

NILS HARTMANN

<https://nilshartmann.net>

Slides: <https://react.schule/wjax2023>

Fullstack-Frameworks mit React

Die **Zukunft** der Frontend-Entwicklung?

W-JAX | MÜNCHEN, 08. NOVEMBER 2023 | @NILSHARTMANN

NILS HARTMANN

nils@nilshartmann.net

Freiberuflicher Entwickler, Architekt, Trainer aus Hamburg

Java, Spring, GraphQL, React, TypeScript



<https://graphql.schule/video-kurs>



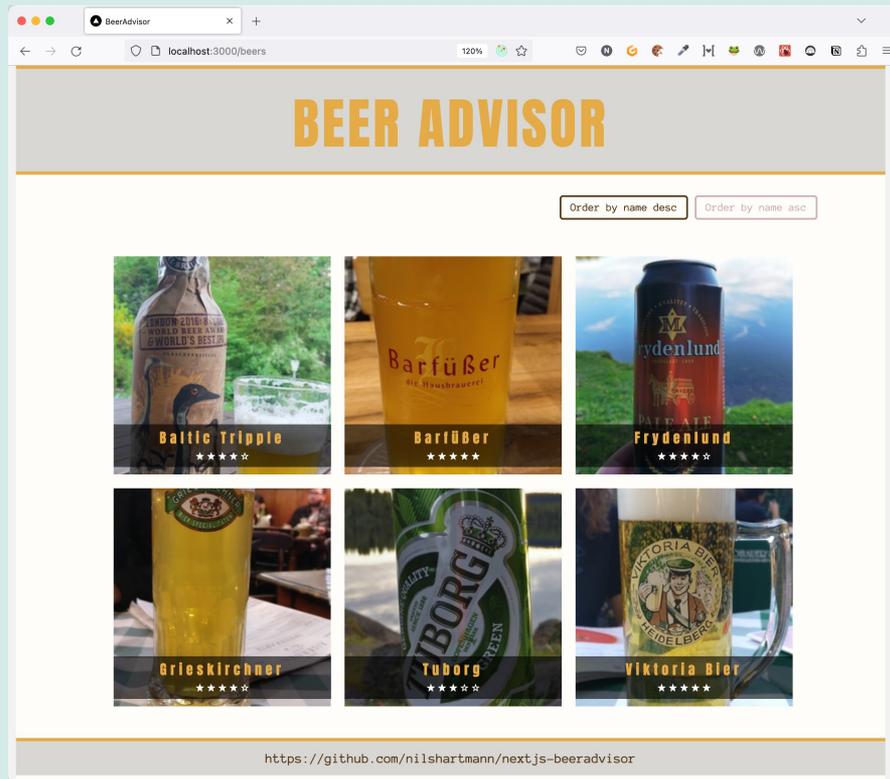
<https://reactbuch.de>

[HTTPS://NILSHARTMANN.NET](https://nilshartmann.net)

Go full-stack with a framework

React is a library. It lets you put components together, but it doesn't prescribe how to do routing and data fetching. To build an entire app with React, we recommend a full-stack React framework like Next.js or Remix.

<https://react.dev/>



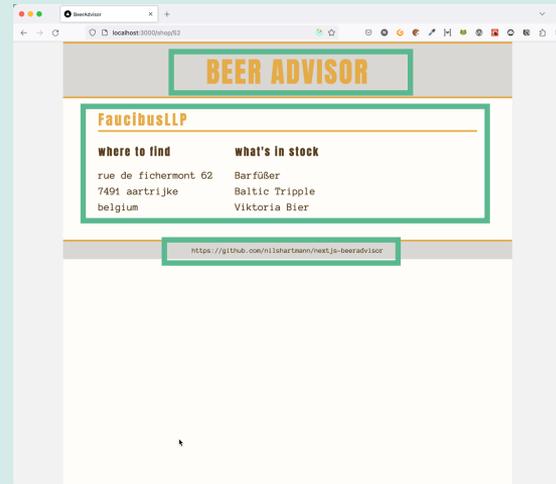
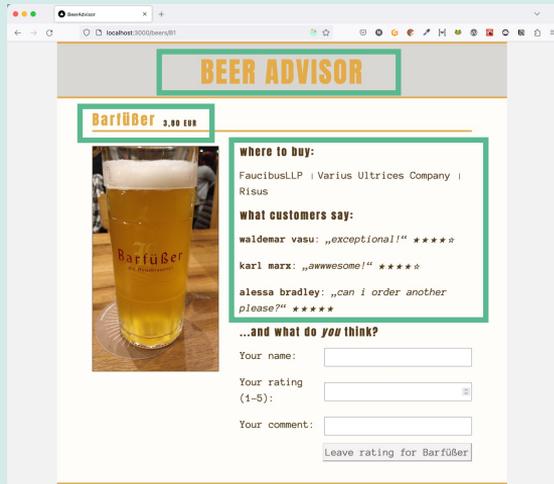
Beispiel-Code: <https://github.com/nilshartmann/nextjs-beeradvisor>

EIN BEISPIEL...

EIN BEISPIEL

Was macht die Beispiel-Anwendung aus?

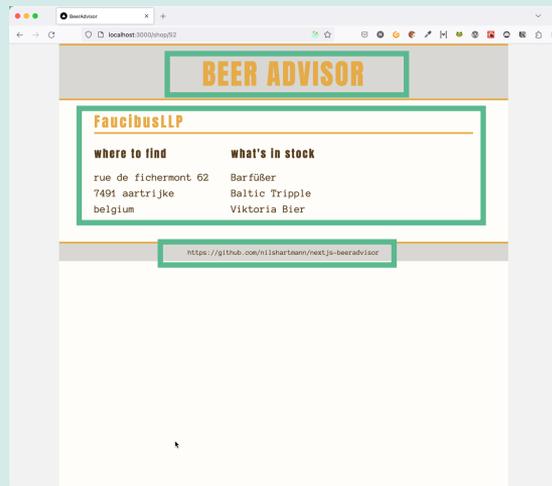
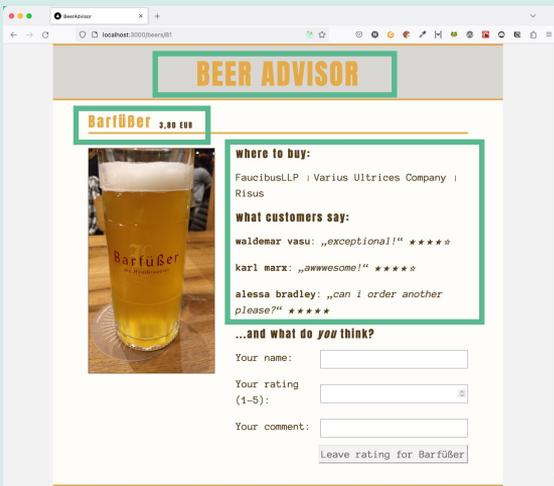
- Viel Bier 🍺



EIN BEISPIEL

Was macht die Beispiel-Anwendung aus?

