

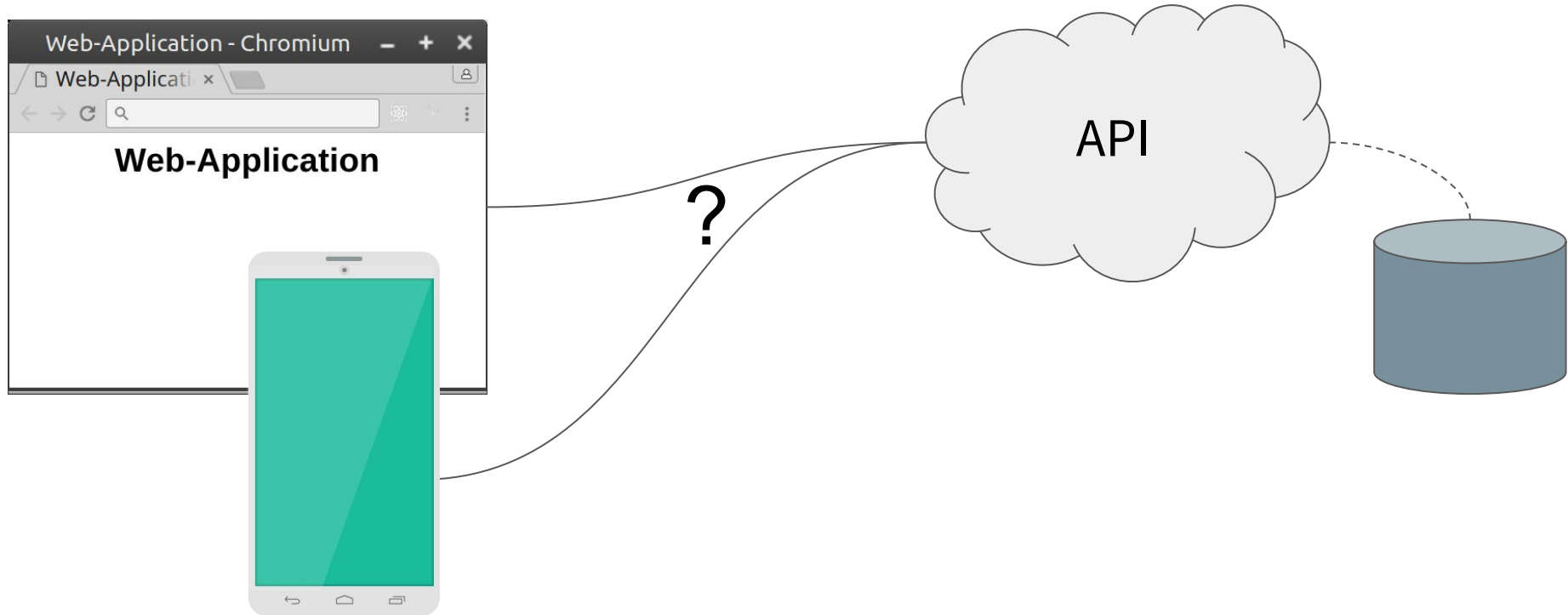
GraphQL



eine Alternative zu REST

Stephan Strehler
stephan.strehler@ottogroup.com
 @StephanStrehler

Themengebiet



Beispiel Shopsystem – Einstieg

Use Cases

- Kategorien / Produkte anzeigen
- Produkte in Warenkorb legen
- Warenkorb anzeigen
- ...

Link zum Shopsystem: <https://github.com/StevieSteven/graphql-example>

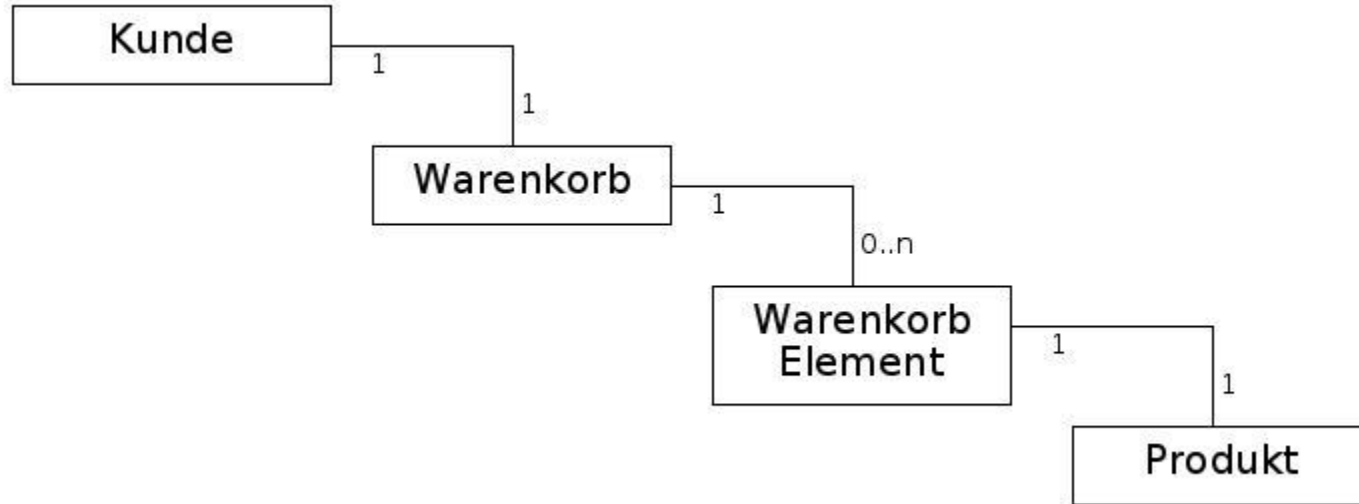
Shopsystem – Anforderungen des Kunden

- soll funktionieren
- soll schnell sein
- (soll gut aussehen)
- (soll billig sein)

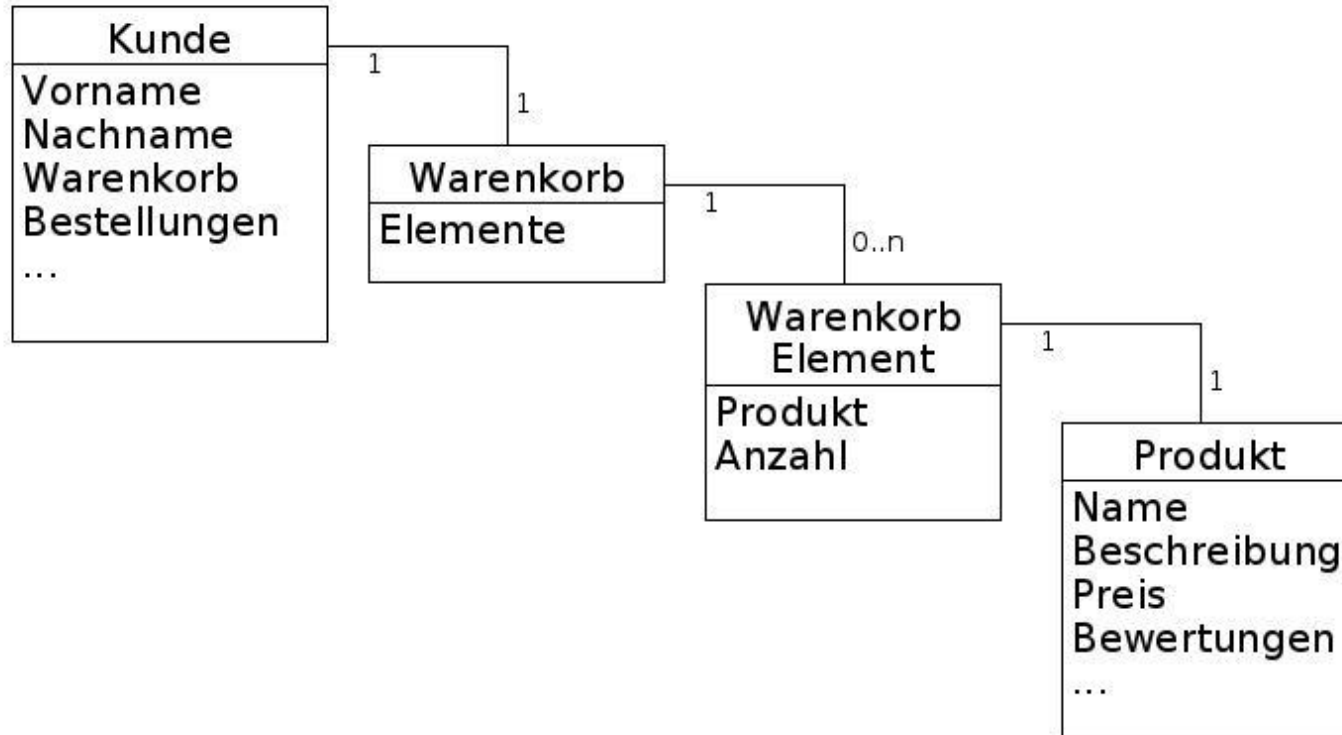
Shopsystem – Anforderungen als Entwickler

