Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рязанский государственный радиотехнический университет

имени В.Ф.Уткина»

Кафедра ЭВМ

Отчет по лабораторной работе №11

на тему

«Разработка Telegram бота»

по дисциплине  
**«Визуальное программирование»**

Выполнили:

Студенты группы №140

Бригада 5

Сафаров Д. А.

Тимохин Е. С.

Проверили:  
ст. преп. Хизриева Н.И.

ст. преп. Бастрычкин А.С.

**Цель работы**

Получение навыков разработки ботов для мессенджеров на языке C#.

**Задание**

Создать Telegram бота со следующим функционалом:

– Отвечает на сообщение «/start»;

– Присылает случайную картинку в ответ на сообщение «/picture»;

– Принимает строку типа число операция число и возвращает результат операции. В качестве операции могут быть сложение, вычитание и умножение. Числа целые. Предусмотреть удаление лишних пробелов;

– Принимает изображение и возвращает его перевернутым на 180 градусов.

**Практическая часть**

Код программы представлен в приложении А. Результат её работы представлен на рисунке 1.



Рисунок 1 – Результаты работы программы

Приложение А. Текст программы

**TgBot.cs**

using System;

using System.IO;

using System.Threading;

using System.Threading.Tasks;

using Telegram.Bot;

using Telegram.Bot.Polling;

using Telegram.Bot.Types;

using Telegram.Bot.Exceptions;

using Newtonsoft.Json;

using Telegram.Bot.Types.Enums;

using Telegram.Bot.Requests;

using static System.Net.WebRequestMethods;

namespace TgBotLogic

{

public class TgBot

{

static string tgBotToken = System.IO.File.ReadAllText(@"..\..\..\..\Токен бота.txt");

static ITelegramBotClient bot = new TelegramBotClient(tgBotToken);

static async Task HandleUpdateAsync(ITelegramBotClient botClient,

Update update,

CancellationToken cancellationToken)

{

//Console.WriteLine(JsonConvert.SerializeObject(update));

if (update.Type == Telegram.Bot.Types.Enums.UpdateType.Message)

{

var message = update.Message;

Console.Write("\n" + message?.From.Username + " >> ");

if(message?.Type == MessageType.Text)

{

string messageText = message?.Text?.ToLower();

Console.WriteLine(messageText);

if (messageText == "/start")

{

await botClient.SendTextMessageAsync(message.Chat, "Даров");

Console.WriteLine("Bot << Даров");

return;

}

if (messageText == "/picture")

{

using (var stream = System.IO.File.Open(GetRandomPicture(), FileMode.Open))

{

await botClient.SendTextMessageAsync(message.Chat, "Пикчу? Ща буит)");

await botClient.SendPhotoAsync(message.Chat, InputFile.FromStream(stream));

Console.WriteLine("Bot << Пикчу? Ща буит)");

Console.WriteLine("Bot << \*картинка\*");

}

return;

}

if (messageText.StartsWith("/calculate"))

{

await botClient.SendTextMessageAsync(message.Chat, "Посчитать? Ща буит...");

Console.WriteLine("Bot << Посчитать? Ща буит...");

// Удаление лишних пробелов

string input = messageText.Replace("/calculate", "").Replace(" ", "");

// Разделение строки на операнды и операцию

string[] parts = input.Split(new char[] { '+', '-', '\*' });

if (parts.Length != 2)

{

await botClient.SendTextMessageAsync(message.Chat, "Бро, ты чёт не то написал");

Console.WriteLine("Bot << Бро, ты чёт не то написал");

return;

}

int operand1 = int.Parse(parts[0]);

int operand2 = int.Parse(parts[1]);

char operation = input[parts[0].Length];

// Выполнение операции

int result = 0;

switch (operation)

{

case '+':

result = operand1 + operand2;

break;

case '-':

result = operand1 - operand2;

break;

case '\*':

result = operand1 \* operand2;

break;

default:

await botClient.SendTextMessageAsync(message.Chat, "Бро, ты чёт не то написал");

Console.WriteLine("Bot << Бро, ты чёт не то написал");

return;

}

await botClient.SendTextMessageAsync(message.Chat, "Держи ответ, малой: " + result);

Console.WriteLine("Bot << Держи ответ, малой: " + result);

return;

}

}

if (message?.Type == MessageType.Photo)

{

Console.WriteLine("\*картинка\*");

await botClient.SendTextMessageAsync(message.Chat, "Ахпхахп, эт чё? Ща как переверну)");

Console.WriteLine("Bot << Ахпхахп, эт чё? Ща как переверну)");

var file = await botClient.GetFileAsync(message.Photo.Last().FileId);

string imagePath = @".\photoToRotate.jpg";

using (var fileStream = new FileStream(imagePath, FileMode.Create))

await botClient.DownloadFileAsync(file.FilePath, fileStream);

var imageRotated = System.Drawing.Image.FromFile(imagePath);

imageRotated.RotateFlip(System.Drawing.RotateFlipType.Rotate180FlipNone);

imageRotated.Save(imagePath);

using (var fileStream = new FileStream(imagePath, FileMode.Open))

await botClient.SendPhotoAsync(message.Chat, InputFile.FromStream(fileStream));

Console.WriteLine("Bot << \*перевёрнутая картинка\*");

System.IO.File.Delete(imagePath);

return;

}

Console.WriteLine("Bot << Ну чё ты начинаешь, нормально же общались");

await botClient.SendTextMessageAsync(message.Chat, "Ну чё ты начинаешь, нормально же общались");

}

}

static async Task HandleErrorAsync(ITelegramBotClient botClient,

Exception exception,

CancellationToken cancellationToken)

{

Console.WriteLine(JsonConvert.SerializeObject(exception));

}

static string GetRandomPicture()

{

string path = @"C:\Users\User\Pictures\gray";

var pictures = Directory.GetFiles(path);

Random random = new Random();

int index = random.Next(pictures.Length);

return pictures[index];

}

public static void Start()

{

Console.WriteLine("\tStarted bot " + bot.GetMeAsync().Result.FirstName + "\n");

var cts = new CancellationTokenSource();

var cansellationToken = cts.Token;

var recieverOptions = new ReceiverOptions

{

AllowedUpdates = { }

};

bot.StartReceiving

(HandleUpdateAsync,

HandleErrorAsync,

recieverOptions,

cansellationToken);

}

}

}

**Program.cs**

using TgBotLogic;

TgBot.Start();

Console.ReadKey();