Подключение распознавания файлов в различных каналах

Редактирование настроек канала VK

В VK доступны реакции на фото, документы, видео, аудио и стикеры.

При создании канала необходимо для каждого типа файлов выбрать свои настройки:

1. Выбрать реакцию на файл:

- Игнорировать - бот не будет реагировать на файлы данного типа;
- Только NLP - бот будет реагировать на файлы данного типа ответом из NLP системы;
- Только меню - бот будет реагировать на файлы данного типа с помощью меню бота;
- Штатная (NLP и меню) - бот будет реагировать обоими способами.

2. Добавить контекст NLP при необходимости.

3. Если для файла не должен срабатывать Fallback скрипт, то проставить метку **Отключить Fallback**.

Редактирование настроек канала MAX

В MAX доступны реакции на фото, документы, видео, аудио и стикеры.

При создании канала необходимо для каждого типа файлов выбрать свои настройки:

1. Выбрать реакцию на файл:

- Игнорировать - бот не будет реагировать на файлы данного типа;
- Только NLP - бот будет реагировать на файлы данного типа ответом из NLP системы;
- Только меню - бот будет реагировать на файлы данного типа с помощью меню бота;
- Штатная (NLP и меню) - бот будет реагировать обоими способами.

2. Добавить контекст NLP при необходимости.

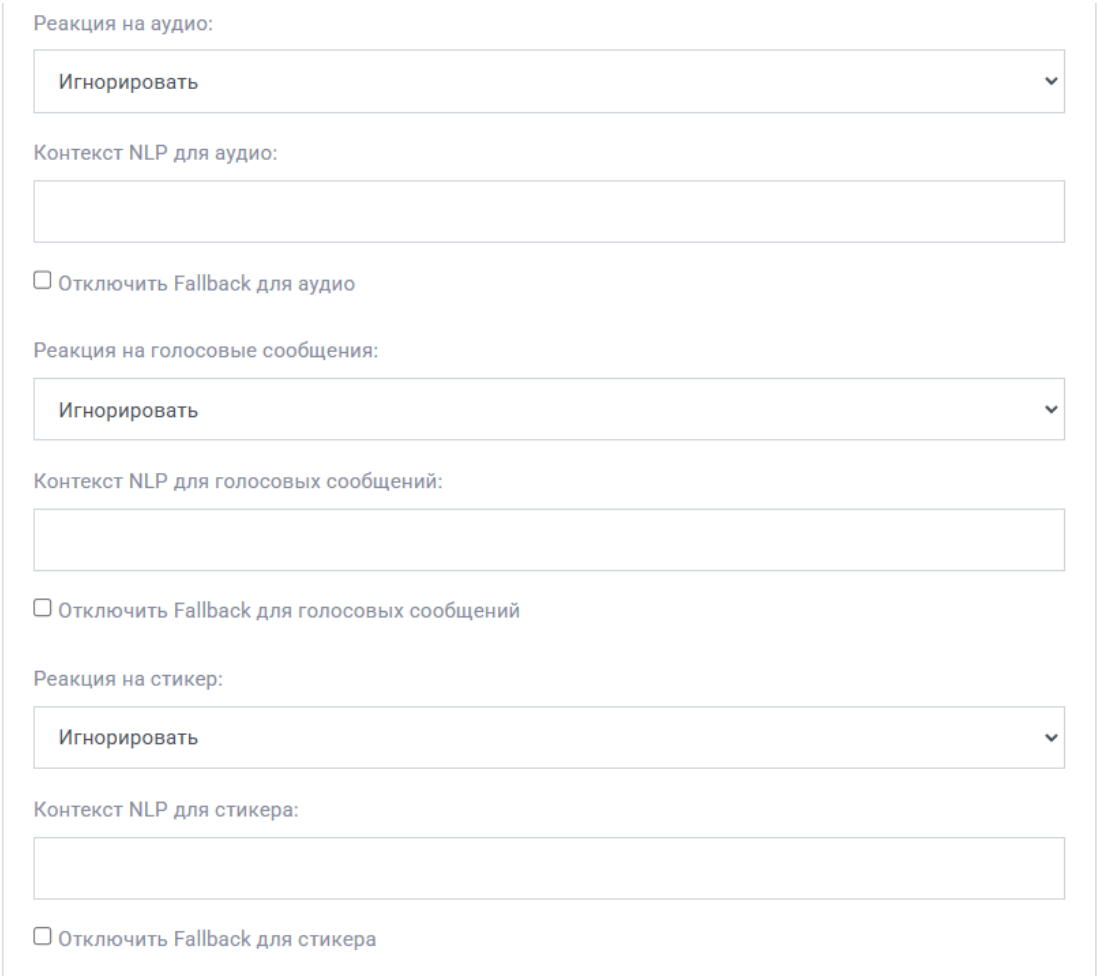
3. Если для файла не должен срабатывать Fallback скрипт, то проставить метку **Отключить Fallback**.

