Министерство образования и науки Кыргызской Республики

Кыргызский Государственный Технический Университет им. И. Раззакова

Факультет Информационной технологии

Кафедра Программное Обеспечение Компьютерных Систем

**Руководство программиста**

**Тема: «Телеграмм бот»**

Выполнил: студент гр.ПИ-1-15 Оконов Урмат

Проверилa: Сабаева К.К.

Бишкек 2018 г.

**СОДЕРЖАНИЕ**

[1 Условия применения программы 3](#_Toc247918475)

[1.1. Требования к техническим средствам 3](#_Toc247918476)

[1.2. Требования к общему программному обеспечению (ОПО) 3](#_Toc247918477)

[2 Характеристика программы 4](#_Toc247918478)

[3 Обращение к программе 4](#_Toc247918479)

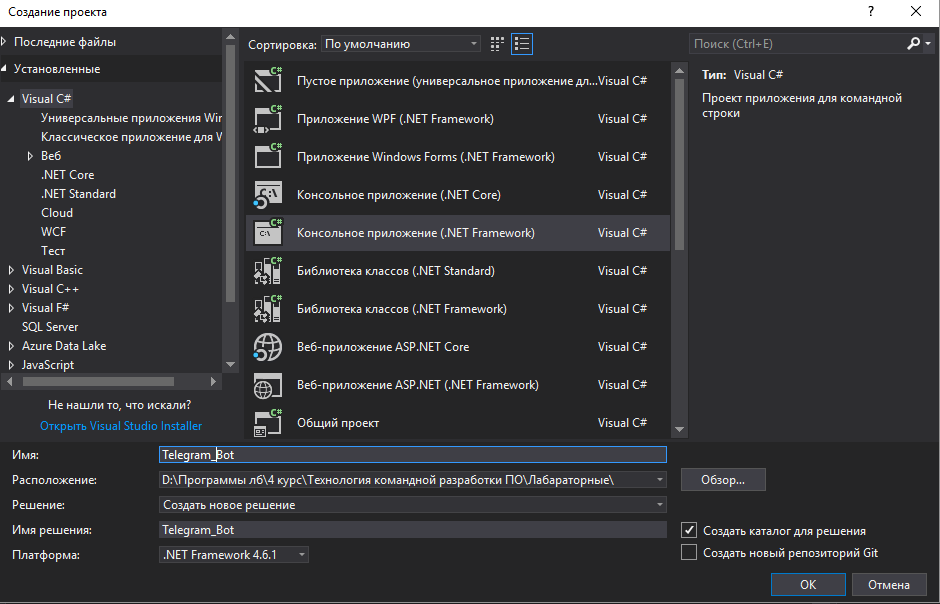
[4 Входные и выходные данные 6](#_Toc247918480)

[5 Сообщения 8](#_Toc247918481)

1. **Условия применения программы**
   1. **Требования к техническим средствам**

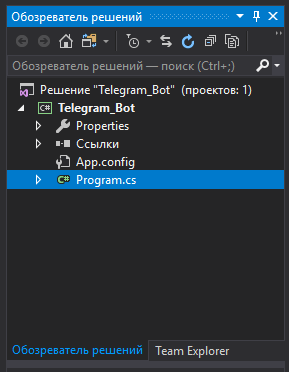
Для разработчиков необходимо установить IDE Visual Studio (либо другое подобное IDE, в котором возможны подключения пакета Telegram). В данном случае вся установка будет происходит в Visual Studio 2017.

