

JSON API – работаем по спецификации

Алексей Авдеев, Neuron.Digital

FC







2018

**Frontend
Conf**

Профессиональная
конференция
фронтенд-разработчиков



0 себе

1.  Алексей Авдеев (<https://github.com/avdeev>)
2.  Из Нижнего Новгорода, работаю в Москве
3.  [Neuron.Digital](https://neuron.digital)
4.  Веду команду из 8 фронтенд-разработчиков
5.  Програмирую с 2002 года
6.  Сейчас активно использую React



Закон тривиальности Паркинсона

**Время, потраченное на
обсуждение пункта, обратно
пропорционально
рассматриваемой сумме**

An aerial photograph of a nuclear power plant. Two massive, white, hourglass-shaped cooling towers dominate the background. In the foreground, there are several large, rectangular industrial buildings with flat roofs, some with HVAC units. To the right, a complex electrical substation with numerous metal structures and insulators is visible. The plant is surrounded by green grass and a dense forest of trees in the distance. The sky is clear and blue.

The Bikeshed Effect

sleep(1)

“

*It was a proposal to make sleep(1) DTRT if given a non-integer argument that set this particular grass-fire off. I'm not going to say anymore about it than that, because it is a much smaller item than one would expect from the length of the thread, and it has already received far more attention than some of the *problems* we have around here.*

Poul-Henning Kamp <phk@freebsd.org>

<http://phk.freebsd.dk/sagas/bikeshed.html>

**Количество шума, создаваемого
изменением, обратно
пропорционально сложности
изменения**

Что важнее?

Формат API или Бизнес задачи

API -

велосипедный сарай

Как избежать

1. Не слушать **советы**
2. Делать так, как **Вы** хотите
3. Спросить себя **важно** ли это?
4. Использовать **объективные** критерии
5. **Не говорить** о том, о чём не хочешь слушать советы
6. **Отпустите**
7. Выберите **любой** предлагаемый вариант

Anti-bikeshedding tool

Начинаем проектировать



REST (2000)

Требования к архитектуре REST

1. Модель клиент-сервер
2. Отсутствие состояния
3. **Единообразие интерфейса**
 1. Идентификация ресурсов
 2. Манипуляция ресурсами через представление
 3. «Самоописываемые» сообщения
 4. Гипермедиа

RESTful-блог

[Блог](#) > [Бизнес](#)

Название статьи один

10 марта 2018 г.

1 комментарий

Категория: [Бизнес](#), [Технологии](#)

Таким образом укрепление и развитие структуры позволяет выполнять важные задания по разработке направлений прогрессивного развития. Равным образом рамки и место обучения кадров обеспечивает широкому кругу (специалистов) участие в формировании системы обучения кадров, соответствует насущным потребностям.

[Читать](#)

Название статьи два

8 марта 2018 г.

2 комментария

Категория: [Бизнес](#), [Менеджмент](#)

Таким образом укрепление и развитие структуры позволяет выполнять важные задания по разработке направлений прогрессивного развития. Равным образом рамки и место обучения кадров обеспечивает широкому кругу (специалистов) участие в формировании системы обучения кадров, соответствует насущным потребностям.

[Читать](#)

Загрузка статей



Подпишитесь на рассылку

Подпишитесь и получайте
полезные статьи на ваше e-mail

[Подписаться](#)

Рубрики

- [Бизнес](#) (5)
- [Технологии](#) (3)
- [Менеджмент](#) (2)

Последние статьи

- [Название статьи один](#)
- [Название статьи два](#)
- [Название статьи три](#)
- [Название статьи четыре](#)

Метки

[Актуальное](#) [Обучение](#) [Технологии](#)
[Руководителям](#) [Менеджмент](#)
[Кейс](#) [Бизнес](#)





CREATE



READ



UPDATE



DELETE

C

R

U

D

Интерфейс

1. Создать статью
2. Получить список статей
3. Получить статью
4. Обновить статью
5. Удалить статью

Интерфейс

Метод	Путь	Действие
GET	/createArticle	Создать статью
GET	/articlesList	Получить список статей
GET	/articles/getById?id=1	Получить статью
GET	/articles/update?newTitle=Заголовок	Обновить статью
GET	/deleteArticle?id=1	Удалить статью

Правильно

Метод	Путь	Действие
POST	/articles	Создать статью
GET	/articles	Получить список статей
GET	/articles/1	Получить статью
PATCH	/articles/1	Обновить статью
DELETE	/articles/1	Удалить статью

**Почему так
правильно?**

Удаляем статью

01. DELETE /articles/1 HTTP/1.1

02. Accept: application/json

01. HTTP/1.1 200 OK

02. Content-Type: application/json

03. null

MIME-типы

- text/plain
- application/octet-stream
- application/pdf
- image/png
- application/json
- application/xml
- application/vnd.ms-excel



