МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

КАФЕДРА «ЭВМ и Системы»

Курсовой проект по теме:

«Разработка игры «Flappy Bird» с использованием JavaScript,HTML,CSS»

БрГТУ.150089-05 12 00

Листов: 24

Выполнил студент группы МС-1

Брашевец И.Н.

Проверила Дмитриева А.В.

Брест 2018

СОДЕРЖАНИЕ

Введение:

Flappy Bird — игра для мобильных устройств, в которой игрок с помощью касаний экрана должен контролировать полёт птицы между рядами зелёных труб, не задевая их.

Правила игры довольно просты:

Цель игры состоит в управлении полётом птицы, которая непрерывно передвигается между рядами зелёных труб. При столкновении с ними происходит завершение игры.

При отсутствии рывков птица падает из-за силы тяжести, и игра также завершается.

Игра доступна на компьютере и на мобильном устройстве это называется кроссплатформенность.

Кроссплатформенность - способность программного обеспечения работать с двумя и более аппаратными платформами и (или) операционными системами. Обеспечивается благодаря использованию высокоуровневых языков программирования, сред разработки и выполнения, поддерживающих условную компиляцию, компоновку и выполнение кода для различных платформ. Типичным примером является программное обеспечение, предназначенное для работы в операционных системах Linux и Windows одновременно.

1 Анализ технического задания:

По заданию необходимо реализовать игру «Flappy Birds» с использованием JavaScript, HTML,CSS.

Рассмотрим по отдельности: JavaScript, HTML,CSS.

JavaScript.

JavaScript изначально создавался для того, чтобы сделать web-странички «живыми». Программы на этом языке называются скриптами. В браузере они подключаются напрямую к HTML и, как только загружается страничка – тут же выполняются.

Программы на JavaScript – обычный текст. Они не требуют какой-то специальной подготовки.

В этом плане JavaScript сильно отличается от другого языка, который называется Java.

Когда создавался язык JavaScript, у него изначально было другое название: «LiveScript». Но тогда был очень популярен язык Java, и маркетологи решили, что схожее название сделает новый язык более популярным.

Планировалось, что JavaScript будет эдаким «младшим братом» Java. Однако, история распорядилась по-своему, JavaScript сильно вырос, и сейчас это совершенно независимый язык, со своей спецификацией, которая называется ECMAScript, и к Java не имеет никакого отношения.

JavaScript может выполняться не только в браузере, а где угодно, нужна лишь специальная программа – интерпретатор. Процесс выполнения скрипта называют «интерпретацией».

Компиляция и интерпретация, для программистов
Для выполнения программ, не важно на каком языке, существуют два способа: «компиляция» и «интерпретация».

- Компиляция – это когда исходный код программы, при помощи специального инструмента, другой программы, которая называется «компилятор», преобразуется в другой язык, как правило – в машинный код. Этот машинный код затем распространяется и запускается. При этом исходный код программы остаётся у разработчика.
- Интерпретация – это когда исходный код программы получает другой инструмент, который называют «интерпретатор», и выполняет его «как

есть». При этом распространяется именно сам исходный код (скрипт). Этот подход применяется в браузерах для JavaScript.

Современные интерпретаторы перед выполнением преобразуют JavaScript в машинный код или близко к нему, оптимизируют, а уже затем выполняют. И даже во время выполнения стараются оптимизировать. Поэтому JavaScript работает очень быстро.

Во все основные браузеры встроен интерпретатор JavaScript, именно поэтому они могут выполнять скрипты на странице. Но, разумеется, JavaScript можно использовать не только в браузере. Это полноценный язык, программы на котором можно запускать и на сервере, и даже в стиральной машинке, если в ней установлен соответствующий интерпретатор.

Что умеет JavaScript:

Современный JavaScript – это «безопасный» язык программирования общего назначения. Он не предоставляет низкоуровневых средств работы с памятью, процессором, так как изначально был ориентирован на браузеры, в которых это не требуется.

Что же касается остальных возможностей – они зависят от окружения, в котором запущен JavaScript. В браузере JavaScript умеет делать всё, что относится к манипуляции со страницей, взаимодействию с посетителем и, в какой-то мере, с сервером:

- Создавать новые HTML-теги, удалять существующие, менять стили элементов, прятать, показывать элементы и т.п.
- Реагировать на действия посетителя, обрабатывать клики мыши, перемещения курсора, нажатия на клавиатуру и т.п.
- Посылать запросы на сервер и загружать данные без перезагрузки страницы (эта технология называется "AJAX").
- Получать и устанавливать cookie, запрашивать данные, выводить сообщения…

Что НЕ умеет JavaScript?

