

NILS HARTMANN

<https://nilshartmann.net>

React

Blick zurück und nach vorne

W-JAX | MÜNCHEN, 7. NOVEMBER 2024 | @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)

Donnerstag, 07. November 2024
12:00 - 13:00

Raum:
Sydney

Beschreibung

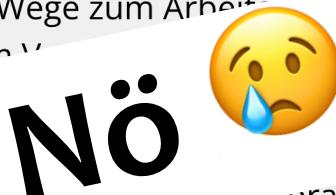
Das zurückliegende Jahr 2024 hat uns nach langer Zeit wieder ein neues React-Release beschert: Die Version 19 mit neuen Hooks, neuen Komponenten und neuen APIs. Dazu kamen interessante Neuerungen im React-Ökosystem, Fullstack-Ansätze, typsicheres Routing und neue Wege zum Arbeiten mit asynchronem Code und serverseitigen Daten. In diesem Vortrag möchte ich einen Blick zurückwerfen und euch die wichtigsten Neuerungen vorstellen, sodass ihr einen Überblick bekommt, welche neuen Möglichkeiten React für uns bereithält, um effizient Anwendungen zu bauen. Dabei werde ich euch mit viel Live Coding praktische Beispiele zeigen, sodass ihr einen realistischen Eindruck bekommt, wie sich zeitgemäßer React Code 2024 anfühlt bzw. 2025 möglicherweise anfühlen wird.

Donnerstag, 07. November 2024
12:00 - 13:00

Raum:
Sydney

Beschreibung

Das zurückliegende Jahr 2024 hat uns nach langer Zeit wieder ein neues React-Release beschert: Die Version 19 mit neuen Hooks, neuen Komponenten und neuen APIs. Dazu kamen interessante Neuerungen im React-Ökosystem, Fullstack-Ansätze, typsicheres Routing und neue Wege zum Arbeitsergebnis. In diesem Vortrag werde ich euch die wichtigsten Neuerungen vorstellen, welche neuen Möglichkeiten React bietet, um effizient Anwendungen zu bauen. Dabei werde ich euch nicht nur theoretische Erklärungen geben, sondern praktische Beispiele zeigen, sodass ihr einen realistischen Eindruck bekommt, wie sich zeitgemäßer React Code 2024 anfühlt bzw. 2025 möglicherweise anfühlen wird.



Release Candidate (seit April!)

BLOG >

React 19 RC

April 25, 2024 by [The React Team](#)

React 19 RC is now available on npm!

- Ihr könnt die Version aber schon installieren und verwenden
 - Siehe dazu: <https://react.dev/blog/2024/04/25/react-19-upgrade-guide>
 - Doku: <https://19.react.dev/>

Grund für die Verzögerung: Das "Suspense Drama"

- ...dazu später mehr!

The screenshot shows a web browser window with the following details:

- Title Bar:** React 19 and Suspense - A Drama in 3 Acts
- URL:** https://tkdodo.eu/blog/react-19-and-suspense-a-drama-in-3-acts
- Page Title:** TkDodo's blog
- Page Content:** React 19 and Suspense - A Drama in 3 Acts
- Post Date:** 16.06.2024
- Tags:** ReactJs, React Query, Suspense, JavaScript
- Read Time:** 7 min read
- Image:** A dark, moody photograph of a person's silhouette against a bright, cloudy sky.
- Caption:** Photo by Jr Korpa

React Versionen

- **latest**: die aktuelle stabile Version (zurzeit 18.3)
- **canary**: "Stabil", zur Integration/Anpassung für Library Autoren
- **experimental**: Neue, noch nicht fertige Features
- **rc**: entspricht aktuell dem canary-Release
- Installation mit dem entsprechenden Version-Tag:
 - `pnpm add --save-exact react@canary react-dom@canary`

Für die Migration: React 18.3

- Als Vorbereitung auf Version 19 könnt ihr React **18.3** installieren
- Darin sind keine neuen Features
- Aber Warnungen etc. die mögliche React 19-Probleme in eurem Code anzeigen

(ASYNCHRONE) TRANSITIONS

Neu: asynchrone Transitions

- Mit React 19 können mit `useTransition` **asynchrone** Funktionen verwendet werden

(ASYNCHRONE) TRANSITIONS

Neu: asynchrone Transitions

- Mit React 19 können mit `useTransition` **asynchrone** Funktionen verwendet werden
- Damit kann man typische Muster beim Arbeiten mit asynchronen Daten vereinfachen

