

# GraphQL

## 前后端协同从未如丝顺滑

2018.07.14

陈嘉

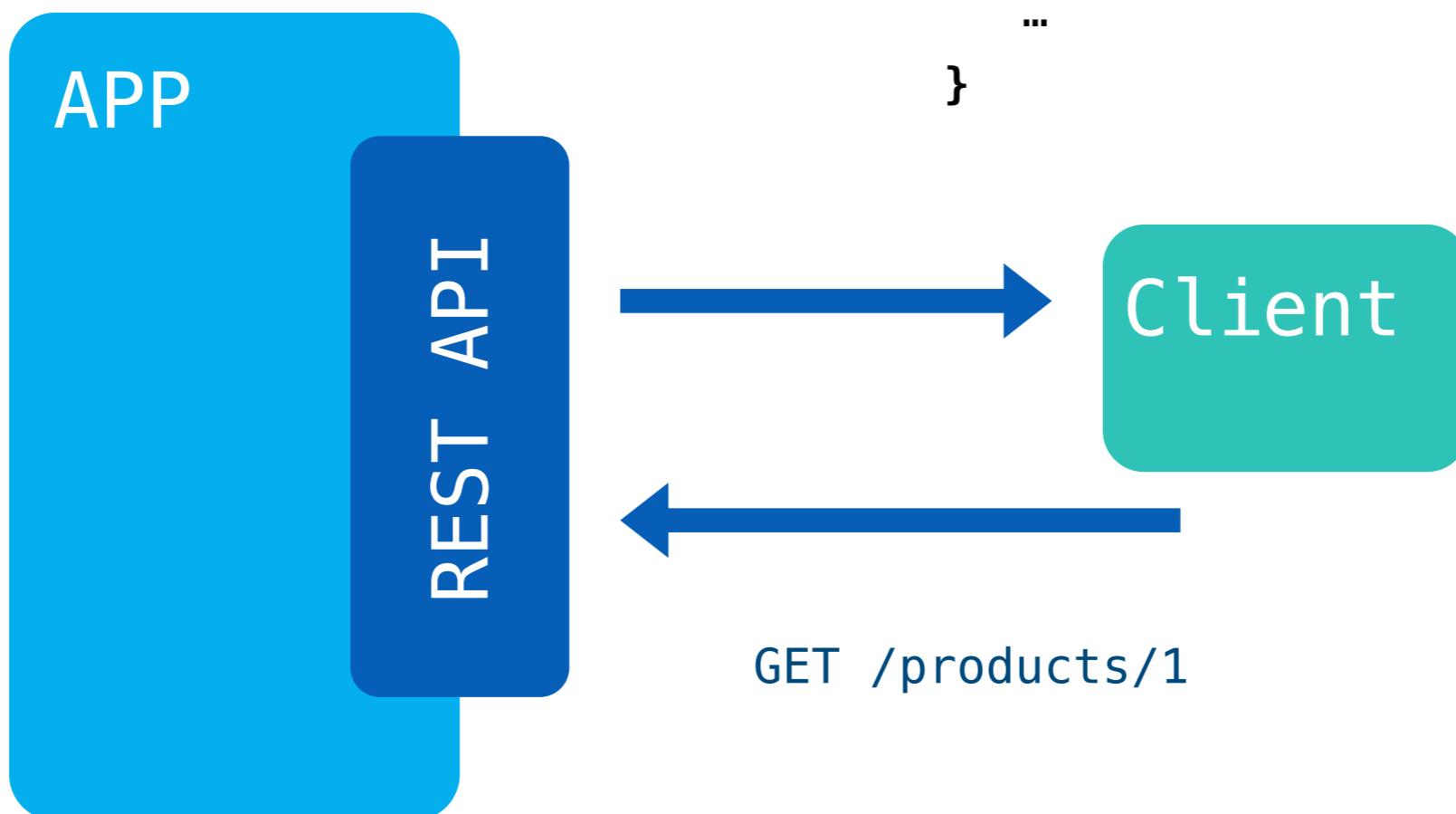


# TOPIC

- 1 什么是 GraphQL
- 2 我们如何使用 GraphQL
- 3 GraphQL in 前端



```
{  
  "id": "1",  
  "name": "product name",  
  "price": 1.23,  
  ...  
}
```



# OVER FETCHING



Tech Fleece cotton-blend jersey  
hooded top

```
[  
  ...  
  {  
    "id": "1111",  
    "name": "Tech Fleece cotton-blend",  
    "price": 143.00,  
    "image": "http://...",  
    "description": "Nike's hooded top is made ..."  
    "skus": [  
      {  
        ...  
        "size": "L"  
      }  
    ]  
  }  
  ...  
]
```

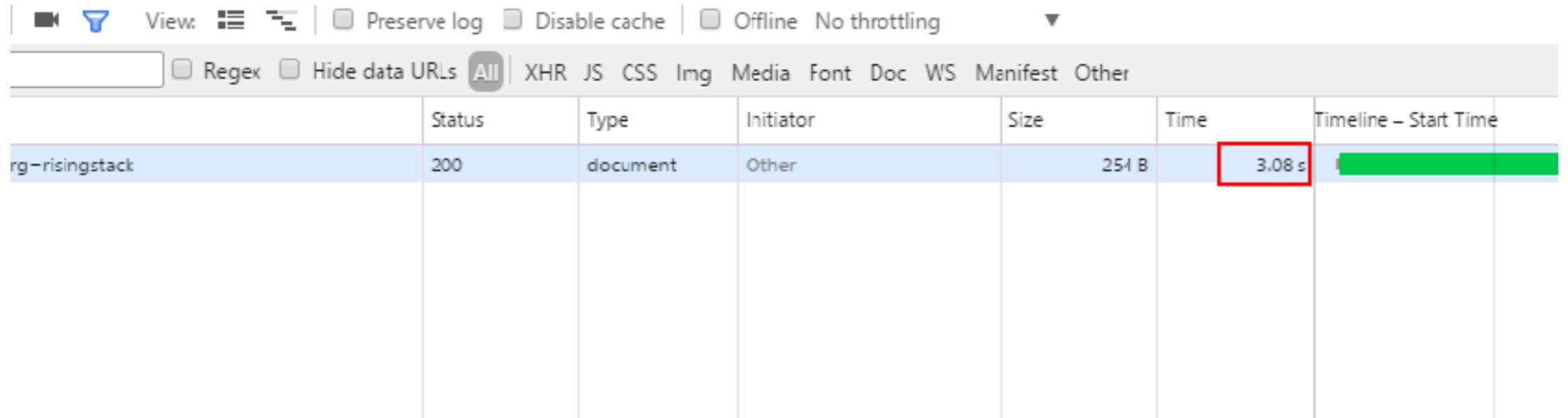
# OVER FETCHING



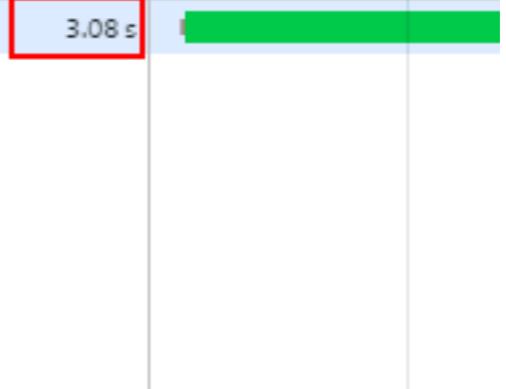
Tech Fleece cotton-blend jersey  
hooded top

```
[  
...  
{  
  "id": "1111",  
  "name": "Tech Fleece cotton-blend",  
  "price": 143.00,  
  "image": "http://...",  
  "description": "Nike's hooded top is made ..."  
  "skus": [  
    {  
      ...  
      "size": "L"  
    }  
  ]  
}  
...  
]
```

# OVER FETCHING



A screenshot of a browser's developer tools Network tab. The tab shows a single entry for a document fetch from 'rg-risingstack'. The entry details are as follows:

	Status	Type	Initiator	Size	Time	Timeline – Start Time
rg-risingstack	200	document	Other	251 B	3.08 s	

The 'Time' column for the document entry is highlighted with a red box, indicating a significant delay in the fetch process.