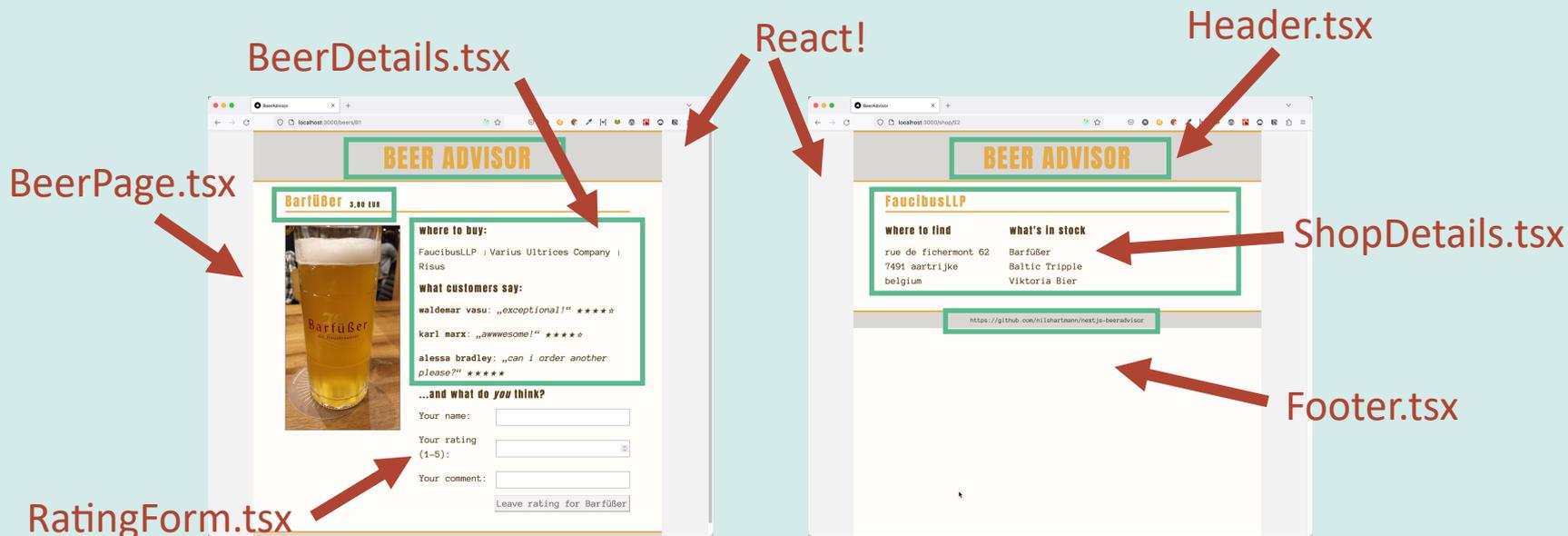
- Viel statischer Content 😊



EIN BEISPIEL

Was macht die Beispiel-Anwendung aus?

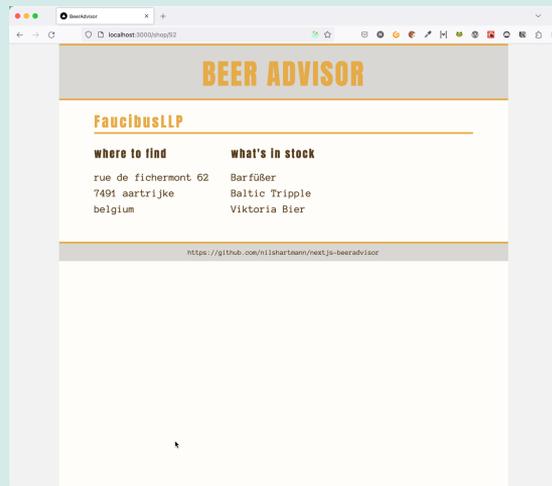
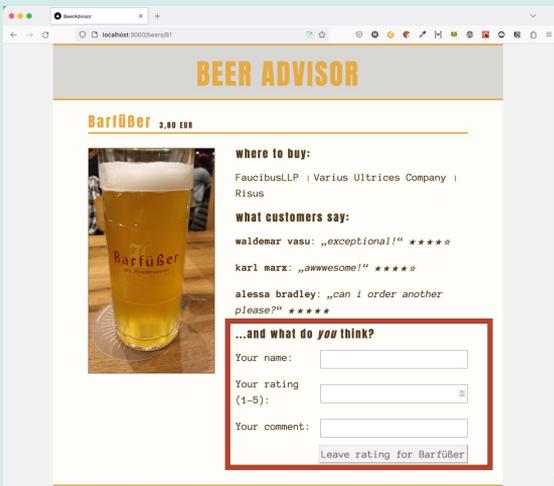
- Viel statischer Content 😊
- Viel JavaScript 😱



EIN BEISPIEL

Was macht die Beispiel-Anwendung aus?

- Viel statischer Content 😊
- Viel JavaScript 😱
- ...gleichzeitig wenig Interaktion 😞



Anforderung

👉 Die Seiten sollen möglichst schnell für den Benutzer **sichtbar** und **bedienbar** sein

Mögliche Probleme

- (Viel) JavaScript-Code, der...

