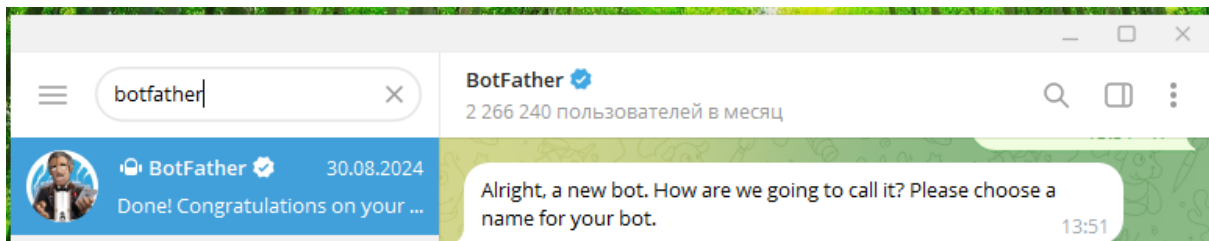


#1

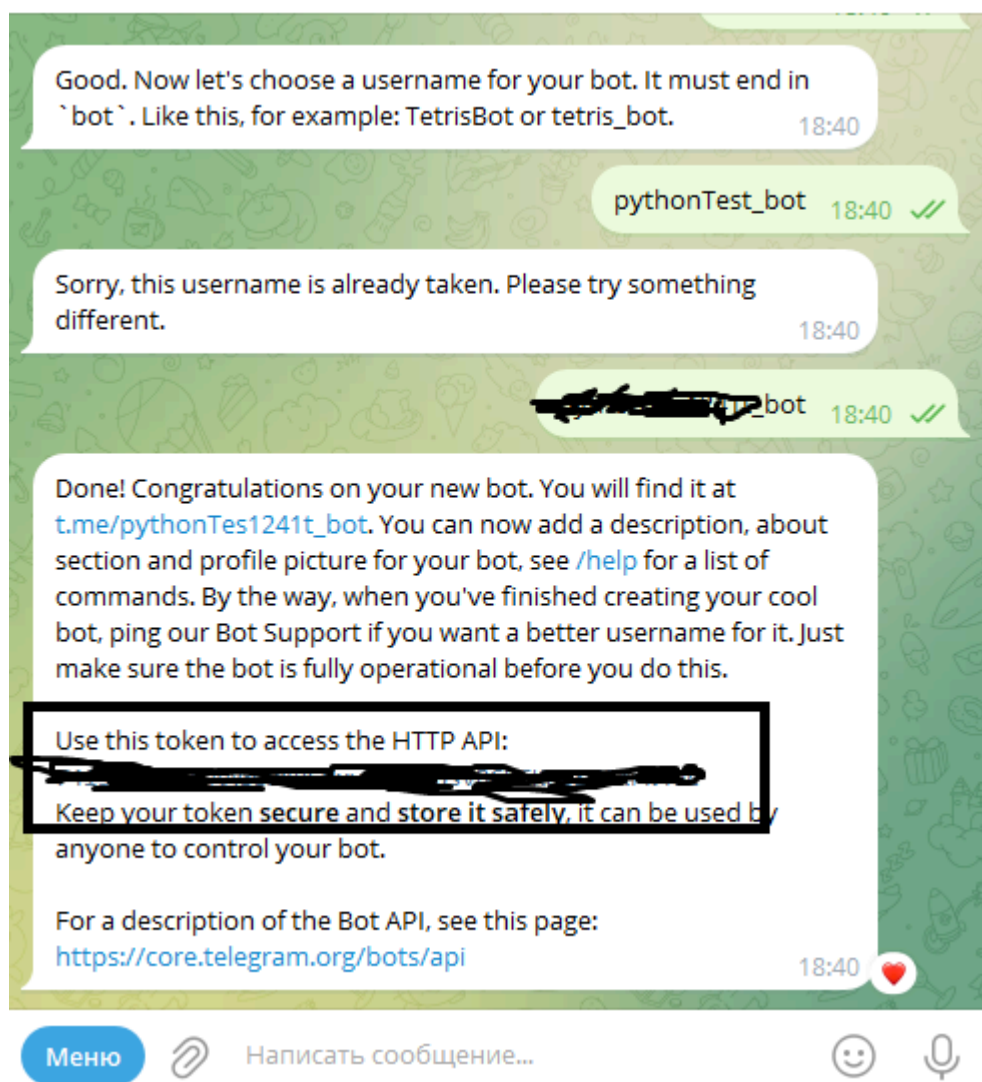


## Создание бота и получение токена

Для получения токена пишем в поиске botfather и нажимаем старт



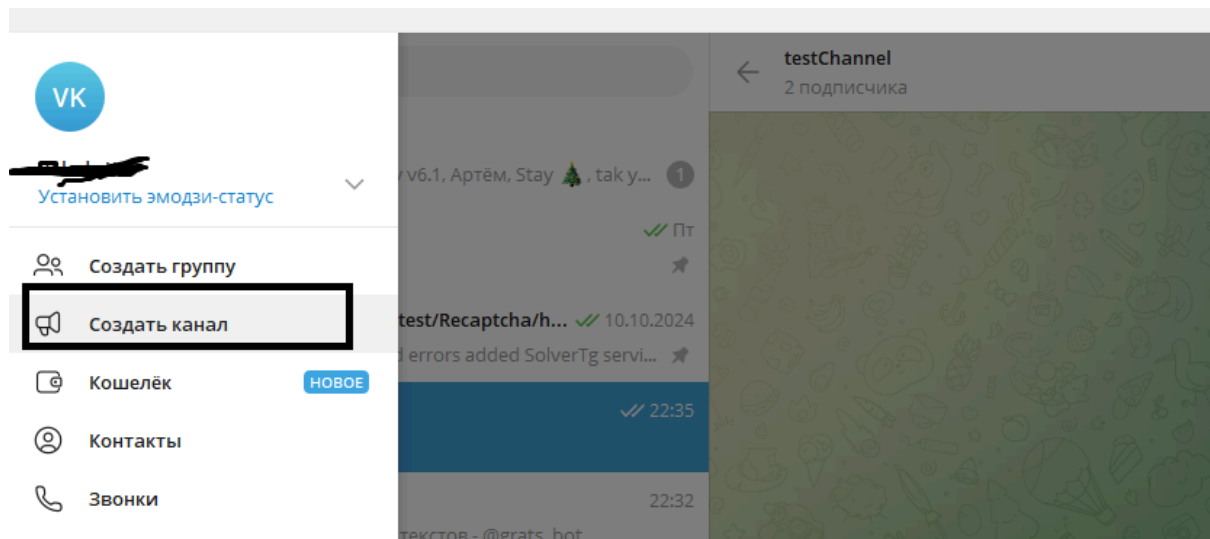
- ☐ выбираем /newbot (т.е. создание нового бота)
- ☐ выбираем имя
- ☐ выбираем никнейм бота (на конце \_bot , например, test\_bot).  
Если имя занято, он попросит выбрать другое имя (Sorry, this username is already taken. Please try something different.)
- ☐ сохраняем себе токен и никнейм бота



## Создание канала

Теперь создадим публичный канал (потом можно сделать его приватным при необходимости) и добавим туда нашего бота.

Жмём “создать канал”



Вводим имя (любое).

The image shows the 'Создать канал' (Create channel) form in the VK mobile application. The form has a title 'Название канала' (Channel name) with the text 'test' entered. Below it is a field for 'Описание (необязательно)' (Description (optional)). At the bottom are two buttons: 'Отмена' (Cancel) and 'Создать' (Create).

Выбираем урл (запоминаем).

☒ **Публичный канал**  
 Все могут найти канал через поиск и подписаться

☐ **Частный канал**  
 Подписаться можно только по ссылке-приглашению.

Ссылка Такая ссылка доступна  
t.me/fghfgjfgj46745

