Next.js.
Как ты вообще рендеришь?

Тёма Сенюков

- Делаю интерфейсы в Кинопоиске
- Читаю лекции в ШРИ
- Пишу в @temaProg

ТГ-канал

Тёма Сенюков

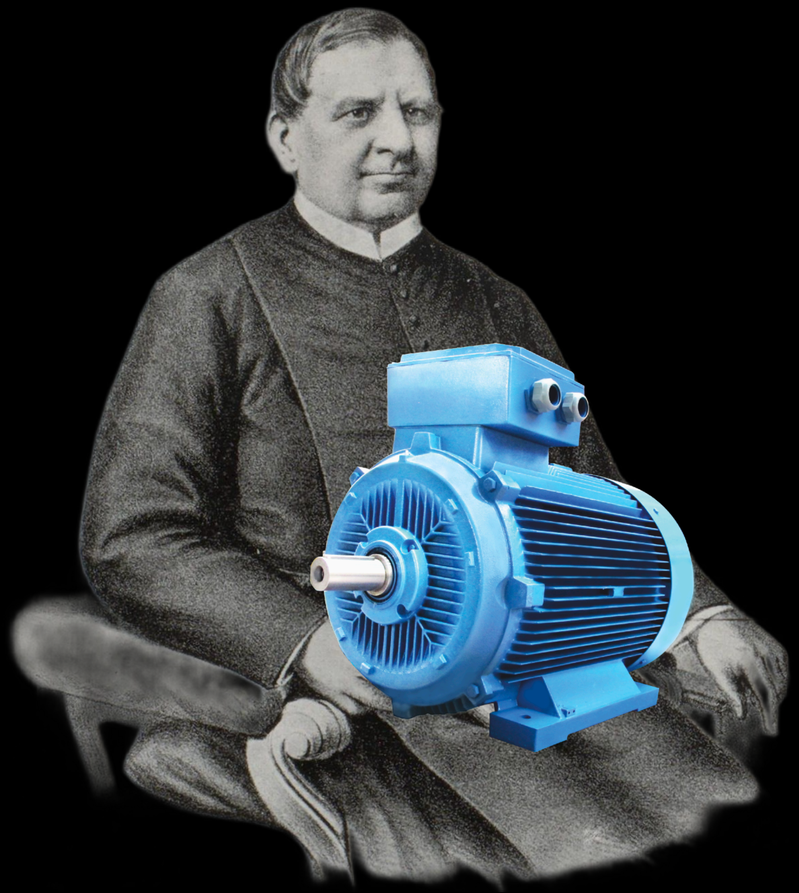
- Делаю интерфейсы в Кинопоиске
- Читаю лекции в ШРИ
- Пишу в @temaProg

Next.js.
Как ты вообще рендеришь?

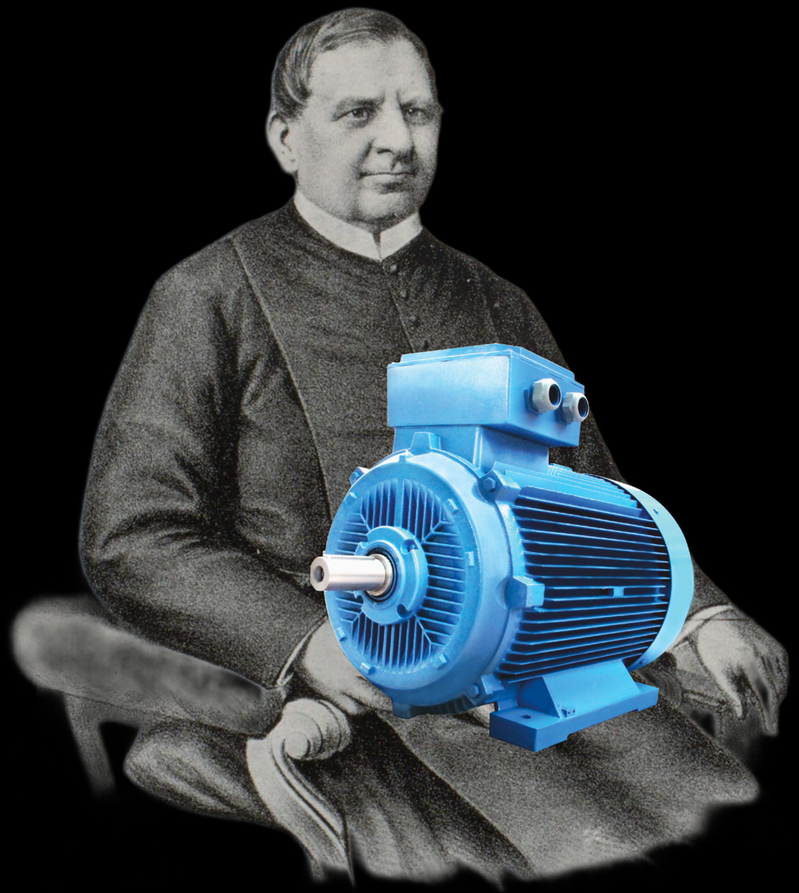
Аньош Йедлик

Аньош Йедлик - 1828

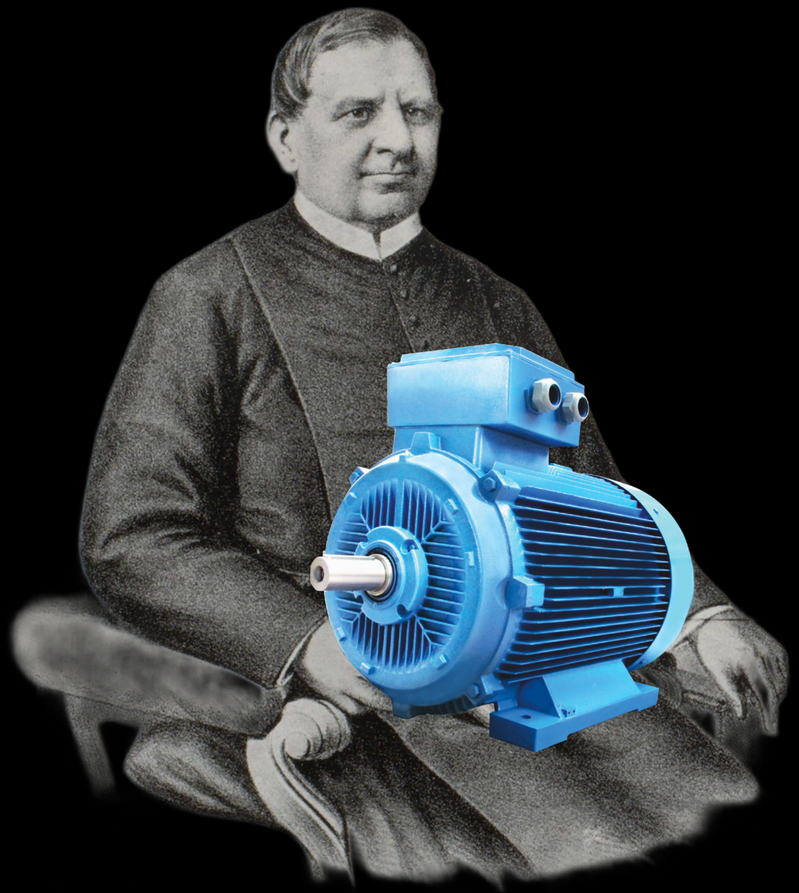
Аньош Йедлик - 1828



Аньош Йедлик - 1828



Аньош Йедлик - 1828

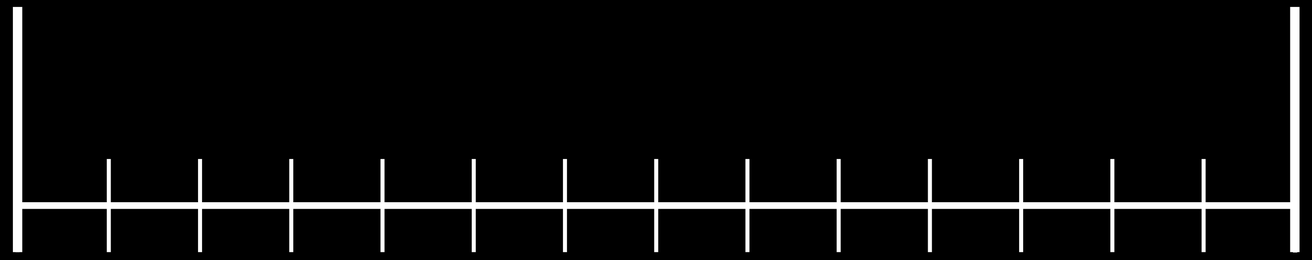


Аньош Йедлик - 1828

NextJs появился

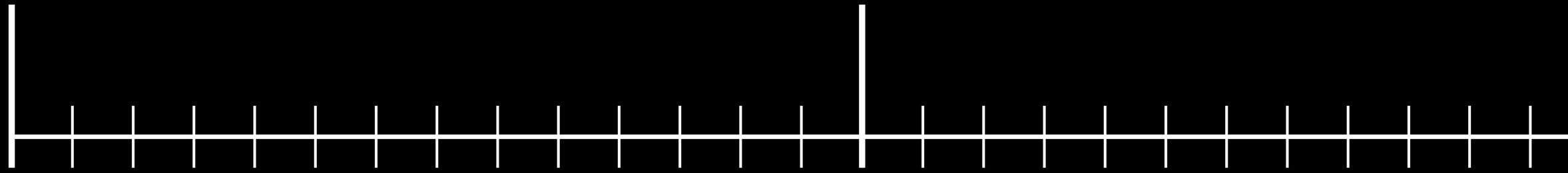
NextJs появился

Появился App Router (13.4)



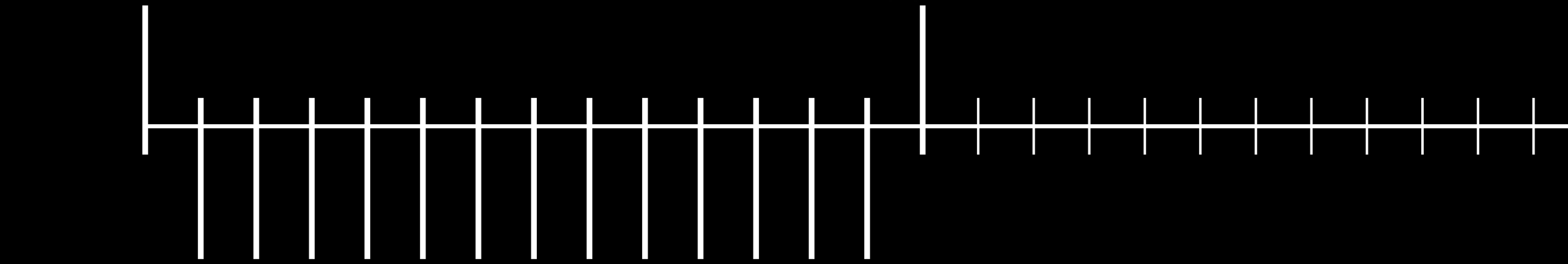
NextJs появился

Появился App Router (13.4)



NextJs появился

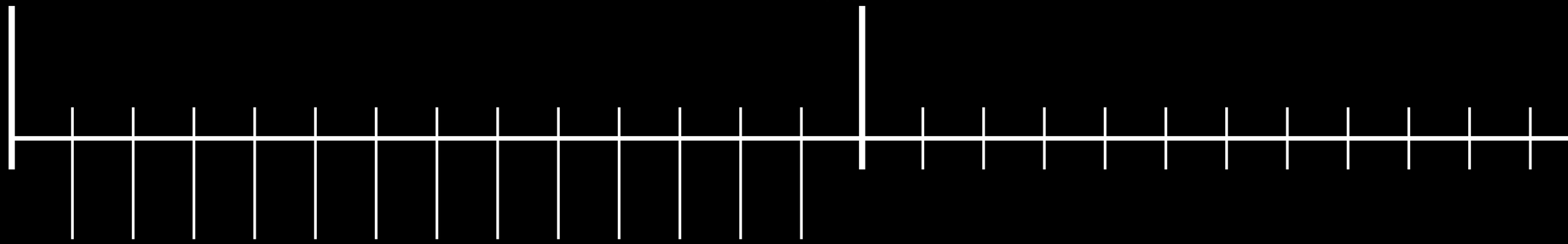
Появился App Router (13.4)



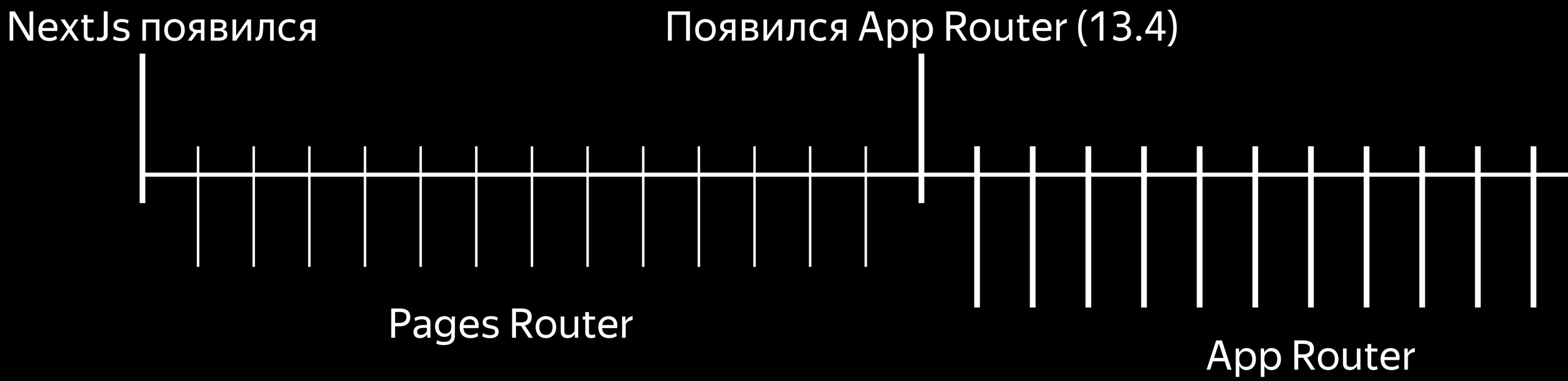
Pages Router

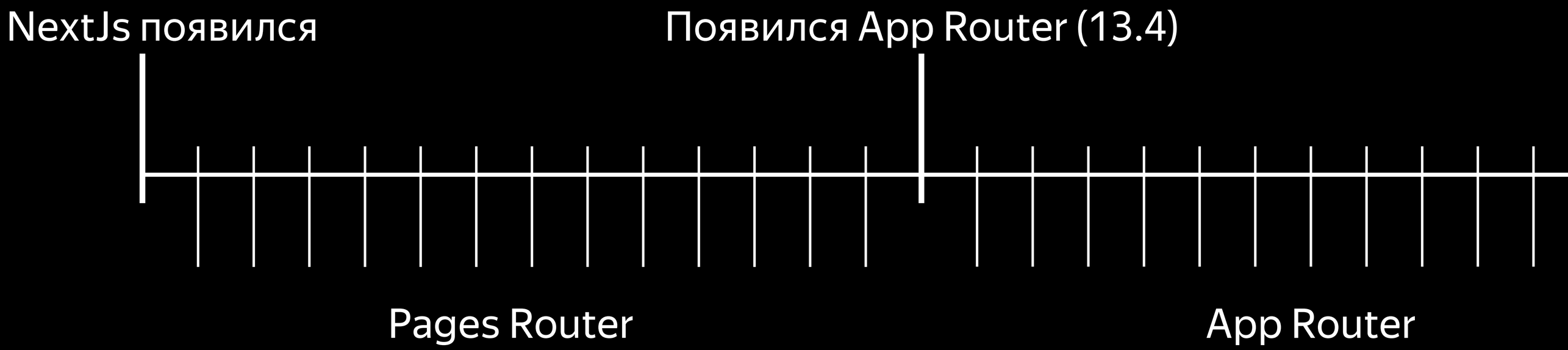
NextJs появился

Появился App Router (13.4)



Pages Router


NextJs появился
Появился App Router (13.4)
Pages Router
App Router


NextJs появился
Появился App Router (13.4)
Pages Router
App Router

Где?

Где? Среды рендеринга

Клиент

Где?

Где?

Клиент 

Сервер 

Где?

Клиент

Сервер

Как?

Где?

Клиент

Сервер

Как? CSR, SSG, SSR, ISR и тд

Next with Pages Router

Next with Pages Router

Pages Router

Клиент 

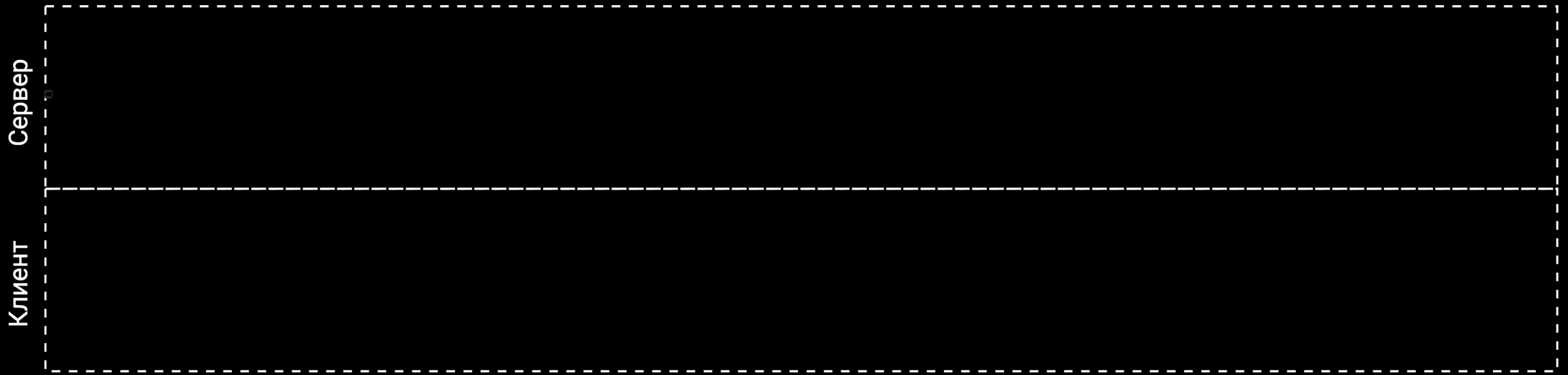
Сервер 

Pages Router

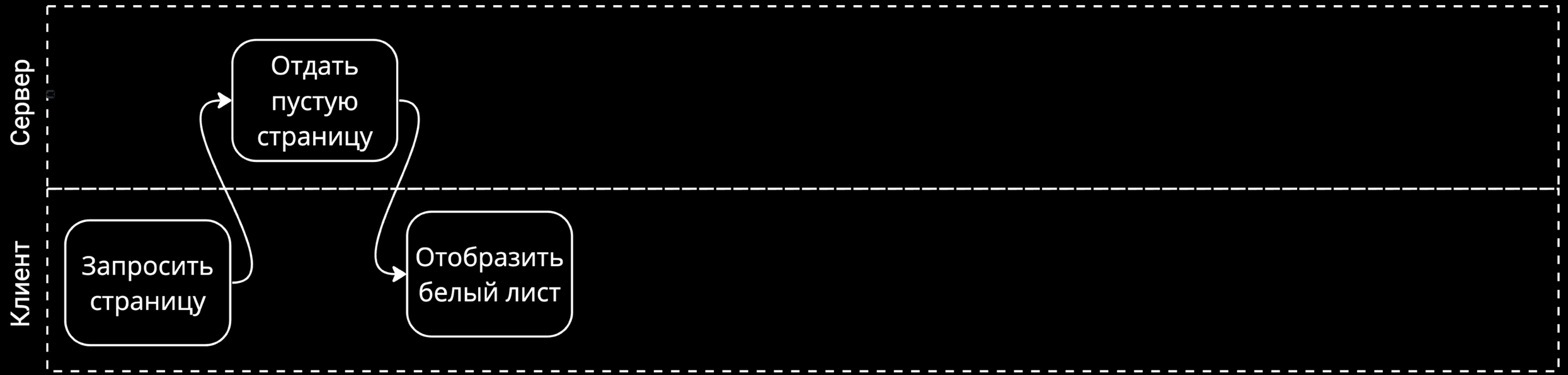
Клиент  CSR

Сервер 

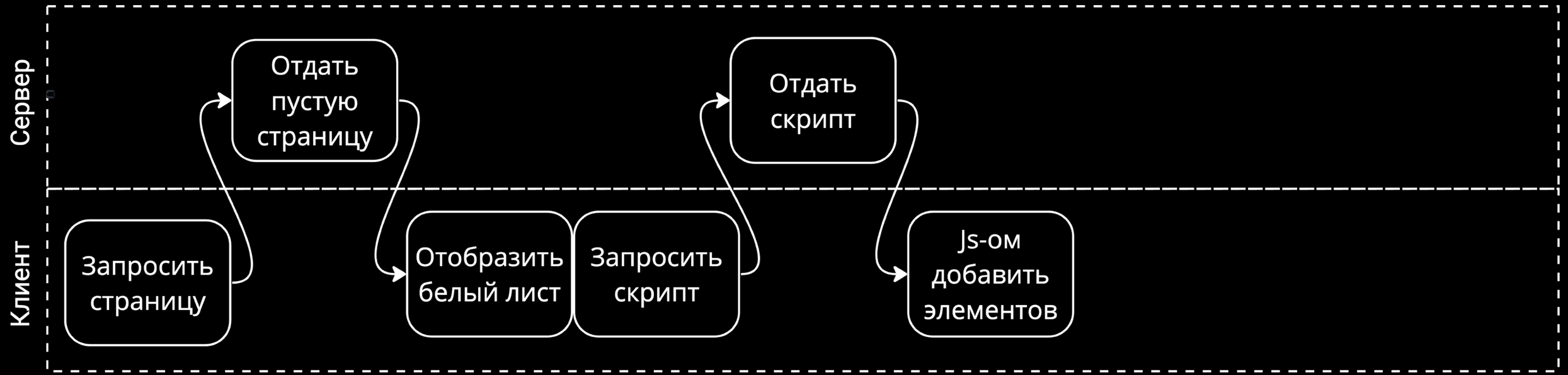
CSR



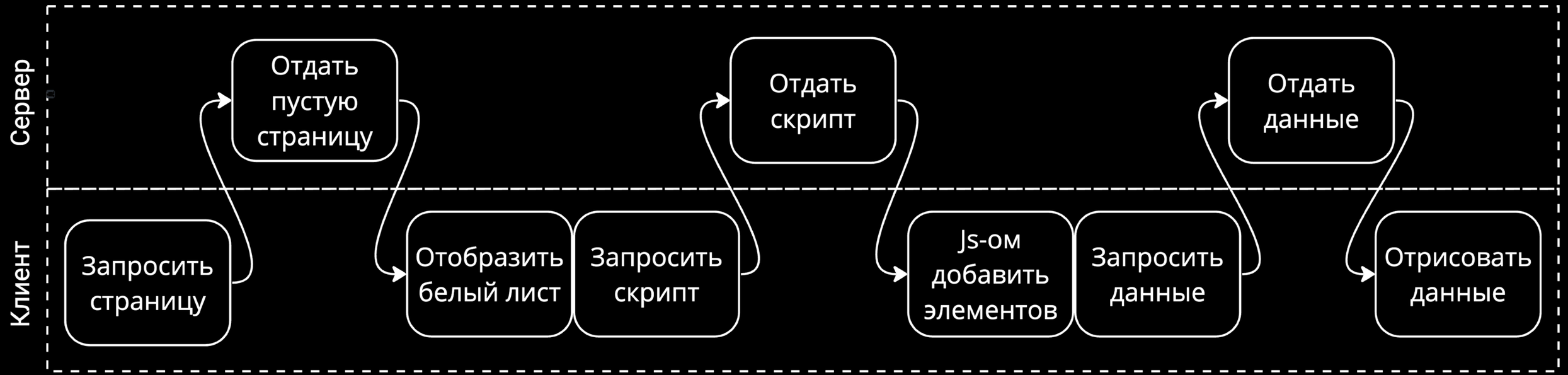
CSR


Сервер
Клиент
Отдать пустую страницу
Запросить страницу
Отобразить белый лист

CSR


Сервер
Клиент
Отдать пустую страницу
Отдать скрипт
Запросить страницу
Отобразить белый лист
Запросить скрипт
Js-ом добавить элементов

CSR


Сервер
Клиент
Отдать пустую страницу
Отдать скрипт
Отдать данные
Запросить страницу
Отобразить белый лист
Запросить скрипт
Js-ом добавить элементов
Запросить данные
Отрисовать данные

Плюсы

Дешево по ресурсам

Быстрые переходы между “страницами”

Плюсы

Дешево по ресурсам

Быстрые переходы между “страницами”

