УДК 004

Использование webhook для telegram bot

Вихляев Дмитрий Романович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье содержится описание способа взаимодействия пользователя с телеграмм ботом через webhook. Скрипт, для создания запросов, написан на языке программирования php. В результате исследования будут рассмотрены способ установки webhook для телеграмм бота и примеры данных приходящих от пользователя.
Ключевые слова: телеграмм бот, php, webhook, json.

Using webhook for telegram bot

Vikhlyaev Dmitry Romanovich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes how a user interacts with a telegram bot via a webhook. The script for creating queries is written in the php programming language. As a result of the study, the method of installing a webhook for bot telegrams and examples of data coming from the user will be considered.
Keywords: telegram bot, php, webhook, json.

1 Введение

1.1 Актуальность

Webhook – это механизм веб-разработки, который позволяет приложению отправлять автоматические HTTP-уведомления или POST-запросы на другое приложение, когда происходят определенные события в первом приложении. Webhook запускаются автоматически, не требуя ручного участия пользователя или приложения. Отправляет HTTP-запросы до тех пор, пока сервер второго приложения не ответит на них положительно или не получит код ошибки. Это обеспечивает более надежную доставку запросов. Webhook позволяет быстро реагировать на определенные события, такие как обновление данных или новые записи в базе данных. Можно использовать для автоматизации рутинных задач, таких как отправка email-уведомлений или обновление статуса заказа. С помощью Webhook телеграмм бот получает сообщения, и обновления в режиме реального времени. Он позволяет боту получать уведомления о новых сообщениях без необходимости постоянного

опроса сервера телеграмма. В результате этого уменьшается нагрузка на сервер бота и ускоряется обмен сообщениями между ботом и пользователем.

1.2Обзор исследований

Д.Д.Тихонов обеспечил мобильность через общение с ботом [1]. С.М.Сарсимбаева, Н.А.Маден провели исследование вопросов разработки чатбота на платформе node.js [2]. И.В.Анищенко, О.Б.Боднарь разработали чат-бота для тестирования персонала [3]. М.С.Лежнин создала telegram бота на python [4]. А.А.Харевич, А.М.Кумратова охарактеризовали Telegram bot api как средство маркетинга и менеджмента [5]. А.Г.Навасардян, Н.А.Зубач, Н.А.Хромова, А.В. Некрасов спроектировали базы данных для телеграмм-бота [6].

1.3Цель исследования

Цель исследования – описать установку webhook для телеграмм бота и привести примеры обрабатываемых данных отправляемых сервером телеграмма.

2 Материалы и методы

Для реализации post запроса и используется язык программирования php. Визуализация данных осуществляется с помощью редактора файлов json.

3 Результаты и обсуждения

Webhook работает по принципу "подписки" - бот регистрируется на сервере телеграмма и получает уникальный URL-адрес (webhook-URL), которому будут отправляться уведомления о новых сообщениях и обновлениях. Когда пользователь отправляет сообщение в чат, сервер телеграмма доставляет это сообщение на webhook-URL, а затем бот получает уведомление с сообщением.

Чтобы подключить webhook в телеграмм боте, необходимо зарегистрироваться в телеграмме и создать бота. Затем получить уникальный URL-адрес (webhook-URL) для бота, используя HTTPS-протокол. И зарегистрировать бота на сервере телеграма, указав webhook-URL.

Для использования самоподписанных SSL-сертификатов для webhook в телеграмм bot необходимо сначала создать и усиановит SSL-сертификат на сервере. После, указать URL-адрес, на который должны приходить уведомления, содержащий адрес сервера с сертификатом.

