

Next.js навсегда?

Год использования в проекте



Владимир Морозов



Владимир Морозов

- Вебмастер;



Владимир Морозов

- Вебмастер;
- Fullstack;



Владимир Морозов

- Вебмастер;
- Fullstack;
- Frontend;



Владимир Морозов

- Вебмастер;
- Fullstack;
- Frontend;
- Frontend lead, команды Портал,
Цифромед.

План

- Создадим приложение на Next.js;

План

- Создадим приложение на Next.js ;
- Поговорим, какие были сложности;

План

- Создадим приложение на Next.js;
- Поговорим, какие были сложности;
- Подумаем, куда движется индустрия.

Примерно год назад



Требования

Выбор редакции:

Радио Т
ЕЖЕНЕДЕЛЬНЫЙ
HI-TECH ПОДКАСТ

Радио-Т

#Технологии Russia


Веб-стандарты

#Технологии Russia


Акулы Перса

#Культура Russia


Три тимлида заходят в бар

#Бизнес Russia

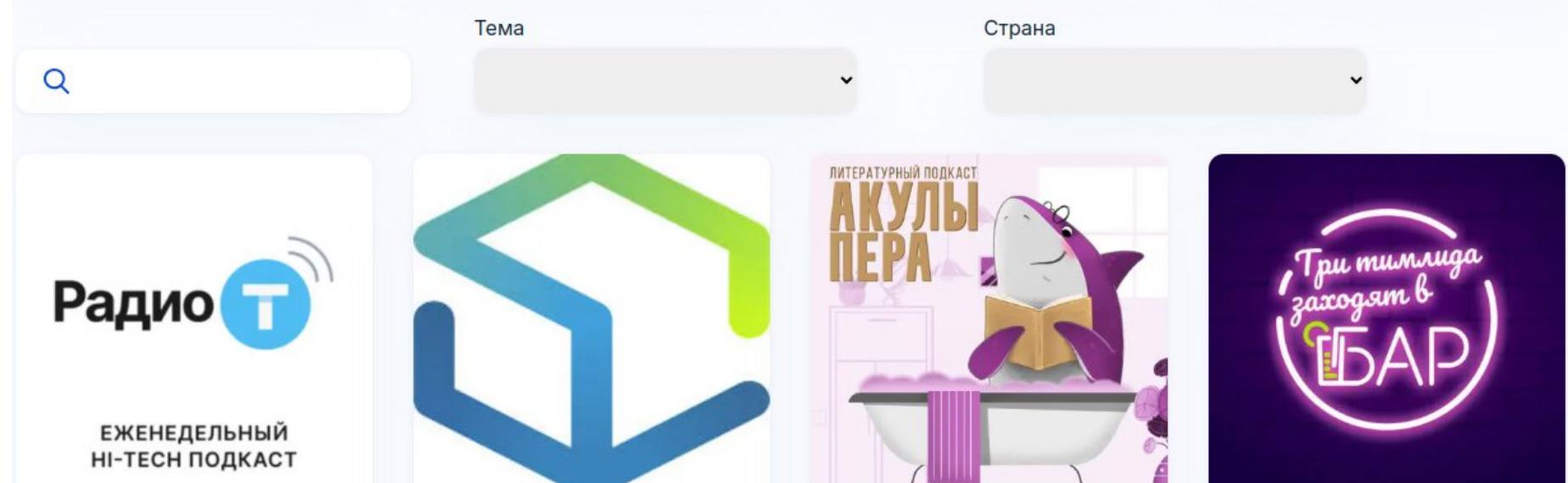
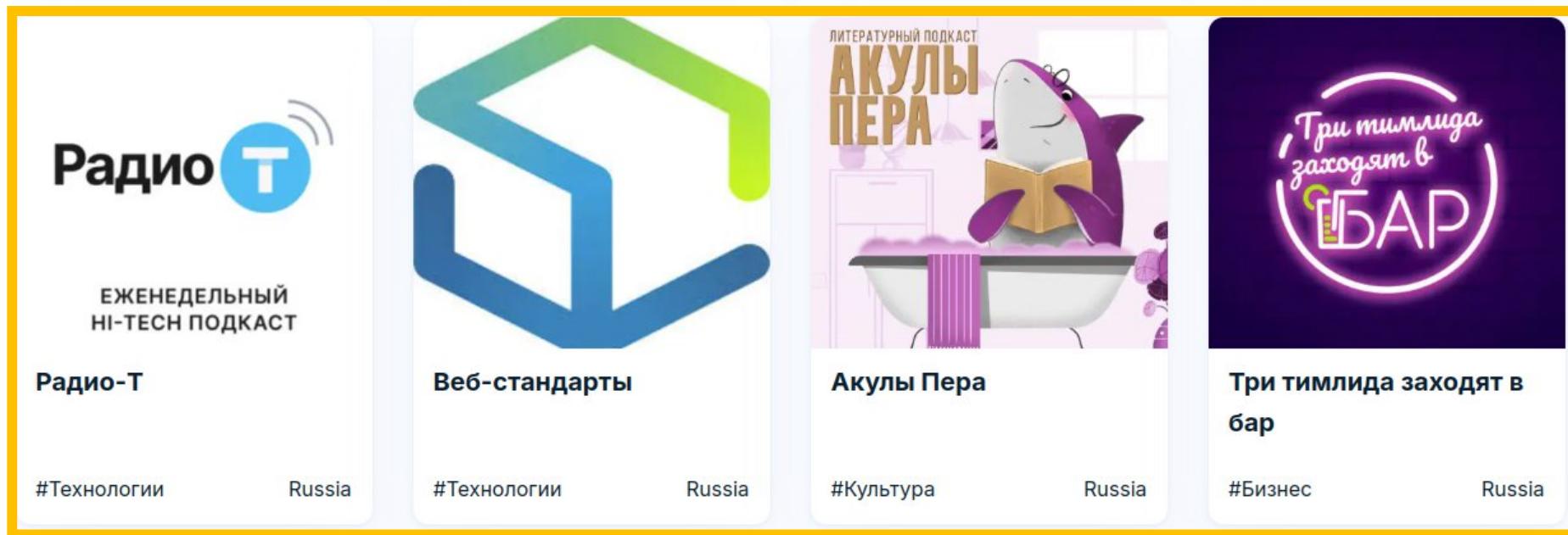
Радио Т
ЕЖЕНЕДЕЛЬНЫЙ
HI-TECH ПОДКАСТ