Пропустить      Сохранить

Вписываем имя нашего бота и добавляем как админа

Добавить участников

🔍 python | ✕

python  
@pythonTes1241t\_bot

## Получаем чат айди

Во многих методах Bot API нам понадобится такая вещь, как `chat_id`. Это уникальный идентификатор чата, куда бот будет отправлять сообщения. Я буду показывать на примере группы, но в принципе сообщения можно слать и на свой аккаунт.

Для получения чата айди группы мы шлём такой запрос (кубик Get):

```
https://api.telegram.org/bot{-Variable.tg_token-}/getChat?chat_id=@{-Variable.tg_nameChanell-}
```

Разберем этот запрос подробнее:

- ☐ <https://api.telegram.org/bot> - любой наш запрос идет на этот урл
- ☐ {-Variable.tg\_token-} - в переменную tg\_token помещаем токен, который нам прислал BotFather
- ☐ getChat - название метода. В данном случае получение chat\_id. Далее ставим знак ? и перечисляем параметры запроса
- ☐ chat\_id - название параметра
- ☐ =@{-Variable.tg\_nameChanell-} - значение параметра. В переменную tg\_nameChanell мы помещаем имя канала. Например, ссылка на ваш канал [t.me/superchannel234235235](https://t.me/superchannel234235235). Тогда в переменную tg\_nameChanell нужно положить [superchannel234235235](https://t.me/superchannel234235235)