Редактирование настроек канала Telegram

В Telegram доступны реакции на фото, документы, видео, аудио, голосовые сообщения и стикеры.

При создании канала необходимо для каждого типа файлов выбрать свои настройки:





1. Выбрать реакцию на файл:

- Игнорировать - бот не будет реагировать на файлы данного типа;
- Только NLP - бот будет реагировать на файлы данного типа ответом из NLP системы;
- Только меню - бот будет реагировать на файлы данного типа с помощью меню бота;
- Штатная (NLP и меню) - бот будет реагировать обоими способами.

2. Добавить контекст NLP при необходимости.

3. Если для файла не должен срабатывать Fallback скрипт, то проставить метку **Отключить Fallback**.

Редактирование настроек канала Metabot Widget

В канале **Metabot Widget** через редактирование необходимо:

1. Поставить активность чекбокса **Разрешить отправку файлов**.

2. В поле **Список поддерживаемых расширений файлов** добавить список разрешенных форматов файлов:

```
pdf,doc,docx,xls,xlsx,csv,txt,png,jpe,jpg,jpeg,gif,bmp,ico,svg,svgz,tif,tiff,jfif,ai,drw,pct,psp,xcf,psd,raw,webp,xbm,dib,pgp,
apng,pjpeg,avif,avi,divx,flv,m4v,mkv,mov,mp4,mpeg,mpg,ogm,ogv,ogx,rm,rmvb,smil,webm,wmv,xvid,3gp,3g2,qt,asx
```

3. Выбрать реакцию на файл:

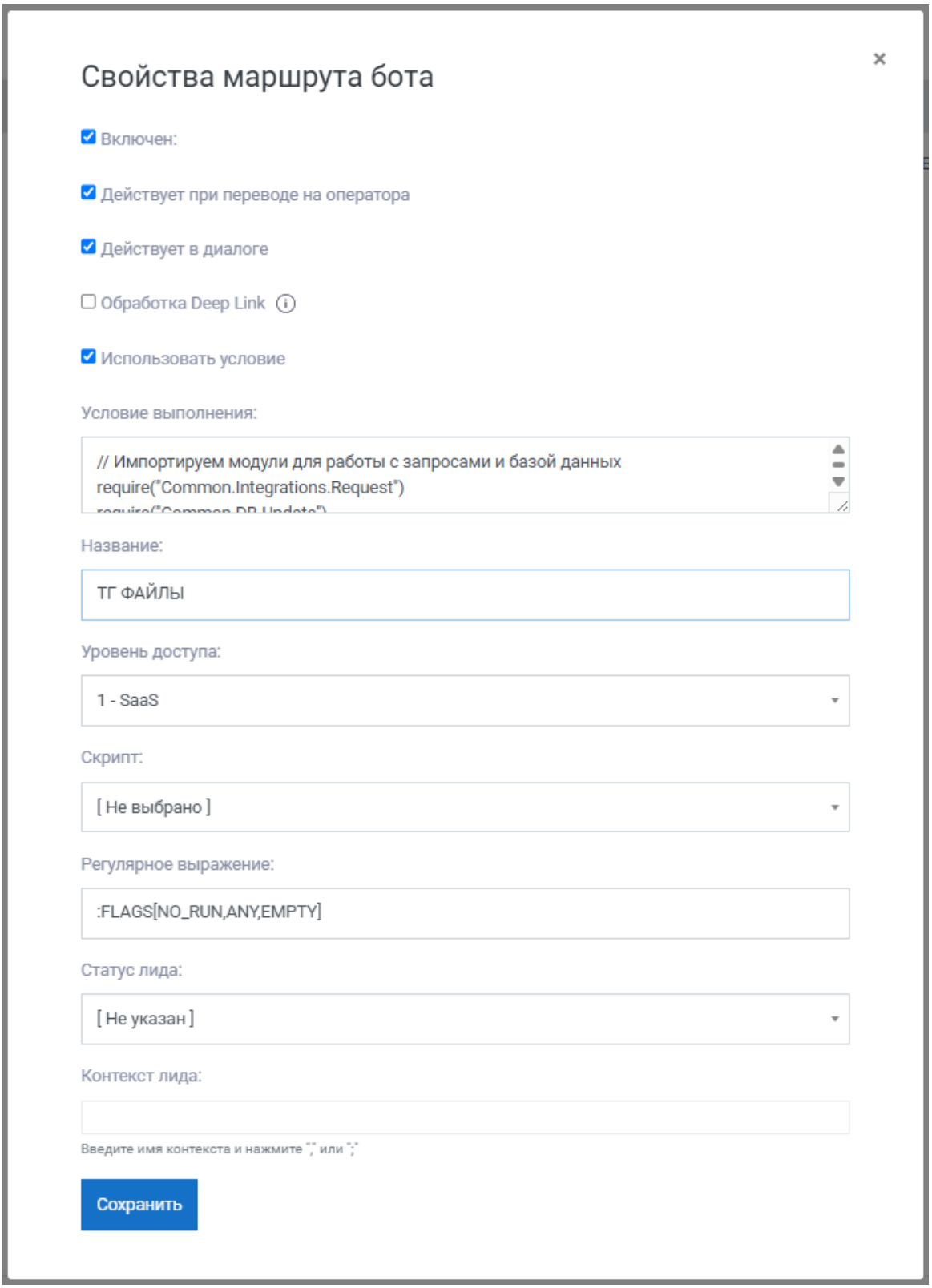
- Игнорировать - бот не будет реагировать на файлы данного типа;
- Только NLP - бот будет реагировать на файлы данного типа ответом из NLP системы;
- Только меню - бот будет реагировать на файлы данного типа с помощью меню бота;
- Штатная (NLP и меню) - бот будет реагировать обоими способами.