Веб-стандарты


Акулы Перса


Три тимлида заходят в бар

Выбор редакции:



Выбор редакции:

Радио Т
ЕЖЕНЕДЕЛЬНЫЙ
HI-TECH ПОДКАСТ

Радио-Т

#Технологии Russia

Веб-стандарты

#Технологии Russia

Акулы Пера
ЛИТЕРАТУРНЫЙ ПОДКАСТ

#Культура Russia

Три тимлида заходят в бар

#Бизнес Russia

Тема	Страна
<input type="text"/>	<input type="text"/>

Радио Т
ЕЖЕНЕДЕЛЬНЫЙ
HI-TECH ПОДКАСТ

Веб-стандарты

Акулы Пера
ЛИТЕРАТУРНЫЙ ПОДКАСТ

Три тимлида заходят в бар

Выбор редакции:

Радио Т
ЕЖЕНЕДЕЛЬНЫЙ
HI-TECH ПОДКАСТ

Радио-Т

#Технологии Russia


Веб-стандарты

#Технологии Russia


Акулы Перса

#Культура Russia


Три тимлида заходят в бар

#Бизнес Russia

Тема

Страна



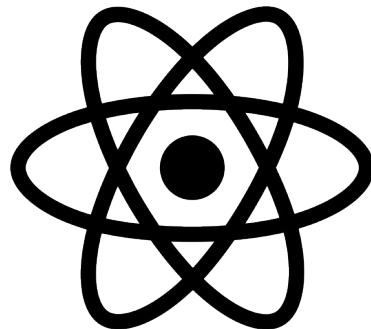
Радио Т