- moderne Technologien
- gekapselte Komponenten zur Wiederverwendbarkeit
- keine Dokumentation notwendig
- leicht zu testen

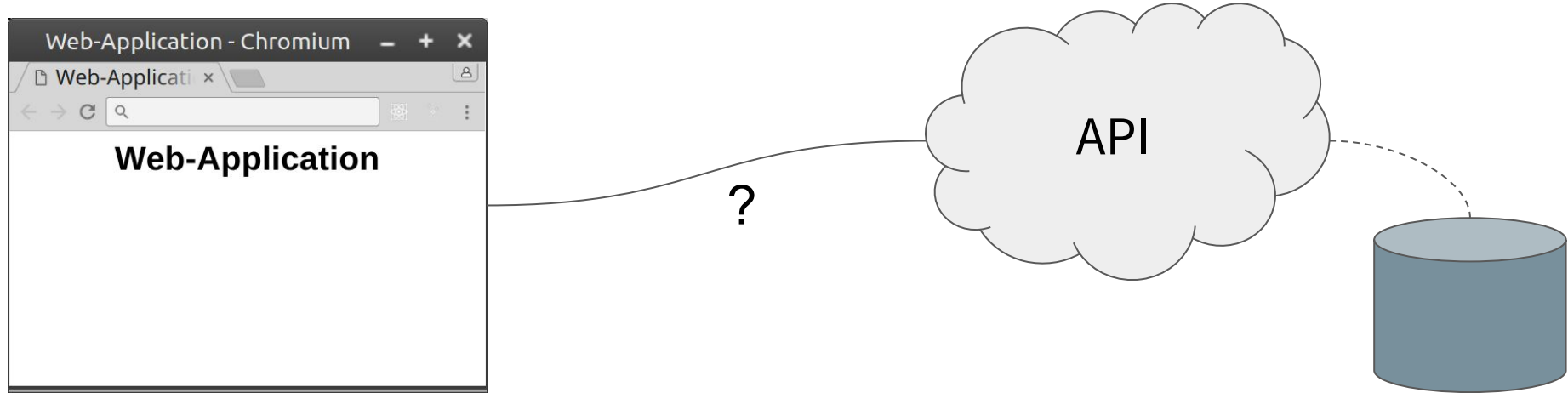
Shopsystem – Warenkorb



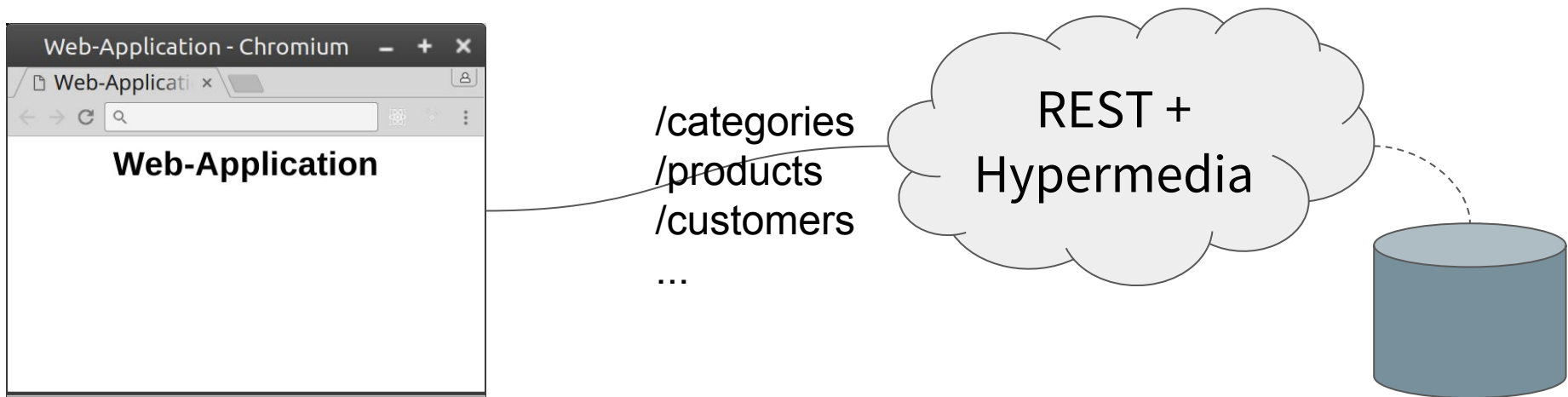
Shopsystem – Warenkorb



Shopsystem



Shopsystem – REST + Hypermedia Entwurf



Vorteile von REST

- Etablierte Technologie
- HTTP als Protokoll
- Caching der Daten auf Netzwerkebene
- Klare Struktur der Schnittstelle
- Lose Kopplung zwischen Client und Server

Shopsystem –Laden des Warenkorbs



Mein Warenkorb 

Request:

/cart

Response:

n Links zu /cart_item/:id

Shopsystem – Laden des Warenkorbs



Mein Warenkorb (5 Produkte)

Rustic Concrete Chicken

Preis: 726 € pro Stück

Beschreibung: Qui voluptatum nulla non consequatur necessitatibus aliquam eum quibusdam aperiores. Ipsa enim quia minus esse namquam sequi tempora commodi ut. Velit voluptatem doloribus qui. Aut facilis voluptatum. Quasi labore voluptas. Sit debita necessitatibus quia ut dolorum accusantium dolorum.

1 Stück für

726 €

Licensed Granite Chair

Preis: 313 € pro Stück

Beschreibung: Nostrum voluptas minus voluptas qui. Repellat ipsum ipsa. Aperiam commodi neque voluptatem. Pariatur nostrum consequatur voluptatem sit eum.

2 Stück für

626 €

Practical Soft Pizza

Preis: 218 € pro Stück

Beschreibung: Porro occaecati ea atque quasi delectus consequatur et. Cumque enim sapiente nemo nulla. Culpa dolores namquam itaque et congue exceptat. Eos tenetur ea nostrum dignissimos qui dolorum velit et consequatur. Quam et non.

4 Stück für

872 €

Handmade Fresh Car

Preis: 79 € pro Stück

Beschreibung: Sit ad fugiat commodi in atque. Aut et voluptatem voluptas molestiae reiciendis. Iste rerum cum iure reprehenderit explicabo ea quis esse. Voluptate impedit enim accusamus fugiat.

