Федеральное государственное бюджетное образовательное учреждение

высшего образования

Ивановский Государственный университет

Факультет математики и компьютерных наук

Кафедра прикладной математики и компьютерных наук

Курсовая работа

на тему:

«Java Script в HTML5»

|  |  |
| --- | --- |
| Работу выполнил:  ФИО  Студент 3 курса, дневного отделения  Факультета математики и компьютерных  наук | Научный руководитель:  кандидат экономических наук  Степович-Цветкова Галина Сергеевна |

Иваново 2018

Оглавление

[Введение 3](#_Toc509683265)

[Описание работы с приложением 3](#_Toc509683266)

[HTML 5 3](#_Toc509683267)

[CSS 5](#_Toc509683268)

[Java Script 7](#_Toc509683269)

[Canvas 9](#_Toc509683270)

[Структура проекта 11](#_Toc509683271)

[Исходный код 12](#_Toc509683272)

[Пример работы с приложением 16](#_Toc509683273)

[Список литературы 17](#_Toc509683274)

# Введение

Целью данной курсовой работы была разработка приложения с использованием Java Script и HTML5. Было разработано интерактивное приложение позволяющие рисовать в окне браузера.

# Описание работы с приложением

Для начала работы с приложением пользователю необходимо открыть файл paint.html с помощью какого-либо браузера. На странице сайта находится окно, в котором пользователь может рисовать. Также доступно меню выбора цвета кисти, её толщины и кнопка очистки холста.

# HTML 5

Язык HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986—1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки.

HTML — стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML (или XHTML). Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

Язык HTML до 5-ой версии определялся как приложение SGML (стандартного обобщённого языка разметки по стандарту ISO 8879). Спецификации HTML5 формулируются в терминах DOM (объектной модели документа).

Во всемирной паутине HTML-страницы, как правило, передаются браузерам от сервера по протоколам HTTP или HTTPS, в виде простого текста или с использованием шифрования.

Текстовые документы, содержащие разметку на языке HTML (такие документы традиционно имеют расширение .html или .htm), обрабатываются специальными приложениями, которые отображают документ в его форматированном виде. Такие приложения, называемые «браузерами» или «интернет-обозревателями», обычно предоставляют пользователю удобный интерфейс для запроса веб-страниц, их просмотра (и вывода на иные внешние устройства) и, при необходимости, отправки введённых пользователем данных на сервер. Наиболее популярными на сегодняшний день браузерами являются Google Chrome, Mozilla Firefox, Opera, Internet Explorer и Safari.

HTML5 — язык для структурирования и представления содержимого всемирной паутины. Это пятая версия HTML. Цель разработки HTML5 — улучшение уровня поддержки мультимедиа-технологий с одновременным сохранением обратной совместимости, удобочитаемости кода для человека и простоты анализа для парсеров.

В HTML5 реализовано множество новых синтаксических особенностей. Например, элементы <video>, <audio> и <canvas>, а также возможность использования SVG и математических формул.

Для целей данной курсовой работы был использован тег <canvas>, он позволяет отображать доступное для рисования поле.

В дополнение к определению разметки HTML5 устанавливает API, который может быть использован с JavaScript.

API — набор готовых классов, процедур, функций, структур и констант, предоставляемых библиотекой или операционной системой для использования во внешних программных продуктах.

* контроль над проигрыванием медиафайлов, который может использоваться, например, для синхронизации субтитров с видео;
* хранение данных в браузере;
* File API: возможность загрузки документа через выбор (тег <input type="file">) или перетаскиванием (Drag-and-drop)
* Drag-and-drop: предоставляет набор событий для каждого элемента DOM, таких как появление и нахождение в его зоне, благодаря которым разработчик может информировать пользователя о необходимых действиях и идентификаторе перетаскиваемого файла, содержащего адрес, имя, тип, размер и дату изменения;
* управление историей браузера;

# CSS

CSS используется создателями для задания цветов, шрифтов, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось разделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом.

Правила CSS пишутся на формальном языке CSS и располагаются в таблицах стилей, то есть таблицы стилей содержат в себе правила CSS. Эти таблицы стилей могут располагаться как в самом веб-документе, внешний вид которого они описывают, так и в отдельных файлах, имеющих формат CSS.

