешивается. Для этого необходимо создать отдельную функцию, которая будет называться «shuffle».

3. Создаются необходимые переменные для организации игрового процесса: количество выбранных ячеек и количество набранных ошибок, а также массив правильно раскрытых ячеек.

4. В начале игры все ячейки раскрыты какое-то время, чтобы пользователь мог запомнить их расположение. Для этого также необходимо создать отдельную функцию, которая будет называться «displayImages».

5. Также необходимо создать функцию отображения содержимого ячейки при нажатии. Эта функция уже есть в HTML коде (рис. 1) и называется «flipCard».

6. Также необходимо создать функцию завершения игры, которая проверяет количество набранных ошибок и если оно совпадает с максимально допустимым значением, то игру необходимо завершить.

После объявления массива, первой вызывается функция перемешивания элементов внутри него. Алгоритм перемешивания можно описать так: запускается цикл, который проходит по каждому элементу, начиная с конца массива. На каждом шаге цикла выбирается случайная позиция в массиве и i-й элемент меняется местами с j-м (рис. 3).

```
function shuffle(array) {
        for (var i = array.length - 1; i > 0; i--) {
                var j = Math.floor(Math.random() * (i + 1));
                var temp = array[i];
                array[i] = array[j];
                array[j] = temp;
        }
}
```

Рисунок 3 – Код функции shuffle

Эта простая в реализации функция позволяет перемешать элементы внутри массива. Функция и не идеальная и сильно зависит от функции «Math.random», но для проекта она будет подходящим вариантом (рис. 4).

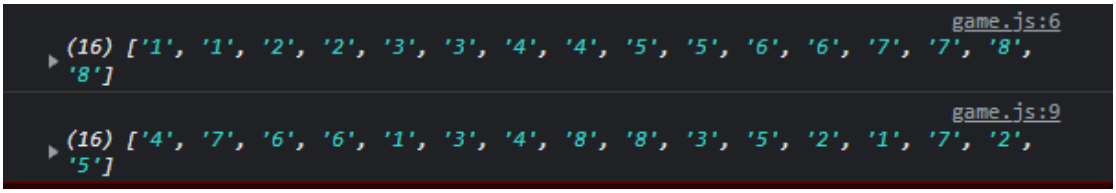

Рисунок 4 – Пример работы функции shuffle

После перемешивания значения выводятся на экран, чтобы пользователь мог их запомнить. Для этого была создана функция «displayImages», а также функция «hideImages», которая будет скрывать содержимое ячеек спустя некоторое время.

Функции работают просто: «displayImages» получает все элементы «td» на странице (то есть все ячейки), сохраняет их в массив и записывает в ячейки числа из массива, который был создан ранее. Функция «hideImages» работает наоборот: содержимое всех ячеек очищается через функцию «innerHTML» (рис. 5).

```
function displayImages() {
        var cards = document.getElementsByTagName("td");
        for (var i = 0; i < cards.length; i++) {
                cards[i].innerHTML = images[i];
        }
}

function hideImages() {
        var cards = document.getElementsByTagName("td");
        for (var i = 0; i < cards.length; i++) {
                cards[i].innerHTML = "";
        }
}
```

Рисунок 5 – Функции «displayImages» и «hideImages»

Таким образом, получилось создать функционал для отображения содержимого ячеек при старте игры (рис. 6).

Игра на развитие памяти

6	1	7	2
3	8	3	8
7	5	4	5
2	6	4	1

Рисунок 6 – Пример работы функции «displayImages»

Самой важной функцией в проекте является «flipCard». Функция нужна для отображения содержимого ячеек по нажатии на них. Принимает 2 аргумента на входе: строку и столбец (то есть расположение ячейки в таблице). Функция работает следующим образом: после нажатия на ячейку запускается проверка количества открытых ячеек и открыта ли данная ячейка в данный момент. Функция сработает, если:

1. Открытых ячеек меньше 2-х.
2. Ячейка не была открыта ранее.
3. Пара для содержимого ячейки не была найдена на момент нажатия.

Эти условия необходимы для того, чтобы пользователь не мог несколько раз выбрать одну и ту же ячейку, что может привести к нечестной победе. Именно для этой цели создавались в начале массивы, которые хранят открытые в данный момент ячейки и правильно открытые пары.