Минусы

Нагрузка на устройство пользователя

Медленный первый старт

Плохое (никакое) SEO

Pages Router

Клиент  CSR

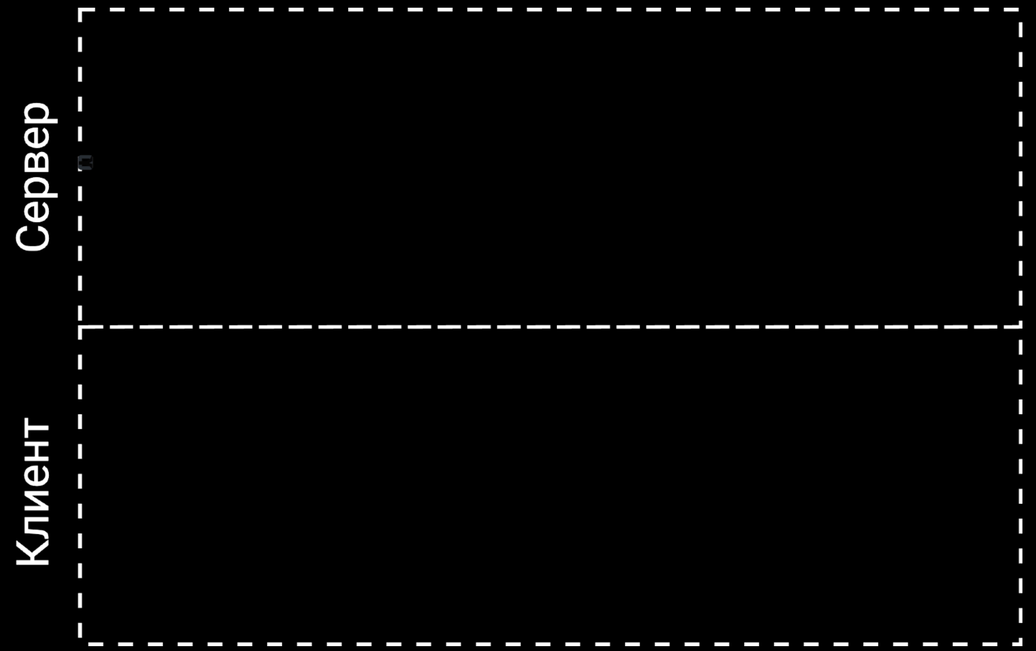
Сервер 

Pages Router

Клиент  CSR

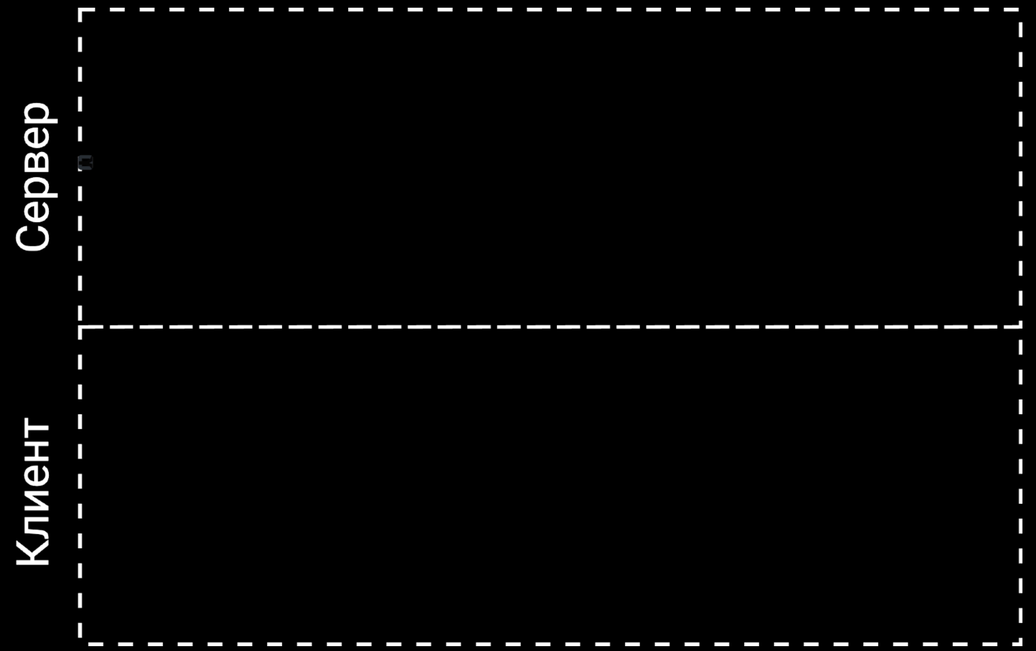
Сервер  SSG

SSG



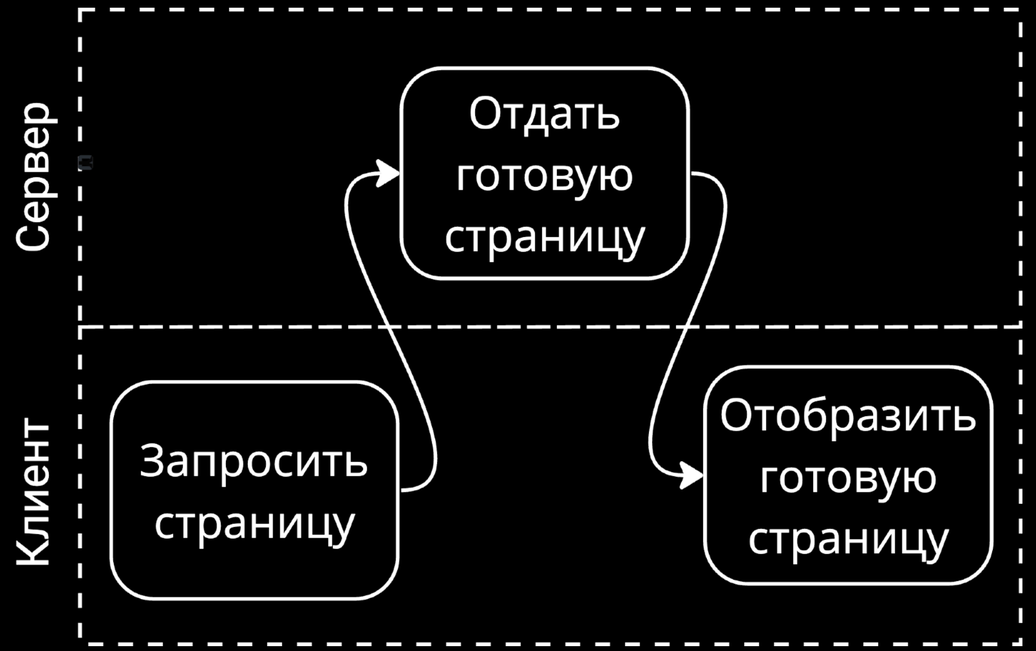
SSG

Генерация страниц во время билда



SSG

Генерация страниц во время билда



Плюсы

Дешево по ресурсам

Быстрое открытие страниц

SEO

Нетребователен к устройству клиента

Плюсы

Дешево по ресурсам

Быстрое открытие страниц

SEO

Нетребователен к устройству клиента

Минусы

Плохая масштабируемость

Pages Router

Клиент  CSR

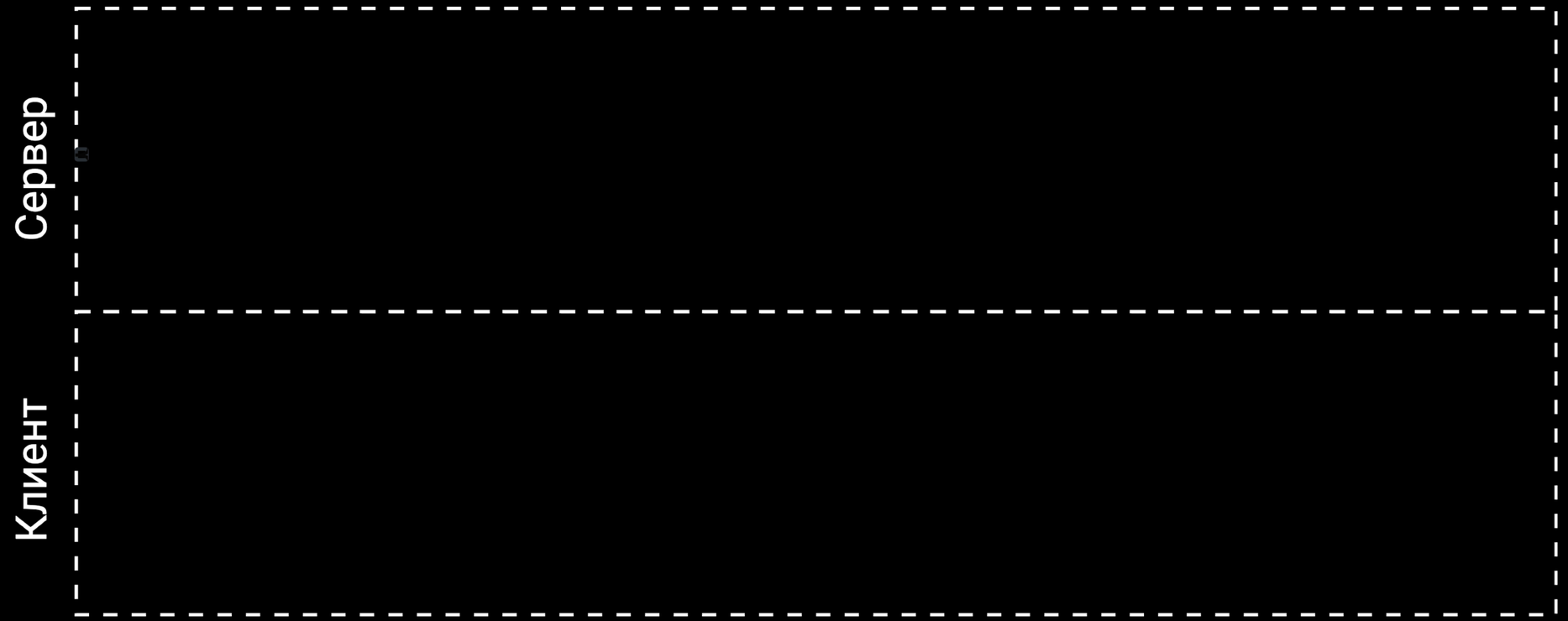
Сервер  SSG

Pages Router

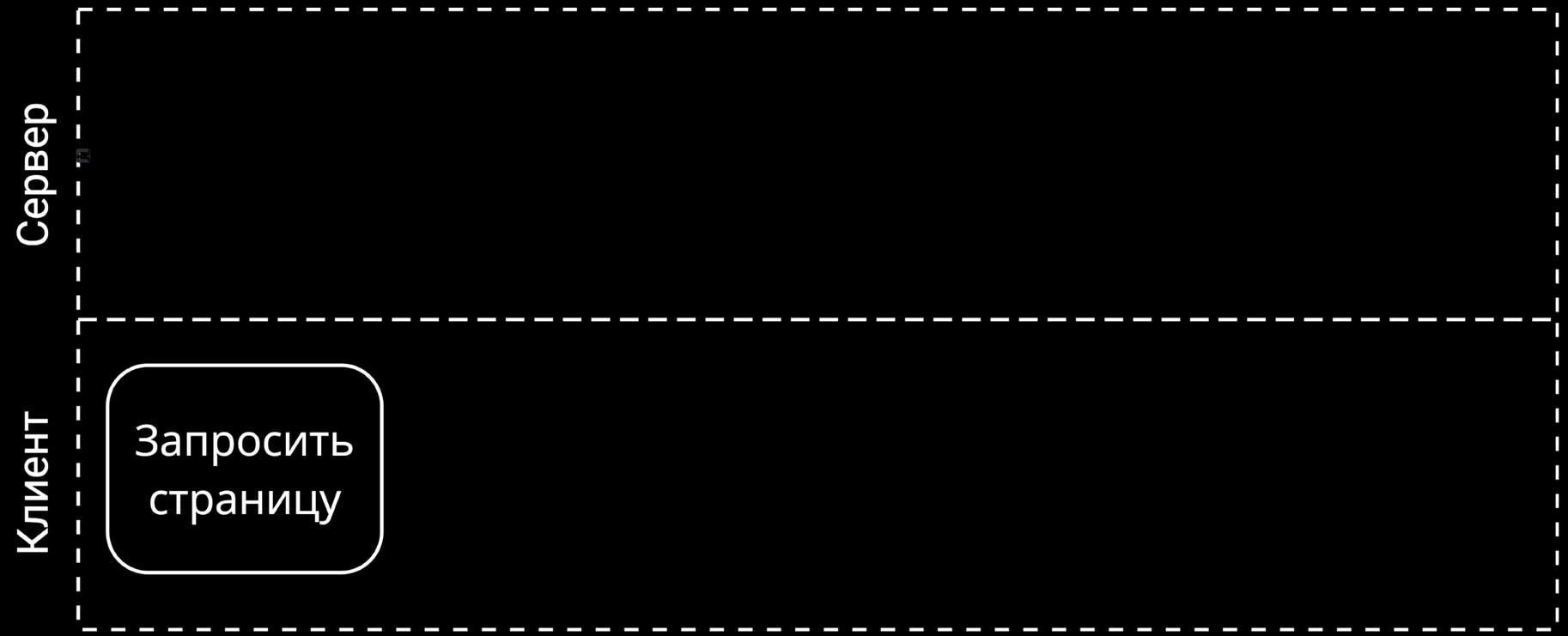
Клиент  CSR

Сервер  SSG SSR

SSR



SSR



SSR


Сервер
Подготовить данные
Отрендерить страницу с данными
Клиент
Запросить страницу

SSR


Сервер
Подготовить данные
Отрендерить страницу с данными
Клиент
Запросить страницу
Отобразить страницу

Плюсы

Быстрая загрузка

SEO

Нетребователен к устройству клиента

Плюсы

Быстрая загрузка

SEO

Нетребователен к устройству клиента

Минусы

Дорого по ресурсам

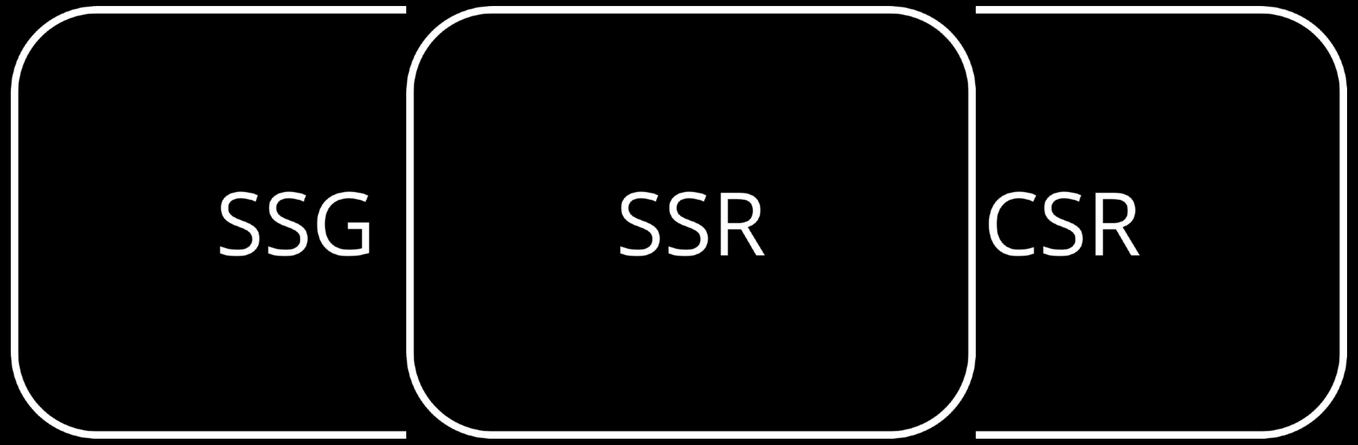
Гибридность

SSG

SSR

CSR

Гибридность



Гибридность

SSG

SSR

CSR

Гибридность

SSG

SSR

CSR

Старается использовать всегда.

Гибридность

SSG

Старается использовать всегда.

SSR

Если подготовить страницу на билде
нельзя или очень большая вариативность

CSR

Гибридность

SSG

Старается использовать всегда.

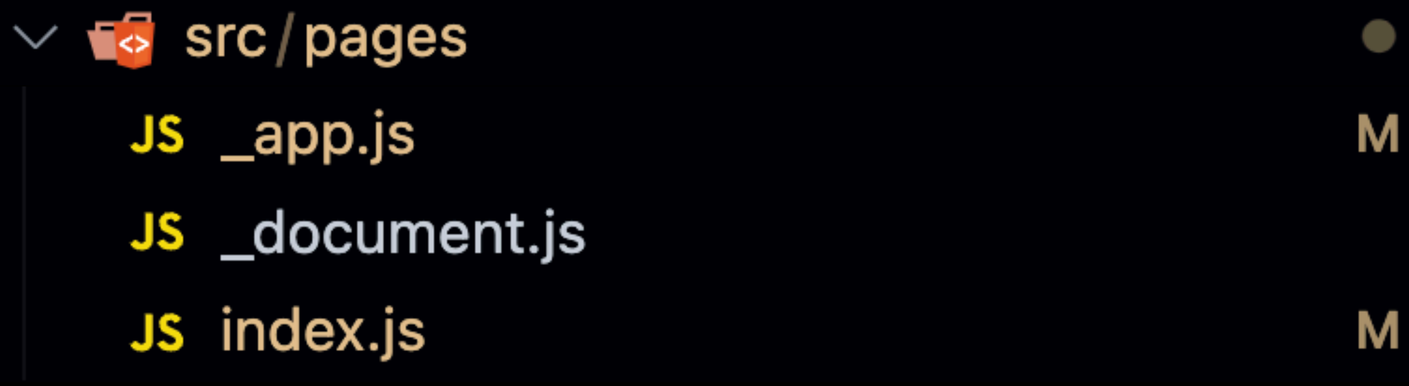
SSR

Если подготовить страницу на билде
нельзя или очень большая вариативность

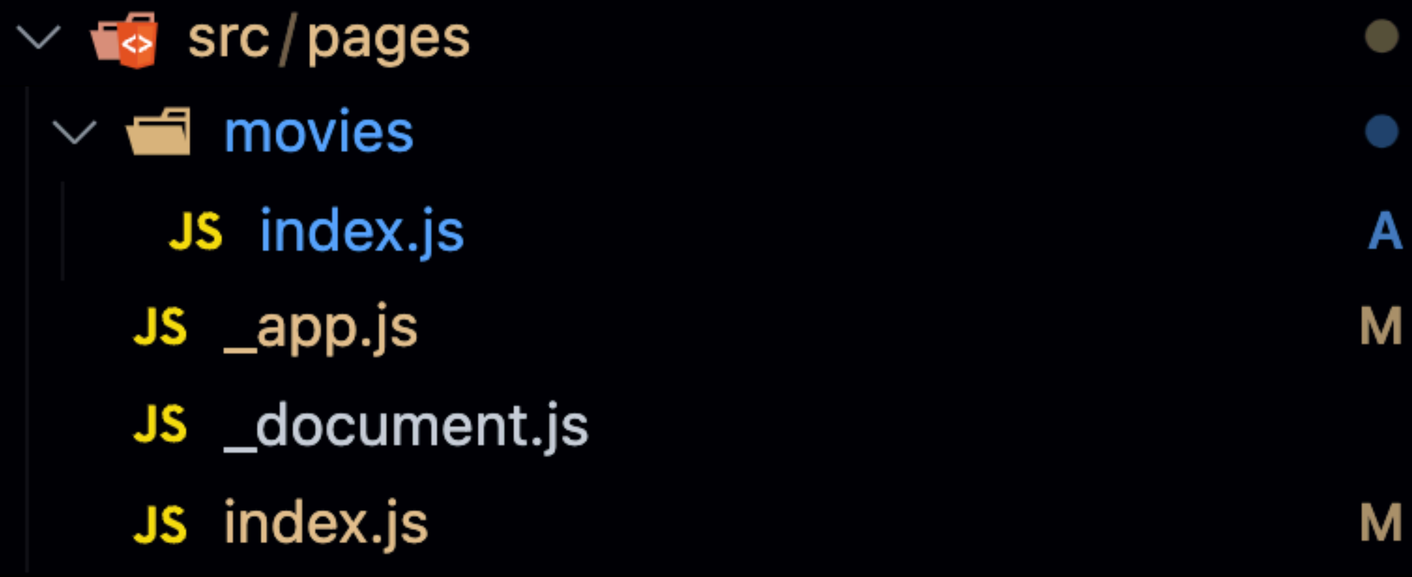
CSR

Если первая страница уже загружена на
клиент, то последующие “переходы”
можно осуществить на клиенте

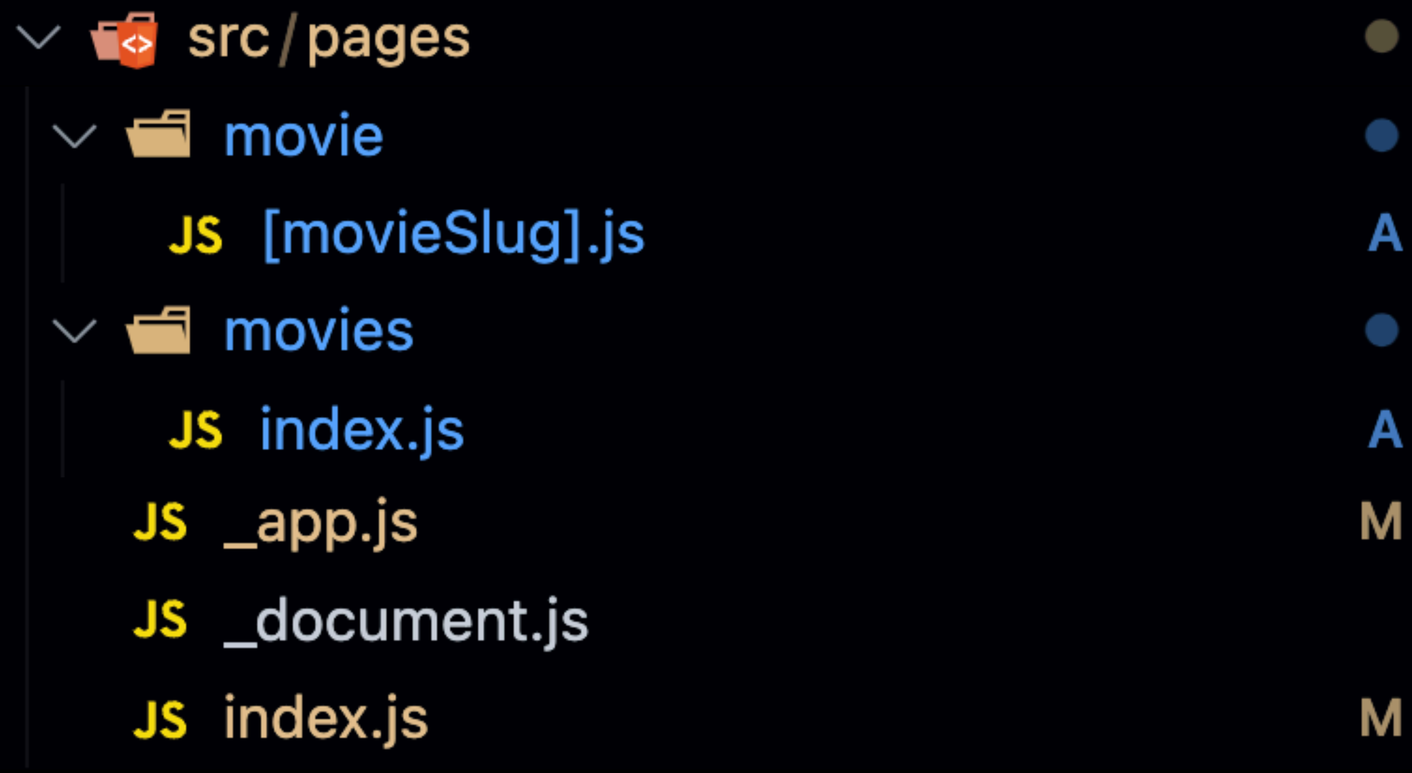
Гибридность



Гибридность



Гибридность



Гибридность

```
Route (pages)                              Size     First Load JS
┌ ○ /                                      272 B          78.5 kB
├   /_app                                  0 B            78.2 kB
├ ○ /404                                   181 B          78.4 kB
├ ƒ /movie/[movieSlug]                     330 B          78.6 kB
└ ○ /movies                                262 B          78.5 kB
+ First Load JS shared by all              78.2 kB
  ├ chunks/framework-5eea1a21a68cb00b.js   45.2 kB
  ├ chunks/main-fadf91780960d9cf.js        32 kB
  └ other shared chunks (total)            1.04 kB

○  (Static)   prerendered as static content
ƒ  (Dynamic)  server-rendered on demand
```

Гибридность

```
Route (pages)                              Size     First Load JS
┌ ○ /                                      272 B          78.5 kB
├   /_app                                  0 B            78.2 kB
├ ○ /404                                   181 B          78.4 kB
├ ƒ /movie/[movieSlug]                     330 B          78.6 kB
└ ○ /movies                                262 B          78.5 kB
+ First Load JS shared by all              78.2 kB
  ├ chunks/framework-5eea1a21a68cb00b.js   45.2 kB
  ├ chunks/main-fadf91780960d9cf.js        32 kB
  └ other shared chunks (total)            1.04 kB

○  (Static)   prerendered as static content
ƒ  (Dynamic)  server-rendered on demand
```