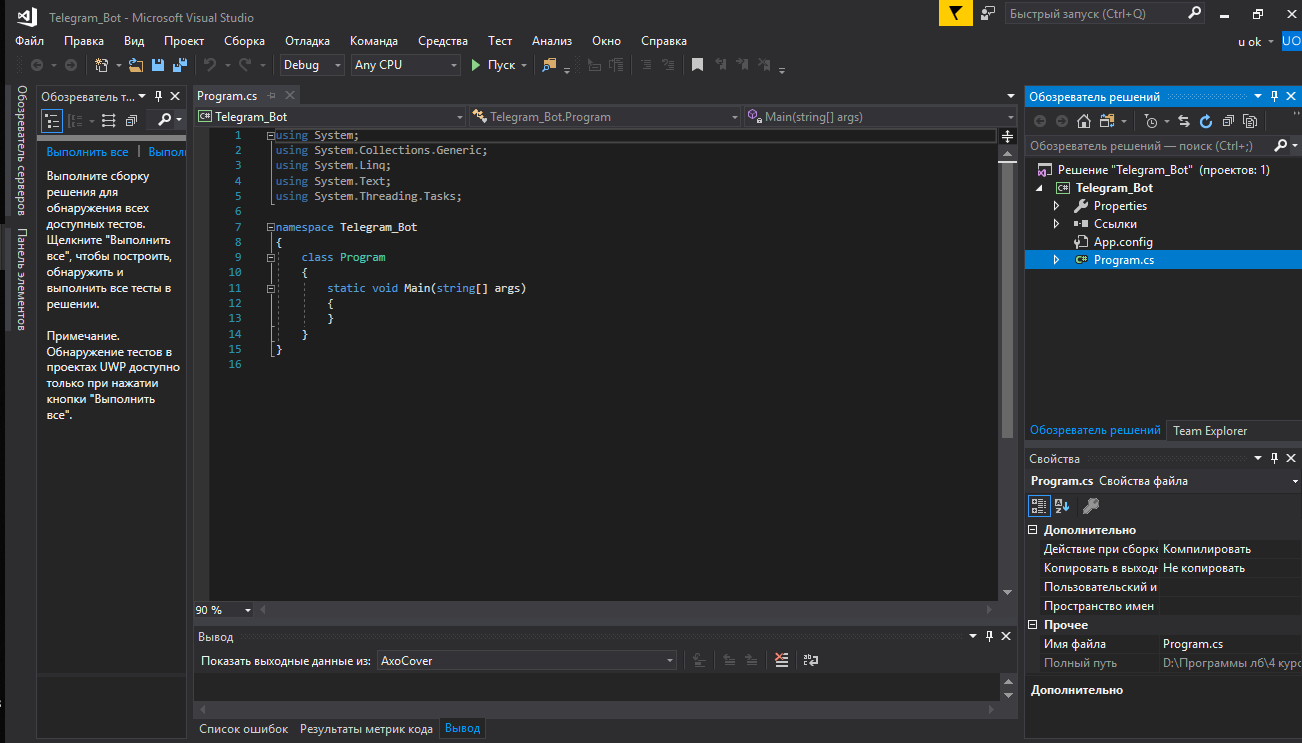
Шаг 1.Создаем проект для этого переходим по вкладке Файл->Создать->Проект (или нажатием комбинации Ctr+Shift+N).



Шаг 2.Выбираем приложение Консольное приложение в Visual C # и нажимаем ОК.

Шаг 3.После открытия проекта, открываем файл Program.cs:





Шаг 4. Добавляем следующие пакеты перед namespace Название\_Проекта:

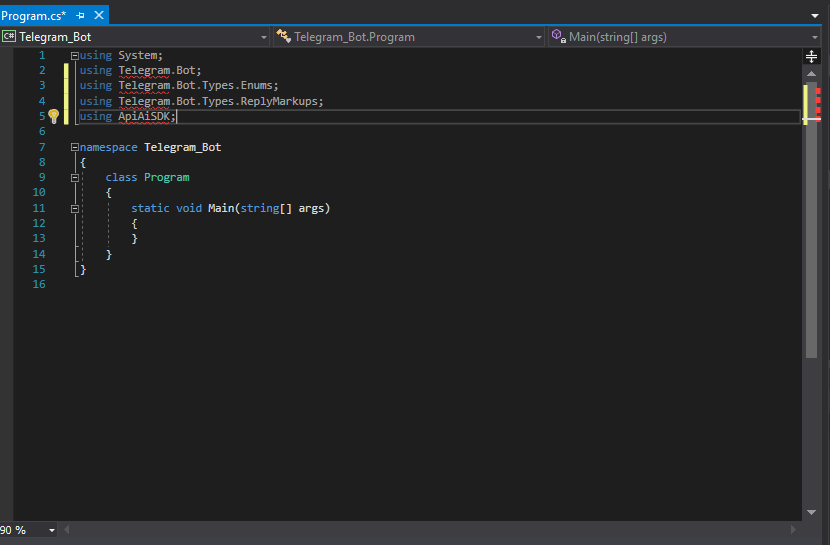
using Telegram.Bot;

using Telegram.Bot.Types.Enums;