Как известно, HTML-документы строятся на основании иерархии элементов, которая может быть наглядно представлена в древовидной форме. Элементы HTML друг для друга могут быть родительскими, дочерними, элементами-предками, элементами-потомками, сестринскими.

Элемент является родителем другого элемента, если в иерархической структуре документа он находится сразу, непосредственно над этим элементом. Элемент является предком другого элемента, если в иерархической структуре документа он находится где-то выше этого элемента.

В CSS могут задаваться при помощи селекторов не только одиночные элементы, но и элементы, являющиеся потомками, дочерними или сестринскими элементами других элементов.

Селектор, расположенный в левой части правила, определяет, на какие части документа распространяется правило. Блок объявлений располагается в правой части правила. Он помещается в фигурные скобки, и, в свою очередь, состоит из одного или более объявлений, разделённых знаком «;». Каждое объявление представляет собой сочетание свойства CSS и значения, разделённых знаком ": ".

селектор, селектор {

свойство: значение;

свойство: значение;

свойство: значение;

}

Например, селектор body применяется к телу документа HTML. Для него указан серый цвет фона.

body {

background-color: gainsboro; /\* Цвет фона \*/

}

Класс или идентификатор может быть присвоен какому-нибудь элементу (тегу) HTML посредством атрибутов class или id этого элемента (тега). Основное отличие между классами элементов и идентификаторами элементов в том, что идентификатор предназначен для одного элемента, тогда как класс обычно присваивают сразу нескольким. Тем не менее, современные браузеры, как правило, корректно отображают множественные элементы с одинаковым идентификатором. Также отличие в том, что могут существовать множественные классы (когда класс элемента состоит из нескольких слов, разделённых пробелами). Для идентификаторов такое невозможно. Важно отметить следующее отличие идентификатора от класса: идентификаторы широко используются в JavaScript для нахождения уникального элемента в документе.

В данной курсовой работы классы CSS использовались для задания стилей элементов, а идентификаторы для нахождения элементов в JavaScript.

# Java Script

JavaScript — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией языка ECMAScript.

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

Несмотря на схожий с Си синтаксис, JavaScript по сравнению с языком Си имеет коренные отличия:

* объекты с возможностью интроспекции;
* функции как объекты первого класса;
* автоматическое приведение типов;
* автоматическая сборка мусора;
* анонимные функции.

Структурно JavaScript можно представить в виде объединения трёх чётко различимых друг от друга частей:

* ядро (ECMAScript),
* объектная модель браузера (Browser Object Model или BOM),
* объектная модель документа (Document Object Model или DOM).

ECMAScript — основа для построения скриптовых языков. Спецификация ECMAScript описывает типы данных, инструкции, ключевые и зарезервированные слова, операторы, объекты, регулярные выражения, не ограничивая авторов производных языков в расширении их новыми составляющими.

Объектная модель браузера — браузер-специфичная часть языка, являющаяся прослойкой между ядром и объектной моделью документа. Основное предназначение объектной модели браузера — управление окнами браузера и обеспечение их взаимодействия. Каждое из окон браузера представляется объектом window, центральным объектом DOM.

Объектная модель документа — интерфейс программирования приложений для HTML и XML-документов. Согласно DOM, документ может быть представлен в виде дерева объектов, обладающих рядом свойств, которые позволяют производить с ним различные манипуляции:

* генерация и добавление узлов,
* получение узлов,
* изменение узлов,
* изменение связей между узлами,
* удаление узлов.
* управление фреймами,
* поддержка задержки в исполнении кода и зацикливания с задержкой,
* системные диалоги,
* управление адресом открытой страницы,
* управление информацией о браузере,
* управление информацией о параметрах монитора,
* ограниченное управление историей просмотра страниц,
* поддержка работы с HTTP cookie.

Для добавления JavaScript-кода на страницу, можно использовать теги <script></script>.

# Canvas

Canvas — элемент HTML5, предназначенный для создания растрового двухмерного изображения при помощи JavaScript. Начало отсчёта блока находится слева сверху. От него и строится каждый элемент блока. Размер пространства координат не обязательно отражает размер фактической отображаемой площади.