Гибридность

```
Route (pages)                              Size     First Load JS
┌ ○ /                                      272 B          78.5 kB
├   /_app                                  0 B            78.2 kB
├ ○ /404                                   181 B          78.4 kB
├ ƒ /movie/[movieSlug]                     330 B          78.6 kB
└ ○ /movies                                262 B          78.5 kB
+ First Load JS shared by all              78.2 kB
  ├ chunks/framework-5eea1a21a68cb00b.js   45.2 kB
  ├ chunks/main-fadf91780960d9cf.js        32 kB
  └ other shared chunks (total)            1.04 kB

○  (Static)   prerendered as static content
ƒ  (Dynamic)  server-rendered on demand
```

Гибридность

```
Route (pages)                              Size     First Load JS
┌ ○ /                                      272 B          78.5 kB
├   /_app                                  0 B            78.2 kB
├ ○ /404                                   181 B          78.4 kB
├ ƒ /movie/[movieSlug]                     330 B          78.6 kB
└ ○ /movies                                262 B          78.5 kB
+ First Load JS shared by all              78.2 kB
  ├ chunks/framework-5eea1a21a68cb00b.js   45.2 kB
  ├ chunks/main-fadf91780960d9cf.js        32 kB
  └ other shared chunks (total)            1.04 kB

○  (Static)   prerendered as static content
ƒ  (Dynamic)  server-rendered on demand
```

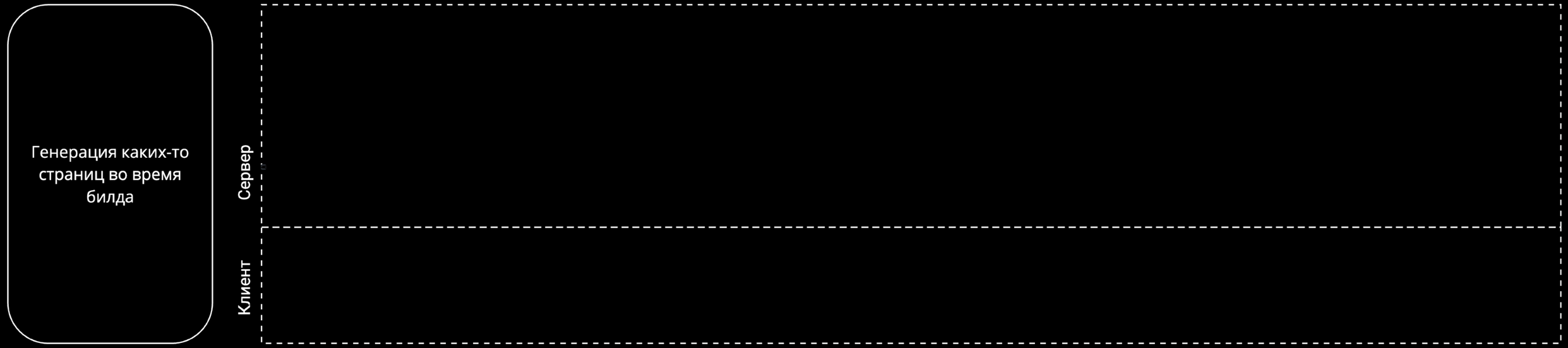
Гибридность

```
Route (pages)                                 Size     First Load JS
┌ ○ /                                         272 B          78.5 kB
├   /_app                                     0 B            78.2 kB
├ ○ /404                                     181 B          78.4 kB
├ ƒ /movie/[movieSlug]                       330 B          78.6 kB
└ ○ /movies                                  262 B          78.5 kB
+ First Load JS shared by all                78.2 kB
  ├ chunks/framework-5eea1a21a68cb00b.js     45.2 kB
  ├ chunks/main-fadf91780960d9cf.js          32 kB
  └ other shared chunks (total)              1.04 kB

○  (Static)   prerendered as static content
ƒ  (Dynamic)  server-rendered on demand
```

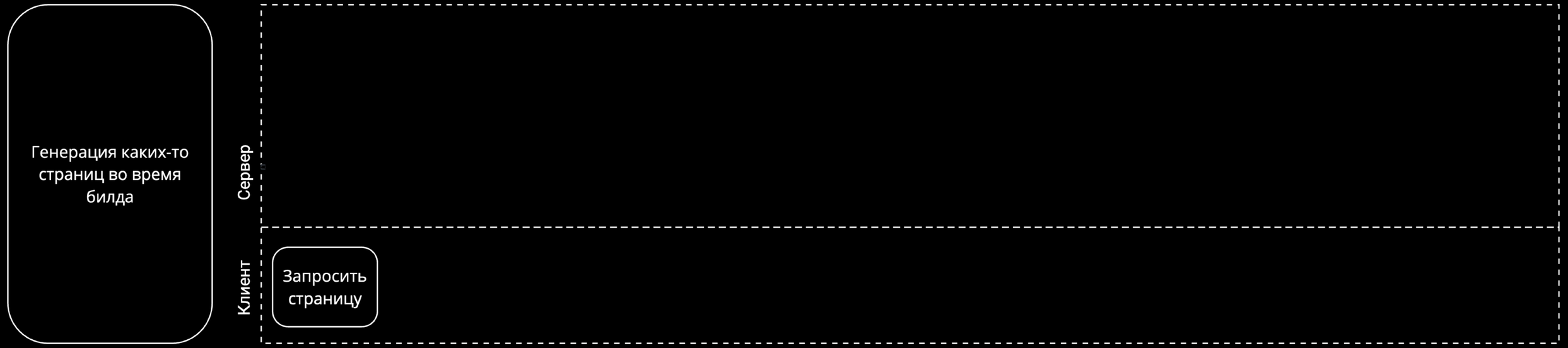
Гибридность

```
Route (pages)                              Size     First Load JS
┌ ○ /                                      272 B          78.5 kB
├   /_app                                  0 B            78.2 kB
├ ○ /404                                   181 B          78.4 kB
├ ƒ /movie/[movieSlug]                     330 B          78.6 kB
└ ○ /movies                                262 B          78.5 kB
+ First Load JS shared by all              78.2 kB
  ├ chunks/framework-5eea1a21a68cb00b.js   45.2 kB
  ├ chunks/main-fadf91780960d9cf.js        32 kB
  └ other shared chunks (total)            1.04 kB

○  (Static)   prerendered as static content
ƒ  (Dynamic)  server-rendered on demand
```

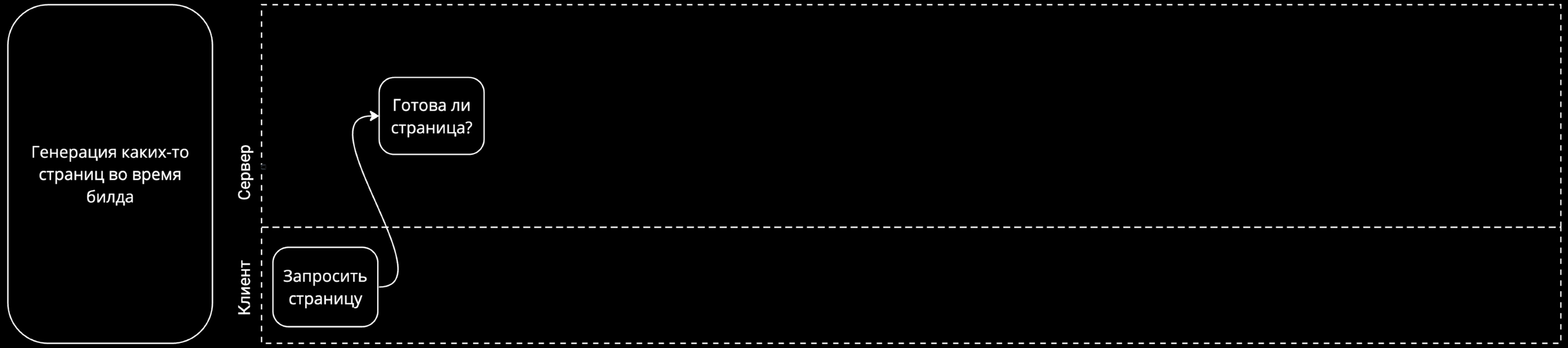
Гибридность



Гибридность



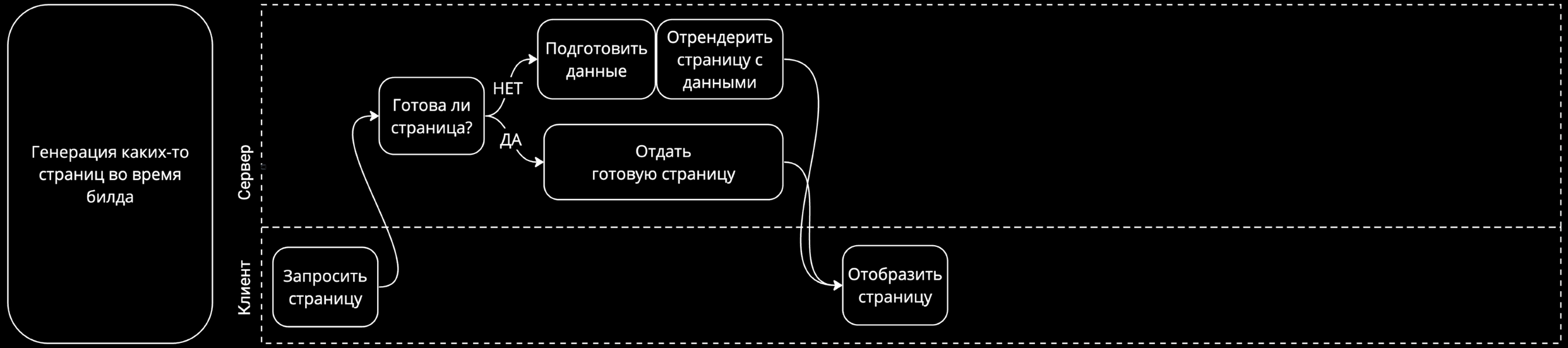
Гибридность



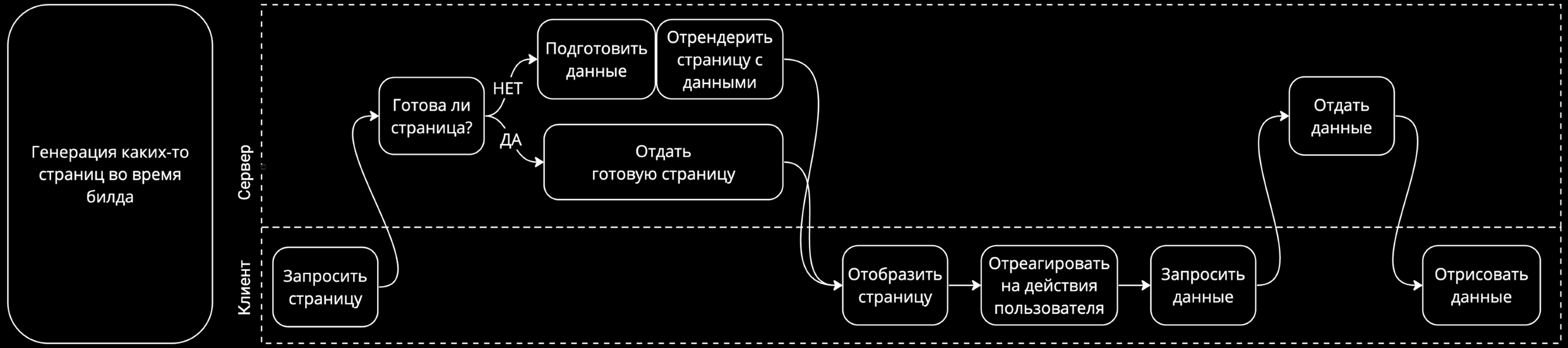
Гибридность

Гибридность

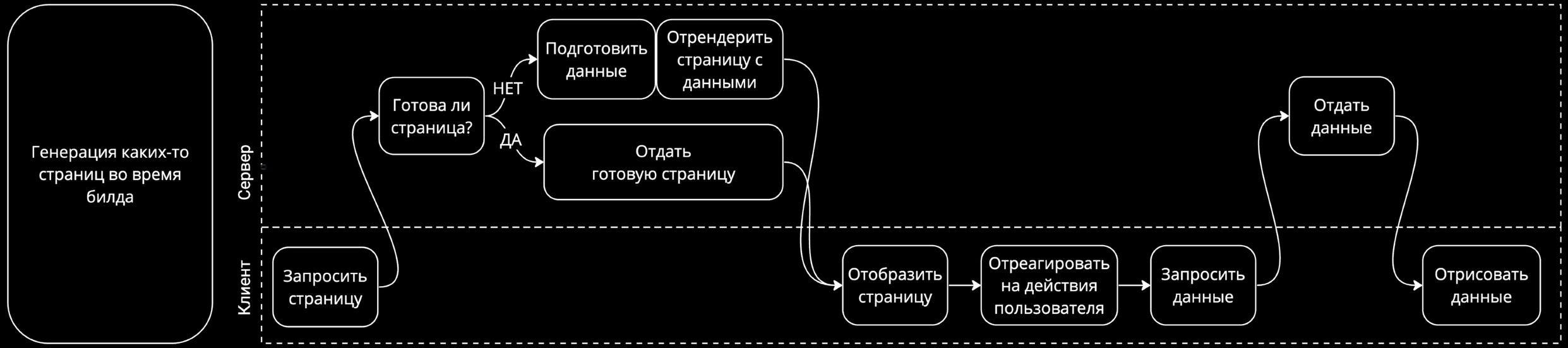
Гибридность



Гибридность



Гибридность

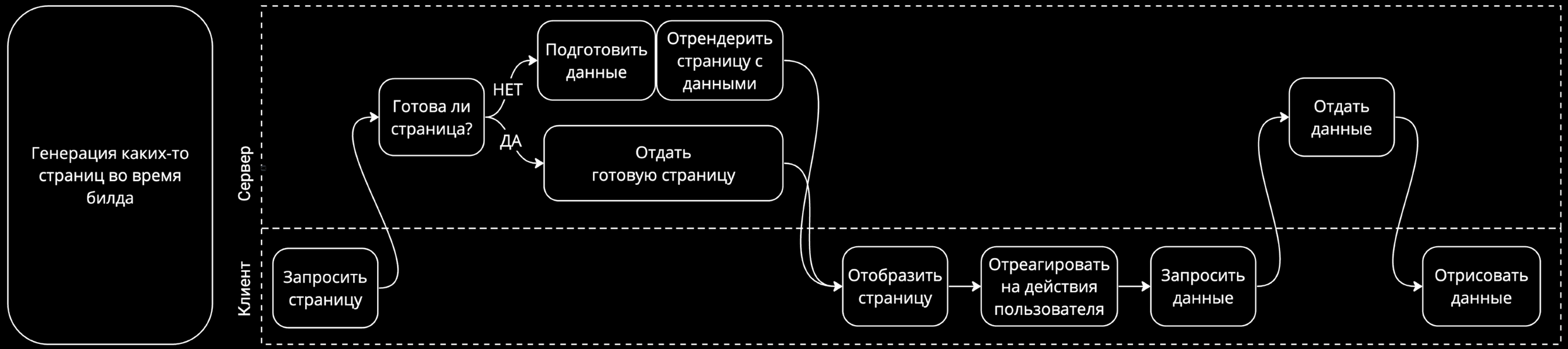


SSG

SSR

CSR

Гибридность

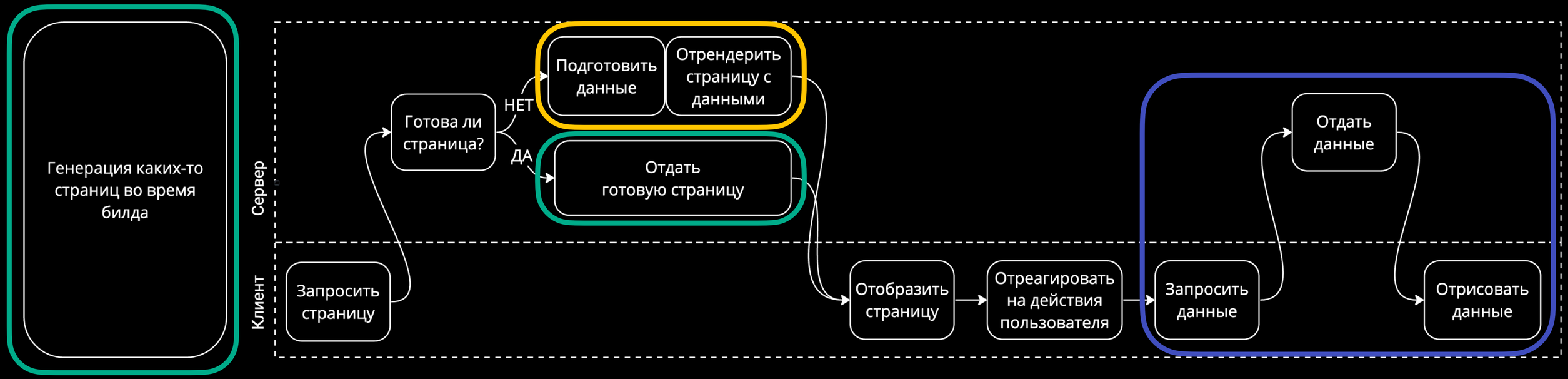


SSG

SSR

CSR

Гибридность



SSG

SSR

CSR

Какие есть проблемы?

Какие есть проблемы?

