
БОТЫ
на .Net
информация и примеры

Боты
Разновидности

Информационные боты

Боты обрабатывающие строго формализованные команды

Боты для автоматизации рабочих процессов
(например, генерация задачи в TFS на основе письма об ошибке)

Поддерживающие естественный язык общения с клиентом/пользователем
(Нейронные сети и другие технологии)

Боты
Web-приложение VS бот

В обоих случаях необходим middle-слой обработки запросов и сервер для его хоста.

Web-приложение

Требуется затраты на разработку интерфейса, способного работать на разных браузерах и ОС. От разработчиков требуется высокая квалификация в части frontend-технологий.

Полный контроль внешнего вида приложения. Нестандартные кейсы

Полный контроль API

Бот

Достаточно придумать названия команд. Всю визуальную часть и поддержку разных ОС и браузеров берёт на себя мессенджер.

Нет контроля внешнего интерфейса бота.

API может поменяться

Бот для Telegram
Способы реализации сервиса

Главный источник информации по созданию ботов
https://core.telegram.org/bots

Бот состоит из двух частей: интерфейса в Telegram и сервиса, обрабатывающий запросы.

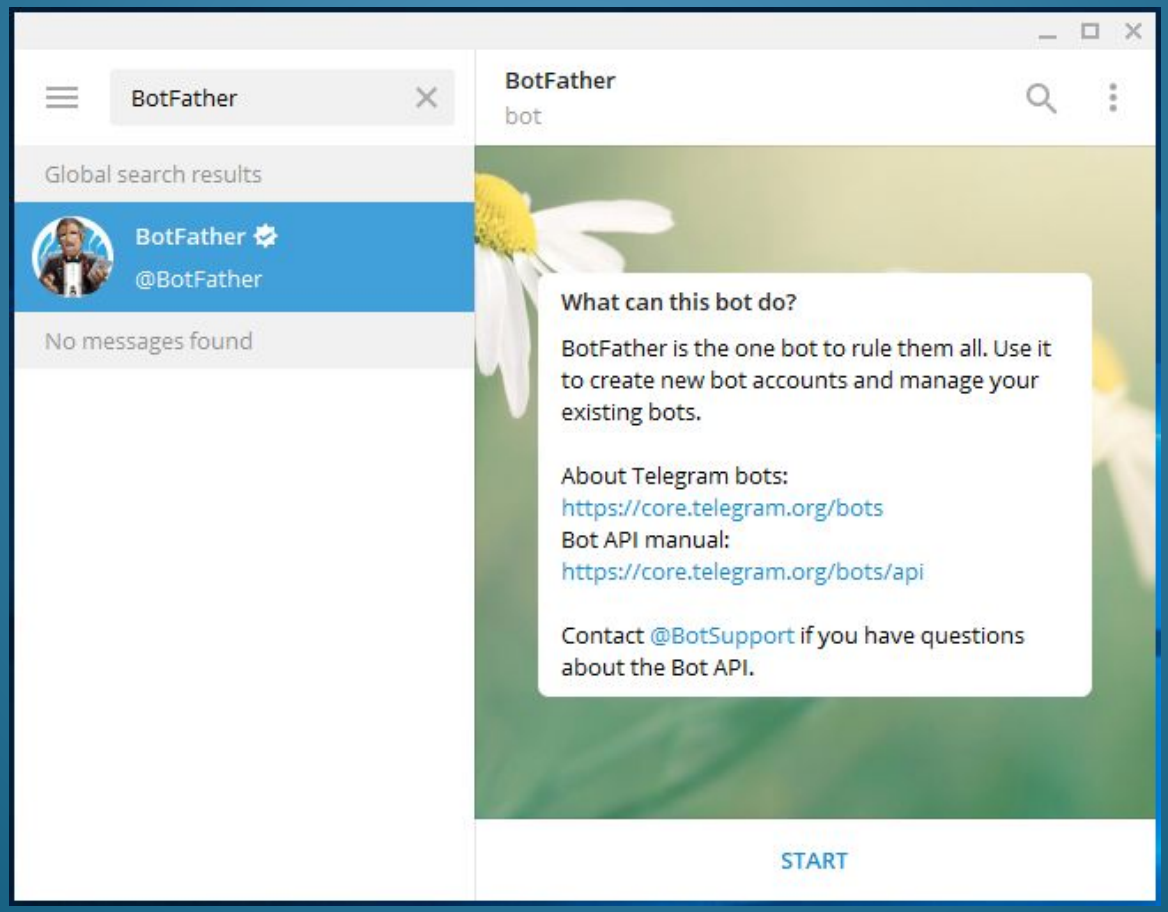
Интерфейс в Telegram создаётся с помощью специального бота BotFather.

Способы создания сервиса

1.
Самописное SDK
2.
Telegram.Bot готовое SDK на .Net (nuget-пакет Telegram.Bot)
https://github.com/TelegramBots/telegram.bot
3.
Bot Builder SDK for .NET (nuget-пакет Microsoft.Bot.Builder)

Бот для Telegram BotFather

\newbot запуск создания бота
\setcommands изменение списка доступных команд бота



Бот для Telegram
Способы реализации сервиса

getUpdates

При этом способе ваше приложение каждые 100мс (или реже) соединяется с сервером Telegram и опрашивает наличие изменений.

Минус подхода в том, что создается большая нагрузка на сервера Telegram.

