НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено
Завідувач кафедри
_____ О.В.Коваль
(підпис) (ініціали, прізвище)
“___”_____2019 р.

ДИПЛОМНА РОБОТА

на здобуття ступеня бакалавра

з напряму підготовки
6.050101 “Комп’ютерні науки”

на тему: Веб-сервіс геометричних чисельних методів на базі модулю до Node.js

Виконав: студент 4 курсу, групи ТР-52

Чубаров Ігор Олександрович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник доцент, Демчишин Анатолій Анатолійович _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент _____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки 6.050101 “Комп’ютерні науки”

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.В. Коваль
(підпис)
”___”_____2019 р.

ЗАВДАННЯ
на дипломну роботу студенту
Чубарову Ігорю Олександровичу
(прізвище, ім’я, по батькові)

1. Тема роботи “Веб-сервіс геометричних чисельних методів на базі модулю до Node.js ”
керівник роботи доцент, Демчишин Анатолій Анатолійович
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)
затверджена наказом вищого навчального закладу від ”___”_____ 201__ р. № _____

2. Строк подання студентом роботи ___ _____ 201__ р.
3. Вихідні дані до роботи встановлена платформа Node.JS, алгоритмічна схема AABB
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) проаналізувати існуючі програмні рішення та можливі засоби реалізації взаємодії, обґрунтувати обрані програмні застосунки та шляхи розробки програмних додатків, розробити програмне забезпечення, розробити користувацький інтерфейс, зробити висновки за результатами роботи
5. Перелік ілюстраційного матеріалу: 1. Проблема, 2. Завдання, 3. Мета роботи, 4. Існуючі методи інтеграції C++ коду у веб-сервіс, 5. Архітектура веб-сервісу, 6. Використані програмні засоби, 7. Вхідна інформація до алгоритму AABB, 8.Інтерфейси, 9. Висновки
6. Публікації: Міжнародна науково-практична конференція молодих вчених та

студентів «Інформаційне, програмне та технічне забезпечення систем управління організаційно-технологічними комплексами».

Дата видачі завдання ”___”_____ 201__ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Вивчення та аналіз задачі		
2.	Розробка архітектури та загальної структури системи		
3.	Розробка структур окремих підсистем		
4.	Підготовка матеріалів		
5.	Програмна реалізація системи		
6.	Захист програмного продукту		
7.	Оформлення пояснювальної записки		
8.	Передзахист		
9.	Захист		

Студент _____ Чубаров І.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Демчишин А. А.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Мета роботи розробка веб-сервісу, що реалізує алгоритм геометричних чисельних методів на базі модулю до Node.js. Завдання роботи: проаналізувати існуючи методи інтеграції C++ коду у веб-сервіс, провести інтеграцію алгоритм геометричних чисельних методів у веб-сервіс, як модуль до Node.js, порівняти швидкість виконання алгоритму, інтегрованого як модуль до Node.js та алгоритму на JavaScript.

Веб-сервіс реалізовано на основі архітектури клієнт-сервер. Клієнтську частину імплементовано за допомогою HTML, CSS, JavaScript, AngularJS. Серверну частину реалізовано за допомогою JavaScript, Node.js. Обмін даними здійснюється за допомогою REST API. Алгоритмом геометричних чисельних методів обрано AABB. Алгоритм лежить в основі визначення зіткнень геометричних об’єктів.

Записка містить 67 сторінок, 14 рисунків, 5 додатків і 34 посилань.

Ключові слова: веб-сервіс, JAVASCRIPT, NODE.JS, C++, AABB, інтеграція.

ABSTRACT

Meta robots razrobka web service, real-time algorithm of geometric numerical methods on the basis of the module to Node.js. The work of the robot: to analyze the integrated methods of integration with the C ++ code on the web service, to integrate the algorithm of geometric numerical methods from the web service, as a module to Node.js;

Web service is implemented on the basis of the architecture key server. The custom part is passed for help with HTML, CSS, JavaScript, AngularJS. The server part is implemented with the help of javascript, Node.js. Obmani danimi come up for additional REST API. The algorithm of geometric numerical methods is AABB. The algorithm lies in the basis of the assignment of geometrical ob'єktiv.

Note note 67 pages, 14 drawings, 5 documents and 34 pose.

Key words: webservice, JAVASCRIPT, NODE.JS, C ++, AABB, integration.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

IDE — Integrated Development Environment.

API — Application Programming Interface.

AABB — Axis-Aligned Bounding Box

CLR — Common Language Runtime.

DLL — Dynamic Link Library.

NAN — Native Abstractions for Node.js.

ARM — Advanced RISC Machine.

MIPS — Microprocessor without Interlocked Pipeline Stages.

HTTP — HyperText Transfer Protocol.

XML — eXtensible Markup Language.

WWW — World Wide Web.

URL — Uniform Resource Locaton.

GNU — GNU's Not Unix.

ВСТУП

Код на мові програмування C++ часто використовується у бізнесі та науці. Для того щоб збільшити цикл його “життя” та збільшити актуальність необхідно інтегрувати його у веб-сервіс.

Більшість веб-розробників не знайомі з C++. Для розробки в Інтернеті використовують Ruby, Go, Node.js, Python. Проте, від того що веб-розробники не використовують C++ проблема інтеграції не зникає.

Завдання: проаналізувати існуючі методи інтеграції C++ коду у веб-сервіс, імплементувати аглоритм AABB, провести інтеграцію алгоритм геометричних чисельних методів у веб-сервіс у вигляді модуля до Node.js, порівняти швидкість виконання алгоритму, інтегрованого як модуль до Node.js та алгоритму на JavaScript.

Метою роботи є розробка веб-сервісу геометричних чисельних методів на базі модулю до Node.js. Створена система є актуальною, тому що дає змогу інтегрувати алгоритм геометричних чисельних методів, спроектований і реалізований на мові програмування C++ у веб-сервіс, а значить використовувати усі можливості, потужності та переваги алгоритму в Інтернеті.

У роботі описано повний цикл розробки веб-сервісу геометричних чисельних методів на базі модулю до Node.js.

Об’єктом роботи є інформаційні технології створення веб-сервісу типу “клієнт-сервер”.

Предметом роботи є інтеграція у веб-сервіс алгоритму геометричних чисельних методів, написаного на мові програмування C++ у вигляді модуля до Node.js.

Алгоритмом геометричних чисельних методів обрано алгоритм AABB, що розшифровується як Axis-aligned bounding box. Цей алгоритм активно використовується в програмуванні (наприклад в фізичних двигунах різних ігор) для

виявлення зіткнень або пересічний різних об'єктів між собою. Об’ктами, зіткнення яких визначає алгортитм AABB є обмежуючі корбоки. Обмежуюча коробка – це паралелепіпед зі сторонами, паралельними осям координат, що обмежує деякий геометричний об'єкт в просторі. Найчастіше за розміри паралелепіпеда береться модуль максимальної різниці проекцій на обрану вісь між двома точками. Однак замість паралелепіпеда може використовуватися куб зі стороною, що дорівнює максимальному розміру об'єкта. При такому прийомі об'єкт ніколи не вийде за межі куба, однак може загубитися точність. Найчастіше за розміри паралелепіпеда береться модуль максимальної різниці проекцій на обрану вісь між двома точками. Однак замість паралелепіпеда може використовуватися куб зі стороною, що дорівнює максимальному розміру об'єкта. При такому прийомі об'єкт ніколи не вийде за межі куба, однак може загубитися точність.

Робота має таку структуру:

У першому розділі описані існуючі способи інтеграції C++ коду з веб-сервісом.

У другому розділі описані методи реалізації програмної системи.

У третьому розділі описана архітектура веб-сервісу.

У четвертому розділі наведений опис програмної реалізації.

У п’ятому розділі описана методика роботи користувача з програмою.

1. ОГЛЯД ІСНУЮЧИХ СПОСОБІВ ІНТЕГРАЦІЇ C++ КОДУ З ВЕБ-СЕРВІСОМ

Існує три варіанти інтеграції C++ коду до веб-сервісу з Node.js [1]:

- запуск файлу з розширенням exe;
- бібліотека DLL;
- модуль до двигуна V8 [1].

Кожен з вище перелічених способів інтеграції C++ коду до веб-сервісу за допомогою Node.js описаний у цій частині роботи. Методи мають свої переваги і недоліки. Вони відрізняються між собою ступенем до якого потрібно модифікувати C++ код та продуктивністю.

Під час розробки програмного продукту використано метод, який передбачає розробку “модулю до двигуна V8”.

1.1. Запуск файлу з розширенням exe

Якщо код, написаний на мові програмування C++, виконується з командного рядка або його можна адаптувати для виконання з командного рядка – код може бути запущенним за допомогою дочірнього API процесору Node [1]. Метод має дві особливості:

- машина виконує додаток на мові програмування C++ в іншому процесі, а значить виконує асинхронно – це дає можливість оброблювати HTTP-трафік, поки виконується програма на C++;
- для того щоб використати цей метод не потрібно виконувати роботу направлену на інтеграцію однієї мови програмування в іншу.

Не має значення, на якій мові програмування написано код і що потрібно запустити – описаний в цьому розділі метод сумісний з будь-якою мовою програмування.

Метод раціонально використовувати у випадку, коли відсутня можливість редагування коду, який написаний на мові програмування C++. Метод також є дуже ресурсномістким.

1.2. Бібліотека DLL

Метод “бібліотеки DLL” раціонально обрати тоді, коли виникає потреба викликати код, написаний на мові програмування C++ за допомогою функції.

Метод “бібліотеки DLL” має велику перевагу над методом “запуску файлу з розширенням exe”. Вона полягає в тому, що в результаті використання методу “бібліотеки DLL буде отримана більш точна координація між Node.js та C++ [2]. Це пов’язано з тим, що у методі наявна можливість визначити і обрати, коли потрібно викликати будь-яку з функцій, які надає бібліотека.

Метод “бібліотеки DLL” доцільно обрати як інтеграційний вибір, якщо метод “запуску файлу з розширенням exe” потребує великої кількості ресурсів. Метод “бібліотеки DLL” також дозволяє уникнути труднощів розробки власних модулів на C++ для Node.js за допомогою V8 API.

1.3. Модуль до двигуна V8

Код, написаний на мові програмування C++, може бути інтегрованим у веб-сервіс у вигляді модулю до Node.js – додаток[3].

