УО «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра «Информационных систем и технологий»

Специальность 1-40 05 01 «Информационные Системы и Технологии»

**Реферат**

**На тему «Работа с JSON в С++»**

|  |  |  |  |
| --- | --- | --- | --- |
| **Выполнил** |  |  |  |
| Студент 1 курса группы 1 |  |  | Д.И. Велютич |
|  | подпись, дата |  | инициалы и фамилия |
| **Проверил** |  |  | Н.И. Белодед |
|  | подпись, дата |  | инициалы и фамилия |

Минск 2023

Оглавление

[Теория: 2](#_Toc134233702)

[Программный код: 3](#_Toc134233703)

[Скриншоты и пояснения: 5](#_Toc134233704)

[Выводы: 8](#_Toc134233705)

В этом коде представлен пример программы на языке программирования C++, которая считывает и записывает данные в формате JSON. Для работы с JSON используется сторонняя библиотека nlohmann/json.   


Рассмотрим подробно каждую функцию и добавим необходимую теорию.

# Теория:

JSON (JavaScript Object Notation) — это легковесный, текстовый и универсальный формат обмена данными, основанный на синтаксисе объектов JavaScript. Однако, несмотря на связь с JavaScript, JSON является независимым от языка форматом данных и может быть использован в сочетании с различными языками программирования.

JSON предоставляет простой и удобный способ представления структурированных данных в виде пар ключ-значение. Ключи представляют собой строки, а значения могут быть объектами, массивами, числами, строками, булевыми значениями (true/false) или null. В отличие от XML, JSON требует меньше символов и пространства для представления данных, что делает его более эффективным для передачи и хранения данных.

JSON используется в различных сферах, таких как:

1. Веб-приложения: JSON является стандартным форматом для обмена данными между клиентом и сервером. Веб-серверы обычно отправляют данные в формате JSON, а клиентские приложения, такие как веб-браузеры, обрабатывают и отображают эти данные пользователю.

2. Конфигурационные файлы: JSON используется в качестве формата для конфигурационных файлов, где он представляет собой удобный и понятный способ хранения настроек приложения или системы.

3. Сериализация объектов: JSON может использоваться для сериализации объектов, то есть преобразования объектов в строковый формат, который затем может быть сохранен в файле или отправлен через сеть.

4. Базы данных: Некоторые базы данных, такие как MongoDB и CouchDB, используют JSON для хранения и обмена данными между клиентом и сервером.

Важно отметить, что существуют различные библиотеки для работы с JSON на разных языках программирования, такие как nlohmann/json для C++, json-simple для Java или json.net для C#. Эти библиотеки облегчают работу с JSON, предоставляя удобные инструменты для парсинга, создания и изменения JSON-объектов и массивов.

**Функция read\_json\_file**

Эта функция принимает имя файла (filename) в качестве аргумента. Затем открывает этот файл с помощью объекта ifstream (input file stream). Если файл успешно открыт, программа считывает данные JSON из файла в объект типа json (j) и выводит его содержимое на экран. В случае неудачи при открытии файла выводится сообщение об ошибке.

**Функция write\_json\_file**

Функция принимает два аргумента: объект json (j) и имя файла (filename). Затем создает объект ofstream (output file stream) и открывает файл с указанным именем. Если файл успешно открыт, программа записывает содержимое объекта json (j) в файл с красивым форматированием (4 пробела на каждый уровень вложенности). В случае ошибки выводится сообщение об ошибке.

**Функция main**

В функции main реализован цикл, в котором пользователь может выбрать одно из пяти действий:

1. Ввод имени, фамилии и возраста - программа считывает введенные пользователем данные и сохраняет их в объекте json (person).

2. Запись данных в JSON файл - пользователь указывает имя файла, в который будут записаны данные из объекта person, и вызывается функция write\_json\_file.

3. Чтение JSON файла - пользователь указывает имя файла для чтения данных, и вызывается функция read\_json\_file.

4. Вывод текущих данных - программа выводит содержимое объекта person на экран.

0. Выход - программа завершает свою работу.

