



NILS HARTMANN

<https://nilshartmann.net>

Das  
**Backend**  
im **Frontend**  
**Next.js**  
für **Java-Entwickler:innen**

W-JAX MÜNCHEN | 5. NOVEMBER 2025 | [HTTPS://REACT.SCHULE](https://react.schule)

# NILS HARTMANN

nils@nilshartmann.net

**Freiberuflicher Software-Entwickler, –Architekt, Coach, Trainer**

**Java, Spring, GraphQL, React, TypeScript**



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



<https://reactbuch.de>

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

# EIN BEISPIEL...

http://localhost:8110

Recipify Next.js demo

localhost:8110/recipes


Don't miss latest news. Subscribe to newsletter

E-Mail

Subscribe

Recipify

+ Newest dishes to impress at you



### French Onion Soup

Rich and flavorful onion soup topped with cheesy toast. Making everyday cooking fun! A delight to the taste buds.


5

1 hour

Mexican

Vegan

Medium



### Stuffed Bell Peppers

Bell peppers filled with flavorful ground meat. Made with love and fresh ingredients. For the love of cooking and eating.


65

1 hour

Vegan

Vegetarian

Medium



### Classic Caesar Salad

Crispy romaine lettuce with creamy Caesar dressing. Bon Appétit! Food that feels like home.

92

10 minutes

Asian

Vegan

Easy

<<

<

1

2

3

4

5

6

>

>>

Dummy content only. Do not cook 🍷.

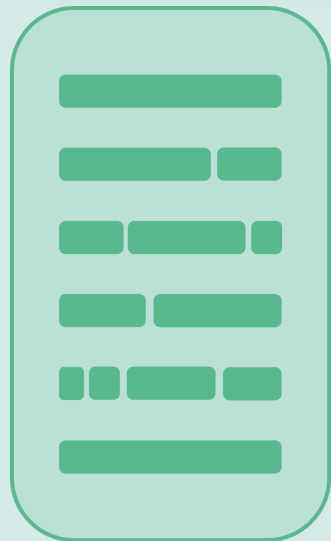
https://nilshartmann.net

### Ein Beispiel...

- Brauchen wir dafür JavaScript im Browser? React?

## *Architekturen für Webanwendungen*

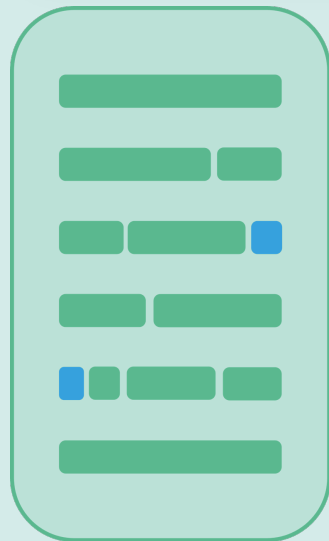
## "Klassisch": Serverseitiges Rendern



HTML-Seite  
mit  
JavaScript

- Server liefert fertige **HTML-Seite** zum Client
- Für jede Interaktion dann Server-Roundtrip (Link, Formular)
- Typische Vertreter: PHP, Java, dotNET, ...

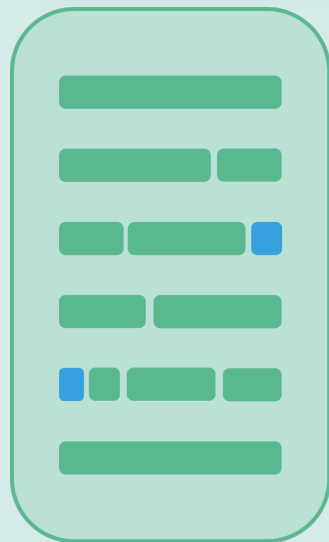
## *"Klassisch": Serverseitiges Rendern plus JavaScript "Schnipsel"*



HTML-Seite  
mit  
JavaScript

- Für feingranulare Interaktion wird **JavaScript** eingestreut

### "Klassisch": Serverseitiges Rendern plus JavaScript "Schnipsel"



HTML-Seite  
mit  
JavaScript

- Für feingranulare Interaktion wird **JavaScript** eingestreut
- Eigentlich optimal:
  - wir haben **JavaScript** nur da, wo wir es **wirklich** brauchen (für Interaktivität)
  - alles andere kann statisches HTML und CSS sein ❤️