Монолитность

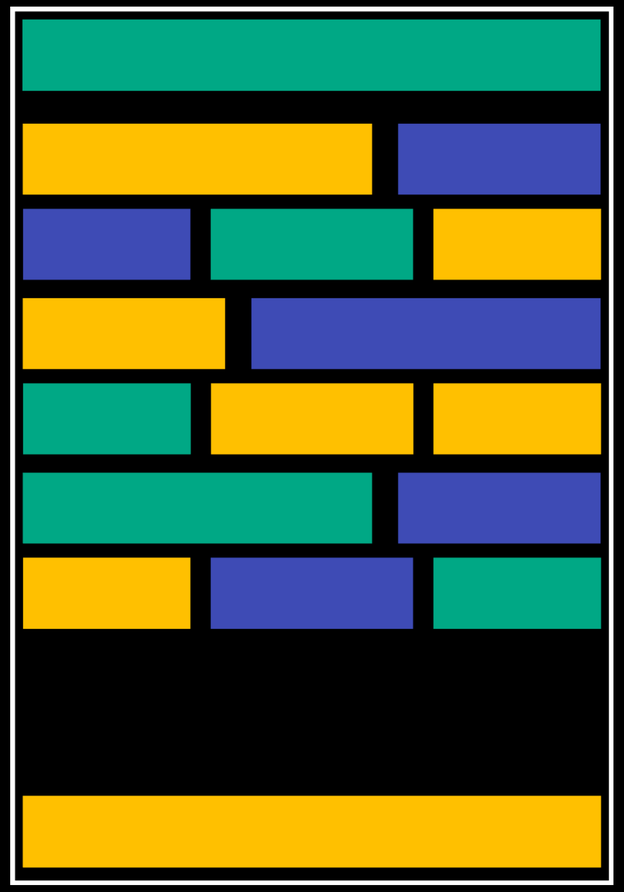
Монолитность

Server

Client

Монолитность

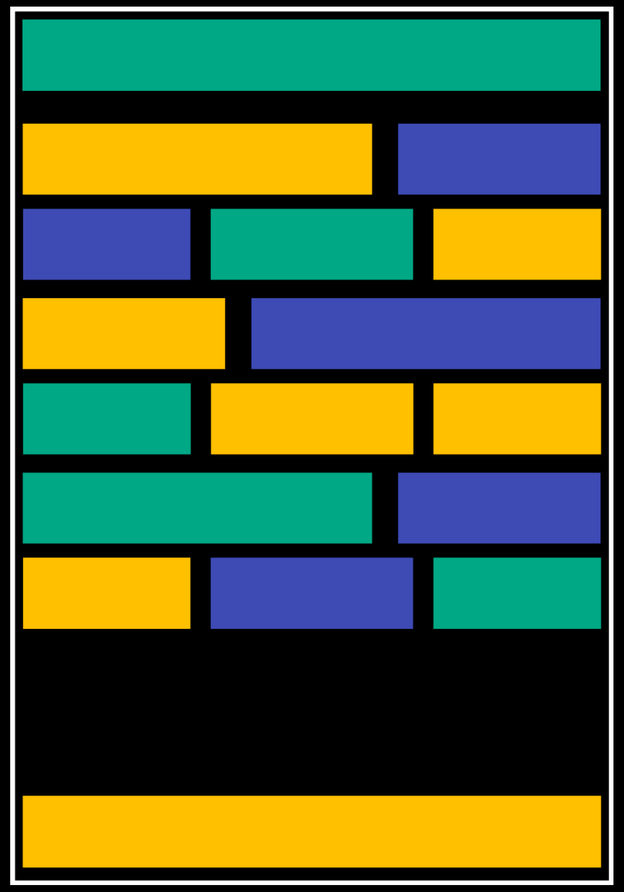
Server



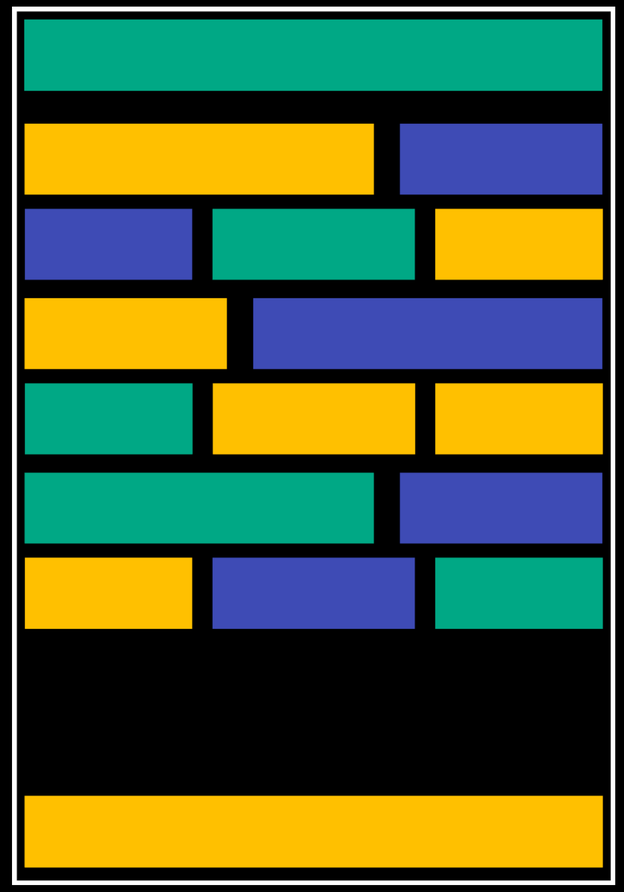
Client

Монолитность

Server



Client



Streaming c suspense

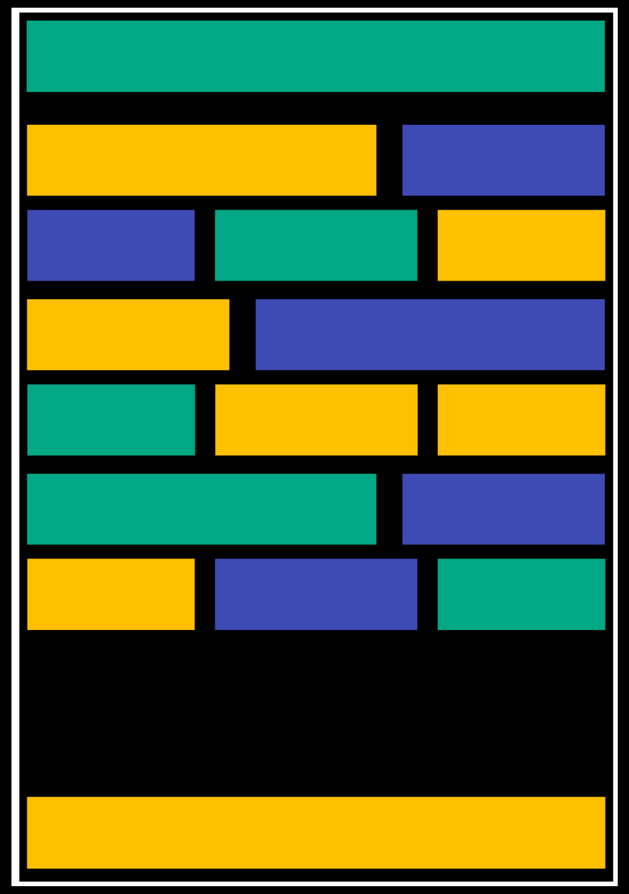
Streaming c suspense

Streaming с suspense

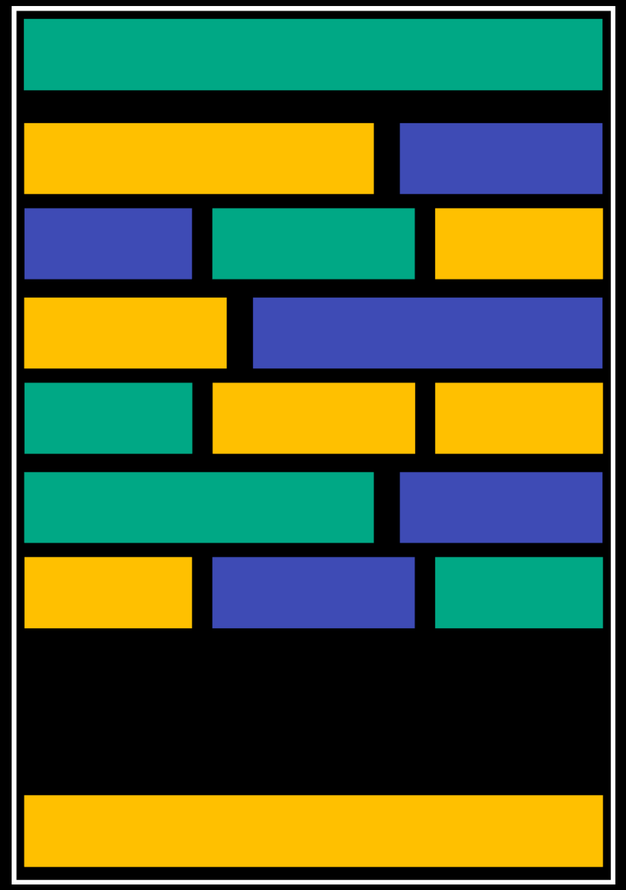
Потоковое вещание в Глобальной сети

Streaming c suspense

Server

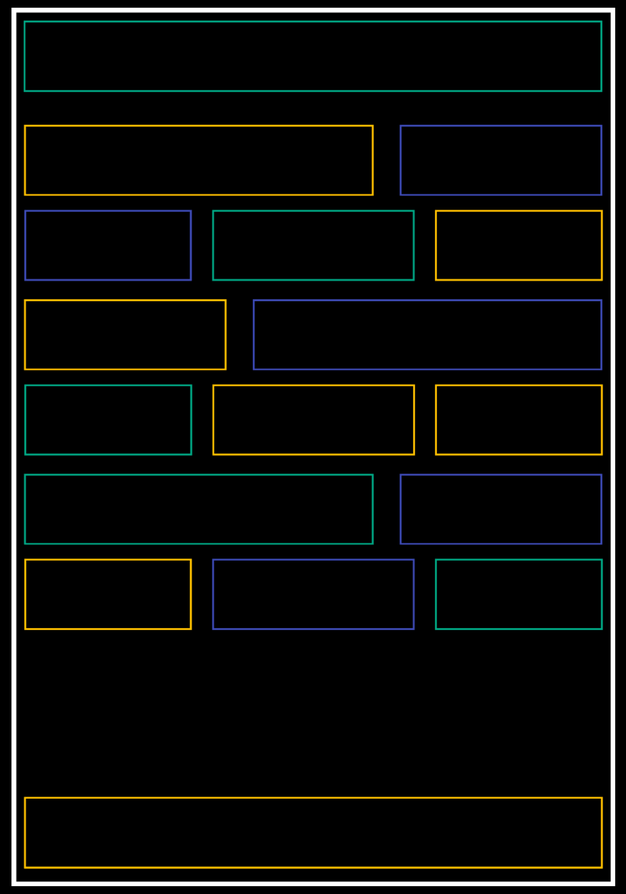


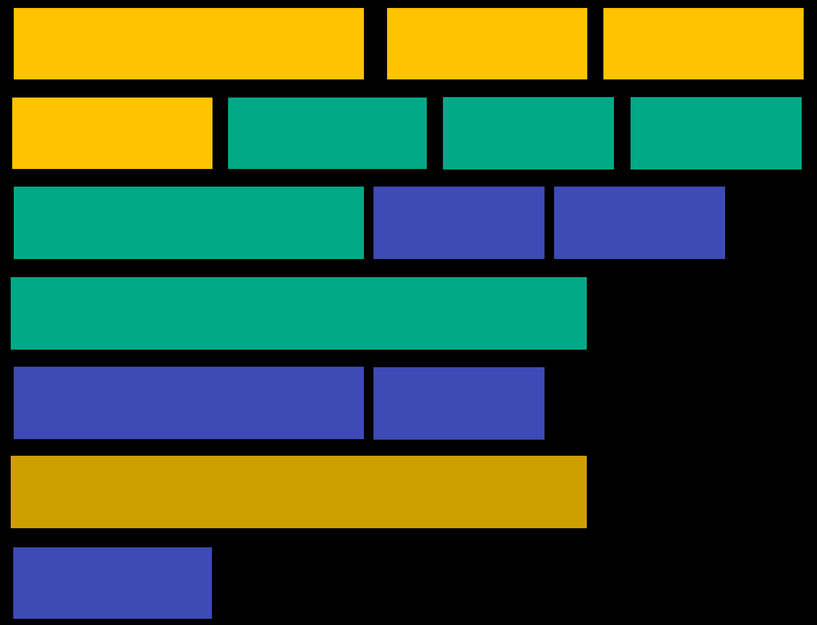
Client



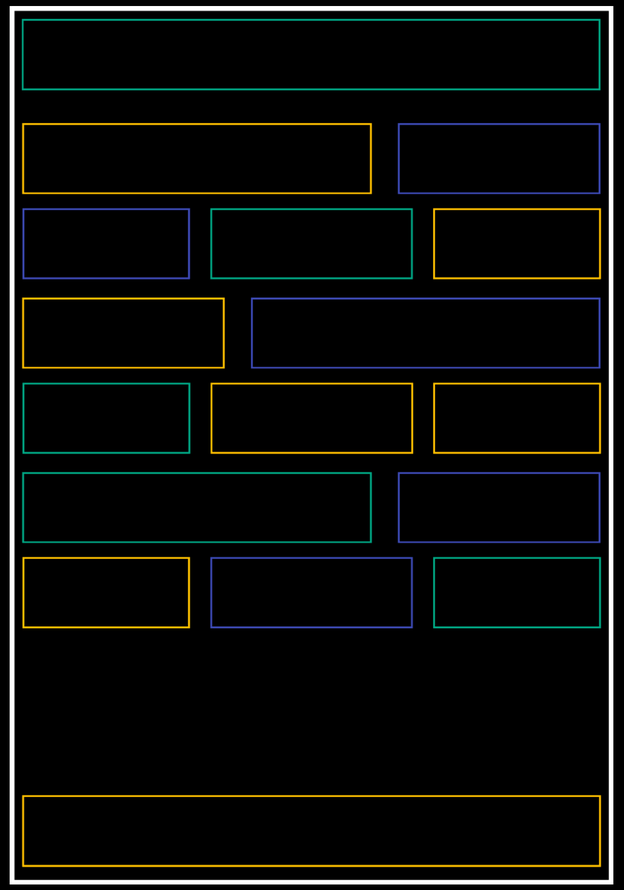
Streaming c suspense

Server



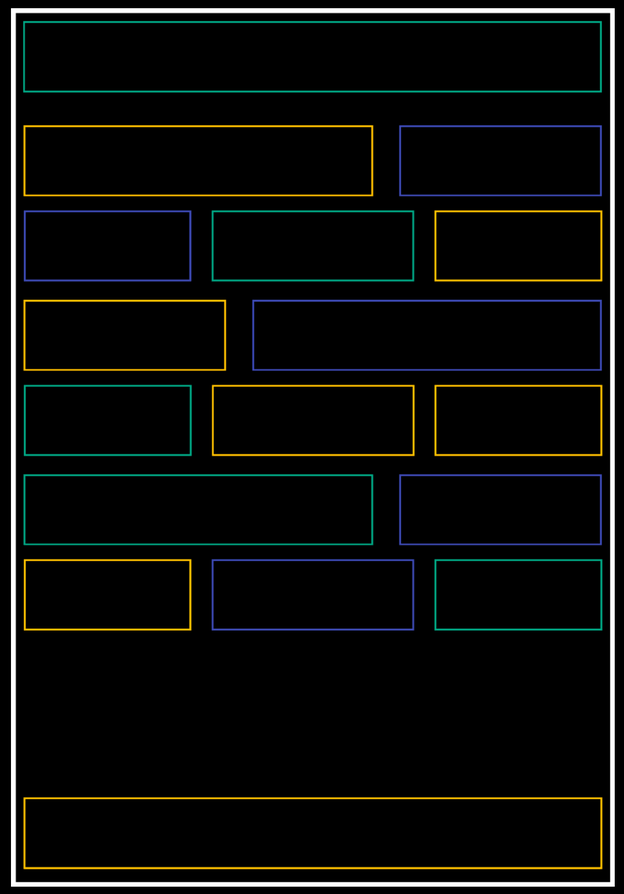


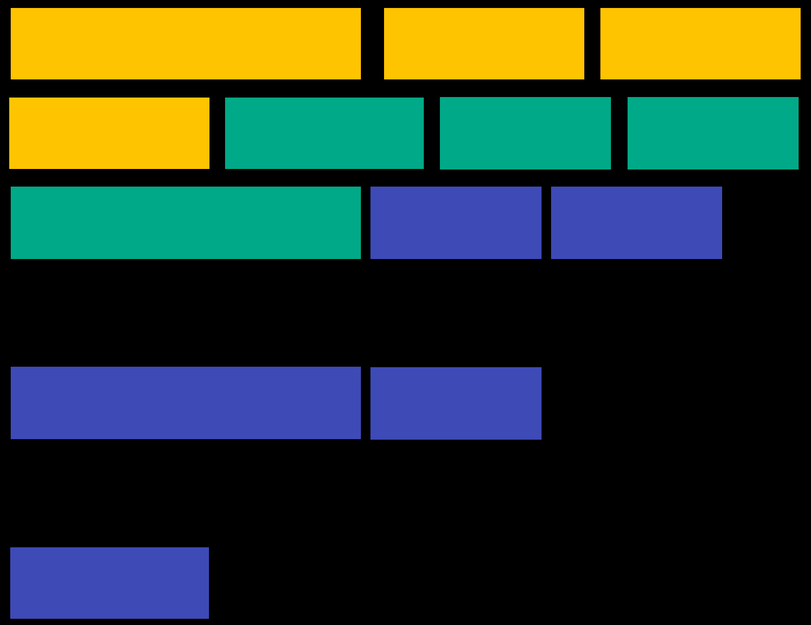
Client



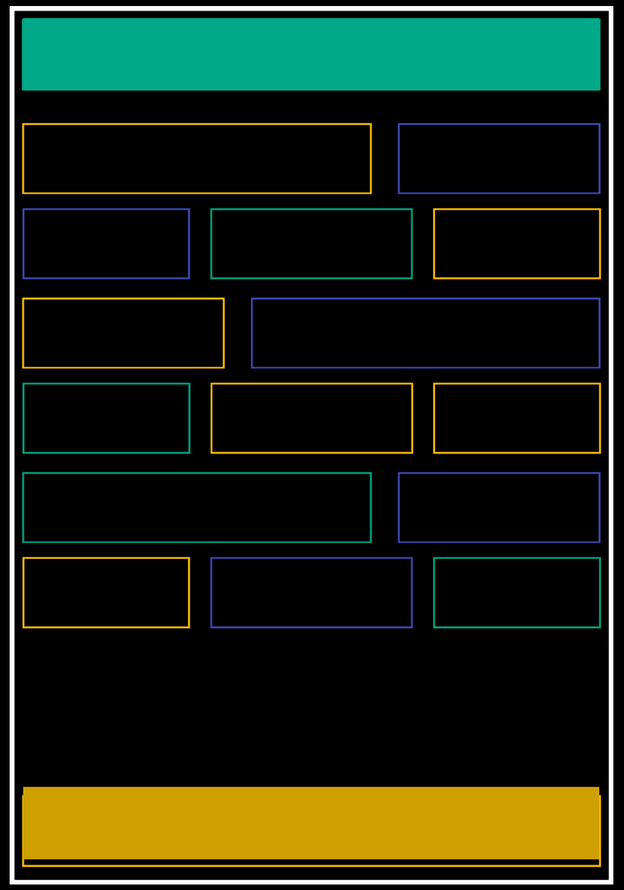
Streaming c suspense

Server



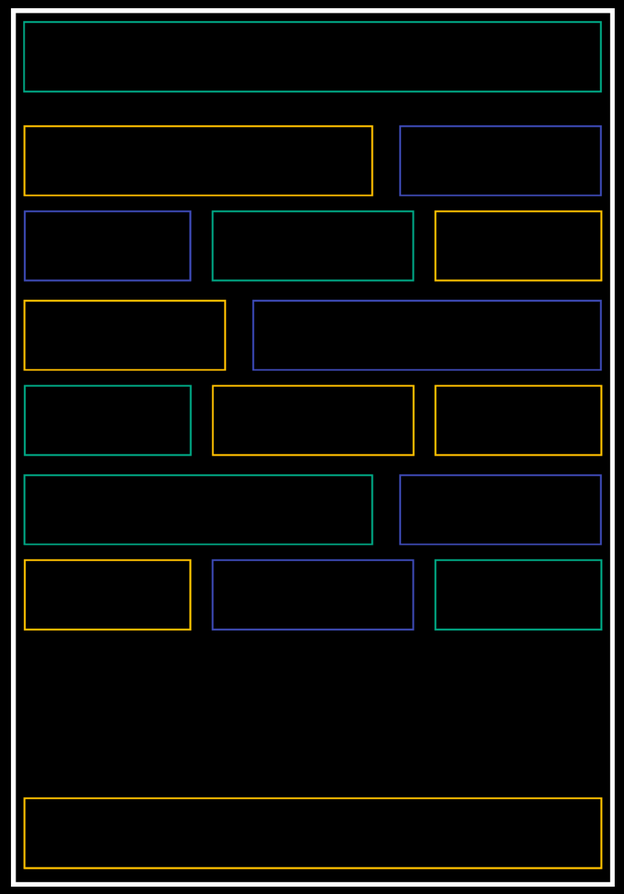


Client

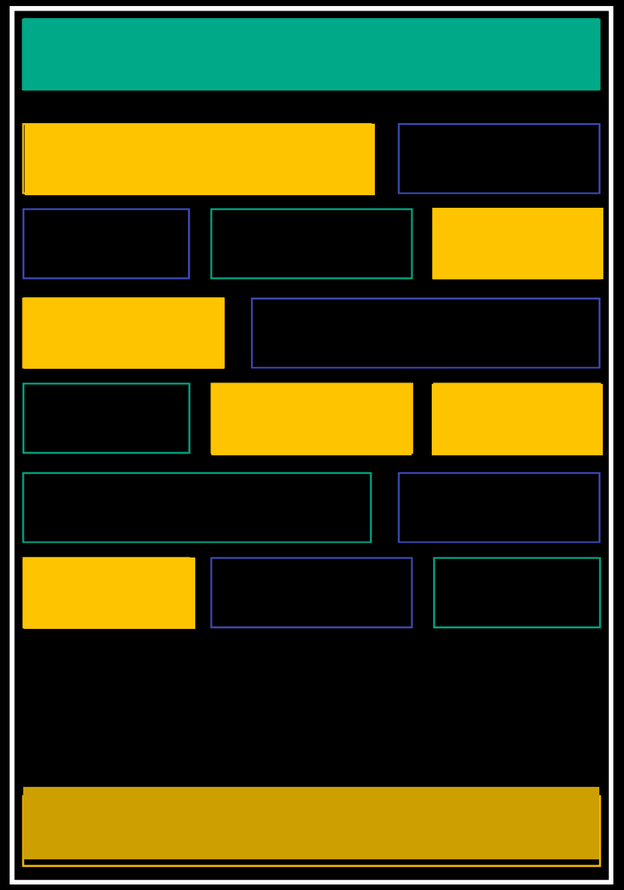


Streaming c suspense

Server

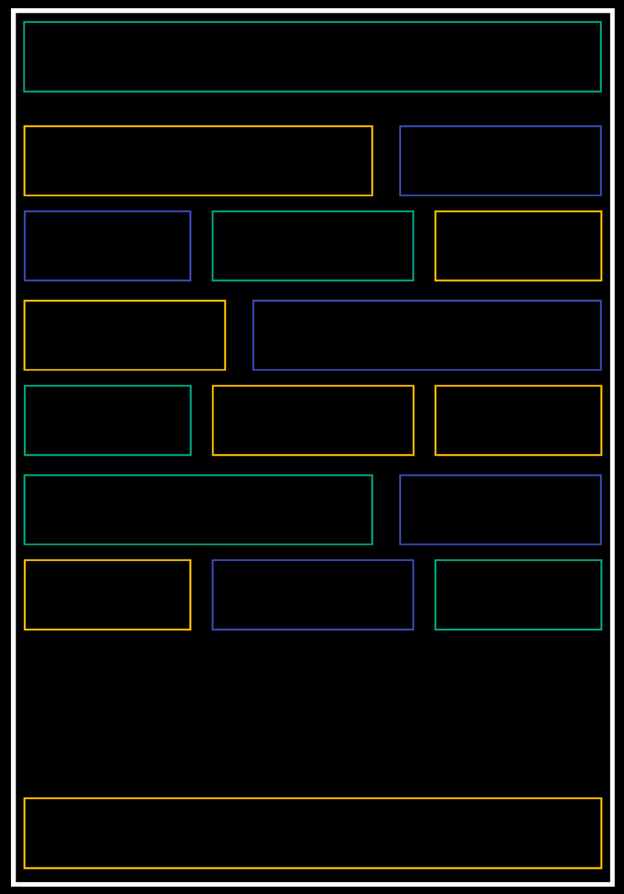


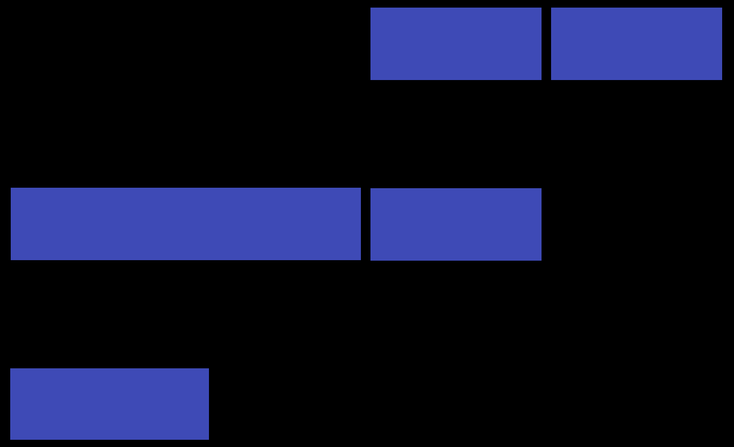
Client



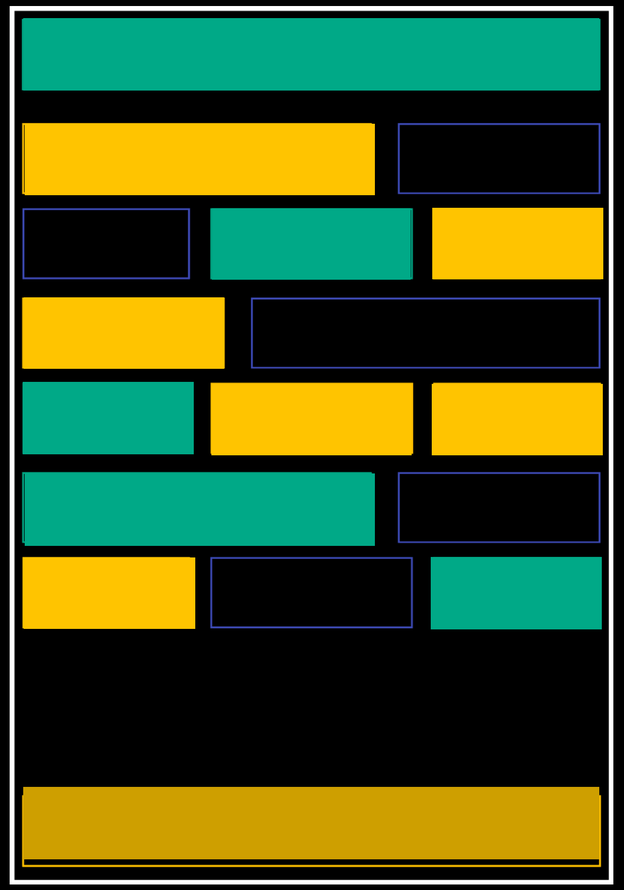
Streaming c suspense

Server



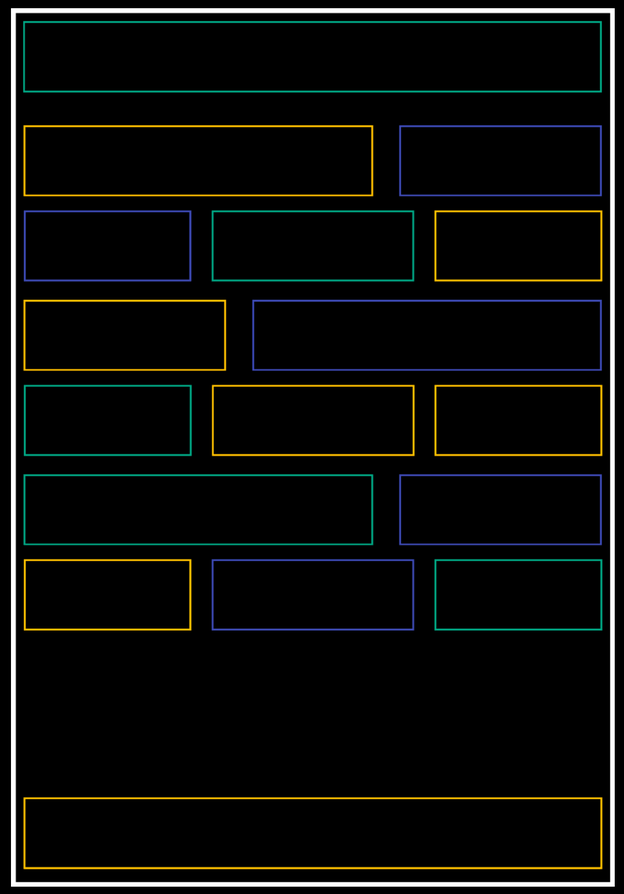


Client

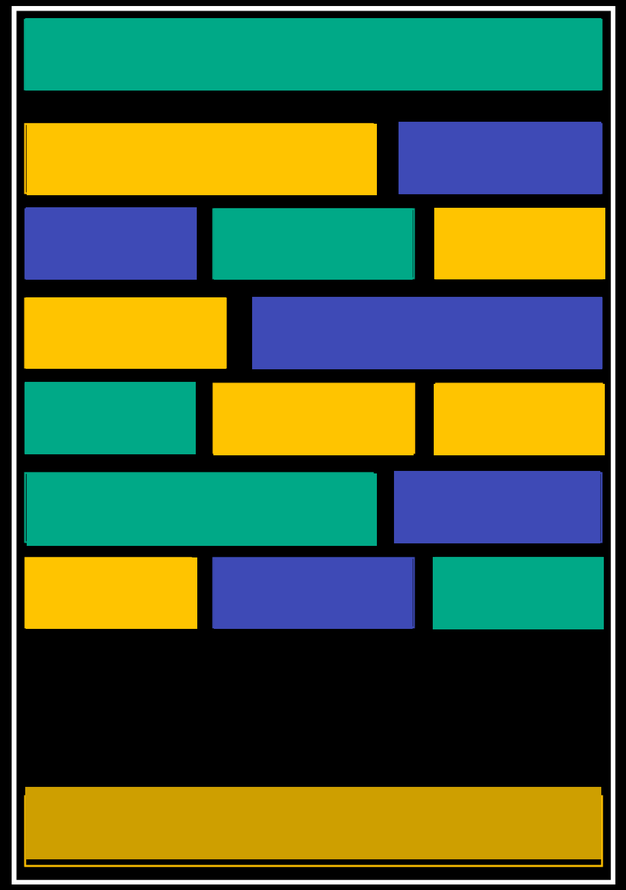


Streaming c suspense

Server

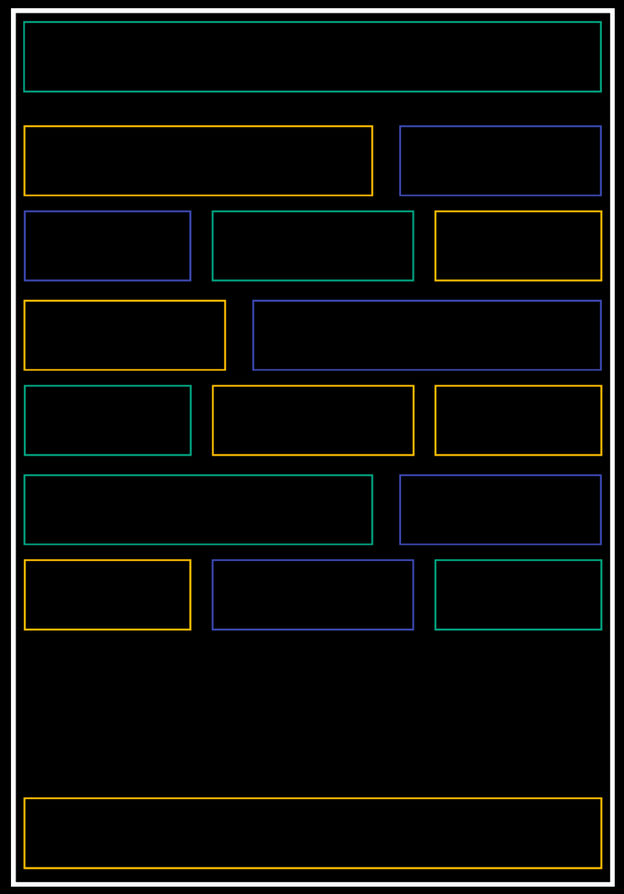


Client

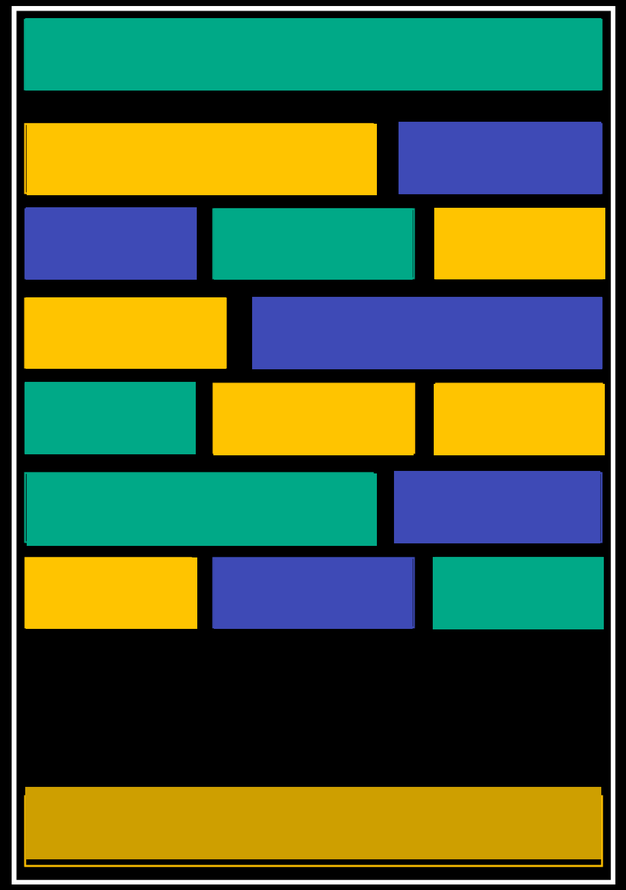


Streaming c suspense

Server



Client



Streaming c suspense

```
<Suspense fallback={<Loading />}>
  <SomeComponent />
</Suspense>
```

Какие есть проблемы?