ЕЖЕНЕДЕЛЬНЫЙ
HI-TECH ПОДКАСТ


Веб-стандарты


Акулы Перса

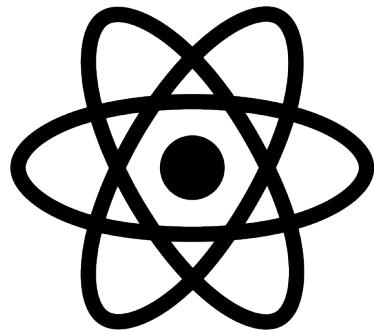

Три тимлида заходят в бар

Требования

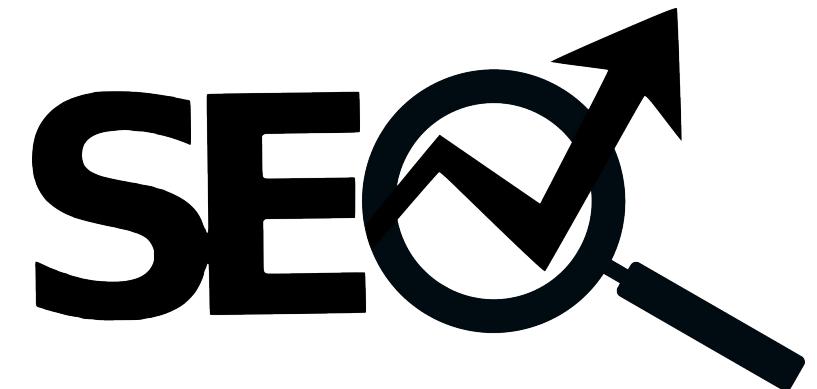


React

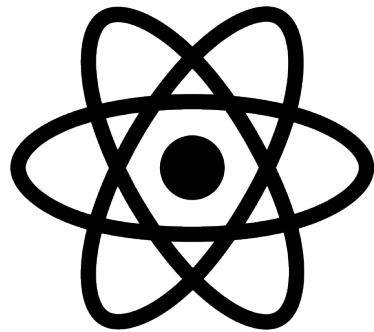
Требования



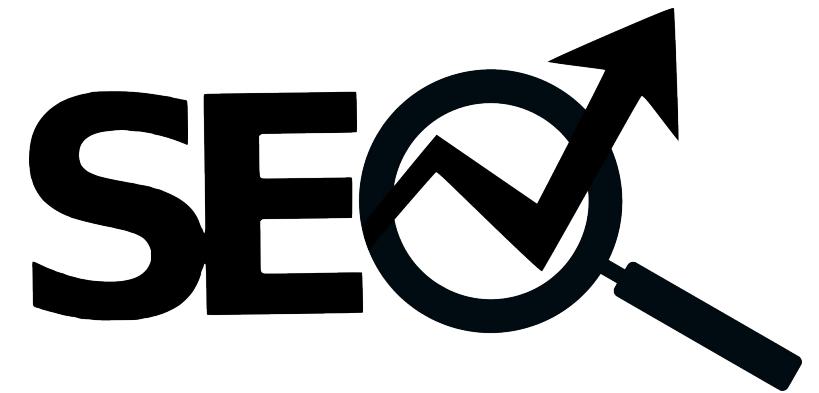
React



Требования



React



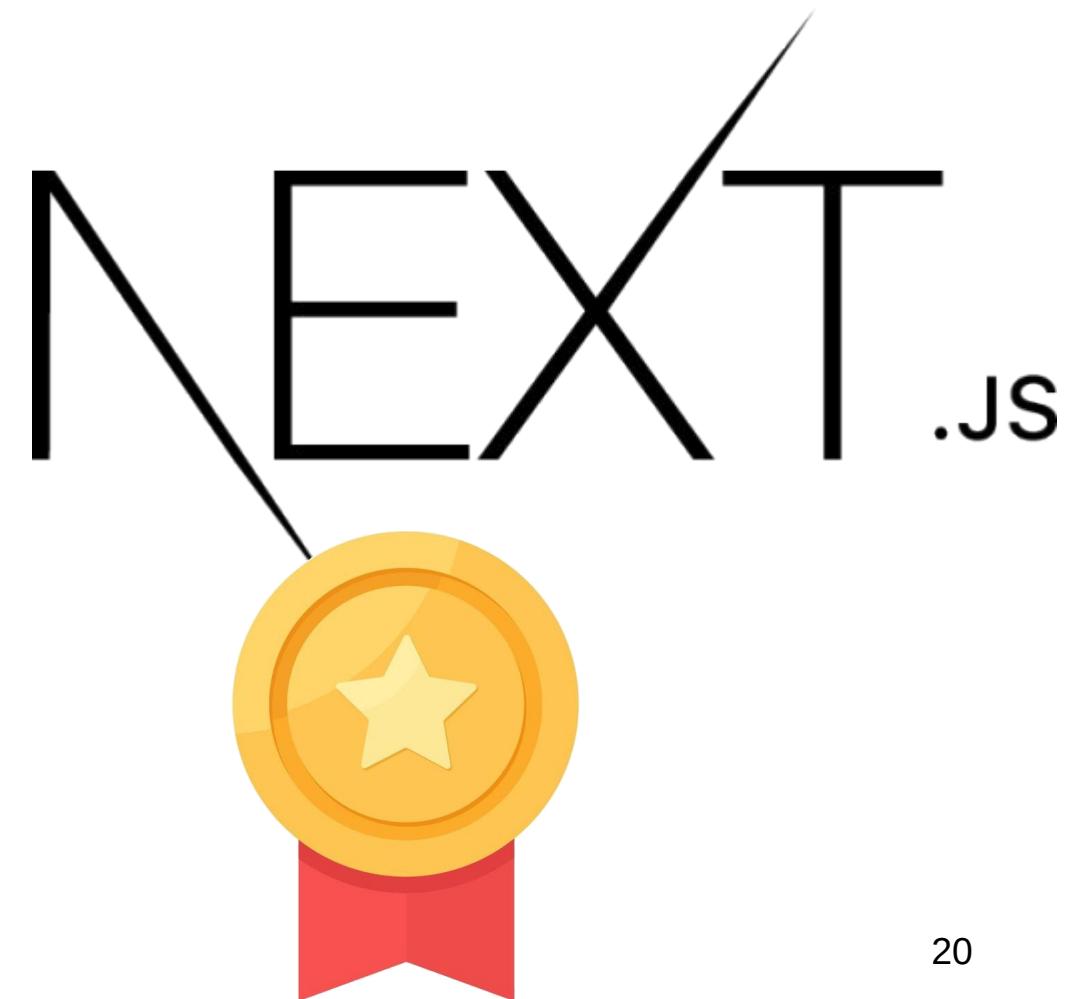
Финалисты

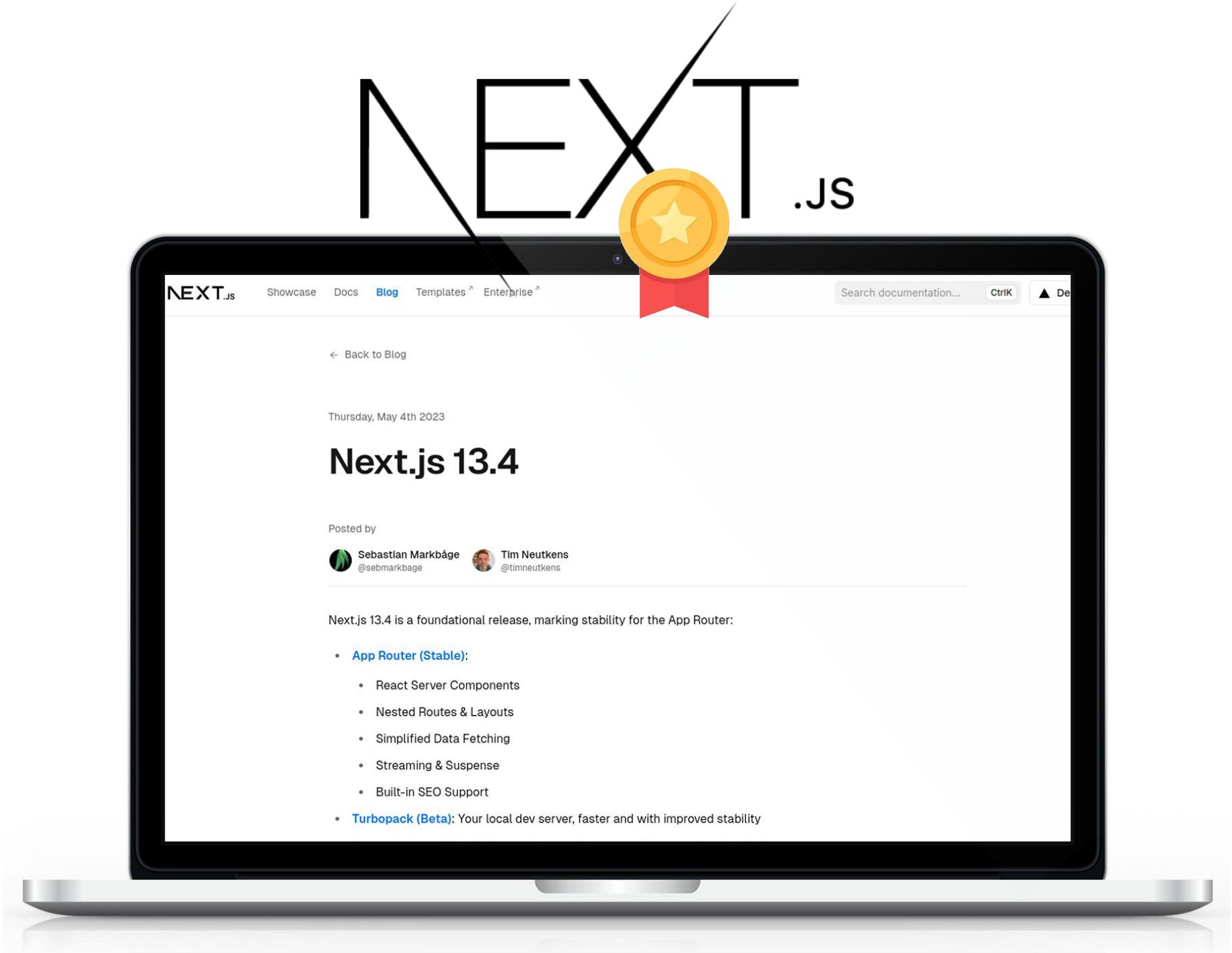
Remix

NEXT.JS

Финалисты

Remix







Количество пользователей больше, чем у Remix

Next.js



Next.js

Next.js is a React framework for building full-stack web applications

90K

Members

26

Online

Top 2%

Rank by size ↗

Remix.js



remixrun

This Subreddit is about the Meta-Framework Remix for React.