При использовании самоподписанных SSL-сертификатов в телеграмм боте может возникнуть проблема с тем, что телеграм не доверяет сертификату. В этом случае необходимо создать корневой сертификат и установить его на устройствах пользователей, которые будут использовать бота. Ниже представлен пример кода на языке php для отправки запроса на установку webhook (рис.1).

```php
// Установка токена бота и URL сервера
$botToken = 'YOUR_BOT_TOKEN';
$serverUrl = 'YOUR_SERVER_URL';

// Получение содержимого самоподписанного SSL-сертификата
$certificate = fopen('path/to/certificate.pem', 'r');

// Формирование данных для запроса
$data = array(
'url' => $serverUrl,
'certificate' => $certificate
);

// Отправка запроса на установку webhook
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "https://api.telegram.org/bot" . $botToken . "/setWebhook");
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$response = curl_exec($ch);
curl_close($ch);

// Обработка ответа
if ($response === false)
{
    echo 'Ошибка запроса';
}
else {
    $responseJson = json_decode($response);
    if ($responseJson->{'ok'})
    {
        echo 'Webhook установлен';
    }
    else
    {
        echo 'Ошибка установки webhook: ' . $responseJson->{'description'};
    }
}
```

Рис. 1. Запрос на установку webhook для телеграмм бота

Необходимо настроить бота, чтобы он отвечал на входящие сообщения. Когда пользователь отправляет, какое либо сообщение боту, данные отправляются на сервер телеграмма. Затем обработанную информацию телеграмм сервер через webhook, отправляет на сервер бота. Все данные приходят в формате json. Все сообщения можно классифицировать на два вида. Либо пользователь сам сформировал текст или файл и отправил боту, либо нажал на кнопку, имеющую возвращаемое значение.

Отправленные json данные имеют определённую структуру, благодаря которой легко можно определить какое сообщение и от кого пришло.

Основным полем, содержащимся в json, является "update_id". Этот идентификатор может быть использован ботом для отслеживания состояния чата и корректного ответа на каждое обновление. Каждый "update_id" является уникальным и увеличивается с каждым новым обновлением.

Второе поля является массивом и может принимать одно из двух значений «message» или «callback_query». Когда пользователь нажимает кнопку в телеграмм боте, то в формате json бот получает объект CallbackQuery, который содержит следующие поля:

- id: уникальный идентификатор для данного CallbackQuery.

- from: информация об отправителе данного запроса.
- message: сообщение, на которое была нажата кнопка.
- chat_instance: уникальный идентификатор чата, в котором была нажата кнопка.
- data: данные, которые были переданы в кнопке в виде строки (этот параметр может отсутствовать, если кнопка не содержит данных).
- inline_message_id: идентификатор сообщения типа inline, которое сгенерировало кнопку (этот параметр может отсутствовать, если кнопка была создана для обычного сообщения).

Пример полного содержания представлен ниже (рис.2).

Рис. 2. Содержание json данных при нажатии пользователем на кнопку

В любом другом случае при отправке какого-либо сообщения боту, json данные будут содержать поле message.

Полный список значений при отправке пользователем текста представлен ниже (рис.3).

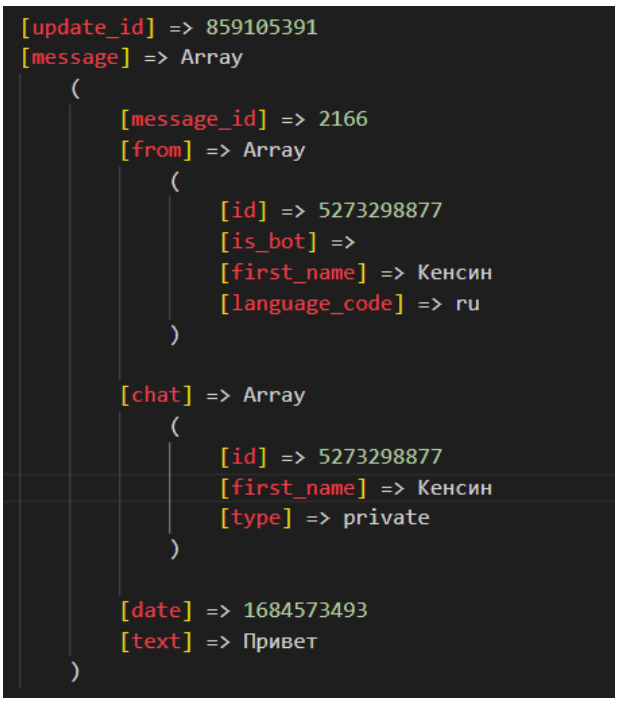

Рис. 3. Содержание json данных при отправлении пользователем текста

При отправке других данных будет манятся только поле со значением «text» на другое. Когда пользователь отправляет документ Telegram-боту, информация о документе отправляется в формате json следующим образом (рис.4).