Плюс в том, что он проще в реализации и тестировании, не нужно заморачиваться с SSL

Webhook

В этом случае Telegram отправляет все изменения на указанный сервис, запущенный на определённом порту

Минус в том, что для тестирования необходим SSL-сертификат

Плюс в том, что бот, не нагружает Telegram, обрабатывая только реально произошедшие изменения

Бот для Telegram
Код getUpdates сервиса

```
class Program
{
        private static TelegramBotClient client;

        static void Main(string[] args)
        {
                const string token = “324323425:fad323A";
                client = new TelegramBotClient(token);
                client.OnMessage += BotOnMessageReceived;
                client.OnMessageEdited += BotOnMessageReceived;


                client.StartReceiving();
                Console.ReadLine();
                client.StopReceiving();
        }
}
```

Бот для Telegram
Код getUpdates сервиса

```
private async void BotOnMessageReceived(object sender, MessageEventArgs
messageEventArgs)
{
        var message = messageEventArgs.Message;
        if (message?.Type == MessageType.TextMessage)
        {
                await client.SendTextMessageAsync(message.Chat.Id, message.Text);
        }
}
```

Бот для Telegram
Код webhook сервиса

```
public sealed class BotService
{
        private static readonly Lazy<BotService> instanceHolder =
        new Lazy<BotService>(() => new BotService());
        private readonly TelegramBotClient client;

        private BotService()
        {
                const string token = “32423425:fadf343fRA";
                client = new TelegramBotClient(token);
                client.SetWebhookAsync("Сервер_сервиса").Wait();
        }

        public static BotService Instance { get { return instanceHolder.Value; } }

        public void Disconnect()  { client.SetWebhookAsync().Wait(); } }
}
```

Бот для Telegram
Код webhook сервиса

```
[Route("bot")]
public class BotController : Controller
{
        // POST bot/update
        [HttpPost]
         public async void Post([FromBody]Update update)
        {
                if (update == null) return;
                var message = update.Message;
                if (message?.Type == MessageType.TextMessage)
                {
                        await BotService.Instance
                        .SendTextMessageAsync(message.Chat.Id, message.Text);
                }
        }
}
```

Инструменты проверки бота
Ngrok

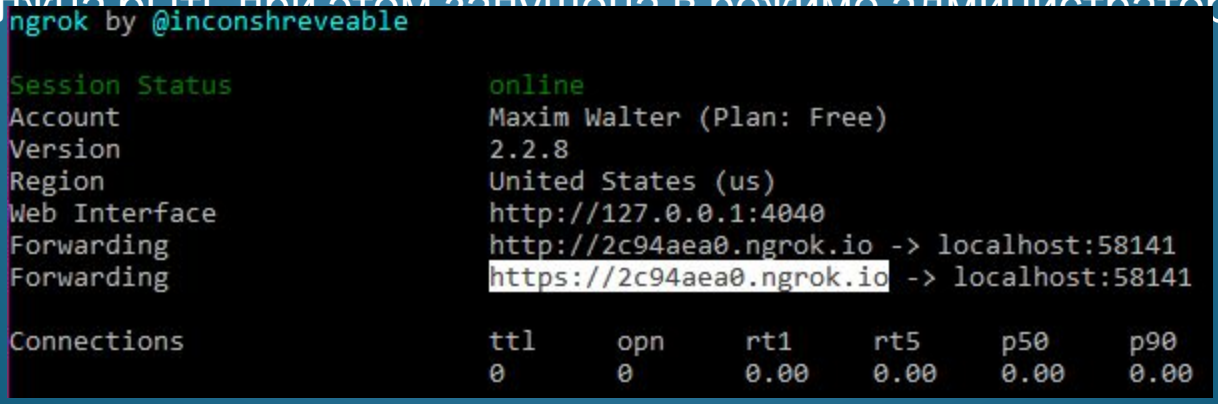
Ngrok (https://ngrok.com/) , генерируется https-адрес, который будет проксировать запросы на ваш сервис, запущенный на локальной машине.

Чтобы воспользоваться сервисом, нужно

- зарегистрироваться на сайте и получить персональный токен
- установить ngrok.exe и в командной строке ввести **ngrok authtoken ваштокен**
- после запуска сервиса, выполнить команду **ngrok http портсервиса**

Также если есть плагин к студии, то Ngrok можно запустить из меню Tools -> Start Ngrok tunnel

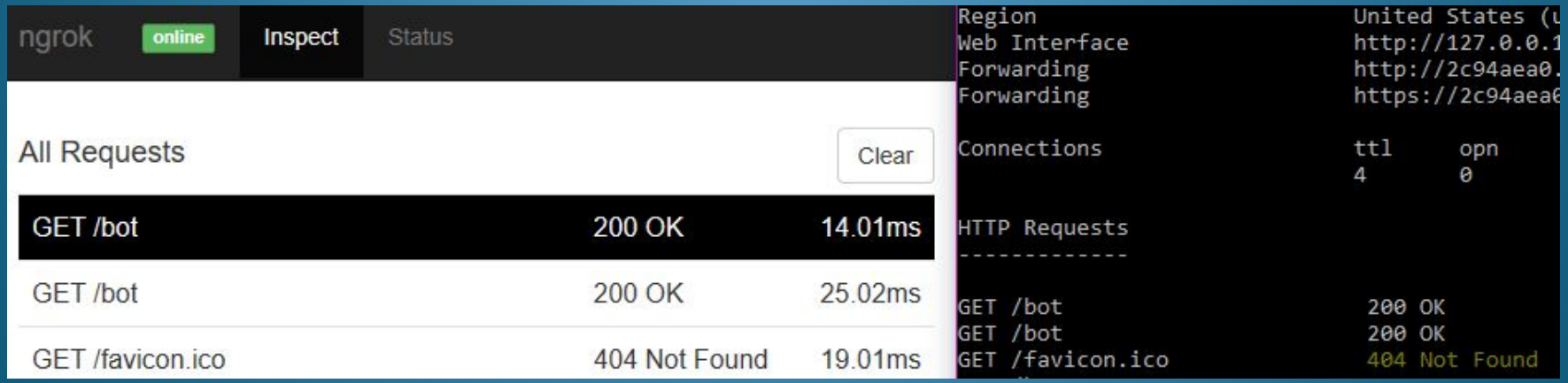
Студия должна быть при этом запущена в режиме администратора



