DOI 10.37882/2223-2966.2024.06.29

ВЫБОР МЕТОДА СБОРА СТАТИСТИКИ В VK API: АНАЛИЗ CALLBACK API И LONG POLL API

CHOOSING A METHOD FOR COLLECTING STATISTICS IN THE VK API: ANALYSIS OF THE CALLBACK API AND LONGPOLL API

A. Pimanov
V. Samokhina

Summary. The documentation for developers of the VK social network describes its two main tools for receiving and processing events. This article provides a comparative analysis of two methods of interaction with VK API — Callback API and Long Poll API — in the context of collecting statistics in group chats. Both methods allow you to monitor user activity and collect various data about messages, but each of them has its own characteristics and advantages.

The purpose of this material is to determine the optimal method for collecting statistics. The article discusses the technical features and examples of the use of both methods, as well as provides recommendations for choosing the most appropriate option, depending on the initial conditions. The results obtained are valuable for developers interested in integrating with the VK API. They will help you make more informed decisions when choosing tools for project implementation, ensuring more effective achievement of your goals.

Keywords: VK, API, Callback, LongPoll, script, event.

Пиманов Андрей Евгеньевич
Технический институт (филиал) федерального государственного автономного образовательного учреждения высшего образования «Северо-Восточный федеральный университет имени М.К. Аммосова» в г. Нерюнгри
epimanov15@gmail.com
Самохина Виктория Михайловна
Доцент, кандидат педагогических наук, Технический институт (филиал) федерального государственного автономного образовательного учреждения высшего образования «Северо-Восточный федеральный университет имени М. К. Аммосова» в г. Нерюнгри
vsamokhina@bk.ru

Аннотация. В документации для разработчиков социальной сети VK описано два её основных инструмента для получения и обработки событий. В данной статье проводится сравнительный анализ двух методов взаимодействия с VK API — Callback API и Long Poll API — в контексте сбора статистики в групповых чатах. Оба метода позволяют производить мониторинг активности пользователей и собирать различные данные о сообщениях, но каждый из них имеет свои недостатки и преимущества.

Цель данного материала заключается в определении оптимального метода для сбора статистики. В статье рассматриваются технические особенности и примеры использования обоих методов, а также представлены рекомендации по выбору наиболее подходящего варианта в зависимости от исходных условий. Полученные результаты представляют ценность для разработчиков, заинтересованных в интеграции с VK API. Они помогут принимать более обоснованные решения при выборе инструментов для реализации проектов, обеспечивая более эффективное достижение поставленных целей.

Ключевые слова: VK, API, Callback, LongPoll, скрипт, событие.

Существует два основных способа взаимодействия с VK API: Callback API и Long Poll API. Оба инструмента описывают формат обмена данными между клиентом и социальной сетью.

Callback API — инструмент отслеживания активности, в котором передача данных инициируется после возникновения одного из событий на которое была оформлена подписка. Сервер обращается к клиенту, сообщая об активности и прикладывает сопутствующую информацию. Основное различие между этими двумя инструментами заключается в том, какая сторона принимает активное участие в обмене данными. В Long Poll API активную позицию занимает клиент, так как он опрашивает сервер для сбора прошедших событий.

Выбор одного из инструментов определит разницу в механизмах взаимодействия между клиентом и сервером, а также в наборе данных, которые предоставляет социальная сеть.

При рассмотрении двух инструментов VK API: Callback API и Long Poll API — для сбора статистики в групповых чатах, было принято решение использовать PHP для демонстрации различий обоих способов взаимодействия. Этот выбор обусловлен простотой и широкой поддержкой языка.

В данном контексте, стоит упомянуть и другие популярные языки программирования и объяснить почему, демонстрационные примеры представлены только на PHP.

Поддержка VK API существует в Java, но является менее широкой и документированной, чем для PHP. Java

Таблица 1.