# UNDER FETCHING



NIKE  
Tech Fleece cotton-blend jersey  
hooded top  
\$143

```
[  
...  
{  
  "id": "1111",  
  "name": "Tech Fleece cotton-blend",  
  "price": 143.00,  
  "image": "http://...",  
  "description": "Nike's hooded top is made ..."  
  "skus": [  
    {  
      ...  
      "size": "L"  
    }  
  ]  
}  
...  
]
```

## TypeError: Cannot read property 'name' of undefined

App.render  
src/App.js:14

```
11 |         <img src={logo} className="App-logo" alt="logo" />
12 |     </div>
13 |     <p className="App-intro">
> 14 |       {user.profile.name}
15 |     </p>
16 |   </div>
17 | );
```

[View compiled](#)

▼ 19 stack frames were expanded.

(anonymous function)  
node\_modules/react-dom/lib/ReactCompositeComponent.js:795

measureLifeCyclePerf  
node\_modules/react-dom/lib/ReactCompositeComponent.js:75

ReactCompositeComponentWrapper.\_renderValidatedComponentWithoutOwnerOrContext  
node\_modules/react-dom/lib/ReactCompositeComponent.js:794

ReactCompositeComponentWrapper.\_renderValidatedComponent  
node\_modules/react-dom/lib/ReactCompositeComponent.js:821

ReactCompositeComponentWrapper.performInitialMount  
node\_modules/react-dom/lib/ReactCompositeComponent.js:361

ReactCompositeComponentWrapper.mountComponent  
node\_modules/react-dom/lib/ReactCompositeComponent.js:257

问题



同志，快醒醒，API少了一个字段

BAO ZUN

## 问题

POST /users 创建用户

**Implementation Notes**  
根据User对象创建用户

**Response Class (Status 200)**

Response Content Type `/*` application/json application/xml

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
<b>user</b>	(required)	用户详细实体user	body	Model <span style="border: 1px solid #ccc; padding: 5px;">Model Schema</span>

Parameter content type: `application/json` application/xml

`{  
 "age": 0,  
 "id": 0,  
 "name": "string"  
}`

Click to set as parameter value

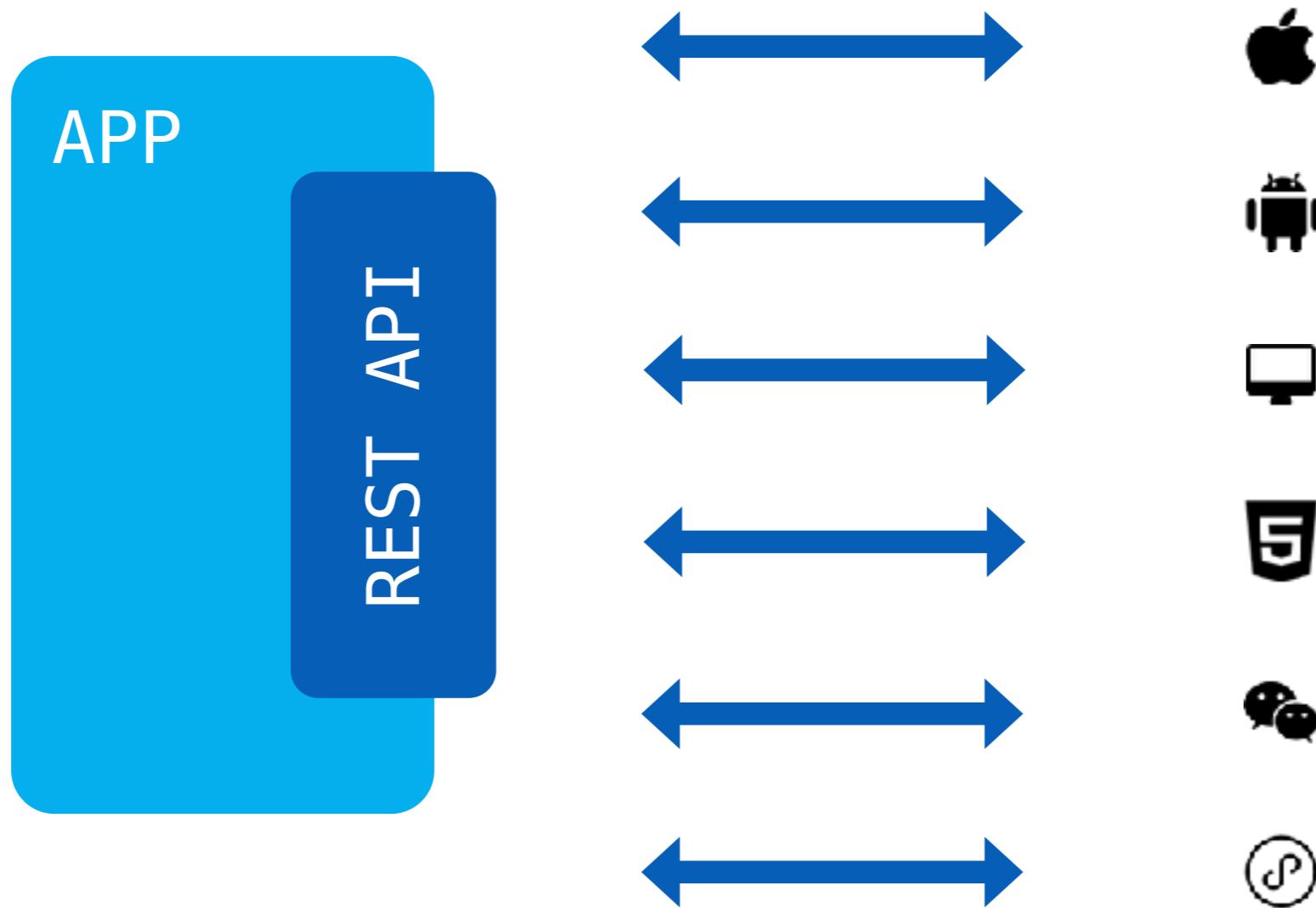
**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
<b>201</b>	Created		
<b>401</b>	Unauthorized		
<b>403</b>	Forbidden		
<b>404</b>	Not Found		