Mögliche Probleme

- (Viel) JavaScript-Code, der...
 - ... vom Browser geladen werden muss

Mögliche Probleme

- (Viel) JavaScript-Code, der...
 - ... vom Browser geladen werden muss
 - ... interpretiert und ausgeführt werden muss

Mögliche Probleme

- (Viel) JavaScript-Code, der...
 - ... vom Browser geladen werden muss
 - ... interpretiert und ausgeführt werden muss
- ...und mit jeder neuen Komponente mehr wird

"Fullstack Architektur-Vision"

<https://react.dev/learn/start-a-new-react-project#which-features-make-up-the-react-teams-full-stack-architecture-vision>

"Fullstack Architektur-Vision"

<https://react.dev/learn/start-a-new-react-project#which-features-make-up-the-react-teams-full-stack-architecture-vision>

- **React Server Components (RSC):**
 - Komponenten, die auf dem Server, Client und im Build gerendert werden können
 - Data Fetching "integriert"

"Fullstack Architektur-Vision"

<https://react.dev/learn/start-a-new-react-project#which-features-make-up-the-react-teams-full-stack-architecture-vision>

- **React Server Components (RSC):**

- Komponenten, die auf dem Server, Client und im Build gerendert werden können
- Data Fetching "integriert"

- **Suspense:**

- Platzhalter für "langsame" Teile einer Seite
- Mit Streaming können diese Teile einer Seite "nachgeliefert" werden, sobald sie gerendert sind

React empfiehlt "Fullstack-Framework"

- **Server Components** erfordern Rendern auf dem Server oder im Build
- Dazu braucht man ein "**Fullstack-Framework**"

React empfiehlt "Fullstack-Framework"

- **Server Components** erfordern Rendern auf dem Server oder im Build
- Dazu braucht man ein "**Fullstack-Framework**"
- "**Framework**" ist verharmlosend, weil es sich in der Regel um einen kompletten Stack samt Build-Tools und Laufzeitumgebung handelt

React empfiehlt "Fullstack-Framework"

- **Server Components** erfordern Rendern auf dem Server oder im Build
- Dazu braucht man ein "**Fullstack-Framework**"
- "**Framework**" ist verharmlosend, weil es sich in der Regel um einen kompletten Stack samt Build-Tools und Laufzeitumgebung handelt
- Deswegen werden solche Frameworks auch als "**Meta-Frameworks**" bezeichnet (=> Sammlung von Frameworks)

React empfiehlt "Fullstack-Framework"

- **Next.js** entspricht den Vorstellungen des React-Teams

React empfiehlt "Fullstack-Framework"

- **Next.js** entspricht den Vorstellungen des React-Teams
- **Remix** unterstützt noch keine RSC, hat aber ähnliche Features

Zero-Bundle-Size
Server
Components

SERVER COMPONENTS

Idee: Komponenten werden nicht im Client ausgeführt

- Sie stehen auf dem Client nur fertig gerendert zur Verfügung
- Der Server schickt lediglich eine *Repräsentation der UI*, aber *keinen Code*

Arten von Komponenten

ARTEN VON KOMPONENTEN

Client-Komponenten (wie bisher)

- Werden auf dem Client gerendert

BEER ADVISOR

Barfüßer 3.80 EUR



where to buy:
FaucibusLLP | Varius Ultrices Company |
Risus

what customers say:
waldemar vasu: „exceptional!“ ★★★★★
karl marx: „awwwesome!“ ★★★★★
alessa bradley: „can i order another
please?“ ★★★★★

...and what do *you* think?

Your name:

Your rating
(1-5):

Your comment:

ARTEN VON KOMPONENTEN

Client-Komponenten (wie bisher)

- Werden auf dem Client gerendert
- oder auf dem Server 😬

BEER ADVISOR

Barfüßer 3.80 EUR



where to buy:
FaucibusLLP | Varius Ultrices Company | Risus

what customers say:
waldemar vasu: „exceptional!“ ★★★★★
karl marx: „awwwesome!“ ★★★★★
alessa bradley: „can i order another please?“ ★★★★★

...and what do *you* think?

Your name:

Your rating (1-5):

Your comment:

ARTEN VON KOMPONENTEN

Client-Komponenten (wie bisher)

- Werden auf dem Client gerendert
- oder auf dem Server 😬

Wie bisher:

- JavaScript-Code immer zum Client gesendet
- Können deshalb interaktiv sein

BEER ADVISOR

Barfüßer 3.80 EUR



where to buy:
FaucibusLLP | Varius Ultrices Company |
Risus

what customers say:
waldemar vasu: „exceptional!“ ★★★★★
karl marx: „awwwesome!“ ★★★★★
alessa bradley: „can i order another
please?“ ★★★★★

...and what *you* think?

Your name:

Your rating
(1-5):

Your comment:

Neu: Server-Komponenten

- werden auf dem Server gerendert

Neu: Server-Komponenten

- werden auf dem Server gerendert
- oder im Build 🤔

Neu: Server-Komponenten

- werden auf dem Server gerendert
- oder im Build 🤖
- liefern UI (!) zum React-Client zurück (kein JavaScript-Code)

ARTEN VON KOMPONENTEN

Komponenten können gemischt werden

BEER ADVISOR

Barfüßer 3.80 EUR



where to buy:
FaucibusLLP | Varius Ultrices Company |
Risus

what customers
waldemar said: „exceptional!“ ★★★★★
marx: „awwwesome!“ ★★★★★
alessa bradley: „can i order another
please?“ ★★★★★

...and what do you think?

Your name:

Your rating (1-5):

Your comment:

server Components

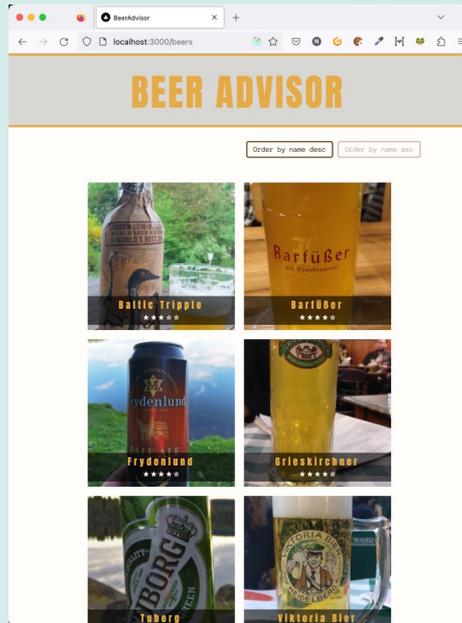
Client Component

RSC am Beispiel

Next.js

Schritt-für-Schritt

- Beispiel-Code: <https://github.com/nilshartmann/nextjs-step-by-step>



Schritt 1: Eine Server Komponente



Demo

- `/page.tsx` zeigen
- `console.log` in Page-Komponente
 - auf dem Server
 - im Browser

Data

Fetching

Idee

- Komponenten, die Daten laden, können das direkt *in der Komponente* tun

SERVER COMPONENTS

Idee

- Komponenten, die Daten laden, können das direkt *in der Komponente* tun
- Server Components können die Server-Infrastruktur nutzen (DB, Filesystem)

SERVER COMPONENTS

Idee

- Komponenten, die Daten laden, können das direkt *in der Komponente* tun
- Server Components können die Server-Infrastruktur nutzen (DB, Filesystem)

