УДК 004

Разработка web-игры на JavaScript

Андриенко Иван Сергеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описывается процесс разработки web-игры на JavaScript. Для создания игры использовался тег Canvas. Итоговая версия игры была протестирована и продемонстрирована.
Ключевые слова: Web-игра, JavaScript, HTML5, разработка игр.

Web game development in JavaScript

Andrienko Ivan Sergeevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes the process of developing a web game in JavaScript. The Canvas tag was used to create the game. The final version of the game was tested and demonstrated.
Keywords: Web game, JavaScript, HTML5, game development.

1 Введение

1.1 Актуальность

В современном информационном обществе специалист в любой сфере деятельности должен уметь создавать Web-страницы, которые являются основой глобальной сети. Современными средствами для создания интерактивных Web-приложений является язык программирования JavaScript.

Используя технологию Canvas и HTML5, можно создать вполне не плохую игру, при этом задействуя минимальные знания.

1.2 Обзор исследований

В своей работе Е.А. Сергеева реализовала анимацию для игры «пятнашки», написанную на JavaScript [1]. А.А. Артёменко реализовал такой перечень возможностей как управление змейкой с помощью стрелок, запись

рекордов и защита от ошибок. [2]. Е.В. Соловьева, Т.А. Максимова подробно описали разработку развивающей компьютерной игры-головоломки с помощью функционального языка JavaScript. Описаны элементы программы, позволяющие реализовать игровой процесс. [3]. В своей работе И.Н. Егорова, Е.А. Бондаренко представили аналитическое

исследование новейших технологий WEB-дизайна, анализ основных алгоритмов продвижения сайтов и их практическая реализация в виде интерактивного мультимедийного обучающего пособия «HTML5/CSS3, JavaScript/SEO. Разработка современных сайтов» [4]. М.Е. Кочитов в своей научной статье рассмотрел возможность веб-рисования на HTML5 с помощью canvas. [5].

1.3 Цель исследования

Цель исследования – разработать web-игру на JavaScript с помощью метода Canvas и продемонстрировать её.

2 Материалы и методы

Для разработки игры использовался язык программирования JavaScript и технологии Canvas и HTML5.

3 Результаты и обсуждения

Использование тега Canvas очень удобно при создании игры на JavaScript. Данная технология позволяет создавать графику, например создать анимацию или нарисовать графики.

Для начала создания игры необходимо подготовить изображения и аудио, которые будут использоваться в игре. В данной работе в игре будут использоваться изображения заднего фона, игрока, земли, по которой будет ходить игрок, и монстра (рис. 1). В качестве аудио будет использоваться звук смерти игрока и получения опыта из игры «Minecraft». Важно, чтобы изображения были в png формате и без заднего фона, иначе в игре картинки будут перекрывать друг друга.

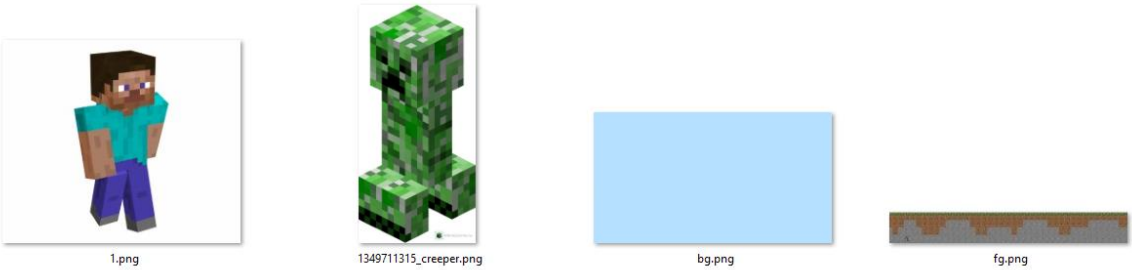

Рисунок 1 – Используемые изображения

Далее создаем каталоги и файлы, в которых будут храниться данные игры. Создаем папку с игрой – «Game», и в ней создаем папку с изображениями – «img», аудио – «audio», и кодом игры – «js» (рис. 2). Помещаем используемые изображения и аудио в созданные для них папки. В каталоге «js» создаем JavaScript файл, где и будет находиться код игры

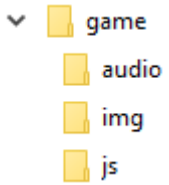

Рисунок 2 – Готовые каталоги для игры

Переходим к написанию кода игры. Сначала создадим переменные, необходимые для загрузки изображения, звука и работы метода Canvas (рис. 3).

```
var cvs = document.getElementById("canvas");
var ctx = cvs.getContext("2d");

var player = new Image();
var bg = new Image();
var fg = new Image();
var creep = new Image();

var score_audio = new Audio();
var death_audio = new Audio();
```