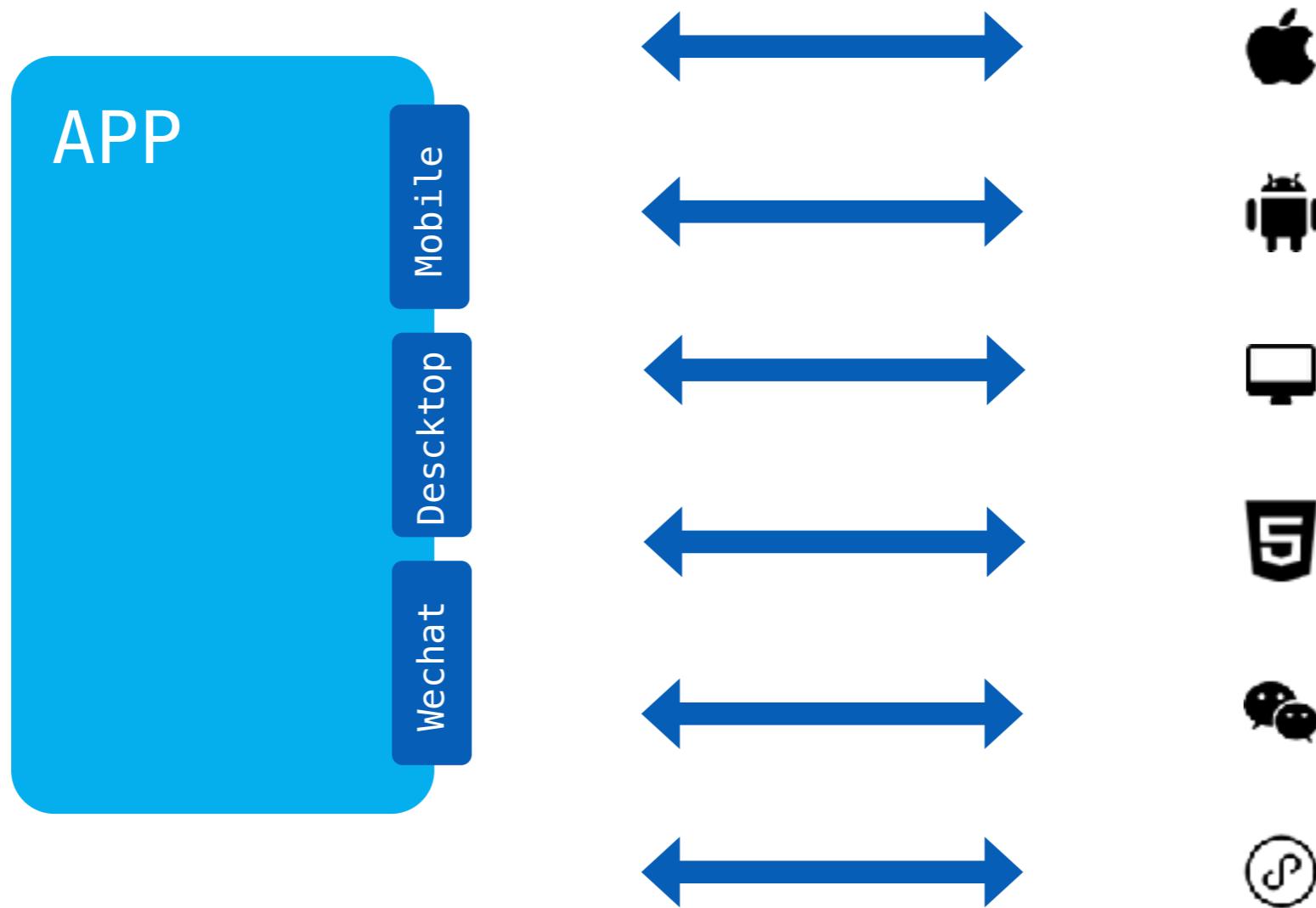
Try it out!

BAO ZUN

# 移动时代

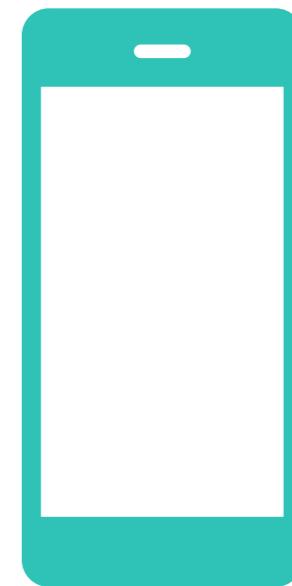


# 移动时代



FQL

```
Select * from timelines;
```



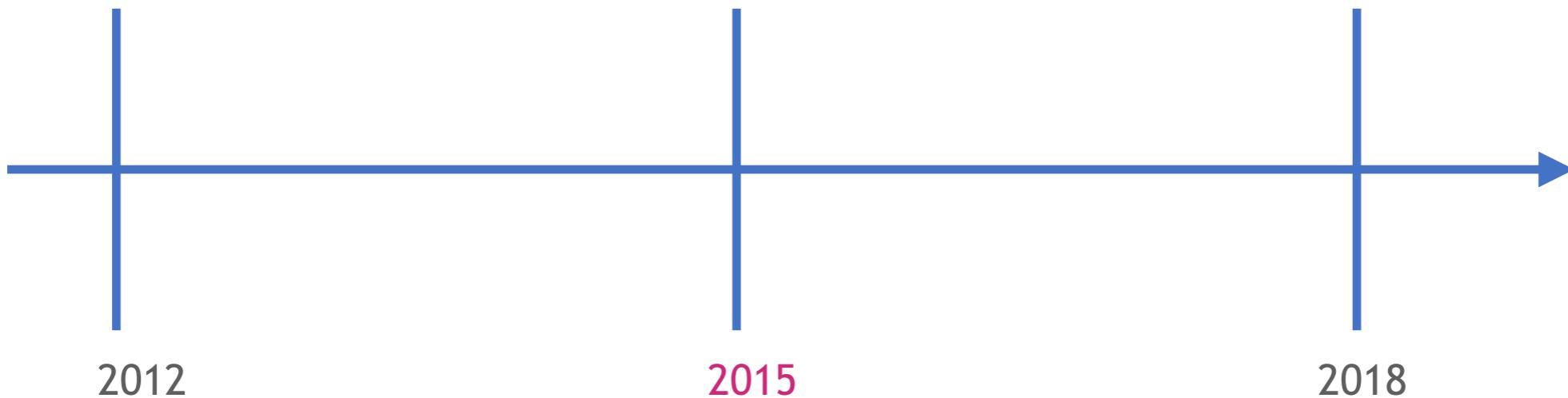
```
{  
  "timelines": {...},  
}
```

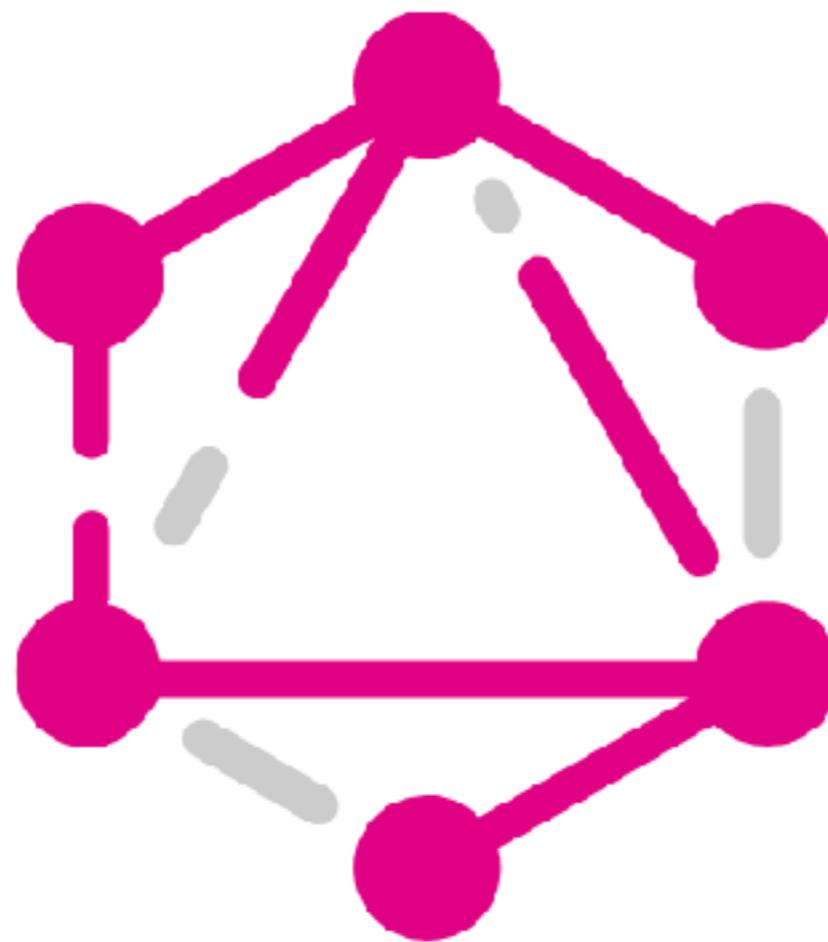


构思GraphQL (Facebook  
mobile app - News Feed)

