**ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ** **към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**ДИПЛОМНА РАБОТА**

Тема: Конфигуратор на продукти

Дипломант: Научен ръководител:

*Костадин Зотев Кирил Митов*

СОФИЯ

2021

**ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ**

**СИСТЕМИ към ТЕХНИЧЕСКИ**  **УНИВЕРСИТЕТ - СОФИЯ**

Дата на заданието: 10.11.2020 г. Утвърждавам:.............................. Дата на предаване: 10.02.2021 г. / /

**ЗАДАНИЕ**

**за дипломна работа**

на ученика Костадин Алексиев Зотев от 12 В клас

1. Тема: Конфигуратор на продукти
2. Изисквания:

Конфигураторът представлява уеб базиран редактор на 3Д сцена със следните опции:

2.1  Качване на готов 3Д модел

2.2  Местене, завъртане и reset на сцената

2.3  Промяна на светлината и цветовете на средата

2.4  Определяне на потребителски настройки на модела и части от него (цвят, текстура, размер, позиция)

2.5  Потребителска настройка за подмяна на модела

2.6  Подреждане на възможните настройки в процес

2.7  Запаметяване и зареждане след всяка стъпка от редакцията

2.8  Експортиране в html формат на конфигуратора

1. Съдържание
   1. Обзор
   2. Същинска част
   3. Приложение

Дипломант:...............................

/Костадин Зотев/

Ръководител:............................

/Кирил Митов/

Директор:.................................

/доц. д-р инж. Ст. Стефанова/

Мнение на научния ръководител

Увод:

Глава Първа

# Методи и технологии за реализиране на уеб приложения

1.1Уебстраница

Уебстраницата е документ или информационен ресурс, който е подходящ за World Wide Web. Достъпът до уебстраниците може да бъде осъществен чрез уеббраузър, а съдържанието им обикновено се показва на монитор на компютър (вкл. на екран на мобилно устройство). Тази информация обикновено се съхранява във формат от тип HTML или XHTML и може да предоставя навигация към други уебстраници чрез хипертекстови връзки. Обикновено уебстраниците включват текст, изображения (картинки, илюстрации, фотографии) и стилове във формат на файла от тип CSS, но може също да включват анимация, мултимедийно съдържание (файл с музика/звук, видеоклип с или без звук) или скриптове, написани на JavaScript, VBScript и др. скриптови програмни езици.

Уебстраниците са основно два вида:

Статична страница е тази страница, чието съдържание се

определя предварително, и се съхранява в този вид на

уеб сървъра. Динамична уебстраница е тази страница, чието

съдържание се създава в зависимост от характеристиките,

предоставени от потребителя и в зависимост от данните,

съхранявани в базата данни на сайта.

1.2 3D компютърната графика

3D компютърната графика е графика, която използва триизмерно представяне на геометрични данни, които могат да се обработват с помощта на компютърна програма. Обработката на тези данни включва извършване на различни изчисления и рендериране на двумерни изображения. Такива образи могат да бъдат съхранявани за се гледат по-късно или показват в реално време.

3D компютърната графика, често се споменават като 3D модели. Отделно от изобразяващата графика, моделът се съдържа в рамките на графичен файл с данни. Въпреки това, съществуват различия: 3D модел е математическото представяне на всеки триизмерен обект. A модел не е технически графичен докато не се показва. Моделът може да се показва визуално като двуизмерен образ чрез процес, наречен 3D рендериране или използва неграфични компютърни симулации и изчисления. С 3D печат, 3D моделите също са изобразявани в 3D физическо представяне на модела, като ограниченията на това как точно изобразяването да съответства на виртуалния модел.

1.2.1 Моделиране

Моделирането описва процесът на формирането на някаква форма от обект. Двата най – често срещани източници на 3D модели са тези, които един художник или инженер прави някаква форма на компютър с някакъв инструмент 3D моделиране и модели сканирани на компютър от предмети в реалния свят. Моделите могат да бъдат получени процедурно или чрез физическа симулация. По принцип, 3D моделът се формира от точки, наречени върхове, които определят структурата и формират полигоните. Полигонът геометрична повърхнина, формирана от най – малко три върхове (триъгълник). Четиристранен полигон се нарича четриъгълник, и полигон с повече от четири страни е п-гон. Общата целокупност на модела и неговото пригодяване за използване в анимацията зависи от структурата на полигоните.

1.2.2 Рендериране

Процесът на превръщането на една сцена в подходяща за рендериране форма включва също и [3D проекция](https://bg.wikipedia.org/w/index.php?title=3D_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%86%D0%B8%D1%8F&action=edit&redlink=1), която дава възможност триизмерния обект да бъде разглеждан в две измерения. Рендирането превръща моделът в изображение или като симулира излъчване на [светлина](https://bg.wikipedia.org/wiki/%D0%A1%D0%B2%D0%B5%D1%82%D0%BB%D0%B8%D0%BD%D0%B0), за да се получат фотореалистични изображения, или като прилага стилове от изкуството както в [не-фотореалистично рендиране](https://bg.wikipedia.org/w/index.php?title=%D0%9D%D0%B5-%D1%84%D0%BE%D1%82%D0%BE%D1%80%D0%B5%D0%B0%D0%BB%D0%B8%D1%81%D1%82%D0%B8%D1%87%D0%BD%D0%BE_%D1%80%D0%B5%D0%BD%D0%B4%D0%B8%D1%80%D0%B0%D0%BD%D0%B5&action=edit&redlink=1). Двете базови операции в реалистичното рендиране са излъчване (колко светлина се получава от едно място към друго) и разсейване (как повърхността взаимодейства със светлината). Съществуват множество алгоритми за рендериране с различна цел. Някои от тях фокусират върху реализъм, други върху работа в реално време.

<https://bg.wikipedia.org/wiki/3D_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8E%D1%82%D1%8A%D1%80%D0%BD%D0%B0_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0>

1.2.3Материали и текстури

Материалите и текстурите са свойства, които двигателят за изобразяване използва за създаване на модела. В безпристрастен двигател за изобразяване, като цикли на смесители, можете да дадете индикация за двигателя въз основа на моделни материали. Например, можете да конфигурирате как да обработвате светлина, когато тя удари повърхността.