👉 Server-Komponenten können dazu asynchron sein

Schritt 2: Eine asynchrone Server-Komponente



Demo

- BeerListPage anlegen
- DB-Zugriff mit loadBeers
 - loadBeers zeigen
- BeerImageList verwenden, um Beers anzuzeigen
- 🔍 **statische Komponenten bislang! (Build!)**

Schritt 3: Eine asynchrone Server-Komponente, die träge ist



Demo

- `beers/[beerId]` Beer-Page aus `material/beer-details-page.txt` einfügen
- Aufruf künstlich verzögern (`sleep` in `loadBeer`)
- `loading.tsx`

Schritt 3: Eine asynchrone Server-Komponente, die zwei Daten Quellen benötigt



Demo

- `beers/[beerId]` Beer-Page wieder schnell machen (slow entfernen)
- `beers/[beerId]` Beer-Page shops erweitern (fertiges fetch in db-queries)
- Zeigen: Promise an Unterkomponente (Shops)
 - -> Parallel fetching!
- Aufruf künstlich verzögern (slow=2400)
- 😞 Jetzt wartet die ganze Seite auf die Shops...
- Suspense um WhereToBuy

Aufteilung

in

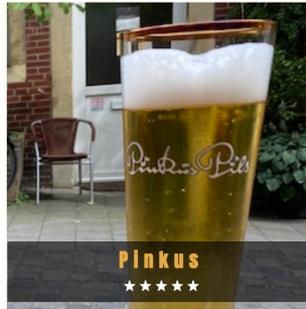
Server-Client:

Konsequenzen

BEER ADVISOR

Order by name desc

Order by name asc



BEISPIEL: ÄNDERN DER SORTIERUNG

```
type BeerListProps = {
  beers: SingleBeer[];
  onToggleOrder(): void;
};

export default function BeerList({ beers, onToggleOrder }: BeerListProps) {
  return (
    <div>
      <h1>Beers</h1>

      <ul>
        {beers.map((b) => (
          <li key={b.id}>{b.name}</li>
        ))}
      </ul>

      <button onClick={onToggleOrder}>Toggle Order</button>
    </div>
  );
}
```

CAN YOU SPOT THE PROBLEM?



```
<button onClick={onToggleOrder}>Toggle Order</button>
```

- **error** Error: Event handlers cannot be passed to Client Component props.

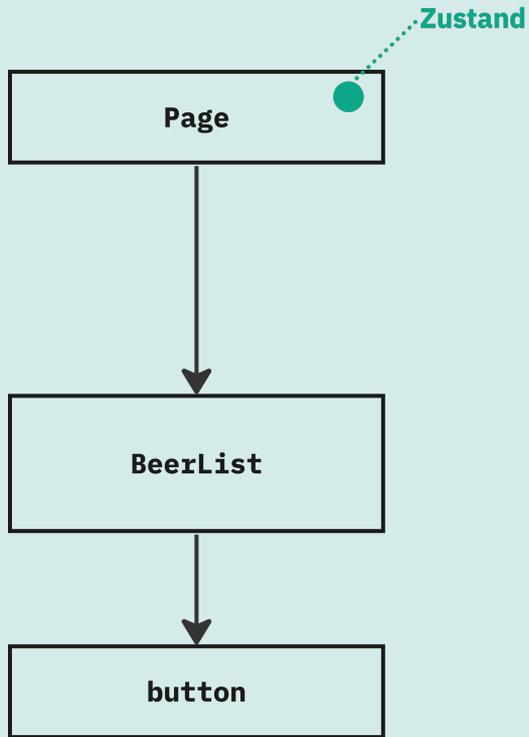
```
<button onClick={function} children=...>  
      ^^^^^^^^^^^
```

If you need interactivity, consider converting part of this to a Client Component.
at stringify (<anonymous>)

CAN YOU SPOT THE PROBLEM?

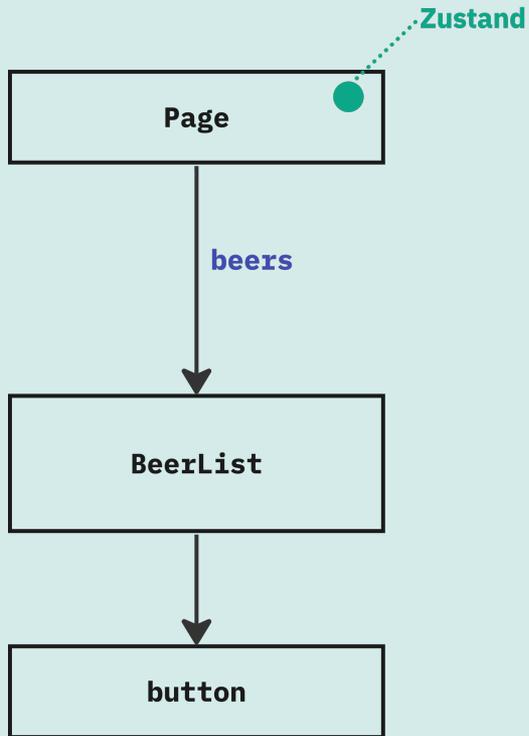
EINE REACT ANWENDUNG IM BROWSER

- State befindet sich oben



Eine "normale" React-Anwendung...

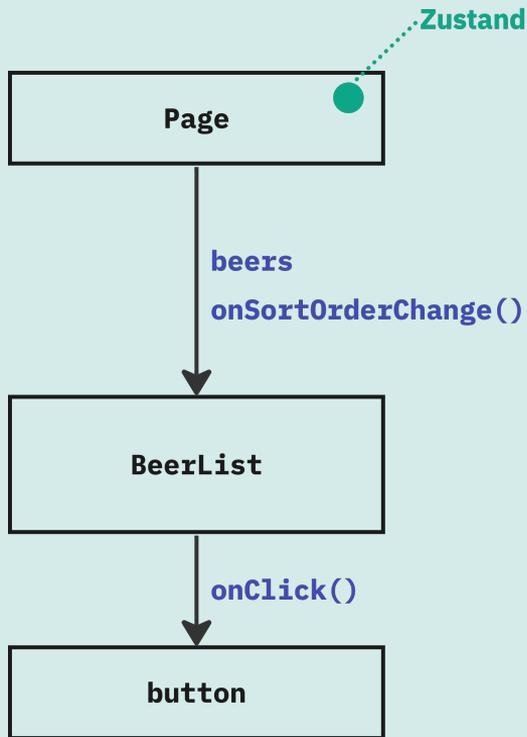
EINE REACT ANWENDUNG IM BROWSER



- State befindet sich oben
- Daten werden runtergereicht ("props")

Eine "normale" React-Anwendung...