(ASYNCHRONE) TRANSITIONS

Neu: asynchrone Transitions

- Mit React 19 können mit `useTransition` **asynchrone** Funktionen verwendet werden
- Damit kann man typische Muster beim Arbeiten mit asynchronen Daten vereinfachen
- Fehlerbehandlung erfolgt dann zum Beispiel über Error Boundaries

OPTIMISTISCHE UPDATES

Neu: Optimistische Updates

- Mit useOptimistic kann ein "optimistischer" State gesetzt werden

OPTIMISTISCHE UPDATES

Neu: Optimistische Updates

- Mit `useOptimistic` kann ein "optimistischer" State gesetzt werden
- Damit kann man das vermutete Ergebnis einer Änderung frühzeitig darstellen

OPTIMISTISCHE UPDATES

Neu: Optimistische Updates

- Mit `useOptimistic` kann ein "optimistischer" State gesetzt werden
- Damit kann man das vermutete Ergebnis einer Änderung frühzeitig darstellen
- Dieser State ist so lange gültig, wie eine Transition läuft

Demo: Transitions und useOptimistic



- Standalone / TransitionApp
 - Transition
 - Error Boundary
 - useOptimistic

React

Compiler BETA

React Compiler: Verbesserte Performance durch weniger Rendern

- Separates Projekt, nicht Teil von React 19
- Könnt ihr mit React ab Version 17 verwenden

React Compiler: Verbesserte Performance durch weniger Rendern

- Separates Projekt, nicht Teil von React 19
- Könnt ihr mit React ab Version 17 verwenden
- Fügt Memoisierungen in euren Code ein

React Compiler: Verbesserte Performance durch weniger Rendern

- Separates Projekt, nicht Teil von React 19
- Könnt ihr mit React ab Version 17 verwenden
- Fügt Memoisierungen in euren Code ein
- Ergebnis dann ähnlich wie React.memo, useMemo, useCallback
 - Technisch über das Analysieren und Verfolgen von Abhängigkeiten

React Compiler: Verbesserte Performance durch weniger Rendern

- Separates Projekt, nicht Teil von React 19
- Könnt ihr mit React ab Version 17 verwenden
- Fügt Memoisierungen in euren Code ein
- Ergebnis dann ähnlich wie React.memo, useMemo, useCallback
 - Technisch über das Analysieren und Verfolgen von Abhängigkeiten
- Das spart Renderzyklen

React Compiler: Verbesserte Performance durch weniger Rendern

- Separates Projekt, nicht Teil von React 19
- Könnt ihr mit React ab Version 17 verwenden
- Fügt Memoisierungen in euren Code ein
- Ergebnis dann ähnlich wie React.memo, useMemo, useCallback
 - Technisch über das Analysieren und Verfolgen von Abhängigkeiten
- Das spart Renderzyklen
- Eure Bundle-Size wird dadurch (etwas) größer

React Compiler: Voraussetzungen

- Ihr müsst die **Rules of React** befolgen

React Compiler: Voraussetzungen

- Ihr müsst die **Rules of React** befolgen
- Bevor ihr den Compiler verwendet, könnt ihr einen "Healthcheck" machen, ob eure Codebasis damit funktioniert

```
npx react-compiler-healthcheck@latest
```

React Compiler: Voraussetzungen

- Ihr müsst die **Rules of React** befolgen
- Bevor ihr den Compiler verwendet, könnt ihr einen "Healthcheck" machen, ob eure Codebasis damit funktioniert
- Es gibt einzelne Bibliotheken, die mit dem Compiler nicht kompatibel sind.

```
npx react-compiler-healthcheck@latest
```

React Compiler: Demo



- Standalone App / IngredientsWidget
 - Renderzyklen analysieren
 - Wie könnten wir optimieren? 🤔
 - Compiler in vite-Konfiguration einschalten
 - Was passiert? 🤔
 - Dev Tools "Auto Memo"

React 19

use
Funktion

USE-FUNKTION

use-Funktion

- Mit der use-Funktion kann auf "Ressourcen" zugegriffen werden:
 - Contexte
 - Promises

use-Funktion

- Mit der use-Funktion kann auf "Ressourcen" zugegriffen werden:
 - Contexte
 - Promises
- Die Funktion ist **kein Hook!**

use-Funktion

- Mit der use-Funktion kann auf "Ressourcen" zugegriffen werden:
 - Contexte
 - Promises
- Die Funktion ist **kein Hook!**
- Die use-Funktion unterliegt nicht den "Rules of Hook"
 - Kann also z.B. in if-Abfragen verwendet werden