JavaScript – быстрый и мощный язык, но браузер накладывает на его исполнение некоторые ограничения…

Это сделано для безопасности пользователей, чтобы злоумышленник не мог с помощью JavaScript получить личные данные или как-то навредить компьютеру пользователя.

Этих ограничений нет там, где JavaScript используется вне браузера, например на сервере. Кроме того, современные браузеры предоставляют свои механизмы по установке плагинов и расширений, которые обладают расширенными возможностями, но требуют специальных действий по установке от пользователя. Рисунок 1

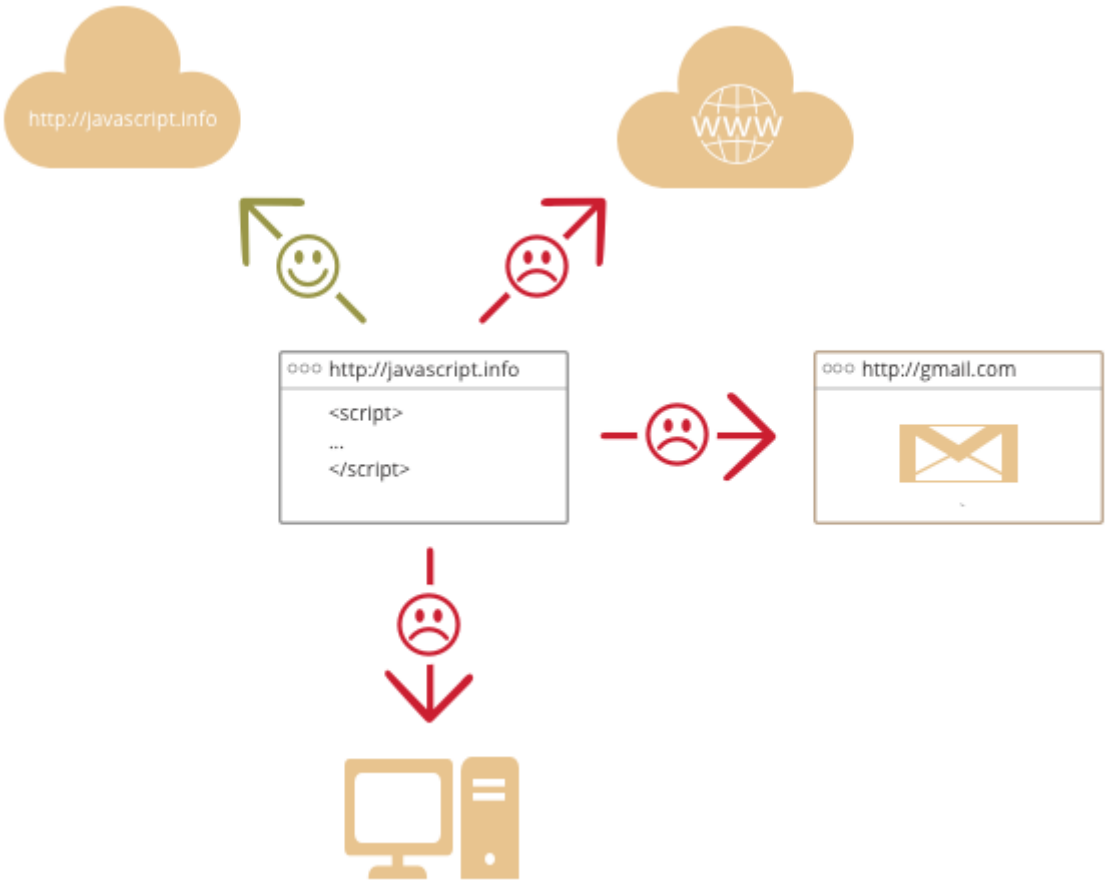

Рисунок 1

- JavaScript не может читать/записывать произвольные файлы на жесткий диск, копировать их или вызывать программы. Он не имеет прямого доступа к операционной системе.

Современные браузеры могут работать с файлами, но эта возможность ограничена специально выделенной директорией – «песочницей».

Возможности по доступу к устройствам также прорабатываются в современных стандартах и частично доступны в некоторых браузерах.

- JavaScript, работающий в одной вкладке, не может общаться с другими вкладками и окнами, за исключением случая, когда он сам открыл это окно или несколько вкладок из одного источника (одинаковый домен, порт, протокол).

Есть способы это обойти, но они требуют специального кода на оба документа, которые находятся в разных вкладках или окнах. Без него, из соображений безопасности, залезть из одной вкладки в другую при помощи JavaScript нельзя.