Инструменты проверки бота Ngrok

Https-адрес, который вернёт Ngrok динамический, его каждый раз повторно необходимо указывать Telegram. Статический адрес – стоит денег (лучше оформить подписку в Azure)

У Ngrok есть админка, доступная по адресу http://127.0.0.1:4040



Bot Framework
Введение

Документация
https://docs.microsoft.com/en-us/bot-framework/overview-introduction-bot-framework
Регистрация ботов
https://dev.botframework.com/bots/new

Каналы

Bing
Cortana
Email
Facebook
GroupMe
Kik
Skype
Skype for Business
Slack
SMS
Microsoft Teams
Telegram

Недостатки

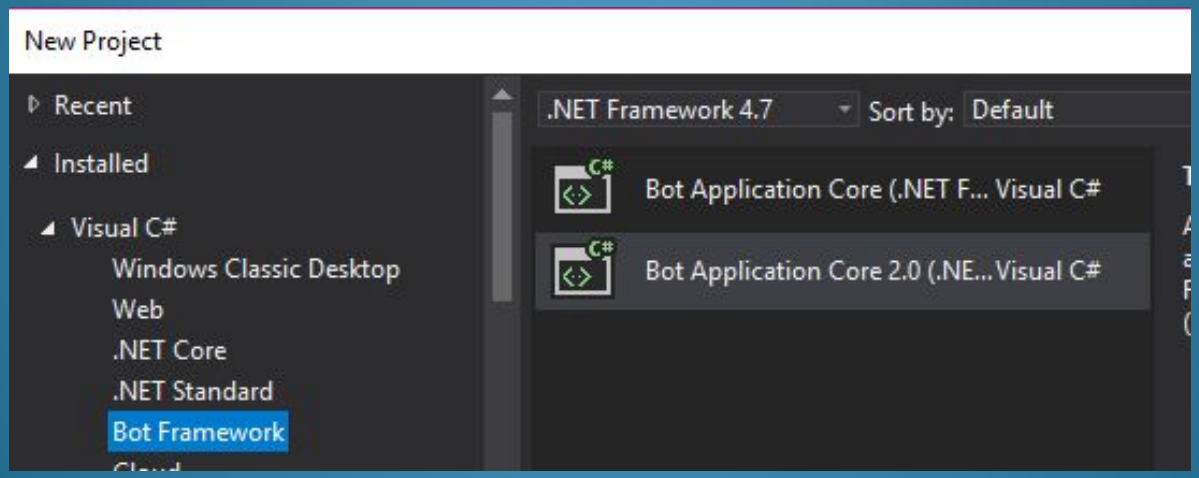
1. Нет поддержки .Net Core
https://github.com/Microsoft/BotBuilder/issues/572
https://designprincipia.com/microsoft-bot-framework-on-asp-net-core/

2. Сервис размещается на серверах Microsoft

Bot Framework SDK

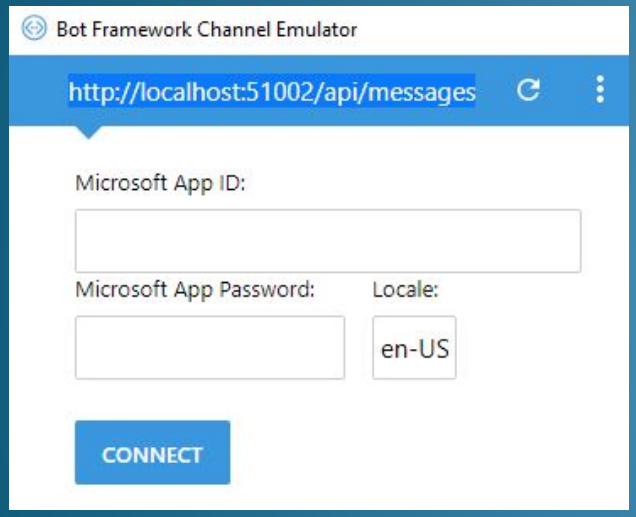
Bot Builder SDK доступен для C# и для Node.js