4 Stück für

316 €

Request:

`n /cart_items/:id`

Response:

- `n cart_items`

- je 1 Link zu `/product/:id`

Shopsystem – Laden des Warenkorbs



Mein Warenkorb (5 Produkte)

Rustic Concrete Chicken

Preis: 726 € pro Stück

Beschreibung: Qui voluptatum nulla non consequat necessitatibus aliquam eum quibusdam asperiores. Ipsa enim quia minus esse numquam sequi tempora commodi ut. Velit voluptatem doloribus qui. Aut facilis voluptatum. Quasi labore voluptas. Sit debitis necessitatibus quia ut dolorum accusantium dolorem.

1 Stück für

726 €

Licensed Granite Chair

Preis: 313 € pro Stück

Beschreibung: Nostrum voluptas minus voluptas qui. Repellat ipsam ipsa. Aperiam commodi neque voluptatem pariatur nostrum consequat voluptatem sit eum.

2 Stück für

626 €

Practical Soft Pizza

Preis: 218 € pro Stück

Beschreibung: Porro occaecati ea atque quasi delectus consequat et. Cumque enim sapiente nemo nulla. Culpa dolores numquam itaque et corrupti excepturi. Eos tenetur ea nostrum dignissimos qui dolorum velit et consequat. Quaerat et non.

4 Stück für

872 €

Handmade Fresh Car

Preis: 79 € pro Stück

Beschreibung: Sit ad fugiat commodi in atque. Aut et voluptatem voluptas molestiae reiciendis. Iste rerum cum iure reprehenderit explicabo ea quis esse. Voluptate impedit enim accusamus fugiat.

4 Stück für

316 €

Request:

n /products/:id

Response:

- n products

REST + Hypermedia: Problem 1

$1 + (2 * n)$ Requests
notwendig um Warenkorb
anzuzeigen!

n: Anzahl an Produkten

REST + Hypermedia: Problem 2

Zu viele Informationen!

Viele Daten...

Links zu weiteren Ressourcen...

... werden nicht benötigt!

REST + Hypermedia: Problem 3

View spezifische Ressourcen

/search

/category_list

...

⇒ Kopplung zwischen Server und Client

REST + Hypermedia: Problem 4

Sortierung der Produkte:

/products?orderBy=???&???=???

⇒ Parameter orderBy unbekannt

REST + Hypermedia: Zusammenfassung

1. Viele Requests notwendig
2. Zu viele Daten werden übermittelt
3. View-spezifische Ressourcen
4. Keine Typisierung

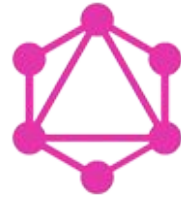
Aussicht für Entwickler

```
export default class CategoryList extends React.Component {  
  state = {categories: []};  
  componentDidMount () {  
    axios.get(`https://webshop.de/categories`)  
      .then(res => {  
        this.setState ({ categories: res.data.categories });  
      })  
  }  
  render () {  
    return (  
      <ul>  
        { this.state.categories.map (category =>  
<li>{category.name}</li>)}  
      </ul>  
    )  
  }  
}
```

Wunsch des Entwicklers

- Beschreibung der benötigten Daten
- Das Resultat soll ein Objekt sein, welches genau diese Daten mit dieser Struktur liefert.

neuer Ansatz



GraphQL

Vorstellung

- Abfragesprache für (Web-)APIs
- Entwickelt von Facebook
- Seit 2012 bei Facebook Mobile App im Einsatz
- Seit Ende 2015 OpenSource

Vorstellung

- Spezifikation existiert (letzte Version: Juni 2018)
- graphql.js als offizielle Implementationsprache von FB
- Bibliotheken auch in anderen Sprachen vorhanden
 - Java
 - Python
 - ...

Ansicht für Entwickler

```
const categoriesQuery = gql`  
  query CategoriesQuery($id: ID!) {  
    category(uuid: $id) {  
      uuid  
      name  
      numberOfProducts  
    }  
  }  
`;  
`;
```

```
const Container = ({data}) => {  
  if (data.loading) {    return <Loading/>  }  
  const category = data.category;  
  return (  
    <Collapse defaultActiveKey={['1']}>  
      <Panel header={category.name} key="1">  
        <p>{category.numberOfProducts}  
          verfügbar</p>  
      </Panel>  
    </Collapse>);  
};
```


Shopsystem: Frameworks



Als Verbesserung

Wie erfüllt GraphQL die Wünsche?

Generelles

- Besitzt ein einzigen Einstiegspunkt
- GET oder POST als HTTP Methode
- Trennung von Query und Mutation

Technik

Schema

Resolver

Query

Definiert serverseitige Schnittstelle:

```
type Cart {  
  uuid: ID!,  
  date: String!  
  items: [CartItem!]  
}
```

```
cart(uuid: ID!): Cart
```

Technik

Schema

Resolver

Query

Funktion zum Bereitstellen der Daten:

```
cart(_, {uuid}) {  
    return Cart.findByUUID(uuid);  
},
```

Technik

Schema

Resolver

Query

Clientseitige Abfrage:

```
{  
  cart(uuid:"...") {  
    items {  
      quantity  
      product {  
        name  
        price  
      }  
    }  
  }  
}
```

GraphQL

Serverseitig

- Typen
- Schemaerzeugung
- Resolver

Typen

Scalare Typen

Int, Boolean, String, Float

⇒ keine extra Resolve
Funktion nötig

komplexe Typen, Enums

⇒ Resolve Funktion
notwendig

Schema Programmatisch (graphql-java)

```
GraphQLObjectType categoryType = GraphQLObjectType.newObject()  
    .name("Category")  
    .field(GraphQLFieldDefinition.newFieldDefinition()  
        .name("uuid")  
        .type(GraphQLID))  
    .field(GraphQLFieldDefinition.newFieldDefinition()  
        .name("name")  
        .type(GraphQLString))  
    .field(GraphQLFieldDefinition.newFieldDefinition()  
        .name("products")  
        .type(new GraphQLList(  
            new GraphQLTypeReference("Product")  
        ))  
    .build();
```

Schema Programmatisch (graphql.js)

```
const CategoryType = new GraphQLObjectType({  
  name: 'Category',  
  fields: {  
    uuid: { type: GraphQLID },  
    name: { type: GraphQLString },  
    products: { type: new GraphQLList(ProductType) }  
  }  
})
```

Resolver (Mit Apollo GraphQL)

```
const resolveFunctions = {  
  Query: {  
    category(_, {uuid}) {  
      return Category.findByUUID(uuid);  
    }  
  },  
  Category: {  
    products: {  
      resolve(root) {  
        return Category.products(root);  
      }  
    }  
  }  
}
```

Der Server unseres Webshops

Der erste praktische Abschnitt des Workshops. :-)

GraphQL

Clientseitig

- Query
- Query Optimierung
- Mutations

Query

- Jede Komponente besitzt eigenen Query
- Query fragt genau die benötigten Informationen ab

Query Optimierung

[Suche](#)[Mein Warenkorb \(5\)](#) [Meine Bestellungen](#)

Mein Warenkorb (5 Produkte)

Small Fresh Soap

Preis: 552 € pro Stück

Beschreibung: Excepturi aperiam quibusdam nesciunt non. Accusantium hic quisquam possimus veritatis animi atque omnis. Modi itaque officia aut ratione. Non non earum dolor beatae mollitia sint voluptatem aut, Ipsam voluptas soluta rerum vero corporis sed temporibus eveniet ea. Voluptas aut alias.

