САНКТ-ПЕТЕРБУРГСКОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

«ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ ГОРОДСКОГО ХОЗЯЙСТВА»

09.02.03 Программирование в компьютерных системах

код специальности наименование специальности

**Отчет о прохождении учебной практики УП 03**

**по профессиональному модулю ПМ 03**

**«Участие в интеграции программных модулей»**

Группа\_\_ИП-17-3\_\_\_\_\_\_

Студент Гаврилов Владимир Дмитриевич

Ф.И.О

Дата проведения УП 03 с «30» ноября 2020 по «12» декабря 2020

Оценка за прохождение практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики А.А.Коноплев (Ф.Ф. Лепешкин)

Ф.И.О

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 3](#_Toc58528718)

[ЗАДАНИЕ №1 4](#_Toc58528719)

[1.1. Основы выполнения задания 4](#_Toc58528720)

[1.2. Блок-схема алгоритма программы 5](#_Toc58528721)

[1.3. Код программы 6](#_Toc58528722)

[ЗАДАНИЕ №2 9](#_Toc58528723)

[2.1. Основы выполнения задания 9](#_Toc58528724)

[2.2. Блок-схема алгоритма программы 10](#_Toc58528725)

[2.3. Код программы 11](#_Toc58528726)

[ЗАДАНИЕ №3 14](#_Toc58528727)

[3.1. Основы выполнения задания 14](#_Toc58528728)

[3.2. Блок-схема алгоритма программы 16](#_Toc58528729)

[3.3. Код программы 16](#_Toc58528730)

## ВВЕДЕНИЕ

Аббревиатура API расшифровывается как «Application Programming Interface» (интерфейс программирования приложений, программный интерфейс приложения). WWW можно представить, как огромную сеть связанных серверов, на которых и хранится каждая страница.

При введении в адресную строку браузера www.facebook.com на удалённый сервер Facebook отправляется соответствующий запрос. Как только браузер получает ответ, то интерпретирует код и отображает страницу.

Каждый раз, когда пользователь посещает какую-либо страницу в сети, он взаимодействует с API удалённого сервера. API — это составляющая часть сервера, которая получает запросы и отправляет ответы.

## ЗАДАНИЕ №1

Тема: «Интеграция с сервером прогноза погоды».

Задание: Используя соответствующий API произвести обращение на сервер для получения информации о погодных условиях в заранее заданном городе с последующим выводом данных на экран пользователя.

### 1.1. Основы выполнения задания

Для выполнения поставленного задания будет использоваться «[Weather API - OpenWeatherMap](https://openweathermap.org/api)», данный API предоставляет множество вариаций вывода данных, для решения выбран вариант «Current Weather Data», позволяющий вывести текущую погоду в заданном городе.

Для начала работы необходимо произвести регистрацию на сайте, после которой вам будет предоставлен персональный ключ для обращения на сервер.

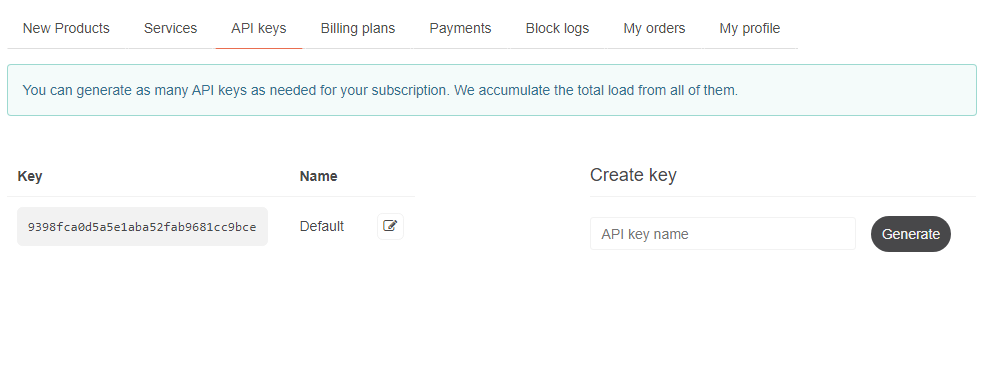


Рисунок 1 – Ключ

Далее необходимо взять строку обращения из документации API.