# Программный код:

#include <iostream> // Подключаем библиотеку для работы с потоками ввода-вывода

#include <nlohmann/json.hpp> // Подключаем библиотеку для работы с JSON

#include <fstream> // Подключаем библиотеку для работы с файлами

using json = nlohmann::json; // Создаем псевдоним для типа nlohmann::json для удобства

// Функция для чтения JSON-файла и вывода его содержимого на экран

void read\_json\_file(const std::string& filename) {

std::ifstream file(filename); // Открываем файл для чтения

if (file.is\_open()) { // Если файл успешно открыт

json j; // Создаем объект JSON

file >> j; // Считываем данные из файла в объект JSON

// Выводим считанный JSON объект с отступом в 4 пробела

std::cout << "Считанный JSON объект:\n" << j.dump(4) << std::endl;

}

else {

// Выводим сообщение об ошибке, если файл не удалось открыть

std::cerr << "Ошибка: Невозможно открыть файл " << filename << std::endl;

}

}

// Функция для записи JSON-объекта в файл

void write\_json\_file(const json& j, const std::string& filename) {

std::ofstream file(filename); // Открываем файл для записи

if (file.is\_open()) { // Если файл успешно открыт

file << j.dump(4); // Записываем JSON объект в файл с отступом в 4 пробела

}

else {

// Выводим сообщение об ошибке, если файл не удалось открыть

std::cerr << "Ошибка: Невозможно открыть файл " << filename << std::endl;

}

}

int main() {

setlocale(0, "rus"); // Устанавливаем локализацию для корректного вывода кириллицы

int choice; // Переменная для выбора действия пользователем

std::string filename; // Переменная для хранения имени файла

json person; // JSON объект для хранения данных о человеке

while (true) { // Бесконечный цикл для выбора действий пользователем

std::cout << "Выберите действие:\n"

<< "1. Ввод имени, фамилии и возраста\n"

<< "2. Запись данных в JSON файл\n"

<< "3. Чтение JSON файла\n"

<< "4. Вывод текущих данных\n"

<< "0. Выход\n";

std::cin >> choice; // Считываем выбор пользователя

switch (choice) { // Ветвление в зависимости от выбора пользователя

case 1: {

std::string name, surname;

int age;

// Вводим имя, фамилию и возраст с клавиатуры

std::cout << "Введите имя: ";

std::cin >> name;

std::cout << "Введите фамилию: ";

std::cin >> surname;

std::cout << "Введите возраст: ";

std::cin >> age;

// Сохраняем введенные данные в JSON объект

person["name"] = name;

person["surname"] = surname;

person["age"] = age;

break;

}

case 2: {

// Вводим имя файла для записи и вызываем функцию записи данных в файл

std::cout << "Введите имя файла для записи: ";

std::cin >> filename;

write\_json\_file(person, filename);

break;

}

case 3: {

// Вводим имя файла для чтения и вызываем функцию чтения данных из файла

std::cout << "Введите имя файла для чтения: ";

std::cin >> filename;

read\_json\_file(filename);

break;

}

case 4: {

// Выводим текущие данные из JSON объекта на экран с отступом в 4 пробела

std::cout << "Текущие данные:\n" << person.dump(4) << std::endl;

break;

}

case 0: // Завершаем работу программы

return 0;

default: // Выводим сообщение об ошибке при некорректном выборе

std::cerr << "Неверный выбор. Повторите попытку." << std::endl;

}

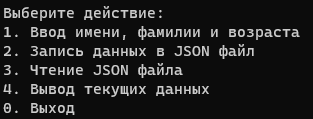
}

return 0;

}

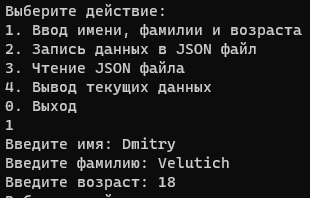
# Скриншоты и пояснения:

Встречает пользователя вот такое меню

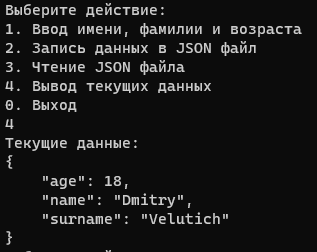


В пункте 1 вводим имя фамилию и возраст, после чего можем посмотреть информацию в пункте 4 (вывод).