- Из JavaScript можно легко посылать запросы на сервер, с которого пришла страница. Запрос на другой домен тоже возможен, но менее удобен, т. к. и здесь есть ограничения безопасности.

В чём уникальность JavaScript?

Есть три замечательных особенности JavaScript:

- Полная интеграция с HTML/CSS.
- Простые вещи делаются просто.
- Поддерживается всеми распространёнными браузерами и включён по умолчанию.

Этих трёх вещей одновременно нет больше ни в одной браузерной технологии.

Поэтому JavaScript и является самым распространённым средством создания браузерных интерфейсов.

Тенденции развития

HTML 5 – эволюция стандарта HTML, добавляющая новые теги и, что более важно, ряд новых возможностей браузерам.

Вот несколько примеров:

- Чтение/запись файлов на диск (в специальной «песочнице», то есть не любые).
- Встроенная в браузер база данных, которая позволяет хранить данные на компьютере пользователя.

- Многозадачность с одновременным использованием нескольких ядер процессора.
- Проигрывание видео/аудио, без Flash.
- 2D и 3D-рисование с аппаратной поддержкой, как в современных играх.

Многие возможности HTML5 всё ещё в разработке, но браузеры постепенно начинают их поддерживать.

ECMAScript 6

Сам язык JavaScript улучшается. Современный стандарт ECMAScript 5 включает в себя новые возможности для разработки, ECMAScript 6 будет шагом вперёд в улучшении синтаксиса языка.

Современные браузеры улучшают свои движки, чтобы увеличить скорость исполнения JavaScript, исправляют баги и стараются следовать стандартам.

JavaScript становится всё быстрее и стабильнее, в язык добавляются новые возможности. Очень важно то, что новые стандарты HTML5 и ECMAScript сохраняют максимальную совместимость с предыдущими версиями. Это позволяет избежать неприятностей с уже существующими приложениями.

Впрочем, небольшая проблема с «супер-современными штучками» всё же есть. Иногда браузеры стараются включить новые возможности, которые ещё не полностью описаны в стандарте, но настолько интересны, что разработчики просто не могут ждать.

…Однако, со временем стандарт меняется и браузерам приходится подстраиваться к нему, что может привести к ошибкам в уже написанном, основанном на старой реализации, JavaScript-коде. Поэтому следует дважды подумать перед тем, как применять на практике такие «супер-новые» решения.

При этом все браузеры сходятся к стандарту, и различий между ними уже гораздо меньше, чем всего лишь несколько лет назад.

Всё идет к полной совместимости со стандартом.

Вместе с JavaScript на страницах используются и другие технологии. Связка с ними может помочь достигнуть более интересных результатов в тех местах, где браузерный JavaScript пока не столь хорош, как хотелось бы.

Чем JavaScript-разработчикам, может быть интересен Java?

В первую очередь тем, что подписанный Java-апплет может всё то же, что и обычная программа, установленная на компьютере посетителя. Конечно, для этого понадобится согласие пользователя при открытии такого апплета.

Достоинства:

- Java может делать всёот имени посетителя, совсем как установленная программа. Потенциально опасные действия требуют подписанного апплета и согласия пользователя.

Недостатки:

- Java требует больше времени для загрузки.
- Среда выполнения Java, включая браузерный плагин, должна быть установлена на компьютере посетителя и включена.
- Java-апплет не интегрирован с HTML-страницей, а выполняется отдельно. Но он может вызывать функции JavaScript.

Языки поверх JavaScript

Синтаксис JavaScript устраивает не всех: одним он кажется слишком свободным, другим – наоборот, слишком ограниченным, третьи хотят добавить в язык дополнительные возможности, которых нет в стандарте. В последние годы появилось много языков, которые добавляют различные возможности «поверх» JavaScript, а для запуска в браузере – при помощи специальных инструментов «трансляторов» превращаются в обычный JavaScript-код.

Это преобразование происходит автоматически и совершенно прозрачно, при этом неудобств в разработке и отладке практически нет.

При этом разные языки выглядят по-разному и добавляют совершенно разные вещи:

- Язык CoffeeScript – это «синтаксический сахар» поверх JavaScript. Он сосредоточен на большей ясности и краткости кода. Как правило, его особенно любят программисты на Ruby.
- Язык TypeScript сосредоточен на добавлении строгой типизации данных. Он предназначен для упрощения разработки и поддержки больших систем. Его разрабатывает Microsoft.

- Язык Dart интересен тем, что он не только транслируется в JavaScript, как и другие языки, но и имеет свою независимую среду выполнения, которая даёт ему ряд возможностей и доступна для встраивания в приложения (вне браузера). Он разрабатывается компанией Google.