В ответ мы получим JSON. Пример ответа:

```
{
  "ok": true,
  "result": {
    "id": -1002305474745,
    "title": "testChannel",
    "username": "testchannelvolody00",
    "type": "channel",
    "active_usernames": [
      "testchannelvolody00"
    ],
    "invite_link": "https://t.me/+Sm21Gq6w38E4YTBi",
    "has_visible_history": true,
    "can_send_paid_media": true,
    "available_reactions": [],
    "max_reaction_count": 11,
    "accent_color_id": 4
  }
}
```

Разберем ответ:

- ☐ "ok": true - true мы получаем, если запрос прошел успешно.  
false (например, неверный токен) - если неудачно.
- ☐ id - тот самый чат айди, ради которого мы делали запрос
- ☐ Остальное разбирать не буду, если интересно читайте и разбирайтесь - <https://core.telegram.org/bots/api#chatfullinfo>

## Отправка сообщения в группу

Пока углубляться во все параметры не будем (может в следующий раз, посмотрим), рассмотрим простейший запрос

`https://api.telegram.org/bot{-Variable.tg_token-}/sendMessage?chat_id={-Variable.tg_chat_id-}&text={-Variable.tg_text-}`

Разбираем запрос:

- ☐ [https://api.telegram.org/bot{-Variable.tg\\_token-}](https://api.telegram.org/bot{-Variable.tg_token-}) - тут понятно
- ☐ sendMessage - название метода для отправки текстовых сообщений
- ☐ chat\_id - чат айди, куда отправляем (бот должен быть админом группы, куда он отправляет)
- ☐ text - отправляемый текст. Максимум 4096 символов

Пример ответа:

```
{
  "ok": true,
  "result": {
    "message_id": 2,
    "sender_chat": {
      "id": -1002306908076,
      "title": "testChannel",
      "username": "testchannelvolody00",
      "type": "channel"
    },
    "chat": {
```

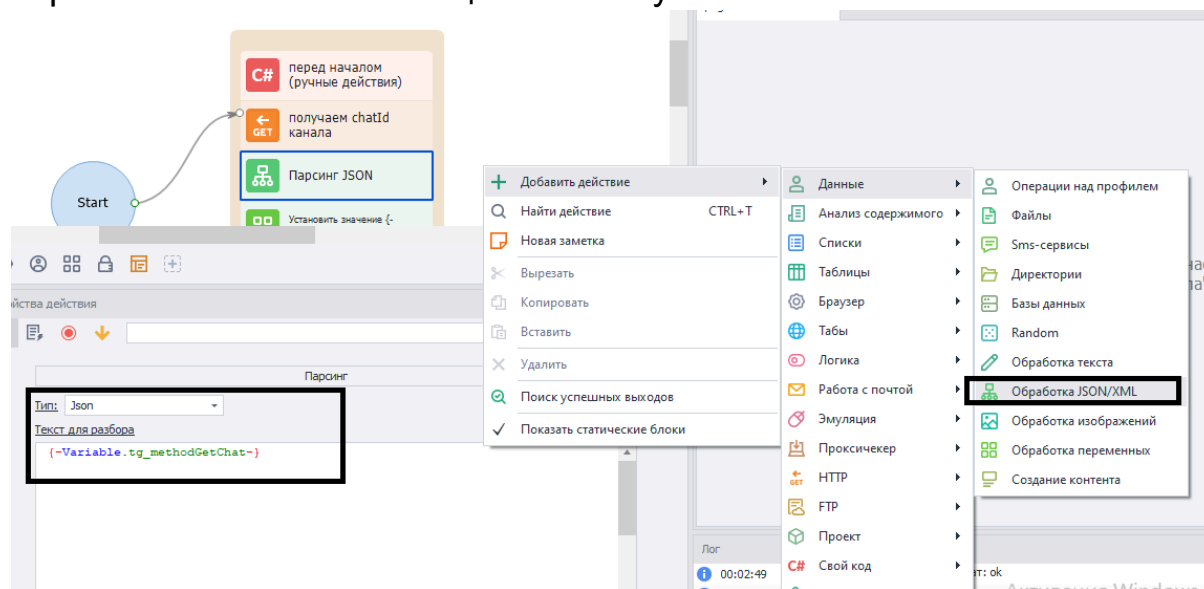
```
{
  "id": -1002306908076,
  "title": "testChannel",
  "username": "testchannelvolody00",
  "type": "channel"
},
{
  "date": 1735851466,
  "text": "123"
}
}
```