Vereinfachungen beim Context

- createContext gibt jetzt direkt den Provider zurück
 - MyContext.Provider damit überflüssig
 - MyContext.Consumer gibt es nicht mehr
- Context kann mit der use-Funktion abgefragt werden

Demo: use-Funktion und Context

-  context/ContextApp
 - Context erzeugen
 - Auf der Konsole das Rendern überprüfen
 - In Price auf use umstellen
 - Auf der Konsole das Rendern überprüfen

use-Funktion mit Promises

- Mit use können wir auch auf Promises zugreifen
- Das Promise muss aber über Renderzyklen hinweg stabil bleiben
- Deswegen muss das in der Regel außerhalb der Komponenten erzeugt werden
- Dazu nimmt man eine Bibliothek, z.B.
 - TanStack Query
 - React Router
 - TanStack Router

TanStack Router

- Vorgestellt 24.12.2023 
- Kernideen:
 - File-basierte Routen
 - Typsicheres Routing
 - Loader zum Laden der Daten (so früh wie möglich)

EXKURS: TANSTACK ROUTER

TanStack Router - Demo

- RouterApp
-  Rezept-Liste zeigen
-  Route-Code für einzelnes Rezept anlegen (\$recipId/index.tsx)
-  RecipeCard Link zum Rezept (Typsicherheit!)
-  tli und loader mit params und Ausgabe RecipId

DIE USE-FUNKTION

Demo: use-Funktion mit Promise vom TanStack Router

- RouterApp
- 🕵️ Route-Code für einzelnes Rezept implementieren
- 🕵️ Suspense um beide Komponenten und einzeln
- 🕵️ ErrorBoundary für not-found

Modernes Data Fetching: TanStack Query

- ehemals React Query

Modernes Data Fetching: TanStack Query

- ehemals React Query
- globaler Cache für alle Daten

Modernes Data Fetching: TanStack Query

- ehemals React Query
- globaler Cache für alle Daten
- Nie mehr useEffect 😴

Modernes Data Fetching: TanStack Query

- ehemals React Query
- globaler Cache für alle Daten
- Nie mehr useEffect 😴
- Unterstützung für Suspense und Fehlerbehandlung

Modernes Data Fetching: TanStack Query

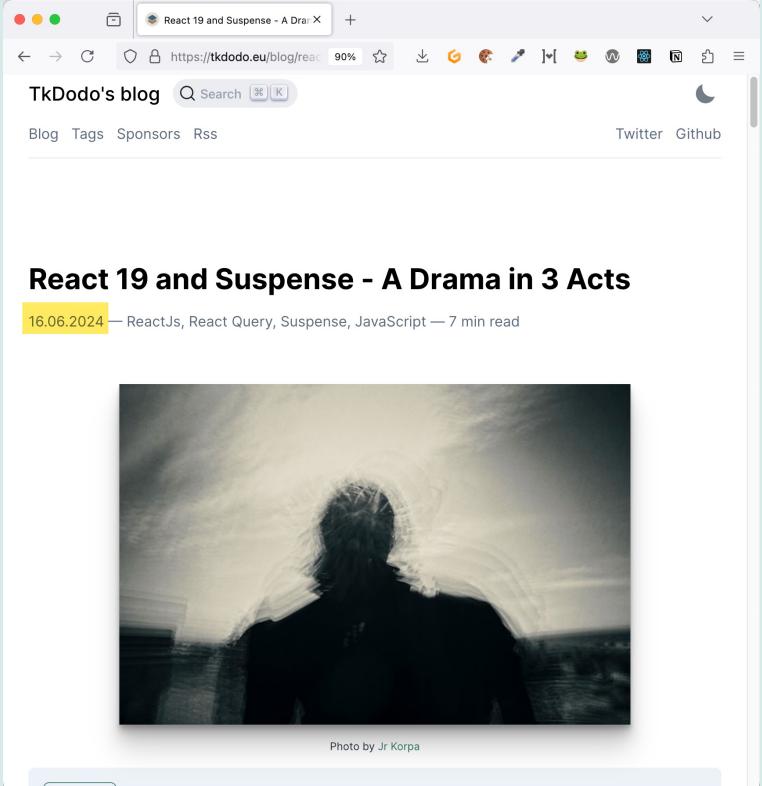
- ehemals React Query
- globaler Cache für alle Daten
- Nie mehr useEffect 😴
- Unterstützung für Suspense und Fehlerbehandlung
- Hohe Typsicherheit