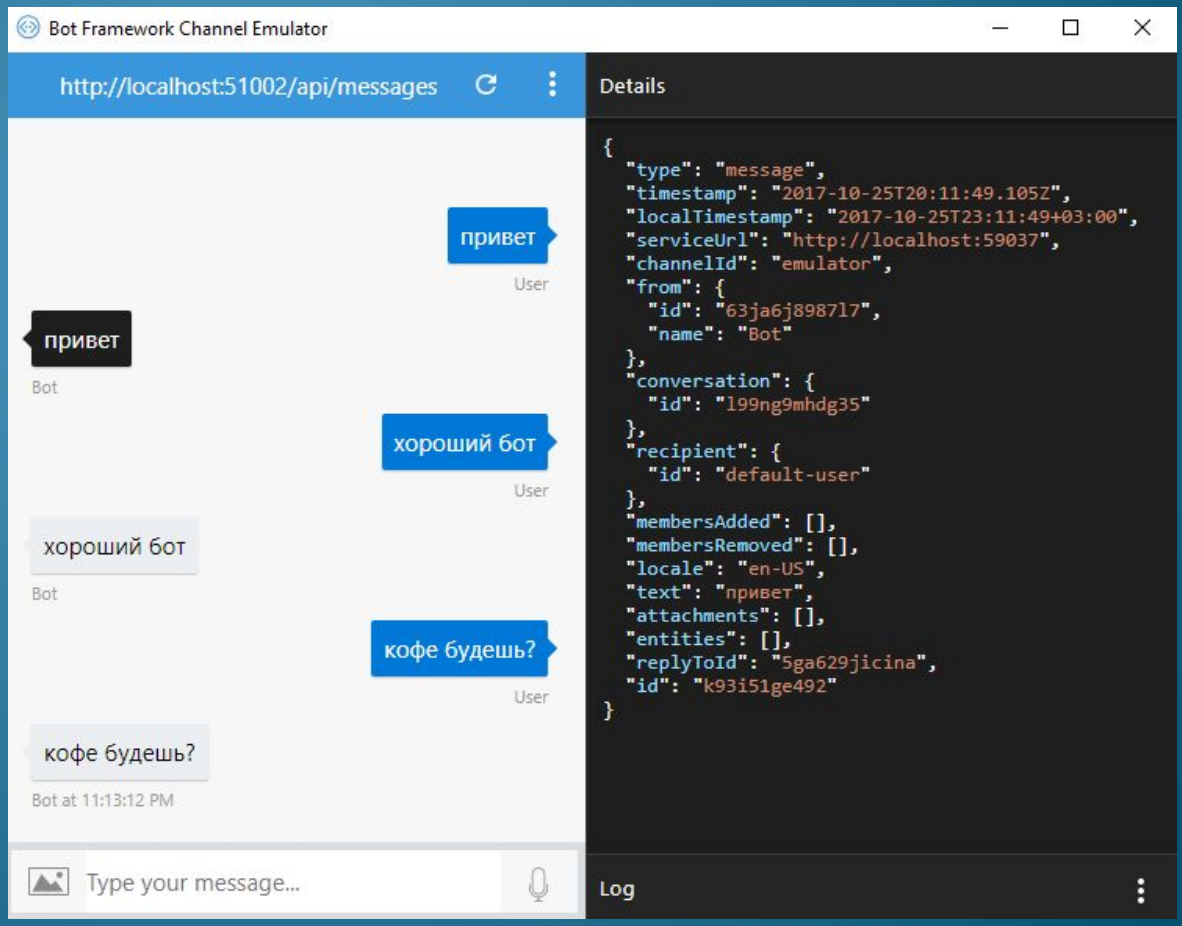
Для .NET ставится через nuget-пакет Microsoft.Bot.Builder



Bot Framework Emulator

https://github.com/Microsoft/BotFramework-Emulator





Bot Framework
Код контроллера и Activity

```
[Route("api/[controller]")]
[BotAuthentication]
public class MessagesController : Controller
{
    [HttpPost]
    public async Task<HttpResponseMessage> Post([FromBody]Activity activity)
    {
        if (activity?.Type == ActivityTypes.Message)
        {
            await Conversation.SendAsync(activity, () => new Dialogs.RootDialog());
        }

        return new HttpResponseMessage(HttpStatusCode.OK);
    }
}
```

Bot Framework
Обработка сообщений (Dialog)

```
[Serializable]
public class RootDialog : IDialog<object>
{
	public Task StartAsync(IDialogContext context)
	{
		context.Wait(MessageReceivedAsync);
		return Task.CompletedTask;
	}

	private async Task MessageReceivedAsync(IDialogContext context,
IAwaitable<object> result)
	{
		var activity = await result as Activity;
		await context.PostAsync(activity.Text);
		context.Wait(MessageReceivedAsync);
	}
}
```