2 Stück für
1104 €

Gorgeous Rubber Chicken

Preis: 189 € pro Stück

4 Stück für
756 €

Query Optimierung

Navigationsbereich:

```
cart {  
  items {  
    quantity  
  }  
}
```

Warenkorb:

```
cart {  
  items {  
    quantity  
    product {  
      uuid  
      name  
      description  
      price  
    }  
  }  
}
```


Query Optimierung

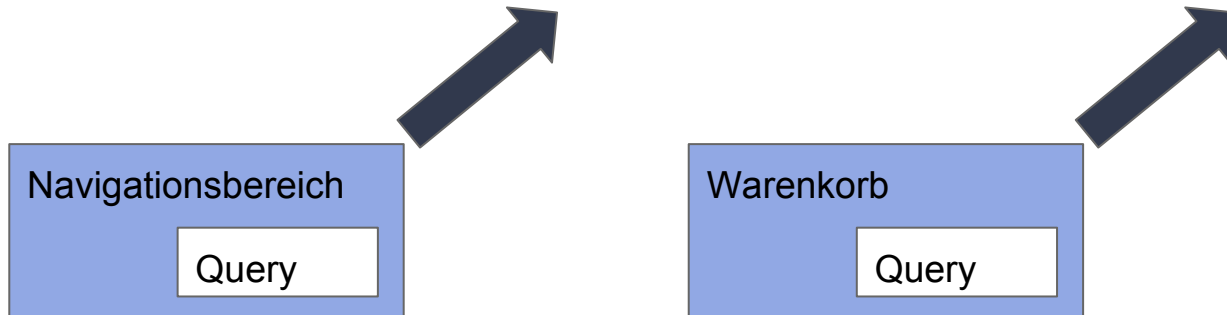
Navigationsbereich:

```
cart {  
  items {  
    quantity  
  }  
}
```

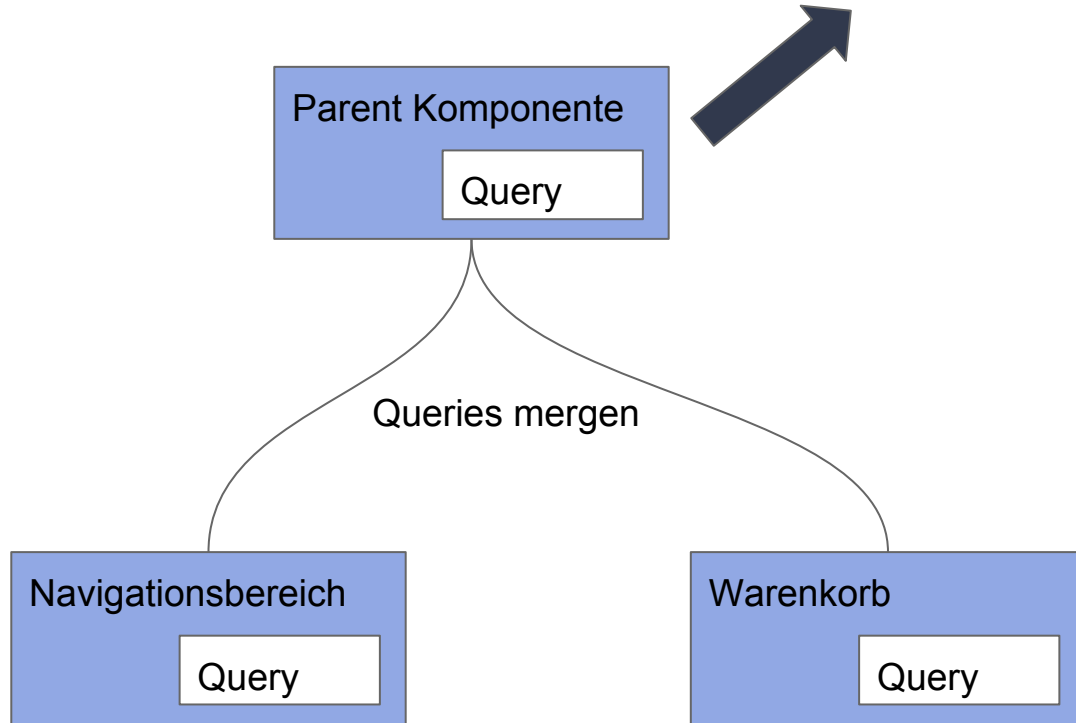
Warenkorb:

```
cart {  
  items {  
    quantity  
    product {  
      uuid  
      name  
      description  
      price  
    }  
  }  
}
```