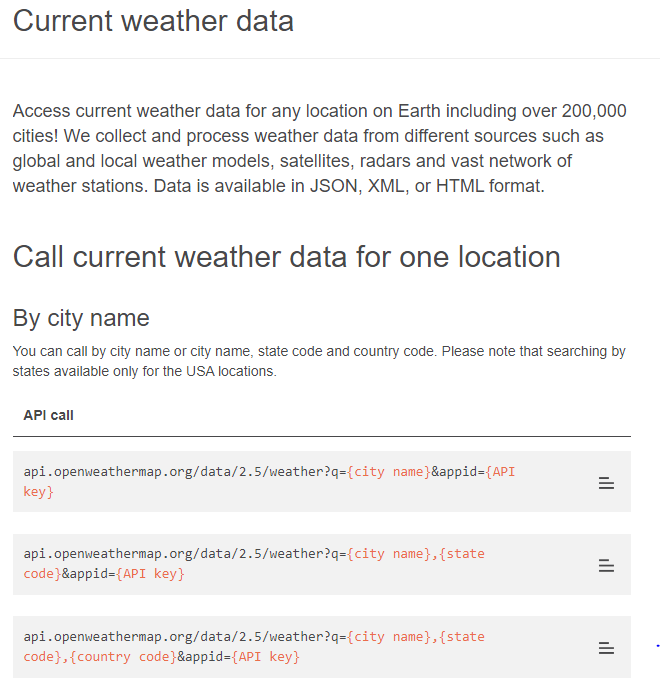


Рисунок 2 - Строка обращения

### 1.2. Блок-схема алгоритма программы

Программа будет следовать по разработанному алгоритму работы представленного ниже:

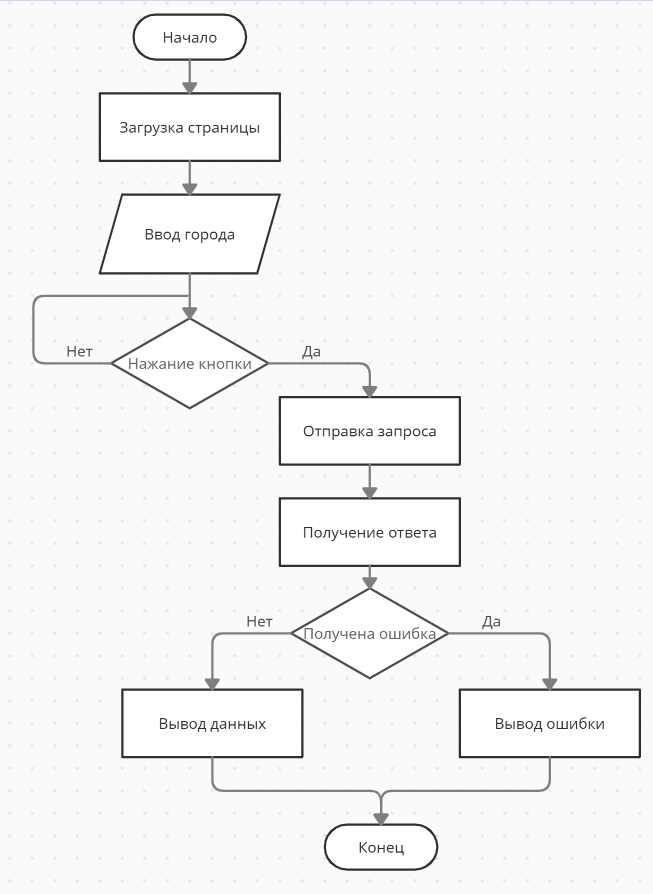


Рисунок 3 - Алгоритм работы

### 1.3. Код программы

Код программы:

<html>

<head>

<meta charset="UTF-8">

<title>Просмотр погоды</title>

<ul id="navbar">

<li><a href="weather.html">Погода</a></li>

<li><a href="news.html">Новости</a>

</li>

</ul>

<style type="text/css">

#header {

width: 100%;

height: 100px;

background-color: #C0C0C0;

text-align:center;