HTML

HTML (англ. "hyper text markup language" — язык гипертекстовой разметки) — это специальный язык разметки, который применяется при создании сайтов в интернете.

Браузеры прекрасно понимают html и могут интерпретировать его в понятном виде. Вообще любая страница сайта — это html-код, который браузер переводит в дружественный вид для пользователя.

Язык разметки html получил широкую популярность. На данный момент — это единственный язык, с помощью которого создается разметка сайта.

2 Выбор метода решения задачи.

Реализую свой проект на JavaScript, HTML,CSS. Как того и требует задание.

Для реализации своего курсового проекта я воспользуюсь Phaser.

Phaser — это фреймворк, который позволяет нам очень быстро создавать игры. Не надо отвлекаться на Actor'ов, рендеринг, физику — фокусируемся на игровой логике.
Его однозначными плюсами есть Pixi.js. Это один из быстрейших движков, который рендерит с помощью WebGL. А в случае, если WebGL не поддерживается — на Canvas.
Также Phaser радует огромным набором готовых классов: SpriteAnimation, TileMap, Timer, GameState и много другое. В том числе, и компоненты физического движка: RigidBody, Physics и т.п.
Наличие данных компонентов значительно упрощает разработку.

Разработку веду в Addobe Brackets.

Brackets — свободный текстовый редактор для веб-разработчиков. Brackets ориентирован на работу с HTML, CSS и JavaScript. Эти же технологии лежат в основе самого редактора, что обеспечивает его кроссплатформенность т.е. совместимость с операционными системами Mac, Windows и Linux. Brackets создан и развивается Adobe Systems под лицензией MIT License и поддерживается на GitHub.

На сегодняшний день сообществом создано множество расширений, добавляющих большинство необходимых инструментов для работы над кодом, таких как система контроля версий Git, просмотр HTML-кода в браузере в реальном времени (Live Preview), синхронизация с FTP (Git-FTP). Принять участие в разработке и поддержке расширений может любой желающий.

Тестировать проект я буду в Google Chrome.

Google Chrome - браузер, разрабатываемый компанией Google на основе свободного браузера Chromium и движка Blink (до апреля 2013 года использовался WebKit). Первая публичная бета-версия для Windows вышла 2 сентября 2008 года, а первая стабильная — 11 декабря 2008 года. По данным StatCounter, Chrome используют около 300 миллионов интернет-пользователей, что делает его самым популярным браузером в мире — его рыночная доля на сентябрь 2018 года составляет 60.6 %

3 Проектирование функциональных частей.

На основании выбранного метода решения можно выделить следующие примерные функциональные части:

- процедура открытия программы будет производится запуском html-файла;
- процедура открытия новой html-страницы с игрой;
- Процедура нажатия на картинку и старт новой игры;
- Процедура генерации блоков стен;
- Процедура проверки игрового поля на вылет птички за пределы игрового поля
- Процедура проверки столкнулась ли наша птичка со стеной
- Процедура завершения игры
- Процедура вывода картинки с надписью «Конец игры»
- Процедура работы таймера
- Процедура перезагрузки страницы и начало новой игры

Для проекта необходимы файлы с расширением .css .html .js и картинки,

Создадим их и добавим вместе с картинками в отдельную папку.

4 Разработка программы

Первое, что нам необходимо сделать – это подключить игровой движок Phaser в нашем .html –файле, сделаем это так:

<script type ="text/javascript" src="phaser.min.js"></script>

Далее в в основном файле с расширением .js объявим функцию :

function ();

Процедура проверки вышла ли птичка за пределы поля:

```
if(this.bird.y < 0 || this.bird.y > 768)
        this.restartGame();
```

Функция «Прыжка» для курсового проекта:

```
  jump: function()
            {
    this.bird.body.velocity.y = -325;
    var animation = game.add.tween(this.bird);
    animation.to({angle: -20}, 100);
    animation.start();
    if(this.bird.alive == false)
        return;
  },
```

Функция генерации стен:

```
addRowOfPipes: function(){
        var hole= Math.floor(Math.random() *10) +2;
        for (var i=0; i<25; i++)
            if (i !=hole && i !=hole + 1 && i !=hole + 2)
                this.addOnePipe(1024, i * 50 + 8);
```

Функция добавления +1 когда птичка проходит сквозь стену:

```
  this.score +=1;
        this.labelScore.text= this.score;


    },
```