Сравнение Callback API и Long Poll API

Характеристика	Callback API	Long Poll API
Способ получения обновлений	Ожидание асинхронного вызова от VK	Опрос сервера VK с заданным интервалом
Требуемые настройки	Необходимо настроить сервер для принятия HTTP-запросов от VK	Не требуется настройка сервера, но требуется выполнение HTTP-запросов к VK
Задержка обновлений	Обновления приходят практически мгновенно	Есть задержка между запросами на сервер VK и получением обновлений
Надежность	Более надежный, поскольку обновления не пропускаются при правильной настройке сервера	Менее надежный из-за возможности пропуска обновлений при сбоях сети или сервера
Загрузка сервера	Большая, поскольку сервер должен постоянно быть готов к приему запросов от VK	Меньше, поскольку запросы на сервер VK отправляются по мере необходимости
Объем трафика	Меньше, так как обновления поступают только при наличии событий	Больше, из-за постоянных запросов к серверу VK
Подходит для	Приложений, где важна скорость реагирования на события	Приложений, где не так критично время реакции на события

обычно требует более объёмных структур, приводящих к существенному увеличению длинны кода для достижения тех же целей, что и PHP. Java не так проста в освоении для новичков и может потребовать больше времени на разработку.

Python также имеет поддержку VK API, но уровень документации и актуальности библиотек является не таким высоким. В некоторых случаях Python может быть медленнее PHP из-за своей интерпретируемой природы. В контексте сбора статистики, где высокая производительность не критична, выбор PHP может быть предпочтительнее из-за более широкого опыта работы веб-разработчиков с этим языком.

Возможно, самый близкий аналог PHP среди языков программирования, используемых на стороне сервера (в контексте решения задачи — клиента), является Node.js. В то же время, Node.js обеспечивает асинхронную обработку событий, но для использования VK API он все равно требует сторонних библиотек, которые могут быть менее стабильными или менее документированными.

В результате PHP является самым предпочтительным выбором для реализации интеграции с VK API, что обусловлено его широкой поддержкой и обширной документацией.

В рамках решения задачи, потребуется не только реализовать сбор статистики, но и обеспечить её хранение. Для этого можно использовать любую реляционную базу данных. Каждая таблица должна описывать чат в социальной сети. Структура такой таблицы представлена ниже:

Таблица 2.

Структура таблицы группового чата

Наименование	Тип данных	Описание
user_id	int	Уникальный числовой идентификатор пользователя в социальной сети, полю задан первичный ключ
messages	int	Количество отправленных пользователем сообщений
messages_length	int	Длина в символах всех отправленных пользователем сообщений

Выбор конкретной системы управления базами данных не имеет особого значения, поскольку рассматриваемая задача не предъявляет высоких требований к функциональности СУБД. Более того, предпочтительно выбирать широко используемые решения для обеспечения простоты в разработке. Именно поэтому для решения задачи рассматривается одна из наиболее распространённых СУБД — MySQL. MySQL широко применяется в веб-разработке, имеет обширное сообщество пользователей и разработчиков, а также обладает хорошей документацией и множеством инструментов для работы с базами данных.

Первой рассмотрена реализация скрипта с помощью Callback API, так как она является наиболее простой.

Чтобы реализовать скрипт, собирающий статистику активности в беседах VK посредством Callback API, в первую очередь необходимо предусмотреть в данном алгоритме механизм подтверждения адреса сервера. Этот шаг необходим для обеспечения соответствия фактического адреса сервера, с адресом, указанным в параметрах Callback API.

Для хранения строки подтверждения, которую сервер должен будет вернуть, в случае получения советующего события, следует завести переменную $confirmation_token, содержащую эту строку подтверждения.

Следующим шагом необходимо получить и декодировать событие от VK, а обработанный результат сохранить в переменную. Сделать это можно следующим способом:

Листинг 1

```
$data = json_decode(file_get_contents('php://input'));
```

Скрипту предстоит обрабатывать только два типа событий: confirmation (подтверждение адреса сервера) и message_new (входящее сообщение). Тип входящего события хранится в $data->type. Обработку событий разных типов можно реализовать с помощью switch-case, как это показано ниже:

Листинг 2

```
switch ($data->type) {
// Если это уведомление для подтверждения адреса, отобразить строку подтверждения
case 'confirmation':
echo $confirmation_token;
break;
case 'message_new':
// Здесь размещён код, для обработки события входящего сообщения
break;
}
```

Код выше полностью описывает только работу с событием confirmation. В случае получения события данного типа, будет выведена строка подтверждения адреса сервера, что позволит его верифицировать. Работу с событием message_new следует начать с проверки: является ли источником сообщения групповой чат. Это можно узнать посмотрев на идентификатор источника сообщения, который хранится в $data->object->message->peer_id. Если он меньше 2000000000, это значит, что сообщение поступило из личных сообщений. В таком случае следует завершить выполнение скрипта командой exit('ok'). Очень важно вывести сообщение «ok», чтобы избежать получения одного и тоже события от VK несколько раз. Отсутствие данного вывода укажет на то, что клиенту не удалось получить входящее событие.