}

#navbar ul{

display: none;

background-color: OrangeRed;

position: absolute;

top: 100%;

}

#navbar li:hover ul { display: block; }

#navbar, #navbar ul{

margin: 0;

padding: 0;

list-style-type: none;

}

#navbar {

height: 50px;

background-color: black;

min-width: 800px;

position:fixed;

margin-top: -10px;

width:3000px;

}

#navbar li {

float: left;

position: relative;

height: 100%;

}

#navbar li a {

display: block;

padding: 6px;

width: 160px;

color: White;

font-size: 25px;

text-decoration: none;

text-align: center;

}

#navbar ul li { float: none; }

#navbar li:hover { background-color: OrangeRed; }

#navbar ul li:hover { background-color: black; }

#marginbox {

height: 50px;

width:100%;

}

.input, .data {

padding:20px;

border: 1px solid Black;

width: 800px;

margin : 10px auto;

height: 50px;

}

.data {

height: 450px;

width: 600px;

}

.input input[type="text"], .data input[type="text"]{

border: solid 1px #E5E5E5;

margin:10px 10px 10px 10px;

padding: 9px;

display:block;

font-size:16px;

width:70%;

background: #FFFFFF;

float:left;

box-shadow: 0px 0px 8px #f0f0f0;

}

.data input[type="text"]{

width:90%;

}

.input input[type="submit"]{

background: #e3e3e3;

border: 1px solid #ccc;

color: #333;

font-family: "Trebuchet MS", "Myriad Pro", sans-serif;

font-size: 14px;

font-weight: bold;

padding: 8px 0 9px;

text-align: center;

width: 125px;

height: 38px;

float:right;

margin:10px 10px 10px 10px;

}

.data label{

display:block;

padding: 0px 30px 0px 10px;

margin: 10px 0px 0px 0px;

text-align: center;

font-size: 20px;

float:left;

}

</style>

</head>

<body>

<div id="marginbox">

</div>

<div class="input">

<input type="text" placeholder="Название города" class="city">

<input type="submit" value="Прогноз" class="submit">

</div>

<div class="data">

<h1 class="name" id="name">City</h1>

<label>Температура:</label>

<input type="text" class="temp">

<label>Облачность:</label>

<input type="text" class="clouds">

<label>Ощущается как:</label>

<input type="text" class="feel">

<label>Ветер:</label>

<input type="text" class="wind">

</div>

<script>

var input = document.querySelector('.city');

var main = document.querySelector('#name');

var temp = document.querySelector('.temp');

var clouds = document.querySelector('.clouds');

var feel = document.querySelector('.feel');

var wind = document.querySelector('.wind');

var button = document.querySelector('.submit');

button.addEventListener('click', function(name){

fetch('https://api.openweathermap.org/data/2.5/weather?q='+input.value+'&appid=9398fca0d5a5e1aba52fab9681cc9bce')

.then(response => response.json())

.then(data => {

console.log(data);

var tempValue = data['main']['temp'];

var nameValue = data['name'];

var cloudsValue = data['weather'][0]['description'];

var feelValue = data['main']['feels\_like'];

var windValue = data['wind']['speed'];

main.value = nameValue;

temp.value = Math.round(tempValue-273,15)+"°C";

clouds.value = cloudsValue;

feel.value = Math.round(feelValue-273,15)+"°C";;

wind.value = windValue+"м/с";

input.value ="";

})

.catch(err => alert("Город не найден"));

});

</script>

</body>

</html>

## ЗАДАНИЕ №2

Тема: «Интеграция с новостным агрегатором».

Задание: Используя соответствующий API произвести обращение на сервер для получения информации о последних новостях с последующим выводом данных на экран пользователя.

### 2.1. Основы выполнения задания

Для выполнения поставленного задания будет использоваться «[News API](https://newsapi.org/)», данный API предоставляет собой возможность вывода новостей с различными фильтрами, новости от различных издательств, новости определенной страны, новости определенной тематики и т. д.

Для начала работы необходимо произвести регистрацию на сайте, после которой вам будет предоставлен персональный ключ для обращения на сервер.