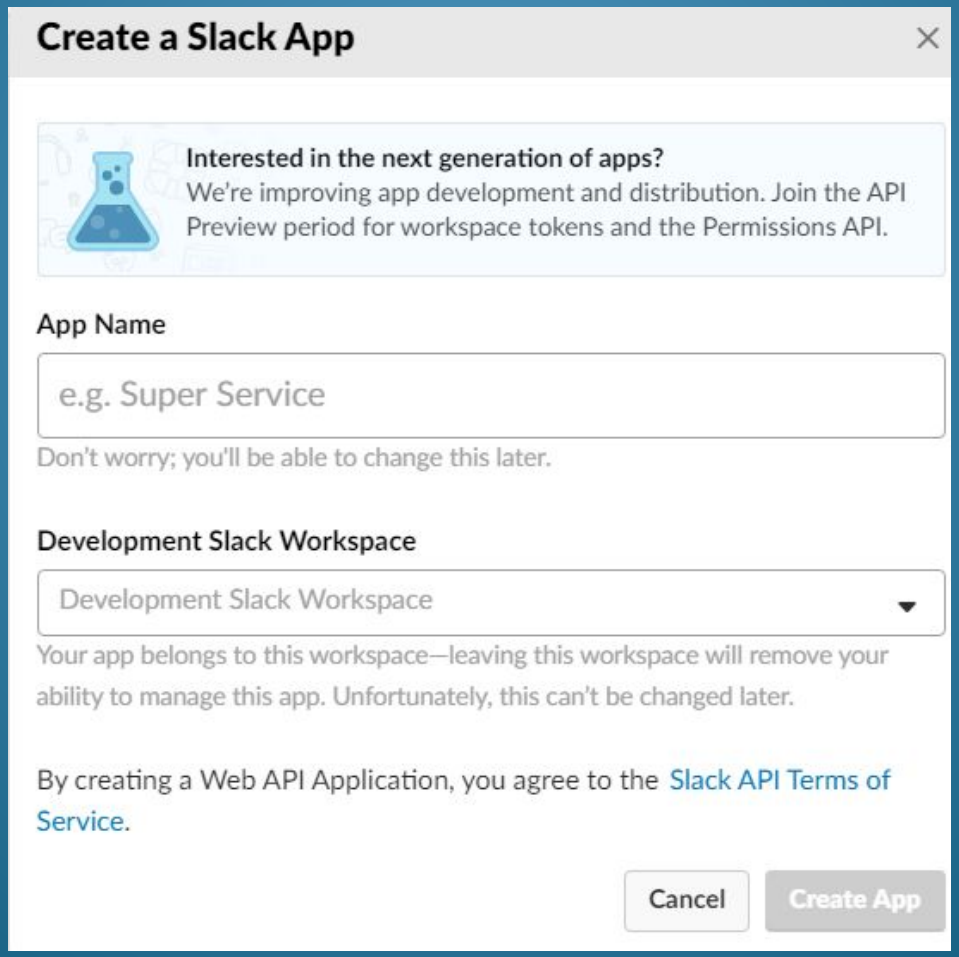
Bot Framework Connector

Connector – API обеспечивающее связь между разными каналами
https://docs.microsoft.com/en-us/bot-framework/dotnet/bot-builder-dotnet-connector

```
[Route("api/[controller]")]
[BotAuthentication]
public class MessagesController : Controller
{
    [HttpPost]
    public async Task<HttpResponseMessage> Post([FromBody]Activity activity)
    {
        var connector = new ConnectorClient(new Uri(activity.ServiceUrl));
        var reply = activity.CreateReply(activity.Text);
        await connector.Conversations.ReplyToActivityAsync(reply);

        return new HttpResponseMessage(HttpStatusCode.OK);
    }
}
```