Додаток Node.js – це динамічно під'єднані об'єкти, які написані на мові програмування C ++. Вони можуть бути завантажені та під’єднані до Node.js за допомогою функції require() і використані так, ніби вони є звичайними модулями Node.js [4]. Додаток Node.js, в основному, використовуються для реалізації інтерфейсів між JavaScript на Node.js і бібліотеками C ++.

Для реалізації даного методу потрібні також знання інших компонентів та API.

Для того реалізації методу потрібно розібратися з поняттям двигуна JavaScript V8 (його ще називають Chrome V8) [5]. Двигун V8 – це двигун з відкритим програмним кодом. Він був розроблений The Chromium Project для веб-браузерів

Google Chrome та Chromium. Керівником проекту був Ларс Бак. Основними задачами, які необхідно було виконати команді розробників були продуктивність і масштабність двигуна. Перша версія двигуна V8 була випущена одночасно разом із першою версією веб-браузера Chrome: 2 вересня 2008 року.

Двигун V8 може виконувати команди в архітектури x86, ARM або MIPS в 32- і 64-розрядних виданнях. Крім того, він був перенесений на PowerPC і IBM s390 для використання на серверах.

Перед виконанням JavaScript коду, двигун V8 компілює його до “рідного машинного коду” (замість того, щоб інтерпретувати байт-код або компілювати код програми до машинного коду, а потім виконувати його з файлової системи). Скомпільований код є додатково оптимізованим.

Особливістю двигуна V8 також є ефективна система розподілу пам’яті. Вона дає можливість швидко резервувати місця в пам’яті для об’єктів та витрачає мало часу на прибирання сміття. Так, двигун V8:

- зупиняє виконання програмного коду під час виконання процесу прибирання сміття;
- зменшує вплив та дію зупинки додатку під час збору сміття у пам’яті;
- може точно визначити де розташовані об’єкти в пам’яті. Це дозволяє уникнути втрати пам'яті під час локальної ідентифікації об'єктів.

Двигун V8 виконує JavaScript-сценарії лише в особливих ситуаціях. Ці ситуації є віртуальними машинами. Що правда, один процес може мати одну і тільки одну віртуальну машину. І це все незважаючи на доступ до використання декількох потоків.

Двигун V8 призначений, як для використання в браузерах, так і в незалежних проектах (в цьому випадку він працює як автономний високопродуктивний двигун). Двигун V8 підтримується в такому програмному забезпеченні:

- веб-сервіс Google Chrome та всі інші веб-сервіси на основі Chromium (включаючи Brave, Opera і Vivaldi);
- база даних Couchbase;
- робоче середовище Node.js;

- робоче середовище Deno;
- програмне забезпечення Electron, базовий компонент для текстових редакторів Atom і Visual Studio Code;
- NativeScript, з відкритим вихідним кодом для створення мобільних додатків з JavaScript;
- документально-орієнтована база даних MarkLogic Server [6].

З моменту створення, двигун V8 зазнав багатьох змін. В результаті, додаток не можна використовувати в різних версіях платформи. Це пов’язано з тим, що кожен з них підтримує різний API. Проте, команда Node.js вирішує це через NAN. NAN розшифровується як “Native Abstractions for Node.js” [7]. З англійської NAN перекладається як нативні абстракції для Node.js. NAN – це бібліотека, яка не відрізняється тією ж гнучкістю, що і n-api. Вона дозволяє в абстрактному вигляді працювати з двигуном V8, але сама по собі залежить від версії релізу двигуна V8. В результаті, у нових релізах, Node.js можуть присутності зміни двигуна V8. Вони здатні порушити роботу нативних розширень.

У випадку, коли наявний доступ до редагування C++ коду, раціонально його інтеграцію до веб-сервісу реалізувати методом “модулю до двигуна V8”.

По-перше, якщо наявний C++ код уже організовано (чітко визначені точки входу та виходу/ повернення), створення додаток не є ресурсовитратним, особливо з використанням NAN.

По-друге, а досить додаток до Node.js є адаптивними: вони можуть бути блокуючими, синхронними, асинхронними. Додаток до Node.js також підтримують передачу та повернення об’єктів об'єкти.

Висновки до розділу 1

У розділі 1 проаналізовано методи інтеграції C++ коду з Node.js. Для реалізації веб-сервісу було обрано метод “модудя до двигуна V8”. Даний спосіб було обрано

тому що наявний доступ до редагування коду алгоритму на мові програмування C++ та наявний доступ до ресурсів, які даяють змогу створити додаток до Node.js

2. МЕТОДИ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ

В цьому розділі описано методи, якими була реалізована програмна система.

У процесі створення веб-сервісу дуже важливу роль відіграє вибір засобу реалізації. Були проаналізовані основні методи інтеграції C++ коду з веб-сервісом, основні помилки при проектуванні та реалізації веб-сервісу. При виборі методів програмної реалізації акцент було зроблено на швидкодію, економію часу (продуктивність роботи), зручність роботи.

Проаналізувавши основні підходи і методи досягнення мети були обрани методи реалізації програмного продукту.

2.1. Середовище розробки Visual Studio Code

Програма Visual Studio Code — редактор вихідного коду [8]. Вона дає можливість створювати, редагувати та налаштовувати програмний код. Програма Visual Studio Code підтримує, майже, всі основні мови програмування:

- JavaScript;
- JSON;
- HTML;
- CSS;
- Typescript;
- Markdown;
- Powershell;
- C++;
- Java;
- Go;
- T-SQL [9].

Деякі, з вищеперелічених мов програмування, наприклад, JavaScript, CSS і HTML доступні одразу після встановлення додатку. Для використання інших мов програмування потрібні плагіни — незалежно скомпільовані програмні модулі. Їх

можна встановити, як через користувальницький інтерфейс програми Visual Studio Code, так і за допомогою платформи VS Code Marketplace. Станом на березень 2019 року, за допомогою вбудованого в програми Visual Studio Code користувальницького інтерфейсу можна завантажити і встановити тисячі розширень з категорії «мови програмування». Плагіни також доступні для завантаження на платформі VS Code Marketplace [10]. Окрім плагінів, які потрібні для використання інших мов програмування, на платформі VS Code Marketplace також можна знайти плагіни, які роблять роботу з програмою зручнішою. Наприклад, Ident-Rainbow (розширення, яке робить відступи більш читабельними), IntelliSense for CSS class names in HTML (розширення, яке автоматично доповнює назву CSS-класу для HTML-атрибуту class на основі визначень, знайдених у робочій області) і багато іншого.

Програма Visual Studio Code має дуже зручний, комфортний та сучасний інтерфейс. Під час роботи у користувача є можливість відкрити кілька незалежних один від одного вікон і панелей. Я активно використовував цю функціональну можливість програми Visual Studio Code під час розробки веб-сервісу: в одному вікні я створював клієнтську частину, в іншому — серверну частину. Також програміст має можливість візуально розділити навпіл програмну область одного вікна. Я активно використовував цю функціональну можливість програми Visual Studio Code під час розробки клієнтської частини веб-сервісу: в одній частині екрану я працював з HTML-кодом, в іншій — SCSS. Загалом, програма Visual Studio Code може відкрити для редагування на екрані до трьох файлів одночасно, розташовуючи їх один за одним.

Програма Visual Studio Code доступна для комп’ютерів на базі ОС Windows, OS X та Linux [9]. Інтерфейс Visual Studio Code у кожній з систем ідентичний. Програміст може без проблем адаптуватись до роботи за іншим комп’ютером.

Програма Visual Studio Code має доступ до командного рядку CMD. Цей функціонал можна знайти у вкладці “Terminal”. Користувачу не потрібно переключатись між командним рядком CMD та програмою Visual Studio Code — запустити додаток можна безпосередньо з редактору коду. Окрім доступу до

командного рядка CMD, вкладка “Terminal” також дає можливість роботи з консоллю PowerShell та Linux Bash.

За замовчуванням, програма Visual Studio Code працює в режимі явного збереження. Для того щоб його виконати, потрібно, натиснути комбінацію клавіш “Ctrl + S”.Проте, можна також і увімкнути режим автоматичного збереження Auto Save. Для цього, потбріно натиснути клавіші “Ctrl + Shift + P”, у вікні. що відкрилося, потрібно набрати команду “ auto”.

Програма Visual Studio Code працює з файлами і папками, у яких знаходяться прокети. Для того щоб відкрити, наприклад, файл для редагування потрібно або запустити команду ./code index.html, або просто обрати потрібний файл у переліку “Open Editor”. Програма Visual Studio Code сама визначає тип проекту в залежності від вмісту папки. Наприклад, якщо в папці знаходяться файли package.json, project.json, tsconfig.json, то програма Visual Studio Code включає багато нових функцій, які забезпечують IntelliSence, підказки, навігацію по коду, і багато іншого. Для того щоб вивести підкази IntelliSence, потрібно просто натиснути на “Ctrl + Space”. При наборі коду редактор буде показувати його автоматично.

Програма Visual Studio Code підтримує абревіатури Emmet. Програміст може використовувати їх при редагуванні файлів HTML, Razor, CSS, Less, Sass, XML або Jade.

Програма Visual Studio Code має велику кількість налаштувань зовнішнього вигляду редактора коду. Користувач може змінити тему оформлення програми, розмір вікон, що відкриваються, видимість панелі інструментів, розмір та гарнітуру шрифтів.

Програма Visual Studio Code — безкоштовна середа розробки.

2.2. Технології для розробки клієнтської частини

Для розробки клієнтської частини використано технології:

- HTML V5;
- CSS;
- JavaScript;

- AngularJS.

WWW, W3 або Web – взаємопов'язана системою публічних веб-сторінок, доступних через Інтернет. Веб не те ж саме що і Інтернет: Веб одним з багатьох додатків, побудованих поверх Інтернету [12].

Система "Веб", складається з таких компонентів:

- протокол HTTP. Він керує передачею даних між сервером і клієнтом;
- URL – унікальний універсальний ідентифікатор. Він необхідний щоб клієнт отримав доступ до веб-компонента;
- HTML – мова гіпертекстової розмітки. Вона є найбільш поширеним форматом для публікації веб-документів [13].

2.2.1. Псевдомова HTML

HTML розшифровується як Hyper Text Markup Language [14]. Українською це перекладається як “мова розмітки гіпертекстових документів”.

HTML використовується для розробки веб-сервісів [14].

Веб-браузери отримують HTML-документи з веб-сервера або з локальної пам’яті і передають документи в мультимедійні веб-сторінки.