Рисунок 3 – Создание переменных для графики и звука

В созданные переменные загрузим изображения и аудио из папок (рис. 4).

```
player.src = "img/player.png";
bg.src = "img/bg.png";
fg.src = "img/fg.png";
creep.src = "img/creep.png";

fly.src = "audio/fly.mp3";
score_audio.src = "audio/score.mp3";
death_audio.src = "audio/death.mp3";
```

Рисунок 4 – Загрузка данных в переменные

Теперь создадим событие, что при нажатии на кнопку игрок будет прыгать. Используем метод «addEventListener». Заданная функция будет срабатывать как только произошло нажатие на заданную кнопку. В данном случае функцией будет прыжок главного героя. Чтобы прыжок был плавный, создадим множество функций, срабатывающих через 20 миллисекунд между друг другом. Так же создадим проверку условия стоит ли игрок на земле, чтобы сделать прыжок (рис. 5).

```
26  document.addEventListener("keydown", moveUp);
27
28  function moveUp() {
29
30  if (yPos >= 204){
31
32  function jump() {
33    yPos -= 15;
34  }
35  setTimeout(jump, 20);
36
37  function jump() {
38    yPos -= 15;
39  }
40  setTimeout(jump, 40);
41
42  function jump() {
43    yPos -= 15;
44  }
45  setTimeout(jump, 60);
46
47  function jump() {
48    yPos -= 15;
49  }
50  setTimeout(jump, 80);
51
52  function jump() {
53    yPos -= 20;
54  }
55  setTimeout(jump, 100);
56  function jump() {
57    yPos -= 20;
58  }
59  setTimeout(jump, 120);
60
61
62  }
63  }
```

Рисунок 5 – Код прыжка для главного героя

Теперь пропишем код, где будет находиться сам главный герой и где будут появляться монстры, атакующие игрока. Для игрока задаем позицию, а для монстра создаем объект (рис. 6).

```
69  var xPos = 30;
70  var yPos = 204;
71
72  var creeper = [];
73
74  creeper[0] = {
75    x : cvs.width,
76    y : 204
77  }
```

Рисунок 6 – Создание позиций игрока и монстра

Начинаем изображать объекты с помощью Canvas. Используем метод «drawImage» и, задав координаты изображений, вставляем их. Чтобы изобразить монстра создадим цикл. Это необходимо для того, чтобы монстры изображались в большом количестве, а не единожды. С помощью метода «Math.random()» создадим случайную генерацию чисел. Это поможет в

создании случайного появления монстра, используя проверку условия (рис. 7). После завершения функции пропишем «creep.onload = draw;».

```
function draw() {

ctx.drawImage(bg, 0, 0);

for(var i = 0; i < creeper.length; i++) {

ctx.drawImage(creep, creeper[i].x, creeper[i].y, 35, 65);

creeper[i].x--;


let value = Math.random()

if(value >= 0.9995) {
 creeper.push({
  x : cvs.width,
  y : 204
 });
}
}

ctx.drawImage(player, xPos, yPos,);
ctx.drawImage(fg, 0, 265);
```

Рисунок 7 – Изображение объектов

В раннее созданный цикл сразу разместим код проверки столкновений и записи очков. Проверка столкновения будет основана на проверки условия. Условие будет проверять не наложена ли изображение монстра на игрока, если это так, то будет воспроизводиться звук смерти игрока и уведомление о проигрыше с количеством очков. После закрытия уведомления будет перезагружена страница, и игра начнется сначала. Для сбора очков будет создано простое условие, если игрок перепрыгнул монстра, то очко зачислится (рис. 8). Перед циклом необходимо создать переменную для очков.

```
// столкновения

if (xPos + player.width >= creeper[i].x
&& xPos <= creeper[i].x + creep.width
&& (yPos >= creeper[i].y )){
death_audio.play();
location.reload();
alert("Вы проиграли. Ваш счет - " + score);}

//o4ki

if(creeper[i].x == 5) {
 score ++;
 score_audio.play();
 }
}
```

Рисунок 8 – Создание столкновений и счета

Если запустить игру, то после прыжка игрок зависнет в воздухе. Для того чтобы игрок после прыжка плавно падал на землю добавим гравитацию. Так же добавим счет в левый угол экрана (рис. 9).

```
126   //гравитация
127
128   if (yPos <= 204){
129   yPos +=0.6
130   }
131
132   //счет
133
134   ctx.fillStyle = "#000";
135   ctx.font = "24px Verdana";
136   ctx.fillText("Счет: " + score, 50, 50);
137
138   requestAnimationFrame(draw);
139
140   }
141
142   creep.onload = draw;
```

Рисунок 9 – Вывод счета и анимация падения игрока

Игра готова. Теперь необходимо создать HTML документ в папке с игрой и прописать в нем код для запуска JavaScript файла (рис. 10). Созданный документ будет работать как ярлык – запускать созданную игру в браузере.