Монолитность

Какие есть проблемы?

Монолитность

Все компоненты работают на клиенте, даже если это не нужно

Все компоненты работают на клиенте

Все компоненты работают на клиенте

Загружаем лишний код на клиент

Все компоненты работают на клиенте

Загружаем лишний код на клиент

Механизмы рендеринг реакта вынуждены всегда учитывать их

React Server Components

RSC

Компоненты, которые рендерятся только на сервере и только 1 раз

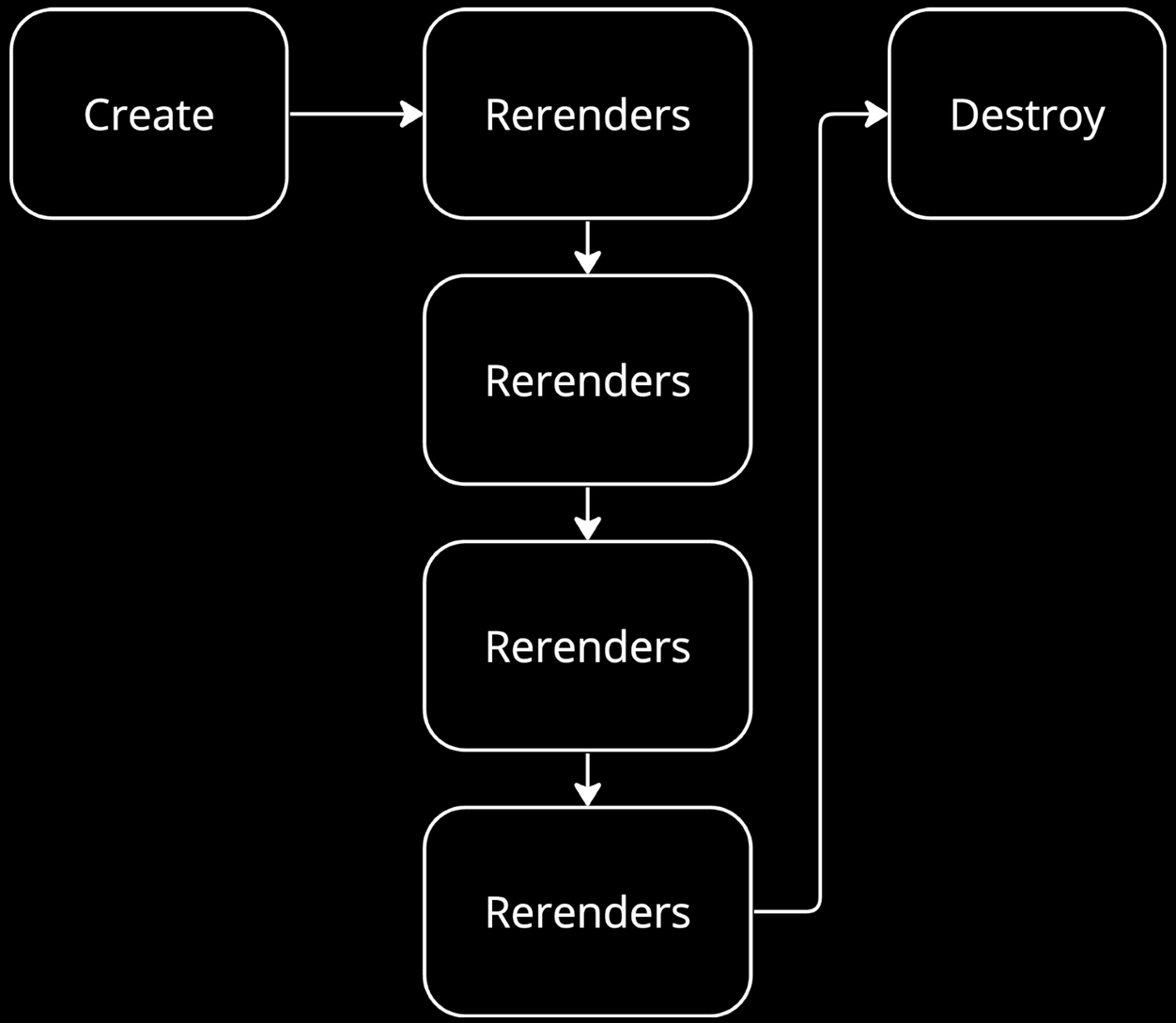
RSC

```
async function MoviesPage() {
  const movies = await getMoviesFromServer();

  return (
    <ul>
      {movies.map((movie) => (
        <li>{movie.name}</li>
      ))}
    </ul>
  );
}
export default MoviesPage;
```

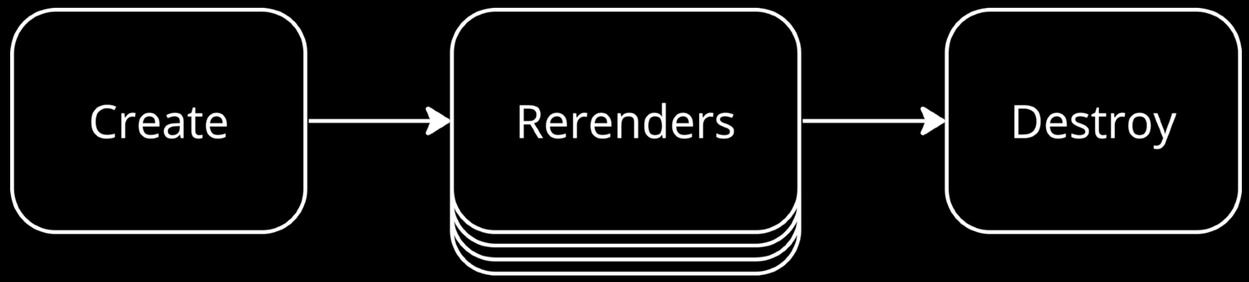
RSC

```
async function MoviesPage() {
  const movies = await getMoviesFromServer();

  return (
    <ul>
      {movies.map((movie) => (
        <li>{movie.name}</li>
      ))}
    </ul>
  );
}
export default MoviesPage;
```

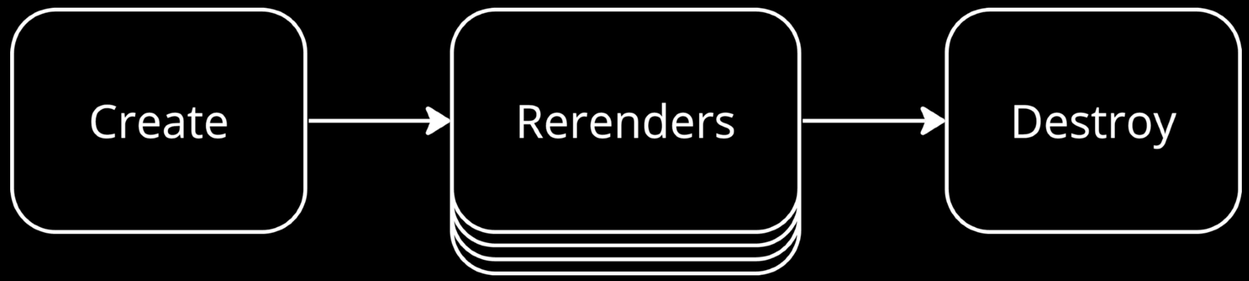
RSC



RSC



RSC



RSC

RSC

Нельзя поменять отображение

RSC

Нельзя поменять отображение

Нельзя использовать апи браузера

RSC

Нельзя поменять отображение

Нельзя использовать апи браузера

Нельзя использовать хуки
(как и большую часть API React)

RSC

Нельзя поменять отображение

Нельзя использовать апи браузера

Нельзя использовать хуки
(как и большую часть API React)

RSC

Простейшая загрузка данных
(никаких эффектов и тп)

RSC

Простейшая загрузка данных (никаких эффектов и тп)

Код необходимый для их рендеринга остается на сервере

RSC

Простейшая загрузка данных (никаких эффектов и тп)

Код необходимый для их рендеринга остается на сервере

Клиентской части механизма рендеринга React эти компоненты неинтересны совсем (почти)

RSC

Простейшая загрузка данных (никаких эффектов и тп)

Код необходимый для их рендеринга остается на сервере

Клиентской части механизма рендеринга React эти компоненты неинтересны совсем (почти)

RSC - можно ли использовать только их?

RSC - можно ли использовать только их?

Нет

Серверные ?