Елементи HTML – це будівельні блоки html-сторінок. За допомогою html-конструкцій, зображення та будь-які інші об'єкти, як, наприклад, інтерактивні форми, можуть бути вбудованими у візуалізовану сторінку. HTML надає засоби для створення структурованих документів, позначаючи структурну семантику тексту, наприклад заголовки, абзаци, списки, посилання, цитати та інші елементи. Елементи HTML називаються тегами – це ключові слова написані з використанням кутових дужок. Наприклад, такі теги, як <img /> і <input /> безпосередньо вводять до змісту сторінки Інші теги, такі як <p>, оточують і надають інформацію про текст документа, вони можуть включати в себе як інші теги, так і інші суб-елементи. Браузери не показують теги HTML, але використовують їх для інтерпретації вмісту сторінки [15].

У роботі використано мову розмітки гіпертекстових документів HTML5, тобто п’ятої версії. HTML5 був рекомендований до використання у 2014, проте він підтримувався браузерами ще з 2013 року.

HTML5 не є наступною версією HTML, HTML5 – це абсолютно нова відкрита платформа. Вона призначена для створення веб-сервісів, які використовують аудіо, відео, графіку та анімації. Мета створення HTML5 – покращення рівня підтримки мультимедійних технологій з одночасним збереженням зворотньої сумісності [17].

У HTML5 є нові, порівняно з попередніми версіями, елементи та атрибути. Вони відображають типове використання користувачем сучасних веб-сервісів [16].

До html-документу можна підключати CSS або JavaScript файли. Така взаємодія впливає на поведінку та вміст веб-сторінок. Включення CSS визначає вигляд і компонування вмісту. Включення JavaScript визначає сценарії поведінки сторінки.

2.2.2. Псевдомова CSS

CSS розшифровується як Cascading Style Sheets [18]. Українською це перекладається як “каскадні таблиці стилей”. CSS використовується для розробки веб-сервісів, це мова, яка описує зовнішній вигляд документа, який був написаний на мові розмітки гіпертекстових документів – HTML.

Стиль в CSS вказує веб-браузеру, як треба форматувати елементи. Форматування може включати установку кольору або фону елемента, тип шрифту і так далі.

Визначення стилю складається з двох частин:

- селектор – вказує на елемент;
- блок оголошення стилю - набір команд, які встановлюють правила форматування.

Багато html-елементів дозволяють встановлювати стилі відображення за допомогою атрибутів. Наприклад, для ряду елементів ми можемо застосовувати атрибути width і height для установки ширини і висоти елемента відповідно. Однак подібного підходу слід уникати і замість вбудованих атрибутів слід застосовувати

стилі CSS. Важливо, що розмітка HTML повинна надавати тільки структуру html-документа, а весь його зовнішній вигляд, стилізацію повинні визначати стилі CSS.

CSS розроблено, щоб розділити код, який стосується розмітки сторінки та код, який стосується зовнішнього оформлення сторінки. Цей поділ дає можливість поліпшити читабельність коду, забезпечує гнучкість і керованість специфікацій характеристик. Каскадні таблиці стилей також дають можливість декільком веб-сторінкам обмінюватися форматуванням, вказавши відповідний CSS в окремому файлі .css. Це зменшує складність і кількість повторів структурного змісті. Поділ зовнішнього оформлення та розмітки сторінки також дає можливість представити одну сторінки розмітки в різних стилях для різних екранів. Одна й та ж сама сторінка HTML може виглядати по-різному – все залежить від змісту CSS файлу, що підключено [19].

Для визначення правильності і коректності стилей використовуєтья css-валідатором.

У своїй роботі я використовую CSS3. Це найбільш масштабна редакція каскадних таблиць стилів в порівнянні з CSS1, CSS2 та CSS2.1. Головною особливістю CSS3 є можливість створювати анімовані елементи без використання JavaScript, підтримка лінійних і радіальних градієнтів, тіней, згладжування.

CSS3 переважно використовується як засіб опису та оформлення зовнішнього вигляду веб-сторінок, написаних за допомогою мов розмітки HTML і XHTML, але може також застосовуватися до будь-яких XML-документах, наприклад, до SVG або XUL.

На відміну від попередніх версій специфікація розбита на модулі, розробка і розвиток яких йде незалежно. CSS3 заснований на CSS2.1, доповнює існуючі властивості і значення і додає нові.

2.2.3. Мова JavaScript

JavaScript – мова програмування, яка призначена для створення сценаріїв для веб-сервісів. JavaScript може оновлювати і змінювати як HTML, так і CSS [20].

Мова програмування JavaScript може бути використана задля:

- створення сценаріїв веб-сервісів;
- створення односторінкових веб-сервісів, для цього можуть бути використані фреймворки AngularJS, React (зокрема, AngularJS я використовую у своїй роботі);
- створення серверів, для цього може бути використаний фреймворк Node.js ;
- створення мобільних додатків, для цього можуть бути використані фреймворки React Native, Cordova [21].

Проте, найбільш широке застосування мова програмування JavaScript знаходить у веб-браузерах як мова сценаріїв для додання інтерактивності веб-сторінкам.

Основні архітектурні риси мови програмування JavaScript:

- динамічна типізація;
- слабка типізація;
- автоматичне керування пам'яттю;
- прототипне програмування;
- функції як об'єкти першого класу.

Синтаксис мови JavaScript багато в чому нагадує синтаксис C і Java, семантично ж мова набагато ближче до Self, Smalltalk. У мові програмування JavaScript:

- всі ідентифікатори чутливі до регістру;
- в назвах змінних можна використовувати букви, підкреслення, символ долара, арабські цифри;
- назви змінних не можуть починатися з цифри;
- для оформлення однорядкових коментарів використовуються «//», багаторядкові і внутрістрочние коментарі починаються з «/ *» і закінчуються «* /».

Під час розробки, на етапі створення на мову програмування JavaScript вплинуло багато інших мов програмування. У розробників була мета зробити мову схожою на Java, але при цьому легкою для використання непрограмістів. Мовою

програмування JavaScript не володіє будь-яка компанія або організація, що відрізняє її від ряду інших мов програмування використовуваних в веб-розробці [21].

2.2.4. Фреймворк AngularJS

AngularJS – це фреймворк з відкритим кодом, оснований на мові програмування JavaScript. Він підтримується Google. AngularJS має на меті спростити як розробку, так і тестування таких веб-сервісів. AngularJS надає фундамент для веб-розробки [22].

Для опису інтерфейсу використовується декларативне програмування, а бізнес-логіка відокремлена від коду інтерфейсу. Це дозволяє поліпшити тестованість і розширюваність додатків.

Іншою відмінною рисою фреймворка є двостороннє зв'язування. Воно дозволяє динамічно змінювати дані в одному місці інтерфейсу при зміні даних моделі в іншому. Таким чином, AngularJS синхронізує модель і уявлення.

Крім того, AngularJS підтримує такі функціональності, як:

- управління структорою DOM;
- анімацію;
- шаблони;
- маршрутизацію.

Міць фреймворка, наявність багатого функціоналу багато в чому вплинула на те, що він знаходить своє застосування у все більшій кількості веб-додатків, будучи на даний момент напевно одним з найпопулярніших javascript-фреймворків [23].

На офіційному сайті фреймворку можна знайти вихідні файли, навчальні матеріали та інші супровідні матеріали щодо бібліотеки.

Основною формою організації додатків в AngularJS є модулі. Модуль представляє сховище різноманітної інформації: директив, фільтрів, контролерів і т.д. При цьому один додаток може мати кілька модулів. Наприклад, різні модулі можуть представляти будь-який функціонал.

Модулі дозволяють асоціювати певну ділянку html-сторінки з додатком AngularJS. Модулі також дозволяють організувати і структурувати різні компоненти

програми AngularJS. Крім того, модульність архітектури додатку підвищує тестованість продукту, і дає можливість використовувати різні частини-модулі додатка в інших додатках [24].

AngularJS спочатку читає сторінку HTML, в яку вбудовано додаткові атрибути HTML. Angular інтерпретує ці атрибути як директиви для прив'язки вхідних або вихідних частин сторінки до моделі, яка представлена стандартними змінними JavaScript. Значення цих змінних JavaScript можна встановити вручну в межах коду або отримати зі статичних або динамічних ресурсів JSON.

Згідно з аналітичною службою JavaScript Libscore, AngularJS використовується на сайтах Wolfram Alpha, NBC, Walgreens, Intel, Sprint, ABC News і близько 12000 інших сайтів з 1 мільйона тестованих у жовтні 2016 року. AngularJS в даний час входить до 100 найпопулярніших проектів GitHub [22].

2.3. Технології для розробки серверної частини

Для розробки серверної частини використано:

- JavaScript;
- Node.js.

2.3.1. Середовище виконання Node.js

Node.js – середовище виконання JavaScript [25]. Робоче середовище Node включає в себе все, що потрібно для виконання програми, написаної на JavaScript.

Програмна платформа Node.js побудована на основі двигуна V8, вона ніби перетворює мову програмування JavaScript з вузькоспеціалізованої мови програмування в мову програмування загального напряму. Node.js дає можливість мові програмування JavaScript:

- взаємодіяти з пристроями вводу-виводу через свій API (він написаний на мові програмування C++);
- підключати інші зовнішні бібліотеки, які написані на різних мовах програмування [25].

Переважно, Node.js використовується для розробки серверної частини проекту. Проте, Node.js також має і можливості для розробки десктопних додатків.

Node.js використовує модульну систему. Тобто весь її вбудований функціонал розбита на окремі пакети або модулі. Модуль представляє блок коду, який може використовуватися повторно в інших модулях [26].

При необхідності ми можемо підключати потрібні нам модулі. Які вбудовані модулі є в node.js і які функції вони надають, можна дізнатися з документації.

Для завантаження модулів застосовується функція require (), в яку передається назва модуля.

У Node.js також входить і npm, що означає менеджер пакетів. Для того того щоб встановити необхідний пакет потрібно у командному рядку ввести команду: «npm install «назва необхідного пакет»».

Для роботи з сервером і протоколом http в Node.js використовується модуль http. Щоб створити сервер, слід викликати метод «http.createServer ():».

За замовчуванням Node.js не має вбудованої системи маршрутизації. Зазвичай вона реалізується за допомогою спеціальних фреймворка типу Express. Однак якщо необхідно розмежувати найпростішу обробку кількох маршрутів, то цілком можна використовувати для цього властивість url об'єкта Request [26].

Node.js була створена у 2009 році. Її засновник –Раян Даль, який два роки експериментував над створенням серверних веб-проектів. В процесі свої досліджень програміст дійшов висновку, що замість традиційної моделі паралелізму на основі потоків варто звернутись до подіє-орієнтованої системи. Ця модель була обрана завдяки її простоті та швидкодії. Мета Node – запропонувати простий спосіб побудови мережевих серверів.