Demo: TanStack Query

- useSuspenseQuery
- ky
- zod
- 🕵️ Query DevTools
- 🗂️ Caching (RouterApp.tsx refetchOnMount = true)

Grund für die Verzögerung: Das "Suspense Drama"

-  Netzwerkanalyse mit zwei Requests



The screenshot shows a web browser window with the title bar 'React 19 and Suspense - A Drama in 3 Acts'. The URL in the address bar is <https://tkdodo.eu/blog/react-19-and-suspense-a-drama-in-3-acts>. The page content includes the heading 'React 19 and Suspense - A Drama in 3 Acts', a date '16.06.2024', and a photo credit 'Photo by Jr Korpa'. The browser interface includes standard navigation buttons, a search bar, and social sharing links for Twitter and GitHub.

React 19 and Suspense - A Drama in 3 Acts

16.06.2024 — ReactJs, React Query, Suspense, JavaScript — 7 min read

Photo by Jr Korpa

Code: TanStack Query - Mutations

- Zwei <Recipe />-Komponenten parallel
- Like Button
- useLikeMutation (use-queries.ts)
 - Aktualisieren des Caches mit invalidateQueries
 - Direktes Aktualisieren des Caches

React 2024/2025

Fazit

React 19

- Schwerpunkt eher auf Fullstack-Themen

React 19

- Schwerpunkt eher auf Fullstack-Themen
- Aber auch für die SPA-Entwicklung gibt es "nette" Features

React 19

- Schwerpunkt eher auf Fullstack-Themen
- Aber auch für die SPA-Entwicklung gibt es "nette" Features
- Insgesamt Fokus auf Performanz bzw. das "Gefühl" von Performance

Ökosystem: Data Fetching

- Immer mehr Typsicherheit, TypeScript und zod sei dank

Ökosystem: Data Fetching

- Immer mehr Typsicherheit, TypeScript und zod sei dank
- TanStack Query ist de-facto-Standard für Data Fetching

Ökosystem: Data Fetching

- Immer mehr Typsicherheit, TypeScript und zod sei dank
- TanStack Query ist de-facto-Standard für Data Fetching
 - (es sei denn, man verwendet Redux oder GraphQL)

Ökosystem: Data Fetching

- Immer mehr Typsicherheit, TypeScript und zod sei dank
- TanStack Query ist de-facto-Standard für Data Fetching
 - (es sei denn, man verwendet Redux oder GraphQL)
- useEffect wird immer weniger benötigt

Ökosystem: Data Fetching

- Immer mehr Typsicherheit, TypeScript und zod sei dank
- TanStack Query ist de-facto-Standard für Data Fetching
 - (es sei denn, man verwendet Redux oder GraphQL)
- useEffect wird immer weniger benötigt
- Man bekommt alle Features, die man für serverseitige Daten benötigt

Ökosystem: Router

- TanStack Router setzt Maßstäbe für Typsicherheit

Ökosystem: Router

- TanStack Router setzt Maßstäbe für Typsicherheit
 - Ob sich TS Router durchsetzt, wird man sehen

Ökosystem: Router

- TanStack Router setzt Maßstäbe für Typsicherheit
 - Ob sich TS Router durchsetzt, wird man sehen
- TanStack Router bietet sehr gute Möglichkeiten, mit Search Params zu arbeiten

Ökosystem: Router

- TanStack Router setzt Maßstäbe für Typsicherheit
 - Ob sich TS Router durchsetzt, wird man sehen
- TanStack Router bietet sehr gute Möglichkeiten, mit Search Params zu arbeiten
 - "State as Search Params"

Ökosystem: Router

- TanStack Router setzt Maßstäbe für Typsicherheit
 - Ob sich TS Router durchsetzt, wird man sehen
- TanStack Router bietet sehr gute Möglichkeiten, mit Search Params zu arbeiten
 - "State as Search Params"
- Ausblick: auch React Router v7 bringt auch mehr Typsicherheit

Ökosystem: Router

- TanStack Router setzt Maßstäbe für Typsicherheit
 - Ob sich TS Router durchsetzt, wird man sehen
- TanStack Router bietet sehr gute Möglichkeiten, mit Search Params zu arbeiten
 - "State as Search Params"
- Ausblick: auch React Router v7 bringt auch mehr Typsicherheit
 - Und serverseitige Features (Zusammenführung mit Remix)

NILS HARTMANN
<https://nilshartmann.net>



vielen Dank!

Slides & Code: <https://react.schule/wjax-2024-react>

Fragen und Kontakt

nils@nilshartmann.net

<https://nilshartmann.net/kontakt>