Впервые элемент canvas был представлен компанией Apple в движке WebKit для Mac OS X с целью последующего его использования в приложениях Dashboard и Safari.

На сегодняшний день canvas чаще используется для построения графиков, простой анимации и игр в браузерах. Группа WHATWG предлагает использовать canvas как стандарт для создания графики в новых поколениях веб-приложений.

Организация Mozilla Foundation ведёт проект под названием Canvas 3D, целью которого является добавить низкоуровневую поддержку графических ускорителей для отображения трёхмерных изображений через HTML-элемент canvas. Наряду с этим существуют библиотеки, реализующие работу с трёхмерными моделями.

Особенности:

* Изменение высоты или ширины холста сотрет всё его содержимое и все настройки, проще говоря он создастся заново;
* Начало отсчёта (точка 0,0) находится в левом верхнем углу. Но её можно сдвигать;
* 3D контекста нет, есть отдельные разработки, но они не стандартизованы;
* Цвет текста можно указывать аналогично CSS, впрочем, как и размер шрифта.

Критика:

* Чрезмерно нагружает процессор и оперативную память;
* Из-за ограничения сборщика мусора нет возможности очистить память;
* Необходимо самому обрабатывать события с объектами;
* Плохая производительность при высоком разрешении;
* Приходится отрисовывать отдельно каждый элемент.
* Возможность создания на страницах специальныx «маячков», т.н. Canvas Fingerprinting, для отслеживания пользователей в сети.

Преимущества:

* В отличие от SVG гораздо удобнее иметь дело с большим числом элементов;
* Имеет аппаратное ускорение;
* Можно манипулировать каждым пикселем;
* Можно применять фильтры обработки изображений;
* Есть много библиотек.

В Canvas реализовано использование WebGL для аппаратного ускорения 3D графики.

WebGL — программная библиотека для языка программирования JavaScript, позволяющая создавать на JavaScript интерактивную 3D-графику, функционирующую в широком спектре совместимых с ней веб-браузеров. За счёт использования низкоуровневых средств поддержки OpenGL, часть кода на WebGL может выполняться непосредственно на видеокартах.

Использование Canvas:

<!doctype html>

<html lang="ru">

<head>

<title>canvas</title>

<!--Подключение Java Script кода-->

<script src="example.js"></script>

</head>

<body>

<canvas id="canvas"> </canvas>

</body>

</html>

Файл example.js:

function onLoadHandler() {

var canvas = document.getElementById('canvas'),

context = canvas.getContext('2d');

/\*

Какие-либо действия над холстом

\*/

}

window.onload = onLoadHandler;

# Структура проекта

Проект состоит из трех файлов:

* paint.html – файл содержит HTML5 разметку страницы.
* paint.js – файл содержит Java Script код необходимый для функционирования приложения.
* paint.css – файл содержит CSS разметку документа (средство оформления внешнего вида веб-страниц).

Файл paint.html является основным, так как в нем происходит подключение вспомогательных файлов paint.js и paint.css.

# Исходный код

Файл paint.html:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<!--Заголовок страницы-->

<title>JS Paint</title>

<!--Подключение таблицы стилей paint.css-->

<link rel="stylesheet" type="text/css" href="paint.css">

</head>

<body>

<div class="paintCanvas">

<!--Холст для рисования-->

<canvas id="c1" width="800" height="600"></canvas>

</div>

<div>

<div>

Цвет:

<!--Диалог выбора цвета кисти-->

<input type="color" id="colorSelector">

</div>

<!--Кнопка для очистки холста-->

<button id="clearCanvas" onclick="clear()">Очистить</button>

<div>

Толщина линии:

<!--Выбор толщины кисти-->

<input id="lineWidthInput" type="number" name="quantity" min="1" max="20"> px

</div>

</div>

</body>

<!--Подключение Java Script кода-->

<script src="paint.js"></script>

</html>

Файл paint.js:

var canvas = document.getElementById('c1'); //ссылка на холст

var context = canvas.getContext('2d'); //ссылка на контекст холста

var clearButton = document.getElementById('clearCanvas'); //ссылка на кнопку очистки холста

var colorSelector = document.getElementById('colorSelector'); //ссылка на элемент выбора цвета кисти

var lineWidthInput = document.getElementById('lineWidthInput'); //ссылка на элемент выбора толщины линии кисти

var myColor = "#ff0000"; //начальный цвет кисти

var myLineSize = 10; //начальная толщина линии

//инициализация обработчика события нажатия кнопки очистки холста

clearButton.onclick = function () {

//очистка холста

context.clearRect(0, 0, canvas.width, canvas.height);

