Лабораторная работа №6 Погодный информер

[c++](https://neroid.ru/category/coding/cpp/) / От [admin](https://neroid.ru/author/admin/)

**Цель:**

1. Закрепить навыки разработки многофайловыx приложений;
2. Изучить способы работы с API web-сервиса;
3. Изучить процесс сериализации/десериализации данных.

Постановка задачи

Разработать приложение на языке С++ предназначенное для отображения прогноза погоды на ближайшие пять дней. В качестве источника данных о погоде использовать сервис <https://openweathermap.org/>. Вывод информации осуществлять при помощи web-интерфейса.

Приложение должно работать следующим образом:

1. Пользователь открывает браузер;
2. Вводит в адресную строку *localhost:3000*. Здесь *3000* — это номер порта;
3. В вашем приложении должен быть сервер «слушающий» 3000 порт. Он принимает запрос от браузера и вызывает функцию обработки запроса;

В функции обработки запроса:

1. Создаётся клиент, который посылает запрос на *openweathermap.org*;
2. Используя ответ, полученный от *openweathermap.org*, заполняется *html*-шаблон погодного виджета;
3. Получившаяся *html*-страница отдаётся браузеру;
4. Пользователь видит *html*-страницу заполненную информацией о погоде на ближайшие пять дней.

Ход работы

I. Подготовка серверной части:

1. Скачайте библиотеку для работы с сетью с <https://github.com/yhirose/cpp-httplib>;
2. Создайте консольное приложение;
3. В папке с проектом создайте папку *include*;
4. В папке *include* создайте папку *httplib*;
5. Из папки *cpp-httplib-master* скопируйте *httplib.h* в папку *include\httplib* своего проекта;
6. В проекте укажите папку с включаемыми файлами: *Проект -> Свойства -> вкладка С/С++ -> Общие*. В строке *Дополнительные каталоги включаемых файлов* добавьте путь к папке *include*;
7. В главный файл проекта подключите библиотеки следующим образом: *#include <httplib/httplib.h>*
8. Изучите пример сервера <https://github.com/yhirose/cpp-httplib#user-content-server-example>
9. Упростим данный код, чтобы он больше подходил под решаемую задачу в итоге получаем:

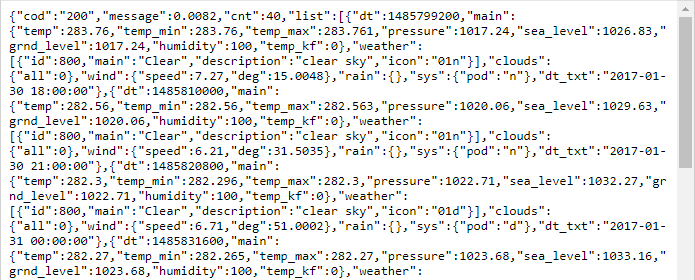


|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | #include <httplib/httplib.h>  using namespace httplib;    // В этой функции формируем ответ сервера на запрос  void gen\_response(const Request& req, Response& res) {    // Команда set\_content задаёт ответ сервера и тип ответа:    // Hello, World! - тело ответа    // text/plain - тип ответа (в данном случае обычный текст)    res.set\_content("Hello, World!", "text/plain");  }    int main(){    Server svr;                    // Создаём сервер (пока-что не запущен)    svr.Get("/", gen\_response);    // Вызвать функцию gen\_response если кто-то обратиться к корню "сайта"    svr.listen("localhost", 1234); // Запускаем сервер на localhost и порту 1234  } |

1. Соберите и проверьте код на работоспособность. Для этого запустите программу, затем перейдите в браузер и в адресной строке введите: *localhost:1234/*
2. Модифицируйте данный код, чтобы «сервер» слушал нужный порт.
3. Изучите следующие ссылки: [*localhost*](https://www.hostinger.ru/rukovodstva/shto-takoje-localhost/), [ещё](http://www.cyberforum.ru/post90174.html), [*port*](http://eqsash.com/articles/chto-takoe-setevoy-port-sayta-kompyutera-ili-drugogo-ustroystva-spisok-portov-po-umolchaniyu)

II. Подготовка к работе с сервисом openweathermap.org

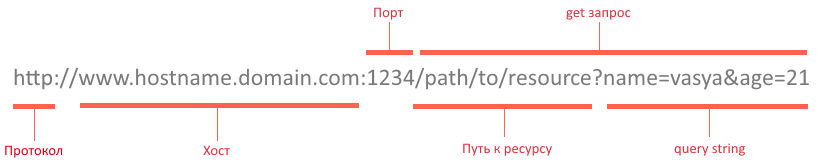
1. Перейдите на главную страницу сервиса [https://openweathermap.org](https://openweathermap.org/);
2. Зарегистрируйтесь и получите *API key* (он также придёт на почту);
3. Изучите пример использования *API key* <https://openweathermap.org/appid#use>;
4. В разделе *openweathermap.org -> API -> 5 day / 3 hour forecast* изучите формат запроса и формат ответа: <https://openweathermap.org/forecast5#parameter>;
5. По умолчанию, сервис возвращает температуру в Фаренгейтах, чтобы понять, как получить температуру в градусах Цельсия ознакомьтесь с разделом <https://openweathermap.org/forecast5#data>
6. Составьте и протестируйте запрос в браузере (для этого понадобится ваш *API Key*). В ответ вы должны получить текст в формате JSON.