Функция таймера:

```
  function ClearClock() {
clearTimeout(clocktimer);
h=1;m=1;tm=1;s=0;ts=0;ms=0;
init=0;
readout='00:00:00.00';
document.getElementById('stopwatch').innerHTML = readout; }
function StartTIME() {
    var cdateObj = new Date();
    var t = (cdateObj.getTime() - dateObj.getTime())-(s*1000);
    if (t>999) s++;
    if (s>=(m*base))  {  ts=0; m++; }
    else {  ts=parseInt((ms/100)+s);
        if(ts>=base) { ts=ts-((m-1)*base); } }
    if (m>(h*base)) { tm=1; h++;  }
    else  { tm=parseInt((ms/100)+m);
        if(tm>=base)  { tm=tm-((h-1)*base);  } }  ms = Math.round(t/10);
    if (ms>99) ms=0;
    if (ms==0) ms='00';
```

```
    if (ms>0&&ms<=9) ms = '0'+ms;
    if (ts>0) {  ds = ts;
        if (ts<10) ds = '0'+ts;      }
    else
    ds = '00';  dm=tm-1;
    if (dm>0)  {
        if (dm<10) dm = '0'+dm; }
    else   dm = '00'; dh=h-1;
    if (dh>0) {
        if (dh<10)  dh = '0'+dh; }
     else
    dh = '00';
    readout = dh + ':' + dm + ':' + ds + '.' + ms;
    document.getElementById('stopwatch').innerHTML = readout;
    clocktimer = setTimeout("StartTIME()",1); }
function StartStop() {
    if (init==0) {
        ClearClock();
        dateObj = new Date();
        StartTIME();
        init=1; }
    else  {
        clearTimeout(clocktimer);
        init=0;}
```

Где непосредственно и будет происходить выполнение основной части скрипта.

Далее, функцией preload предзагрузим всё, что нам необходимо, а именно:

- Изображение начала и окончания игры
- Изображение нашей птички

- Изображение блока

Изображение начала игры. Рисунок 2:

Рисунок 2

Опишем скорость падения функцией и зададим скорость «падения» this.bird.body.gravity.y = 700;

Далее необходимо выбрать клавишу на какую будет осуществляться оправление «прыжком» нашей птички, можно выбрать любую клавишу на клавиатуре. Я выбрал клавишу SPACEBAR.

```
var spaceKey = game.input.keyboard.addKey (Phaser.Keyboard.SPACEBAR);

        spaceKey.onDown.add(this.jump, this);
```

Далее сделаем стены и промежуток(в миллисекундах) через который они будут появляться. Я выбрал 2000 миллисекунд.

Далее стоит добавить таймер для подсчета очков.

Следующей функцией будет проверка птички «на смерть» если птичка выйдет за пределы нашей игровой области или столкнётся со стенкой, появится изображение «Конец игры» . Рисунок 3:

Рисунок 3

Далее необходимо добавить функцию «прыжок»

Логика функции очень простая, при нажатии на клавишу наша птичка подпрыгивает вверх по оси y .

Рисование стен. Стены рисуются при помощи random выбирая один из блоков и удаляя соседний с двух сторон. Так у меня образуется разрыв в стене в 3 блока, что вполне достаточно для того, чтобы птичка пролетала без проблем.

Функция restartGame начинает новую игру.

Задаём размер нашего игрового поля.

И делаем таймер.

Как видим, игра не такая уж и сложная в исполнении.

5 Тестирование и анализ результатов:

Запустим программу, нам должно показаться окно Google Chrome с начальной картинкой Рисунок 4.