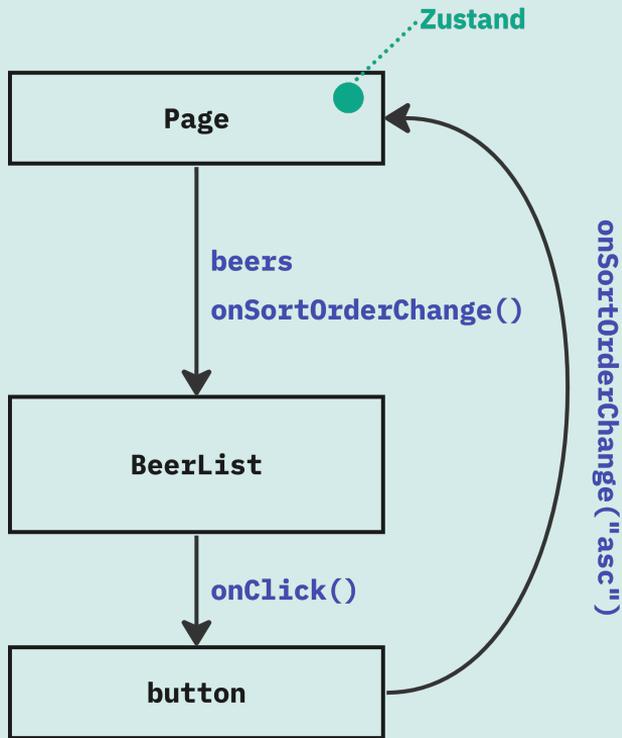
EINE REACT ANWENDUNG IM BROWSER



- State befindet sich oben
- Daten werden runtergereicht ("props")
- Callbacks werden runtergereicht

Eine "normale" React-Anwendung...

EINE REACT ANWENDUNG IM BROWSER



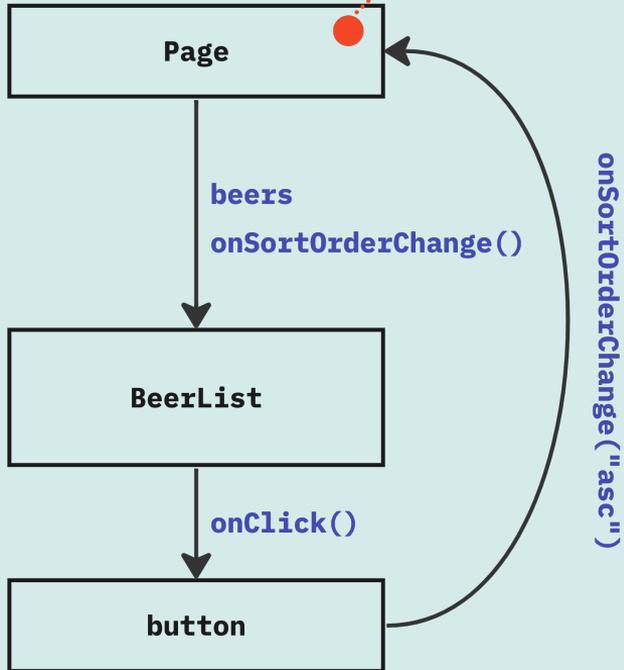
- State befindet sich oben
- Daten werden runtergereicht ("props")
- Callbacks werden runtergereicht
- Über Callbacks kann State-Veränderung ausgelöst werden

Eine "normale" React-Anwendung...

...UND AUF DEM SERVER

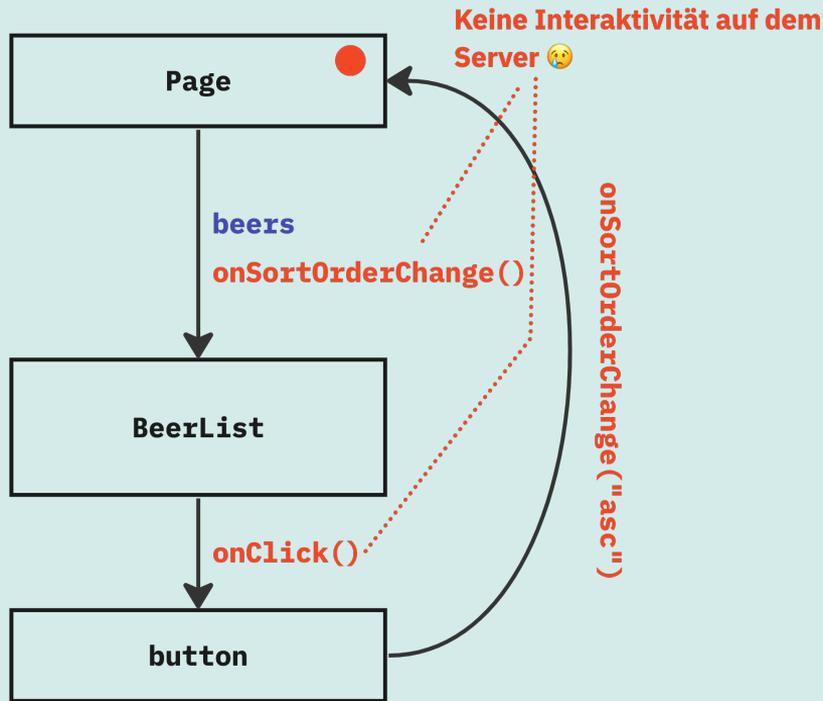
Kein Zustand auf dem Server 🤔

- Auf dem Server gibt es keinen State!