Работу непосредственно с базой, данной в рамках события message_new следует начать с создания подключения к базе данных. Для СУБД MySQL создать подключение можно следующим образом:

Листинг 3

```
$mysqli = new mysqli('localhost', 'statistics_user', 'statistics_password', 'statistics');
```

Если таблицы для хранения статистики беседы не существует, её необходимо создать. Вначале данного действия следует проверить существование таблицы с помощью запроса:

Листинг 4

```
$result = $mysqli->query(«SELECT * FROM `information_schema`.`tables` WHERE table_schema = 'statistics' AND table_name = '{$data->object->message->from_id}' LIMIT 1`»);
```

В переменной $result будет сохранён объект. Обратившись к полю объекта $result->num_rows можно узнать количество строк, полученных в результате выполнения запроса. Если запрос не вернёт ни одной строки, это будет говорить о том, что данной таблицы не существует и её необходимо создать. Для этого можно использовать запрос ниже:

Листинг 5

```
CREATE TABLE `statistics`.`{$data->object->message->peer_id}` ( `user_id` INT NOT NULL , `messages` INT NOT NULL DEFAULT '0' , `messages_length` INT NOT NULL DEFAULT '0' , PRIMARY KEY (`user_id`)) ENGINE = InnoDB;
```

Придерживаясь точно такой же последовательности, следующим шагом необходимо проверить существование записи в таблице, для пользователя, отправившего сообщение. В этом случае поменяются только запросы. Первый запрос будет выглядеть следующим образом:

Листинг 6

```
«SELECT * FROM `{$data->object->message->peer_id}` WHERE `user_id` = {$data->object->message->from_id}».
```

Второй запрос будет выглядеть следующим образом:

Листинг 7

```
«INSERT INTO `{$data->object->message->peer_id}` (`user_id`, `messages`, `messages_length`) VALUES ('{$data->object->message->from_id}', '0', '0')».
```

В поле $data->object->message->from_id хранится идентификатор пользователя отправившего сообщение.

Обновить количество сообщений и их суммарную длину можно следующим образом:

Листинг 8

```
UPDATE `{$data->object->message->peer_id}` SET `messages` = `messages` + 1, `messages_length` = `messages_length` + « . mb_strlen($data->object->message->text) . « WHERE `user_id` = {$data->object->message->from_id}
```

Финальным шагом при работе с Callback API должен стать вывод сообщения «ok». Если этого не сделать, VK отправит тот же запрос клиенту ещё несколько раз.

Реализация данного алгоритма посредством Long Poll API существенно отличается. На старте необходимо инициализировать переменные $access_token (токен сообщества), $group_id (идентификатор сообщества, положительное число) и $api_version (версия API). Следующим шагом создаётся подключение к базе данных. Делается это сразу, так как скрипт будет выполняться до его принудительной остановки.

С помощью метода messages.getLongPollServer необходимо получить набор данных включающий поля key (секретный ключ сессии), server (адрес сервера быстрых сообщений) и ts (номер последнего события). Извлечь и сохранить этот набор можно следующим образом:

Листинг 9

```
$data = json_decode(file_get_contents(«https://api.vk.com/method/groups.getLongPollServer?group_id={$group_id}&access_token={$access_token}&v={$api_version}»));
```

Значения ключа, сервера и метки времени хранятся соответственно в полях объекта $data->response->key, $data->response->server и $data->response->ts. Остальная часть кода должна располагаться в бесконечно выполняющемся цикле while (true). Это позволяет прослушивать VK, для получения событий. В переменную $updates будет записан результат прослушивания сервера мгновенных сообщений. В качестве параметров по этому адресу будет переданы значения полей key и ts. Реализовано это следующим образом:

Листинг 10

```
$updates = json_decode(file_get_
contents(«{$server}?act=a_check&key={$key}&ts={$ts}&wait=25»));
```

В данном наборе хранится массив событий, произошедших с последнего изменения параметра ts. В контексте LongPoll термин «ts» обычно означает «временную метку» (timestamp). Это значение представляет собой момент времени последнего события или обновления, полученного клиентом от сервера. При использовании LongPoll, клиент отправляет запрос на сервер и ждет ответа. Если на сервере нет новых данных для передачи клиенту, сервер может задерживать ответ (длинный опрос) до тех пор, пока не произойдет событие или обновление.