Серверные

Клиентские

Серверные

рендерятся только на сервере

Клиентские

рендерятся только на клиенте

Серверные

рендерятся только на сервере

Клиентские

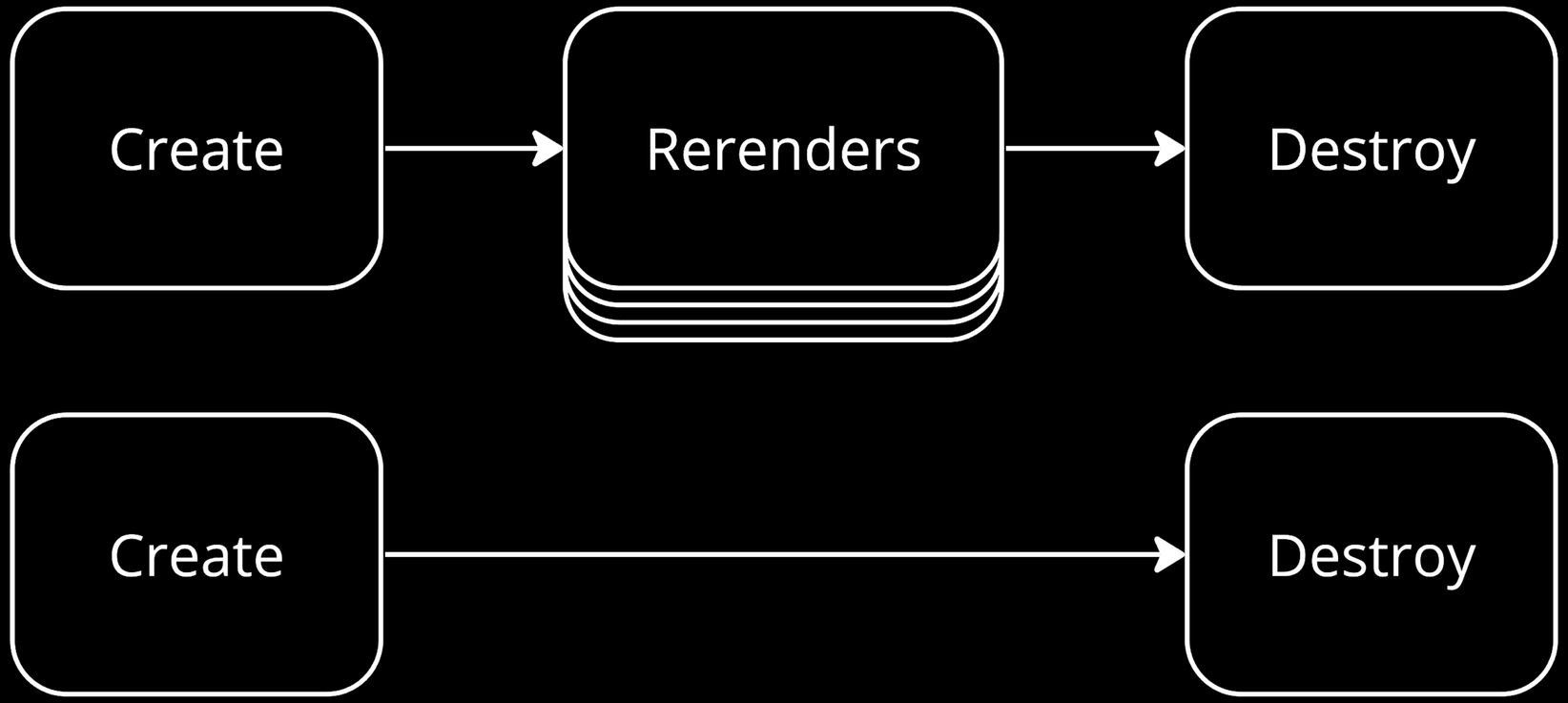
рендерятся только на клиенте

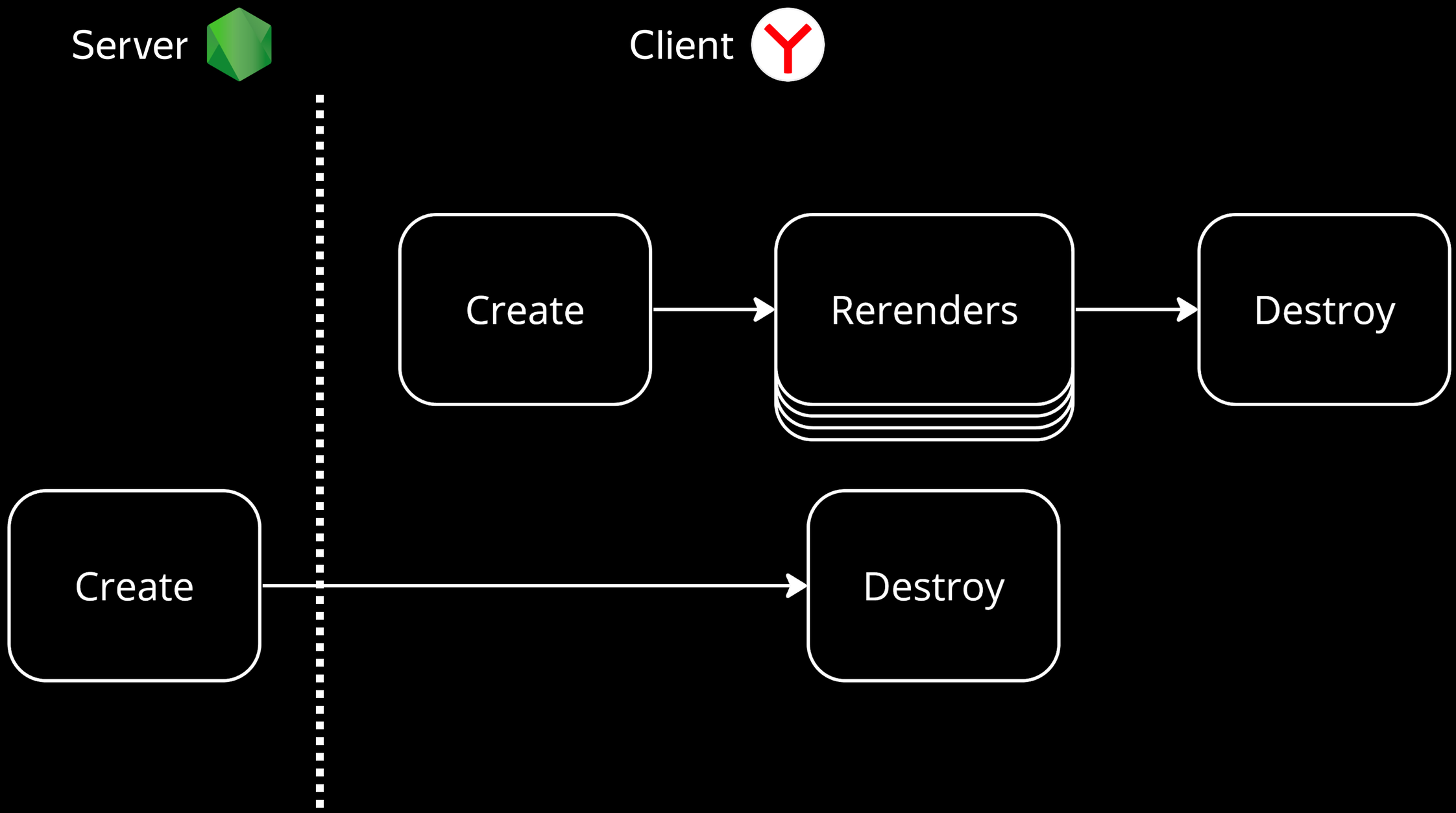
Серверные

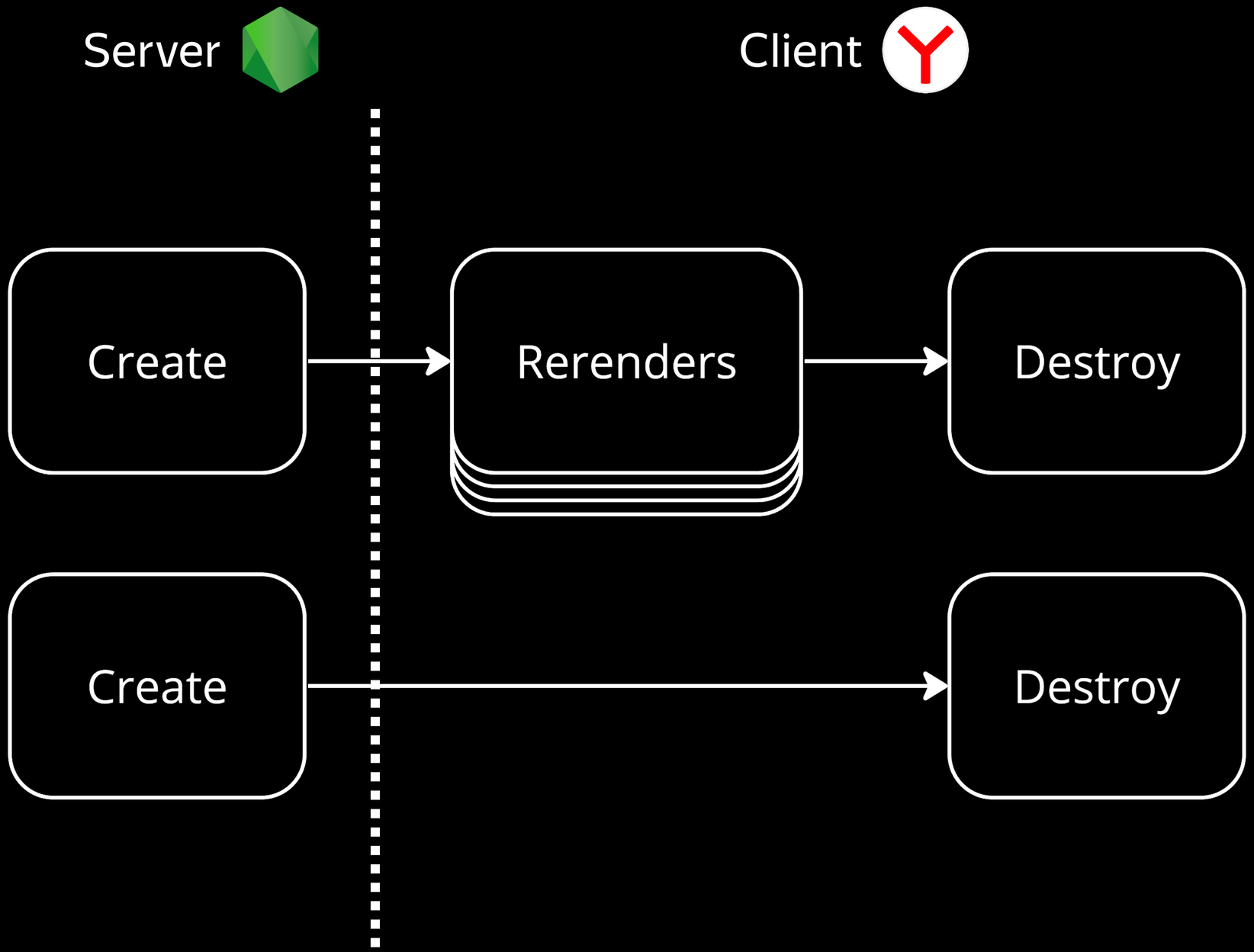
без
клиентской логики

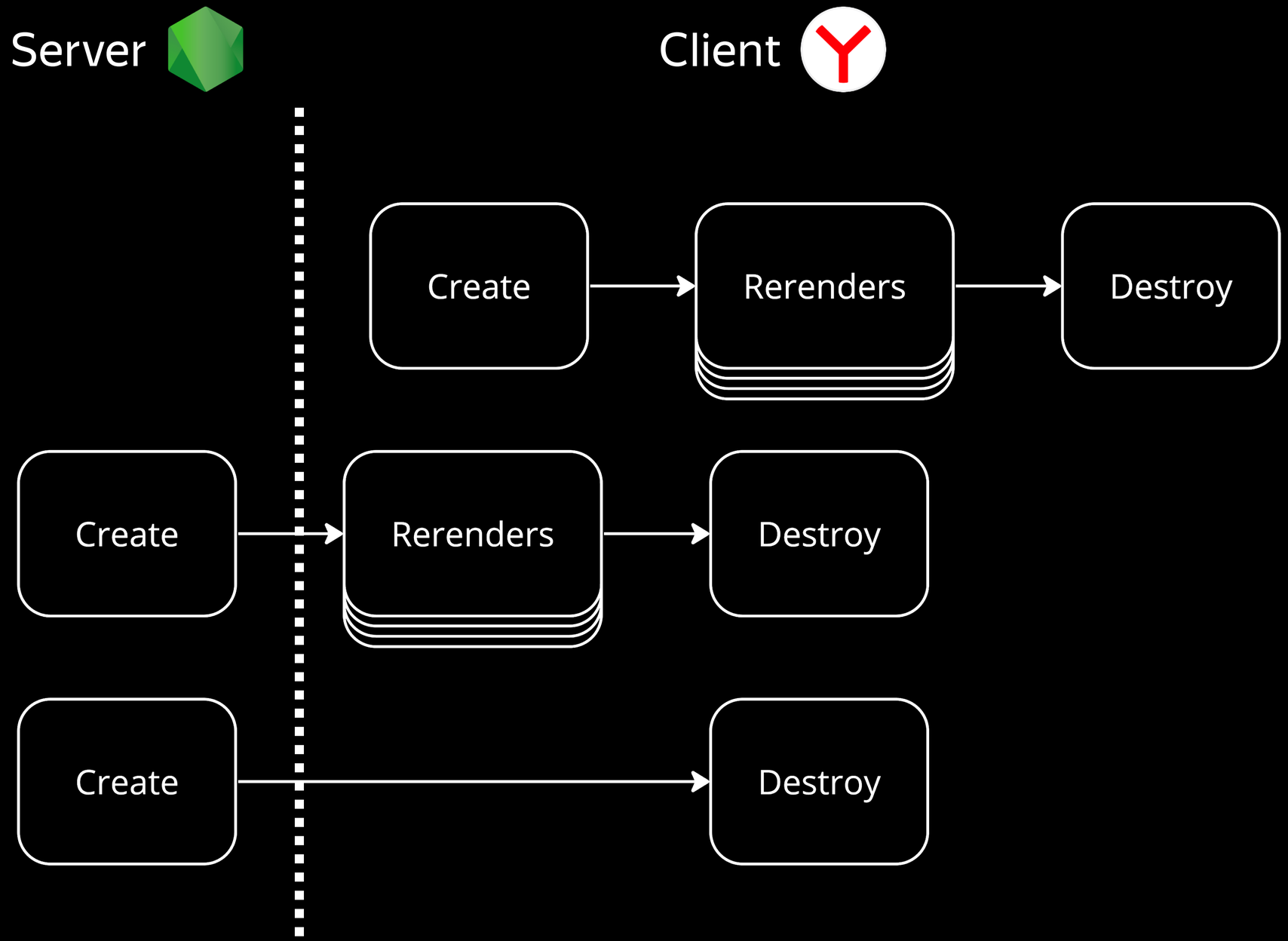
Клиентские

с
клиентской логикой

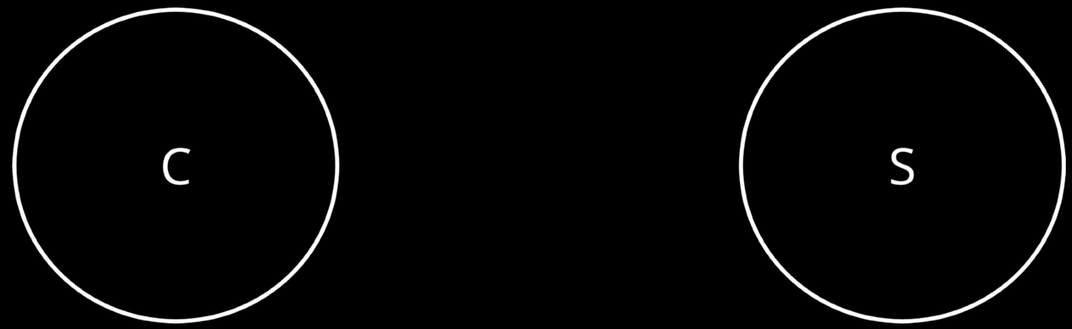

Create
Rerenders
Destroy
Create
Destroy


Server
Client
Create
Rerenders
Destroy
Create
Destroy


Server
Client
Create
Rerenders
Destroy
Create
Destroy


Server
Client
Create
Rerenders
Destroy
Create
Rerenders
Destroy
Create
Destroy

RSC



RSC

RSC

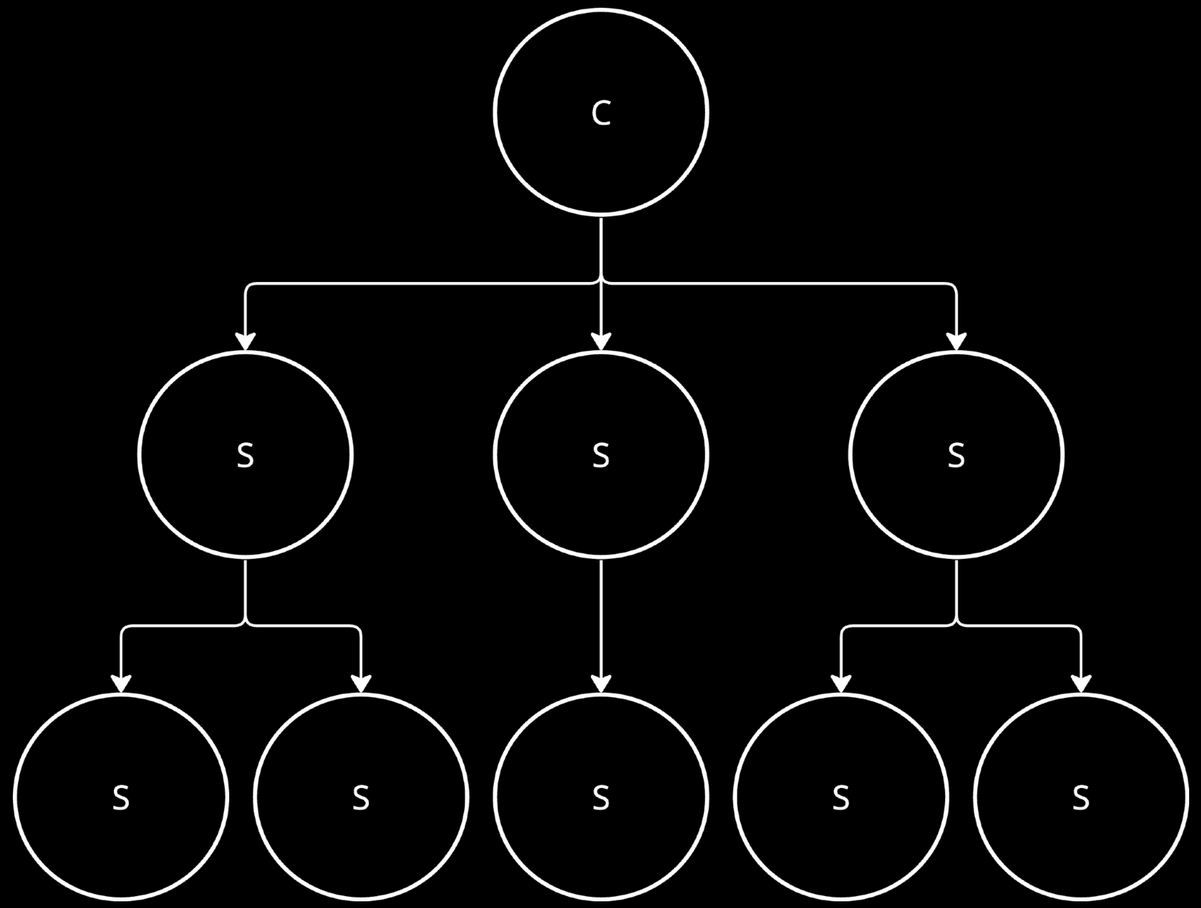
RSC

RSC

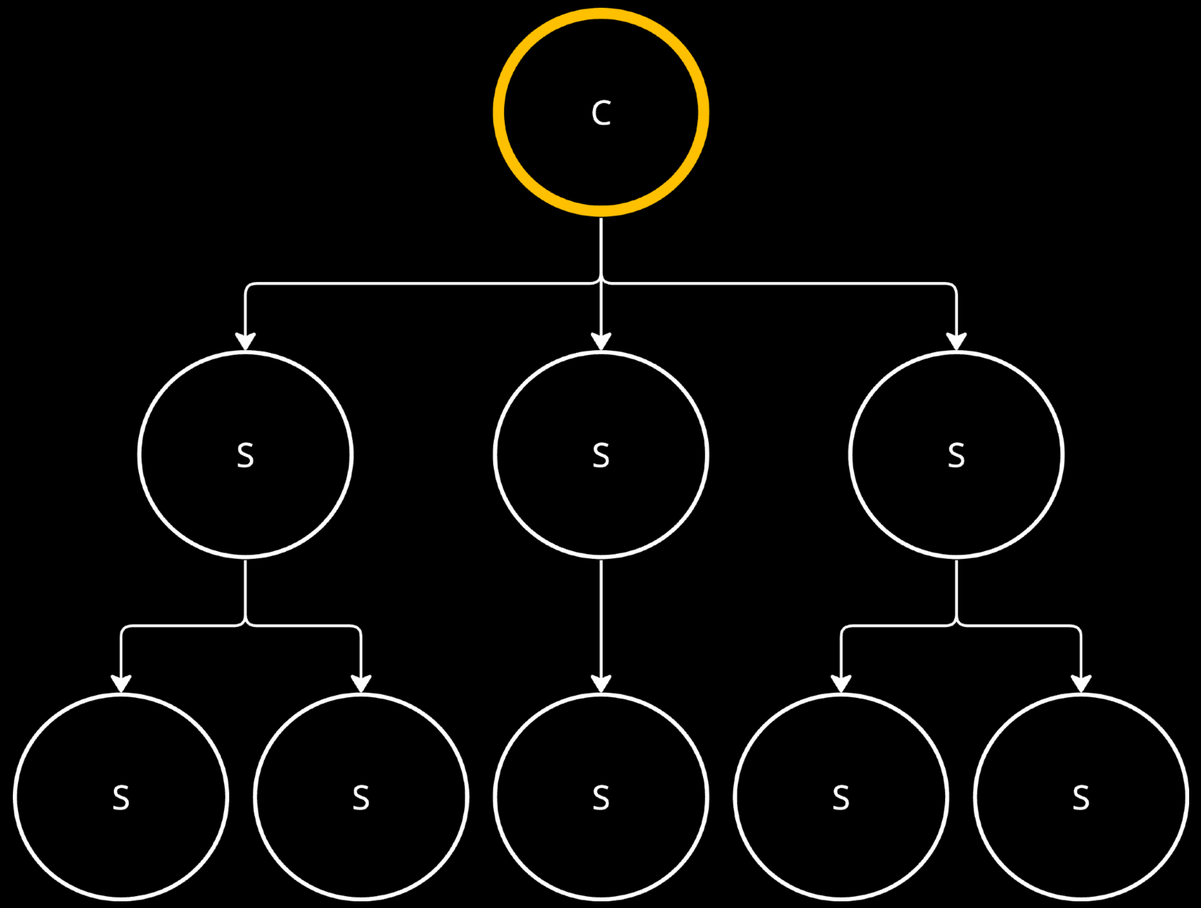
RSC

RSC

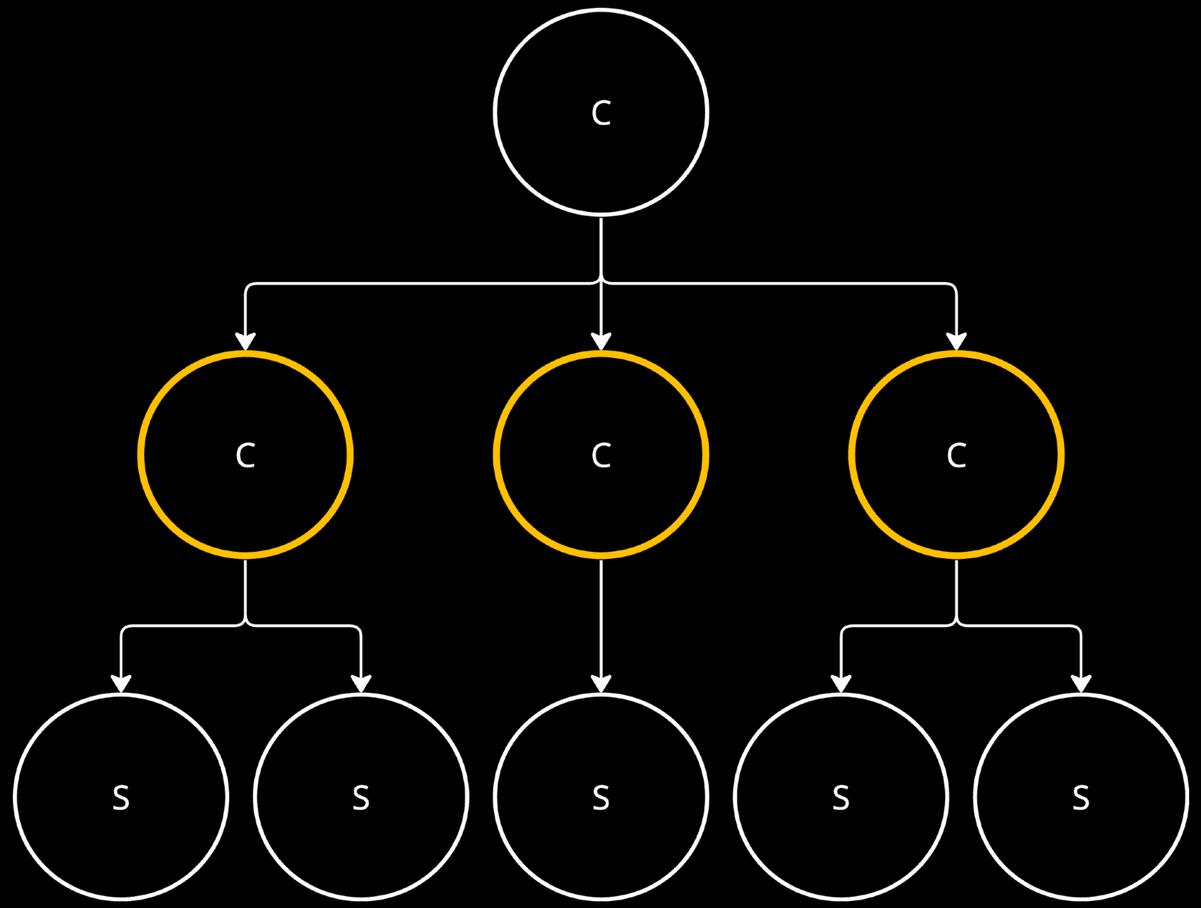
RSC



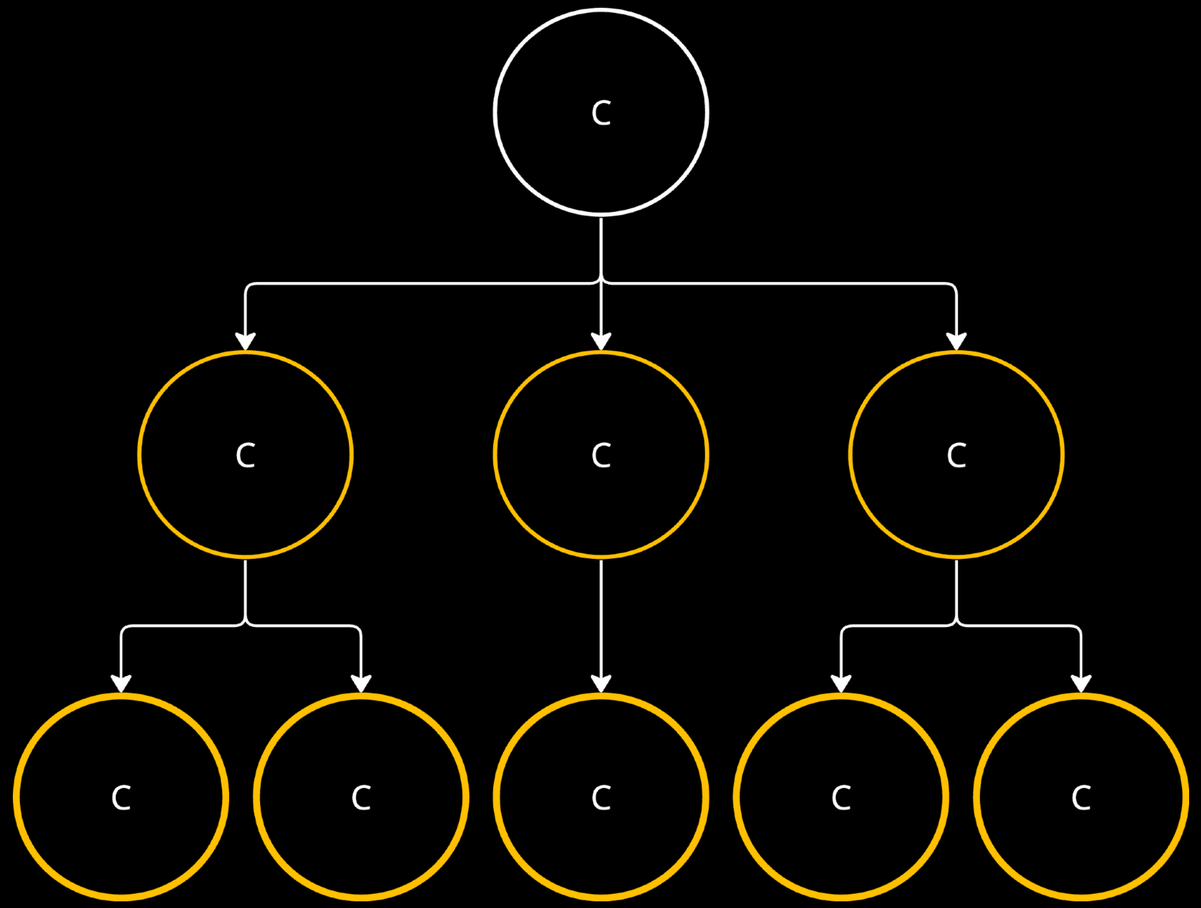
RSC



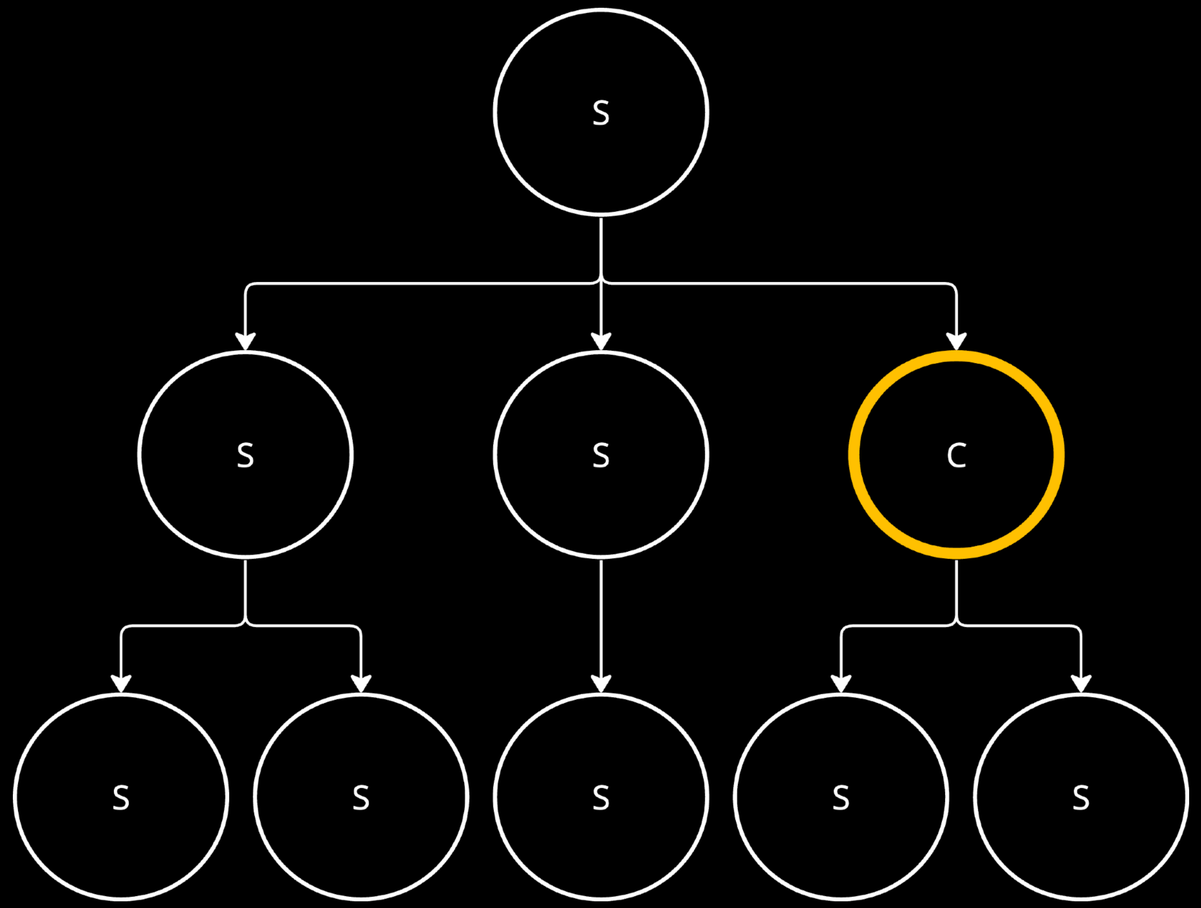
RSC



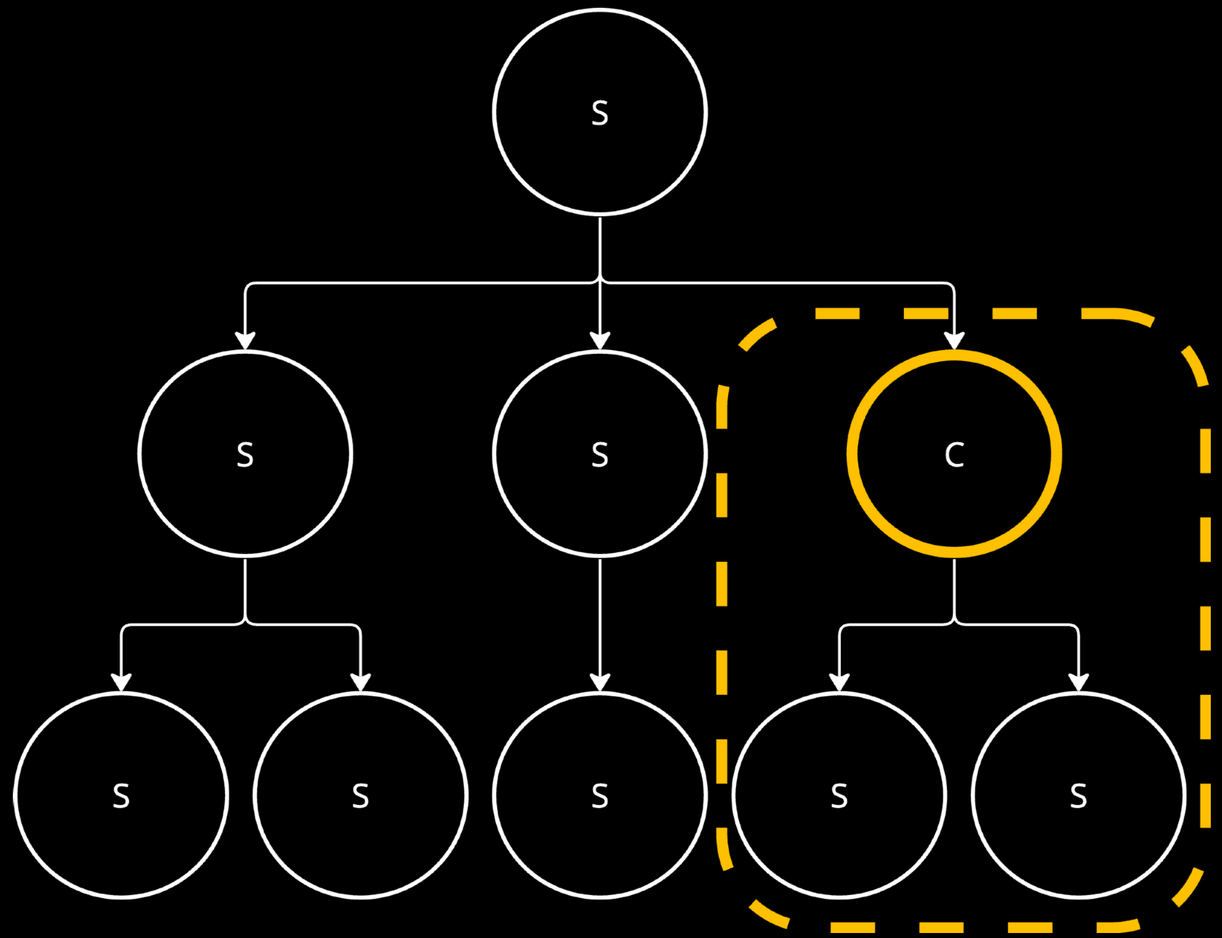
RSC



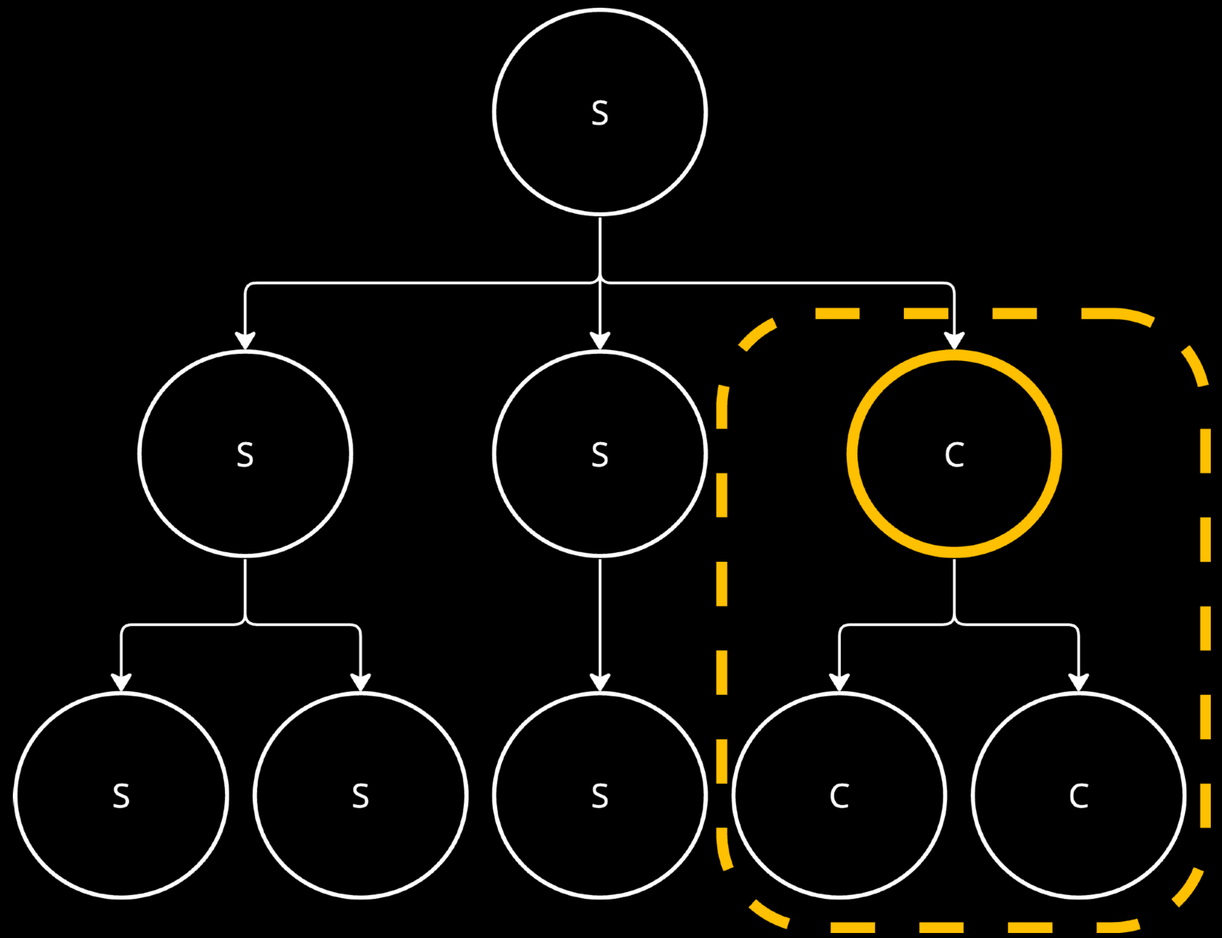
RSC



Client Boundary



Client Boundary



Next with App Router

Next with App Router

App Router

Клиент	CSR	
Сервер	SSG	SSR

App Router

Клиент  CSR

Сервер  STATIC DYNAMIC

App Router

Клиент  CSR

Сервер  STATIC
DYNAMIC

STREAMING

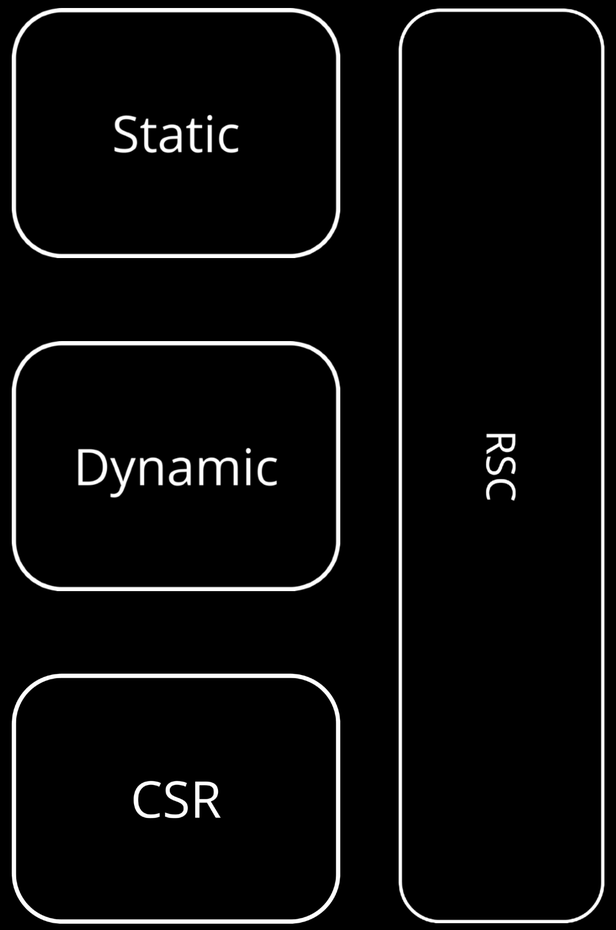
App Router

Static

Dynamic

CSR

App Router



RSC в NextJS

RSC в NextJS

Используются по умолчанию

RSC в NextJS

```
export function Movies() {
  return <div>Content</div>;
}
```

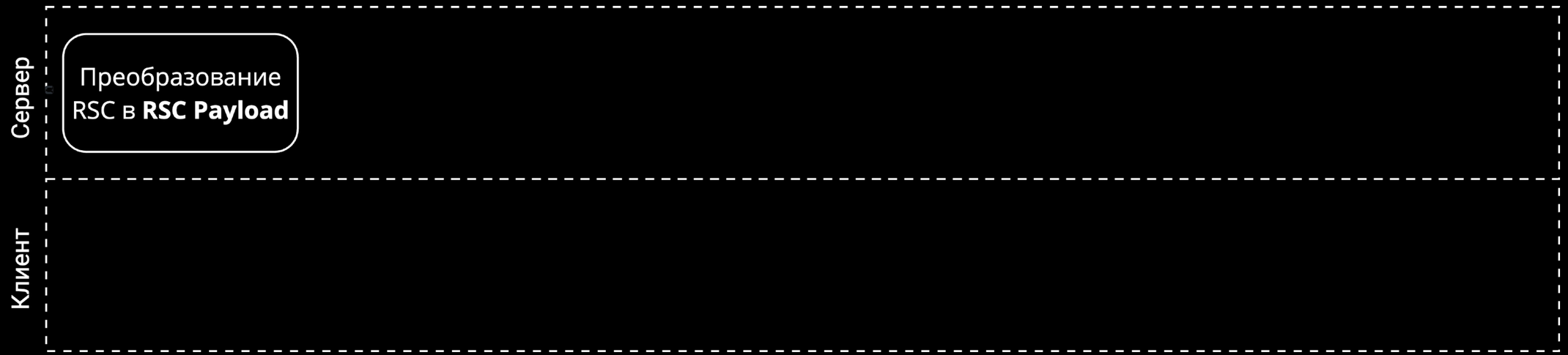
RSC в NextJS

```
export default async function Movie({ movieId })
  const movie = await getMovie(movieId);

  return <main>Movie: {movie.name}</main>;
}
```

RSC в NextJS

```
export default async function Movie({ movieId })
  const movie = await getMovie(movieId);

  return <main>Movie: {movie.name}</main>;
}
```

RSC в NextJS



RSC Payload

Результат рендеринга RSC

RSC Payload

Результат рендеринга RSC

Плейсхолдеры для клиентских компонентов и ссылки на их файлы

RSC Payload