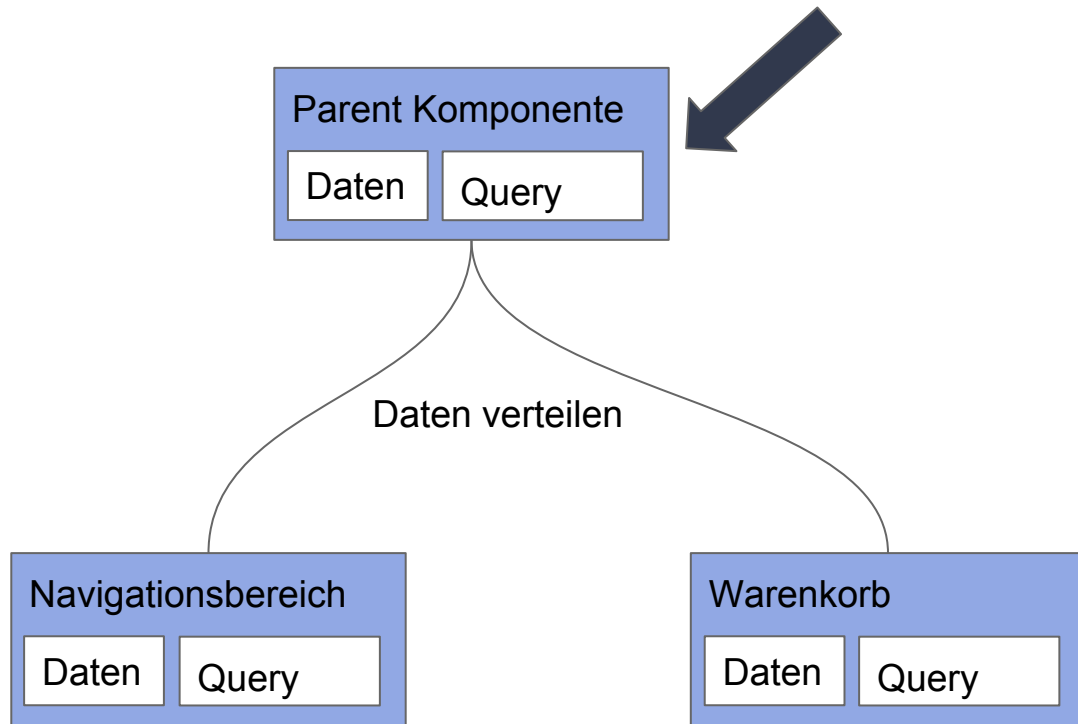
Query Optimierung



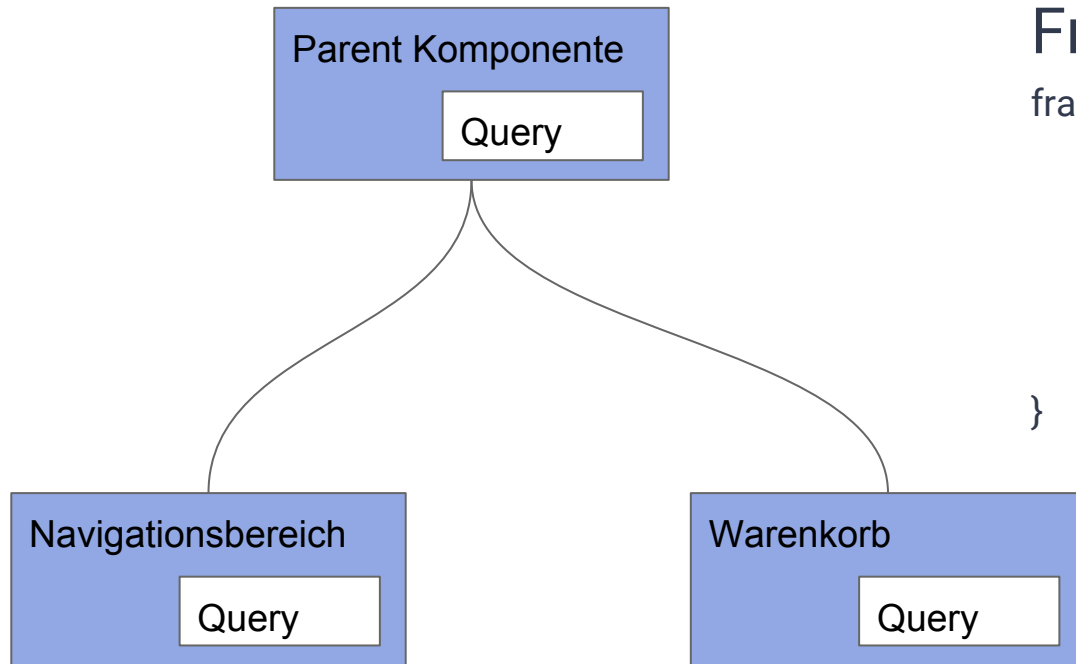
Query Optimierung



GraphQL - Query Optimierung



Query Optimierung



Fragments:

```
fragment generalFields on CartElement {  
  quantity  
  product {  
    uuid  
    price  
    name  
  }  
}
```

Query Optimierung

Aktueller Stand:

- Funktioniert noch nicht problemlos
- Problem: Wann werden die Daten benötigt?

Bisherige Lösung:

- Merging der Requests auf Http-Link-Ebene

Query Optimierung

[Suche](#)[Mein Warenkorb \(5\)](#) [Meine Bestellungen](#)

Handcrafted Rubber Computer

[In den Warenkorb legen](#)

Allgemeine Informationen

Preis: 867€

Beschreibung: Ea et sit distinctio sed error. Rerum praesentium modi consequatur molestiae quasi possimus ut dignissimos. Nihil nulla et et magnam qui est modi. Error et ab ut est mollitia cupiditate et quam. Aliquid aut praesentium minima sit et quam vel deserunt omnis. Id quia consequatur et ipsam hic illum ut quia magnam.

Andere Nutzer sagen



von Sammie Greenholt

Minus quibusdam unde quam animi distinctio. Quo omnis ad. Voluptatum nostrum similique corrupti eum commodi error officia et. Recusandae sunt asperiores laboriosam fugit.

Mutations

- Ähnlich Queries, einziger Unterschied:
- Root Element ist nicht Query, sondern Mutation

⇒ Vergleichbar mit dem CQRS Pattern

Mutations

```
schema {  
  query: Query  
  mutation: Mutation  
}
```

```
type Query {  
  ...  
}
```

```
type Mutation {  
  ...  
}
```

Mutations

Abfrage an Server

```
const addProductToCart = gql`  
  mutation AddProductToCart($uuid: ID!, $quantity: Int!) {  
    addProductToCart(uuid: $uuid, quantity: $quantity) {  
      uuid  
    }  
  }  
`;  
;
```

Mutations

Aufruf der Mutation

```
const handleCartBtn = () => {  
  this.props.mutate({  
    variables: {  
      uuid: this.props.product.uuid,  
      quantity: this.state.numberOfProducts  
    }  
  }).then(({data}) => {  
    ...  
  }).catch((error) => {  
    ...  
  });  
};
```

Mutations

[Suche](#)[Mein Warenkorb \(5\)](#) [Meine Bestellungen](#)

Handcrafted Rubber Computer

 ▾[In den Warenkorb legen](#)

Allgemeine Informationen

Preis: 867€

Beschreibung: Ea et sit distinctio sed error. Rerum praesentium modi consequatur molestiae quasi possimus ut dignissimos. Nihil nulla et et magnam qui est modi. Error et ab ut est mollitia cupiditate et quam. Aliquid aut praesentium minima sit et quam vel deserunt omnis. Id quia consequatur et ipsam hic illum ut quia magnam.


Andere Nutzer sagen



von Sammie Greenholt

Minus quibusdam unde quam animi distinctio. Quo omnis ad. Voluptatum nostrum similique corrupti eum commodi error officia et. Recusandae sunt asperiores laboriosam fugit.

Mutations



Mein Warenkorb (5)

Meine Bestellungen

beeinflusst

1

In den Warenkorb legen

Handcrafted Rubber Computer

Allgemeine Informationen

Preis: 867€

Beschreibung: Ea et sit distinctio sed error. Rerum praesentium modi consequatur molestiae quasi possimus ut dignissimos. Nihil nulla et et magnam qui est modi. Error et ab ut est mollitia cupiditate et quam. Aliquid aut praesentium minima sit et quam vel deserunt omnis. Id quia consequatur et ipsam hic illum ut quia magnam.

Andere Nutzer sagen

★★★★☆

von Sammie Greenholt

Minus quibusdam unde quam animi distinctio. Quo omnis ad. Voluptatum nostrum similique corrupti eum commodi error officia et. Recusandae sunt asperiores laboriosam fugit.

GraphQL

Fazit

Fazit – Vorteile

- Anzahl der Requests wurden minimiert
- Datenmenge reduziert
- “maßgeschneiderte” Daten
- Saubere Komponententrennung möglich
- exakte Definition der Schnittstelle
- Schnittstellenexplorer (GraphiQL)
- Verbesserte parallele Arbeitsprozesse in der Entwicklung

Fazit – Nachteile

- Sehr hohe Entwicklungsgeschwindigkeit
- Kinderkrankheiten
- Serverauslastung kann höher sein
- Kein Cache?