Текстурите се използват за оцветяване на материала с помощта на цветна карта или албедо или добавяне на характеристики на повърхността, като се използва карта на неравности или нормали. Може да се използва и за промяна на самия модел във форма, като се използва карта на изместване. Въпреки факта, че той е създаден чрез изграждането на триизмерни графики, тези процеси оказват силно влияние.

<https://bul.comparisonnetwork123.com/trehmernaya-grafika-eto-chto-takoe-read-743860>

1.3Формати

1.3.1 STL

A стереолитография файл обикновено се класифицира като CAD файл се използва като стандартен стереолитография дизайн формат, в който стереолитография е известен като технология, разработена за бързи процеси прототипиране прилагани за създаването и развитието на дигиталните 3D компоненти на CAD проект. Съдържанието на тези стереолитография файлове може да се състои на дигитални 3D дизайн обекти, разположени като елементи на основния модел CAD проект за създаването на множество опции за дизайн прототип. Тези стереолитография файлове са разработени за съхранение на CAD файлове в стандартизиран формат интегрирана с характеристики, които позволяват данните да се четат и тълкуват много CAD програми. Тези стереолитография файлове са интегрирани с компилация CAD данни спецификации, използвани за формат на STL файла и определени стандарти за кодиране са въведени в съдържанието на тези стереолитография файлове за да го приберете във формат STL файл. Тези стереолитография файлове се поставят с [.stl](https://www.reviversoft.com/bg/file-extensions/stl) разширението и могат да се отварят и редактират с помощта на софтуера Dassault Systemes CATIA наред с други CAD програми.

<https://www.reviversoft.com/bg/file-extensions/stl>

1.3.2 OBJ

Разширението на OBJ файл е известен като Wavefront 3D Object File която беше разработена от вълновия фронт Technologies. Тя е файлов формат, използван за триизмерен обект, съдържащ 3D координати (полигон линии и точки), текстура карти и друга информация, обект. Той съдържа стандартен формат 3D изображение, което може да бъде изнесено и открита от няколко програми за редактиране на 3D изображения. Обект файлове могат да бъдат в ASCII формат (. Об) или двоичен формат (. Мод) обаче не съдържа цветни определения за лицата. Защото от техния формат, те са човешки разбираемо. Тя поддържа и двата полигонални обекти и свободна форма обекти. Polygonal геометрия използва точки, линии, а изложението да определя предметите, докато в свободна форма, геометрия използва криви и повърхнини. Файлове в OBJ формат могат да бъдат отворени с Autodesk Maya 2013, Blender, и MeshLab в Microsoft Windows, Mac OS и Linux платформи.

https://www.reviversoft.com/bg/file-extensions/obj?ncr=1

1.2 Езици за програмиране на уебстраница

1.2.1 HTML

HTML (съкращение от термина на английски: HyperText Markup Language, в превод „език за маркиране на хипертекст“) е основният маркиращ език за описание и дизайн на уеб страници. HTML е стандарт в Интернет, а правилата се определят от международния консорциум W3C.

В повечето случаи HTML кодът е написан в текстови файлове и се хоства на сървъри, свързани към Интернет. Тези файлове съдържат текстово съдържание с маркери – инструкции за браузъра за това как да се показва текстът.

Например <маркер> Някакъв текст. </край на маркера>

Предназначението на уеб браузърите е да могат да прочетат HTML документите и да ги превърнат в уеб страници. Браузърите не показват HTML таговете, а ги използват, за да интерпретират съдържанието на страницата.

Основното предимство на HTML е, че документите, оформени по този начин, могат да се разглеждат на различни устройства, а не само на екрана. Документът може да бъде правилно оформен и върху монитора на персонален компютър, и върху миниатюрния дисплей на пейджър или мобилен телефон.

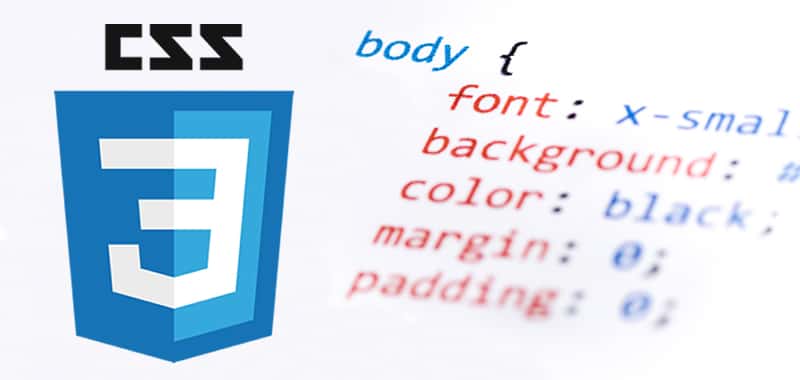
HTML може да прикрепя скриптове писани на езици като JavaScript, което променя поведението на уеб страницата. Може да се използва Cascading Style Sheets (CSS), който определя изгледа и оформлението на текста и други материали. World Wide Web Consortium (W3C) поддържа и двете CSS и HTML и насърчава използването на CSS в HTML страниците от 1997. Това допринася за разделяне съдържанието и структурата на уеб страниците от тяхното визуално представяне.



1.2.2 CSS

CSS (Cascading Style Sheets) е език за описание на стилове (език за стилови листове, style sheet language) – използва се основно за описване на представянето на документ, написан на език за маркиране. Най-често се използва заедно с HTML, но може да се приложи върху произволен XML документ. Официално спецификацията на CSS се поддържа от W3C.

CSS е създаден с цел да бъдат разделени съдържанието и структурата на уеб страниците отделно от тяхното визуално представяне. Преди стандартите за CSS, установени от W3C през 1995 г., съдържанието на сайтовете и стила на техния дизайн са писани в една и съща HTML страницата. В резултат на това HTML кода се превръща в сложен и нечетлив, а всяка промяна в проекта на даден сайт изисквала корекцията да бъде нанасяна в целия сайт страница по страница. Използвайки CSS, настройките за форматиране могат да бъдат поставени в един-единствен файл, и тогава промяната ще бъде отразена едновременно на всички страници, които използват този CSS файл.