4. Сохранить настройки.

bq780Nt2tykBm36gFthumdrunUxWhBTNWSzYk3DaXkAa094c193ZB2AW8ApXd

Создание маршрута

Для приема файлов необходимо создать маршрут со следующими настройками:



- Активен параметр **Действует при переводе на оператора**;
- Активен параметр **Действует в диалоге**;
- Регулярное выражение - **:FLAGS[NO_RUN,ANY,EMPTY]**;
- Условие выполнения:

```
// Импортируем модули для работы с запросами и базой данных
require("Common.Integrations.Request")
```

```
require("Common.DB.Update")

let httpRequest = CommonIntegrationsRequest
let database = CommonDBUpdate

// Входные данные - массив с информацией о файле
let attachments = bot.getAllAttachments()

// Проверяем, есть ли вложения,
// если нет - прекращаем выполнение
if (!attachments.length) {
    return false
}

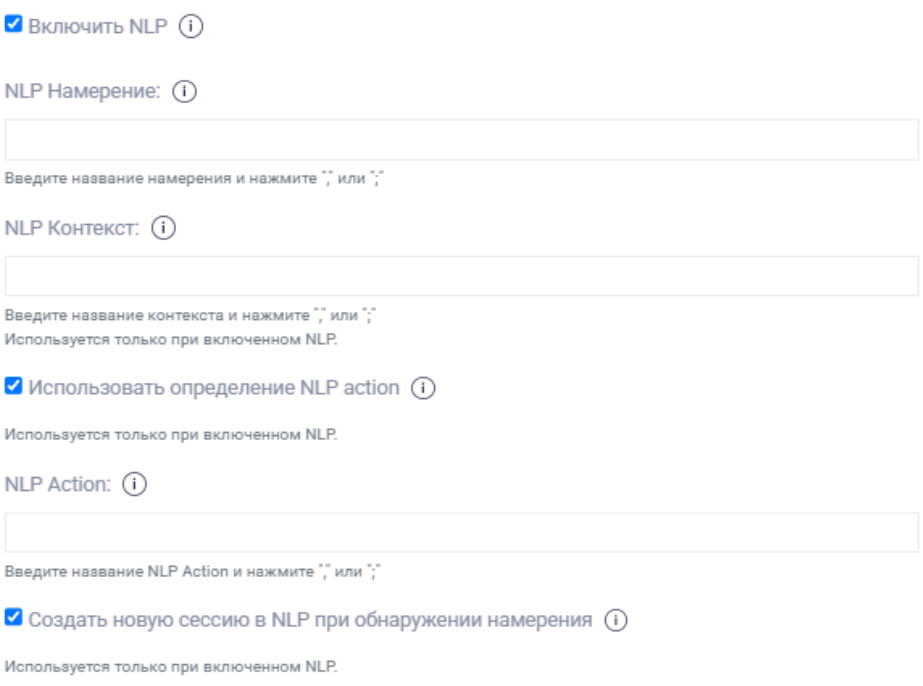
// URL API для загрузки файла
const channelData = lead.getChannelData();
const platformToken = channelData.webhook_url;
const regexBaseUrl = /^https?:\/\/[^\/]+/;
const platformUrl = channelData
  .full_webhook_url
  .match(regexBaseUrl)?.[0] ?? null;
if (!platformUrl) {
  return false;
}
const apiUrl = platformUrl
  +`/api/v1/storage/uploadLeadFile/`
  + platformToken;
let fileUrl = "@" + attachments[0]?.url
let requestBody = {
    files: [fileUrl], // Передаём массив файлов
    lead_id: leadId // ID лида, к которому прикрепляется файл
}
let requestHeaders = {}
if (lead.getChannelCode() === 'telegram') {
  httpRequest.useProxy(true)
}
let uploadResponse = httpRequest.sendFileFromUrl(
  "POST",
  apiUrl,
```

```
  requestBody,
  requestHeaders,
  true
)
database.insertBotLogs(uploadResponse?.response)

lead.setAttr("file", JSON.stringify(uploadResponse?.response))
return false
```