720

Members

Top 23%

Rank by size ↗

Отображение данных

Client-Side Rendering

```
1  useEffect(() => {
2    const load = async () => {
3      const res = await fetch(
4        `${process.env.NEXT_PUBLIC_APP_BASE_URL}/api/podcasts?${searchParams}`
5      );
6      const data = await res.json();
7      setPodcasts(data);
8    };
9
10   load();
11
12 }, [searchParams]);
```



Тема

Страна



Радио Т

ЕЖЕНЕДЕЛЬНЫЙ
HI-TECH ПОДКАСТ

Радио-Т



Веб-стандарты



Акулы Перса



Три тимлида заходят в Бар

Elements Components Network Console Sources Performance Memory Application Security Lighthouse Recorder Performance insights Profiler

Preserve log Disable cache No throttling

Filter Invert Hide data URLs Hide extension URLs All Fetch/XHR Doc CSS JS Font Img Media Manifest WS Wasm Other Blocked response cookies Blocked requests 3rd-party requests

podcasts statictics?_rsc=1wtp7

Name	Headers	Preview	Response	Initiator	Timing	Cookies
podcasts						
statictics?_rsc=1wtp7						

50 ms 100 ms 150 ms 200 ms 250 ms 300 ms 350 ms 400 ms 450 ms 500 ms

[{id: 0, name: "Радио-Т", imageKey: "radiot", topic: "Технологии", country: "Russia"},...]
► 0: {id: 0, name: "Радио-Т", imageKey: "radiot", topic: "Технологии", country: "Russia"}
► 1: {id: 1, name: "Веб-стандарты", imageKey: "webstandards", topic: "Технологии", country: "Russia"}
► 2: {id: 2, name: "Акулы_Перса", imageKey: "sharks", topic: "Культура", country: "Russia"}
► 3: {id: 3, name: "Три_тимлида_заходят_в_бар", imageKey: "teamleads", topic: "Бизнес", country: "Russia"}
► 4: {id: 4, name: "Syntax", imageKey: "syntax", topic: "Технологии", country: "USA"}
► 5: {id: 5, name: "Запуск_завтра", imageKey: "tomorrow", topic: "Технологии", country: "Russia"}
► 6: {id: 6, name: "Бреслав_и_Ложечкин", imageKey: "bl", topic: "Бизнес", country: "Russia"}
► 7: {id: 7, name: "Подлодка", imageKey: "podlodka", topic: "Технологии", country: "Russia"}
► 8: {id: 8, name: "This_Week_in_Tech", imageKey: "twih", topic: "Технологии", country: "USA"}

Server-Side Rendering

```
1 async function getFilters() {
2   const res = await fetch(
3     `${process.env.NEXT_PUBLIC_APP_BASE_URL}/api/podcasts/filters`
4   );
5   const { countries, topics } = (await res.json()) satisfies FiltersResponse;
6
7   return (
8     <div className={styles.filters}>
9       <Suspense>
10         <Search />
11         <div className={styles.selects}>
12           <Select searchParam="topic" elements={topics} placeholder="Тема" />
13           <Select
14             searchParam="country"
15             elements={countries}
16             placeholder="Страна"
17             />
18           </div>
19         </Suspense>
20       </div>
21     );
22 }
```

Server-Side Rendering

```
1 async function getFilters() {
2   const res = await fetch(
3     `${process.env.NEXT_PUBLIC_APP_BASE_URL}/api/podcasts/filters`
4   );
5   const { countries, topics } = (await res.json()) satisfies FiltersResponse;
6
7   return (
8     <div className={styles.filters}>
9       <Suspense>
10         <Search />
11         <div className={styles.selects}>
12           <Select searchParam="topic" elements={topics} placeholder="Тема" />
13           <Select
14             searchParam="country"
15             elements={countries}
16             placeholder="Страна"
17           />
18         </div>
19       </Suspense>
20     </div>
21   );
22 }
```

Server-Side Rendering

```
1 async function getFilters() {
2   const res = await fetch(
3     `${process.env.NEXT_PUBLIC_APP_BASE_URL}/api/podcasts/filters`
4   );
5   const { countries, topics } = (await res.json()) satisfies FiltersResponse;
6
7   return (
8     <div className={styles.filters}>
9       <Suspense>
10         <Search />
11         <div className={styles.selects}>
12           <Select searchParam="topic" elements={topics} placeholder="Тема" />
13           <Select
14             searchParam="country"
15             elements={countries}
16             placeholder="Страна"
17           />
18         </div>
19       </Suspense>
20     </div>
21   );
22 }
```

Server-Side Rendering

```
1 async function getFilters() {
2   const res = await fetch(
3     `${process.env.NEXT_PUBLIC_APP_BASE_URL}/api/podcasts/filters`
4   );
5   const { countries, topics } = (await res.json()) satisfies FiltersResponse;
6
7   return (
8     <div className={styles.filters}>
9       <Suspense>
10         <Search />
11         <div className={styles.selects}>
12           <Select searchParam="topic" elements={topics} placeholder="Тема" />
13           <Select
14             searchParam="country"
15             elements={countries}
16             placeholder="Страна"
17             />
18           </div>
19         </Suspense>
20       </div>
21     );
22 }
```

```
24 export default async function MainPage() {
25   return (
26     <main>
27       <p>Выбор редакции:</p>
28       {await getFilters()}
29     </main>
30   );
31 }
```

Server-Side Rendering

Root podcasts Статистика Выйти

Радио Т
ЕЖЕНЕДЕЛЬНЫЙ HI-TECH ПОДКАСТ
Радио-Т

Веб-стандарты

ЛИТЕРАТУРНЫЙ ПОДКАСТ
АКУЛЫ ПЕРА
Акулы Пера

Три тимлида заходят в бар

#Технологии Russia #Технологии Russia #Культура Russia #Бизнес Russia

Elements Components Network Console Sources Performance Memory Application Security Lighthouse Recorder Profiler

