

Как я переоткрыл NGINX

История одного фронтендера

Максим Северухин

Senior Frontend Developer
at amma

- занимаюсь веб-разработкой с 2011
- ex. EPAM-er
- с 2017 frontend-разработчик
- спикер на разных конференциях и митапах
- пробовал в **ютуб**, возможно еще вернусь
- Мой стек: **Vue, React, Node**



Ко мне пришел бизнес и...
принес (Не)типичную задачу

(Не)типичная задача

Разрабатываем гибридное приложение.

(Не)типичная задача

Разрабатываем гибридное приложение.

- используем в `WebView` нативного приложения на iOS и Android.

(Не)типичная задача

Разрабатываем гибридное приложение.

- используем в `WebView` нативного приложения на iOS и Android.
- пользователи должны иметь сквозную авторизацию, мы не показываем им повторно форму логина.

(Не)типичная задача

Разрабатываем гибридное приложение.

- используем в `WebView` нативного приложения на iOS и Android.
- пользователи должны иметь сквозную авторизацию, мы не показываем им повторно форму логина.
- нативные приложения хранят данные/настройки пользователя и они нужны нам.

Способ передачи данных

При загрузке WebView мобильное приложение передает данные в HTTP заголовках.

```
GET https://my-super-web-app.me HTTP/1.1
```

```
Authorization: Bearer ...
```

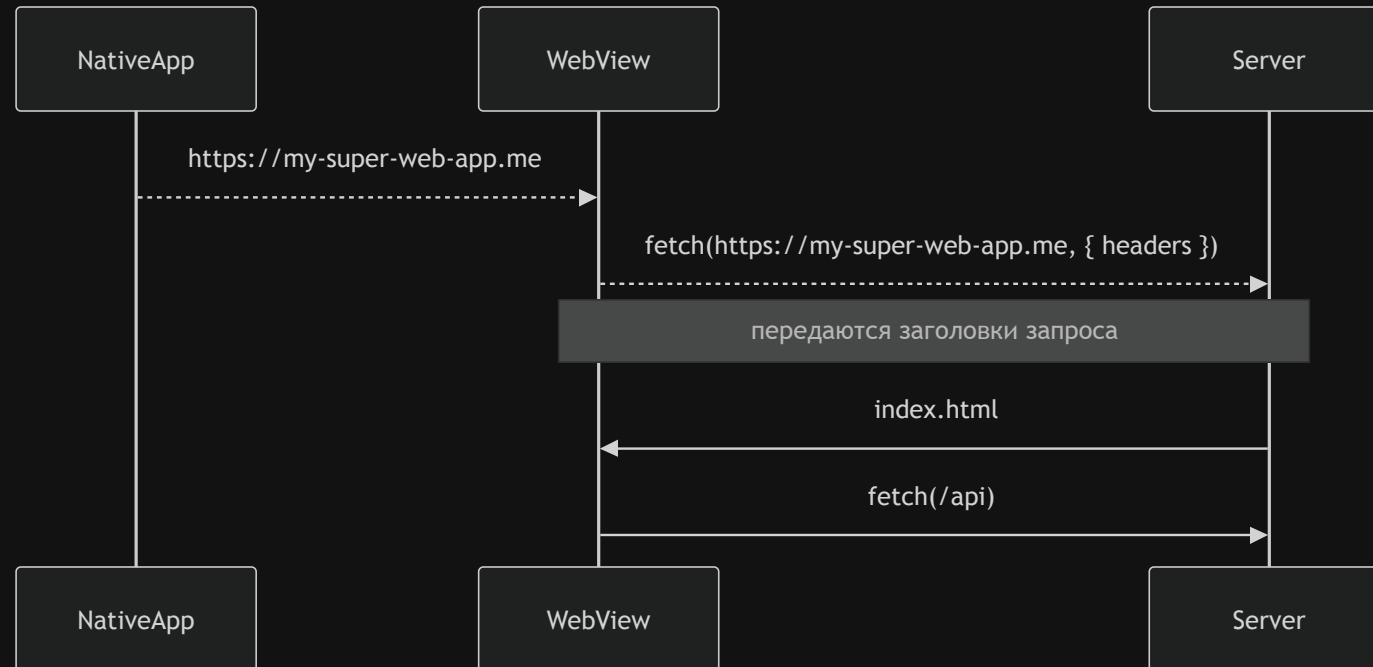
```
X-App-Mode: pregnancy
```

```
X-App-Version: 1.2.3
```

The проблема

The проблема

У нас классическое SPA приложение без SSR и мы не можем получить/сохранить информацию о параметрах, с которым браузер загрузил наш index.html



Прототипируем решение

Идея Number 1

Переводим наше приложение на Next.js рельсы.

