

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ рОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Дальневосточный федеральный университет»**

(ДВФУ)

|  |
| --- |
| **ИНСТИТУ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**  **Департамент математического и компьютерного моделирования** |
|  |
| **Освоение библиотеки three.js** |
|  |

**КУРСОВАЯ РАБОТА**

по образовательной программе подготовки бакалавров

по направлению 09.03.03 «Прикладная информатика»

|  |  |
| --- | --- |
| Работа защищена  с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Регистрационный номер \_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022г. | Студент группы № Б9121-09.03.03пикд  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Чурганов Н.С.  (подпись)  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022г.  Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (должность, ученое звание)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) (ФИО)  «\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022г. |

г. Владивосток

2022

**Содержание**

[Введение 3](#_Toc107270461)

[Проект 4](#_Toc107270462)

[Реализация проекта 5](#_Toc107270463)

[Создание шаблона 5](#_Toc107270464)

[Импортирование файлов 5](#_Toc107270465)

[Создание основной функции 6](#_Toc107270466)

[Создание сцены, камеры, визуализатора 6](#_Toc107270467)

[Добавление модели и контролера камеры 7](#_Toc107270468)

[Создание функции анимации 7](#_Toc107270469)

[Добавление источников света 8](#_Toc107270470)

[Заключение 10](#_Toc107270471)

[Список литературы 11](#_Toc107270472)

# Введение

Предметная область – решение проблемы с помещением 3D-модели на веб-сайт.

Данная работа достаточно актуальна, так как веб-сайты начали переходить на использование 3D – моделей. К примеру, в интернет-магазинах есть   
возможность осмотреть товар со всех сторон.

Цели: поместить 3D – модель на сайт, осуществить взаимодействия с ней.

Задачи:

1. Ознакомиться с библиотекой three.js;
2. Найти 3D – модель в формате gltf.

# Проект

Рассмотрим два варианта, с помощью которых можно поместить   
3D – модель. Первый вариант – это поместить последовательные   
фотографии, при взаимодействии с которыми кадру будут последовательно сменяться. Таким образом, способ с использованием фотографий будет неудобен с точки зрения использования, так как картинка будет не плавная и прерывистая. Следовательно, такой вариант нам не подходит.

Второй вариант – это использование библиотеки three.js и 3D – модели в формате gltf. Данный способ заключается в том, что мы создаем сцену,  
в которую помещается модель, и с помощью скриптов можно  
взаимодействовать с объектов, то есть передвигать, приближать и вращать объект.

Для начала мы должны составить план работы, которому мы будем следовать в течение

1. Поиск информации, которая поможет ознакомиться с библиотекой;
2. Загрузка и установка программы Visual Studio Code;
3. Загрузка архива с файлами для подключения библиотеки three.js;
4. Загрузка 3D – модели;
5. Написание программы на языке html с использованием three.js для  
   реализации нашей задач.

# Реализация проекта

## Создание шаблона

Подготовив компьютер к работе, мы должны создать файл в формате html, при запуске которого мы будем проверять работоспособность нашего кода.

Изображение выглядит как текст

Автоматически созданное описаниеПосле создания файла мы запускаем программу Visual Studio Code и   
открываем наш файл. Для того, чтобы приступить к написанию кода надо создать шаблон для языка html, написав в перовой строке *восклицательный знак*, а затем нажать кнопку “Tab”. Таким образом у нас создаться шаблон (рис. 1).

Рисунок 1. Шаблон HTML

## Импортирование файлов

Изображение выглядит как текст

Автоматически созданное описаниеПосле того, как мы выполнили предыдущие действия, можно приступить к написанию кода. Для начала нам нужно подключить библиотеку three.js и все нужные для работы файлы. В нашем случаи такими файлами будут являться:  
Orbitcontrols.js, GLTFLoader.js. OrbitControls.js будет отвечать за взаимодействия с камерой, а GLTFLoader.js – за загрузки 3D – модели (рис. 2).

Рисунок 2. Импортирование файлов

## Создание основной функции

После успешного импортирования файлов нам нужно написать функцию, которая будет выполняться при запуске программы. Данная функция будет иметь название “init”, в ней будет создаваться сцена, в которую мы добавим свет, объект и задний фон. Так же внутри функции будут создаваться камера,  
отслеживание взаимодействий с камерой, обработка изображения и функция “animate”, которая будет анимировать всю нашу сцену.

## Создание сцены, камеры, визуализатора

Изображение выглядит как текст

Автоматически созданное описаниеДля создания сцены нам нужно завести переменную(scene), которая будет содержать объект – Scene из библиотеки three.js. Для того, чтобы задать цвет фону нам надо обратиться к “scene”, указать, что мы хотим работать с задним фоном и создаем новый цвет (рис. 3).

Рисунок 3. Создание сцены

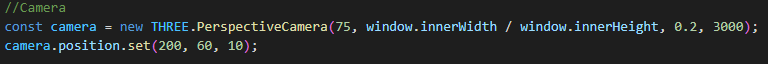
 Далее нам нужно создать камеру, указав ее координаты и области  
видимости (рис. 4).

Рисунок 4. Постановка камеры

Изображение выглядит как текст

Автоматически созданное описаниеКогда у нас есть сцена, которая содержит задний фон и камеру, мы можем создать визуализатор. Для него нужно создать новую переменную (renderer), которая будет обрабатывать содержимое файла (рис. 5).