В этом примере ключ «document» содержит информацию о отправленном документе, включая его имя файла, тип MIME, идентификатор файла и размер файла.

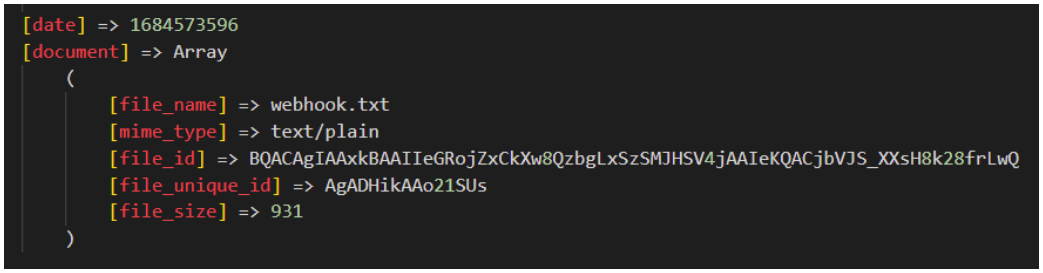

Рис. 4. Содержание поля «document»

При отправке изображения Telegram-боту, он получает информацию о фото в формате json с массивом из четырех элементов, где каждый элемент представляет одну из доступных размеров этого фото. Эти четыре элемента соответствуют четырем размерам, предоставляемым Telegram при отправке фото:

1. file_id – идентификатор файла;
2. file_unique_id – уникальный идентификатор файла;
3. width – ширина изображения в пикселях;
4. height – высота изображения в пикселях.

Для каждой полученной фотографии Telegram создает идентификатор файла и уникальный идентификатор файла, чтобы бот мог использовать эту информацию в дальнейшем. В свою очередь, ширина и высота помогают боту понимать размеры изображения и при необходимости выполнять его обработку(рис.5).