Node.js – відкритий проект. Його вихідний код доступний на github.com

Висновки до розділу 2

У розділі 2 досліджені та описані технології інтеграції алгоритму геометричних чисельних методів до веб-сервісу. Для інтеграції обрано метод

“модулю до двигуна V8” тому що це середовище виконання побудовано на базі двигуна V8, що дає змогу реалізувати логіку на C++.

3. АРХІТЕКТУРА ВЕБ-СЕРВІСУ

Сервіс-орієнтована архітектура – модульний підхід до розробки програмного забезпечення, заснований на використанні розподілених, слабо пов'язаних (англ. Loose coupling) замінних компонентів, оснащених стандартизованими інтерфейсами для взаємодії за стандартизованими протоколами.

3.1. Перелік необхідних інструментів

Розроблений веб-сервіс геометричних чисельних методів на базі модулю до Node.js має клієнт-серверну архітектуру. Вона складається з трьох компонентів [11]:

- серверів;
- клієнтів;
- мережі.

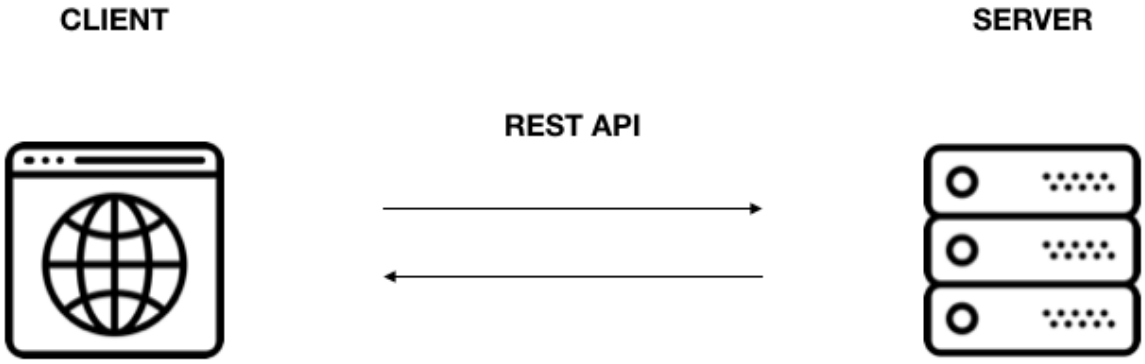


Рисунок 3.1 – Архітектура веб-сервісу

Обмін даними між клієнтом та сервером проводиться за рахунок REST API.

REST (скорочення від англ. Representational State Transfer - «передача стану уявлення») - архітектурний стиль взаємодії компонентів розподіленого додатка в мережі. REST є узгоджений набір обмежень, що враховуються при проектуванні розподіленої гіпермедіа-системи. У певних випадках (інтернет-магазини, пошукові

системи, інші системи, засновані на даних) це призводить до підвищення продуктивності і спрощення архітектури. У широкому сенсі [уточнити] компоненти в REST взаємодіють на зразок взаємодії клієнтів і серверів у Всесвітній павутині. REST є альтернативою RPC.

Висновки до розділу 3

У розділі 3 описана архітектура веб-сервісу. Для реалізації була використана архітектура типу клієнт-сервер, тому що вона дає змогу впровадити доступ до модулю на C++ коді, який інтегровано в Node.js платформу.

4. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Інтеграцію алгоритму геометричних чисельних методів, написаного на мові програмування C++, до веб-сервісу у роботі показано на базі алгоритму AABB – axis-aligned bounding box. Даний алгоритм лежить в основі виявлення зіткнень у 2D і 3D графіці.

Задача полягає в тому щоб порівняти правильність і швидкодію AABB алгоритму, написаного на мові програмування C++ і алгоритму, написаного на мові програмування JavaScript.

4.1. Перелік необхідних інструментів

Компіляція вібувається за допомогою Node-gyp пакету. Node-gyp — це інструмент, який компілює Node.js Addons. Додаток Node.js є модулями Node.js, написаними на C або C++, тому вони повинні бути скомпільовані на комп'ютері. Після того, як за допомогою Node-gyp додаток скомпільовано, їх функціональність може бути доступна через require (), як і будь-який інший модуль Node.js.

Node-gyp реалізовано і написано на мові програмування Python.

Для реалізації веб-сервісу геометричних чисельних методів на базі модулю до Node.js необхідно встановити на машину:

- Python 2.7;
- утиліту Make;
- g++ компілятор C++ коду.

Утиліта Make входить до складу фактично усіх Unix-подібних операціних систем. Вона є універсальною, де одні файли повинні автоматично оновлюватися при зміні інших файлів [27]. Утиліта Make автоматично будує програми. При запуску, програма make читає файл з описом проекту, тобто make-файл і, інтерпретуючи його вміст, робить необхідні дії. Файл із описом проекту являє

собою текстовий файл, де описані відносини між файлами проекту, і дії, які необхідно виконати для його складання.

Правила виконаная задані в скрипті з ім'ям Makefile, який повинен знаходитися в корені робочої директорії проекту. Скрипт складається з набору правил, які в свою чергу описуються:

- цілями (то, що дане правило робить);
- реквізитами (то, що необхідно для виконання правила і отримання цілей);
- командами (які виконують дані перетворення).

g++ – традиційне позначення для GNU C++. GNU (GCC) – це система компіляторів, створена проектом GNU, що підтримує різні мови програмування. GCC є ключовим компонентом інструментарію GNU і стандартним компілятором для більшості проектів, пов'язаних з GNU і Linux, найбільш помітним є ядро Linux.

Компілятор g++ компілює програмний код у об’єктні модулі, лінкує об’єктні модулі в одну цілу прграму. Компілятор сам визначає мову програмування на якій написаний код.

Щоб запустити вихідний код, за допомогою компілятора G++, необхідно ввести в командній стрічці “g++ filename”,.

4.2. Сценарій створення додатку до Node.js

Перше, що потрібно зробити для побудови додатку – створити файл binding.gyp. Розширення “gyp”, розшифровується як generate your project. Це конфігураційний файл. Він виступає у ролі шаблону до будь-якого додатка. Файл містить такі характеристики:

- “target_name” – визначає вихідну директорію додатку, у випадку мого веб-сервісу випадку він повинен бути побудований у папку “result”;
- “sources” – включає всі файли cpp, які пов'язані з додатком. В моєму випадку це файл “main.cpp” – він містить алгоритм геометричних чисельних методів, який я інтегрую у веб-сервіс,

- “include_dirs” – включає додаткові директиви, які бути включені під час створення до Якщо виконати цей скрипт у терміналі, в результаті буде отримано шлях до вузла-модуля для Nan.

Рисунок 4.1– Структура файлу “binding.gyp”

Далі, необхідно створити файл “package.json”. Він зберігає список пакетів та їх версій, які необхідні для проекту. Завдяки файлу “package.json” можна встановити всі необхідні пакети на іншій машині. Це можна зробити за допомого команди “npm i ” [28].

Рисунок 4.2 – Структура файлу “binding.gyp”

Єдине, що залишилося зробити перед тим, як перейти до реалізації, буде установка Nan. Це можна зробити за допомогою команди “npm i”.

Під час виконання роботи я зіткнувся з проблемою. Команда “npm i”, якF виклика файл node-gyp не працює. Для того щоб її вирішити необхідно:

- комп’ютер повинен працювати на ОС Windows,
- перевіритичи встановлений на комп’ютері Python. Для того щоб це зробити, напишіть у командному рядку “python --version”. Якщо Python встановлено на комп’ютері, як результат ви отримаєте версію Python, яка встановлена на комп’ютері. Якщо на комп’ютері не встановлено Python, скачайте та встановіть Python версії 2.7,
- перевірити чи встановлений на комп’ютері NET 4.5.1,
- встановити Microsoft Visual C++ Build Tools 2015 Technical Preview. Оберіть кастомну установку. Необхідно встановити лише Windows 8.1 SDK,
- встановити npm конфігурацію msvs_version на 2015. Для того щоб це зробити, напишіть у командному рядку “npm config -g set msvs_version 2015” [29].

Після змін, які були проведені у конфігураційних файла, можна переходити до створення вхідного для файлу додатку.

Кожен додаток бере свій початок з виклику макросу. Шаблон макросу NODE_MODULE приймає target name як перший аргумент (те ж саме щ обуло встановлено як target name у файлі GYP, пам'ятаєте) і метод ініціалізації для нашого модуля. NAN_MODULE_INIT створює функцію з заданим іменем. Він приймає ціль як перший аргумент, еквівалентний експорту NodeJS [5].

```
NAN_MODULE_INIT(Initialize) {
    NAN_EXPORT(target, aabb);
}

NODE_MODULE(addon, Initialize);
```

Рисунок – 4.3 Створення додатку

AABB було експортовано за допомогою макросу NAN_EXPORT. Макрос NAN_METHOD було використано, щоб визначити нову функцію, допустиму для вузла. Він приймає інформацію як перший аргумент, і це те ж саме, що і вектор аргументів JavaScript.

Функція To () використовується для перетворення першого аргументу у потрібний тип. Потім викликано метод ToLocalChecked (). Він перетворює результат у v8's Local. Якщо аргумент не визначено – буде визначена помилка.

```
v8::Local<v8::String> firstFigureXFromJS = Nan::New("firstFigureX").ToLocalChecked();
v8::Local<v8::String> firstFigureYFromJS = Nan::New("firstFigureY").ToLocalChecked();
v8::Local<v8::String> firstFigureWidthFromJS = Nan::New("firstFigureWidth").ToLocalChecked();
v8::Local<v8::String> firstFigureHeightFromJS = Nan::New("firstFigureHeight").ToLocalChecked();
v8::Local<v8::String> secondFigureXFromJS = Nan::New("secondFigureX").ToLocalChecked();
v8::Local<v8::String> secondFigureYFromJS = Nan::New("secondFigureY").ToLocalChecked();
v8::Local<v8::String> secondFigureWidthFromJS= Nan::New("secondFigureWidth").ToLocalChecked();
v8::Local<v8::String> secondFigureHeightFromJS = Nan::New("secondFigureHeight").ToLocalChecked();
```

Рисунок – 4.4 Використання методу ToLocalChecked ()

Потім ми знову перетворюємо функцію To () для перетворення результату в потрібний тип даних, тільки на цей раз це примітив, тому ми використовуємо FromJust () замість ToLocalChecked (). Двигун v8 використовує лише подвійну точність, а не з плаваючою точкою. Ми можемо легко отримувати властивості з даного об'єкта JS, використовуючи метод Get (). Зверніть увагу на використання ->, а не на період, тому що пам'ятайте, що Local є покажчиком, це не є справжній об'єкт.

```
if (Nan::HasOwnProperty(firstFigure, firstFigureXFromJS).FromJust()) {
    v8::Local<v8::Value> value = (Nan::Get(firstFigure, firstFigureXFromJS).ToLocalChecked());
    firstFigureX = value->NumberValue();
}
```

Рисунок 4.5 – Використання методу FromJust ()

Тепер залишається лише визначити значення, що повертається. Значення повинно бути повернуто через поточну сферу дії v8, а не спочатку, тому використання ключового слова return буде марним.