Mit Next.js sind wir aber auf dem Server (by Default)

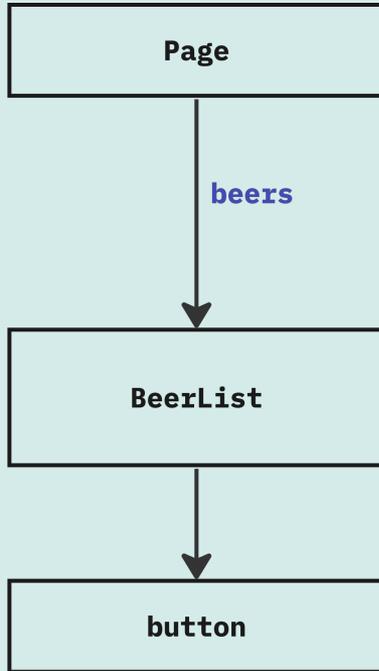
...UND AUF DEM SERVER



- Auf dem Server gibt es keinen State!
- ...und keine Interaktion

Mit Next.js sind wir aber auf dem Server (by Default)

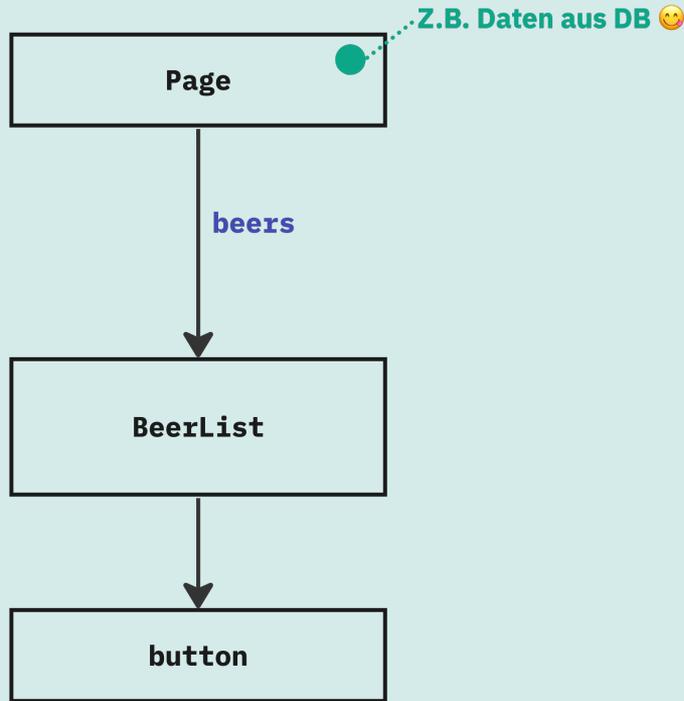
...UND AUF DEM SERVER



- Auf dem Server gibt es keinen State!
- ...und keine Interaktion
- Wir haben nur statischen Content

Mit Next.js sind wir aber auf dem Server (by Default)

...UND AUF DEM SERVER

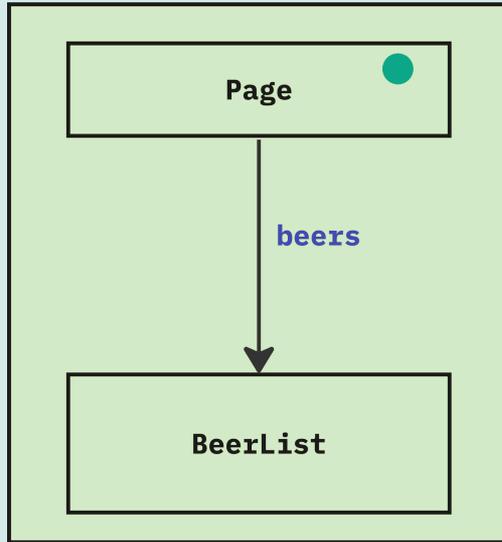


- Auf dem Server gibt es keinen State!
- ...und keine Interaktion
- Wir haben nur statischen Content
- Wir haben aber **Daten**
z.B. aus DB, Microservice, Filesystem...

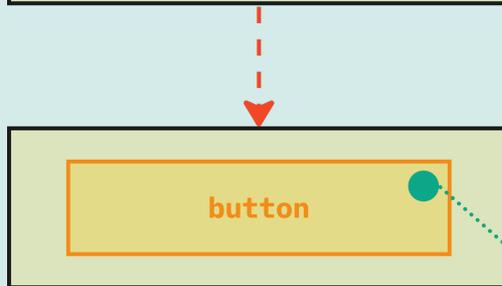
Mit Next.js sind wir aber auf dem Server (by Default)

...UND AUF DEM SERVER

Server



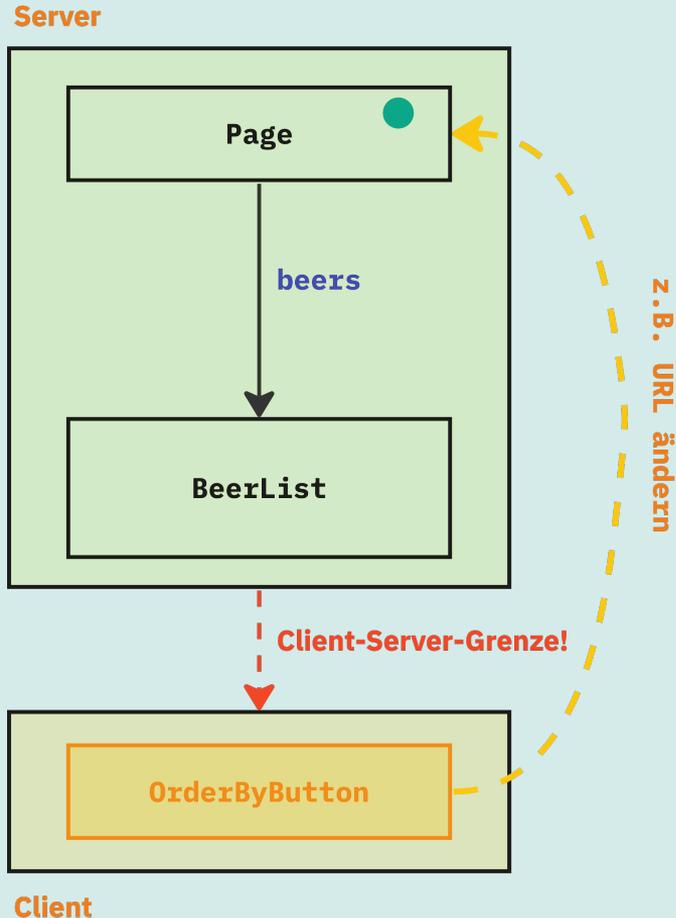
- Bestimmte Teile **müssen** auf den Client
 - z.B. Event-Handler



Client

Interaktives muss auf den Client 🤖

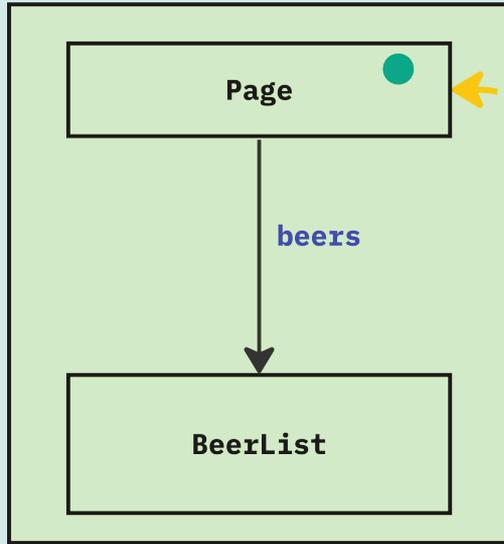
...UND AUF DEM SERVER



- Properties müssen Client-Server-Grenze überwinden
- Müssen serialisierbare Daten sein
- Keine (Callback-)Funktionen!
- Zur Kommunikation müssen **Server-Requests** durchgeführt werden
 - z.B. URL ändern

...UND AUF DEM SERVER

Server



Client-Server-Grenze!