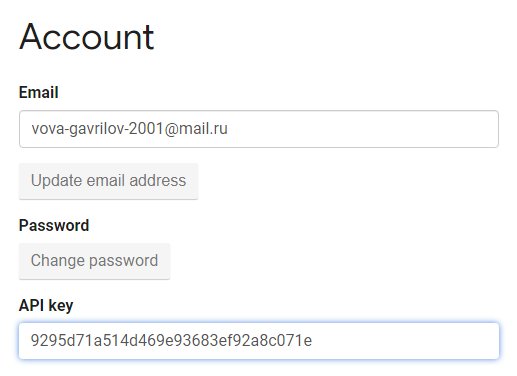


Рисунок 4 – Ключ

Далее необходимо взять строку обращения из документации API.

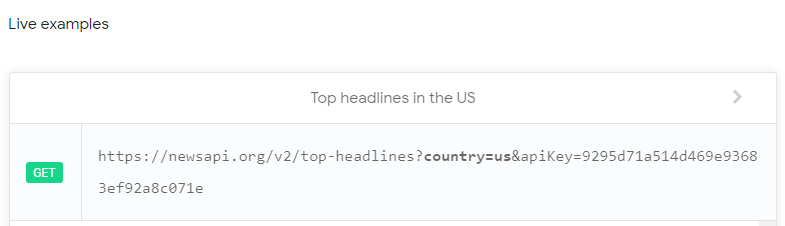


Рисунок 5 - Строка обращения

### 2.2. Блок-схема алгоритма программы

Программа будет следовать по разработанному алгоритму работы представленного ниже:

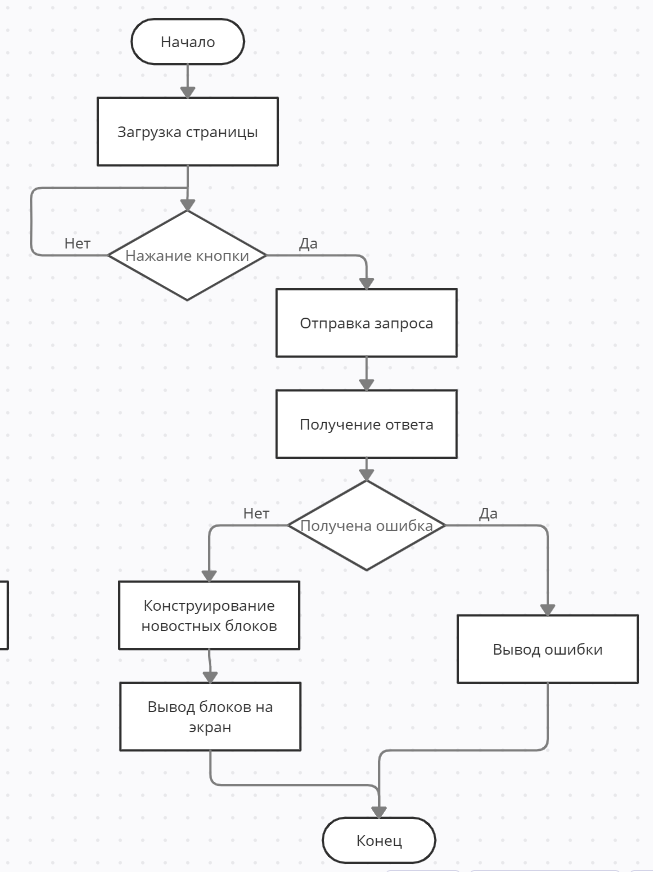


Рисунок 6 - Алгоритм работы

### 2.3. Код программы

Код программы:

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Просмотр новостей</title>

<ul id="navbar">

<li><a href="weather.html">Погода</a></li>

<li><a href="news.html">Новости</a>

</li>

</ul>

<style type="text/css">

#header {

width: 100%;

height: 100px;

background-color: #C0C0C0;

text-align:center;

}

#navbar ul{

display: none;

background-color: OrangeRed;

position: absolute;

top: 100%;

}

#navbar li:hover ul { display: block; }

#navbar, #navbar ul{

margin: 0;

padding: 0;

list-style-type: none;

}

#navbar {

height: 50px;

background-color: black;

min-width: 800px;

position:fixed;

margin-top: -10px;

width:3000px;

