два тарифа. В разделе «Архив» доступны все выпуски журналов, а также существует возможность неавторизированному пользователю в электронном виде просмотреть текущий выпуск журнала.

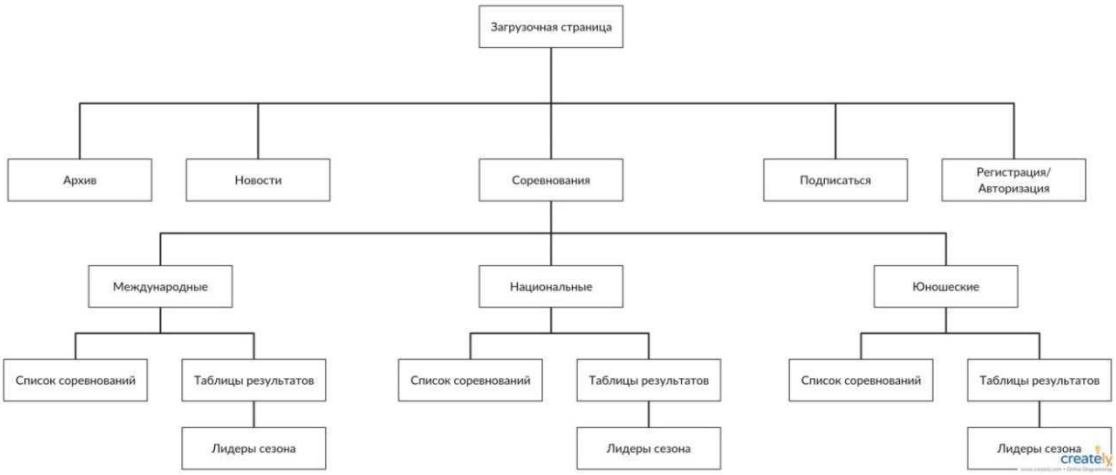

Рисунок 2 – Функциональная схема

Сам журнал необходим для дальнейшего создания и продвижения интернет-издания. Хороший журнал состоит из 40% наполнения, подразумевает тему, направленную на свою аудиторию, из 20% оформления, то есть формат, плотность бумаги и так далее, для привлечения покупателя, и из 40% дизайна, который направлен на пользователя. В нём должны быть соблюдены правила UX, для удобочитаемости, правильного расположения информации.

УДК 339.138

Студ. Е.И. Жаденова
Научн. рук. ст. преп. Н.И. Потапенко
(кафедра информатики и веб-дизайна)

ТЕХНОЛОГИЯ WEBGL В ДИЗАЙНЕ САЙТОВ

Появление в HTML5 тега canvas привело к появлению технологии WebGL для рендера 3D изображений. Создание приложений на WebGL, используя нативный JavaScript несколько затруднительно, поэтому различными компаниями были разработаны фреймворки, работающие исключительно с 3D визуализацией. Одним из таких фреймворков является three.js. Это позволило создавать в браузере 3D игры, сцены и эффекты. Пока большинство из них, но некоторые могут быть уже сегодня использоваться на живых сайтах.

Есть мнение, что WebGL используется как API для 3D. На самом деле WebGL – это средство растеризации. Он отображает точки, линии и треугольники на основе написанного кода.

WebGL позволяет веб-контенту использовать API, основанный на OpenGL ES 2.0 для визуализации трехмерной графики без использования элемента canvas в браузерах. WebGL программы состоят из кода управления, написанного на JavaScript, и кода специальных эффектов (шейдерного кода), который выполняется на графическом процессоре. WebGL элементы могут быть смешаны с другими HTML элементами и собраны с другими частями веб-страницы или фоном веб-страницы [1].

Шейдерный код представлен в виде пары функций. Эти две функции – вершинный и фрагментный шейдер, обе они написаны на очень строго типизированном языке, подобном C/C++, который называется GLSL. (GL Shader Language). Вместе эта пара функций называется *программа.*Задача вершинного шейдера – вычислять положения вершин. Основываясь на положениях вершин, которые возвращает функция, WebGL может растеризовать различные примитивы, включая точки, линии или треугольники. В процессе растеризации этих примитивов WebGL прибегает к использованию второй функции – фрагментному шейдеру. Задача фрагментного шейдера – вычислять цвет для каждого пикселя примитива, который в данный момент отрисовывается [2].