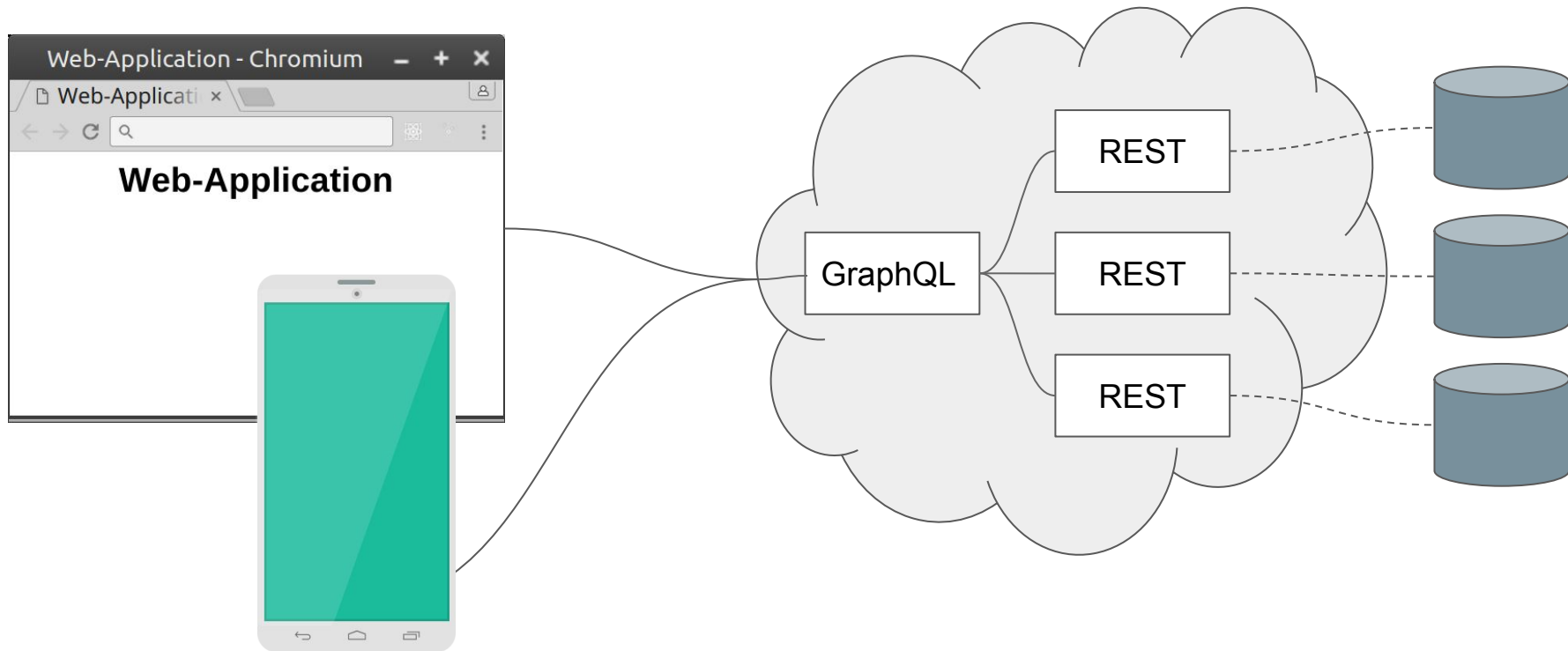
Cache

- Netzwerk Cache nicht möglich
- Cache muss von Client Bibliothek übernommen werde