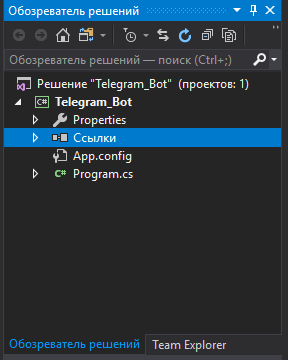
using Telegram.Bot.Types.ReplyMarkups;

using ApiAiSDK;

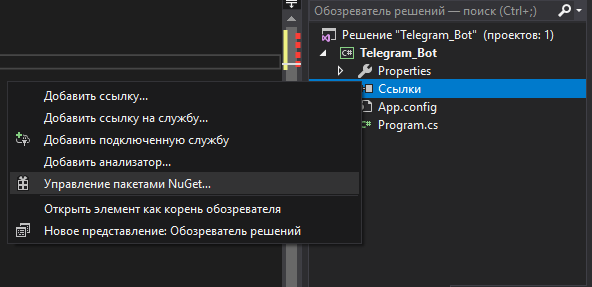
Если у вас не установлены пакеты Telegram, то программа будет выводить ошибку, как на картинке снизу.



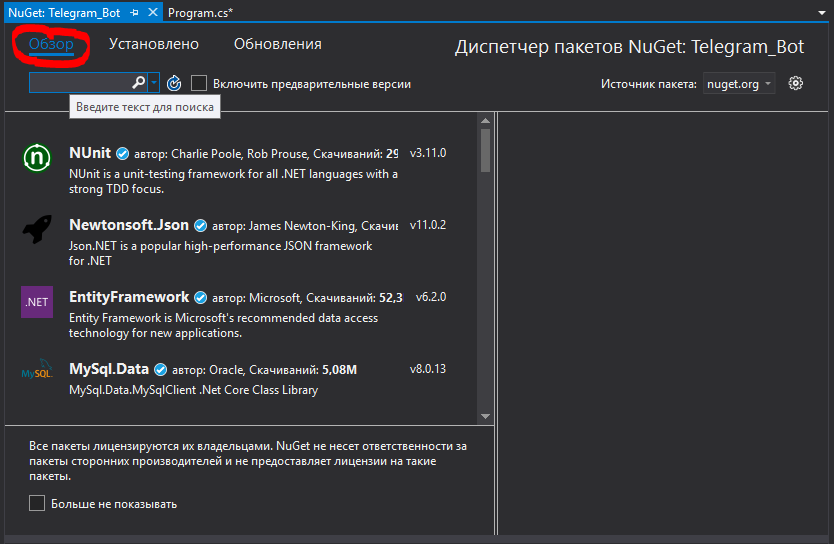
Шаг 5.Чтобы исправить эти ошибки нажмите правой кнопкой на Ссылки в Обозревателе решений