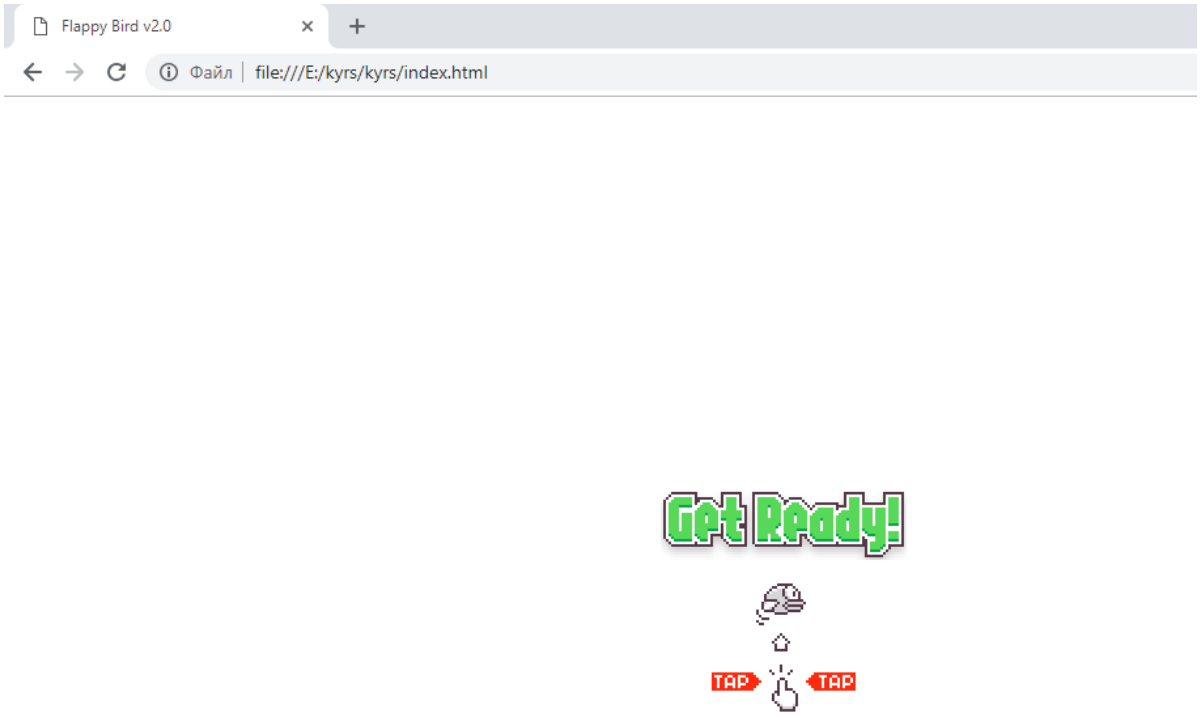

Рисунок 4

Нажмём на картинку и появится окно игры с таймером и счётчиком очков Рисунок 4.