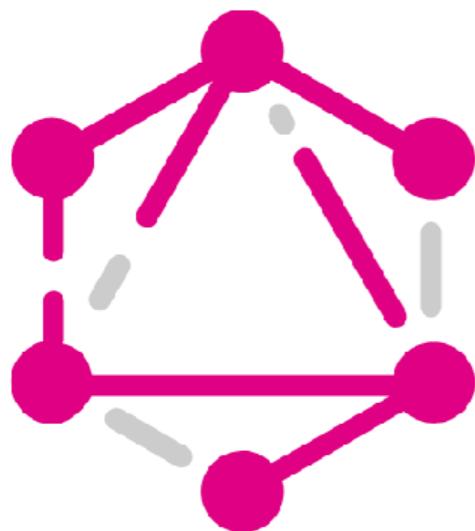
GraphQL Spec 及  
Reference 开源

现在  
开源已三年





# GraphQL



# GraphQL 是一个数据查询语言

GraphQL 不是什么！

GraphQL 不是一个编程语言

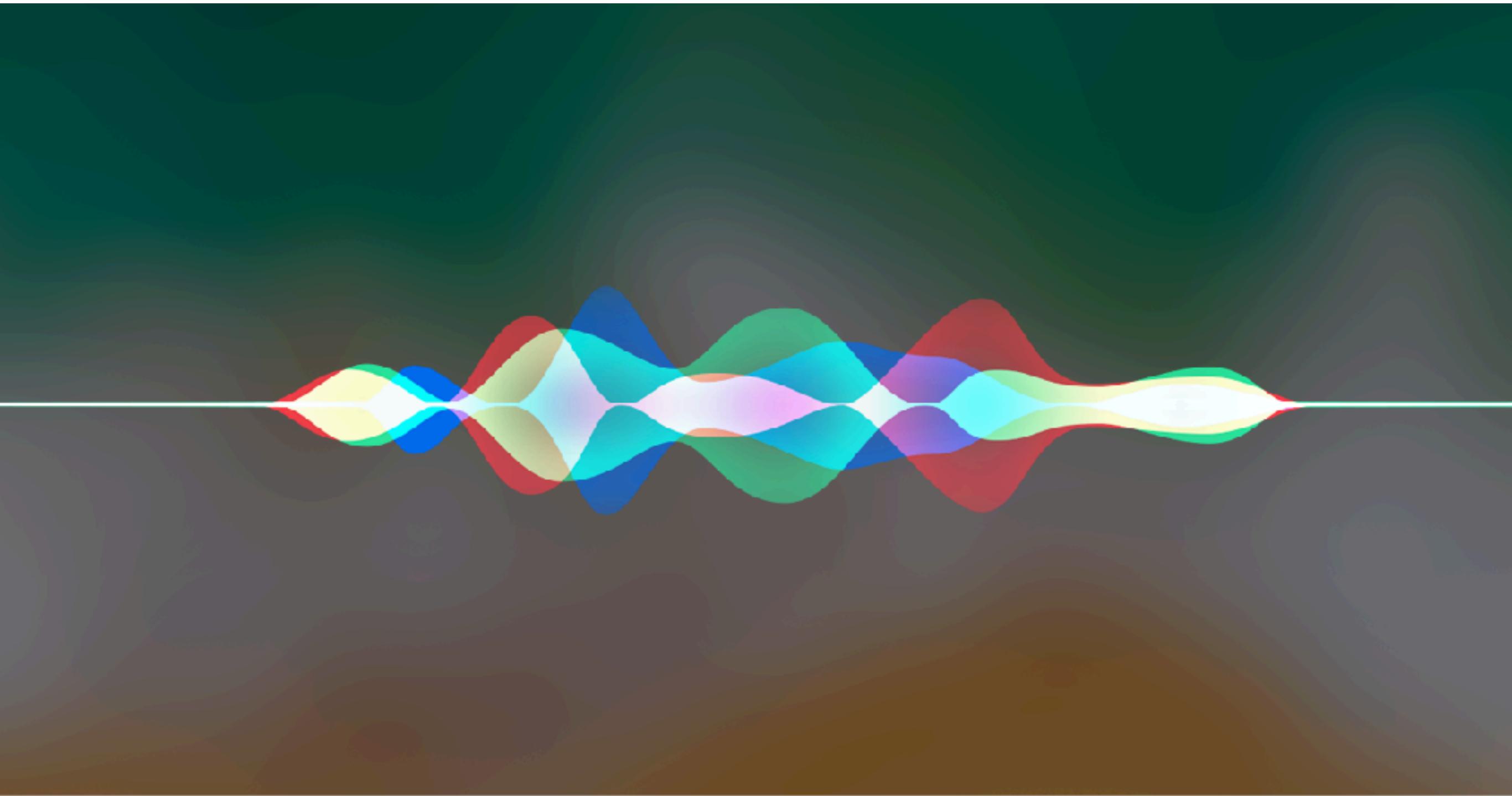
GraphQL 不是数据库

GraphQL 不是某一个框架

GraphQL 不限制于某一个语言



GraphQL

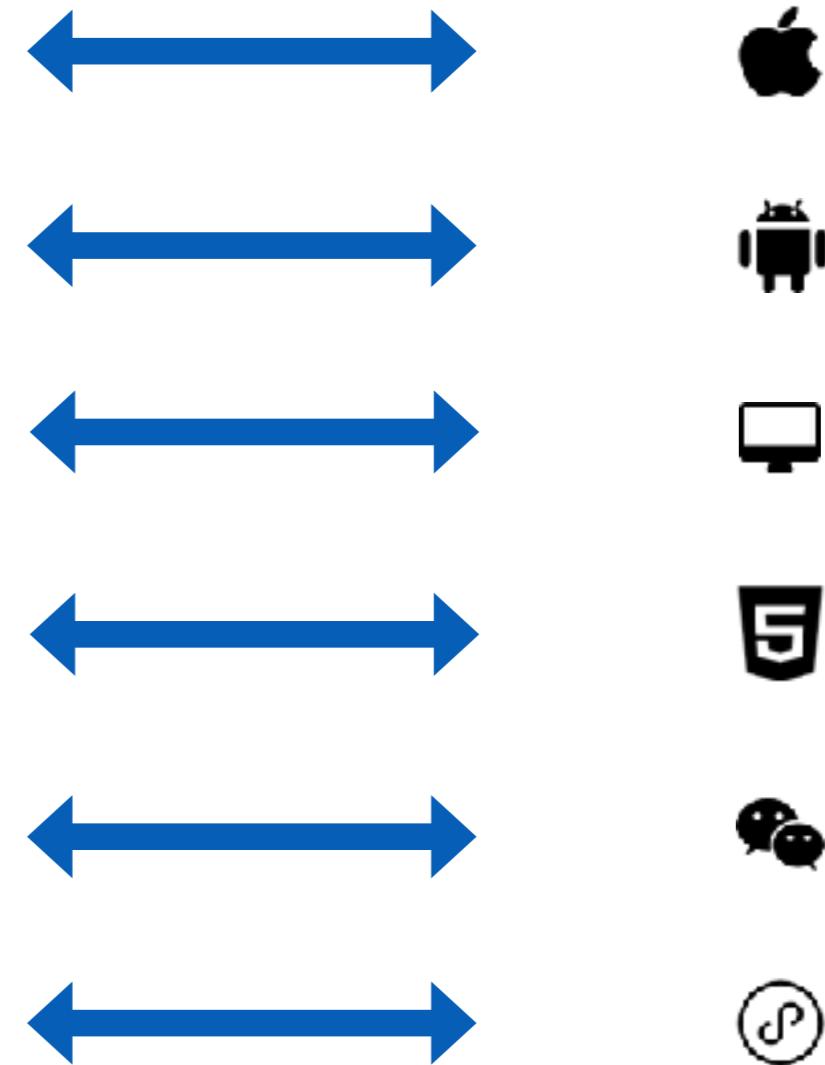


## Graphql - response 结构

客户端驱动开发： 可预测的响应结构

```
query productList {  
  product(id: 123) {  
    name  
    description  
    picture {  
      url  
    }  
  }  
}
```

```
{  
  "data": {  
    "product": {  
      "name": "官方AIR JORDAN 1",  
      "description": "Just do it!"  
      "picture": {  
        "url": "http://..."  
      }  
    }  
  }  
}
```