1.2.3JavaScript

JavaScript е интерпретируем език за програмиране, разпространяван с повечето Уеб браузъри. Поддържа обектно-ориентиран и функционален стил на програмиране. Създаден е в Netscape през 1995 г. Най-често се прилага към HTML-а на Интернет страница с цел добавяне на функционалност и зареждане на данни. Може да се ползва също за писане на сървърни скриптове JSON, както и за много други приложения. JavaScript не трябва да се бърка с Java, съвпадението на имената е резултат от маркетингово решение на Netscape.

JavaScript е програмен език, който позволява динамична промяна на поведението на браузъра в рамките на дадена HTML страницата. JavaScript се зарежда, интерпретира и изпълнява от уеб браузъра, който му осигурява достъп до Обектния модел на браузъра. JavaScript функции могат да се свържат със събития на страницата (например: движение/натискане на мишката, клавиатурата или елемент от страницата, и други потребителски действия). JavaScript е най-широко разпространеният език за програмиране в Интернет. Прието е JavaScript програмите да се наричат скриптове.

1.2.4 BabylonJS

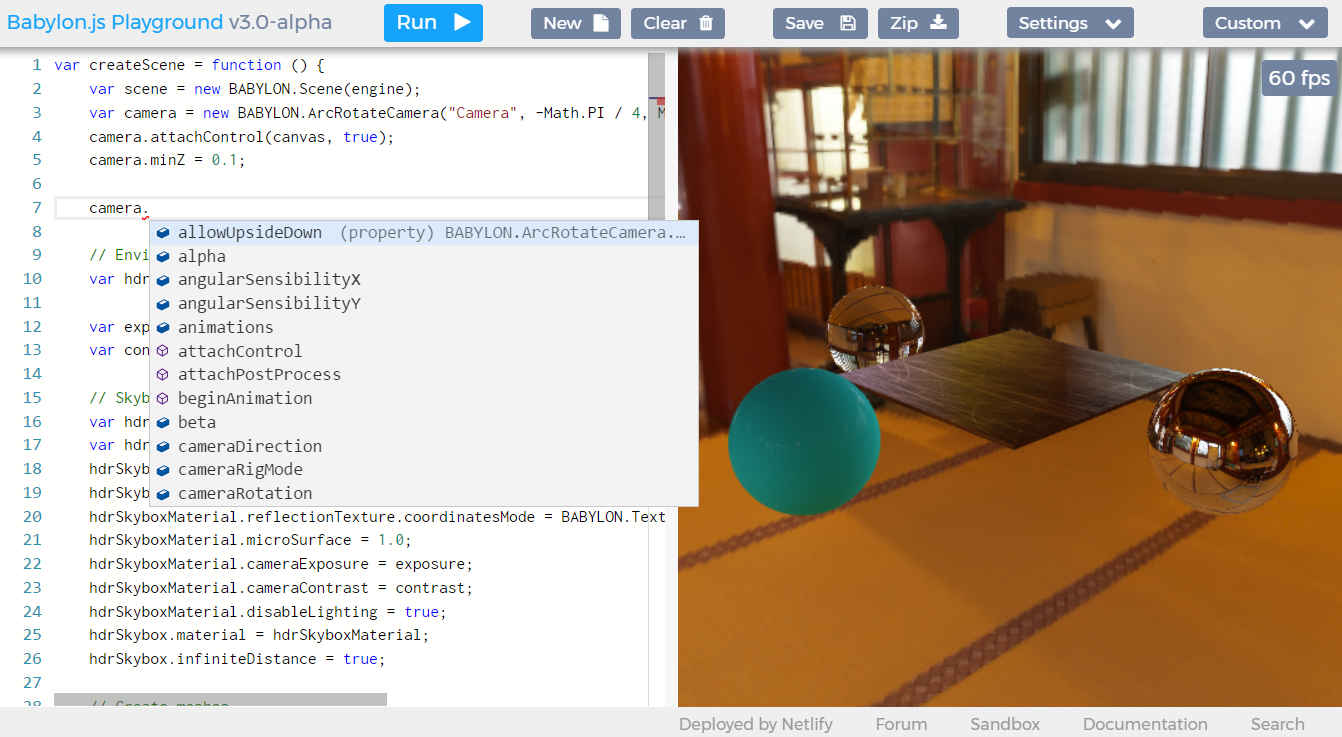
Babylon.js е javascript рамка с отворен код, която се използва за разработване на 3D приложения / видео игри за мрежата.

Осветлението е от съществено значение за реализма на сцената. Във Babylon.js сенките стават още по-добри с добавената способност да могат да изобразяват меки прозрачни сенки за прозрачни обекти. С въвеждането на PBR филтриране в реално време, усъвършенстваното пречупване и отражение вече е само на няколко реда код.

В Babylon.js е въведен Babylon Native, система за изграждане на собствени приложения със същата мощност на изобразяване и код, използвани за уеб приложения Babylon.js. Тази инициатива прави голяма стъпка напред с Babylon React Native. Този проект използва силата на рамката React Native и я комбинира със силата, красотата и простотата на Babylon.js, за да отключи способността да създава наистина невероятни изживявания на различни платформи.

Babylon.js разполага с богата библиотека от документация, за да ми помогне да започнете и да овладеете платформата. Babylon.js е с напълно реорганизирана документацията, за да насочват обучението с напредването на знанията. Документацията за Babylon.js ще бъде по-проста и полезна от всякога.

<https://www.babylonjs.com/>



1.2.5 Three.js

Повечето проекти на three.js, които виждаме, включват 3D графики в реално време, при които взаимодействието на потребителя води до незабавна визуална обратна връзка. Другият тип 3D графики са или различни ефекти, или изкуствени символи във филми, или различни „изобразявания“, които може да видите отпечатани или в уеб каталог (например уебсайтът на IKEA е пълен с 3D графики, тъй като всичките им продуктови снимки са генерирани от компютър ). Виждал съм, че това се нарича „офлайн (3d) рендиране“.