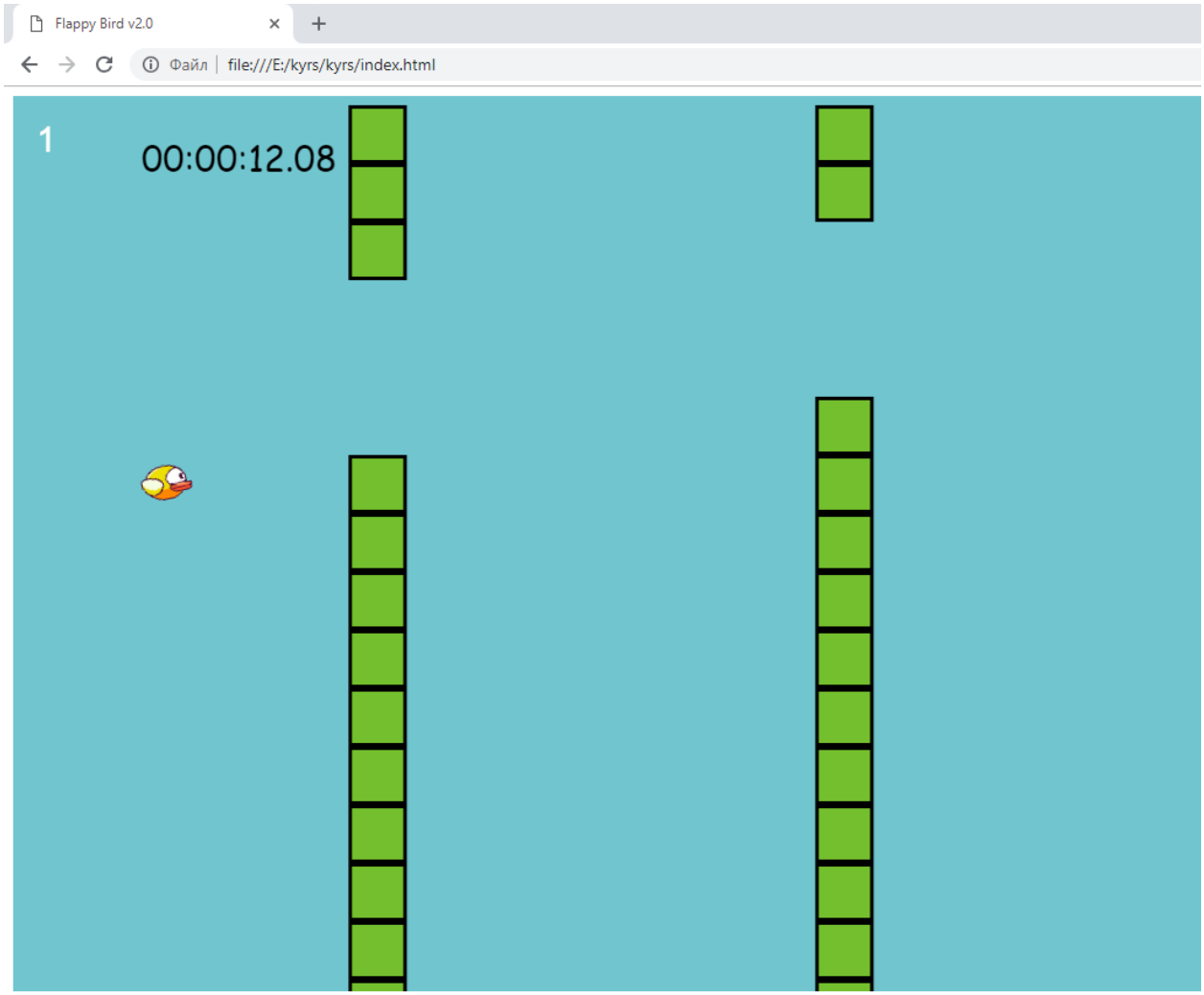

Рисунок 4

После проигрыша появляется экран «Игра закончена» Рисунок 5

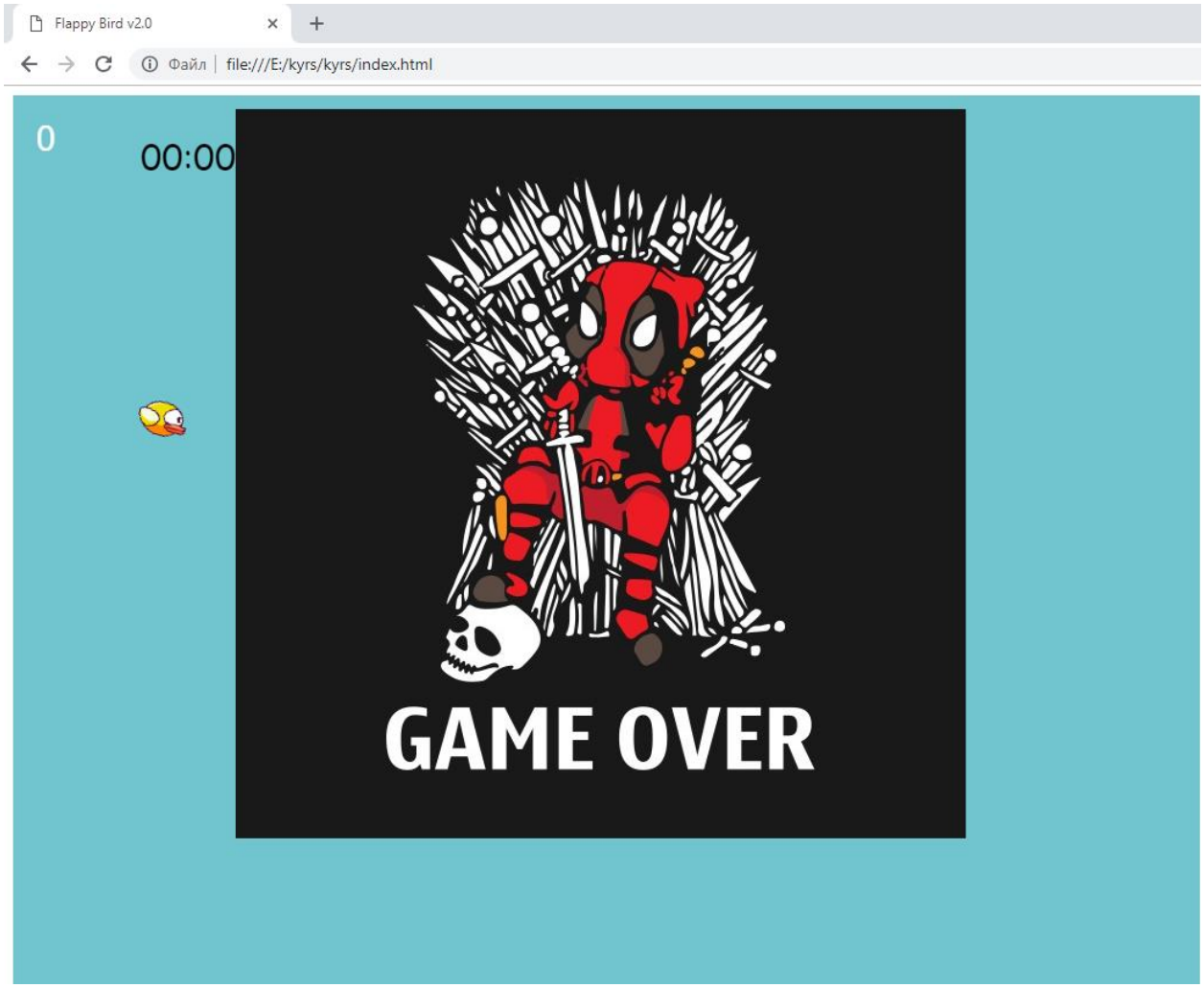
Flappy Bird v2.0
Файл | file:///E:/kyrs/kyrs/index.html
0
00:00
GAME OVER