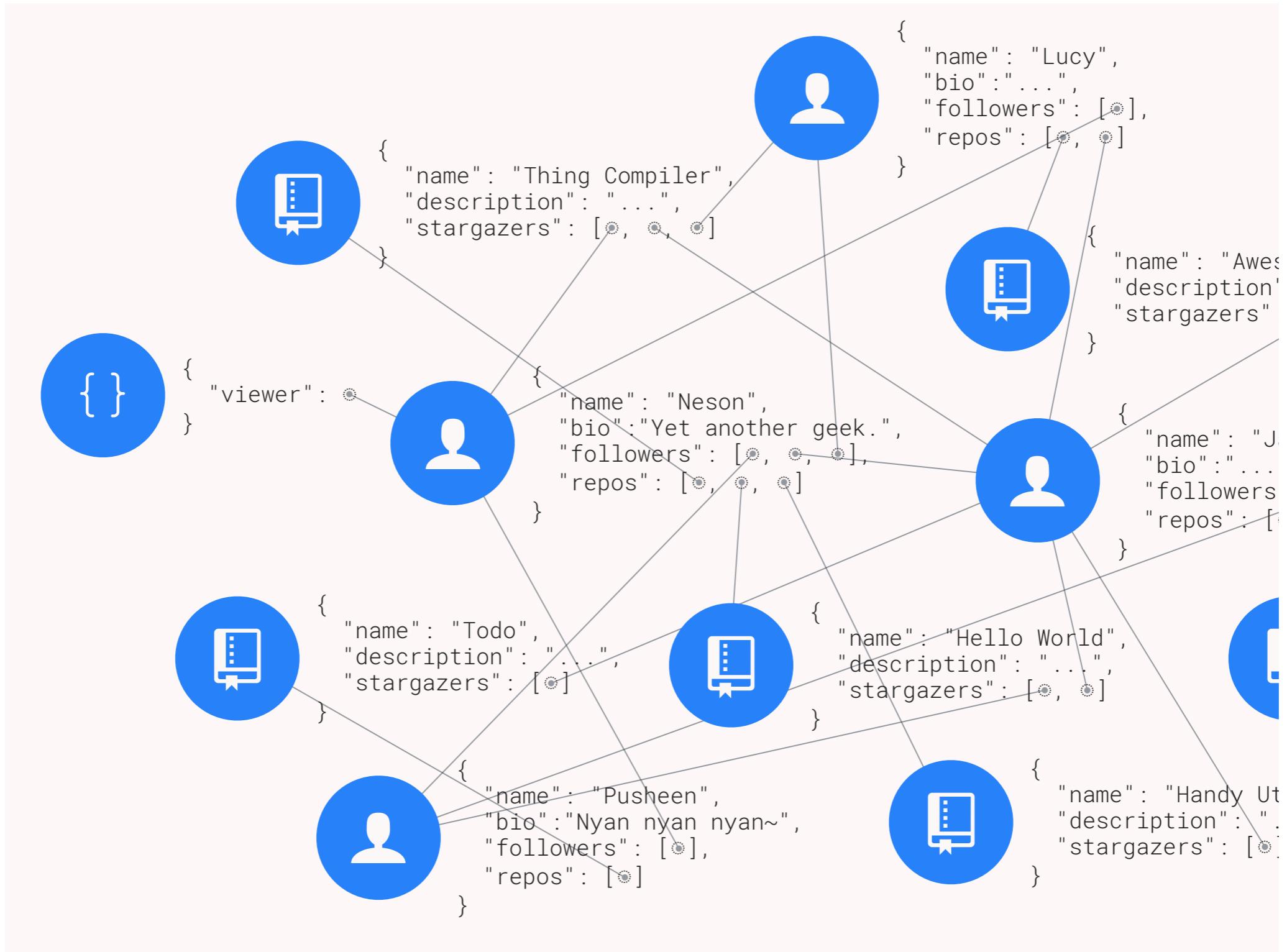


```
query products {  
  product1: product(id: 1) {  
    name  
    price  
  }  
  product2: product(id: 2) {  
    description  
  }  
  categories: {  
    categories {  
      name  
    }  
  }  
}
```

```
type Product {  
    id: String!  
    name: String!  
    description: String  
    price: Float  
}
```

```
type Category {  
    id: String!  
    name: String!  
    products: [Product!]  
}
```

# Graphql - type system



BAO ZUN

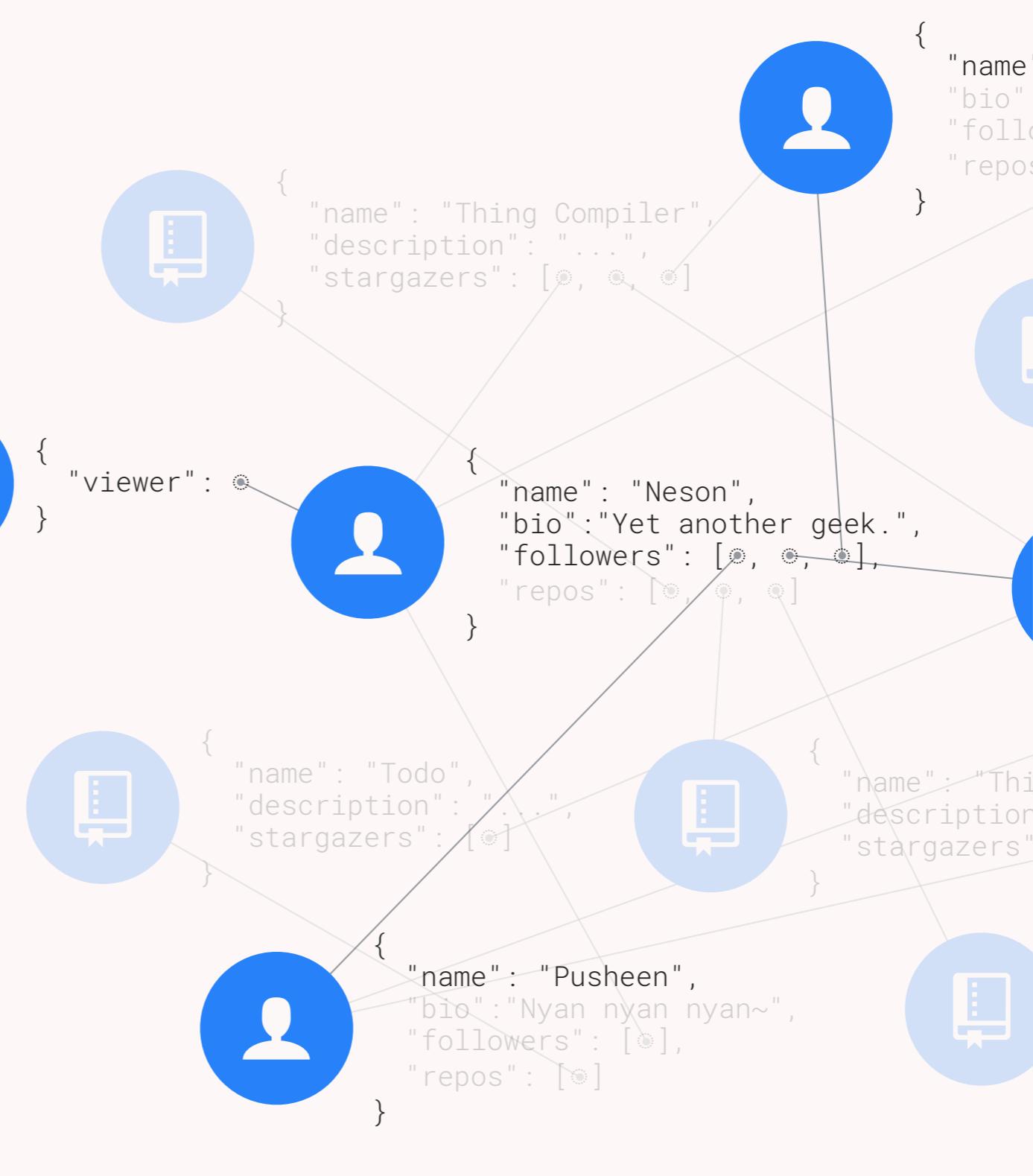
## Graphql - type system

```
query {  
  viewer {  
    name  
    bio  
  }  
}
```