Результат рендеринга RSC

Плейсхолдеры для клиентских компонентов и ссылки на их файлы

Данные передаваемые от RSC к клиентскому компоненту

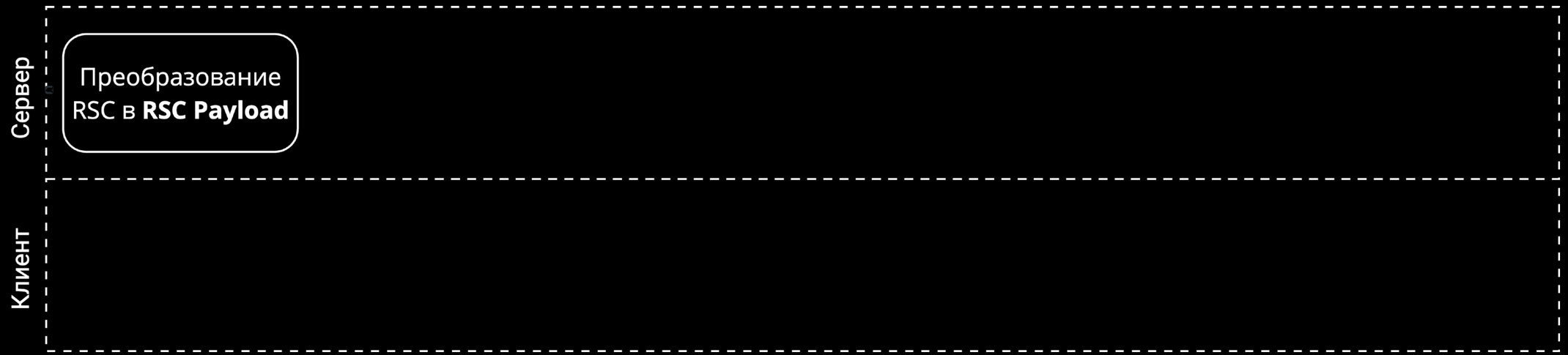
RSC Payload

Результат рендеринга RSC

Плейсхолдеры для клиентских компонентов и ссылки на их файлы

Данные передаваемые от RSC к клиентскому компоненту

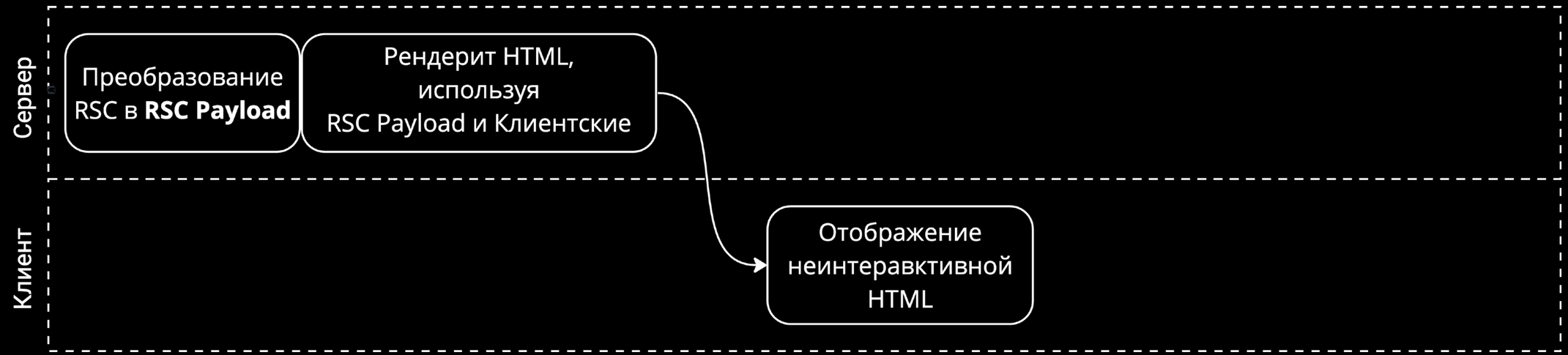
RSC в NextJS



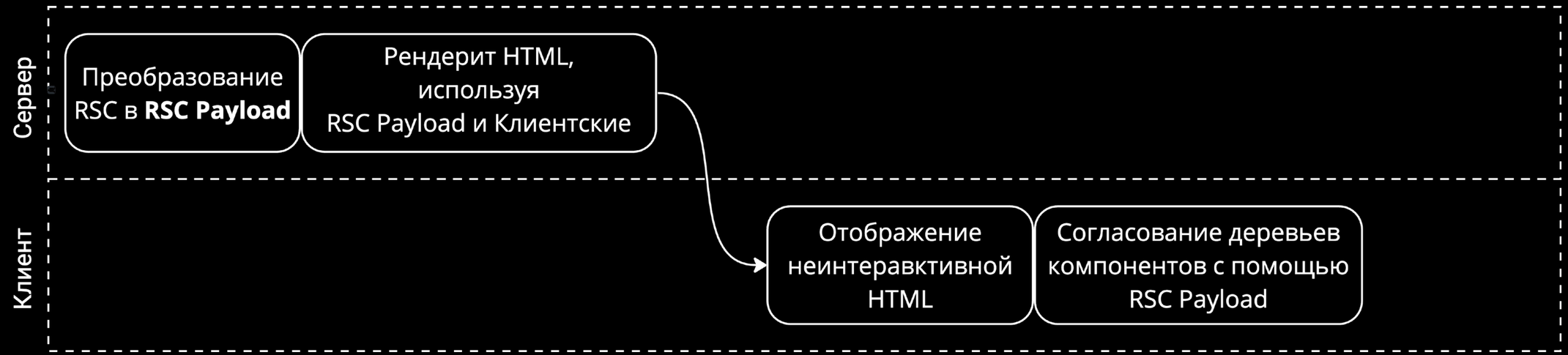
RSC в NextJS


Сервер
Преобразование
RSC в **RSC Payload**
Рендерит HTML,
используя
RSC Payload и Клиентские
Клиент

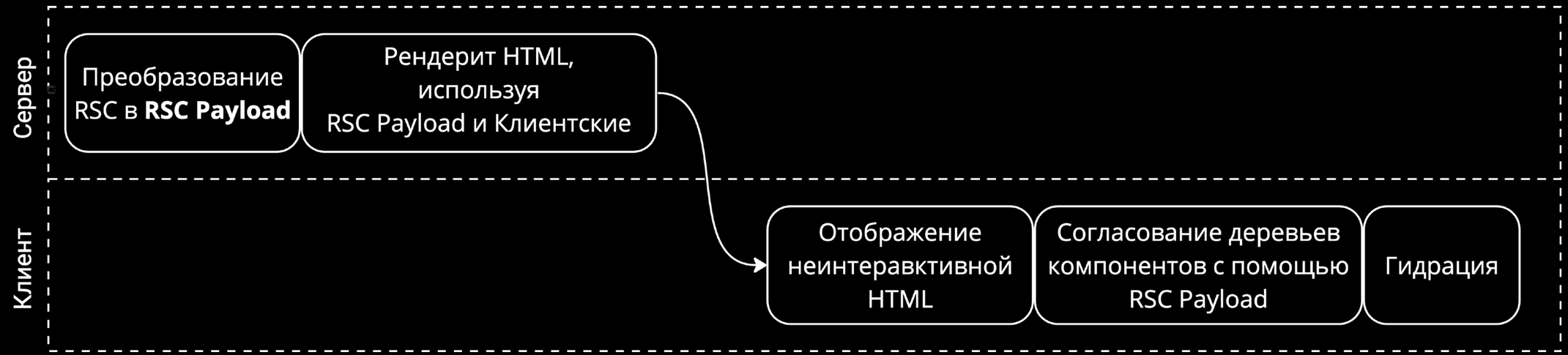
RSC в NextJS


Сервер
Преобразование RSC в **RSC Payload**
Рендерит HTML, используя RSC Payload и Клиентские
Клиент
Отображение неинтеравктивной HTML

RSC в NextJS


Сервер
Преобразование RSC в **RSC Payload**
Рендерит HTML, используя RSC Payload и Клиентские
Клиент
Отображение неинтеравктивной HTML
Согласование деревьев компонентов с помощью RSC Payload

RSC в NextJS


Сервер
Преобразование RSC в **RSC Payload**
Рендерит HTML, используя RSC Payload и Клиентские
Клиент
Отображение неинтеравктивной HTML
Согласование деревьев компонентов с помощью RSC Payload
Гидрация

RSC в NextJS

```
export function Movies() {
  return <div>Content</div>;
}
```

RSC в NextJS

```
export function Movies() {
  return (
    <div>
      Content
      <button onClick={() => {}}>Click</button>
    </div>
  );
}
```

RSC в NextJS

```
export function Movies() {
  return (
    <div>
      Content
      <button onClick={() => {}}>Click</button>
    </div>
  );
}
```

RSC в NextJS

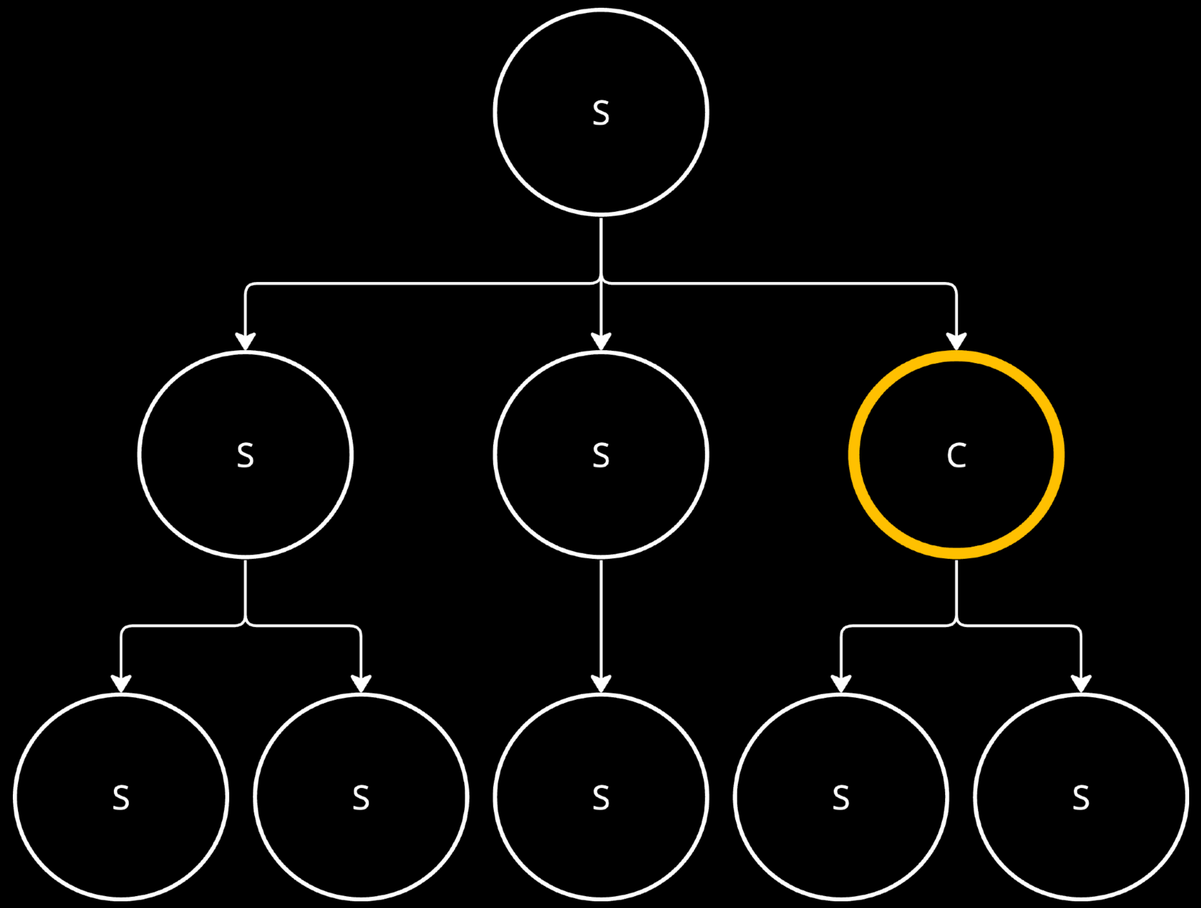
```
"use client"

export function Movies() {
  return (
    <div>
      Content
      <button onClick={() => {}}>Click</button>
    </div>
  );
}
```

RSC в NextJS

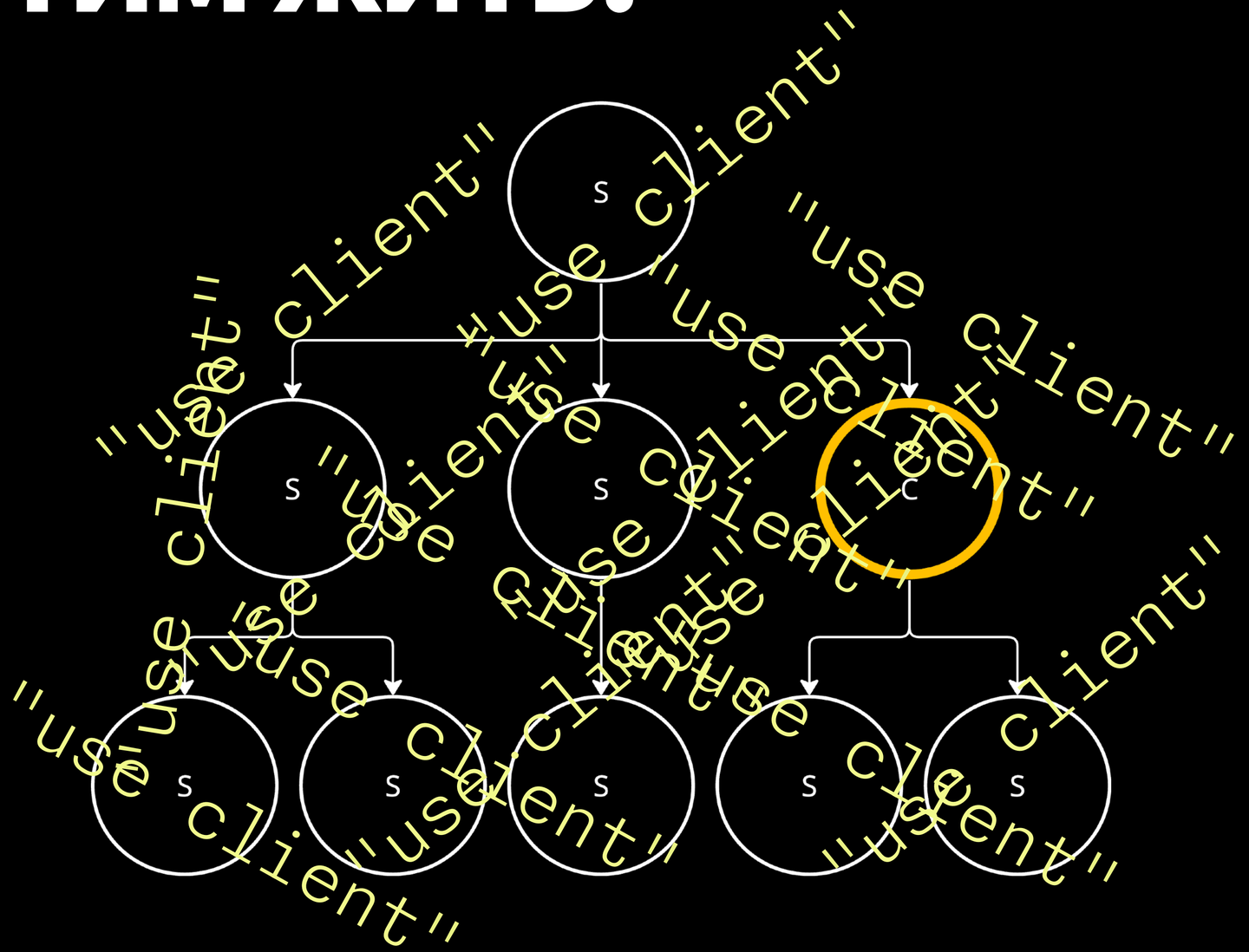
```
"use client"

export function Movies() {
  return (
    <div>
      Content
      <button onClick={() => {}}>Click</button>
    </div>
  );
}
```

Как с этим жить?

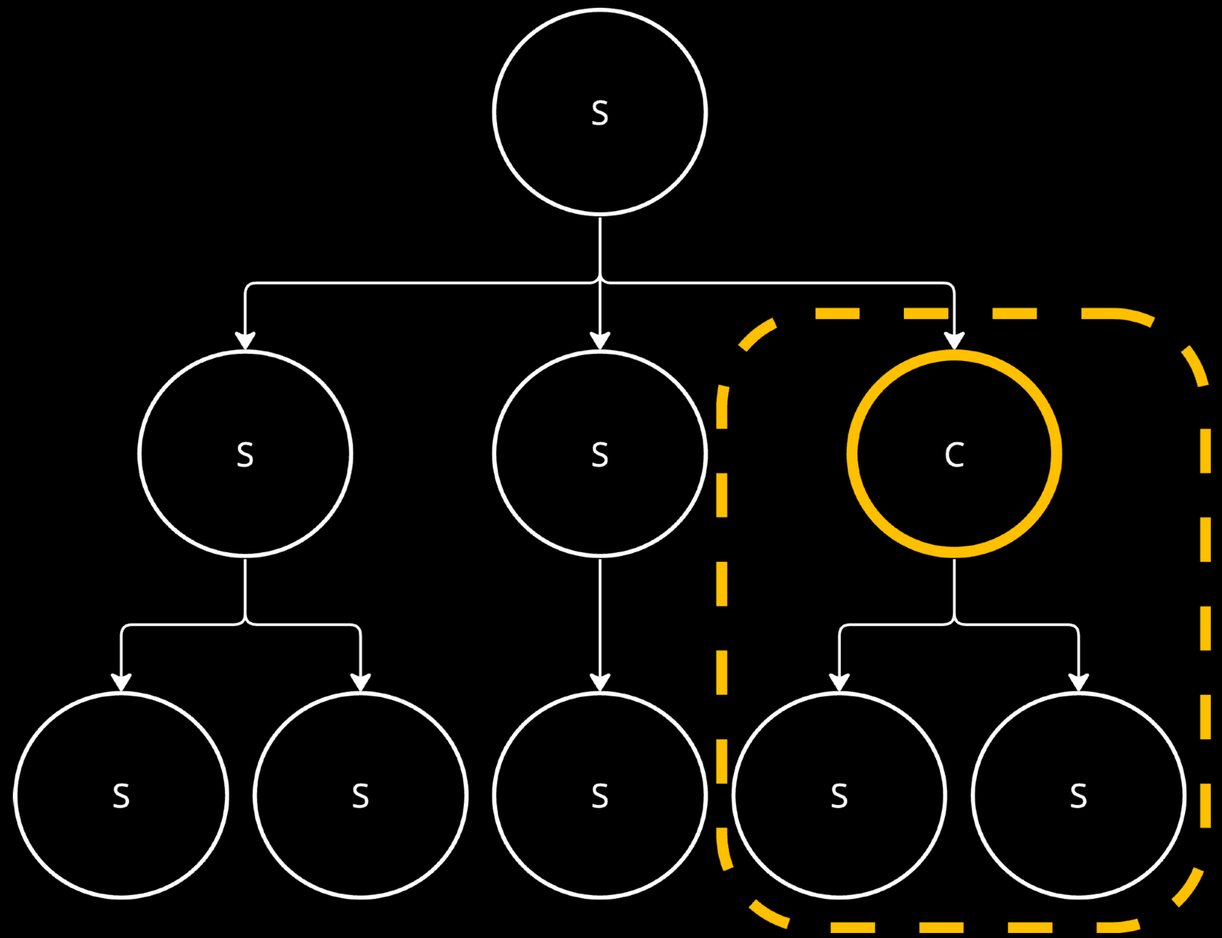
Как с этим жить?