Коды ответов



200
OK

Создание сущности

- 201 Created
- 202 Accepted
- 204 No Content
- 403 Forbidden
- 404 Not Found
- 409 Conflict

Создание сущности

01. POST /articles HTTP/1.1

02. Content-Type: application/json

03. { "id": 1, "title": "Про JSON API" }

01. HTTP/1.1 422 Unprocessable Entity

02. HTTP/1.1 403 Forbidden

03. HTTP/1.1 500 Internal Server Error

Возвращайте ошибки

👉 После

01. HTTP/1.1 422 Unprocessable Entity

02. Content-Type: application/json

03.

04. { "errors": [{

05. "status": "422",

06. "title": "Title already exist",

07.]}]}

**Добавим
паджинацию**

< PREV 1 2 3 4 ... 335 336 NEXT >

Запрос списка

01. GET /articles HTTP/1.1

02. Content-Type: application/json

01. HTTP/1.1 200 OK

02. [

03. { "id": 1, "title": "Про JSON API"},

04. { "id": 2, "title": "Про XML-RPC"}

05.]

После

01. GET /articles?page[size]=30&page[number]=2

02. Content-Type: application/json

01. HTTP/1.1 200 OK

02. {

03. "data": [{ "id": 1, "title": "JSON API"}, ...],

04. "meta": { "count": 10080 }

05. }

Или так

01. GET /articles?page[offset]=30&page[limit]=30

02. Content-Type: application/json

01. HTTP/1.1 200 OK

02. {

03. "data": [{ "id": 1, "title": "JSON API"}, ...],

04. "meta": { "count": 10080 }

05. }

Или так

01. GET /articles?page[published_at]=1538332156

02. Content-Type: application/json

01. HTTP/1.1 200 OK

02. {

03. "data": [{ "id": 1, "title": "JSON API"}, ...],

04. "meta": { "count": 10080 }

05. }

Проблема N + 1

Выведем 10 статей с указанием автора

1. 1 запрос на получение статей
2. 10 запросов для получения авторов каждой статьи

Итого: 11 запросов

Добавляем связи

Запрос списка со связями

01. GET /articles?include=author

02. Content-Type: application/json



Решение: запрос списка со связями

01. HTTP/1.1 200 OK

02. { "data": [{

03. { attributes: { "id": 1, "title": "JSON API" },

04. { relationships: {

05. "author": { "id": 1, "name": "Avdeev" } }

06. }, ...

07.]}]}

Проблема дублирования данных

Выведем 10 статей с указанием автора, у всех статей один автор

Итого: один автор включен в ответ 10 раз



Решение: нормализация данных

01. HTTP/1.1 200 OK

02. { "data": [{

03. "id": "1", "type": "article",

04. "attributes": { "title": "JSON API" },

05. "relationships": { ... }

06. }, ...]

07. }



Решение: нормализация данных

01. HTTP/1.1 200 OK

02. { "data": [{

03. ...

04. "relationships": {

05. "author": { "id": 1, "type": "people" } }

06. }

07. }, ...]

08. }



Решение: нормализация данных

01. HTTP/1.1 200 OK

02. {

03. "data": [...],

04. "included": [{

05. "id": 1, "type": "people",

06. "attributes": { "name": "Avdeev" }

07. }]

08. }

**Нужны не все
поля ресурса**

Решение

GET /articles/1?fields[article]=title HTTP/1.1

01. HTTP/1.1 200 OK

02. { "data": [{

03. "id": "1", "type": "article",

04. "attributes": { "title": "Про JSON API" },

05. }, ...]

06. }

Поиск по статьям

Решение

GET /articles/1?filters[search]=api HTTP/1.1

GET /articles/1?filters[from_date]=1538332156 HTTP/1.1

GET /articles/1?filters[is_published]=true HTTP/1.1

GET /articles/1?filters[author]=1 HTTP/1.1

Нужна сортировка

Решение

GET /articles/1?sort=title HTTP/1.1

GET /articles/1?sort=published_at HTTP/1.1

GET /articles/1?sort=-published_at HTTP/1.1

GET /articles/1?sort=author,-published_at HTTP/1.1

Нужно поменять URLs

Решение: гипермедиа

```
01. GET /articles HTTP/1.1
02. {
03.   "data": [{
04.     ...
05.     "links": { "self": "http://localhost/articles/1" },
06.     "relationships": { ... }
07.   }],
08.   "links": { "self": "http://localhost/articles" }
09. }
```