4.3. Алгоритм AABB

Виявлення зіткнень, або англійською – collision detection – основа для задач візуалізації [33].

Задача алгоритму – визначити зіткнулися два об'єкти чи ні.

Складність задачі полягає в тому, що при тестуванні одних примітивів візуалізації з іншими примітивами, обчислення колізій займає дуже великі ресурси. Об’єм задіяних ресурсів особливо збільшується зі збільшення кількості об'єктів, які беруть участь в симуляції.

Для того, щоб полегшити обчислення і не проводити перевірку безпосередньо між примітивами візуалізації (подібне займає найбільше ресурсів і часу), були вигадані технології, обмежуючих контейнерів, в англійській мові ще зустрічається поняття – “bounding volume”. Обмежуючі контейнери – це прості геометричні фігури, які описують більш складні геометричні об’єкти. Вони використовуються для підвищення ефективності геометричних операцій. Це пов’язано з тим, що як правило, прості томи мають простіші способи тестування на перекриття. Найбільш широковикористовуваними обмежуючими контейнерами є:

- сфера – sphere;

- граничний паралелепіпед, вирівняний по координатним осях – axis-aligned bounding boxes – AABB;

- об'єктно-орієнтований обмежуючий паралелепіпед – object-oriented bounding boxes – OB;

- дискретно-орієнтовані многогранники. – discrete oriented polytopes [30].

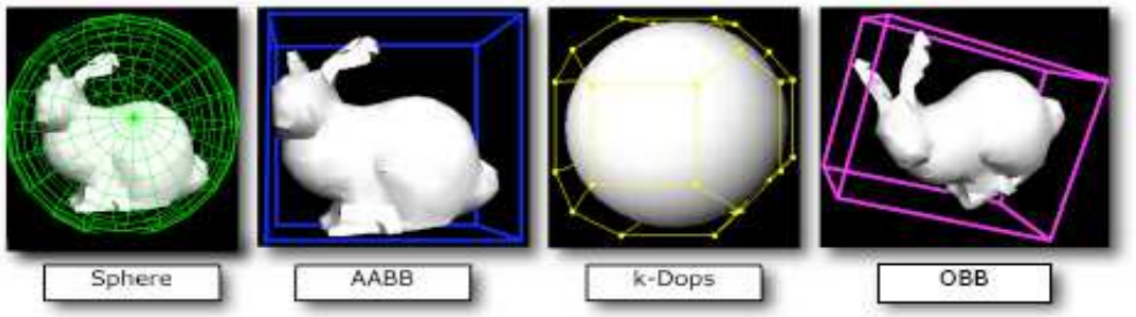


Рисунок 4.6 – Найбільш широковикористовувані обмежуючі томи

Вибір типу обмежувального контейнеру для конкретного застосування визначається такими факторами:

- необхідна точність перетину тесту;
- об’єм ресурсів, які необхідні для обчислення обмежувального тому для об'єкта [31].

Обмежуючі контейнери називають k-DOP. Кожен із перелічених вище обмежуючих контейнерів має свій ряд переваг та недостатків.

Дискретно-орієнтовані многогранники – це заздалегідь відома кількість обмежують площин і їх орієнтації. У назві описує кількість таких площин. Наприклад, двомірний 4-DOP є звичайним квадратом, що описує якийсь також двомірний об'єкт. Алгоритм AABB – розшифровується як Axis-Aligned Bounding Box. На українську це перекладається як “обмежуючий малогабаритний том”. По суті AABB – це окремий випадок k-DOP.

Як і у випадку виявлення 2D зіткнення алгоритм AABB, є найшвидшим алгоритмом для визначення того, чи перекриваються ці два об'єкти [33].

Двомірний AABB є таким же квадратом, як і 4-DOP.

Тривимірний AABB є таким же кубом, як і 6-DOP.

Але k-DOP так само може використовувати більшу кількість площин. Орієнтація таких площин вибирається заздалегідь і не змінюється [31].

k-DOP описується всього лише мінімальними і максимальними інтервалами – slabs або дистанціями. По великому рахунку, мінімальним інтервал, якщо говорити

про AABB алогоритм, є ліва верхня точка обмежуючого тому. В той же час максимальним інтервалом є довжина і висота обмежуючого тому.

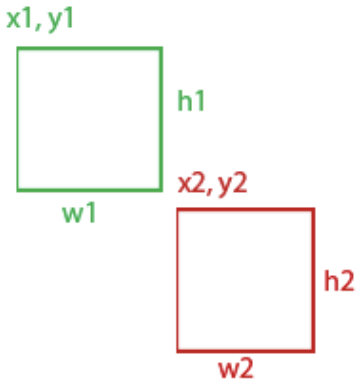


Рисунок 4.7 – Дані необхідні для визначення наявності зіткнення

Область перекриття між двома обмежуючими малогабаритними контейнерами може бути перевірена лише за допомогою логічних порівнянь.

Два значення на одну площину, що в підсумку виходить досить просто обробляти і зберігати. Наприклад, для квадрата, як ми запам'ятали, треба 4 площині (k = 4). Значить, буде потрібно 2 (k / 2) мінімальних і 2 максимальних значень.

Якщо принаймні одна площина не перетинається, то вся структура також не перетинається. Тільки якщо всі інтервали площині перетинаються, тоді перетинається сама структура k-DOP. Але слід зауважити, що описуваний об'єкт може і не перетинатися.

Тому маємо те, що k-DOP структури двох об'єктів стикаються (collide), то їх описувані об'єкти можуть стикатися.

Результатом роботи алгоритму є значення, яке визначає: є зіткнення чи немає.

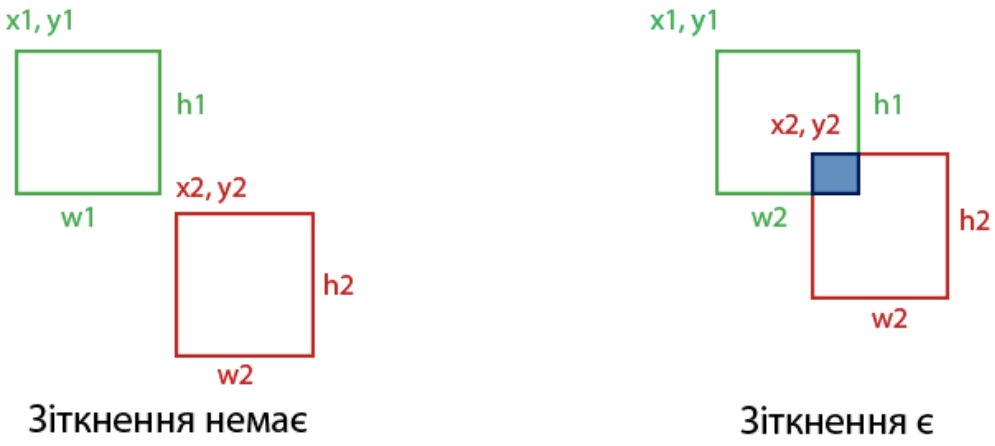


Рисунок 4.8 – Результат роботи алгоритму

У роботі порівнюються правильність роботи та швидкодія алгоритму AABB, який написаний на мові програмування C++ і на мові програмування JavaScript.

Код алгоритму на мові програмування C++ виглядає так:

```
if(firstFigureX < secondFigureX + secondFigureWidth &&
   firstFigureX + firstFigureWidth > secondFigureX &&
   firstFigureY < secondFigureY + secondFigureHeight &&
   firstFigureY + firstFigureHeight > secondFigureY){
   result = true;
};
```

Рисунок 4.9 – Алгоритм AABB на C++

Код алгоритму на мові програмування JavaScript виглядає так:

```
function check(firstFigure, secondFigure) {
    if (firstFigure.x < secondFigure.x + secondFigure.width &&
        firstFigure.x + firstFigure.width > secondFigure.x &&
        firstFigure.y < secondFigure.y + secondFigure.height &&
        firstFigure.y + firstFigure.height > secondFigure.y) {
        return true;
    } else return false;
}
```

Рисунок 4.10 – Алгоритм AABB на C++

4.4. Порівняння швидкодій алгоритмів

Проведена робота дає можливість порівняти швидкодію алгоритму AABB, написаного на мові програмування C++ та оформленого у форматі додатку до Node.js та алгоритму AABB, написаного на мові програмування JavaScript.

Порівняння швидкодії алгоритмів було проведено для алгоритмів різної степені складності, що дало змогу оцінити роботу системи у різних умовах. Наведену вибірку не можна вважати репрезентативною. Проте, вона дає змогу детально дослідити і порівняти швидкості виконання алгоритмів.

При виконанні алгоритму для 1 пари об’єктів, шивдкість алгоритму на C++ у 73 рази повільніше ніж на JavaScript.

При виконанні алгоритму для 100 пар об’єктів, шивдкість алгоритму на C++ у 3,5 рази повільніше ніж на JavaScript. Спостерігається приріст у швидкодії алгоритму на C++.

При виконанні алгоритму для 1 000 пар об’єктів, шивдкість алгоритму на C++ у 2,5 рази повільніше ніж на JavaScript. Знову спостерігається приріст у швидкодії алгоритму на C++.

При виконанні алгоритму для 10 000 пар об’єктів, шивдкість алгоритму на C++ у 7 рази повільніше ніж на JavaScript. Спостерігається приріст у швидкодії алгоритму на JavaScript.