Подмножество на всичко това е 3d математика. 3D графиките не могат да се правят без математика и компютърните езици не разбират 3D концепциите по подразбиране. Тук влиза библиотека, която абстрахира тези математически операции, може би ги оптимизира и излага интерфейс на високо ниво като Matrix4 или .dot ().

Three.js идва със собствена математическа библиотека със специфични класове за 3D математика. Има самостоятелни библиотеки, които се занимават само с тази математика, но с три това е просто подмножество на много по-голяма система.

Предаването е другата голяма отговорност на библиотеката, но и тук нещата стават малко сложни. WebGL е доста специален и се откроява от тази група.

С платното, svg и css отговорността на три е чисто 3D изобразяване. Тези приложни програмни интерфейси имат много други библиотеки, за да нарисуват не 3D неща или всъщност го правят по подразбиране (css рисува 2d правоъгълници, платно различни 2d фигури), но се нуждаят от докосване на магия и 3d математика, за да направят 3D визуализация.

Докосването на магията идва най-вече под формата на абстракция на интерфейса. Например, доста е сложно да управлявате 3d състояние на <div> елемент, който е превърнат в 3d чрез CSS. Необходима е много логика, за да се превърне API на платното да нарисува нещо, което изглежда като 3d. WebGL е по-ангажиран с порядъци.

Три абстрахират всички тези API в нещо толкова просто като render (), но за да го направят, е необходимо общо представяне на това, което е „3d свят“.

<https://medium.com/@pailhead011/what-is-three-js-7a03d84d9489>

1.2.6 Webgl

WebGL (съкратено от Web Graphics Library ) е API на JavaScript за изобразяване на интерактивни 2D и 3D графики във всеки съвместим уеб браузър без използването на приставки . WebGL е напълно интегриран с други уеб стандарти , което позволява ускорено използване на GPU на физика и обработка на изображения и ефекти като част от платното на уеб страницата. WebGL елементите могат да се смесват с други HTML елементи и да се композират с други части на страницата или фона на страницата. Програмите WebGL се състоят от контролен код, написан на JavaScript и шейдър код, написан на OpenGL ES Shading Language(GLSL ES), език, подобен на C или C ++ , и се изпълнява на графичния процесор на компютъра (GPU).

1.2.6.1Помощни програми

Природата на ниското ниво на API на WebGL, която осигурява малко за бързо създаване на желана 3D графика, допринесе за създаването на библиотеки, които обикновено се използват за изграждане на нещата в 3D графика. Основни задачи като зареждане на графики на сцени и 3D обекти в популярните индустриални формати също се абстрахират от библиотеките (някои от които са пренесени в JavaScript от други езици), за да осигурят допълнителна функционалност. Неизчерпателен списък с библиотеки, които предоставят много функции на високо ниво, включва A-Frame (VR) , BabylonJS , PlayCanvas ,three.js , OSG.JS и CopperLicht .

1.2.6.2Създаване на съдържание

Както за всеки друг графичен API, създаването на съдържание за WebGL сцени изисква използването на обикновен инструмент за създаване на 3D съдържание и експортиране на сцената във формат, четим от читателя или помощната библиотека. За тази цел може да се използва настолен софтуер за 3D авторство като Blender , Autodesk Maya или SimLab Composer . По-специално, Blend4Web позволява да се създаде сцена на WebGL изцяло в Blender и да се експортира в браузър с едно щракване, дори като самостоятелна уеб страница. Съществуват и някои специфични за WebGL софтуери като CopperCube и онлайн базиран на WebGL редактор Clara.io . Онлайн платформи като Sketchfab и Clara.io позволяват на потребителите директно да качват своите 3D модели и да ги показват с помощта на хостван WebGL преглед.

1.3 Текстови редактори. Интегрирани среди за програмиране.

1.3.1 Visual Studio Code

Visual Studio Code е редактор на програмен код за Windows, Linux и OS X.[2] Това е първият редактор на Microsoft, който може да се ползва под Linux и macOS. Поддържа богат набор от инструменти за разработване като дебъгване, вграден Git Control, IntelliSense, „Side-by-Side Editing“ (позволява работа едновременно върху 2 файла отворени един до друг) и др. Той също така дава възможност за персонализиране, което означава, че потребителите могат да променят темата на редактора, клавишните комбинации, настройките и др. Все още не е известно дали потребителите ще могат да разширяват функциите на редактора чрез разработване на различни разширения, както е във Visual Studio. Редакторът е продукт на Microsoft и е безплатен, публично достъпен за преглед.

Visual Studio Code е базиран на Electron, който е базиран на Chromium, използван да разгръща io.js приложения за десктопа. Visual Studio Code използва Blink layout engine, за да направи интерфейса.

1.3.2 Notepad ++

Notepad++ е текстов редактор и редактор за изходен код за Microsoft Windows. Той поддържа редактиране в раздели, което позволява работа с много отворени файлове в един прозорец. Името на проекта идва от оператора за увеличаване с единица в езика C.

Notepad++ се разпространява като свободен софтуер. Отначало проектът се хоства в SourceForge.net, от където е изтеглен над 28 милиона пъти, и двукратно печели наградата на общността на SourceForge Community за най-добър инструмент за разработване.

Notepad++ е редактор за изходен код. Той предлага функционалност за оцветяване на синтаксиса, сгъване и разгъване на части от кода и ограничено автоматично довършване за програмни, скриптови и маркиращи езици, но не и интелигентно довършване на кода или проверка на синтаксиса. Това значи, че е в състояние да оцветява правилно кода, написан в рамките на поддържана схема, но не и да провери дали синтаксисът е логически правилен и подлежи на компилиране.

1.3.3 IntelliJ IDEA

IntelliJ IDEA е интегрирана среда за разработка (IDE) в Java за компютърен софтуер. Създадена от JetBrains (известни преди като IntelliJ), и е достъпна като Apache 2 Licensed community edition,[1] и в патентованото commercial edition. И двете могат да се използват за търговски разработки.

Community

Community версията е с отворен код и предоставена безплатно. Използването му се регулира от Apache 2.0 License и се създава заедно с отворената среда около www.jetbrains.org.