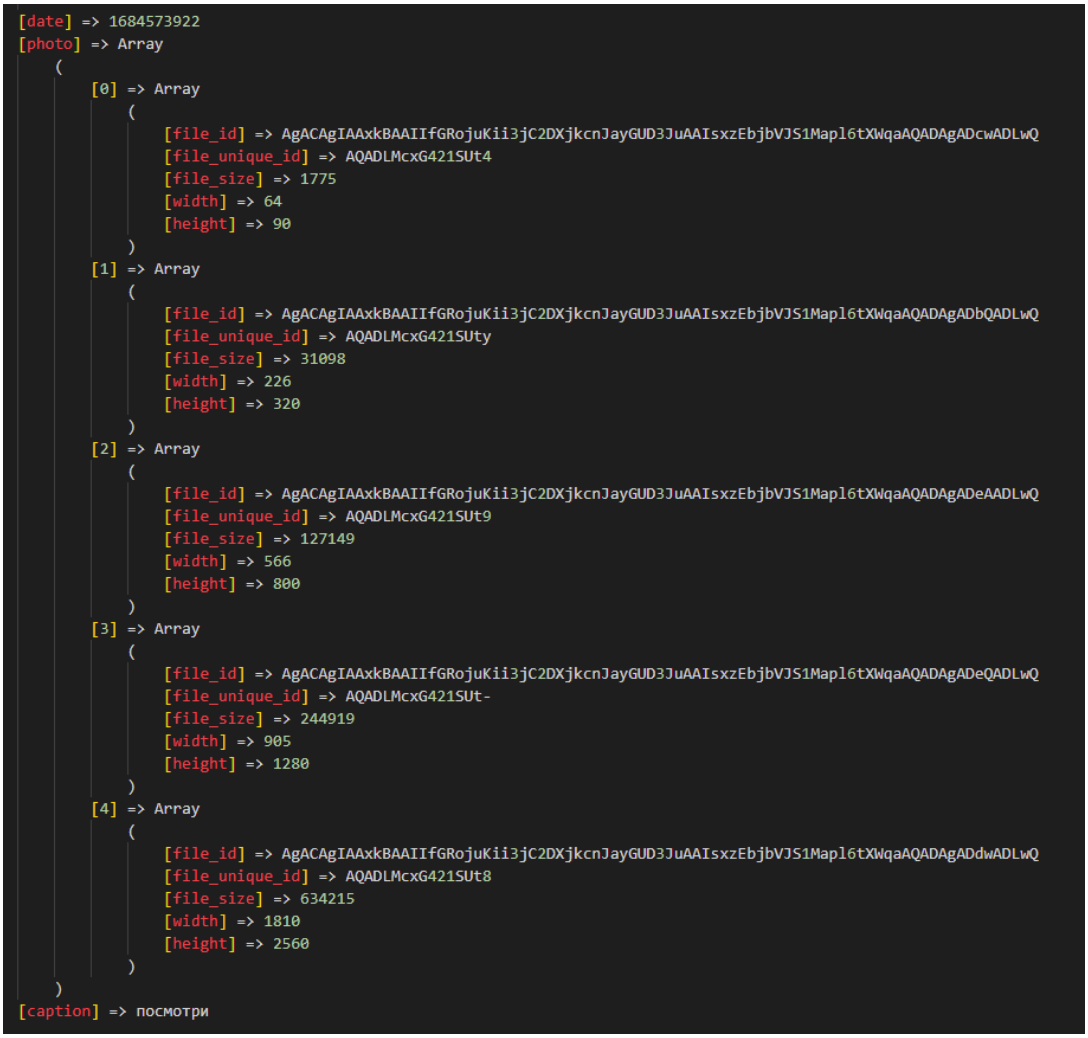

Рис. 5. Содержание поля «Photo»

Когда пользователь отправляет видео в Telegram бота, Telegram отправляет информацию о видео в формате json в виде объекта Video. Объект Video состоит из следующих полей:

- file_id (string): уникальный идентификатор файла;
- width (integer): ширина видео;
- height (integer): высота видео;
- duration (integer): длительность видео (в секундах);
- thumb (PhotoSize, опционально): миниатюра к видео;
- file_name (string, опционально): имя файла видео;
- mime_type (string, опционально): MIME-тип файла;
- file_size (integer, опционально): размер файла.

При добавлении надписи к объекту, формируется отдельное полк «caption» (рис.6).