Редактирование настроек скрипта с типом Fallback

В название скрипта Fallback рекомендуем добавить “+ ФАЙЛЫ” для оперативного поиска в списке скриптов. Необходимо проверить корректность настроек NLP в скрипте:



Затем наполните скрипт командами.

Для работы с разными мессенджерами добавьте отдельные команды с условиями:

```
return lead.getChannelCode()
  === "telegram";
```

Выполнить JavaScript

```
return lead.getChannelCode()
  === "max";
```

```
// Все входящие файлы
let attachments = bot.getAllAttachments()

if (Boolean(attachments?.[0]?.url)) {
  const uploadData = bot.downloadFileFromUrl(attachments[0].url)

  if (lead.getData('bot_debug_level')) {
    let data = 'Пользователь отправил файл: ' + uploadData.url;
    bot.sendMessage("🛠 Отладочная информация \n\n" + data);
  }

  //lead.setAttr('file', uploadData.url)
  lead.setJsonAttr('fileDebug', uploadData)
  // Запуск скрипта...
  bot.runScriptByCodeForLead("recieveFile", leadId)

  bot.stop()
}
```

Выполнить JavaScript

```
return lead.getChannelCode()
  === "vk";
```

```
// Все входящие файлы
let attachments = bot.getAllAttachments()

if (Boolean(attachments?.[0]?.url)) {
  const uploadData = bot.downloadFileFromUrl(attachments[0].url)

  if (lead.getData('bot_debug_level')) {
    let data = 'Пользователь отправил файл: ' + uploadData.url;
```

- Для начала добавьте команды обработки файлов для каждого канала бота (описаны ниже).
- Следующей командой должна идти команда **Выполнить JavaScript**, которая срабатывает в случае, когда файла в сообщениях **нет** (проверка в условии). JavaScript подбирает случайный ответ из списка, фиксирует дату сообщения, заносит в Google таблицу нераспознанный текст;

```
// Условие
return !(memory.getAttr('is_attachment')*1)
```

и

```
// JavaScript

var randomStrings = [
        "Не совсем понимаю, о чём вы.",
        "Извините, я вас не понял. Перефразируйте, пожалуйста.",
        "Попробуйте воспользоваться разделами меню или сказать другими словами.",
        "Простите, не понял вас, попробуйте снова."
    ];

randomIndex = Math.ceil((Math.random()*randomStrings.length-1));
```

```
lead.setAttr('randomAnswer', randomStrings[randomIndex]);
```

- **Второй командой** должна идти команда **Отправить текст** с рандомным текстом, которая срабатывает в случае, когда файла в сообщениях **нет** (проверка в условии);

```
// Условие
return !(memory.getAttr('is_attachment')*1)
```

Текст команды
{{ $randomAnswer }}

- **Третей командой** должна идти команда **Отправить текст** с текстом, которая срабатывает в случае, когда в сообщении **есть** файл (проверка в условии);

```
// Условие
return (memory.getAttr('is_attachment')*1)
```

Текст команды
Файл принят, передадим на изучения группе поддержки. С вами свяжутся либо через чат-бота, либо через имеющиеся контакты. А пока что можете воспользоваться автоматическим функционалом чат-бота.