Тут отмечу следующее:

- ☐ message\_id, судя по всему, это порядковый номер сообщения. Чем больше сообщений в группе, тем больше будет становиться message\_id
- ☐ date - дата у нас в unix time, для перевода в обычную пользоваться конвертерами

## Как обрабатывать ответ

Вы наверняка это знаете, но на всякий случай напишу. Для обработки JSON есть специальный кубик



После запуска кубика в переменных можно переключиться на вкладку "json" и посмотреть, что там напарсилось

Переменные		
<div> <div>Свой</div> <div>Авто</div> <div>Окружение</div> <div>Глобальные</div> <div><b>Json</b></div> <div>Xml</div> <div>?</div> <div>+</div> <div>-</div> <div>x</div> </div>		
Имя	Значение	
→ ok	True	
result.id	-1002306908076	
result.title	testChannel	
result.username	testchannelvolody00	
result.type	channel	
result.active_usernames.Count	1	
result.active_usernames[0]	testchannelvolody00	
result.invite_link	https://t.me/+Sm21Gq6w38E4YTBi	
result.has_visible_history	True	
result.can_send_paid_media	True	
result.available_reactions.Count	0	
result.max_reaction_count	11	
result.accent_color_id	4	

Пример, как можно обратиться к переменной

{-Json.result.id-}

## Реализация на C#

Ну а вот так это всё выглядит на c#

```
//заполняем переменные
project.Variables["tg_token"].Value =
"7413525589:AAGCkYuGWxvtzMK58svBCKGKB8SYCXrk4ec";
project.Variables["tg_nameChanell"].Value = "testchannelvolody00";

//получаем chat id
string urlChatId =
$"https://api.telegram.org/bot{project.Variables["tg_token"].Value}/getCha
t?chat_id=@{project.Variables["tg_nameChanell"].Value}";
string responseChatId =
ZennoPoster.HTTP.Request(ZennoLab.InterfacesLibrary.Enums.Http.HttpMethod.GET, urlChatId);
```



```

project.Json.FromString(responseChatId);

if(project.Json.ok.GetType() == typeof(bool))
    project.SendInfoToLog("в переменной ok лежит bool");

if(project.Json.ok == true)
{
    project.SendInfoToLog("запрос на получение айди выполнен успешно");
    project.Variables["tg_chatId"].Value =
project.Json.result.id.ToString();
}
else throw new Exception("неудачный запрос на получение айди - " +
responseChatId);

//отправляем сообщение
string text = "Тестовый текст";
string urlSendMessage =
$"https://api.telegram.org/bot{project.Variables["tg_token"].Value}/sendMessage?chat_id={project.Variables["tg_chatId"].Value}&text={text}";
string responseSendMessage =
ZennoPoster.HTTP.Request(ZennoLab.InterfacesLibrary.Enums.Http.HttpMethod.GET, urlSendMessage);

project.Json.FromString(responseSendMessage);

if(project.Json.ok) project.SendInfoToLog("запрос на отправку сообщения выполнен успешно");
else throw new Exception("неудачный запрос на отправку сообщения - " + responseChatId);

```

Пару слов про project.Json. Это свойство возвращает тип dynamic, из-за этого у вас не будут всплывать подсказки при написании кода.

Метод project.Json.FromString() работает как кубик “обработка json”. В качестве параметра принимает Json и возвращает нам результат

в виде вот этих переменных.

Имя	Значение
ok	True
result.id	-1002306908076
result.title	testChannel
result.username	testchannelvolody00
result.type	channel
result.active_usernames.Count	1
result.active_usernames[0]	testchannelvolody00
result.invite_link	https://t.me/+Sm21Gq6w38E4YTBi
result.has_visible_history	True
result.can_send_paid_media	True
result.available_reactions.Count	0
result.max_reaction_count	11
result.accent_color_id	4

Далее к этим переменным можно обращаться

```
project.Json.имя_переменной.
```

Только следите за типом данных. Например, на скрине выше “ok” возвращает bool, именно поэтому мы проверку в if делаем так:

```
if (project.Json.ok == true)
```

Если сходу непонятно, какой там тип данных, можно его посмотреть таким образом:

```
return project.Json.имя_переменной.GetType()
```

а проверить в коде так:

```
if(project.Json.result.username.GetType() == typeof(string))  
    project.SendInfoToLog("oki");
```