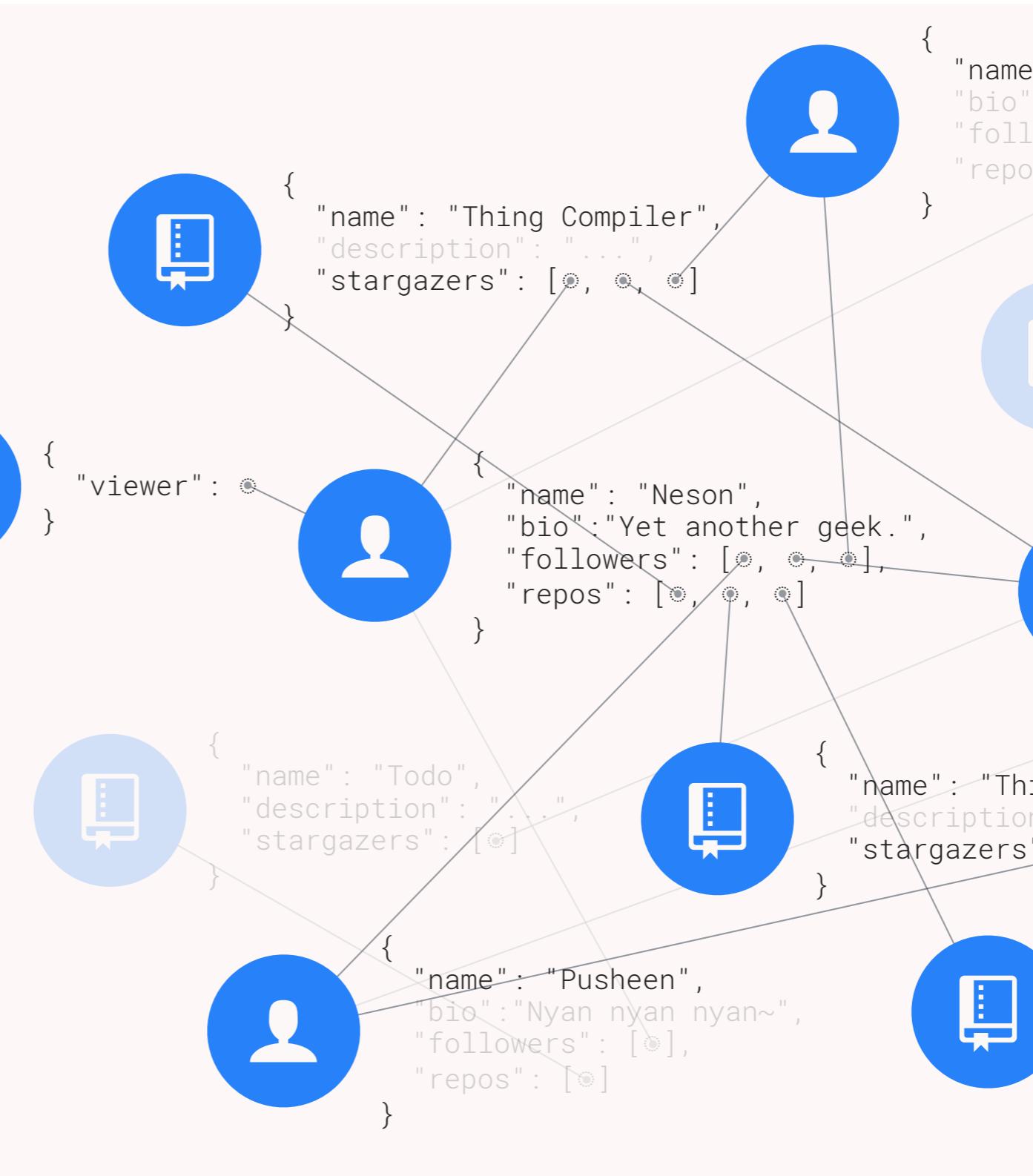
## Graphql - type system

```
query {  
  viewer {  
    name  
    bio  
    followers {  
      name  
    }  
  }  
}
```



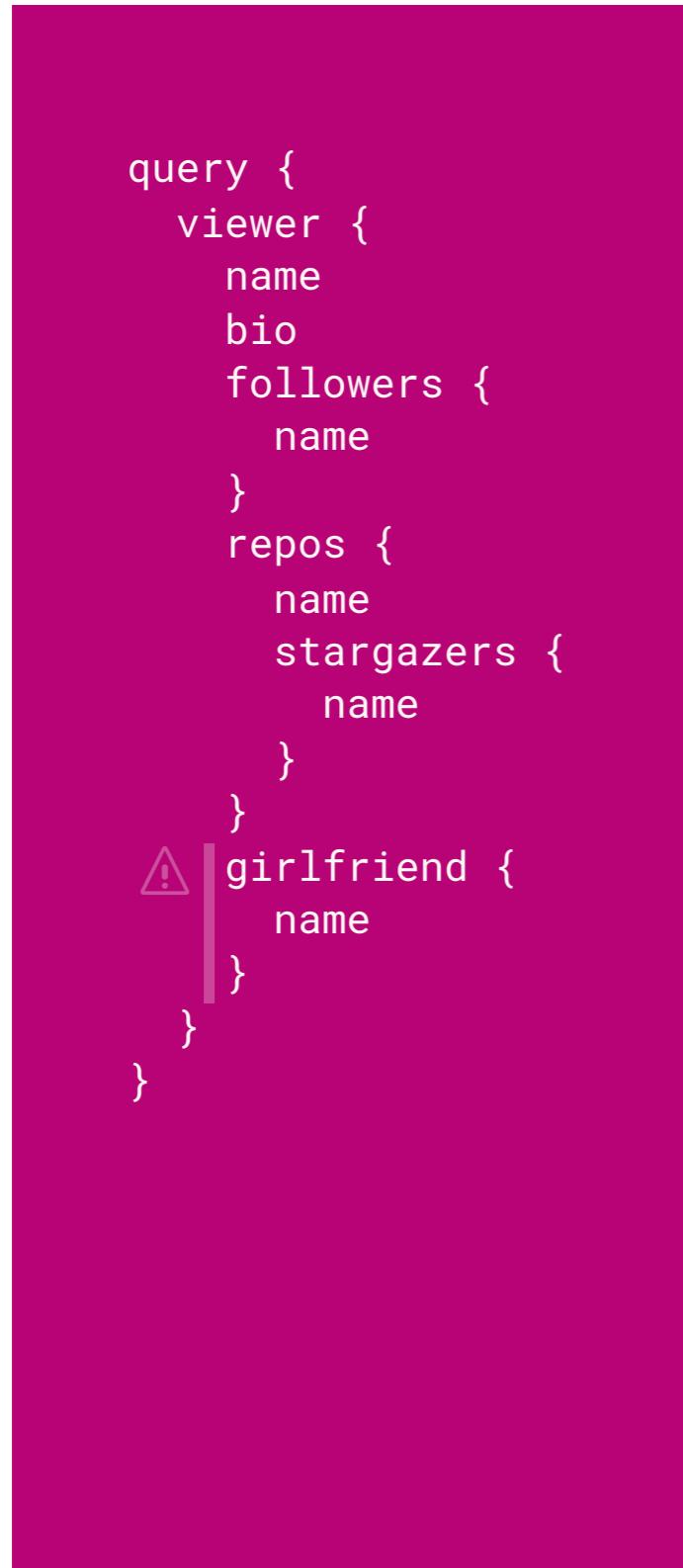
## Graphql - type system

```
query {  
  viewer {  
    name  
    bio  
    followers {  
      name  
    }  
    repos {  
      name  
      stargazers {  
        name  
      }  
    }  
  }  
}
```