Плюсы:

Минусы

Идея Number 1

Переводим наше приложение на Next.js рельсы.

Плюсы:

Минусы

- сможем обработать параметры запроса нашего `index.html` и передать их на фронтенд часть

Идея Number 1

Переводим наше приложение на Next.js рельсы.

Плюсы:

Минусы

- сможем обработать параметры запроса нашего `index.html` и передать их на фронтенд часть
- "ускорим" загрузку приложения

Идея Number 1

Переводим наше приложение на Next.js рельсы.

Плюсы:

- сможем обработать параметры запроса нашего `index.html` и передать их на фронтенд часть
- "ускорим" загрузку приложения

Минусы

- надо пересмотреть все приложение и расставить директивы "`use client`" и "`use server`"

Идея Number 1

Переводим наше приложение на Next.js рельсы.

Плюсы:

- сможем обработать параметры запроса нашего `index.html` и передать их на фронтенд часть
- "ускорим" загрузку приложения

Минусы

- надо пересмотреть все приложение и расставить директивы "`use client`" и "`use server`"
- мы используем сторонние SDK, UIKit которые не поддерживают SSR

Идея Number 1

Переводим наше приложение на Next.js рельсы.

Плюсы:

- сможем обработать параметры запроса нашего `index.html` и передать их на фронтенд часть
- "ускорим" загрузку приложения

Минусы

- надо пересмотреть все приложение и расставить директивы "`use client`" и "`use server`"
- мы используем сторонние SDK, UIKit которые не поддерживают SSR
- Node / Next.js - большие зависимости, которые необходимо поддерживать

Идея Number 1

Переводим наше приложение на Next.js рельсы.

Плюсы:

- сможем обработать параметры запроса нашего `index.html` и передать их на фронтенд часть
- "ускорим" загрузку приложения

Минусы

- надо пересмотреть все приложение и расставить директивы "`use client`" и "`use server`"
- мы используем сторонние SDK, UIKit которые не поддерживают SSR
- Node / Next.js - большие зависимости, которые необходимо поддерживать
- Необходимо изменить весь CI/CD-флоу и работа со стороны DevOps

Идея Number 1

~~Идея Number 1~~

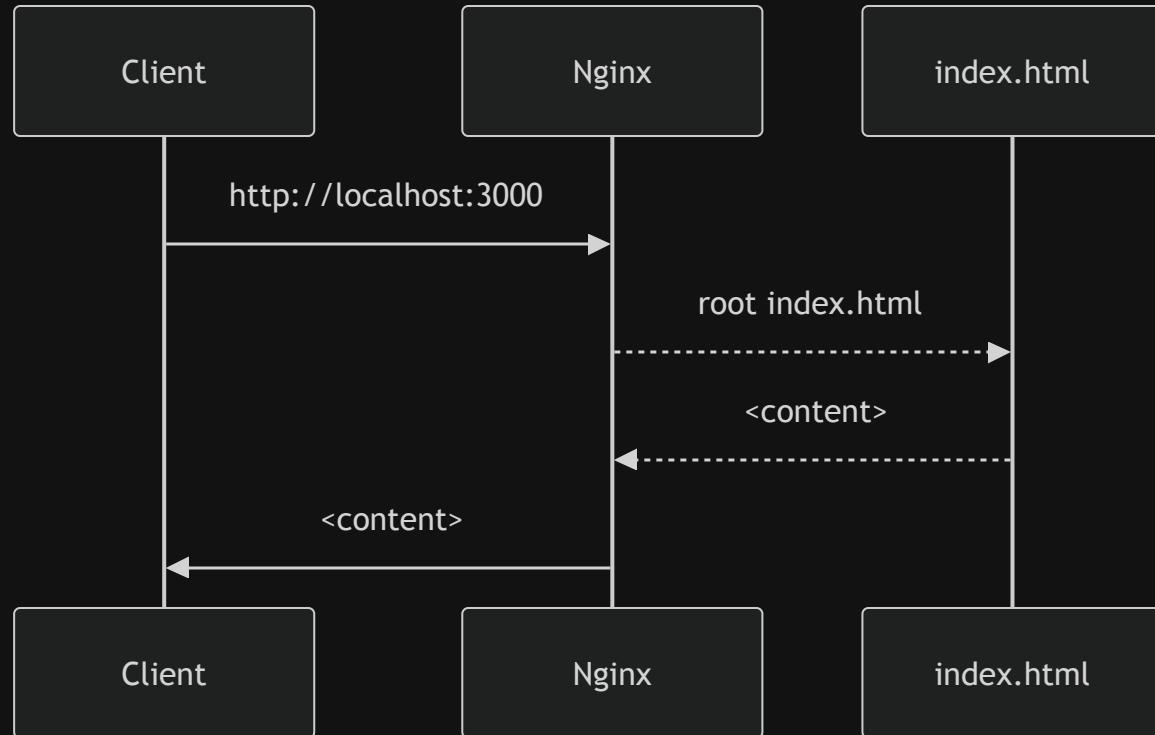
~~Идея Number 1~~

Дорого!