}

#navbar li {

float: left;

position: relative;

height: 100%;

}

#navbar li a {

display: block;

padding: 6px;

width: 160px;

color: White;

font-size: 25px;

text-decoration: none;

text-align: center;

}

#navbar ul li { float: none; }

#navbar li:hover { background-color: OrangeRed; }

#navbar ul li:hover { background-color: black; }

#marginbox {

height: 50px;

width:100%;

}

.input, .data {

padding:20px;

width: 800px;

margin : 10px auto;

height: 50px;

}

.data {

width: 600px;

}

.input {

border: 1px solid Black;

}

.data input[type="text"], .data textarea{

border: solid 1px #E5E5E5;

margin:10px 10px 10px 10px;

padding: 9px;

display:block;

font-size:16px;

width:70%;

background: #FFFFFF;

float:left;

box-shadow: 0px 0px 8px #f0f0f0;

}

.data input[type="text"]{

width:90%;

}

.input input[type="submit"]{

background: #e3e3e3;

border: 1px solid #ccc;

color: #333;

font-family: "Trebuchet MS", "Myriad Pro", sans-serif;

font-size: 14px;

font-weight: bold;

padding: 8px 0 9px;

text-align: center;

width: 100%;

height: 38px;

margin:10px 10px 10px 10px;

}

.data label{

display:block;

padding: 0px 30px 0px 10px;

margin: 10px 0px 0px 0px;

text-align: left;

font-size: 20px;

float:left;

width: 100%;

}

.data textarea{

font-size: 16px;

width: 90%;

height: 130px;

}

</style>

<head>

<body>

<div id="marginbox">

</div>

<div class="input">

<input type="submit" value="Узнать новости" class="submit">

</div>

<div id="data" class="data">

<h1 class="news" id="news">Новости</h1>

</div>

<script>

//9295d71a514d469e93683ef92a8c071e

var button = document.querySelector('.submit');

button.addEventListener('click', function(name){

fetch('http://newsapi.org/v2/top-headlines?country=ru&apiKey=9295d71a514d469e93683ef92a8c071e')

.then(response => response.json())

.then(data => {

for (var i = 0; i < data.articles.length; i++)

{

var div=document.getElementById('data'); //получение элемента страницы

var field=document.createElement('fieldset');//создание группировки

title=document.createElement('label');//создание абзаца

title.innerHTML='Тема:';

field.appendChild(title);//устанавливаем родителя

titledata=document.createElement('input');//создание абзаца

titledata.type = "text";

titledata.value=data.articles[i].title;

field.appendChild(titledata);//устанавливаем родителя

desc=document.createElement('label');

desc.innerHTML='Описание:';

field.appendChild(desc);

descdata=document.createElement('textarea');

descdata.value=data.articles[i].description;

field.appendChild(descdata);

url=document.createElement('label');

url.innerHTML='Ссылка:';

field.appendChild(url);

urldata=document.createElement('input');

urldata.type = "text";

urldata.value=data.articles[i].url;

field.appendChild(urldata);

div.appendChild(field);

}

console.log(data);

})

.catch(err => console.log(err));

});

</script>

</body>

</html>

## ЗАДАНИЕ №3

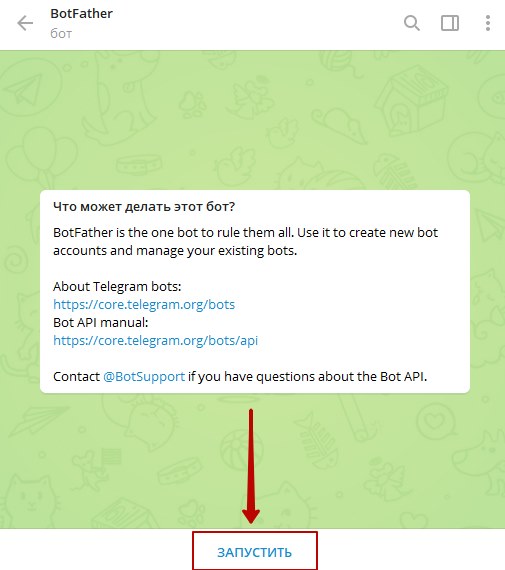
Тема: «Интеграция с Telegram».