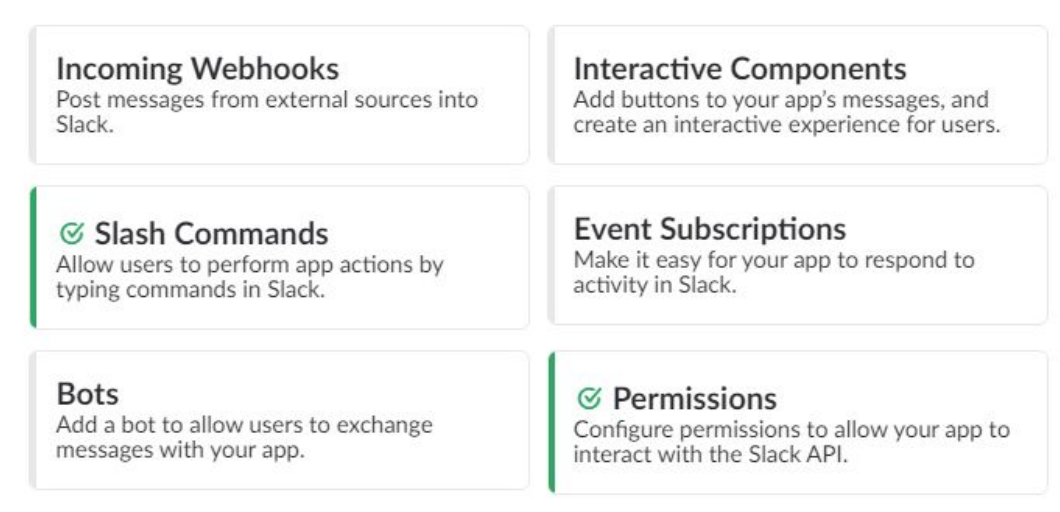
Бот для Slack

https://api.slack.com/apps
https://api.slack.com/apps/new



Бот для Slack
Настройка бота

Add features and functionality

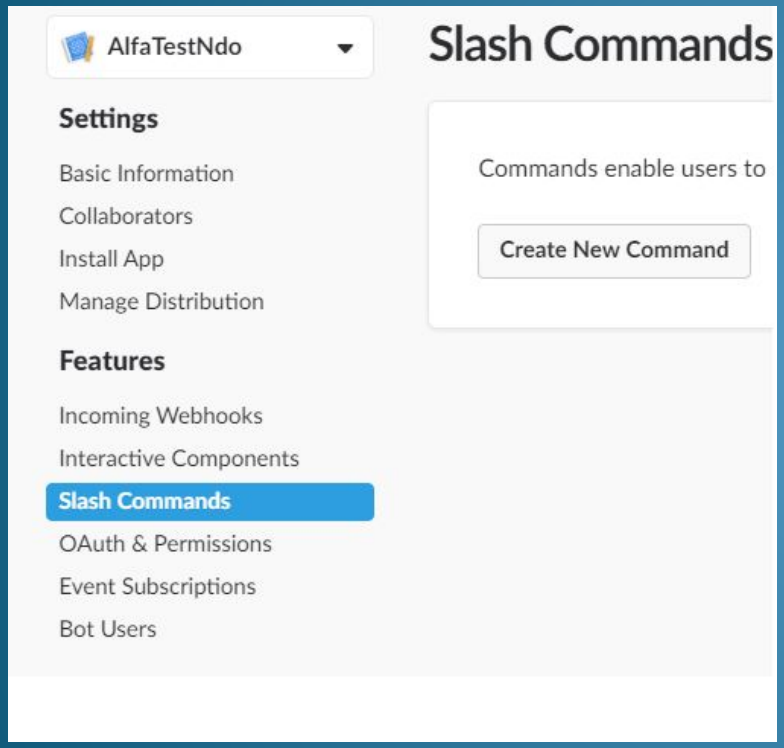


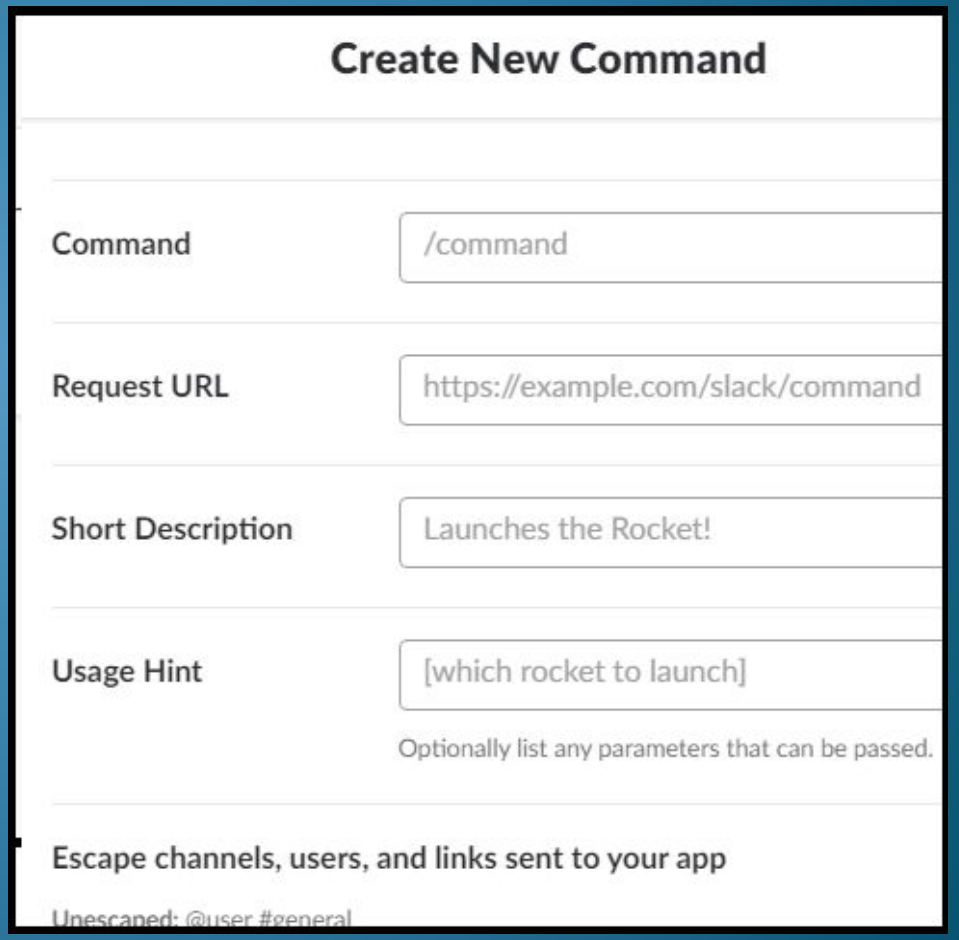
Install your app to your workspace
Установка приложения в ваше пространство.

Manage distribution
Позволяет открыть бота для других пространств.

App Credentials
Display Information

Бот для Slack
Настройка команд

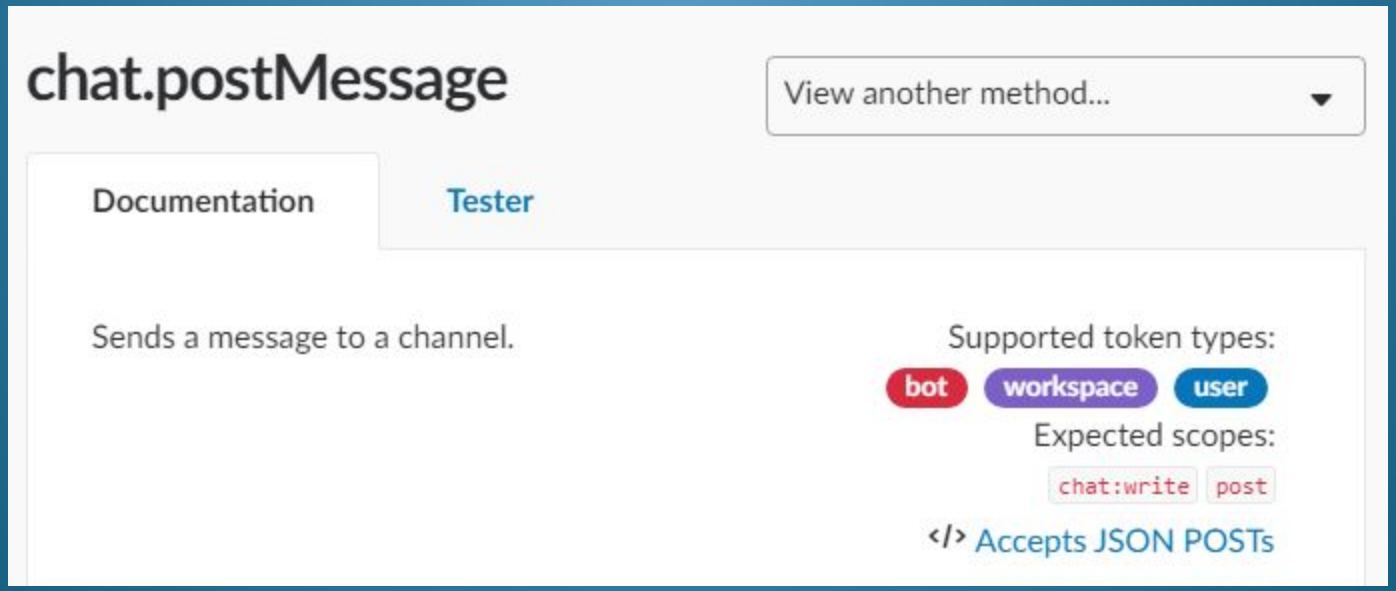




Бот для Slack
Типы ключей

User tokens - ключи пользователей, авторизованных через Oauth
Bot user tokens – ключ, специального созданного пользователя для бота
Workspace tokens – ключ рабочей област**и**
Legacy tokens – ни к чему не привязанный ключ (устаревший)
Verification token – ключ приложения