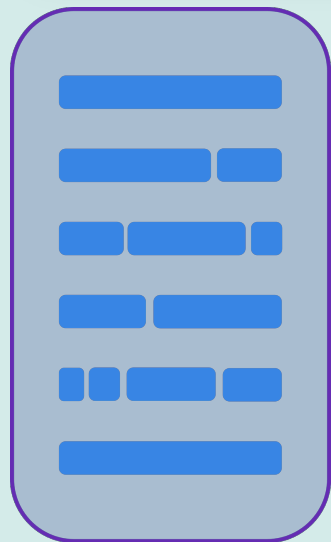
## *Konsequenz Serverseitiges Rendern plus JavaScript "Schnipsel"*



HTML-Seite

- Bunter Strauß an Server- und Client-Technologien (Backend-Sprache, Template-Sprache, JavaScript)

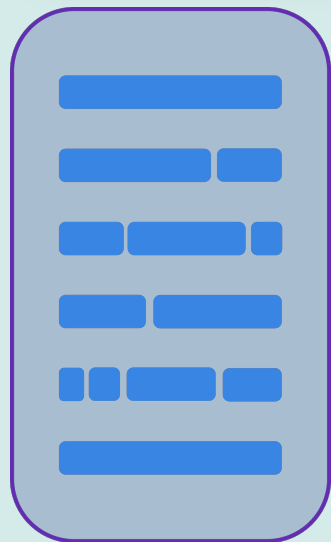
## Single-Page-Anwendungen



JavaScript  
im  
Browser

- Vollständig in **JavaScript** implementiert
- Ausgeführt komplett **im Browser**
- Statisches HTML spielt (fast) keine Rolle

## Konsequenz Single-Page-Anwendungen

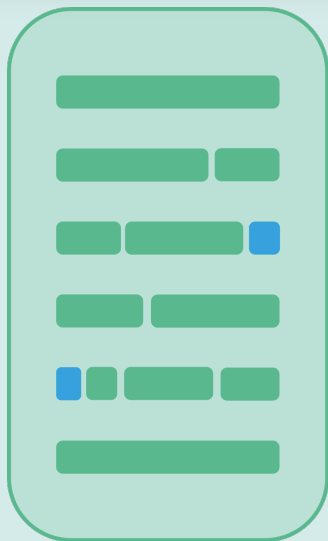


JavaScript  
im  
Browser

- Sehr viel **JavaScript zur Laufzeit**
- Auch für **statische** Inhalte
- Das JavaScript muss laufen, bevor etwas dargestellt wird
- ❌ Die Anwendung funktioniert nicht ohne JavaScript

***„Fullstack“ mit Next.js***

## „Fullstack“ mit Next.js

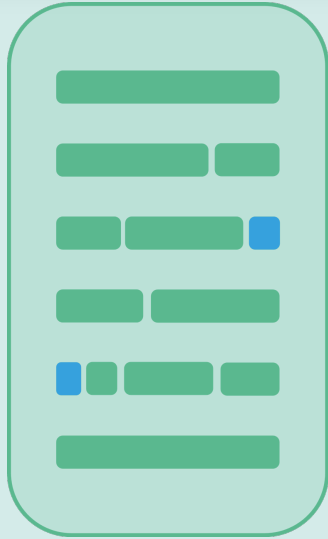


Klassisch  
(im Browser: HTML+**JS**)

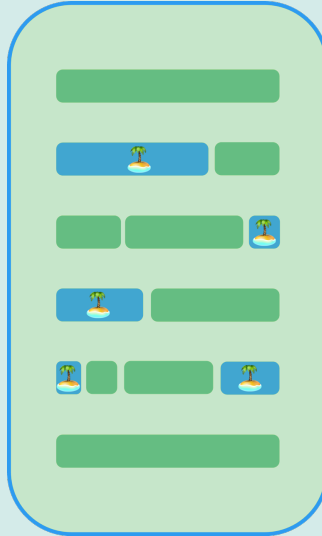


Single-Page-App  
(im Browser: nur **JS**)

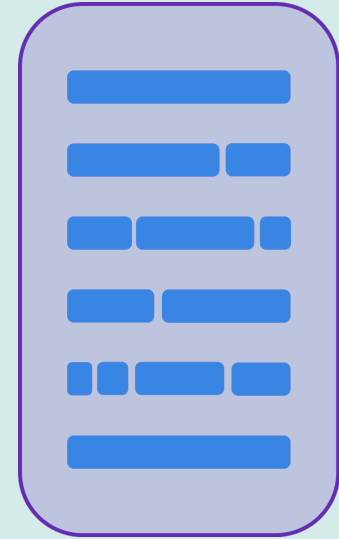
## „Fullstack“ mit Next.js



Klassisch  
(im Browser: HTML+**JS**)