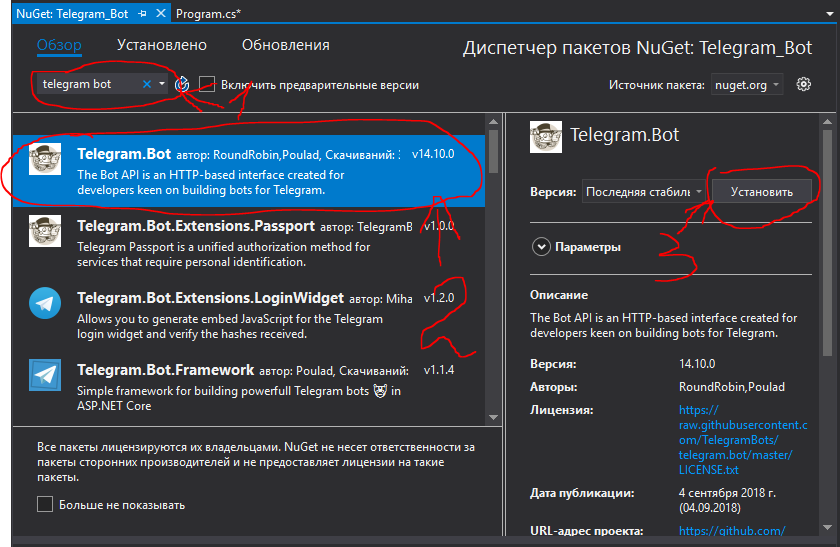
Шаг 6.Нажмите Управление пакетами NuGet



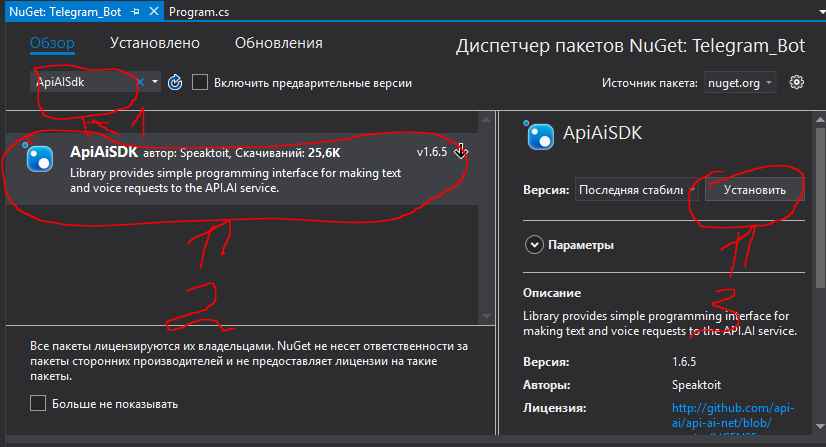
Шаг 7.Откроется окно управления пакетами NuGet в котором вам надо перейти к вкладке Обзор, как на картинке снизу:



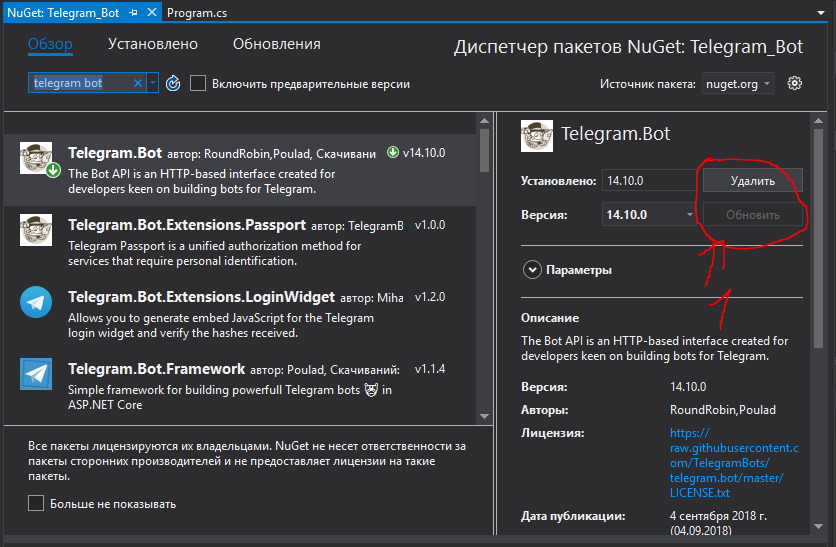
Шаг 8.В поиске пропишите «telegram bot» (1) и выберите пакет Telegram.Bot (2) и нажмите на кнопку установить (3):