};

colorSelector.value = myColor; //инициализация элемента выбора цвета начальным цветом

//инициализация обработчика события выбора цвета

colorSelector.oninput = function () {

//присвоение нового цвета кисти

myColor = this.value;

}

lineWidthInput.value = myLineSize; //инициализация элемента выбора толщины линии начальным значением

//инициализация обработчика события выбора толщины линии

lineWidthInput.oninput = function () {

//обновление толщины линии

myLineSize=lineWidthInput.value;

}

//инициализация обработчика события нажатия на холст

canvas.onmousedown = function (event) {

//инициализация обработчика события передвижения мыши по холсту

canvas.onmousemove = function (event) {

var x = event.offsetX;

var y = event.offsetY;

//рисование прямоугольника с заданными координатами и цветом

context.fillStyle = myColor;

context.fillRect(x - myLineSize/2, y - myLineSize/2, myLineSize, myLineSize);

context.fill();

}

//инициализация обработчика события отпускания кнопки мыши

canvas.onmouseup = function () {

canvas.onmousemove = null;

}

}

Файл paint.css:

/\* Стиль тела страницы \*/

body {

background-color: gainsboro; /\* Цвет фона \*/

}

/\* Стиль холста \*/

.paintCanvas{

margin: 0 auto; /\* Внешний отступ элемента \*/

width: 800px; /\* Ширина элемента \*/

height: 600px; /\* Высота элемента \*/

background: white; /\* Цвет фона \*/

border: solid 1px black; /\* Толщина границы элемента \*/

}

# Пример работы с приложением

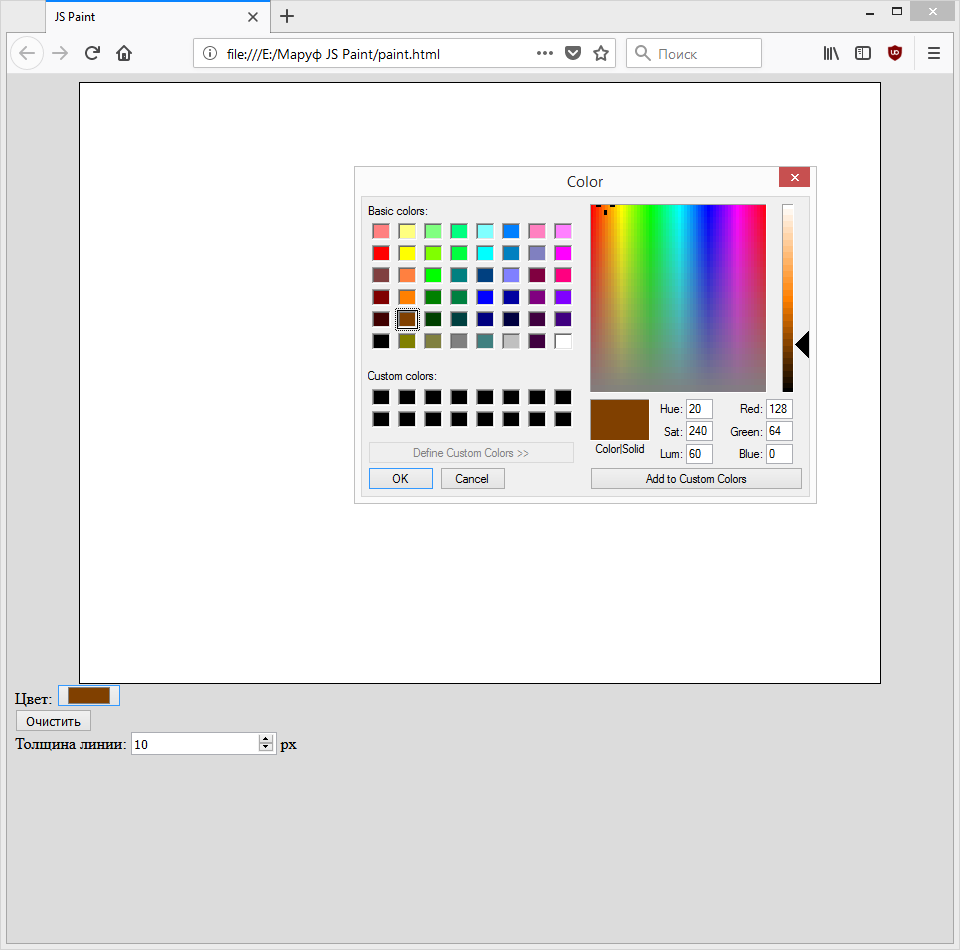


Рисунок 1. Начальное окно приложения и диалоговое окно выбора цвета

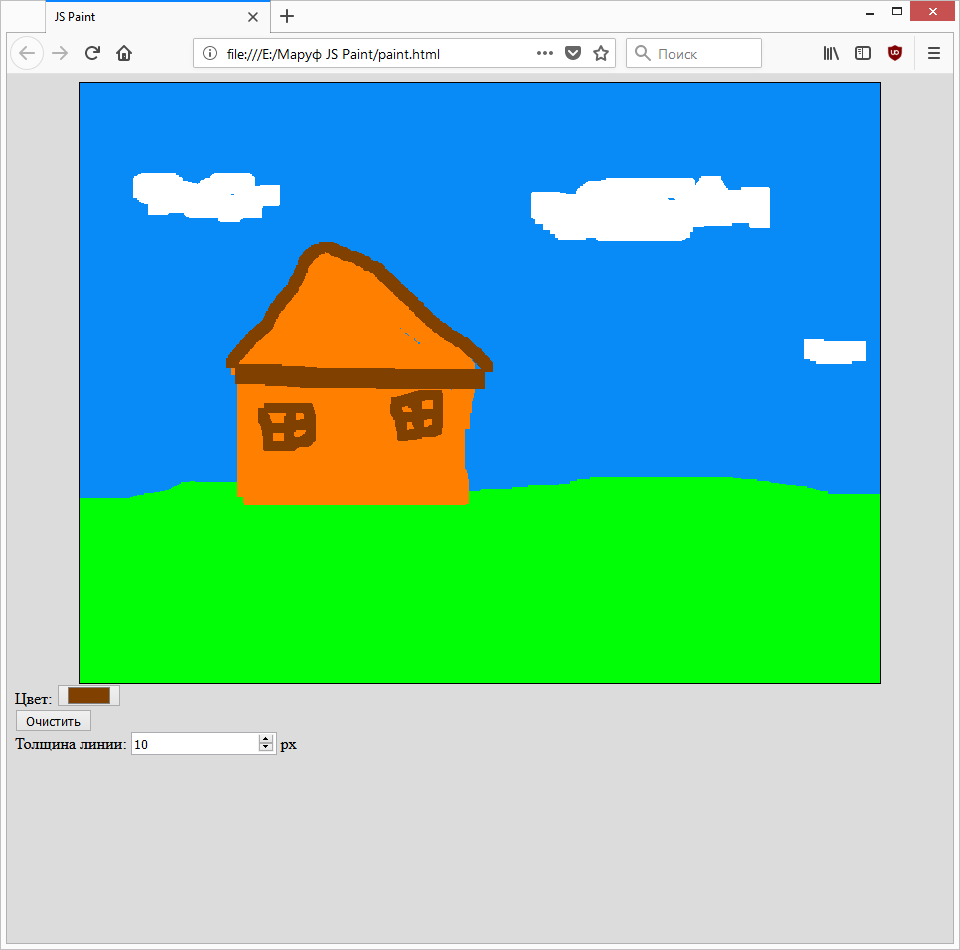


Рисунок 2. Пример использования приложения

# Список литературы

1. Дэвид Флэнаган JavaScript. Подробное руководство — М. : Символ-Плюс, 2013.
2. Брайан Хоган HTML5 и CSS3. Веб-разработка по стандартам нового поколения— СПб: Питер, 2014.
3. Руководство по Canvas [Электронный ресурс]. – Режим доступа:URL: <https://developer.mozilla.org/ru/docs/Web/API/Canvas_API/Tutorial>.