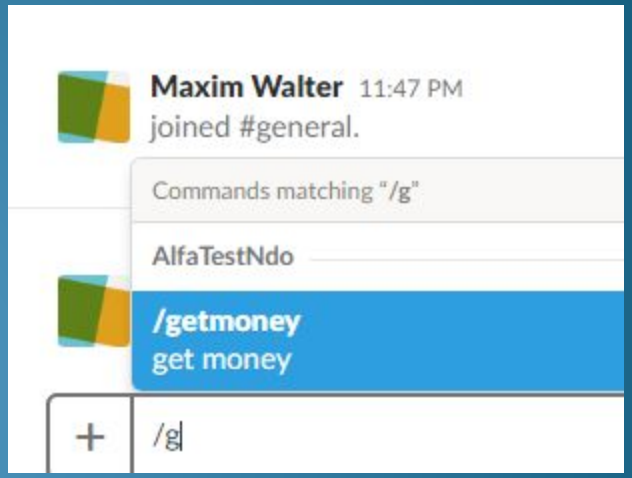
https://api.slack.com/methods/chat.postMessage

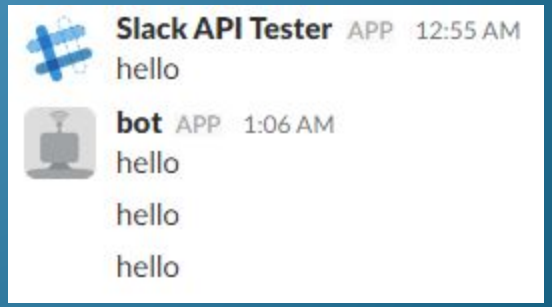


Slack API Tester – специальный бот для тестирования API

Бот для Slack
Запрос из Slack

```
public class Message
{
        public string channel_id { get; set; }
        public string channel_name { get; set; }
        public string command { get; set; }
        public string response_url { get; set; }
        public string team_domain { get; set; }
        public string team_id { get; set; }
        public string text { get; set; }
        public string token { get; set; }
        public string trigger_id { get; set; }
        public string user_id { get; set; }
        public string user_name { get; set; }
}
```