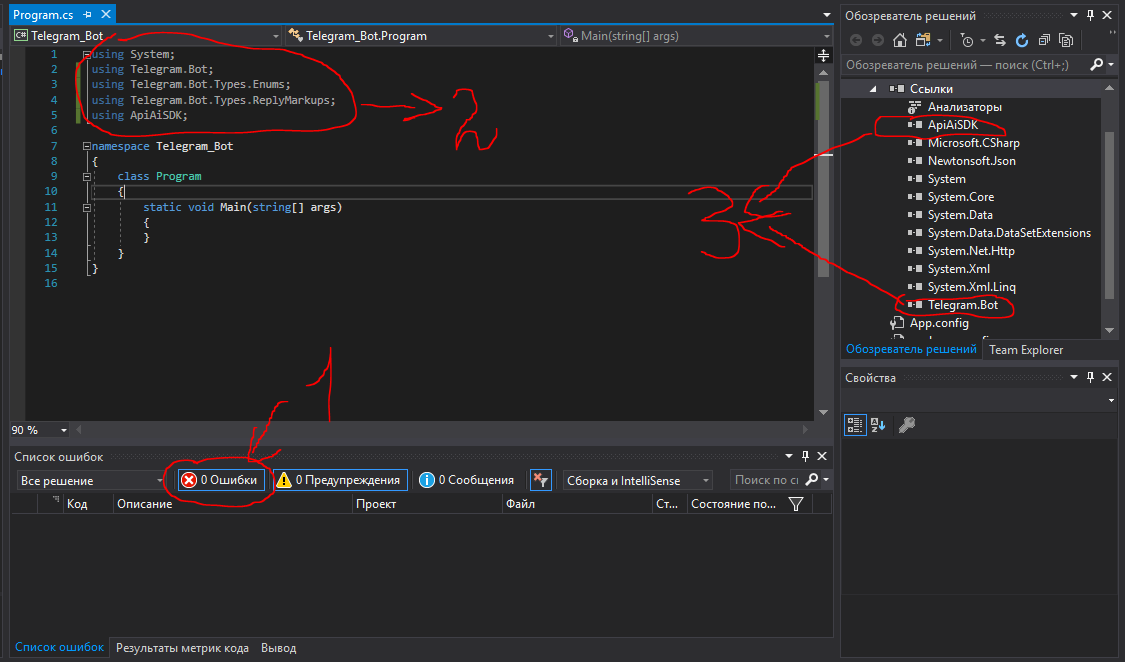
Так же необходимо установить Api SDK, который скачивается таким же образом. Вы пишете в поиске ApiAISdk (1) и выбираете пакет ApiAiSDK (2) и нажимаете установить (3):



Шаг 9.После окончания установки, где в окне выходных данных появится сообщение об окончании установки, и соответственно у вас появится возможность удалить или обновить пакет Telegram.Bot (1), вы можете смело закрывать окно управления пакетами NuGet :



Шаг 10.После установки пакетов вы можете заметить, что ошибок больше нет (1) и строки кода подключения е подчеркнуты красным, а все необходимые ссылки уже установлены и присутствуют в выпадающем меню Ссылка (3):



После этого вы можете использовать методы и функции Телеграмм бота

1. Характеристика программы

Боты — специальные аккаунты в Telegram, созданные для того, чтобы автоматически обрабатывать и отправлять сообщения. Пользователи могут взаимодействовать с ботами при помощи сообщений, отправляемых через обычные или групповые чаты. Логика бота контролируется при помощи HTTPS запросов к нашему [API для ботов](https://tlgrm.ru/docs/bots/api). В данной программе система открывает консоль, который является сервером, с помощью которого бот запускается, и вы можете работать с ним. Программа использует id бота, который был выдан BotFather`ом и бот целиком и полностью зависит от логики программы.

1. Обращение к программе

Внешнее управление содержимым системы осуществляется посредством приложения Telegram. Внутренним управлением занимается код программы и отвечает за весь функционал бота.

Процесс запуска бота

Чтобы работать программисту с программой необходимо выполнить ряд шагов:

1. Запустить программу
2. Ожидать запуска консоли
3. Просмотреть логику работы в приложении Telegram в чате с ботом

Входящие запросы будут приниматься от любых пользователей, которые начали работу с ботом.

1. Входные и выходные данные

В данном разделе будет объяснятся элементы входных данных и ожидающие выходные данные

Входными данными будет:

1.Id Телеграмм бота

2.Команды бота

3.Сообщения бота

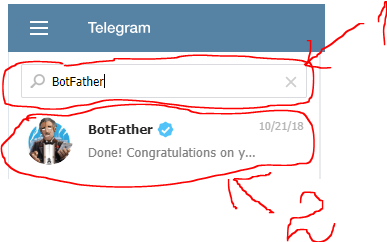
Выходные данными будут являться:

1.Именя подключившихся пользователей

2.Просмотр использованных команд

Рассмотрим пример работы Telegram бота:

После установки пакетов Telegram вам необходимо зарегистрироваться в приложении Telegram путем скачивания его и прохождения регистрации внутри приложения Telegram. Вы можете работать с телеграмом как и на компьютере так и в телефоне. Затем потребуется зарегистрировать и получить его уникальный id бота, являющийся одновременно и токеном. Для этого в Telegram существует специальный бот —[@BotFather](https://telegram.me/botfather). Пишем в поиске BotFather (1) и затем нажимаем на найденный бот как на картинке снизу (2):



Пишем ему /start и получаем список всех его команд.   
Первая и главная — /newbot — отправляем ему и бот просит придумать имя нашему новому боту. Единственное ограничение на имя — в конце оно должно оканчиваться на «bot». В случае успеха BotFather возвращает токен бота и ссылку для быстрого добавления бота в контакты.