Идея Number 2

Идея Number 2

Надо найти того, кто участвует в общении между клиентом и сервером(приложением)



Nginx

Плюсы:

Минусы:

Nginx

Плюсы:

- Уже используется

Минусы:

Nginx

Плюсы:

- Уже используется
- Знает про все Request и Response

Минусы:

Nginx

Плюсы:

- Уже используется
- Знает про все Request и Response
- Можно дополнять Response, писать условия

Минусы:

Nginx

Плюсы:

- Уже используется
- Знает про все Request и Response
- Можно дополнять Response, писать условия

Минусы:

- Собственный синтаксис

Nginx

Плюсы:

- Уже используется
- Знает про все Request и Response
- Можно дополнять Response, писать условия

Минусы:

- Собственный синтаксис

```
map $http_cookie $a {  
    default "";  
    "~*my_cookie=( [^\\D+; ]+)" $1;  
}
```

```
location / {  
    if ($a) {  
        add_header ...  
    }  
}
```

Идея Number 2

Идея Number 2

Идея Number 2

Nginx. Берем в работу!

Подход 1

Подход 1

Попробуем переложить значение из заголовком в куки.

Значение из кук мы сможем получить в JS-коде и использовать в запросах к API или выводить в интерфейсе пользователю.

```
location /first {  
    if ($http_authorization ~* "^\$1;(.*)") {  
        add_header Set-Cookie "token=$1;"  
    }  
}
```

Повторим упражнение для версии приложения

```
# nginx.conf

location /second {
    if ($http_authorization ~* "^\$http_authorization\s(.*)") {
        add_header Set-Cookie "token=\$1";
    }

    if ($http_x_app_version ~* "^(.*)") {
        add_header Set-Cookie "version=\$1";
    }
}
```

```
GET /second HTTP/1.1
X-App-Version: 0.1.1
Authorization: Bearer 1234
```

```
HTTP/1.1 200 OK
Server: nginx/1.26.2
Content-Type: text/plain
Content-Length: 6
Connection: keep-alive
```

```
Set-Cookie: version=0.1.1;
```

```
GET /second HTTP/1.1
X-App-Version: 0.1.1
Authorization: Bearer 1234
```

```
HTTP/1.1 200 OK
Server: nginx/1.26.2
Content-Type: text/plain
Content-Length: 6
Connection: keep-alive
```

```
Set-Cookie: version=0.1.1;
```



Надо разобраться

```
location /third {
    add_header Set-Cookie "first=1;";

    if ($http_authorization ~* "^Bearer\s(.*)") {
        add_header Set-Cookie "token=$1;";
    }

    if ($http_x_app_version ~* "^(.*)") {
        add_header Set-Cookie "version=$1;";
    }

    add_header Set-Cookie "last=9;";
}
```

```
GET /third HTTP/1.1
X-App-Version: 0.1.1
Authorization: Bearer 1234
```

```
HTTP/1.1 200 OK
Server: nginx/1.26.2
Content-Type: text/plain
Content-Length: 6
Connection: keep-alive
```

```
Set-Cookie: version=0.1.1;
```

```
GET /third HTTP/1.1
X-App-Version: 0.1.1
Authorization: Bearerer 1234
```

```
HTTP/1.1 200 OK
Server: nginx/1.26.2
Content-Type: text/plain
Content-Length: 6
Connection: keep-alive
```

```
Set-Cookie: version=0.1.1;
```



Почему исчезают куки? 🙄

Причина:

Директива `add_header` может наследоваться от предыдущего уровня конфигурации, если в текущем уровне она не используется, а ветвление создает собственный уровень.

Непринятие

Что я еще знаю о Nginx?

Непринятие

Что я еще знаю о Nginx?

- можно написать модуль на lua

Непринятие

Что я еще знаю о Nginx?

- можно написать модуль на lua
- есть директива `tar`, через которую решают подобные задачи

Непринятие

Что я еще знаю о Nginx?