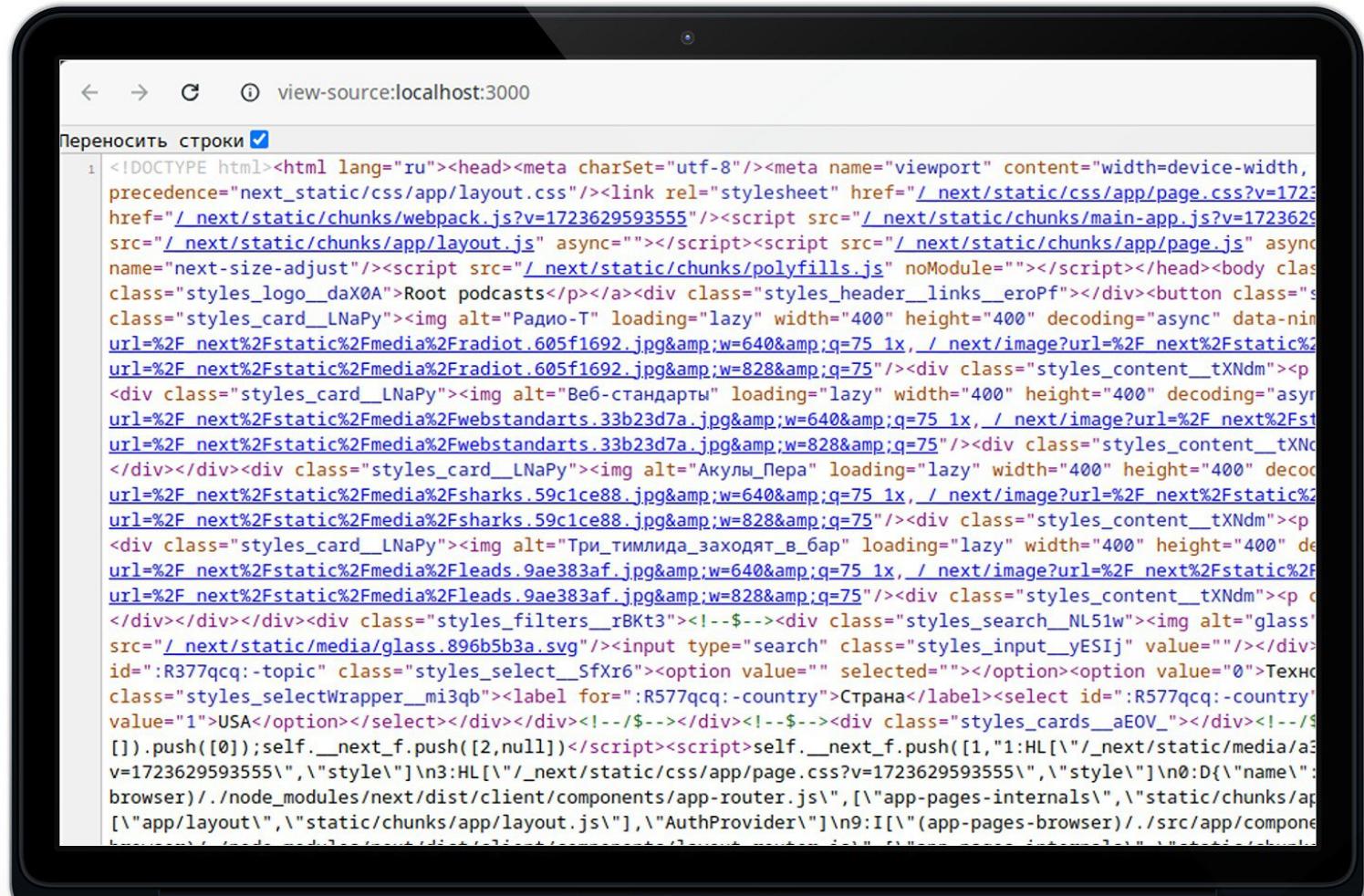
Preserve log Disable cache No throttling

Filter Invert Hide data URLs Hide extension URLs All Fetch/XHR Doc CSS JS Font Img Media Manifest WS Wasm Other Blocked response cookies Blocked requests 3rd-party requests

100 ms 200 ms 300 ms 400 ms 500 ms 600 ms 700 ms 800 ms 900 ms 1000 ms 1100 ms 1200 ms 1300 ms 1400 ms 1500 ms 1600 ms 1700 ms 1800 ms 1900 ms 2000 ms

Name	Status	Protocol	Type	Initiator	Size	Time

Server-Side Rendering



Сила в объединении



Сила в объединении

- Динамический контент загружаем на клиенте;



Сила в объединении

- Динамический контент загружаем на клиенте;
- Условно-статический на сервере.



Как получать данные?

Как получать данные?



Как получать данные?

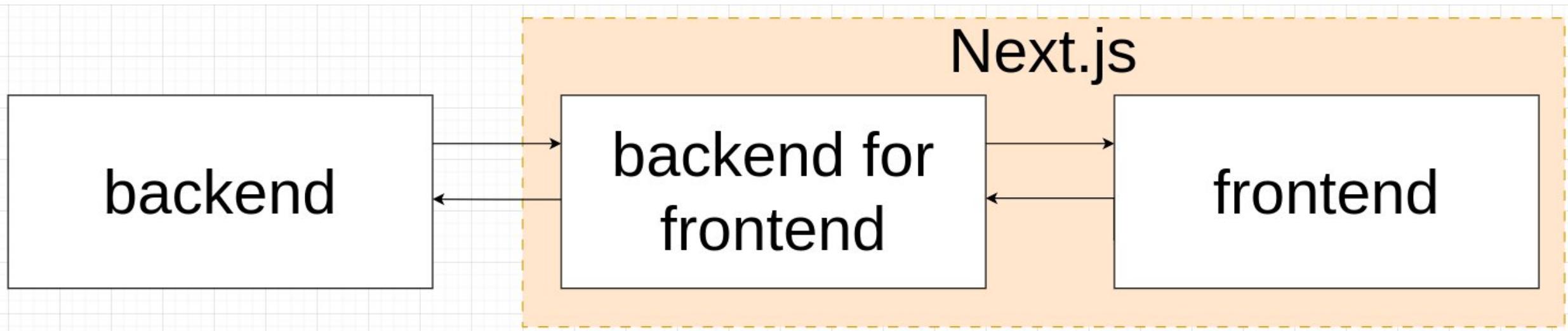
/src/app/page.tsx

```
1 import { Dictionary } from "@/types";
2 import styles from "./styles.module.css";
3
4 async function getFilters() {
5   const [countriesResponse, topicsResponse] = await Promise.all([
6     fetch(`process.env.BASE_BACKEND}/countries`),
7     fetch(`process.env.BASE_BACKEND}/topics`),
8   ]);
9
10  const [countries, topics]: [Dictionary[], Dictionary[]] = [
11    await countriesResponse.json(),
12    await topicsResponse.json(),
13  ];
14
15  return <div>{/*отображаем данные*/}</div>;
16 }
17
18 export default async function MainPage() {
19   return <main className={styles.main}>{await getFilters()}</main>;
20 }
21
```

Как получать данные?