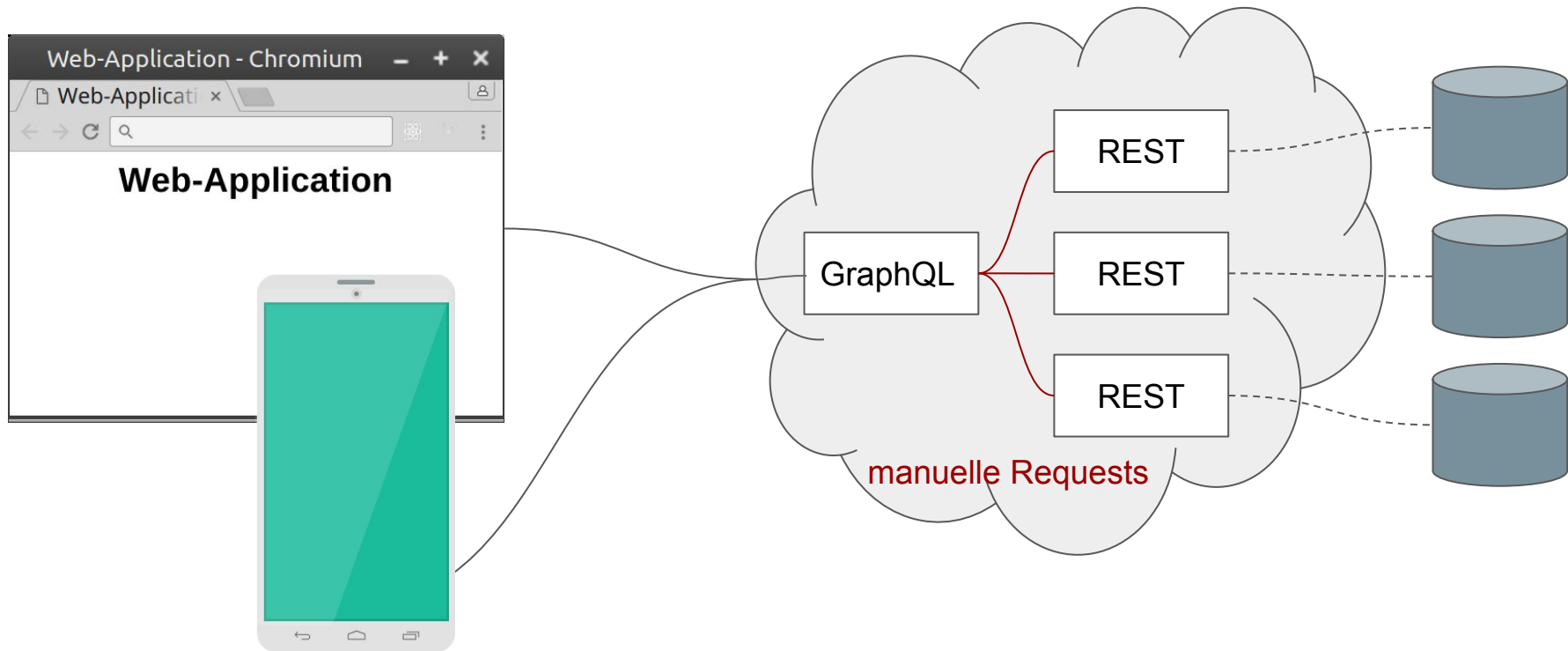
GraphQL

neue Möglichkeiten

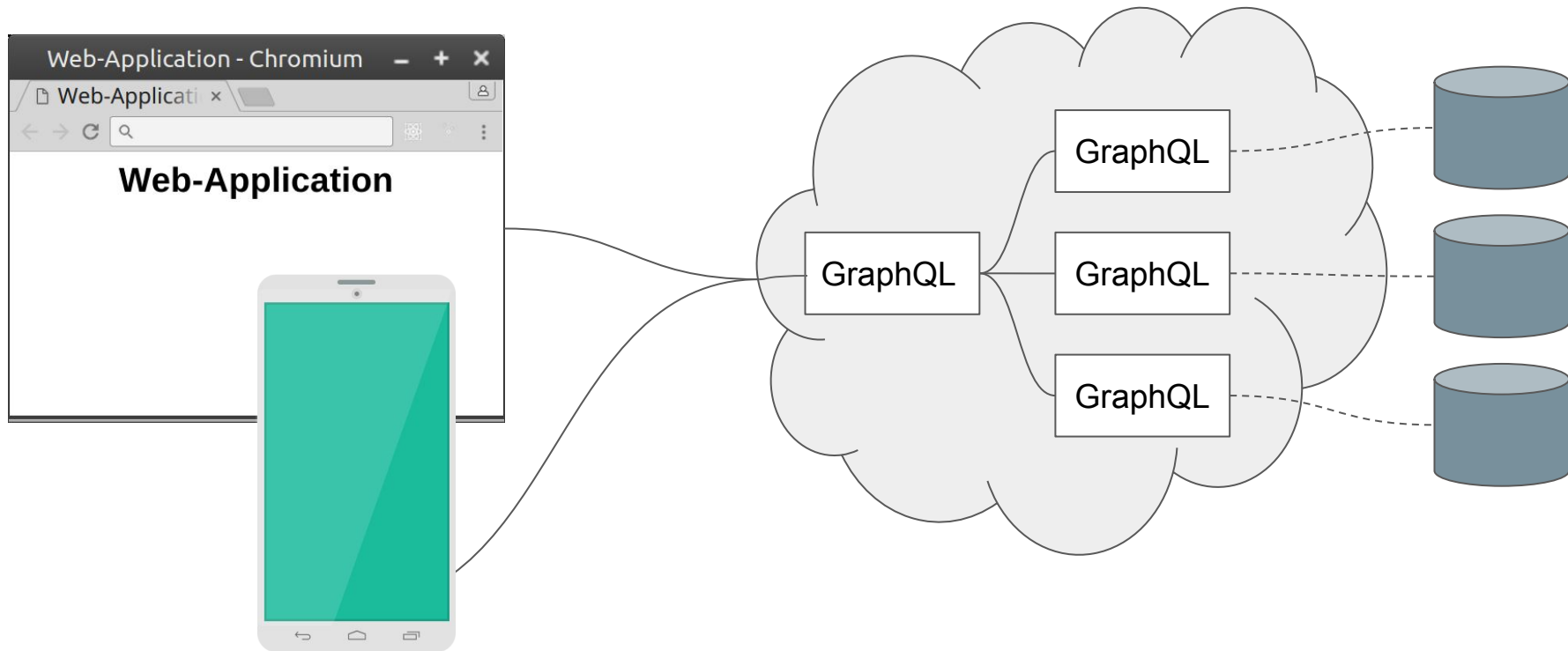
GraphQL Schema Stitching



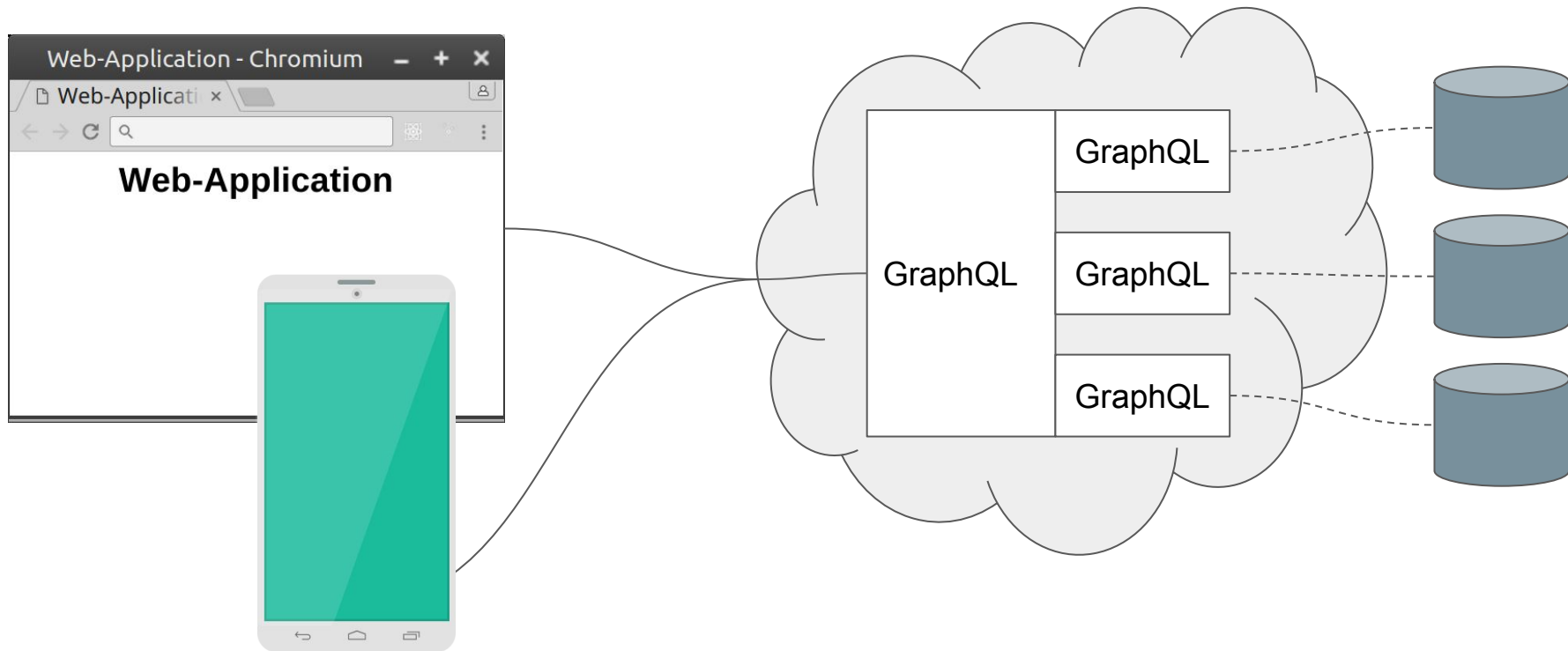
GraphQL Schema Stitching



GraphQL Schema Stitching



GraphQL Schema Stitching



Schema Stitching

- *createRemoteSchema* Teil von *graphql-tools* (Apollo)
- Laden von allen Schemas anderer Servern
- Koppeln von Schemas
- Requests werden automatisch weitergeleitet
- Java: graphql-braid

Beispiel: <https://github.com/StevieSteven/graphql-schema-stitching>

Abschluss

Abschluss

Take Aways

- GraphQL besteht aus Schema, Resolver und Query
- GraphQL besitzt statisch definiertes Schema
- Aufbau Query richtet sich nach Schema
- Komplexere Clientbibliotheken nötig

Abschluss

Further Reading

Projekt mit anderen Technologien (Manuel Mauky):

<https://github.com/lestard/bloggie>

Apollo GraphQL:

<https://www.apollographql.com/>

React und GraphQL:

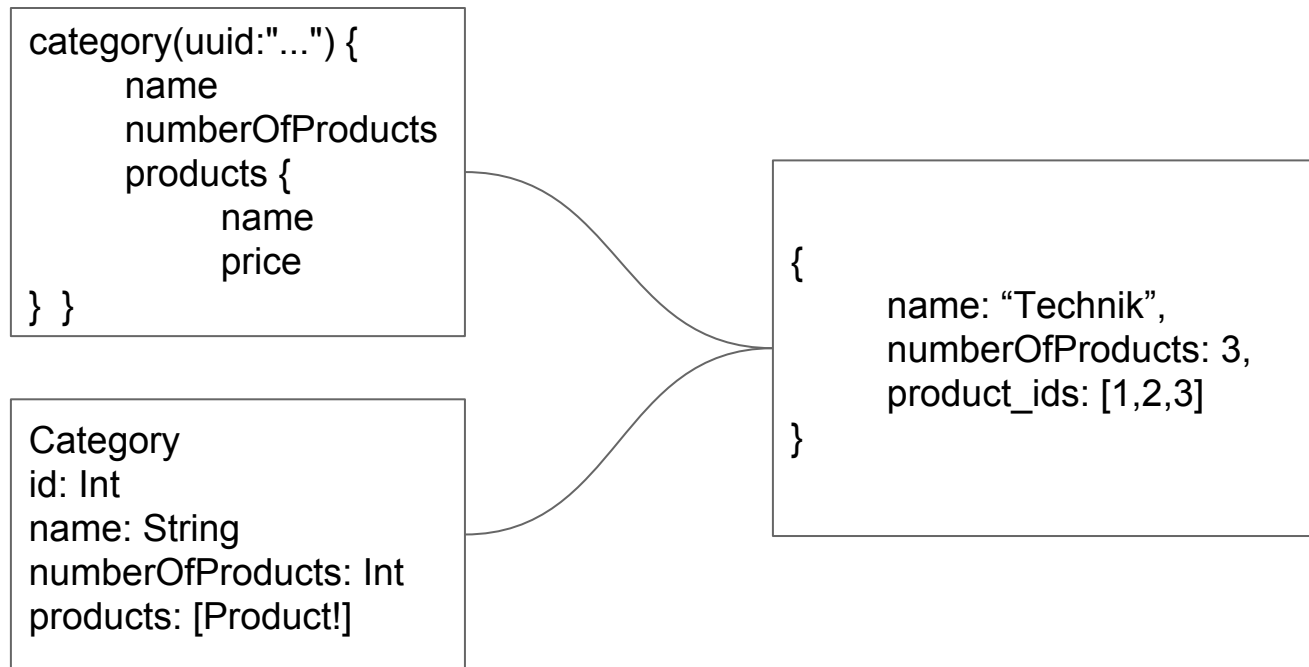
<https://dev-blog.apollodata.com/seamless-integration-for-graphql-and-react-6ffc0ad3fead>

Fragen?