Client

```
"use client";
```

```
function OrderByButton({orderBy}) {
```

```
  const updateUrl = () => { ... }
```

```
  return (
```

```
    <Button
```

```
      onClick={updateUrl}
```

```
    >
```

```
      Order by name {orderBy}
```

```
    </Button>
```

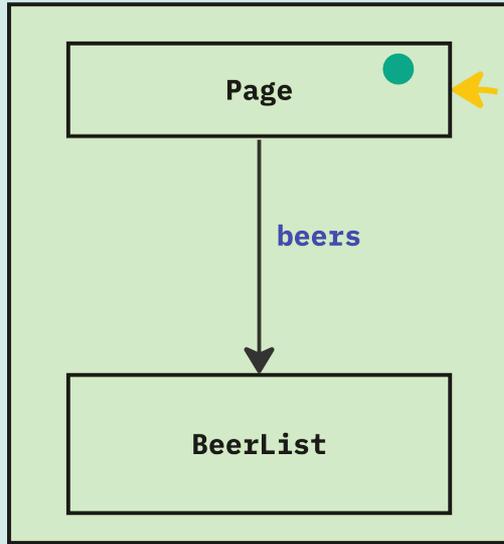
```
  )
```

```
}
```

z.B. URL ändern

...UND AUF DEM SERVER

Server



Client-Server-Grenze!

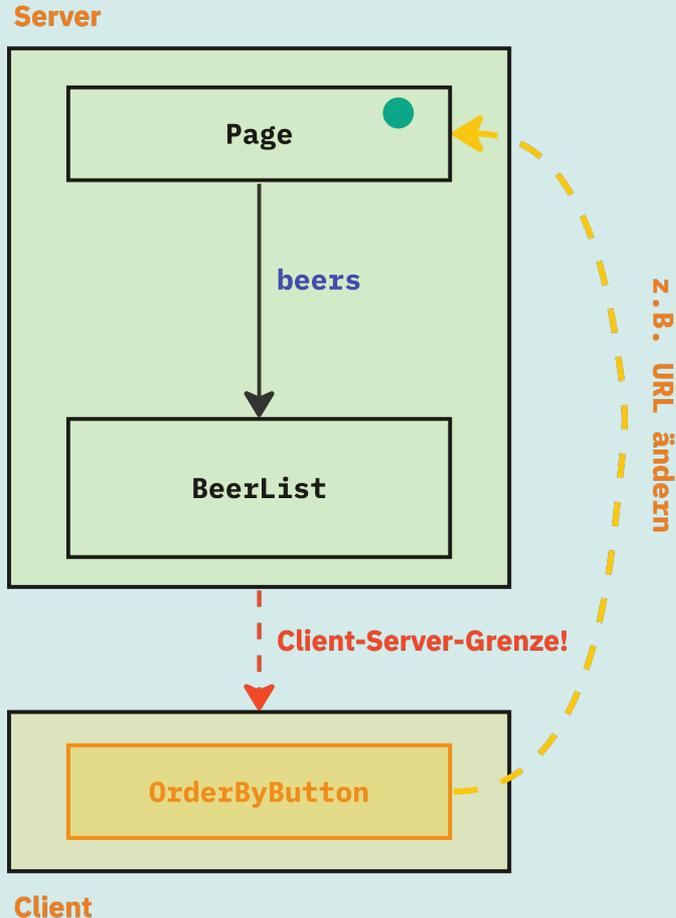


Client

z.B. URL ändern

```
export function BeerListPage() {  
  
  const beers = loadBeers();  
  // ...  
  
  return (  
    <>  
    <OrderByButton orderBy="asc" />  
    <OrderByButton orderBy="desc" />  
  
    <BeerList beers={beers} />  
    </>  
  );  
}
```

...UND AUF DEM SERVER



• Konsequenz

- Code wird durch URL-Handling komplexer?
- Wo ziehen wir Server/Client-Grenze?
 - Button? Ganzes Formular?
 - Hier werden sich Patterns entwickeln
- Ganze Seite (oder Teile) werden neu gerendert
- Fertiges UI kommt dafür vom Server
 - Das kann mehr Daten als bei (REST-)API-Call bedeuten!