Рисунок 5

При нажатии на эту картинку опять появляется кнопка начала игры Рисунок 6

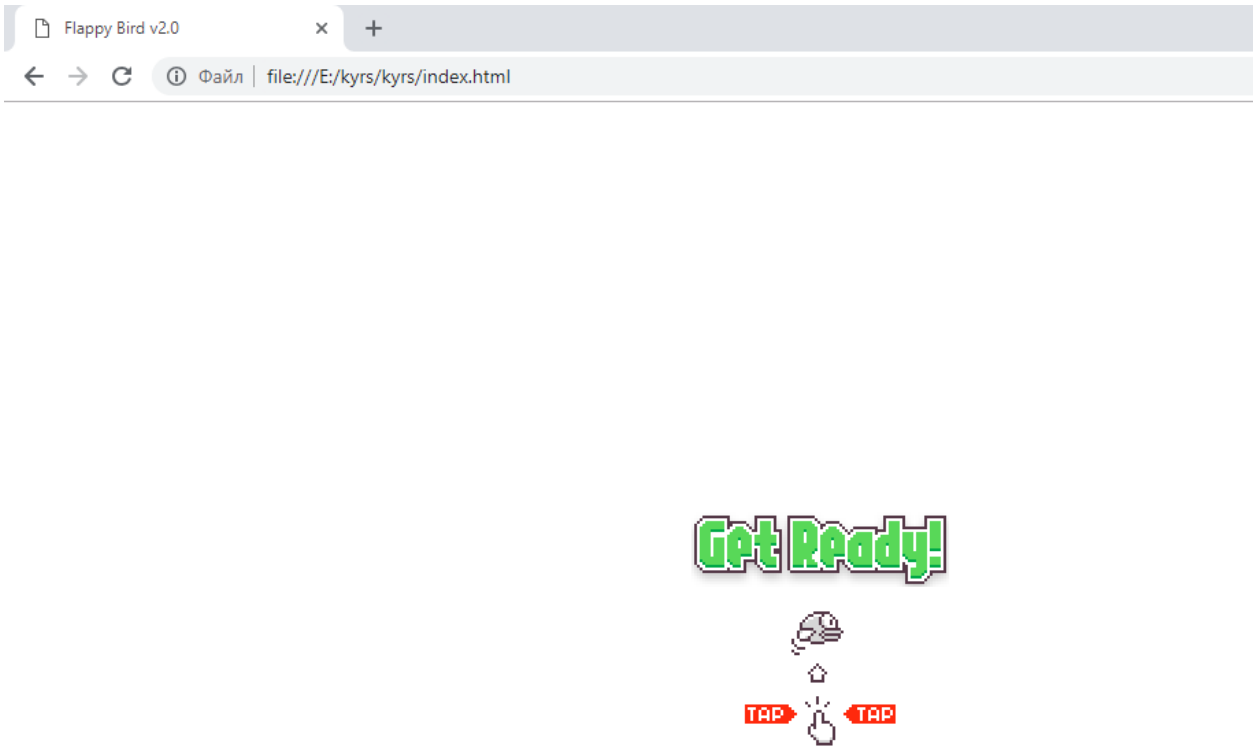

Рисунок 6

Заключение:

В ходе данного курсового проекта было реализовано приложение «Flappy Birds» с использованием JavaScript, HTML,CSS. Программа реализует запуск новой игры, проверку птички на «смерть», проверка вылета птички за пределы игрового поля, запуск таймера. Были выделены Начало и Конец игры картинками. Можно улучшить некоторые функции:

- Сделать выбор уровня сложности
- Сделать выбор изображения птички
- Добавление движущихся препятствий

СПИСОК ЛИТЕРАТУРЫ

1. Введение [Электронный ресурс]. Режим доступа: http://historygames.ru/logicheskie-igryi/istoriya-morskogo-boya.html. – Дата доступа 04.11.18.

2. JavaScript [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/JavaScript. – Дата доступа 14.11.18.

3. HTML [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/HTML. – Дата доступа 14.11.18.

4. Visual Studio Code [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Visual_Studio_Code. – Дата доступа 14.11.18.

5. CSS [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/CSS. – Дата доступа 14.11.18.