Как получать данные?



your tango



Ogres have layers.
Onions have layers. — Shrek

/src/app/api/podcasts/route.ts

```
1 import { Podcast } from "@/types";
2 import { NextRequest, NextResponse } from "next/server";
3
4 export const dynamic = "force-dynamic";
5
6 export async function GET(request: NextRequest) {
7   const url = new URL(request.url);
8   const search = url.searchParams.get("s");
9
10  const baseUrl = `${process.env.BASE_BACKEND}/podcasts`;
11  let res;
12
13  if (search) {
14    res = await fetch(`/${baseUrl}/search?s=${search}`);
15  } else {
16    res = await fetch(`/${baseUrl}?${url.searchParams}`);
17  }
18
19  const data = (await res.json()) satisfies Podcast[];
20
21  return NextResponse.json(data, { status: res.status });
22}
23
```

/src/app/api/podcasts/route.ts

```
1 import { Podcast } from "@/types";
2 import { NextRequest, NextResponse } from "next/server";
3
4 export const dynamic = "force-dynamic";
5
6 export async function GET(request: NextRequest) {
7   const url = new URL(request.url);
8   const search = url.searchParams.get("s");
9
10  const baseUrl = `${process.env.BASE_BACKEND}/podcasts`;
11  let res;
12
13  if (search) {
14    res = await fetch(`#${baseUrl}/search?s=${search}`);
15  } else {
16    res = await fetch(`#${baseUrl}?${url.searchParams}`);
17  }
18
19  const data = (await res.json()) satisfies Podcast[];
20
21  return NextResponse.json(data, { status: res.status });
22}
23
```

/src/app/api/podcasts/route.ts

```
1 import { Podcast } from "@/types";
2 import { NextRequest, NextResponse } from "next/server";
3
4 export const dynamic = "force-dynamic";
5
6 export async function GET(request: NextRequest) {
7   const url = new URL(request.url);
8   const search = url.searchParams.get("s");
9
10  const baseUrl = `${process.env.BASE_BACKEND}/podcasts`;
11  let res;
12
13  if (search) {
14    res = await fetch(`/${baseUrl}/search?s=${search}`);
15  } else {
16    res = await fetch(`/${baseUrl}?${url.searchParams}`);
17  }
18
19  const data = (await res.json()) satisfies Podcast[];
20
21  return NextResponse.json(data, { status: res.status });
22}
23
```

/src/app/page.tsx

```
1 async function getFilters() {
2   const res = await fetch(
3     `${process.env.NEXT_PUBLIC_APP_BASE_URL}/api/podcasts/filters`
4   );
5   const { countries, topics } = (await res.json()) satisfies FiltersResponse;
6
7   return (
8     <div className={styles.filters}>
9       {/*отображение компонентов*/}
10      </div>
11    );
12 }
```

/src/app/page.tsx

```
1 async function getFilters() {
2   const res = await fetch(
3     `${process.env.NEXT_PUBLIC_APP_BASE_URL}/api/podcasts/filters`
4   );
5   const { countries, topics } = (await res.json()) satisfies FiltersResponse;
6
14 export default async function MainPage() {
15   return (
16     <main>
17       <p>Выбор редакции:</p>
18       {await getFilters()}
19       <Suspense>
20         <CardBlock />
21       </Suspense>
22     </main>
23   );
24 }
```

backend for frontend

Было



backend for frontend

Было



Стало



backend for frontend



**Закрыть доступ к
странице**



Вот так вот

Page

/src/app/statictics

```
1 import { cookies } from "next/headers";
2 import { redirect } from "next/navigation";
3 import { MAIN_LINK } from "@/constants";
4
5 export default function StaticticsPage() {
6   const user = cookies().get("user");
7   if (!user) {
8     redirect(MAIN_LINK);
9   }
10
11   return (
12     <main>
13       //контент страницы
14     </main>
15   );
16 }
```

Page

/src/app/statictics

```
1 import { cookies } from "next/headers";
2 import { redirect } from "next/navigation";
3 import { MAIN_LINK } from "@/constants";
4
5 export default function StaticticsPage() {
6   const user = cookies().get("user");
7   if (!user) {
8     redirect(MAIN_LINK);
9   }
10
11   return (
12     <main>
13       //контент страницы
14     </main>
15   );
16 }
```

Page

/src/app/statictics

```
1 import { cookies } from "next/headers";
2 import { redirect } from "next/navigation";
3 import { MAIN_LINK } from "@/constants";
4
5 export default function StaticticsPage() {
6   const user = cookies().get("user");
7   if (!user) {
8     redirect(MAIN_LINK);
9   }
10
11   return (
12     <main>
13       //контент страницы
14     </main>
15   );
16 }
```

Layout

/src/app/(private)/statictics/layout.tsx

```
1 import { MAIN_LINK } from "@/constants";
2 import { cookies } from "next/headers";
3 import { redirect } from "next/navigation";
4
5 export default function Layout({
6   children,
7 }: Readonly<{
8   children: React.ReactNode;
9 }>) {
10   const user = cookies().get("user");
11   if (!user) {
12     redirect(MAIN_LINK);
13   }
14
15   return children;
16 }
```

Layout

/src/app/(private)/statictics/layout.tsx

```
1 import { MAIN_LINK } from "@/constants";
2 import { cookies } from "next/headers";
3 import { redirect } from "next/navigation";
4
5 export default function Layout({
6   children,
7 }: Readonly<{
8   children: React.ReactNode;
9 }>) {
10  const user = cookies().get("user");
11  if (!user) {
12    redirect(MAIN_LINK);
13  }
14
15  return children;
16 }
```