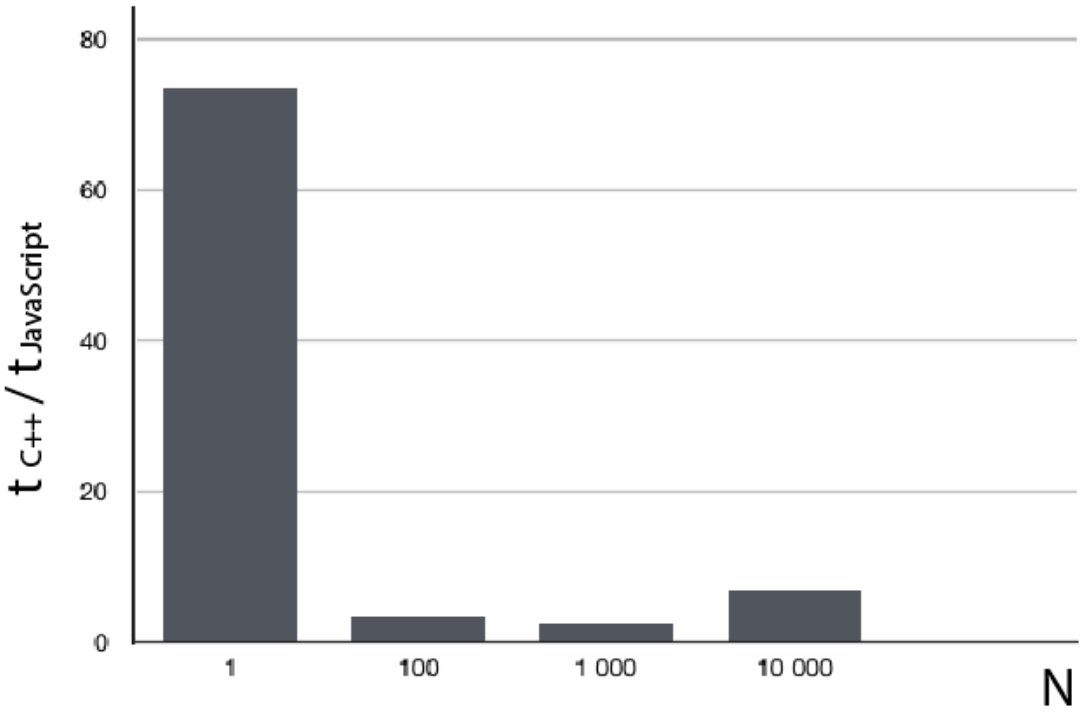


Рисунок 4.11 – Графік порівняння швидкодії алгоритму AABB на мові C++ та шивдкодії алгоритму на JavaScript

Висновки до розділу 4

У розділі описана інструкція по інтеграції алгоритму геометричних чисельних методів до веб-сервісу, опис алгоритму ABBB, наведено таблиця порівняння швидкодій алгоритмів.

5. МЕТОДИКА РОБОТИ КОРИСТУВАЧА З ПРОГРАМОЮ

В цьому розділі наведено системні вимоги для роботи з застосунком та сценарії роботи користувача з ним.

5.1 Системні вимоги до програмного продукту

Програмний продукт було розроблено та протестовано на двох комп'ютерах. Комп'ютер №1 — Apple MacBook Air 13" 1.8GHz 128GB. Він має такі технічні характеристики: двоядерний процесор Intel ® Core i5 (1.8 - 2.9 ГГц), об'єм оперативної пам'яті — 8 ГБ, SSD на 128 ГБ та графічний чип Intel HD Graphics 6000. Комп'ютер №2 — Lenovo V130-14IKB. Він має такі технічні характеристики: Intel ® Core i5-7200U (2.5 - 3.1 ГГц), об'єм оперативної пам'яті — 8 ГБ, SSD на 128 ГБ та графічний чип Intel HD Graphics 620. На обидвох комп'ютерах програмний продукт працював коректно в процесі тестування жодних багів я не виявив.

Проте, програмний продукт також буде коректно і безвідмовно працювати і на комп'ютерах з гіршими технічними характеристиками. Мінімальними необхідними технічними характеристиками є: процесор не гірше, ніж Intel ® Pentium ® / Celeron ® / Xeon™ або з тактовою частотою не менше 1,8 GHz або AMD 6 / Turion ™ / Athlon ™ / Duron ™ / Sempron ™; об'єм оперативної пам'яті не менше 2 ГБ; графічне ядро не гірше, ніж Intel ® HD Graphics 2000, воно еквівалентне графічним картам з об'ємом 128 Mb.

Програмний продукт коректно працює як на комп'ютерах з операційною системою Windows, так і на комп'ютерах з операційною системою OS X (комп'ютерах від торгової марки Apple).

Попередні тестування програмного продукту на інших пристроях проведені не були. Проте, враховуючи, що програмний продукт – це веб-серівс, можна стверджувати, що програмний продукт також коректно працюватиме на

комп’ютерах з операційною системою Linux. Перегляд програмного продукту на різних пристроях можливий за допомогою веб-браузера.

5.2 Сценарій роботи користувача з програмою

Веб-сервіс має зрозумілий користувачу інтерфейс. Інтерфейс веб-сервісу має такий вигляд (рис. 5.1.):

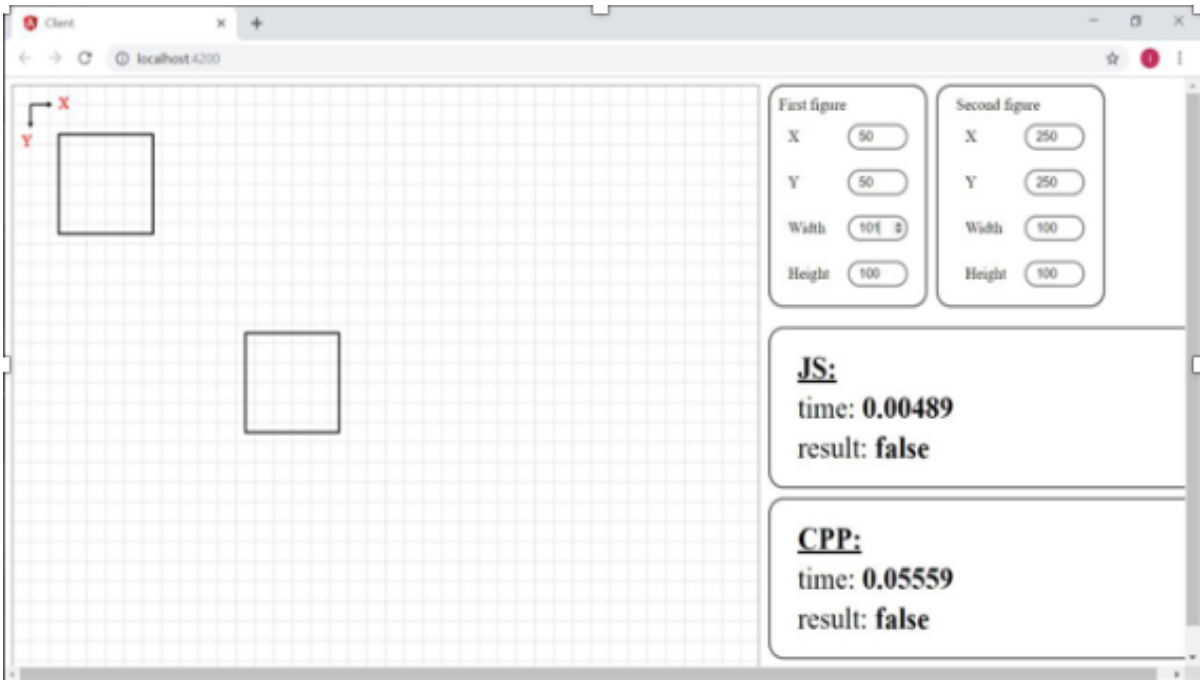


Рисунок 5.1 – Головне вікно веб-сервісу

В його лівій частині можна побачити область, на якій малюються Об’єкт №1 та Об’єкт №2. Об’єкт №1 має початкові характеристики: x — 50; y — 50; width — 100; height — 100. Об’єкт №1 на початку має форму квадрату. Об’єкт №2 має початкові характеристики: x — 250; y — 250; width — 100; height — 100. Об’єкт №2 на початку теж має форму квадрату. Розмір області становить 800 пікселів у ширину і 600 пікселів у висоту.

Кожен з об’єктів, і Об’єкт №1 і Об’єкт №2 мають по 4 характеристики. Ці характеристики необхідні для того щоб визначити: відбулось зіткнення об’єктів чи ні. Користувач має два варіанти, як змінити характеристики об’єктів: за допомогою

курсору: при наведенні на значення характеристики поруч із нею з’являються стрілки. Натискаючи на стрілки можна змінити значення величини;

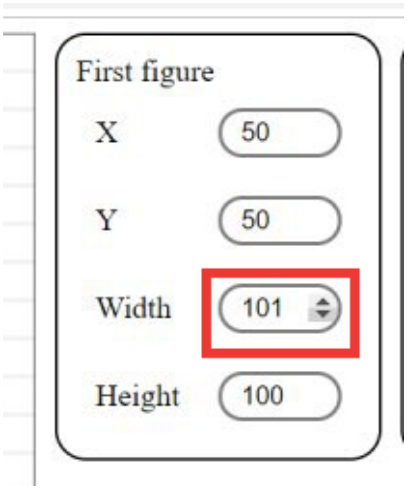


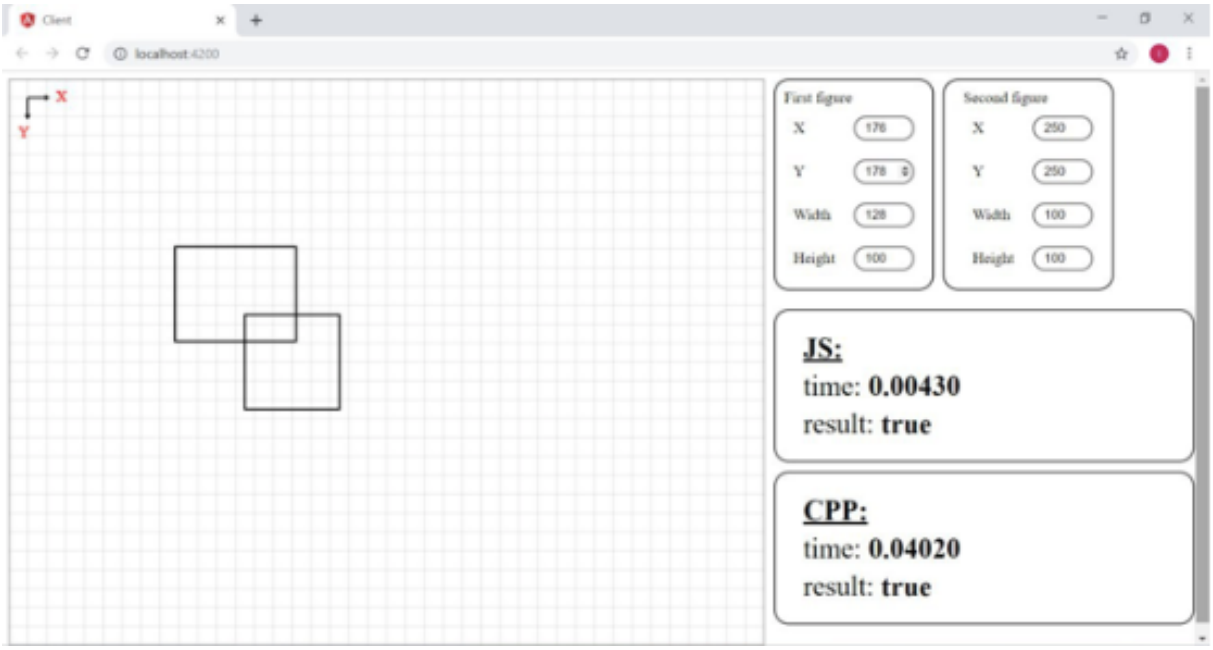
Рисунок 5.2 – Демонстрація способу зміни величини “Width”

Також змінити величину характеристики можна за допомогою вводу з клавіатури. Змінюючи, будь-яку з величин характеристик кожної об’єктів, користувач може побачити її переміщення в лівій частині екрану.