Рисунок 5. Создания визуализатора

## Добавление модели и контролера камеры

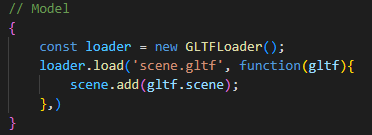
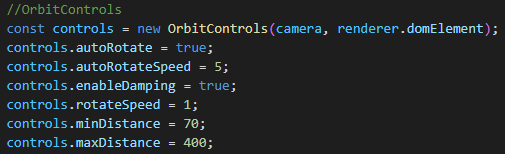
На данном этапе мы уже можем посмотреть, как выглядит наша сцена в браузере, но ничего кроме серого фона мы не увидим. Поэтому нам надо  
добавить хоть какую-то модель. Я взял модель телефона на бесплатном сайте  
3D-моделями и загрузил ее в формате gltf. Для того, чтобы добавить модель в сцену нам надо создать переменную, которая будет загружать файл с   
компьютера (рис. 6). Так же сразу добавим OrbitControls, чтобы можно было   
взаимодействовать с камерой. Для этого нужно создать переменную, которая   
будет выполнять роль контролера. Помимо обычного взаимодействия, был добавлен авто поворот, минимальное и максимальное расстояние до объекта, была изменена скорость вращения камеры (рис. 7)

Рисунок 7. OrbitControls

Рисунок 6. Добавление модели.

## Создание функции анимации

Изображение выглядит как текст

Автоматически созданное описаниеДля того, чтобы наши анимации работали надо создать функцию, которая будет отвечать за это. Данная анимация будет содержать метод requestAnimationFrame, которая сообщает браузеру, что нужно выполнить анимацию, и запрашивает, чтобы браузер вызвал указанную функцию для обновления анимации перед   
следующей перерисовкой. Также функция содержит обновление контролера и визуализатор, который обрабатывает сцену и камеру (рис. 8).

Рисунок 8. Функция Animate

## Добавление источников света

Если на данном этапе сделано все правильно, то можно будет увидеть  
черную модель, так как у нас нет света (рис. 9).

Изображение выглядит как знак

Автоматически созданное описание Для того, чтобы на модели увидеть детали нам надо добавить свет. Для начала создадим переменную, который будет называться “ambient” и выполнять роль основного света. Присвоим переменной значение, обратимся к библиотеке three.js и используем метод AmbientLight, который содержится в ней. Затем  
добавим переменную ambient в сцену (рис. 10).

Рисунок 9. Промежуточный результат без источников света

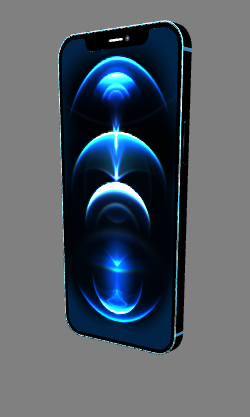
Изображение выглядит как текст

Автоматически созданное описание Теперь мы видим, что появились некоторые детали, но модель все равно имеет недостаточное освещение (рис. 11).

Рисунок 10. AmbientLight

 Добавим еще несколько источников света, чтобы все детали были видны. Второстепенные источники будут создаваться по подобию основного, но будет использоваться другой метод, DirectionalLight, т. к. у DirectionalLight можно   
изменить позицию, у AmbientLight нельзя. В нашем случаи было добавлено три дополнительных источника света (рис. 12), т. к. с двумя источниками модель освещена недостаточно хорошо, а с четырьмя и более искажается цвет модели, чем больше источников, тем светлее модель. То есть если мы добавим много  
источников, то у нас модель будет белая. Поэтому три второстепенных  
источника света подходят нам больше всего (рис. 13).

Рисунок 11. Модель с одним источником света

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 13. Результат с дополнительными источниками света

Рисунок 12. Второстепенные источники свет

# Заключение

По результату проведенной нами работы можно сказать, что поместить   
3-модель на веб-сайт с помощью библиотеки three.js не так сложно, а результат выглядит неплохо. Так как мы получаем плавную анимацию просмотра модели без потери кадров. Единственные проблемы, которые можно встретить во время написания кода – это то, что библиотека часто обновляется и некоторые методы меняются. В моем случаи была проблема с импортированием файлов, так как во многих источниках используется информация, которой более 2 лет, а библиотека за это время была обновлена несколько раз.

# Список литературы

[Three.js – JavaScript 3D Library (threejs.org)](https://threejs.org/) – библиотека three.js

[Window.requestAnimationFrame() — веб-API | МДН (mozilla.org)](https://developer.mozilla.org/en-US/docs/Web/API/window/requestAnimationFrame) – описание requestAnimationFrame

[OrbitControls – three.js docs (threejs.org)](https://threejs.org/docs/#examples/en/controls/OrbitControls) – описание OrbitControls

[AmbientLight – three.js docs (threejs.org)](https://threejs.org/docs/#api/en/lights/AmbientLight) – описание AmbientLight

[DirectionalLight – three.js docs (threejs.org)](https://threejs.org/docs/#api/en/lights/DirectionalLight) – описание DirectionalLight

[AmbientLight – three.js docs (threejs.org)](https://threejs.org/docs/#api/en/lights/AmbientLight) – описание AmbientLight

[Popular 3D models - Sketchfab](https://sketchfab.com/3d-models/popular) – сайт с 3D-моделями