Ultimate

Ultimate версията е професионална и изцяло функционална интегрирана среда за търговска разработка, предоставена от JetBrains. Ultimate Edition е надградена версия на Community Edition. Добавени са много допълнителни функции за уеб разработки. Може да се изтегли, да се изпробва безплатно за 30 дни, след което да се закупи лиценза.

Ultimate Edition разширява Community Edition чрез предоставяне на подкрепа за много уеб рамки, допълнителни езици и технологии, и въвежда много възможности за разгръщане. Тази среда за разработване е най-логичния избор за професионални разработчици, без да е необходимо да се търсят приставки/плъгини/. Ultimate Edition предоставя множество уникални функции, като анализ на зависимостта, код за търсене на дубликати, покритие на код, интеграция с всички популярни VCS системи, и др. Благодарение на своите уникални JavaScript, HTML, CSS, JSP и други уеб езикови редактори тя дава избор на уеб разработчиците. Ultimate Edition е по-добрият избор за търговско ползване, като се има предвид, че всички функции са предоставени в пълен пакет.

Втора глава

# Изисквания към приложението. Проектиране на структурата на Конфигуратор на продукти

2.1 Функционални изисквания към конфигуратора за продукти

2.1.1 Качване на 3D модел

2.1.2 Местене на обекта

Да можем да натиснем върху 3D модела на обекта и да го теглим по дължината на подложката и

2.1.3 Завъртане на обекта

Потребителят да има възможността да завърти обекта.

2.1.4 Reset на сцената

Чрез натискане на бутона Reset потребителят ще може да се върне в първоначалната сцена, но по този начин ще изтрие напредъка му в разработката му на 3D модел.

2.1.5 Промяна на светлината

Потребителят може да избира кои от светлините да работят и по този начин да прецени, колко и кои от тях са му нужни за по-добър изглед на модела му.

2.1.6 Промяна на цветовете на средата

Потребителят решава дали иска цвета на средата да остане същият, като на първоначалната сцена или да го промени от дадените избори.

2.1.7 Смяна на цвета и текстурата на модела

Потребителят ще може да си избере по какъв начин да изглежда моделът му, като смени облика му с цвят или текстура.

2.1.8 Смяна на размера на обекта

2.1.9 Настройка за подмяна на обекта

2.1.10 Подреждане на възможните настройки в процес

2.1.11 Запазване и зареждане след всяка стъпка от редакцията

При промяна на каквото и да е в сцената, програмата трябва да запаметява всяка стъпка. По този начин ще се предотврати загубата на прогреса на потребителя.

2.1.12 Експортиране в html формат на конфигуратора

Авторът натиска Export. Това генерира конфигуратор в html документ. Този документ съдържа цялата информация за проекта по който е работил до сега в сайта.

2.2 Аргументация на избора на средите за разработка

2.2.1 Visual Studio Code

Visual Studio на Microsoft отдавна е среда за разработка само на Windows. Но през годините той се е развил, за да поддържа междуплатформена, уеб и мобилна разработка, добавяйки все повече и повече функции. За един гигант, той се движи бързо, но остава фокусиран върху изграждането на мащабни Windows приложения за клиентски компютри и за сървъри, независимо дали са локални или в облака.

През 2015 г. стартира Visual Studio Code, неговият първи инструмент за развитие на различни платформи, бързо получиха възторжени отзиви и, по-важното, бързо нарастващ брой потребители - и не само на Windows.

Изграден с помощта на платформата Electron на GitHub, Visual Studio Code е пълнофункционален редактор за разработка, който поддържа широк избор от езици и платформи, от познатите C и C # до модерни среди и езици като Go и Node.js, с паритет между Издания за Windows, MacOS и Linux.

Кодът на Visual Studio бързо се превърна в стандартна част от моята лична настройка на устройството, като замени Notepad като моят текстов редактор по подразбиране и сега той е един от първите инструменти, които инсталирам на нов компютър. Със своята поддръжка за осветяването на IntelliSense код, сега той е и моят стандартен програмен преглед на уеб съдържание, където изграждам и тествам JSON и JavaScript, за работа с микрослужби и за конфигуриране на контейнери.

<https://bul.small-business-tracker.com/get-started-with-visual-studio-code-751598>

2.2.2 HTML

Основното предимство на HTML е, че документите, оформени по този начин, могат да се разглеждат на различни устройства, а не само на екрана. Документът може да бъде правилно оформен и върху монитора на персонален компютър, и върху миниатюрния дисплей на пейджър или мобилен телефон.

HTML може да прикрепя скриптове писани на езици като JavaScript, което променя поведението на уеб страницата. Може да се използва Cascading Style Sheets (CSS), който определя изгледа и оформлението на текста и други материали. World Wide Web Consortium (W3C) поддържа и двете CSS и HTML и насърчава използването на CSS в HTML страниците от 1997. Това допринася за разделяне съдържанието и структурата на уеб страниците от тяхното визуално представяне.

https://bg.wikipedia.org/wiki/HTML

2.2.3 CSS

Благодарение на каскадата на Cascading Style Sheet вградените стилове имат най-голямо предимство или специфичност в един документ. Това означава, че те ще се прилагат, независимо какво друго е продиктувано във вашия външен стилов лист (с единственото изключение са всякакви стилове, на които е дадена! Важната декларация на този лист, но това не е нещо, което трябва да се прави в производствените сайтове, ако може да се избегне). Единствените стилове, които имат по-голямо предимство от вградените стилове, са потребителските стилове, прилагани от самите читатели. Ако имате проблеми с прилагането на промените, можете да опитате да зададете стил на вграждане на елемента. Ако стиловете ви все още не се показват, използвайки вграден стил, знаете, че се случва нещо друго.

Вградените стилове се добавят лесно и бързо и не е нужно да се притеснявате да напишете правилния CSS селектор, тъй като добавяте стиловете директно към елемента, който искате да промените (този елемент по същество замества селектора, който бихте написали във външен стилов лист ). Не е необходимо да създавате съвсем нов документ (както при външните таблици със стилове) или да редактирате нов елемент в главата на вашия документ (както при вътрешните таблици със стилове). Просто добавяте атрибута на стил, който е валиден за почти всеки HTML елемент. Всичко това са причини, поради които може да се изкушите да използвате вградени стилове, но трябва да сте наясно и с някои много важни недостатъци на този подход.