Зі зміною характеристик фігур змінюється і результат алгоритму.

Результатом роботи веб-сервісу і для C++ коду, і для JavaScript коду є два значення:

- наявність зіткнення. Ця характеристика може приймати лише два значення «true» або «false». Характеристика приймає і виводить значення «true», якщо є зіткнення між Об’єктом №1 і Об’єктом №2;



- Рисунок 5.3 – Робота програми при зіткненні об’єктів

характеристика приймає і виводить значення «false», якщо зіткнення між Об’єктом №1 і Об’єктом №2 немає;

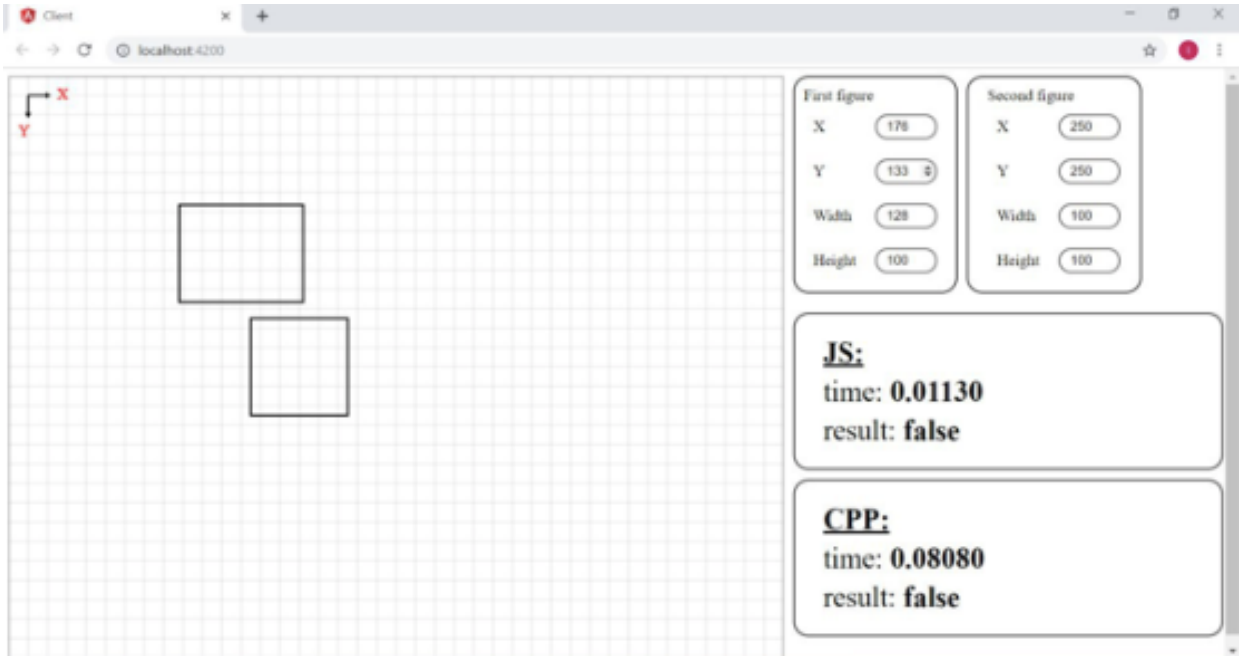


Рисунок 5.4 – Робота програми при відсутності зіткнення об’єктів

- час роботи алгоритму. Ця характеристика виводиться в секундах, довжина виводу 5 символів. Такого розміру чисел достатньо задля того щоб порівняти швидкодію алгоритму AABB. За потреби, кількість символів, що виводить це поле можна збільшити.

Висновки до розділу 5

У розділі 5 описано логіку роботи системи й подано інформацію для користувачів системи з метою полегшення їхньої подальшої взаємодії із розробленими додатками. Подано зразки інтерфейсу для спрощення сприйняття інструкції по використанню програми.

ВИСНОВКИ

Робота присвячена створенню веб-сервісу геометричних чисельних методів на базі модулю до Node.js. При вирішенні поставлених задач були отримані наступні висновки:

1. В процесі аналізу методів інтеграції C++ коду було встановлено, що для реалізації системи раціонально використати метод “модулю до двигуна V8”, тому що наявний доступ до редагування коду алгоритму на мові програмування C++ та наявний доступ до ресурсів, які даяють змогу створити додаток до Node.js

2. Було проведено інтеграцію алгоритму геометричних чисельних методів у веб-сервіс у якості додатку до Node.js, що дало змогу використовувати в Інтернеті алгоритм, написаний на мові програмування C++.

3. Порівняно швидкодію алгоритмів AABB на мові програмування C++, оформленого як додаток до Node.js та на мові програмування JavaScript, який показав що для однієї пари об’єктів код на мові програмування JavaScript виконується в 73 рази швидше ніж код на мові програмування на C++. Проте, при збільшенні кількості пар об’єктів ррізниця між швидкостями виконання алгоритмів зменшується в 36 разів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Automating a C++ program from a Node.js web app [Електронний ресурс]. – 2015. – Режим доступу до ресурсу: https://nodeaddons.com/automating-a-c-program-from-a-node-js-web-app/.
2. Calling Native C++ DLLs from a Node.js Web App [Електронний ресурс]. – 2015. – Режим доступу до ресурсу: https://nodeaddons.com/calling-native-c-dlls-from-a-node-js-web-app/.
3. Building an Asynchronous C++ Addon for Node.js using Nan [Електронний ресурс] // 2015 – Режим доступу до ресурсу: https://nodeaddons.com/building-an-asynchronous-c-addon-for-node-js-using-nan/.
4. Пишем модуль на C++ для nodejs на примере работы с MySQL [Електронний ресурс] // 2012 – Режим доступу до ресурсу: https://habr.com/ru/post/154007/.
5. NodeJS Advanced—How to create a native add-on using C++ [Електронний ресурс] // 2017 – Режим доступу до ресурсу: https://medium.com/the-guild/nodejs-advanced-how-to-create-a-native-add-on-using-c-588b4f2248cc.
6. Chrome V8 [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Chrome_V8.
7. Разработка нативных расширений для Node.js [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/company/ruvds/blog/353072/.
8. Visual Studio Code Getting started [Електронний ресурс] – Режим доступу до ресурсу: https://code.visualstudio.com.
9. Visual Studio Code Getting started [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Visual_Studio_Code.
10. Extensions for the Visual Studio family of products [Електронний ресурс] – Режим доступу до ресурсу: https://marketplace.visualstudio.com.
11. Клієнт-серверна архітектура [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Клієнт-серверна_архітектура.
12. World Wide Web Consortium [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/World_Wide_Web_Consortium.

13. W3 [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Glossary/World_Wide_Web.
14. What is HTML? [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3schools.com/whatis/whatis_html.asp.
15. HTML5 [Електронний ресурс] – Режим доступу до ресурсу: http://htmlbook.ru/html5.
16. FIVE THINGS YOU SHOULD KNOW ABOUT HTML5 [Електронний ресурс] – Режим доступу до ресурсу: https://diveintohtml5.info/introduction.html.
17. HTML5 [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/HTML5.
18. Cascading Style Sheets [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Cascading_Style_Sheets.
19. CSS3 [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/web/html5/5.1.php.
20. What is JavaScript? [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3schools.com/whatis/whatis_js.asp.
21. JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/JavaScript.
22. AngularJS [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/AngularJS.
23. AngularJS [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/web/angular/1.1.php.
24. Модули в AngularJS [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: https://metanit.com/web/angular/2.7.php.
25. What exactly is Node.js? [Електронний ресурс] // 2018 – Режим доступу до ресурсу: https://www.freecodecamp.org/news/what-exactly-is-node-js-ae36e97449f5/.
26. Модули Node.js [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/web/nodejs/2.1.php.

27. Утиліта Make [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/211751/.
28. Зачем нужен package.json [Електронний ресурс] – Режим доступу до ресурсу: https://monsterlessons.com/project/lessons/zachem-nuzhen-packagejson.
29. Npm install that requires node-gyp fails on Windows [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://stackoverflow.com/questions/35293117/npm-install-that-requires-node-gyp-fails-on-windows/42713948.
30. JavaScript vs. C++: создание одной и той же 3D-игры на обоих языках [Електронний ресурс] // 2016 – Режим доступу до ресурсу: https://tproger.ru/translations/js-and-c-fps/.
31. Введение в дискретно-ориентированные многогранники для задачи определения столкновений [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/257339/.
32. AABB [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/AABB.
33. AABB: Axis-Aligned Bounding Box [Електронний ресурс] – Режим доступу до ресурсу: http://www.gamedev.ru/terms/AABB.
34. Axis-aligned bounding boxes [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_collision_detection.