Как с этим жить?



Client Boundary



Что делать? Как быть?

Грамотно разделяйте на компоненты

Разделяйте на компоненты

```
function Movies() {
  const [value, setValue] = useState();
  // prepare field logic
  return (
    <div>
      Many fields about Movie
      <button onClick={() => setValue(value + 1)}>Click</button>
    </div>
  );
}
```

Разделяйте на компоненты

```
function Movies() {
  const [value, setValue] = useState();
  // prepare field logic
  return (
    <div>
      Many fields about Movie
      <button onClick={() => setValue(value + 1)}>Click</button>
    </div>
  );
}
```

Разделяйте на компоненты

```
function Counter() {
  const [value, setValue] = useState();

  return (
        <button
          onClick={() => setValue(value + 1)}
         >
          Click
        </button>
    );
}
```

Разделяйте на компоненты

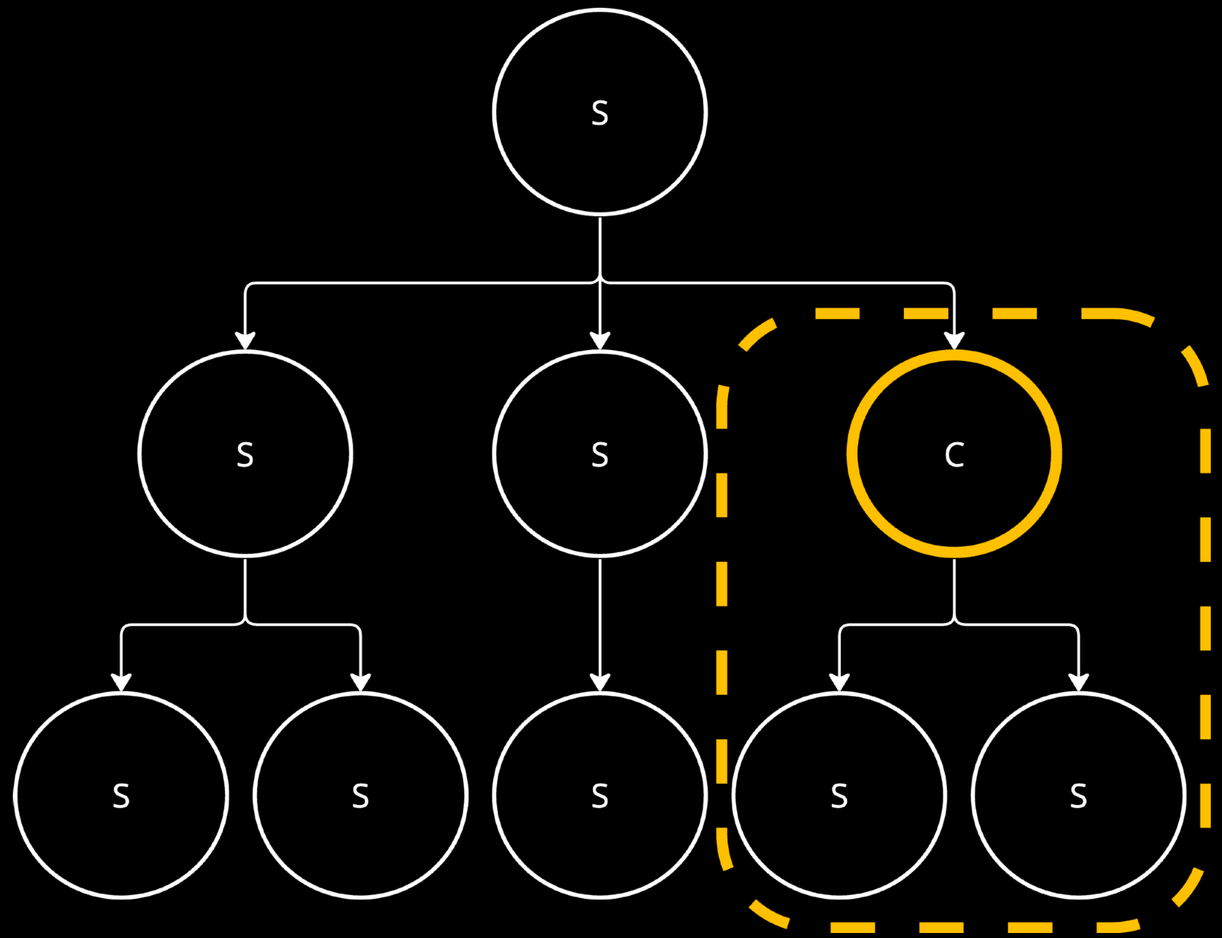
```
function Movies() {
  // prepare field logic

  return (
    <div>
      Many fields about Movie
      <Counter />
    </div>
  );
}
```

Что делать? Как быть?

Грамотно разделяйте на компоненты

Старайтесь держать клиентские компоненты ближе к листьям дерева

Client Boundary



Что делать? Как быть?

Грамотно разделяйте на компоненты

Старайтесь держать клиентские компоненты ближе к листьям дерева

Используйте псевдородители

Псевдородители

```
"use client"

function Page() {
  const [user, setUser] = useState();

  return (
    <UserContext.Provider
        value={{ user, setUser }}
      >
      <MyComponent />
    </UserContext.Provider>
  );
}
```

Псевдородители

```
"use client"

function Page() {
  const [user, setUser] = useState();

  return (
    <UserContext.Provider
        value={{ user, setUser }}
      >
       <MyComponent />
    </UserContext.Provider>
  );
}
```

Псевдородители

```
"use client"

function Page() {
  const [user, setUser] = useState();

  return (
    <UserContext.Provider
      value={{ user, setUser }}
    >
      <MyComponent />
    </UserContext.Provider>
  );
}
```

Псевдородители

```
"use client"

function Page() {
  const [user, setUser] = useState();

  return (
    <UserContext.Provider
        value={{ user, setUser }}
      >
      <MyComponent />
    </UserContext.Provider>
  );
}
```

Псевдородители

```
"use client"

function UserProvider({ children }) {
  const [user, setUser] = useState();

  return (
    <UserContext.Provider
        value={{ user, setUser }}
      >
      {children}
    </UserContext.Provider>
  );
}
```

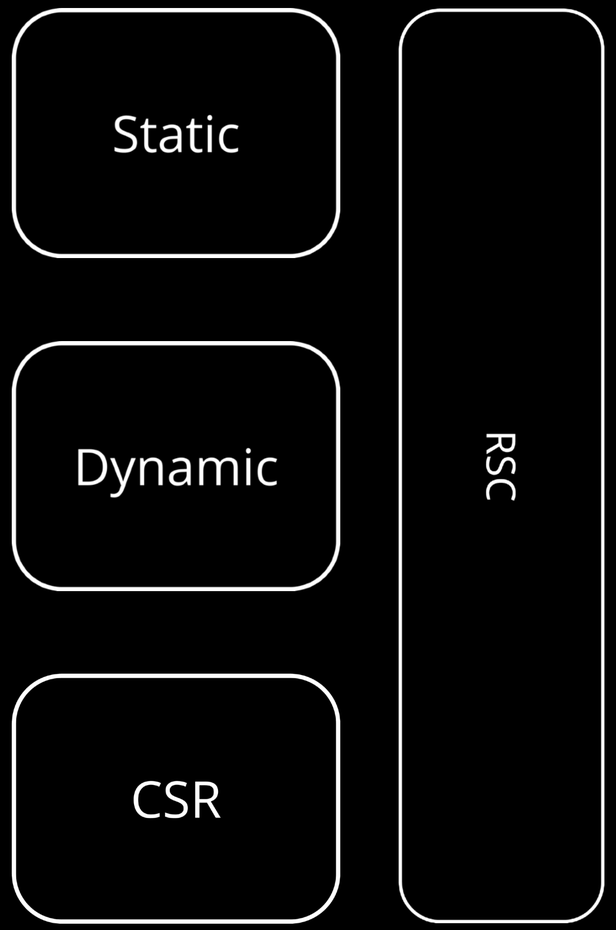
Псевдородители

```
"use client"

function Page() {

  return (
    <UserProvider>
      <MyComponent />
    </UserProvider>
  );
}
```

Псевдородители

```
function Page() {

  return (
    <UserProvider>
      <MyComponent />
    </UserProvider>
  );
}
```

App Router



App Router

Static

Использует во время билда и в фоне во время работы сервера

Dynamic

Использует во время работы сервера на запрос

CSR

Использует на клиенте

RSC

App Router

Static

Роут не динамический

App Router

Static

Роут не динамический

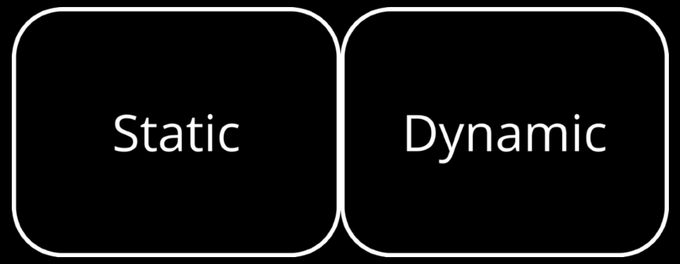
Данные либо не загружаются, либо кешируются

App Router

Dynamic

Если не получилось использовать Static

App Router



Но иногда их можно объединить

App Router

Static

Dynamic

```
export default async function Movie(
    { params: { movieId } }
) {
  const movie = await getMovie(movieId);

  return <main>Movie: {movie.name}</main>;
}
```

App Router

Static

Dynamic

```
Route (app)                                 Size     First Load JS
┌ ○ /                                       150 B          87 kB
├ ○ /_not-found                            871 B        87.8 kB
├ ƒ /[movieId]                             150 B          87 kB
└ ○ /movies                                150 B          87 kB
+ First Load JS shared by all              86.9 kB
  ├ chunks/214-5fd29cabce1feffa.js         31.4 kB
  ├ chunks/f6bcb39e-beaad6bc607e33d6.js    53.6 kB
  └ other shared chunks (total)            1.84 kB


○  (Static)   prerendered as static content
ƒ  (Dynamic)  server-rendered on demand
```

App Router

Static

Dynamic

```
Route (app)                              Size     First Load JS
┌ ○ /                                    150 B          87 kB
├ ○ / not-found                          871 B          87.8 kB
├ ƒ /[movieId]                           150 B          87 kB
└ ○ /movies                              150 B          87 kB
+ First Load JS shared by all            86.9 kB
  ├ chunks/214-5fd29cabce1feffa.js       31.4 kB
  ├ chunks/f6bcb39e-beaad6bc607e33d6.js  53.6 kB
  └ other shared chunks (total)          1.84 kB


○  (Static)   prerendered as static content
ƒ  (Dynamic)  server-rendered on demand
```

App Router

Static

Dynamic

```
export default async function Movie(
    { params: { movieId } }
) {
  const movie = await getMovie(movieId);

  return <main>Movie: {movie.name}</main>;
}
```

App Router

Static

Dynamic

```
export async function generateStaticParams() {
}

export default async function Movie(
    { params: { movieId } }
) {
  const movie = await getMovie(movieId);

  return <main>Movie: {movie.name}</main>;
}
```

App Router

Static

Dynamic

```
export async function generateStaticParams() {
  const movies = await getTop10();

  return movies.map(({ id }) => ({
    movieId: id,
  }));
}
```

App Router

Static

Dynamic

```
Route (app)                                Size     First Load JS
┌ ○ /                                      150 B          87 kB
├ ○ /_not-found                            871 B          87.8 kB
├ ● /[movieId]                             150 B          87 kB
├   ├ /d2d32
├   ├ /rr34d2ddasd32
├   ├ /dfvd2d32
├   └ [+6 more paths]
└ ○ /movies                                150 B          87 kB
+ First Load JS shared by all              86.9 kB
  ├ chunks/214-5fd29cabce1feffa.js         31.4 kB
  ├ chunks/f6bcb39e-beaad6bc607e33d6.js    53.6 kB
  └ other shared chunks (total)            1.84 kB


○  (Static)  prerendered as static content
●  (SSG)     prerendered as static HTML (uses getStaticProps)
```

App Router

Static

Dynamic

```
Route (app)                                 Size     First Load JS
┌ ○ /                                       150 B          87 kB
├ ○ / not-found                             871 B          87.8 kB
├ ● /[movieId]                              150 B          87 kB
├   ├ /d2d32
├   ├ /rr34d2ddasd32
├   ├ /dfvd2d32
├   └ [+6 more paths]
└ ○ /movies                                 150 B          87 kB
+ First Load JS shared by all               86.9 kB
  ├ chunks/214-5fd29cabce1feffa.js          31.4 kB
  ├ chunks/f6bcb39e-beaad6bc607e33d6.js     53.6 kB
  └ other shared chunks (total)             1.84 kB


○  (Static)  prerendered as static content
●  (SSG)     prerendered as static HTML (uses getStaticProps)
```

App Router

Static

Использует во время билда и в фоне во время работы сервера

Dynamic

Использует во время работы сервера на запрос если был использован компонент loading

CSR

Использует на клиенте

RSC

App Router

Static

Используется во время билда и в фоне во время работы сервера

Streaming

Dynamic

Используется во время работы сервера на запрос если был использован компонент loading

CSR

Используется на клиенте

RSC

App Router

Streaming

```
export default async function Movie(
    { params: { movieId } }
) {
  const movie = await getMovie(movieId);

  return <main>Movie: {movie.name}</main>;
}
```

App Router

Streaming

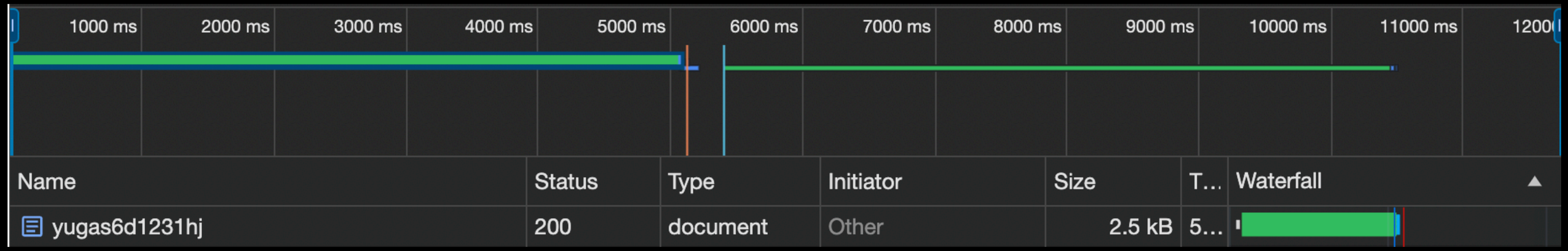
```
export default async function Movie(
    { params: { movieId } }
) {
  const movie = await getMovie(movieId);

  return <main>Movie: {movie.name}</main>;
}
```

App Router

Streaming

App Router

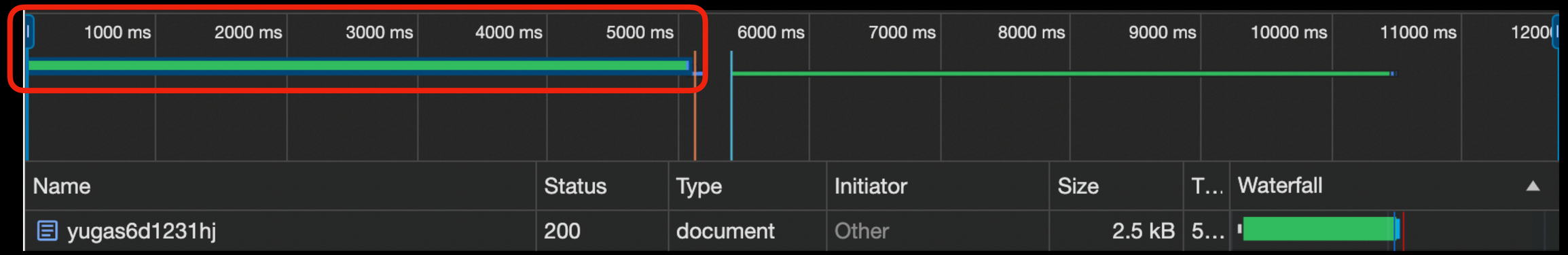
Streaming



App Router

Streaming

5000 ms



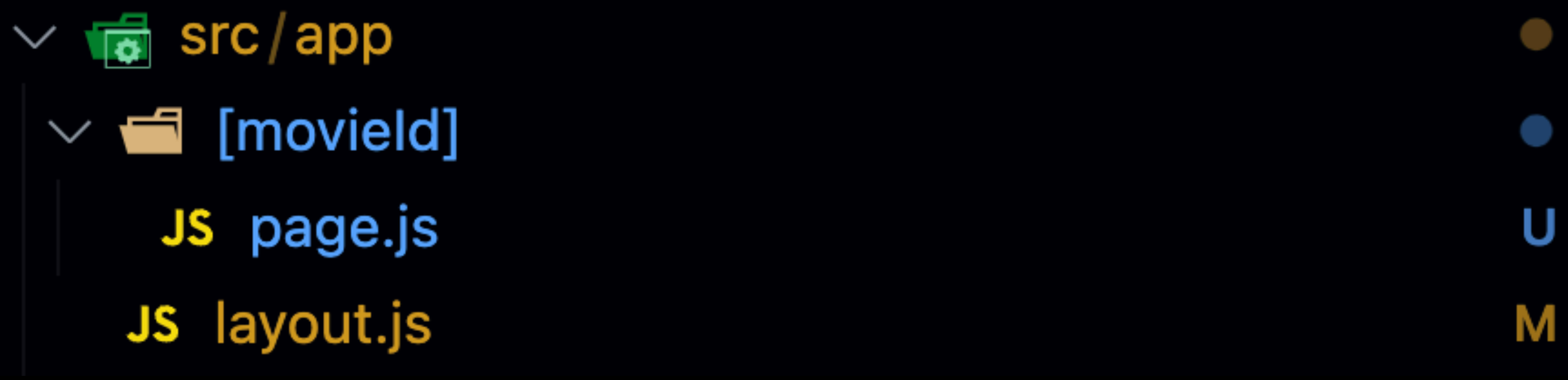
МОНОЛИТ

App Router

Streaming

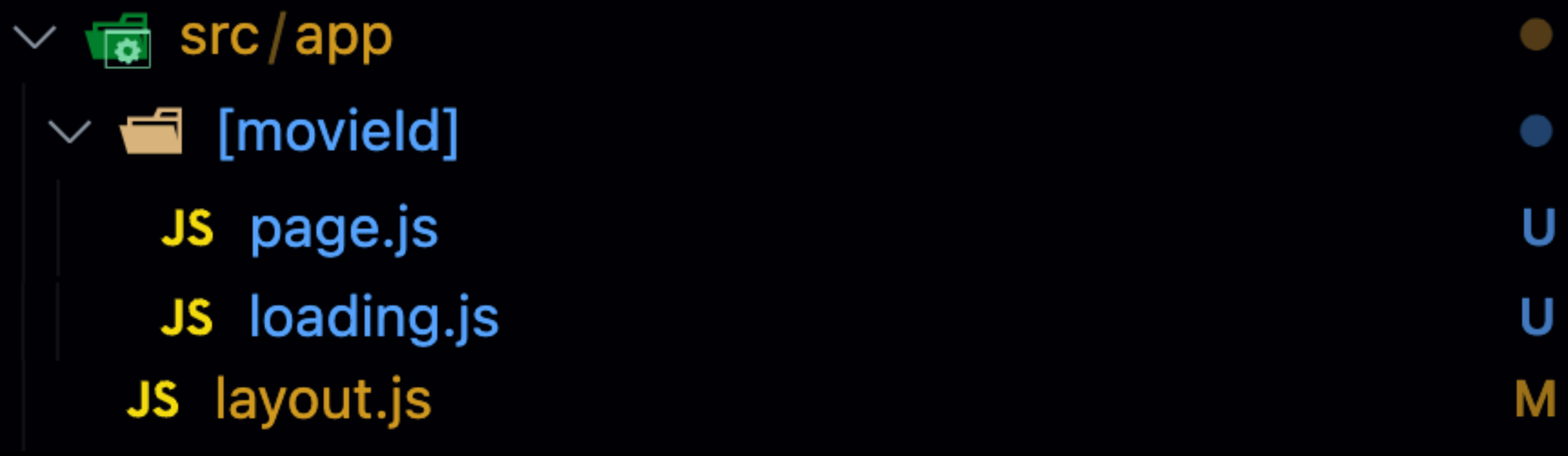
App Router

Streaming



App Router

Streaming



App Router

Streaming

App Router

Streaming

```
export default function Loading() {
  return <Skeleton />;
}
```

App Router

Streaming

```
export default function Loading() {
  return <Skeleton />;
}
```

Streaming c suspense

```
<Suspense fallback={<Loading />}>
  <SomeComponent />
</Suspense>
```

Как помочь NextJs рендерить?

Как помочь NextJs рендерить?

Меньше думать про рендеринг

Как помочь NextJs рендерить?

Меньше думать про рендеринг

Не мешать кэшу (fetch)

Как помочь NextJs рендерить?

Меньше думать про рендеринг

Не мешать кэшу (fetch)

Использовать loading.js и Suspence

Как помочь NextJs рендерить?

Меньше думать про рендеринг

Не мешать кэшу (fetch)

Использовать loading.js и Suspence

Грамотно составлять композицию компонентов

Как помочь NextJs рендерить?

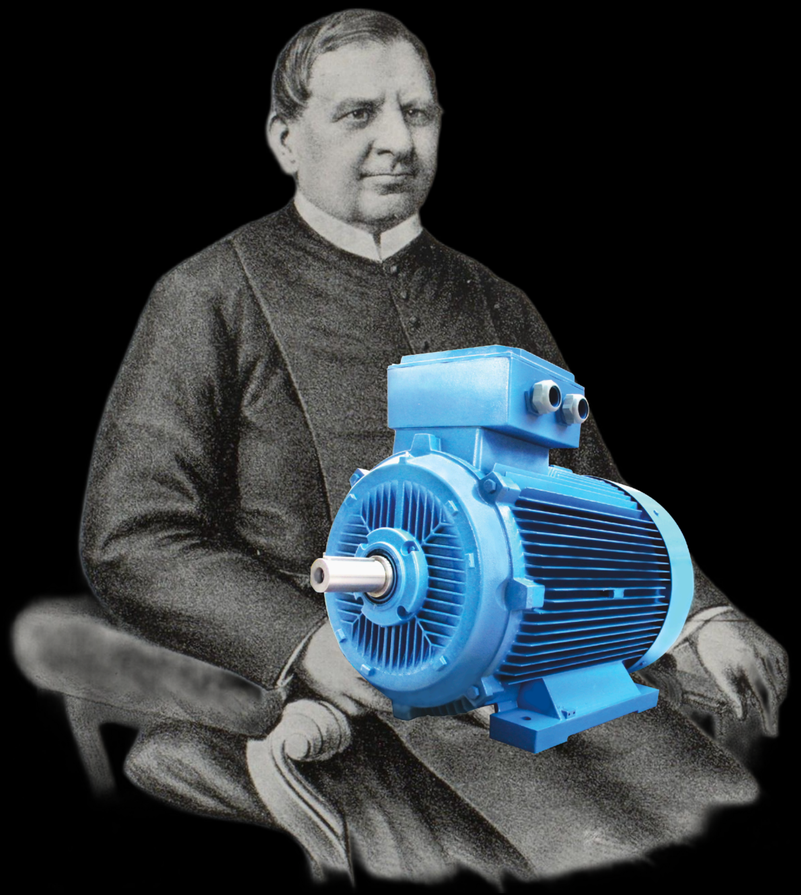
Меньше думать про рендеринг

Не мешать кэшу (fetch)

Использовать loading.js и Suspence

Грамотно составлять композицию компонентов

Аньош Йедлик - 1828



Илон Маск

Илон Маск - 2023


PHP


PHP

Спасибо за внимание

Тёма Сенюков
Кинопоиск
@temaProg