- При необходимости **четвертой командой** должна идти команда **Email**, которая срабатывает в случае, когда в сообщении **есть** файл (проверка в условии). На почту отправляется письмо с файлом пользователя;

```
// Условие
return (memory.getAttr('is_attachment')*1)
```

Получатель	email, на который необходимо отправить письмо. (**{{ $bot.botSupportMails }}**)
Тема	***Название бота поддержки***: лид отправил в бота файл
Содержимое	В чат-боте пользователь прислал в бота файл ID клиента для поиска в диалогах на платформе Metabot: {{ &$leadId }} Ссылка на файл: {{ $is_attachment }} С уважением, Ваш Metabot

> При необходимости, можно создать системный атрибут с ключом **botSupportMails**, типом **variable** и значением равным всем почтам, на которые должно поступить

письмо.

- **Последней командой** должна идти команда **Повторить вопрос**.

Виджет

Для обработки файлов виджета необходим следующий перечень команд:

- **Первой командой** должна идти команда **Выполнить JavaScript**, которая получает из вебхука событие: есть ли в сообщении с распознанным текстом файл и добавляет переменную **is_attachment**, что файл есть;

```
let whJob = bot.getWebhookJob()
let eventType = whJob.event_type

memory.setAttr('is_attachment', false)
if (eventType === 'user_attachments'){
  memory.setAttr('is_attachment', true) bot.disableRepeatMessageText()
  // погасит вывод сообщения с повтором вопроса
  bot.hideRepeatMessageButtons()
  // погасит вывод меню с повтором вопроса
}
```

- **Второй командой** должна идти команда **Выполнить JavaScript**, которая фиксирует сам файл в хранилище и формирует ссылку для передачи в последствии в сообщениях/в письме на почту;

```
// Библиотека с помощью которой можно записать в карочку лида данные не отправля их в тг
let TelegramMessage = require('Common.Integrations.Telegram')
let msg = new TelegramMessage()

// Все входящие файлы let attachments = bot.getAllAttachments()

if(Boolean(attachments?.[0]?.url)){
  uploadData = bot.downloadFileFromUrl(attachments[0].url) msg.debug('Пользователь отправил
файл: ' + uploadData.url)
  lead.setAttr('file', uploadData.url) // Запуск скрипта...
  bot.runScriptByCodeForLead("recieveFile", lead.getData('id'))
  bot.stop()
}
```

Telegram

Для обработки файлов Telegram необходим следующий код:

```
// Библиотека с помощью которой можно записать в карочку лида данные не отправля их в тг
let TelegramMessage = require('Common.Integrations.Telegram')
let msg = new TelegramMessage()

// Все входящие файлы
let attachments = bot.getAllAttachments()

if(Boolean(attachments?.[0]?.url)){

uploadData = bot.downloadFileFromUrl(attachments[0].url)
msg.debug('Пользователь отправил файл: ' + uploadData.url)

lead.setAttr('file', uploadData.url)
// Запуск скрипта...
bot.runScriptByCodeForLead("recieveFile", lead.getData('id'))

bot.stop()
}
```

VK и MAX

Для обработки файлов VK и MAX необходим следующий код:

```
// Все входящие файлы
let attachments = bot.getAllAttachments()

if (Boolean(attachments?.[0]?.url)) {
  const uploadData = bot.downloadFileFromUrl(attachments[0].url)

  if (lead.getData('bot_debug_level')) {
    let data = 'Пользователь отправил файл: ' + uploadData.url;
    bot.sendMessage("□□□Отладочная информация \n\n" + data);
  }

  lead.setAttr('file', uploadData.url)
  lead.setJsonAttr('fileDebug', uploadData)
```

```
  // Запуск скрипта...
  bot.runScriptByCodeForLead("recieveFile", leadId)

  bot.stop()
}
```

Как достать данные вложений

Для Telegram используйте метод bot.**getAllAttachments**()

```
let attachments = bot.getAllAttachments()
if(Boolean(attachments?.[0]?.url)){
  uploadData = bot.downloadFileFromUrl(attachments[0].url) msg.debug('Пользователь отправил
файл: ' + uploadData.url)
  lead.setAttr('file', uploadData.url)
}
```

Для других каналов bot.**getWebhookPayload**()

```
let attachments = bot.getWebhookPayload()
if(attachments.payload.attachments[0].url){
  uploadData = bot.downloadFileFromUrl(attachments.payload.attachments[0].url)
  lead.setAttr('file', uploadData.url)
}
```

В данных примерах в атрибут записывается url вложения для скачивания с помощью метода bot.**downloadFileFromUrl**().

Версия #6
Ирина Петрова создал 2024-04-23 12:47:32 UTC
Алена Рвачева обновил 2026-07-24 09:31:17 UTC