BAO ZUN

## Graphql - type system



BAO ZUN

## Graphql - type system

```
query {
  viewer {
    name
    bio
    followers {
      name
    }
    repos {
      name
      stargazers {
        name
      }
    }
    ⚠ girlfriend {
      name
    }
  }
}
```

```
{
  "errors": [
    {
      "message": "Field 'girlfriend' doesn't exist on type 'User'", ...
    }
  ]
}
```

BAO ZUN

BAO ZUN

# 我们为什么要用

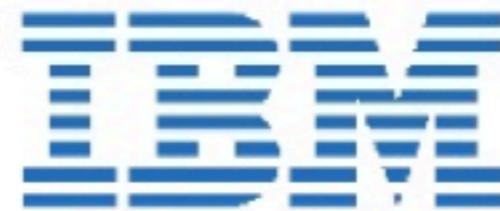


# 家里有矿啊？

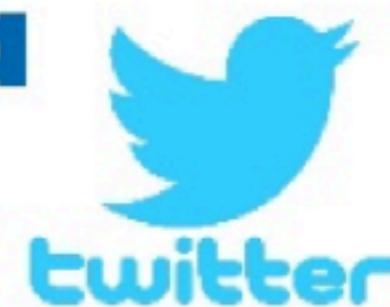


人，时间，资源是有限的

BAO ZUN



**facebook** **The New York Times**



BAO ZUN



# 大家都在用



[awesome-graphql](#)

BAO ZUN

BAO ZUN

# 我们怎样使用





# 构建前后端通用语言

## Schema First Development





BAO ZUN

# Graphql - type system

NikePlus CONVERSE

加入/登录 NikePlus 账号 帮助 购物车

男子 女子 男孩 女孩 NIKEID 专属定制 搜索

**筛选**

**鞋类**

- 拖鞋/凉鞋
- 鞋子
- 钉鞋

**鞋高**

- 中高帮
- 低帮
- 中帮

**NIKEID**

- 全方位定制

**最适宜**

- 冬季保暖系列
- 防水

**品牌**

- Nike
- Jordan
- Nike Sportswear
- NikeLab

3 色 Nike EXP-X14 男子运动鞋 ¥ 899

2 色 Nike EXP-X14 SE 男子运动鞋 ¥ 899

14 色 ★★★★★ (5) Nike Epic React Flyknit 男子跑步鞋 ¥ 1,299

18 色 ★★★★★ (4) Nike Air VaporMax Flyknit 2 男子跑步鞋 ¥ 1,599

3 色 Nike Zoom KD11 EP 男子篮球鞋 ¥ 1,299

4 色 ★★★★★ (1) Kyrie Flytrap EP 男子篮球鞋 ¥ 599

2 色 ★★★★★ (18) Air Jordan 11 Retro Low 复刻男子运动鞋 ¥ 1,099

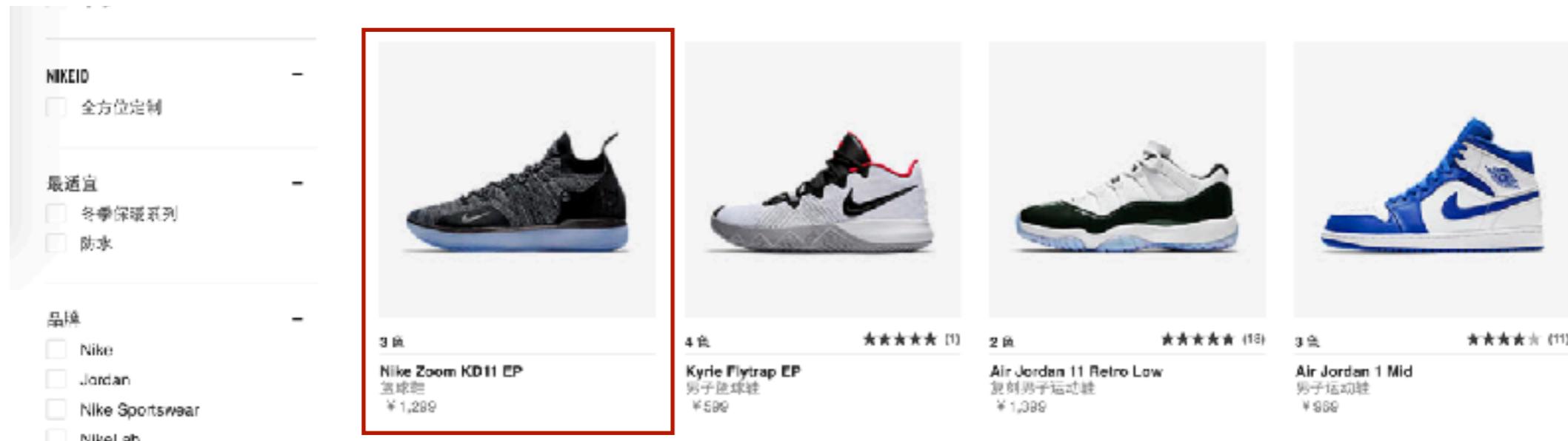
3 色 ★★★★★ (11) Air Jordan 1 Mid 男子运动鞋 ¥ 999

**BAO ZUN**

## Graphql - type system

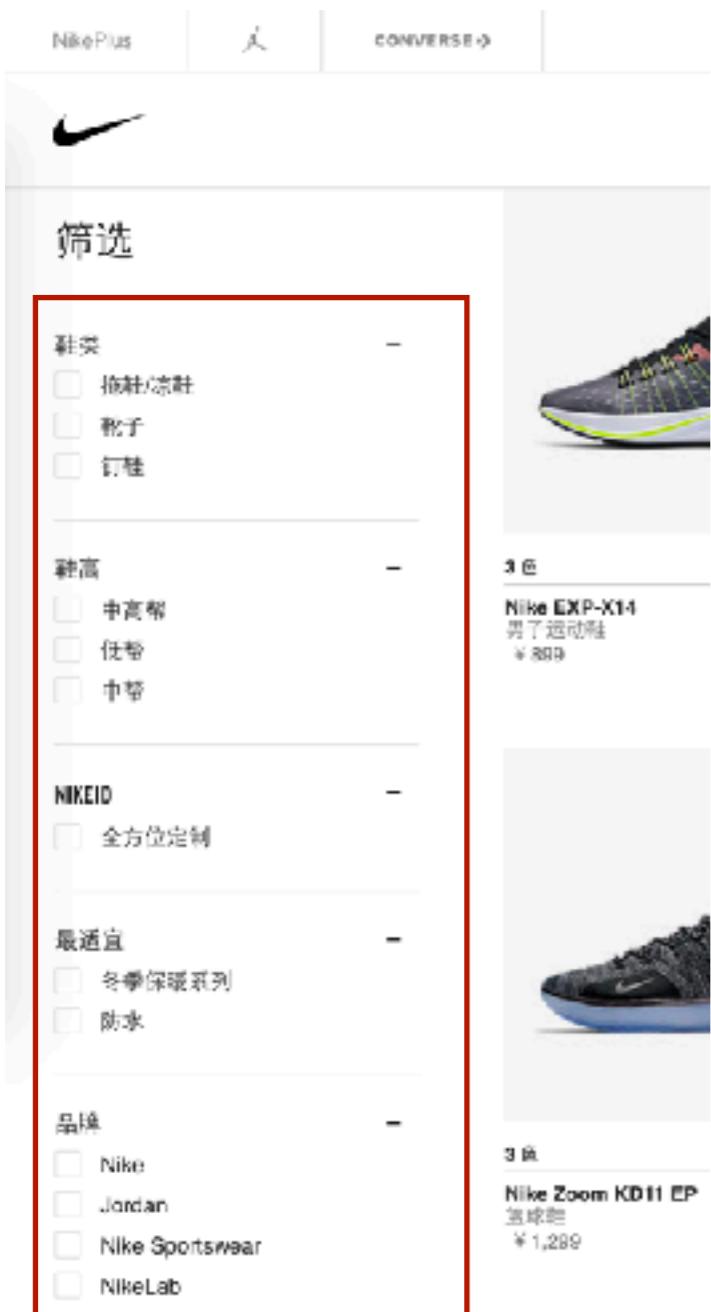
```
# 商品信息  
type Product {  
    id: ID!  
    name: String!  
    description: String  
    image: String  
    price: Float!  
    rating: Rating  
}
```

```
# 商品评分  
type Rating {  
    count: Int!  
    average: Float!  
}
```