После создания бота можно и программировать.

using System;

using Telegram.Bot;

using Telegram.Bot.Types.Enums;

using Telegram.Bot.Types.ReplyMarkups;

using ApiAiSDK;

namespace Telegram\_Bot\_Console

{

class Program

{

static TelegramBotClient Bot;

static ApiAi apiAi;

static void Main(string[] args)

{

Bot = new TelegramBotClient("638687925:AAH1-dP7iqNfQ0UsBFcJEWqqj274oIVmPWw");

Bot.OnMessage += Bot\_OnMessageReceived;

Bot.OnCallbackQuery += Bot\_OnCallbackQueryReceived;

var me = Bot.GetMeAsync().Result;

Console.WriteLine(me.FirstName);

Bot.StartReceiving();

Console.ReadLine();

Bot.StartReceiving();

}

private static async void Bot\_OnCallbackQueryReceived(object sender, Telegram.Bot.Args.CallbackQueryEventArgs e)

{

string buttonText = e.CallbackQuery.Data;

string name = $"{e.CallbackQuery.From.FirstName} {e.CallbackQuery.From.LastName}";

Console.WriteLine($"{name} нажал кнопку {buttonText}");

if (buttonText == "Картинка")

{

await Bot.SendTextMessageAsync(e.CallbackQuery.From.Id, "Сслыка на фото");

}

else if (buttonText == "Видео")

{

await Bot.SendTextMessageAsync(e.CallbackQuery.From.Id, "Ссылка на видео");

}

await Bot.AnswerCallbackQueryAsync(e.CallbackQuery.Id, $"Вы нажали кнопку {buttonText}");

}

private static async void Bot\_OnMessageReceived(object sender, Telegram.Bot.Args.MessageEventArgs e)

{

var message = e.Message;

if (message == null || message.Type != MessageType.Text)

return;

string name = $"{message.From.FirstName} {message.From.LastName}";

Console.WriteLine($"{name} отправил сообщение: '{message.Text}'");

switch (message.Text)

{

case "/start":

string text =

@"Список команд:

/start - запуск бота

/inline - вывод меню

/keyboard - вывод клавиатуры

";

await Bot.SendTextMessageAsync(message.From.Id, text);

break;

case "/inline":///это для вывод меню

var inlineKeyboard = new InlineKeyboardMarkup(new[]

{

new []

{

InlineKeyboardButton.WithUrl("Instagram","Instagram ccskrf"),

InlineKeyboardButton.WithUrl("Telegram","Ссылка на телеграм"),

},

new[]

{

InlineKeyboardButton.WithCallbackData("Картинка"),

InlineKeyboardButton.WithCallbackData("Видео")

}

});

await Bot.SendTextMessageAsync(message.From.Id, "Выберите пункт меню",

replyMarkup: inlineKeyboard);

break;

case "/keyboard":

var replyKeyboard = new ReplyKeyboardMarkup(new[]

{

new[]

{

new KeyboardButton("Привет"),

new KeyboardButton("Как дела?")

},

new[]

{

new KeyboardButton("Контакт") { RequestContact = true },

new KeyboardButton("Геолакация") {RequestLocation = true }

}

});

await Bot.SendTextMessageAsync(message.Chat.Id, "Сообщение",

replyMarkup: replyKeyboard);

break;

default:

var response = apiAi.TextRequest(message.Text);

string answer = response.Result.Fulfillment.Speech;

if (answer == "")

answer = "Я не понял что ты сказал";

await Bot.SendTextMessageAsync(message.From.Id, answer);

break;

}

}

}

}

После этого кода можете запускать консоль и писать боту сообщения путем написании команда /start.

1. Сообщения

В ходе выполнения программы, программисту выдаются сообщения. Здесь приводится описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

Оператор await можно использовать только в методах с модификатором async. Попробуйте пометить этот метод модификатором async и изменить тип его возвращаемого значения на Task.

Решение: Проверить асинхронность метода, в котором прописаны команды await