  
*Рис. 1. Ответ сервиса openweathermap.org*

1. Для более глубокого понимания вопроса изучите следующие ссылки:
   * API: [Web API](https://ru.wikipedia.org/wiki/API#Web_API), [API key](https://en.wikipedia.org/wiki/Application_programming_interface_key), [OpenWeatherMap API — Overview [Видео]](https://youtu.be/SXsaB9TUfkk);
   * JSON: [О JSON](https://ru.wikipedia.org/wiki/JSON) [Красиво форматирует JSON](http://jsonprettify.com/).

III. Подготовка клиента для получения информации от openweathermap.org

1. В своём проекте, временно, закомментируйте код серверной части или создайте новый проект с аналогичными настройками;
2. Изучите пример клиента написанного с помощью библиотеки *httplib* <https://github.com/yhirose/cpp-httplib#user-content-client-example> и структуру *http(s)* запроса;

  
*Рис.2. Структура http(s) запроса*

В примере клиент осуществляет *get* запрос на сервер расположенный по адресу ***localhost*** (на скриншоте http://www.hostname.domain.com);  
***1234*** — номер порта, по которому обращаемся к хосту. Для *http* по умолчанию 80 (в браузере дописывается автоматически);  
Клиент просит ресурс ***/hi*** (на скриншоте */path/to/resource*);  
Так же на скриншоте присутствует *query string* — набор параметров, которые перечисляются после знака **?**. Параметры разделены символов **&**. *query string* является частью *get* запроса;  
В примере ***res->status == 200*** проверяет, удалось ли загрузить страницу;  
***res->body*** — это строка типа *std::string* содержащая тело ответа (например *html* страница).

1. Модифицируйте код клиента, чтобы отправить запрос составленный в пункте II.6. Ответ должен быть такой же как и в браузере.  
   **Внимание**:
   * У хоста НЕ указывайте протокол (*http/https*). Хост просто *api.openweathermap.org*;
   * Порт 80 (порт по умолчанию для *http*);
   * Всё что после хоста — это *Get* запрос

IV. Подготовка к работе с JSON

1. Скачайте библиотеку для работы с JSON с <https://github.com/nlohmann/json>;
2. Из папки *json-develop\include* скопируйте содержимое в папку *include* своего проекта;
3. В главный файл проекта подключите библиотеку следующим образом: *#include <nlohmann/json.hpp>*
4. Изучите пункт сериализации/десериализации строки в JSON [https://github.com/nlohmann/json#user-content-serialization—deserialization](https://github.com/nlohmann/json#user-content-serialization--deserialization)
5. Распарсите (десериализуйте) ответ, полученный от сервиса *openweathermap.org*, в JSON объект.
6. В качестве теста попробуйте отобразить на экран консоли город для которого получен прогноз. Преобразовать JSON в текст можно, например, вызвав метод *dump();* ( json\_obj["cod"].dump(); )

V. Подготовка шаблона виджета

1. Скачайте шаблон виджета [*informer\_template.html*](https://neroid.ru/wp-content/uploads/2020/02/informer_template.html) и поместите его в папку с проектом;
2. Итоговый заполненный виджет должен выглядеть похожим образом [*informer.html*](https://neroid.ru/wp-content/uploads/2020/02/informer.html) (для корректного отображения нужен интернет);
3. В программе загрузите файл *informer\_template.html* в строковую переменную.
4. Изучите шаблон виджета. В данном шаблоне есть ряд элементов требующих замены на конкретные данные, полученные от *openweathermap.org*. Все такие элементы расположены в фигурных скобках:
   * *{city.name}* — Название города, для которого составлен прогноз. Встречается 1 раз;Каждый из следующих элементов встречается 5 раз (по разу для каждого из 5 дней):
   * *{list.dt}* — дата в формате год-месяц-день. Например: 2019-01-21;
   * *{list.weather.icon}* — Имя иконки погоды. Например: 04d;
   * *{list.main.temp}* — температура в градусах Цельсия;  
     **Внимание!** В ответе от *openweathermap.org* вы получите прогноз погоды на 5 дней с интервалом в 3 часа, т.е. для каждого дня будет более 1 записи. В виджете предусмотрен только один блок на 1 день. Информацию для виджета берите из первой записи для каждого дня, остальные игнорируйте.

VI. Сборка итогового проекта

1. Раскомментируйте серверный код;
2. Процесс загрузки шаблона и запроса к *openweathermap.org* поместите в функцию *gen\_response*;
3. Там же заполните шаблон данными. Тут могут пригодится методы: [find](http://www.cplusplus.com/reference/string/string/find/), [substr](http://www.cplusplus.com/reference/string/string/substr/), [erase](http://www.cplusplus.com/reference/string/string/erase/);
4. Заполненный шаблон передайте первым параметром в функцию *set\_content*, в качестве второго параметра передайте *text/html*;
5. Протестируйте работу программы.

Что должно быть в отчёте

1. Ваш *API* ключ от *openweathermap.org*;
2. Полный код программы;
3. Скриншот экрана с демонстрацией работы программы (заполненный виджет).