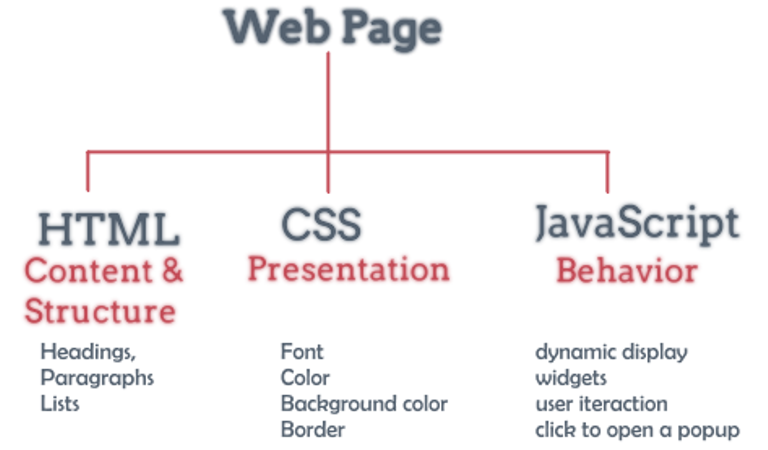
https://bul.4meahc.com/advantages-drawbacks-inline-styles-css-57355

2.2.4 JavaScript

JavaScript е език за програмиране на високо ниво, който отговаря на спецификацията ECMAScript. Това е език, който се характеризира като динамичен, слабо типизиран и многопарадигмен.

Наред с HTML и CSS, JavaScript е една от трите основни технологии на World Wide Web. JavaScript позволява интерактивни уеб страници и по този начин е съществена част от уеб приложенията. По-голямата част от уебсайтовете го използват, а всички основни уеб браузъри имат специален JavaScript механизъм, който да го изпълни.

Много често JavaScript се използва за динамично модифициране на HTML и CSS за обновяване на потребителския интерфейс чрез използването на Document Object Model API.



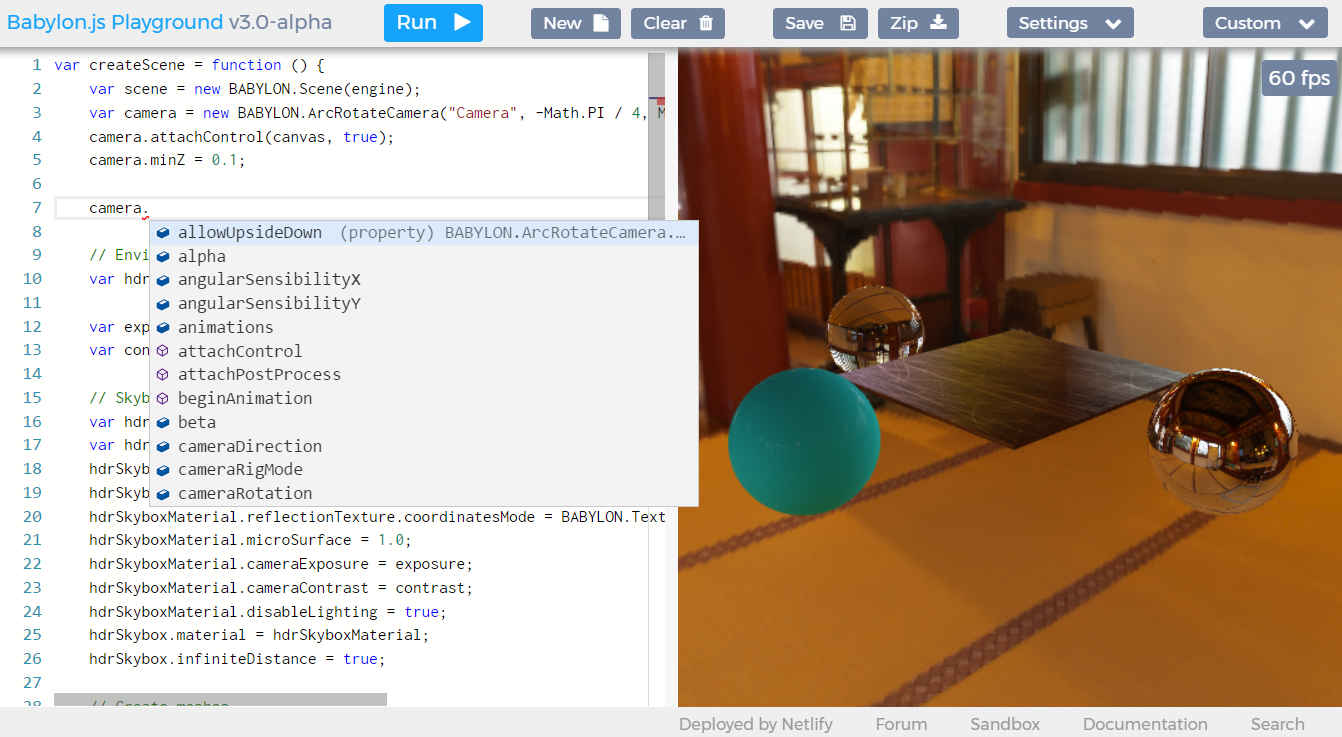
2.2.5 Babylon JS

Осветлението е от съществено значение за реализма на сцената. Във Babylon.js сенките стават още по-добри с добавената способност да могат да изобразяват меки прозрачни сенки за прозрачни обекти. С въвеждането на PBR филтриране в реално време, усъвършенстваното пречупване и отражение вече е само на няколко реда код.

В Babylon.js е въведен Babylon Native, система за изграждане на собствени приложения със същата мощност на изобразяване и код, използвани за уеб приложения Babylon.js. Тази инициатива прави голяма стъпка напред с Babylon React Native. Този проект използва силата на рамката React Native и я комбинира със силата, красотата и простотата на Babylon.js, за да отключи способността да създава наистина невероятни изживявания на различни платформи.

