

GraphQL mit Prisma 2 & TypeScript

Web Worker Saar Meetup Juni 2020



Sven Hennessen

 [@svenhennessen](https://twitter.com/svenhennessen)

 hennessen.net

GraphQL Definition

GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data. GraphQL provides a complete and understandable description of the data in your API, gives clients the power to ask for exactly what they need and nothing more, makes it easier to evolve APIs over time, and enables powerful developer tools.

— <https://graphql.org>

GraphQL Features

Syntax

- Feldliste
- Interaktiv
- Lists, (Non-)Null, Enums
- Client und Server validieren Typen

Typ-Sicherheit

- Primitive GraphQL Typen: Int, Float, String, Boolean, ID (docs)
- Eigene Typen definierbar
- IDE mit Schema & API Dokumentation
- Referenz-Implementierung GraphiQL, verbreitete Variante GraphQL Playground

```
{
  headers {
    title
    items {
      title
      type
      createdAt
    }
  }
}
```

```
{
  "data": {
    "headers": [
      {
        "title": "test-title",
        "items": [
          {
            "title": "test",
            "type": "B",
            "createdAt": "..."
          }
        ]
      }
    ]
  }
}
```

Prisma 2 Definition

Prisma is an open-source database toolkit. It replaces traditional ORMs and makes database access easy with an auto-generated query builder for TypeScript & Node.js.

— <https://prisma.io>

Prisma 2 Hintergrund

Projekte

- **prisma 1** bis v1.34
 - Scala (JVM)
 - Deployment im eigenen Docker-Container
- **prisma 2** ([GitHub](#))
 - **prisma-client** ([GitHub](#)) + migrate + studio
 - Re-write in Rust (Query Engine) & TypeScript (Client)
 - Deployment als npx Binary mit dem eigenen Projekt

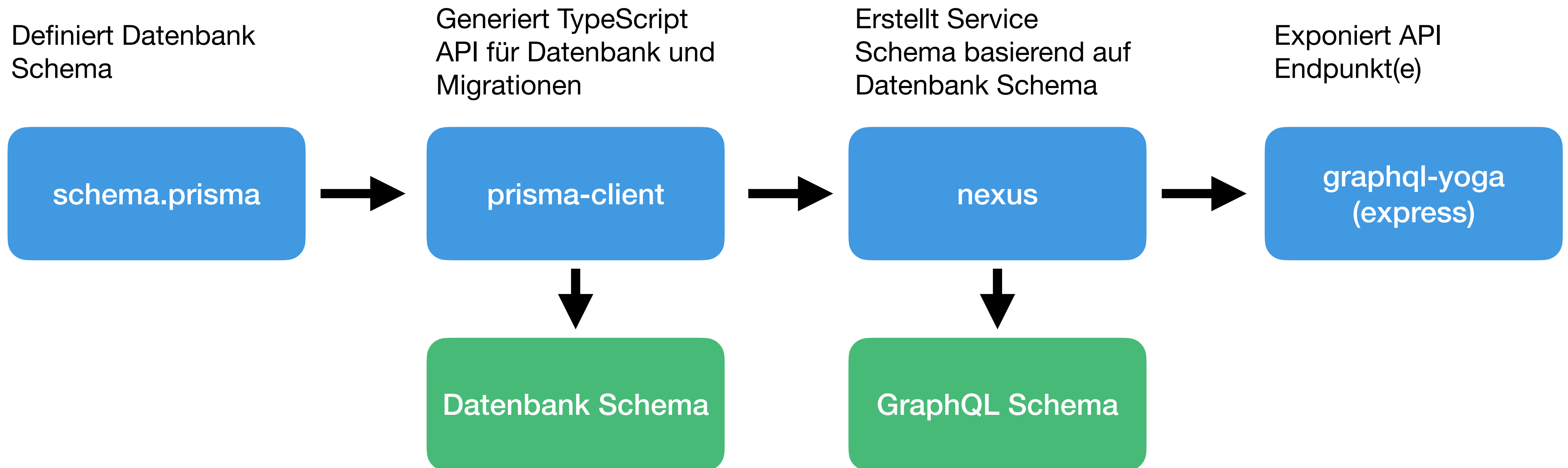
Wer?