Практически всё API WebGL заключается в настройке состояния для работы этих двух функций. Любые данные, которые будут использованы в этих двух функциях, должны быть переданы на графический процессор. Есть четыре способа, которыми шейдер может получить данные:

– атрибуты и буферы. Буферы – это массивы бинарных данных, загруженных в графический процессор. Обычно буферы содержат положения вершин, нормалей, координаты текстур, цветов вершин и т.д., хотя вы вольны положить в них что угодно. Атрибуты определяют, каким образом данные из буферов передаются в вершинный шейдер. Доступ к буферам не произвольный. Вместо этого вершинный шейдер выполняется заданное количество раз и каждый раз, когда он выполняется, выбирается следующее значение каждого из указанных буферов и назначается атрибуту;

– uniform-переменные. Uniform-переменные – это глобальные переменные, которые устанавливаются перед выполнением программы шейдера;

– текстуры. Текстуры – это массивы данных, к которым есть произвольный доступ в программе шейдера. Чаще всего в текстуру помещается картинка, но текстура – это просто набор данных можно запросто поместить в неё что-то отличное от набора цветов;

– varying-переменные. Varying-переменные позволяют передавать данные из вершинного шейдера фрагментному шейдеру. Во фрагментном шейдере получаются интерполированные значения вершинного шейдера – зависит от того, отображаются ли точки, линии или треугольники [3–5].

Three.js – это библиотека, которая опирается на возможности WebGL и делает над ним надстройку, что позволяет работать с WebGL более удобным образом и писать меньше строчек кода.

Использование библиотеки three.js, а не чистого WebGL, даёт быстрый результат и удобство. Нарпимер, сделать первую сцену, вывести 3D модель и сразу иметь возможность получить результат.

Three.js выполняет всю низкоуровневую работу с WebGL, предоставляя набор классов с понятным интерфейсом для удобной работы с 3D графикой, и кроме этого за счёт скрытия сложных механизмов отрисовки 3D объектов значительно упрощает разработку [6].

Рассмотрим, что представляет 3D изображение в терминах библиотеки three.js.

Основой 3D пространства является рендерер. Рендерер – это сущность, отображающая на canvas выбранную сцену с выбранной камерой. Сцена – это 3D пространство, в котором располагаются нужные объекты: элементы (меши) и источники света, которые освещают элементы с какой-либо стороны. Рендерер, чтобы сделать отображение чего-либо на canvas, всегда принимает в себя сцену и камеру. Именно эти две ключевые сущности формируют отображение пространства на полотно canvas.

Источник света – элемент, который создаёт освещение. Он может быть рассеянным, точечным, направленным.

Следующая важная сущность – это меш. Меш – это элемент сцены, который состоит из геометрии и материала. Поскольку любой 3D объект является сложным.

Геометрия – это набор вершин, которые при генерации соединяются между собой графическими примитивами. Если взять простейшую плоскость как геометрию, то в рамках этой плоскости точки при отображении соединяются между собой прямыми линиями, и виден результат.

Материал – понятие намного более сложное, чем цвет. Материал – это способ отображения и внешний вид элемента. Какие-то элементы отражают свет и/или отбрасывают тени, а какие-то нет. Материал ведёт себя по-разному в рамках представления на сцене. Например, тени и отражение.

Текстура – это изображение, которое может использоваться в рамках материала, чтобы задать внешний вид объекта.

На рисунке 1 представлена взаимосвязь сущностей рендерера.

Рисунок 1 – Взаимосвязь сущностей рендерера

Подводя итог вышерассмотренному, можно сделать вывод, что WebGL – технология, позволяющая создавать запоминающиеся интерактивные веб-приложения и объёмные визуализации данных. Однако, несмотря на то, что большинство браузеров поддерживает эту технологию, не так много ещё разработчиков её используют. Технология WebGL со временем займёт достойное место в арсенале веб-разработчиков.

ЛИТЕРАТУРА

1. Подготовка к использованию WebGL / Frontender Magazine [Электронный ресурс]. – 2013. – Режим доступа: https://frontender.info/Getting_started_with_WebGL/. – Дата доступа: 19.03.2019.

2. WebGL tutorial / Mozilla [Электронный ресурс]. – 2019. – Режим доступа: https://developer.mozilla.org/ru/docs/Web/API/WebGL_API/Tutorial. – Дата доступа: 01.04.2019.

3.WebGL Fundamentals / WebGLFundamentals [Электронный ресурс]. – 2017. – Режим доступа: https://webglfundamentals.org/webgl/lessons/ru/webgl-fundamentals.html. – Дата доступа: 10.03.2019.

4. Основы WebGL / DevBurn [Электронный ресурс]. – 2019. – Режим доступа: https://devburn.ru/2017/04/22/. – Дата доступа: 27.03.2019.

5. Learn WebGL / LearnWebGL [Электронный ресурс]. – 2018. – Режим доступа: http://learnwebgl.brown37.net/. – Дата доступа: 25.03.2019.

6. Three.js for beginers / Medium [Электронный ресурс]. – 2019. – Режим доступа: https://medium.com/@PavelLaptev/three-js-for-beginers-32ce451aabda. – Дата доступа: 10.03.2019.