ДОДАТОК А

Веб-сервіс геометричних чисельних методів на базі модулю до Node.js

Специфікація

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР-5294_19Б

Аркушів 1

Київ 2019

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР5294_19Б	Записка.docx	Текстова частина дипломної роботи
Компоненти		
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР5294_19Б 1	server	Основна папка, яка містить необхідні файли для запуску сервера
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР5294_19Б	client	Основна папка, яка містить файли для клієнта

ДОДАТОК Б

Веб-сервіс геометричних чисельних методів на базі модулю до Node.js

Текст програми

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР-5294_19Б

Аркушів 2

Київ 2019

```
const express = require("express");

const cors = require('cors');

const app = express();

var performance = require('perf_hooks').performance;

const { aabb } = require('./build/Release/result');

// var bodyParser = require('body-parser');

// app.use(bodyParser.urlencoded({ extended: false }))

app.use(express.json())

app.use(cors());

app.listen(3000, () => {

});

app.get("/", (req, res) => {

    res.send({ code: 200 });

});

//Обраховую час і результат алгоритму

app.post("/calculate", (req, res) => {



    const { firstFigure, secondFigure } = req.body;



    var t1 = performance.now();

    const cppResult = aabb({ "firstFigureX": firstFigure.x,

                             "firstFigureY": firstFigure.y,

                             "firstFigureWidth": firstFigure.width,

                             "firstFigureHeight": firstFigure.height },

                           { "secondFigureX": secondFigure.x,

                             "secondFigureY": secondFigure.y,

                             "secondFigureWidth": secondFigure.width,

                             "secondFigureHeight": secondFigure.height });

    const cppTime = performance.now() - t1;

    t1 = performance.now();

    const jsResult = check(firstFigure, secondFigure);

    const jsTime = performance.now() - t1;
```

```
    // Надсилаю результати для клієнта
    res.send({ code: 200, results: {js: { time: jsTime, result: jsResult }, cpp: { time: cppTime, result: cppResult }} });
});
function check(firstFigure, secondFigure) {
    if (firstFigure.x < secondFigure.x + secondFigure.width &&
        firstFigure.x + firstFigure.width > secondFigure.x &&
        firstFigure.y < secondFigure.y + secondFigure.height &&
        firstFigure.y + firstFigure.height > secondFigure.y) {
        return true;
    } else return false;
}
function check(firstFigure, secondFigure) {
    if (firstFigure.x < secondFigure.x + secondFigure.width &&
        firstFigure.x + firstFigure.width > secondFigure.x &&
        firstFigure.y < secondFigure.y + secondFigure.height &&
        firstFigure.y + firstFigure.height > secondFigure.y) {
        return true;
    } else return false;
}
}
```

ДОДАТОК В

Веб-сервіс геометричних чисельних методів на базі модулю до Node.js

Опис програми

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР-5294_19Б

Аркушів 8

Київ 2019

АНОТАЦІЯ

Даний додаток містить опис веб-сервісу геометричних чисельних методів на базі модулю до Node.js розробленого для інтеграції динамічної бібліотеки математичних розрахунків. Створено веб-сервіс типу “ клієнт – сервер”. Сервіс виконує наступні завдання:

1. Введення даних для обчислення
2. Надсилання даних на сервер
3. Обчислення даних на сервері
4. Надсилання результату обрахунків на клієнт.

При розробці веб-сервісу використовувався мова програмування Javascript, середовище Node.js та фреймворк Angular 2+ (в основі яких лежить Javascript).

ЗМІСТ

1. ЗАГАЛЬНІ ВІДОМОСТІ

У цьому додатку міститься опис веб-сервісу геометричних чисельних методів на базі модулю до Node.js. У додатку Б міститься програмний код модулю розроблюваного серверу.

Веб-сервіс виконано для подальшої інтеграції на серверну віддалену машину. Для коректної роботи серверу необхідно встановити всі залежні модулі на сервер, в тому числі і Node-gyp.

При розробці веб-сервісу використовувалась мова Javascript з використанням середовища Microsoft Visual Studio Code.

2. ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розроблений сервіс геометричних чисельних методів на базі модулю до Node.js виконує завдання інтеграції алогоритму, написаного за допомогою C++ мови у веб-сервіс.

Розроблений сервіс може використовуватись в якості учбових матеріалів при опрацюванні технологій C++, Javascript мов програмування. Функціональних обмеження на використання веб-сервісу полягає лише в діапазоні введення даних для обчислення швидкості.

3. ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Для реалізації взаємодії C++ коду з Node.js середовищем необхідно спочатку визначитись яким методом необхідно інтегрувати C++ коду. У роботі було вибрано метод модулю до двигуна V8. Згодом код C++ необхідно скомпілювати засобами Node-gyp. Отриманий модуль завантажено у Node.js за допомого функції require ().

При запиті клієнта до сервера, запускається функція обробник, яка в свою чергу, запускає функцію з скомпільованої бібліотеки та надсилає результат для клієнта.

4. ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ

Для забезпечення повноцінної роботи та досягнення високої ефективності роботи веб-сервісу для демонстрації взаємодії C++ і Node.js у процесі інтеграції динамічної бібліотеки було обрано Visual Studio Code яка показала себе надійним та гнучким середовищем розробки програм.

Розроблений веб-сервіс працює на будь-якій платформі у будь-якому браузері. У подальшому необхідно розмістити веб-сервіс на віддаленому сервері для його стабільної роботи та доступу у мережі інтернет.

5. ВИКЛИК І ЗАВАНТАЖЕННЯ

Розроблений веб-сервіс потребує інсталяції. Для того, щоб запустити сервер необхідно спочатку встановити всі залежності командою “npm i” в консолі. Після чого необхідно запустити сервер командою “node app”.

Для інсталяції клієнту необхідно також встановити залежності тією самою командою, що і для серверу. Після чого необхідно написати команду “ng s -o” та клієнт буде завантажено автоматично у браузері.

6. ВХІДНІ І ВИХІДНІ ДАНІ

Вхідними даними є результат роботи алгоритму наявність чи відсутність зіткнення; час виконання алгоритму.

ДОДАТОК Г

Веб-сервіс геометричних чисельних методів на базі модулю до Node.js

Апробації

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР-5294_19Б

Аркушів 4

Київ 2019

INFORMATION, SOFTWARE AND TECHNICAL SUPPORT FOR CONTROL SYSTEMS USED IN ORGANIZED TECHNOLOGY COMPLEXES

Abstracts of the International Science and Practice Conference for Young Scientists and Students

ІНФОРМАЦІЙНЕ, ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМ УПРАВЛІННЯ ОРГАНІЗАЦІЙНО-ТЕХНОЛОГІЧНИМИ КОМПЛЕКСАМИ

Збірник тез доповідей міжнародної науково-практичної конференції молодих вчених та студентів

ИНФОРМАЦИОННОЕ, ПРОГРАММНОЕ И ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ СИСТЕМ УПРАВЛЕНИЯ ОРГАНИЗАЦИОННО-ТЕХНОЛОГИЧЕСКИМИ КОМПЛЕКСАМИ

Сборник тезисов докладов международной научно-практической конференции молодых ученых и студентов

21-22.05.2019

Луцьк
2019

Чубаров І.О., Демчишин А.А.
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

ВЕБ-СЕРВІС ГЕОМЕТРИЧНИХ ЧИСЕЛЬНИХ МЕТОДІВ НА БАЗІ МОДУЛЮ ДО NODE.JS

На сьогодні, в період технічного розвитку існує понад двадцять популярних мов програмування та технологій [1], що полегшують роботу з ними. Деякі з мов мають схожу семантику, але в той же час їх суттєва різниця полягає в можливостях, і, як наслідок, сферах застосування: наприклад, одні мови спеціалізуються для веб-розробки (JavaScript), інші (C#, C++) для створення десктоп додатків. У зв'язку з цим виникає потреба у раціональному використанні можливостей/переваг/швидкодії кількох мов в контексті одного проекту.

Мета роботи: зкомпілювати C++ клас та використати його у веб-сервісі Node.js.

Для реалізації мети створено веб-сервіс типу «клієнт-сервер». Клієнтську частину розроблено засобами фреймворку Angular 7. В його основі лежить мова програмуваня JavaScript. Сервер розроблено за допомогою програмної платформи Node.js [2]. В її основі також лежить JavaScript. Вхідні дані з клієнта до сервера передаються за допомогою REST API. Коли клієнт виконує запит сервер запускає скомпільовану функцію з C++ класу та отриманий результат надсилає клієнту.

Для того щоб відбулась компіляція необхідні C++ клас та конфігураційний файл. У ньому вказано кінцеву директорію, файл для компіляції. Додатково можна вказати, які папки та файли будуть ввикористані у скомпільваному коді.

Компіляція вібувається за допомогою Node-gyp пакету [3]. Він написаний на мові програмування Python. Для його коректної роботи необхідно встановити на машину: Make – утиліту, що автоматизує процес перетворень файлів з однієї форми в іншу, G++ – компілятор C++ коду та Python 2.7. Далі, у командному вікні виконуємо команду “node-gyp rebuild”. Після цього node-gyp звертається до конфігураційного файлу, дивиться який файл потрібно скомпілювати та компілює його. В результаті отримаємо папку build, у ній знаходиться папка Release. У папці створюється багато файлів з одним іменем але різним розширенням. Сервер може виконуватись у різних середовищах. Я виконав роботу на Windows. Для різних середовищ виконуються файли з різним розширенням, усе залежить від того, яке розширення файлу підходить певному середовищу.

В роботі показано можливість використання існуючого коду для реалізації ресурсоємних розрахунків (AABB алгоритму) у вигляді веб-сервісу із сучасним прозорим інтерфейсом.

Як висновок, хочу сказати, що ми отримаємо приріст у швидкодії, тому що C++ є багатопоточною мовою, а JavaScript – однопоточною. Окрім того, є дуже багато готового коду написаного на C++, для того щоб не переписувати його на інші мови програмування – можна використовувати мою роботу.

Список використаних джерел:

1. TIOBE C. TIOBE Index for March 2019 [Електронний ресурс] / COMPANY TIOBE // TIOBE. – 2019. – Режим доступу до ресурсу: https://www.tiobe.com/tiobe-index/.

2. Hahn E. Express in Action / Evan Hahn. – Shelter Island, NY 11964: Manning Publications Co., 2016. – 217 p.

3. Baraniecki M. Extending Node.js with native C++ modules [Електронний ресурс] / Marcin Baraniecki // Medium. – 2017. – Режим доступу до ресурсу: https://medium.com/@marcinbaraniecki/extending-node-js-with-native-c-modules-63294a91ce4.

ДОДАТОК Д

Веб-сервіс геометричних чисельних методів на базі модулю до Node.js

Апробації

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР-5294_19Б

Аркушів 8

Київ 2019