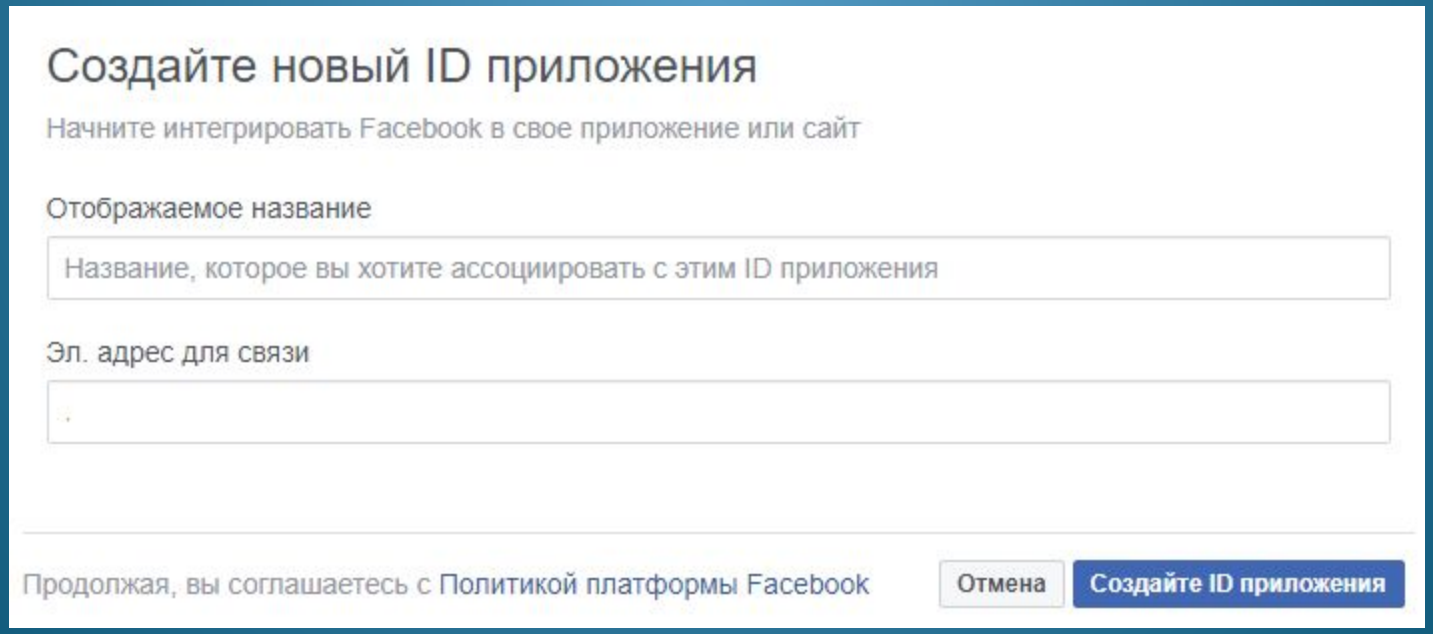
Бот для Slack
Контроллер

https://api.slack.com/community

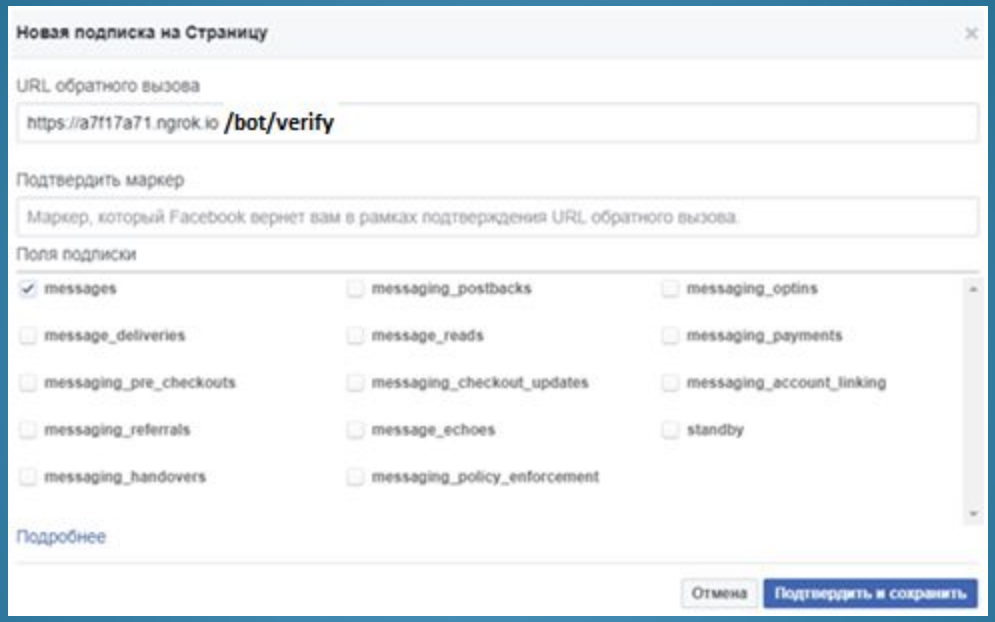
```
[Route("bot")]
public class BotController : Controller
{
	[HttpPost]
	public async void Post(Message message)
	{
	 var uri = new Uri("https://slack.com/api/chat.postMessage?token="
	 + token + "&channel=“ + message.channel_id + "&text=hello");

	 var httpClient = new HttpClient();
	 await httpClient.GetAsync(uri).ConfigureAwait(false);
	}
}
```

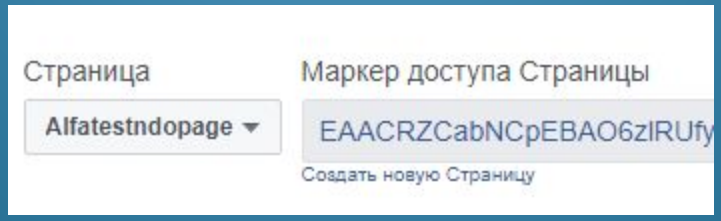
Бот для Facebook

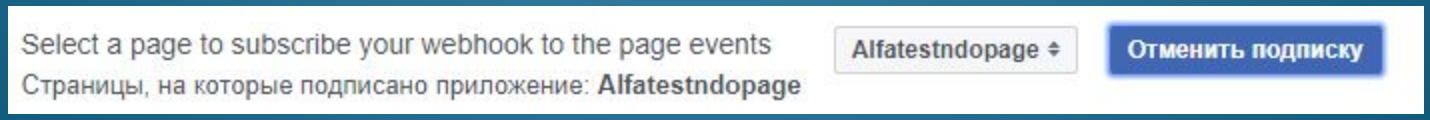
Официальная инструкция
https://developers.facebook.com/docs/messenger-platform/getting-started/quick-start
Страница создания приложений
https://developers.facebook.com/apps



Бот для Facebook
Настройка Webhook







Бот для Facebook
Настройка Webhook

```
[Route("bot")]
public class BotController : Controller
{
    [HttpGet]
    public string Verify()
    {
        var mode = Request.Query["hub.mode"].FirstOrDefault();
        var challenge = Request.Query["hub.challenge"].FirstOrDefault();
        var token = Request.Query["hub.verify_token"].FirstOrDefault();
        return challenge ?? string.Empty;
    }

    [HttpPost]
    public void Post([FromBody]string value)
    {
    }
}
```

Бот для Facebook
Входящее сообщение


```
{
    "object": "page",
    "entry": [
        {
            "id": "507370816302631",
            "time": 1509659789298,
            "messaging": [
                {
                    "sender": {
                        "id": "1187895957978601"
                    },
                    "recipient": {
                        "id": "507370816302631"
                    },
                    "timestamp": 1509659769025,
                    "message": {
                        "mid": "mid.$cAAGXgDziYCdlsAFAwVffrwNYg_C1",
                        "seq": 269,
                        "text": "hello"
                    }
                }
            ]
        }
    ]
}
```

Бот для Facebook
Отправка ответа

```
[HttpPost]
public void Post([FromBody] Letter letter)
{
        var content = letter.entry[0].messaging[0];
        const string token = “yourtoken";
        var uri = new Uri("https://graph.facebook.com/v2.6/me/messages?access_token="
+ token);

        var request = (HttpWebRequest)WebRequest.Create(uri);
        request.ContentType = "application/json";
        request.Method = "POST";
        using (var requestWriter = new StreamWriter(request.GetRequestStream()))
        {
            requestWriter.Write($@" {{recipient: {{  id: {content.sender.id}}},message: {{text:
""{content.message.text}"" }}}}");
        }
        var response = (HttpWebResponse)request.GetResponse();
}
```

Ссылки

Примеры использования API telegram.bot
https://github.com/TelegramBots/telegram.bot.examples
Пишем бота Telegram на C#
http://aftamat4ik.ru/pishem-bota-telegram-na-c/
Как легко написать бота для Telegram на C#
https://habrahabr.ru/sandbox/103396/

Microsoft Bot Framework (статья введение)
http://ru.bmstu.wiki/Microsoft_Bot_Framework
Цикл видео по Microsoft Bot Framework
https://www.youtube.com/playlist?list=PLgF-CyaX1p3FE55OTRNH-kOb16zqeBZCo
Microsoft Bot Framework на Linux под Node.JS
https://habrahabr.ru/post/333824/

Azure Bot Service (описание и цены на хостинг ботов)
https://azure.microsoft.com/ru-ru/services/bot-service/

Ссылки

Разработка чат-бота для Facebook Messenger
https://habrahabr.ru/post/281559/
Facebook Chatbot in ASP.NET
https://tutorials.botsfloor.com/facebook-chatbot-in-asp-net-2f9379a238b0

Создание бота в контакте (для сообществ)
https://vk.com/dev/bots
https://vk.com/dev/bots_docs

Сводная страница информации по ботам в Viber
https://habrahabr.ru/post/338970/