- можно написать модуль на lua
- есть директива `tar`, через которую решают подобные задачи
- ...

О чём я не знал?

О чём я не знал?



NJS

NJS

- поддерживает 2 движка: njs и QuickJS

NJS

- поддерживает 2 движка: njs и QuickJS
- njs

NJS

- поддерживает 2 движка: njs и QuickJS
- njs
 - поддерживает ES5.1 с некоторыми расширениями ES6 и позже

NJS

- поддерживает 2 движка: njs и QuickJS
- njs
 - поддерживает ES5.1 с некоторыми расширениями ES6 и позже
 - предоставляет дополнительные методы и свойства nginx:
 - HTTP Request
 - Stream
 - Headers
 - Crypto
 - Querystring
 - File System
 - process

NJS

- поддерживает 2 движка: njs и QuickJS
- njs
 - поддерживает ES5.1 с некоторыми расширениями ES6 и позже
 - предоставляет дополнительные методы и свойства nginx:
 - HTTP Request
 - Stream
 - Headers
 - Crypto
 - Querystring
 - File System
 - process
 - Есть поддержка TypeScript

Попробуем посчитать сумму

Попробуем посчитать сумму

Попробуем посчитать сумму

```
// main.nginx.js
export function sum(r) {
    return r.args.a + r.args.b;
}
```

Попробуем посчитать сумму

```
// main.nginx.js
export function sum(r) {
    return r.args.a + r.args.b;
}
```

```
# nginx.conf
```

```
server {
# ....
js_set $sum main.sum;

location = /sum {
    return 200 "$sum";
}
# ....
}
```

Запрос

GET /sum?a=1&b=2 HTTP/1.1

Host: localhost:8080

Ответ

HTTP/1.1 200 OK

Server: nginx/1.26.2

Date: Sun, 06 Oct 2024 13:14:13 GMT

Content-Type: text/plain

Content-Length: 7

Connection: keep-alive

Sum: 12

Ура! Работает!



Попробуем реализовать решение

```
export function cookie(r) {
  const authToken = r.headersIn.Authorization.slice(7); // Bearer 1234
  const appVersion = r.headersIn['X-App-Version'];

  let cookies = [];

  if (authToken) {
    cookies.push(`token=${authToken};`);
  }

  if (appVersion) {
    cookies.push(`version=${appVersion}`);
  }

  r.headersOut['Set-Cookie'] = cookies;
}
```

Попробуем реализовать решение

```
# main.nginx.js

location /cookie {
    js_header_filter main.cookie;

    return 200 "Cookie";
}
```

Запрос

```
GET /cookie HTTP/1.1
Host: localhost:8080
X-App-Version: 1.2.3
Authorization: Bearer 12345
Accept: */*
```

Ответ

```
HTTP/1.1 200 OK
Server: nginx/1.26.2
Date: Sun, 06 Oct 2024 14:30:37 GMT
Content-Type: text/plain
Content-Length: 6
Connection: keep-alive
Set-Cookie: token=12345;
Set-Cookie: version=1.2.3;
```

"Cookie"

Логирование

Логирование ошибки синтаксиса

```
18:05:07 2024/10/06 14:05:07 [error] 30#30: *1 js exception: TypeError: cannot g
18:05:07     at cookie (/etc/nginx/njs/main.nginx.js:15)
18:05:07 , client: 192.168.65.1, server: , request: "GET /cookie HTTP/1.1", host
```

Логирование исключений

```
export function error() {
  throw Error('Oppps!');

  r.return(200, 'Hello world');
}
```

```
19:19:23 [error] 30#30: *1 js exception: Error: Oppps!
19:19:23     at error (/etc/nginx/njs/main.nginx.js:39)
19:19:23 , client: 192.168.65.1, server: , request: "GET /error HTTP/1.1", host:
```

Логирование в лог файл

```
/***
 * @param {NginxHTTPRequest} r
 */
function console(r) {
  r.error('Oppps!');
  // r.log('Oppps!');
  // r.warn('Oppps!');

  r.return(200, 'Hello world');
}
```

```
HTTP/1.1 200 OK
Server: nginx/1.26.2
Date: Sun, 06 Oct 2024 15:39:09 GMT
Content-Type: text/plain
Content-Length: 11
Connection: keep-alive
15:39:09 [error] 30#30: *1 js: Oppps!
```

Unhandled promise

```
async function promise(r) {  
    await Promise.reject('Error!');  
  
    r.return(200, 'Hello world');  
}
```

```
15:58:30 [error] 30#30: *1 js exception: Error: unhandled promise rejection: Err
```

Используйте инструменты, которые у вас уже есть

The End!