Middleware

/src/middleware.ts

```
1 import { NextRequest, NextResponse } from "next/server";
2 import { STATISTICS_LINK, MAIN_LINK } from "@/constants";
3
4 const isAuth = (request: NextRequest) => request.cookies.has("user");
5
6 export async function middleware(request: NextRequest) {
7   const pathname = request.nextUrl.pathname;
8
9   if (!isAuth(request) && pathname.startsWith(STATISTICS_LINK)) {
10     return NextResponse.redirect(new URL(MAIN_LINK, request.url));
11   }
12   return NextResponse.next();
13 }
```

Middleware

/src/middleware.ts

```
1 import { NextRequest, NextResponse } from "next/server";
2 import { STATISTICS_LINK, MAIN_LINK } from "@/constants";
3
4 const isAuth = (request: NextRequest) => request.cookies.has("user");
5
6 export async function middleware(request: NextRequest) {
7   const pathname = request.nextUrl.pathname;
8
9   if (!isAuth(request) && pathname.startsWith(STATISTICS_LINK)) {
10     return NextResponse.redirect(new URL(MAIN_LINK, request.url));
11   }
12   return NextResponse.next();
13 }
```

Middleware

/src/middleware.ts

```
1 import { NextRequest, NextResponse } from "next/server";
2 import { STATICICS_LINK, MAIN_LINK } from "@/constants";
3
4 const isAuth = (request: NextRequest) => request.cookies.has("user");
5
6 export async function middleware(request: NextRequest) {
7   const pathname = request.nextUrl.pathname;
8
9   if (!isAuth(request) && pathname.startsWith(STATICICS_LINK)) {
10     return NextResponse.redirect(new URL(MAIN_LINK, request.url));
11   }
12   return NextResponse.next();
13 }
```



Развитие Next.js

- 13.4 - React Server Components;

Развитие Next.js

- 13.4 - React Server Components;
- 13.5 - Исправление ошибок, увеличение скорости;

Развитие Next.js

- 13.4 - React Server Components;
- 13.5 - Исправление ошибок, увеличение скорости;
- 14.1 - Лучший вывод ошибок, logger;

Развитие Next.js

- 13.4 - React Server Components;
- 13.5 - Исправление ошибок, увеличение скорости;
- 14.1 - Лучший вывод ошибок, logger;
- 14.2 - turbopack In dev, Tree-shaking.

Авторизация





Ресурсы



3 Runtime



3 Runtime

API	Purpose	Where	Status Code
<code>redirect</code>	Redirect user after a mutation or event	Server Components, Server Actions, Route Handlers	307 (Temporary) or 303 (Server Action)
<code>permanentRedirect</code>	Redirect user after a mutation or event	Server Components, Server Actions, Route Handlers	308 (Permanent)
<code>useRouter</code>	Perform a client-side navigation	Event Handlers in Client Components	N/A
<code>redirects in next.config.js</code>	Redirect an incoming request based on a path	<code>next.config.js</code> file	307 (Temporary) or 308 (Permanent)
<code>NextResponse.redirect</code>	Redirect an incoming request based on a condition	Middleware	Any

3 Runtime

- `document.cookie;`

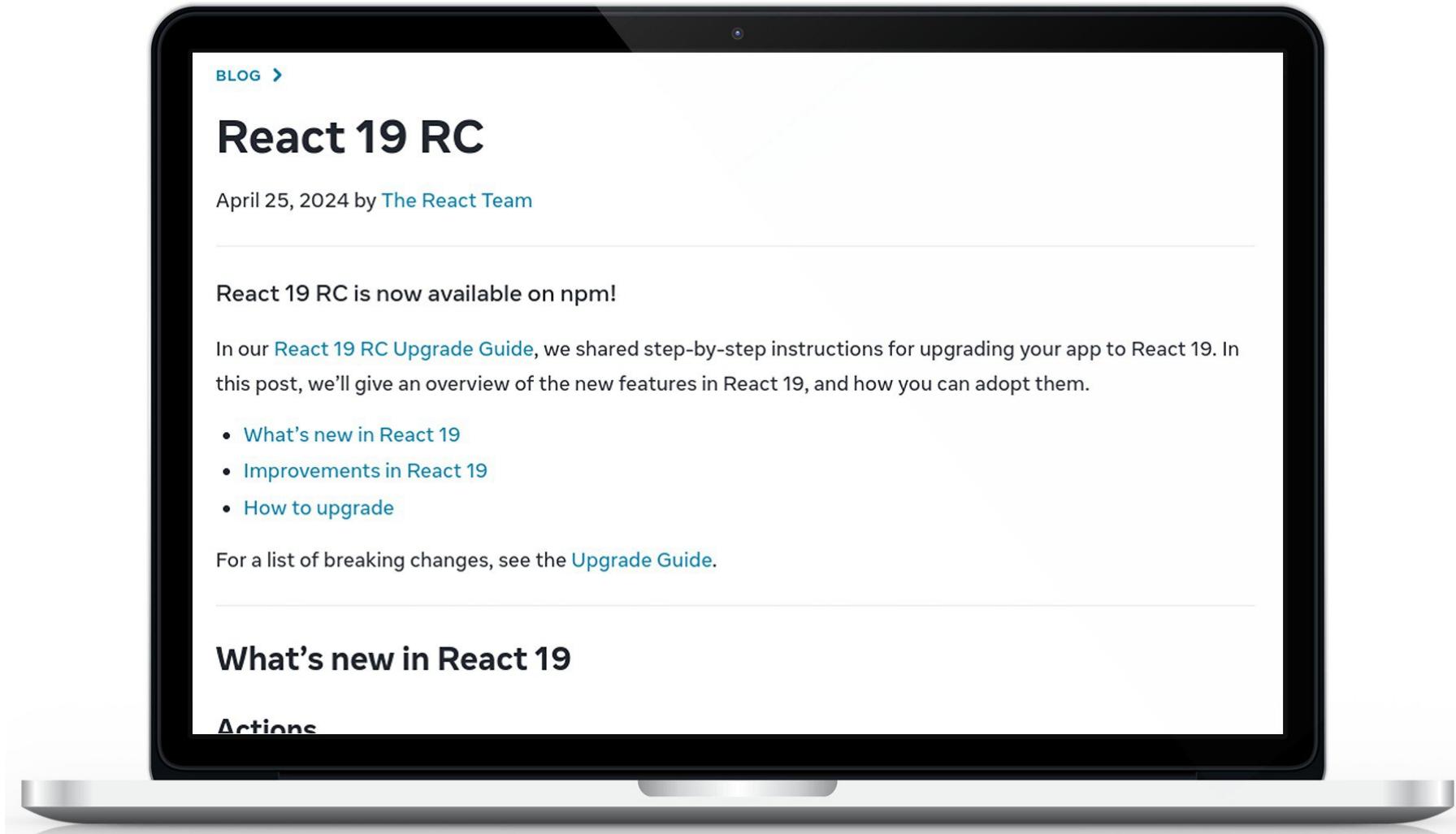
3 Runtime

- `document.cookie;`
- `(res | req).cookies;`

3 Runtime

- `document.cookie;`
- `(res | req).cookies;`
- `cookies()` .