Решение: гипермедиа

```
01.  ...
02.  "relationships": {
03.    "comments": {
04.      "links": {
05.        "self": "http://localhost/articles/1/relationships/comments",
06.        "related": "http://localhost/articles/1/comments"
07.      }
08.    }
09.  }
```

Новый Media Type

“

(registered 2013-07-21, last updated 2013-07-21)

Name : Steve Klabnik

Email : steve&steveklabnik.com

MIME media type name : Application

MIME subtype name : Vendor Tree - vnd.api+json

<http://www.iana.org/assignments/media-types/application/vnd.api+json>

{json:api}

Экосистема JSON API

Список реализаций спецификации — <http://jsonapi.org/implementations/>

1. 170 различных реализаций
2. для 32 языков программирования
3. и это только добавленные в каталог
4. PR Welcome

+ Плюсы JSON API

1. Общее соглашение для всех
2. Меньше споров внутри команды
3. Высокая производительность разработки
4. Популярные проблемы уже решены
5. Прост для понимания
6. Лаконичен
7. Open Source



Минусы JSON API

1. **Фронтенд**: надо парсить ответы
2. **Бэкенд**: контроль вложенности
3. **Бэкенд**: сложность запросов к БД
4. **Бэкенд**: безопасность
5. Спека сложно читается (не все смогли)
6. Не все либы реализуют спеку хорошо

Подводные камни JSON API



Количество relationships в выдаче неограничено

GET /articles/1?include=comments HTTP/1.1

01. ...

02. "relationships": {

03. "comments": {

04. "data": [0 ... ∞]

05. }

06. }

Правильно

GET /comments?filters[article]=1&page[size]=30 HTTP/1.1

01. {

02. "data": [0 ... 29]

03. }

Неоднозначность

GET /articles/1?include=comments HTTP/1.1

GET /articles/1/comments HTTP/1.1

GET /comments?filters[article]=1 HTTP/1.1



Полиморфные связи "один ко многим"

```
01. GET /comments?include=commentable HTTP/1.1
02.
03. ...
04. "relationships": {
05.   "commentable": {
06.     "data": { "type": "article", "id": "1" }
07.   }
08. }
```



Сложные связи "многие ко многим"

01. GET /users?include=users_comments HTTP/1.1

02.

03. ...

04. "relationships": {

05. "users_comments": {

06. "data": [{ "type": "users_comments", "id": "1" }, ...]

07. },

08. }

Swagger

POST **/pet** Add a new pet to the store



PUT **/pet** Update an existing pet



GET **/pet/findByStatus** Finds Pets by status



GET **/pet/findByTags** Finds Pets by tags



GET **/pet/{petId}** Find pet by ID



POST **/pet/{petId}** Updates a pet in the store with form data



DELETE **/pet/{petId}** Deletes a pet



POST **/pet/{petId}/uploadImage** uploads an image



Pet ∨ {

id

category

name*

photoUrls*

tags

status

}

integer(\$int64)

Category > {...}

string

example: doggie

> [...]

> [...]

string

pet status in the store

Enum:

∨ [available, pending, sold]

▼ {
 data

MatchWithRelationships ▼ {

 id string(\$uuid)

 type string

 attributes

 > {...}

 relationships

 ▼ {

 group

 > {...}

 }

}

included

 ▼ [▼ {

 oneOf ->

 Group > {...}

 Team > {...}

 Stage > {...}

 Match > {...}

 Stadium > {...}

 }]

}

Альтернативы

OData (2015)

0Data (2015) - the best way to REST

01. GET `http://services.odata.org/v4/TripRW/People` HTTP/1.1

02. OData-Version: 4.0

03. OData-MaxVersion: 4.0

0Data (2015) - the best way to REST

```
01. HTTP/1.1 200 OK
02. Content-Type: application/json; odata.metadata=minimal
03. OData-Version: 4.0
04. {
05.   '@odata.context': 'http://services.odata.org/V4/...
06.   '@odata.nextLink': 'http://services.odata.org/V4/...
07.   'value': [{
08.     '@odata.etag': 'W/'08D1D5BD423E5158'',
09.     'UserName': 'russellwhyte',
10.     ...
```

Что с GraphQL?

Высокий порог входа



Эффект большого взрыва

1. Нет никакого API

2. 

3. GraphQL

Или

1. 🦎 Простейший RESTful
2. 🦎 Прогрессивное улучшение
3. 🦎 JSON API



Ад на бэкенде

1. Контроль вложенности
2. Рекурсия
3. Ограничение частоты
4. Контроль доступа

За подробностями

1. <http://jsonapi.org>
2. <http://www.odata.org>
3. <https://graphql.org>
4. <http://xmlrpc.scripting.com>
5. <https://www.jsonrpc.org>



**STOP
BIKESHEDDING
AND
DO
SOMETHING**