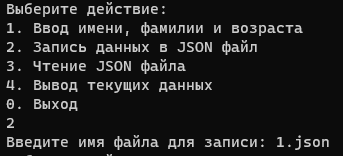
1)



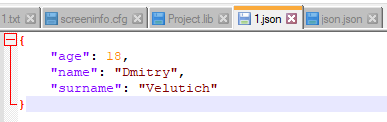
4)



Когда мы ввели данные в пункте 1, мы можем записать их во внешний json файл (пункт 2)

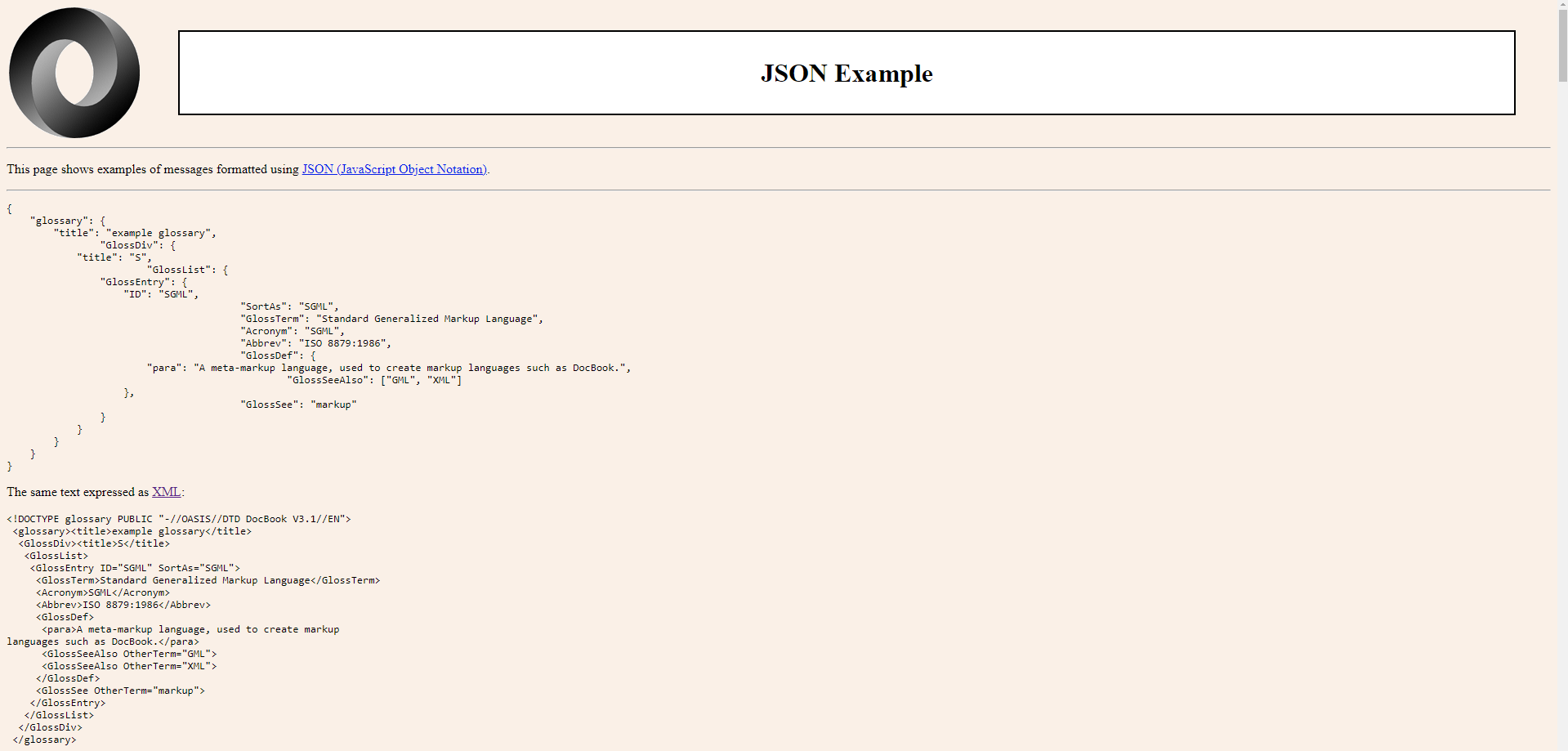


Находим в папке с проектом созданный программой файл 1.json, структура которого идентична тому, что мы вводили ранее