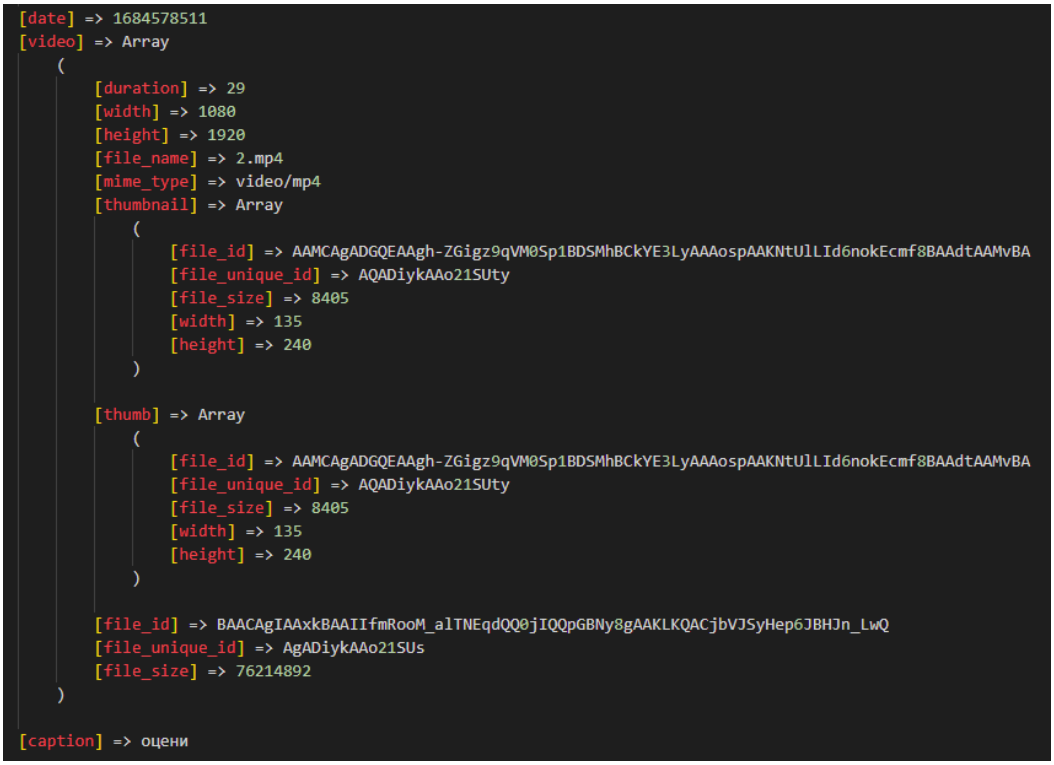

Рис. 6. Содержание поля «Video»

Все данные полученные с помощью webhook можно использовать для:

- Анализа пользовательского поведения: данные json могут содержать информацию о том, как пользователь взаимодействует с ботом, и что ему нравится.
- Оптимизации работы бота: данные json могут помочь оптимизировать работу бота, улучшить его функциональность и удобство использования.
- Построения аналитики: данные json могут быть использованы для построения различных видов аналитики, например, посещаемости, частотности использования функций бота, доли успешных выполнений задач и других параметров.
- Сохранения данных: данные json можно сохранять в базе данных для последующего использования.
- Отправки ответов на сообщения: при получении новых сообщений от пользователя, бот может анализировать json-данные и отправлять наиболее подходящий ответ.
- Создания отчетов: на основе данных можно создавать отчеты, включающие параметры, такие как количество запросов, количество пользователей, время отклика и т.д.
- Разработки новых функций: данные могут быть использованы для оптимизации существующих функций или для разработки новых. Например, на основе данных о предпочтениях пользователей можно

разработать новый функционал, который будет наиболее интересен этой аудитории.

Выводы

В результате данной статьи, был описан способ взаимодействия пользователя с ботом в мессенджере телеграмм через webhook. Также продемонстрированы примеры отправляемых боту данных при разных действиях пользователя.

Библиографический список

1. Тихонов Д.Д. Обеспечение мобильности через общение с ботом // Энигма. 2019. № 12-1. С. 136-139.
2. Сарсимбаева С.М., Маден Н.А. Исследование вопросов разработки чатбота на платформе node.js// Актуальные научные исследования в современном мире. 2020. № 4-1 (60). С. 210-215.
3. Анищенко И.В., Боднарь О.Б. Разработка чат-бота для тестирования персонала // В сборнике: Инновационные технологии, экономика и менеджмент в промышленности. сборник научных статей IX международной научной конференции. Научно-производственное предприятие «Медпромдеталь». Волгоград, 2021. С. 71-76.
4. Лежнин М.С. Разработка telegram бота на python // В сборнике: Юность Большой Волги. сборник статей лауреатов XXIV Межрегиональной конференции-фестиваля научного творчества учащейся молодежи. Чебоксары, 2022. С. 178-179.
5. Харевич А.А., Кумратова А.М. Telegram bot api как средство маркетинга и менеджмента // В сборнике: информационное общество: современное состояние и перспективы развития. сборник материалов X международного студенческого форума.. 2018. С. 201-202.
6. Навасардян А.Г., Зубач Н.А., Хромова Н.А., Некрасов А.В. Разработка базы данных для телеграмм-бота // В сборнике: Технологии 2022: основные проблемы и направления развития. Сборник статей II Международной научно-практической конференции. Пенза, 2022. С. 67-69.
7. Шакиров Р.И., Татаурова А.С. Автоматизация учебного расписания через telegram-bot // Символ науки: международный научный журнал. 2020. № 1. С. 37-41.