Когда происходит событие или обновление, сервер отправляет ответ клиенту с новыми данными и обновленной временной меткой (ts). Эта временная метка указывает клиенту, какие данные были получены последними. В связи с этим необходимо передавать актуальное значение параметра ts, обновляя его при каждом выполнении цикла.

Для одного значения ts может быть возвращено несколько событий. Наиболее эффективно их можно перебрать с помощью цикла foreach, следующим образом:

Листинг 11

```
foreach ($updates->updates as $update) {
// Проверка типа события
if ($update->type != ‘message_new’) {
continue;
}
 // Проверки источника сообщения
 if ($update->object->message->peer_id < 2000000000)
{
 continue;
 }
 // Здесь располагается код сбора статистики
}
```

В данном коде уже встречаются проверки, реализованные в скрипте, использующем Callback API. Проверяется, что полученное событие представляет собой новое сообщение, а его инициатором является пользователь из группового чата. На месте последнего комментария располагается код учёта статистки беседы, с тем лишь изменением, что источником данных становится переменная $update. После завершения цикла foreach, цикл while начинается заново.

Оба скрипта предназначены для сбора статистики об общении пользователей в групповых чатах, однако можно выделить различия в назначении используемых инструментов и их эффективности в зависимости от задачи.

Второй скрипт выполняет бесконечный цикл, который ожидает и обрабатывает обновления, тогда как первый скрипт лишь ожидает новые события и запускается только при их получении. Для первого скрипта нет необходимости в постоянном опросе сервера, что делает его устройство более простым. Реализация через Long Poll API больше нагружает сервер и потребляет больше трафика, так как требует постоянной работы скрипта и, если события приходят не часто, эффективность данного скрипта снижается. Важным различием так же станет то, что первый скрипт существенно требовательнее к архитектуре, выстроенной вокруг него. Он не сможет функционировать, если VK не будет иметь возможности напрямую обращаться к клиенту. Для чего понадобится статичный IP адрес. Таким образом, простота в устройстве первого скрипта нивелируется потребностью в дополнительных инструментах.

Callback API подходит для обработки сообщений в реальном времени, так как сервер VK немедленно отправляет уведомления клиенту о новых событиях. Он неэффективен для больших объемов данных или частых запросов, так как требует постоянного подключения к серверу VK и базе данных. Может иметь задержку при обработке большого количества событий, особенно при использовании общедоступных серверов.

Long Poll API эффективен для непрерывной обработки сообщений в беседах или группах с большим трафиком, так как он использует длинные запросы к серверу. Позволяет мгновенно реагировать на новые сообщения без необходимости ожидания уведомлений от сервера. Может потреблять больше ресурсов, особенно при обработке большого объема данных или при работе на общедоступных серверах.

В целом, для небольших сообществ или при небольшом трафике Callback API может быть более простым и эффективным вариантом, в то время как Long Poll API лучше подходит для более крупных и активных сообществ.

ЛИТЕРАТУРА

1. Официальный веб-сайт VK, раздел «Для разработчиков» [Электронный ресурс]. URL: https://dev.vk.com/ru/reference (дата обращения: 08.04.2024).
2. Официальный веб-сайт «php.net», раздел «Руководство по PHP» [Электронный ресурс]. URL: https://www.php.net/manual/ru/index.php (дата обращения: 08.04.2024).
3. Пиманов, А.Е. Возможности языка программирования VKScript / А.Е. Пиманов // Современная наука: актуальные проблемы теории и практики. Серия: Естественные и технические науки. — 2023. — № 7. — С. 123–126. — DOI 10.37882/2223-2982.2023.07.30. — EDN GGFGZW.
4. Джеймс Р. Грофф, Пол Н. Вайнберг, Эндрю Дж. Оппель. SQL. Полное руководство. — Вильямс, 2018. С. 95–148.
5. Кузнецов М.В., Симдянов И.В. Самоучитель PHP 7. — СПб.: БХВ-Петербург, 2018. С. 143–151.

Журнал «Современная наука: актуальные проблемы теории и практики»