Babylon.js разполага с богата библиотека от документация, за да ми помогне да започнете и да овладеете платформата. Babylon.js е с напълно реорганизирана документацията, за да насочват обучението с напредването на знанията. Документацията за Babylon.js ще бъде по-проста и полезна от всякога.

<https://www.babylonjs.com/>

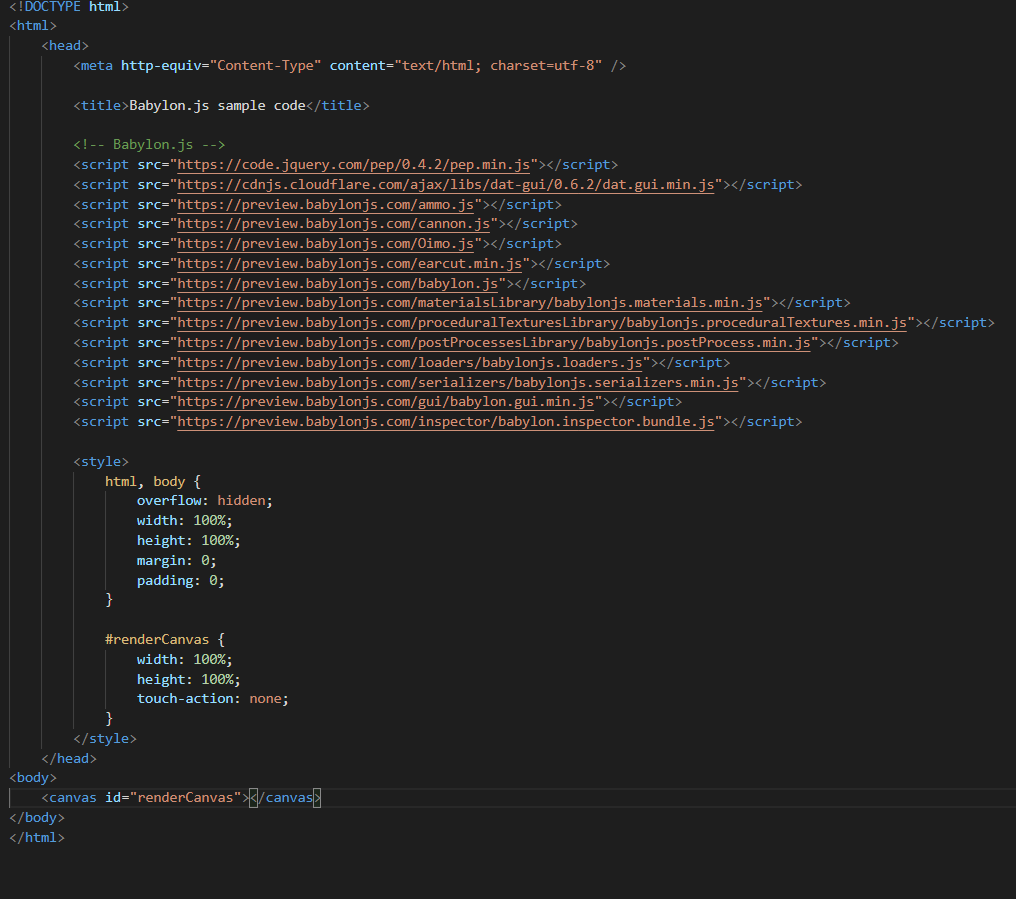


Трета глава

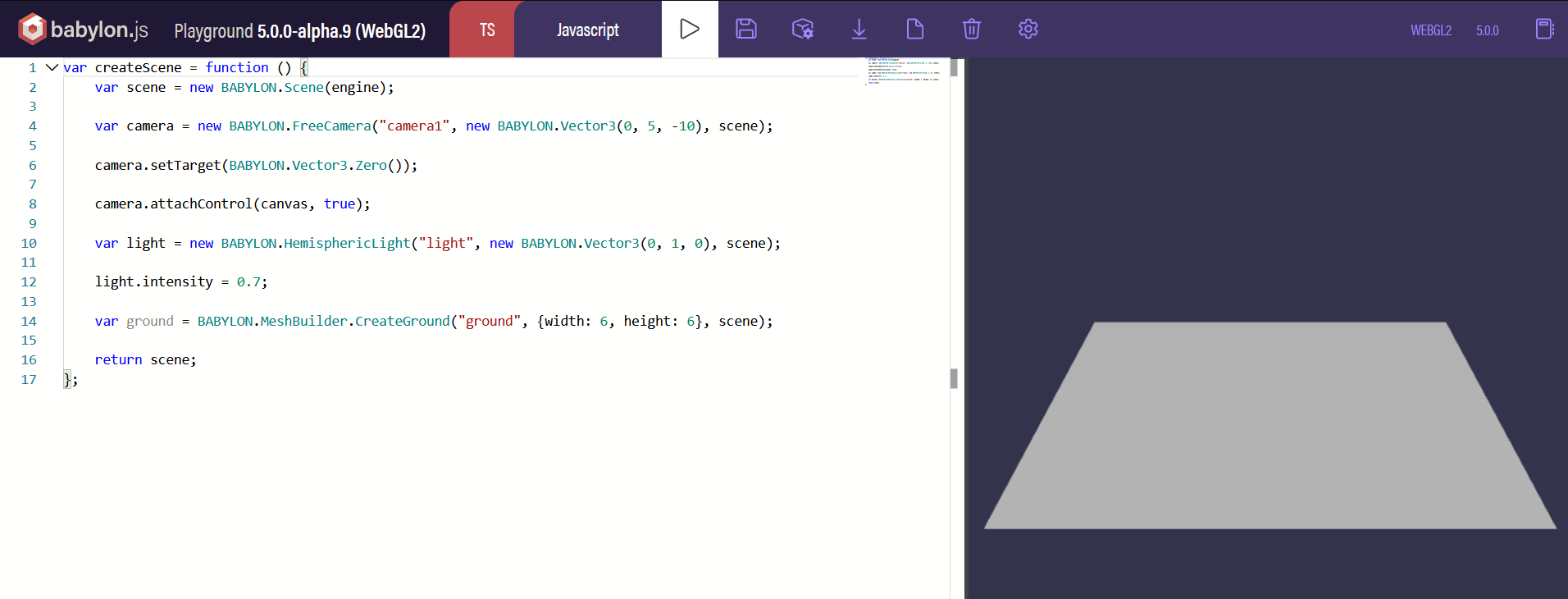
# Разработка на уеб приложението

3.1Начална сцена

В началото правим базов html файл, с включен към него CSS.