- Prisma Labs
- Firma in Berlin/San Francisco
- Investoren-Finanziert, Open Source

Verwendung mit GraphQL

- Typ-Definition mit nexus
- Server mit graphql-yoga
- Prisma 2 in Beta ⚠️ <https://www.isprisma2ready.com/>

Prisma 2 Service Architektur

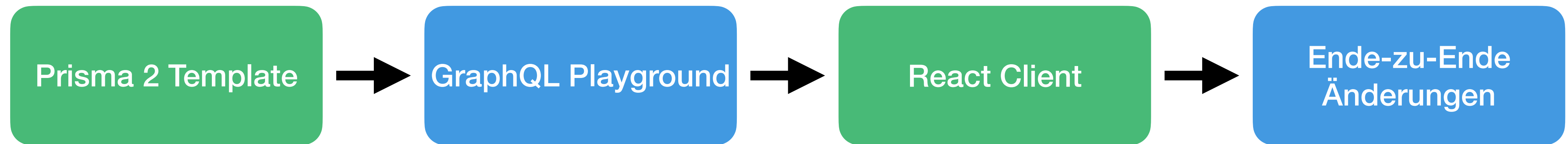


Demo React Client

Basierend auf apollo-client

- Generiert TypeScript auf Basis von verwendeten Queries/Mutations und der Server API
- Bietet React Komponenten und Hooks um API zu konsumieren
 - Apollo React Hooks
 - Apollo React Komponenten

Demo



Kernpunkte

- Weniger fehleranfällig durch Typsicherheit
- Funktioniert besonders gut wenn man Full-Stack entwickelt
- Refactorings werden einfacher
- Schnelles Iterieren als Entwickler mit “echten” Daten
- Gute Testbarkeit durch
 - apollo-client im Frontend und
 - custom resolver im Backend

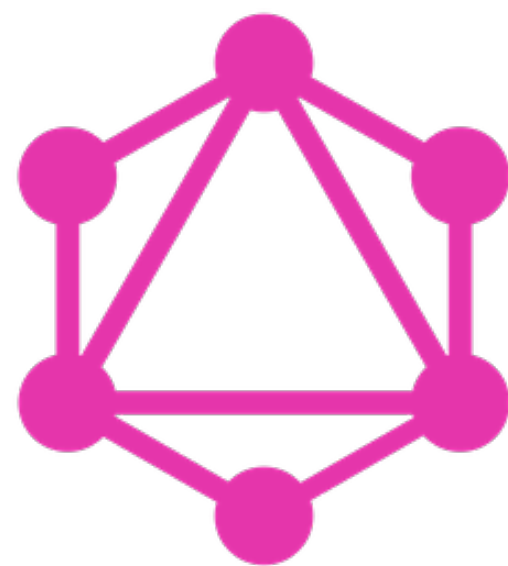
Weitere Themen

- Subscriptions (Web Sockets)
- Schema und Migrations im Produktionsbetrieb
- UI-First Entwicklung
- Introspection
 - API Introspection
 - DB Schema Introspection

Links

- graphql.org
 - [Datentypen](#)
- [GraphiQL](#)
- [GraphQL Playground](#)
- prisma.io
 - [GitHub Projekt](#)
 - [Prisma Client](#)
- [Apollo Client](#)
 - [React Hooks](#)
 - [React Komponenten](#)
- [prisma-2/boilerplate](#)
- [Demo-Script](#)
- Permissions mit [graphql-shield](#)
- [API Introspection und Schema Introspection](#)
- Logos
 - https://commons.wikimedia.org/wiki/File:GraphQL_Logo.svg
 - <https://github.com/prisma/presskit>
 - github.com/remojansen/logo.ts

Fragen?



Danke!



Sven Hennessen

 [@svenhennessen](https://twitter.com/svenhennessen)

 hennessen.net