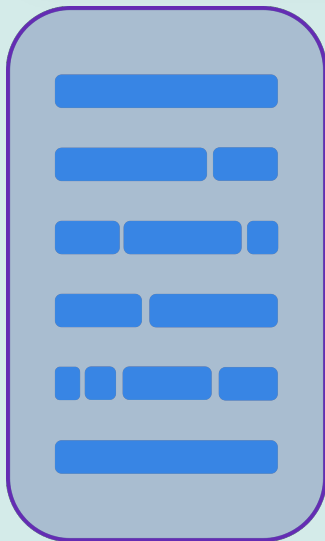
Next.js-Anwendung  
(im Browser: HTML+**JS**)



Single-Page-App  
(im Browser: nur **JS**)

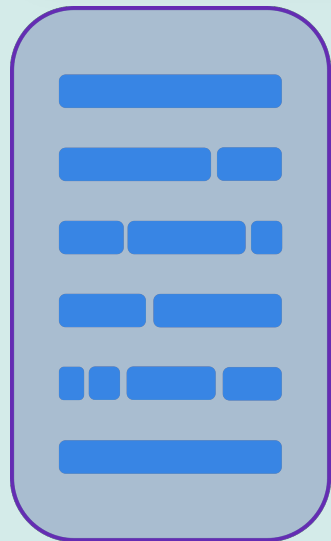
## Next.js-Anwendungen

- Vollständig in **JavaScript** entwickelt



JavaScript  
auf dem Server

## Next.js-Anwendungen

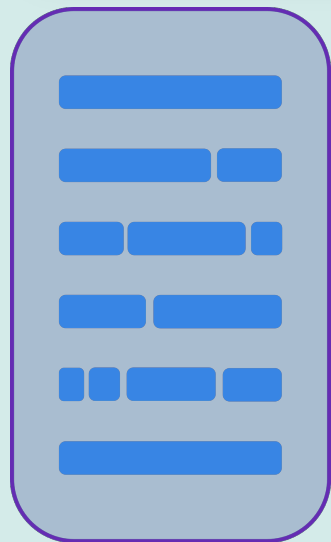


JavaScript  
auf dem Server

- Vollständig in **JavaScript** entwickelt
- Erzeugt **HTML** auf dem Server



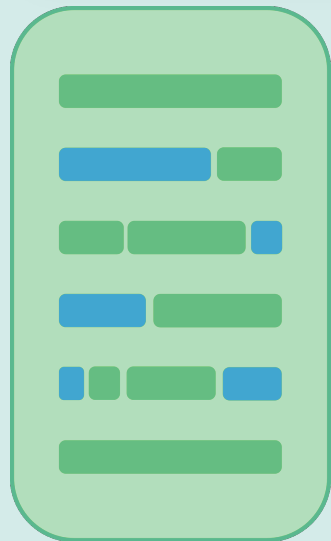
## Next.js-Anwendungen



JavaScript  
auf dem Server

- Vollständig in **JavaScript** entwickelt
- Erzeugt **HTML** auf dem Server
- Nur der **JavaScript-Code** für **Interaktionen** wird zum **Browser** geschickt

## Next.js-Anwendungen



im Browser:  
HTML+JS

- Wir sind zurück zur **JavaScript-Schnipsel**-Architektur
  - aber: die Schnipsel werden von Next.js erzeugt
  - die Schnipsel existieren nur zur **Laufzeit**
  - In der **Entwicklung** eine Code-Basis



Bestes aus beiden Welten?

<https://nextjs.org>

<https://nextjs.org>

- basiert auf React

<https://nextjs.org>

- basiert auf React
- fügt serverseitiges Rendern, Caching und Routing hinzu

<https://nextjs.org>

- basiert auf React
- fügt serverseitiges Rendern, Caching und Routing hinzu
- ähnlich wie Spring Boot, das z.B. Webserver + JSP/Thymeleaf integriert

<https://nextjs.org>

- basiert auf React
- fügt serverseitiges Rendern, Caching und Routing hinzu
- ähnlich wie Spring Boot, das z.B. Webserver + JSP/Thymeleaf integriert
- Typische Einsatzgebiete:
  - Eher Webseiten als Webanwendungen
  - Zum Beispiel eCommerce, Nachrichten- und Produktseiten

# Live

# Demo



[HTTP://LOCALHOST:8100](http://localhost:8100)



**NILS HARTMANN**

<https://nilshartmann.net>

# Vielen Dank!

Code: <https://react.schule/wjax2025-nextjs>

Fragen und Kontakt:

[nils@nilshartmann.net](mailto:nils@nilshartmann.net)

<https://nilshartmann.net/kontakt>