Официально React 18



Официально React 18

React 19 RC

April 25, 2024

Компоненты из UI Kit только
клиентские





high power tools for HTML

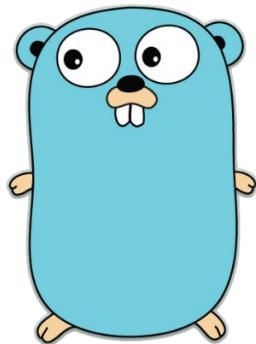
</>htmX

high power tools for HTML



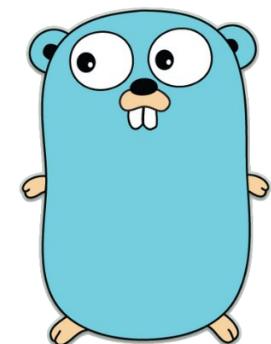
</>htmX

high power tools for HTML



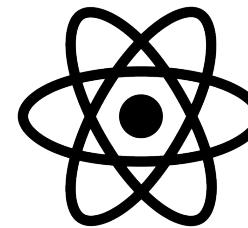
</>htmX

high power tools for HTML





high power tools for HTML

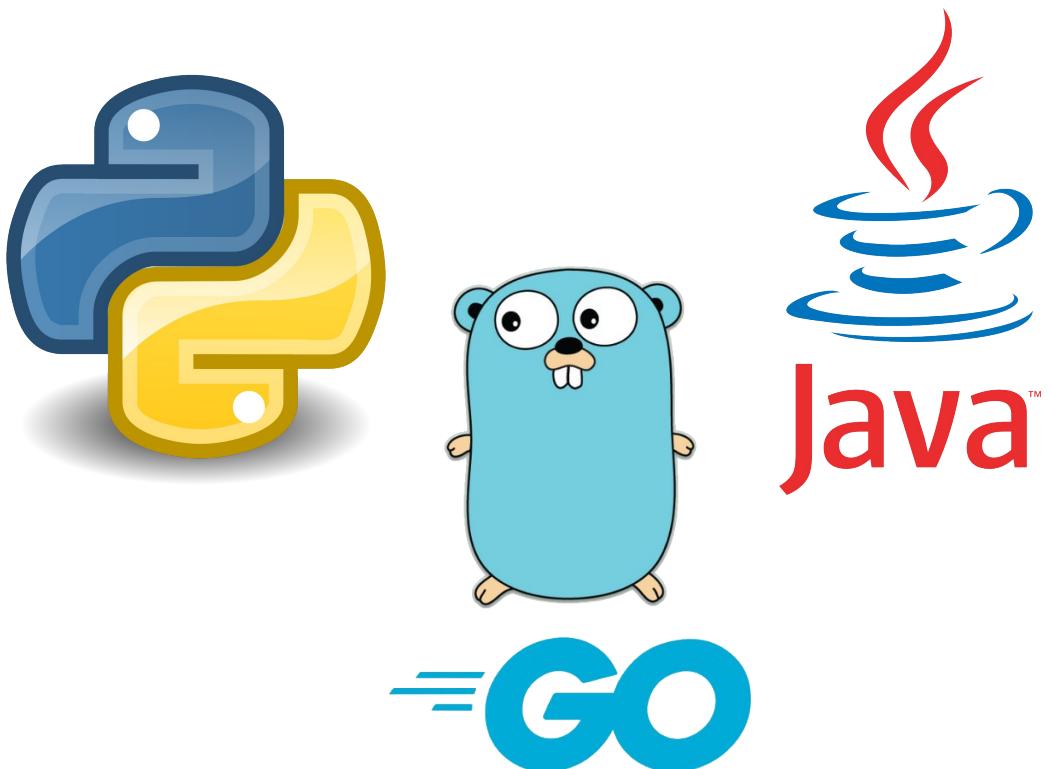


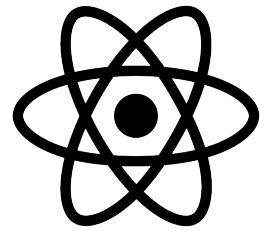
React





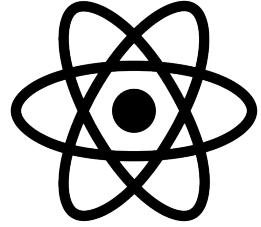
high power tools for HTML





React

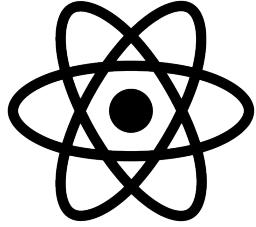




React

NEXT.JS



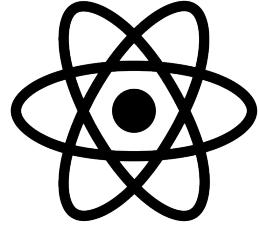


React

NEXT.JS

Remix





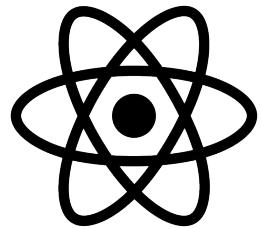
React

NEXT.JS

Remix



waku



React

NEXT.JS

Remix



waku

Redwoodjs



Возвращение Fullstack-а



Что сегодня обсудили

- Построили приложение на Next.js;



Что сегодня обсудили

- Построили приложение на Next.js;
- Обсудили сложности в работе с Next.js;



Что сегодня обсудили

- Построили приложение на Next.js;
- Обсудили сложности в работе с Next.js;
- Предположили куда будет двигаться индустрия.





Материалы