Задание: Используя соответствующий API произвести подключение к серверу для реализации возможности общения с ботом Telegram.

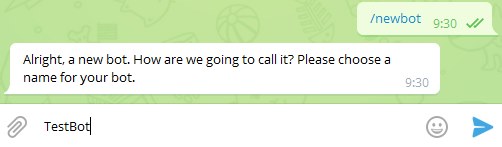
### 3.1. Основы выполнения задания

Для выполнения поставленного задания необходимо создать бота и получить специальный ключ для работы с ним, данный бот будет отвечать на сообщения пользователя предоставляя какую-либо информацию.

1. Для начала работы необходимо произвести регистрацию в Telegram;
2. Перед созданием бота нужно зарегистрировать его в BotFather. Это делается для того, чтобы ваш бот мог работать в Telegram;



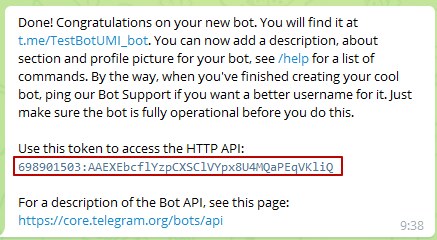
1. В окне чата отобразятся все доступные команды. Нас интересует /newbot. Можете это ввести вручную или же кликнуть по соответствующей записи прямо в командном списке;
2. BotFather предложит вам выбрать имя для своего будущего бота;



1. Затем необходимо придумать username боту. Он должен быть уникальным. В конце обязательно нужно указать приставку \_bot;



1. После успешной регистрации BotFather пришлет вам сообщение с токеном, который нужно записать где-нибудь у себя и сохранить.



Далее необходимо разработать программу которая будет обрабатывать сообщения от пользователя и отвечать какой-либо информацией, в данном случае бот отправит в ответ на любое сообщение текущий день недели.

### 3.2. Блок-схема алгоритма программы

Программа будет следовать по разработанному алгоритму работы представленного ниже:

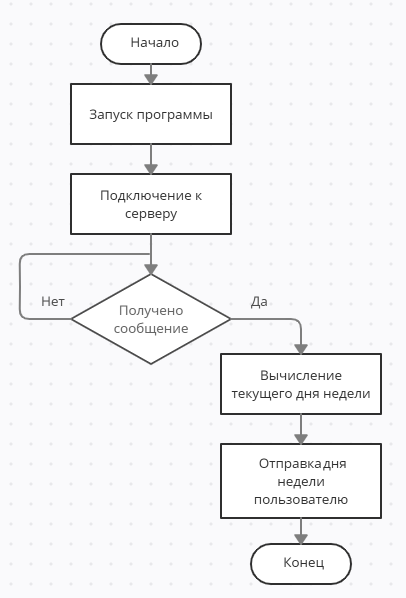


Рисунок 7 - Алгоритм работы

### 3.3. Код программы

Код программы:

using MihaZupan; //proxy

using System;

using Telegram.Bot;

using Telegram.Bot.Args;

namespace ConsoleApp2

{

class Program

{

private static ITelegramBotClient botClient;

static void Main(string[] args)

{

//бот будет выводить день недели в ответ на любое сообщение

var proxy = new HttpToSocks5Proxy("207.97.174.134", 1080);

botClient = new TelegramBotClient("1400277860:AAFJJcFooQm1LlcZaH\_SWoeeGz3EZVUJY1k", proxy);

var me = botClient.GetMeAsync().Result;

Console.WriteLine($"name: {me.FirstName}"); //проверка подключения

botClient.OnMessage += BotMessage;

botClient.StartReceiving();

Console.ReadKey();

botClient.StopReceiving();

}

private async static void BotMessage(object sender, MessageEventArgs e)

{

var text = e.Message.Text;

if (text == null)

return;

DateTime thisDay = DateTime.Today;

await botClient.SendTextMessageAsync(chatId: e.Message.Chat.Id, text: $"Сегодня {thisDay.ToString("dddd")}").ConfigureAwait(false);

}

}

}