Mutations

MUTATIONS

Verändern von Daten: Hinzufügen einer Bewertung

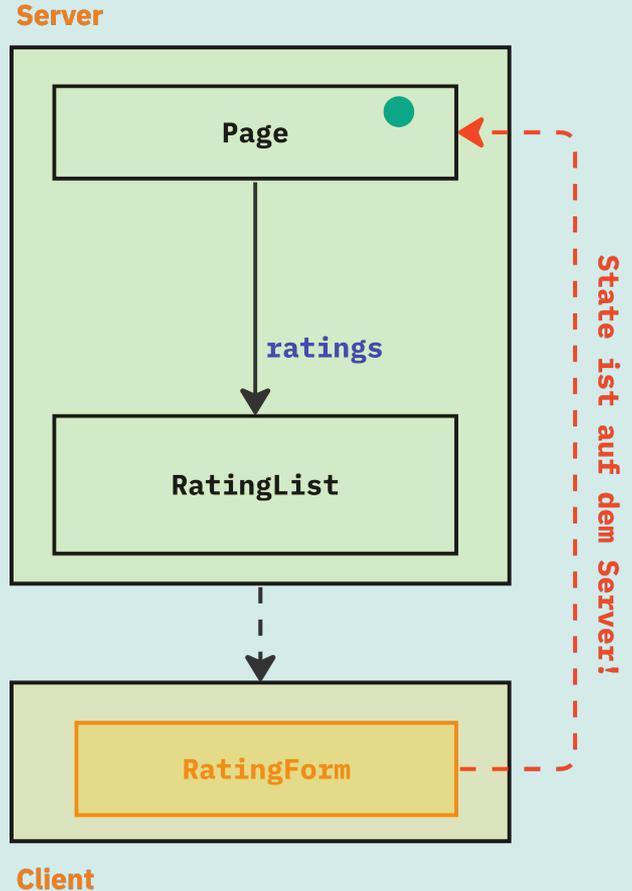
Server-Komponente
Client-Komponente

The screenshot shows a web browser window with the URL `localhost:3000/beers/B1`. The page title is "BEER ADVISOR". The main content area is titled "Barfüßer 3,80 EUR" and includes a "Page" link. A photo of a beer glass is shown on the left. The "where to buy:" section lists "FaucibusLLP | Varius Ultrices Company | Risus". The "what customers say:" section is titled "RatingList" and contains three reviews: "waldemar vasu: „exceptional!“ ★★★★★", "karl marx: „awwwesome!“ ★★★★★", and "alessa bradley: „can i order another please?“ ★★★★★". A "RatingForm" is highlighted with a red dashed box, containing the following fields: "Your name:" with a text input, "Your rating (1-5):" with a dropdown menu, and "Your comment:" with a text area. A "Leave rating for Barfüßer" button is located at the bottom of the form.

MUTATIONS

Verändern von Daten

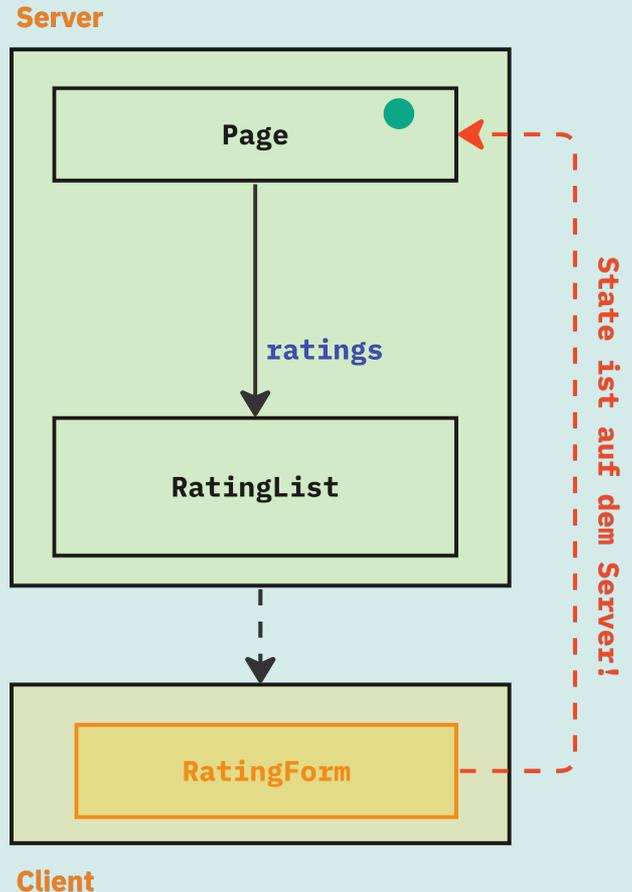
- Nach dem Verändern von Daten muss die UI aktualisiert werden
- Mangels State auf dem Client geht das aber nicht wie bislang



MUTATIONS

Verändern von Daten

- Nach dem Verändern von Daten muss die UI aktualisiert werden
- Mangels State auf dem Client geht das aber nicht wie bislang
- Der **Server** muss nach Datenänderungen **aktualisierte UI** liefern

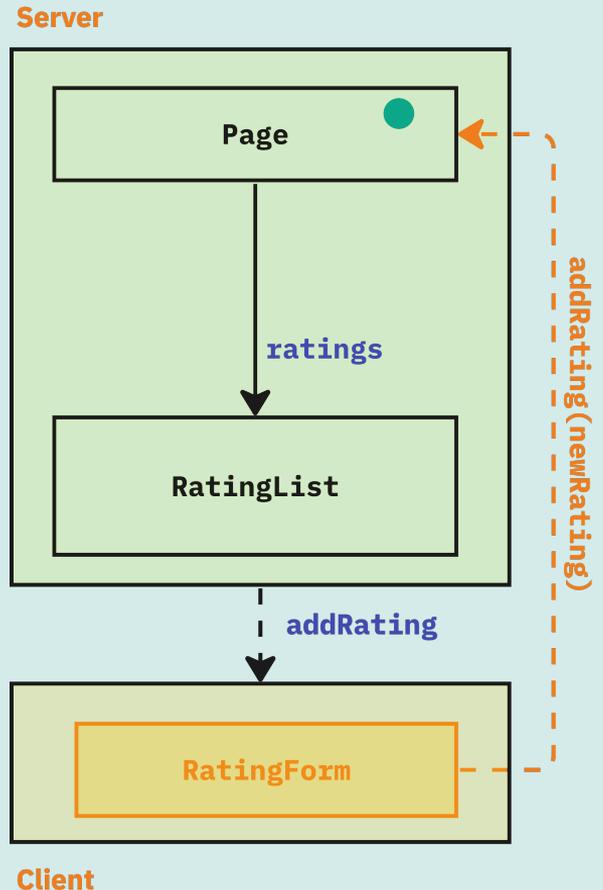


Server Actions

MUTATIONS

Server Actions

- Neues "canary" React-Feature
- Remote Funktion, die aus einer Komponente aufgerufen werden kann



Server Actions

Demo

- AddRatingForm in BeerDetailPage einfügen
- AddRatingForm:
 - - action `saveNewRating` hinzufügen
- Netzwerkverkehr: Antwort vom Server, keine UI
- in `saveNewRating` invalidateRoute machen
- **? Menge an Daten?**



Fullstack Architekturen

Das Ende der Single-Page-Anwendungen?

Fullstack Architekturen

Nein!

Das Ende der Single-Page-Anwendungen?

Fullstack Architekturen

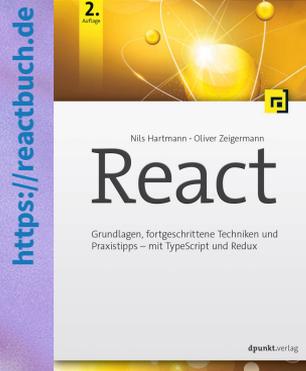
Nein!

Das Ende der Single-Page-Anwendungen?

...es kommt drauf an, was ihr baut!

NILS HARTMANN

<https://nilshartmann.net>



Vielen Dank!

Slides: <https://react.schule/wjax2023>

Source-Code: <https://github.com/nilshartmann/nextjs-beeradvisor>

Fragen & Kontakt: nils@nilshartmann.net

Twitter: [@nilshartmann](https://twitter.com/nilshartmann)