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <meta http-equiv="X-UA-Compatible" content="ie=edge">
7    <title>MainCube</title>
8   </head>
9   <body>
10
11   <canvas id="canvas" width="480" height="320"></canvas>
12
13   <script src="js/game.js"></script>
14  </body>
15  </html>
```

Рисунок 10 – Содержимое HTML файла

Запускаем и проверяем игру (рис. 11 и 12).

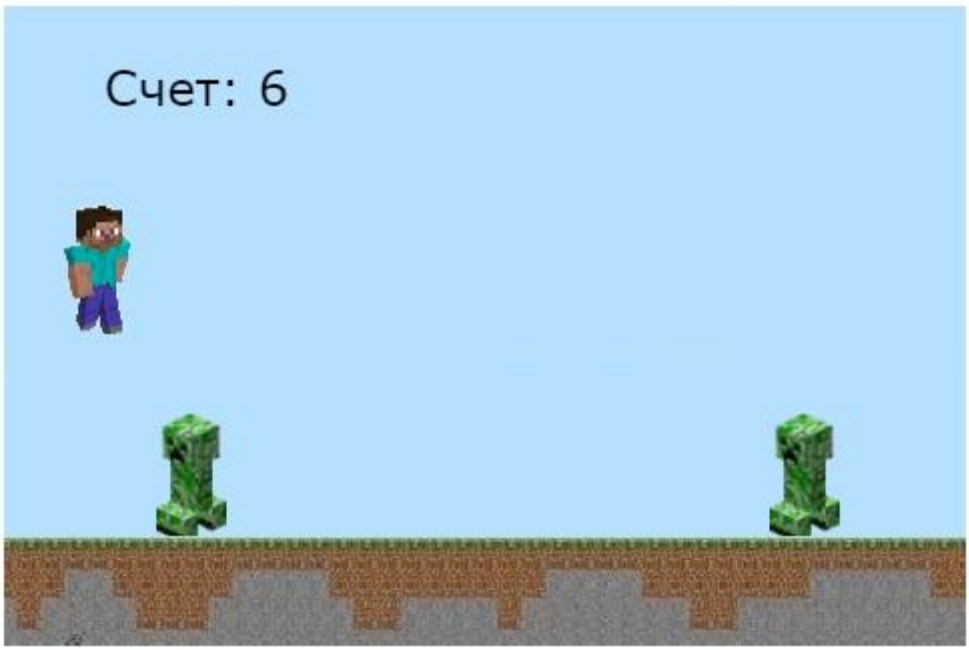

Рисунок 11 – Геймплей игры

Рисунок 12 – Конец игры

Выводы

В данной работе была разработана web-игра на JavaScript с помощью метода Canvas. Итоговая версия игры была протестирована и продемонстрирована.

Библиографический список

1. Сергеева Е.А. Анимация в игре "пятнашки" на JavaScript // Ступени роста - 2019. тезисы 71-й межрегиональной научно-практической конференции молодых ученых. 2019. С. 62-63.
2. Артёменко А.А. Создание браузерной игры «змейка» на языке JavaScript // Новые математические методы и компьютерные технологии в проектировании, производстве и научных исследованиях. Материалы XXII Республиканской научной конференции студентов и аспирантов. Гомель, 2019. С. 354-355.
3. Соловьева Е.В., Максимова Т.А. Компьютерная игра-головоломка «Судоку» на языке JavaScript // Студенческие научные достижения.

Сборник статей VIII Международного научно-исследовательского конкурса. 2020. С. 21-25.
4. Егорова И.Н., Бондаренко Е.А. Исследование новейших веб-технологий и алгоритмов продвижения сайтов // Восточно-Европейский журнал передовых технологий. 2012. Т. 6. № 2 (60). С. 63-66.
5. Кочитов М.Е. Рассмотрение возможности веб-рисования на HTML5 с помощью canvas. // Постулат. 2018. № 7 (33). С. 17.