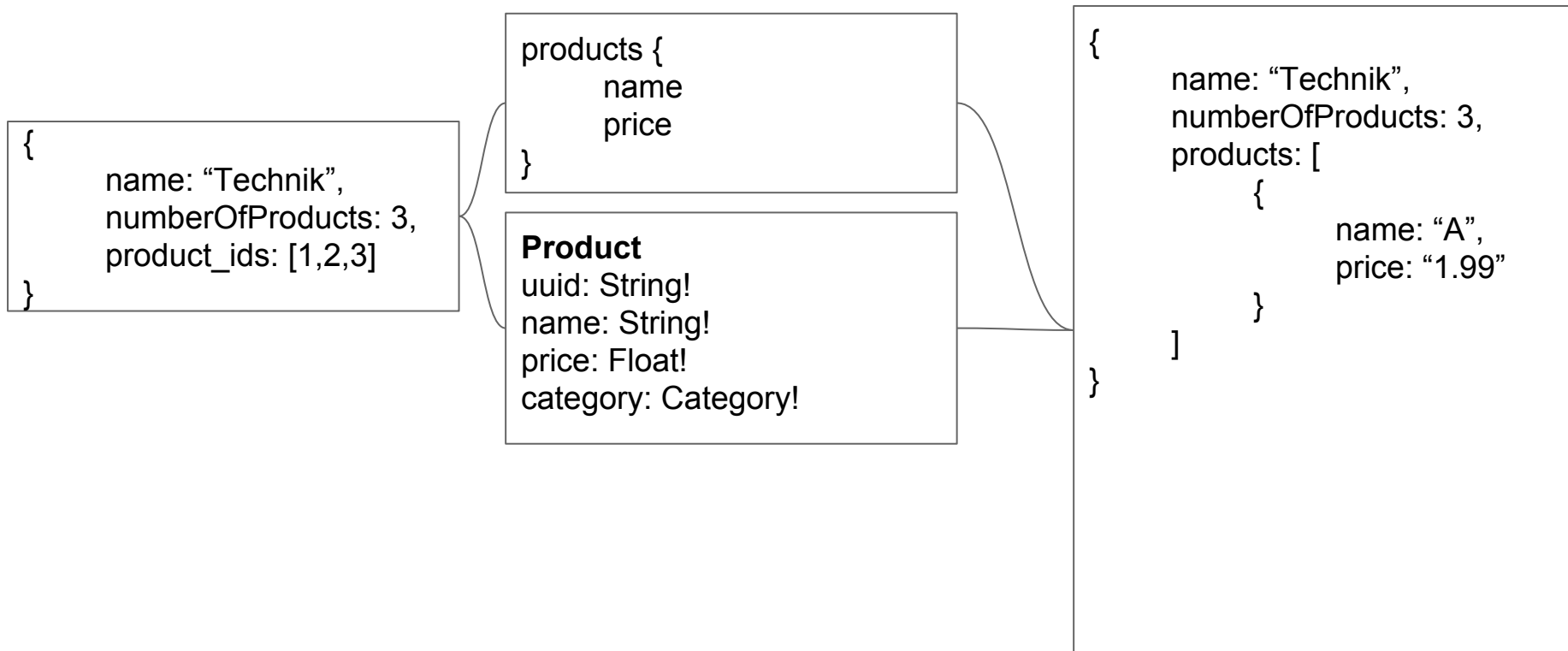
Stephan Strehler
@StephanStrehler

technische Details

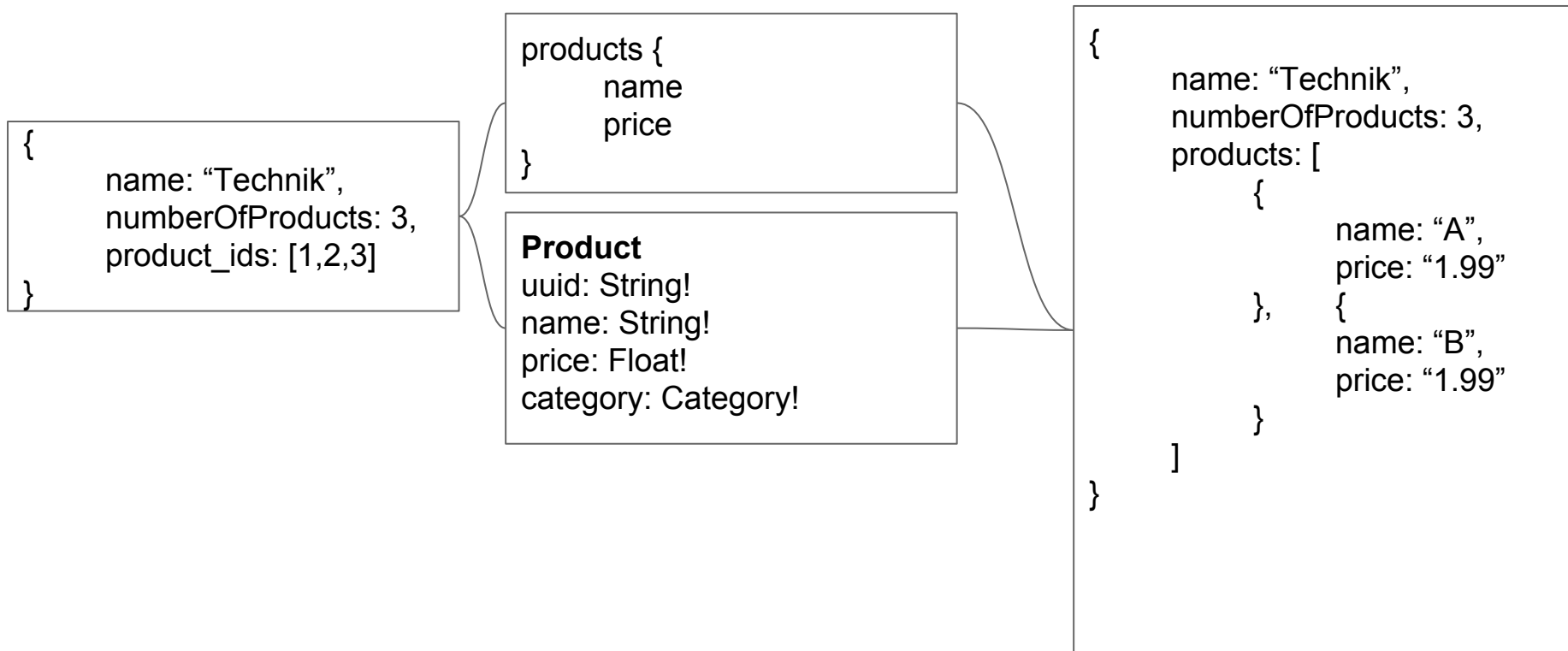
Ablauf Request



Ablauf Request



Ablauf Request



Ablauf Request