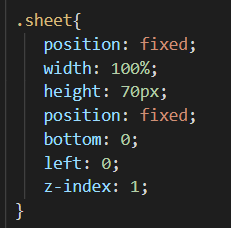
Kъм него слагаме Babylon частта, която е елементарна сцена взета от Babylon.js Playground. На нея са изобразени фона, който е едноцветен, поставка, на която ще бъде поставен самият 3D модел, камера, която е насочена към тази поставка, и единствена светлина насочена отново към поставката.



3.2 Поставяне на бутони

Първоначалната идея за направата на бутоните беше да ги сложа в Babylon.js, с помощта на специална библиотека, и те да бъдат инициализирани и самите функции, които ще изпълняват ще бъдат на едно място. Но след дълго проучване на метода за правене на бутоните, установих, че те се правят твърде сложно. И от там ми дойде идеята да го направя в html частта и функциите да си останат в java script.

Правя един div и го разполагам върху цялата повърхност на екрана. После чрез помощта на CSS го правя прозрачен и в него мога да разполагам бутоните по цялата повърхност на сайта. Чрез тази идея мога по лесно да контролирам бутоните, да променям местоположението и формата им.

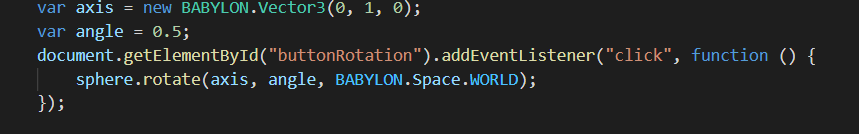


3.3 Поставяне на бутон за качване на 3D обект

Направен е бутон, при който като взимаме id-то му извикваме функцията за.

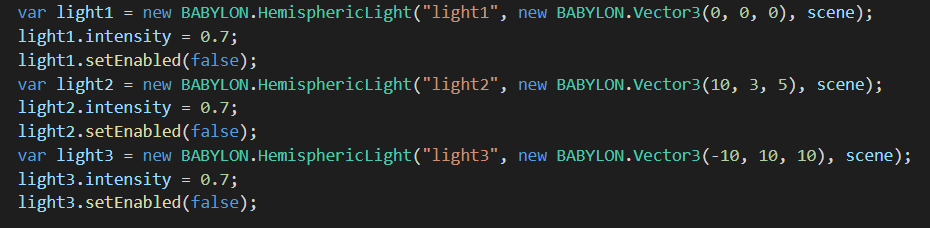
3.4 Завъртане

Първият аргумент ни показва по коя от осите ще се върти 3D обектът, след това показва с колко ще се завърти самият обект и с последния показваме, че ние се намираме в 3D свят и че има 3 оси, а не както в 2D света да има 2.



3.4 Светлини

На сцената взета от Babylon.js Playground имплементирана светлина всети през цялото време. Светлините са имплементирани всяка на различно място в пространството, за да могат да осветяват обекта от различни ъгли.



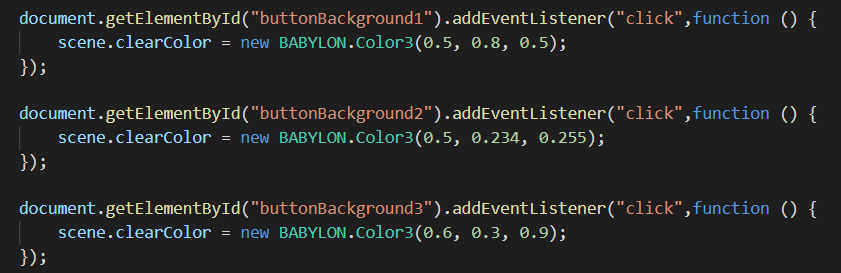
След това при натискане на бутоните, от падащото меню Lights, гледам дали променливите са натиснати за четен или нечетен път и така определям дали трябва да спра или да пусна светлината.

Направено е падащо меню, от което може да се изберат една от трите дадени опции. При повторно натискане на бутона светлината се изключва. По този начин може да се използват всичките светлини, ако потребителят иска или само дадени от него. Не може да бъде изключвана светлината, която е имплементирана от началото, защото без светлина ще се вижда черен екран.



3.5 Смяна на цвета на сцената

При натискането на бутоните, от падащото меню Background, сменяме цвета на фона чрез функцията по-долу.



3.6 Смяна на цвета на обекта

За всеки един от бутоните имплементираме материал с име и казваме че той принадлежи към сцената, която показваме на екрана. След това задавам стойности за червено, зелено и синьо и по този начин определяме какъв ще е цвета. После слагаме този цвят на дадения 3D обект.

3.7 Смяна на текстурата на обекта

За всеки един от бутоните имплементираме материал с име и втория аргумент казва, че той принадлежи към сцената, която показваме на екрана. След това задавам показвам какъв е пътят на снимката, която ще е текстурата сложена върху обекта и че принадлежи към сцената. После слагаме тази текстура на дадения 3D обект.

Четвърта глава

# Ръководство на потребителя

Светлини

Направено е падащо меню, от което може да се изберат една от трите дадени опции. При повторно натискане на бутона светлината се изключва. По този начин може да се използват всичките светлини, ако потребителят иска или само дадени от него. Не може да бъде изключвана светлината, която е имплементирана от началото, защото без светлина ще се вижда черен екран.

3.5 Смяна на цвета на сцената

От падащото меню Background могат да бъдат избрани една от трите опции за смяна на цвета на фона на сцената.

3.6 Смяна на текстурата на обекта

От падащото меню Background могат да бъдат избрани една от трите опции за смяна на цвета на фона на сцената.