Если все условия соблюдены, то выполняются следующие действия: алгоритм получает значение ячейки по формуле «строка * 4 + столбец» (например, если мы выбрали ячейку во второй строке и третьем столбце (нужно учитывать, что подсчёт начинается с нуля в программировании), то в массиве это будет элемент с индексом 6), после этого позиция ячейки сохраняется (можно сохранять и значения элементов массива) в отдельном массиве и производится проверка на количество открытых ячеек (рис. 7).

```
if (flippedCards.length < 2 && !flippedCards.includes(row + "-" + col) &&
!matchedCards.includes(row + "-" + col)) {
        var card = document.getElementsByTagName("td")[row * 4 + col];
        card.innerHTML = images[row * 4 + col];
        flippedCards.push(row + "-" + col);
```

Рисунок 7 – Первая часть кода «flipCard»

Если количество открытых ячеек равно 2-м, то срабатывает следующая часть кода функции: делается выборка позиции ранее выбранных ячеек (для этого используется функция «parseInt») и сравниваются элементы в массиве по полученным значениям. Если выбранные ячейки совпадают, то они добавляются в массив правильно выбранных значений, очищается массив выбранных ячеек и производится проверка – если открыты все элементы таблицы, то игра завершается.

Но если значения выбранных ячеек не совпадают, то запускается функция «setTimeOut», которая выполняет содержимое с задержкой в указанное количество секунд. Внутри этой функции задан следующий алгоритм: сначала выбираются открытые ячейки (ссылка на элементы на странице) и после они очищаются, игроку засчитывается ошибка. Также происходит проверка на достижение максимально возможного количества ошибок (рис. 8).

```
if (flippedCards.length === 2) {
        var card1Row = parseInt(flippedCards[0].split("-")[0]);
        var card1Col = parseInt(flippedCards[0].split("-")[1]);
        var card2Row = parseInt(flippedCards[1].split("-")[0]);
        var card2Col = parseInt(flippedCards[1].split("-")[1]);

        if (images[card1Row * 4 + card1Col] === images[card2Row * 4 + card2Col]) {
                matchedCards.push(flippedCards[0], flippedCards[1]);
                flippedCards = [];

                if (matchedCards.length === images.length) {
                        endGame("Поздравляем! Вы выиграли!");
                }
        } else {
                setTimeout(function() {
                        var card1 = document.getElementsByTagName("td")[card1Row * 4 + card1Col];
                        var card2 = document.getElementsByTagName("td")[card2Row * 4 + card2Col];
                        card1.innerHTML = "";
                        card2.innerHTML = "";
                        flippedCards = [];
                        errorCount++;

                        if (errorCount === 3) {
                                endGame("Вы проиграли. Попробуйте еще раз.");
                        }
                }, 1000);
        }
}
```

Рисунок 8 – Вторая часть кода функции «flipCard»

Остаётся добавить функцию, которая будет завершать игру и выводить сообщение о победе или поражении. Для отключения функционала используется функция «pointerEvents» (рис. 9). Также после отключения функции «flipCard» появляется на странице кнопка, которая переводит игрока на главную страницу игры.

```
function endGame(message) {
        var table = document.getElementsByTagName("table")[0];
        table.style.pointerEvents = "none";

        var messageDiv = document.getElementById("message");
        messageDiv.innerHTML = message;

        var mainDiv = document.getElementById("gameFrame");
        var restartButton = document.createElement("input");
        restartButton.setAttribute("type","button");
        restartButton.setAttribute("value","Начать сначала");
        restartButton.setAttribute("onclick",'location.href="index.html";');
        mainDiv.appendChild(restartButton);
}
```

Рисунок 9 – Код функции «endGame»

На вход функция получает сообщение, которое выводится в зависимости от результата прохождения игры. Таким образом была создана игра, рассчитанная на развитие памяти у игроков (рис. 10).

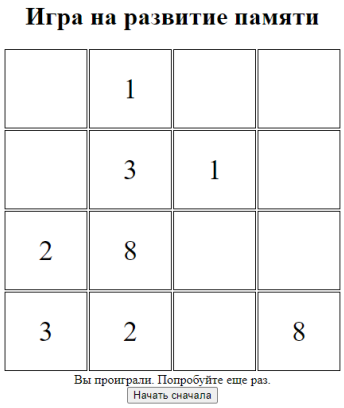

Рисунок 10 – Получившийся результат в ходе разработки

4 Выводы

Таким образом была создана игра на развитие памяти с использованием языка программирования JavaScript.

Библиографический список

1. Сейтказиева Н. С. Внедрение элементов геймификации в образовательный процесс // Вестник Международного Университета Кыргызстана. 2021. № 1(42). С. 131-136.
2. Байболотов Б. А. Геймификация как средство формирования математической логики в средней школе // Вестник Иссык-Кульского университета. 2023. № 55. С. 101-106.
3. Семченко Р. В., П. А. Еровлев Создание игры "Змейка" на JavaScript // Постулат. 2019. № 8(46). С. 39.
4. Поварницын, Е. Н. Создание браузерной игры «Тетрис» // Форум молодых ученых. 2020. № 2(42). С. 289-296.