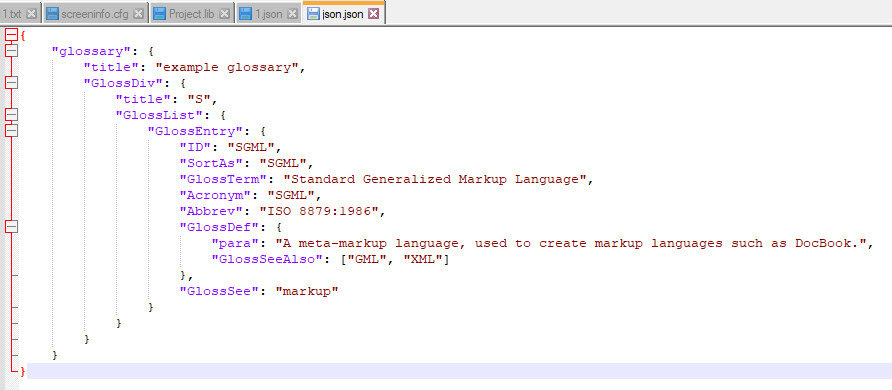
Программа также умеет считывать внешние json файлы, что будет крайне полезно в реальном мире, где большие объёмы данных передаются в формате JSON

Для примера считывания я взял первый попавшийся json код с просторов интернета

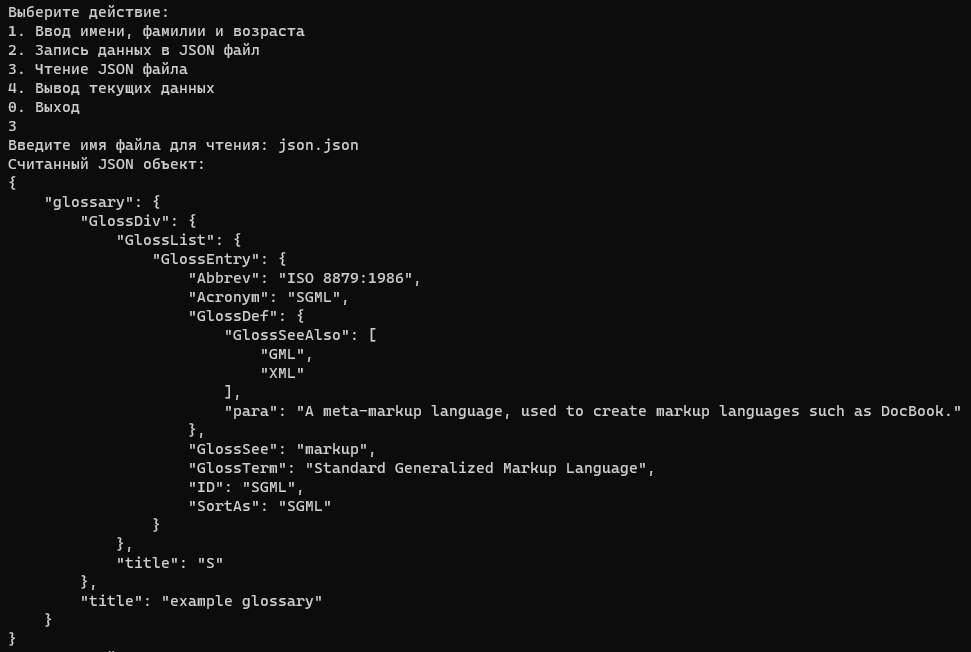


Скопировав все данные вручную в файл json.json мы заходим обратно в программу и видим, что данные у вывода программы и у первоначального источника кода идентичны

Файл json.json



Вывод считывания файла в программе



# Выводы:

Программа на C++ демонстрирует работу с форматом данных JSON с помощью библиотеки nlohmann/json. Пользователь может вводить данные, сохранять их в файлы, читать данные из файлов и выводить их на экран. Работа с JSON важна, так как данный формат широко используется для обмена данными между клиентом и сервером, а также в других областях, где требуется легкочитаемый и удобный для обработки формат данных.

Данный пример кода может быть полезным для изучения работы с библиотекой nlohmann/json на языке программирования C++, а также для освоения работы с файлами и базового взаимодействия с пользователем. Он может быть доработан и расширен с использованием других возможностей библиотеки, таких как обработка массивов, вложенных объектов и других типов данных, представленных в формате JSON. Таким образом, программа служит хорошей отправной точкой для изучения работы с JSON на C++ и может быть применена в реальных проектах для обработки и хранения данных в удобном формате.