BAO ZUN

## Graphql - type system



# 商品分类

```
type Category {
```

# 分类名字

**name**: String!

# 子分类

**subCategory**: [Category!]

# 商品

**products**: [Product!]

```
}
```

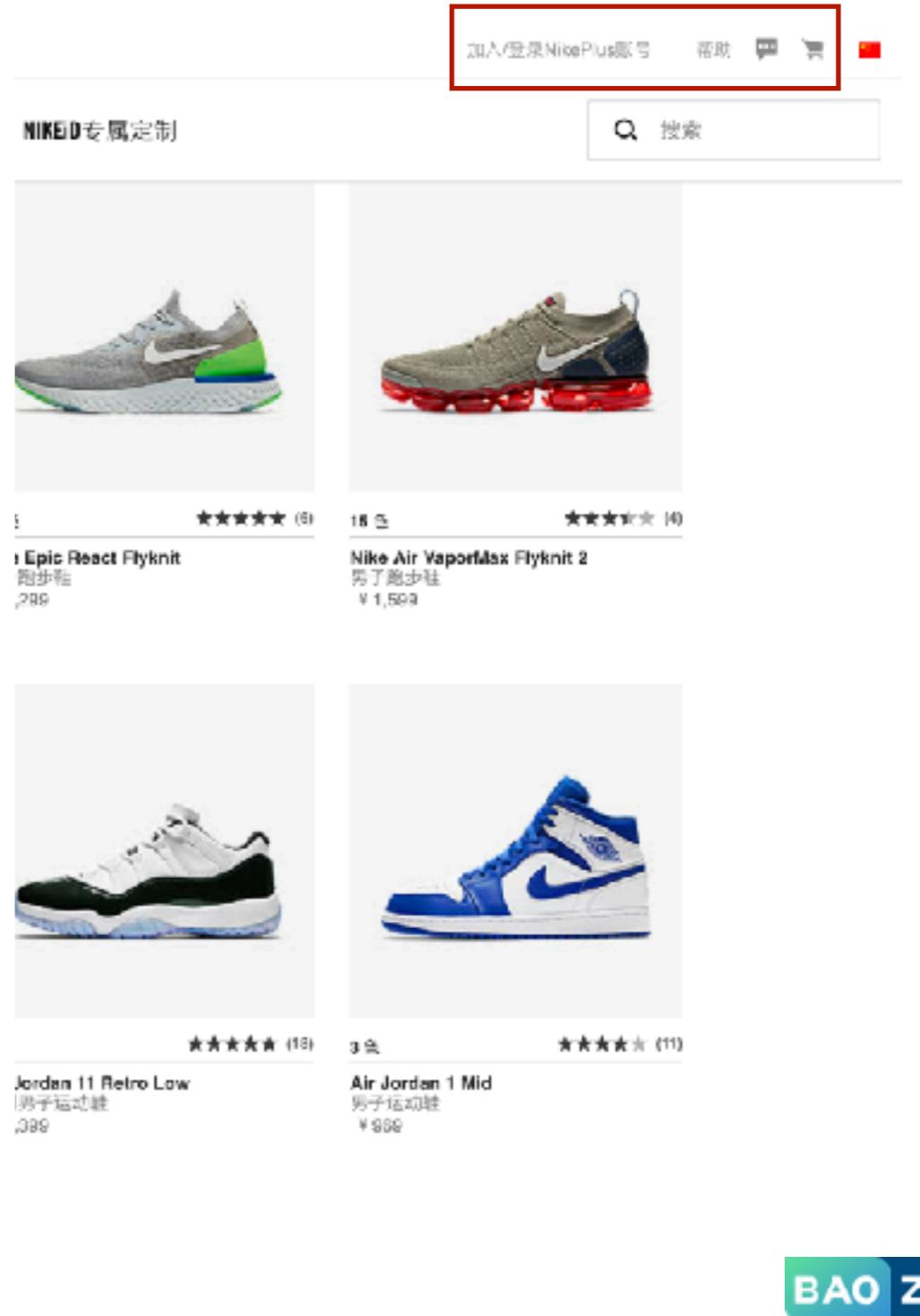
BAO ZUN

## Graphql - type system

```
# 用户信息
type User {
    name: String!
    cart: Cart!
}

# 购物车
type Cart {
    owner: User
    cartItem: [CartItem!]!
}

# 购物车项
type CartItem {
    product: Product!
    # 数量
    amount: Int!
}
```



BAO ZUN

## Graphql - type system

```
# 商品信息  
type Product {  
    id: ID!  
    name: String!  
    description: String  
    image: String  
    price: Float!  
    rating: Rating  
}
```



前端

API



后端

resolver

# Graphql

The screenshot shows a Keynote presentation slide titled "GraphQL - type system". The slide contains the following content:

```
# 用户信息
type User {
    name: String!
    cart: Cart!
}

# 购物车
type Cart {
    owner: User
    cartItem: [CartItem]!
}

# 购物车项
type CartItem {
    product: Product!
    # 数量
    amount: Int!
}
```

Below the schema, there are two rows of product images. The first row shows two Nike Air Max sneakers: "Style: Nike Air Max 90" and "Style: Nike Air Max 95". The second row shows two Nike Air Jordan sneakers: "Style: Air Jordan 1" and "Style: Air Jordan 5".

The Keynote interface is visible, including the toolbar, sidebar, and font panel on the right.

```
type Query {
    # SPU列表搜索
    products: [Product]!
    # 销售属性和商品属性集合
    categories: [Category]!
}
```

## Graphql - type system

The screenshot shows the GraphiQL interface with the following components:

- Left Panel (Schema):** Displays the GraphQL schema with various types and their fields. It includes:
  - Query Type:** products: [Product], user: User
  - Rating Type:** count: Int!, averages: Float!
  - Product Type:** id: ID!, name: String!, description: String, image: String, price: Float!, rating: Rating
  - Category Type:** name: String!, subCategory: [Category!], products: [Product!]
  - User Type:** name: String!, cart: Cart!
  - Cart Type:** owner: User
- Middle Panel (Query):** Shows a query to retrieve product information. The query is:

```
query a {
  products {
    name
    price
  }
}
```

The response is:

```
{
  "data": {
    "products": [
      {
        "name": "Hello World",
        "price": 6.321546348453003
      },
      {
        "name": "Hello World",
        "price": 92.68901891084552
      }
    ]
  }
}
```
- Right Panel (Product Details):** A sidebar with the title "Product" containing:
  - Search bar: Search Product...
  - Section: 商品信息
  - Section: FIELDS
    - id: ID!**
    - name: String!**
    - description: String**
    - image: String**
    - price: Float!**
    - rating: Rating**

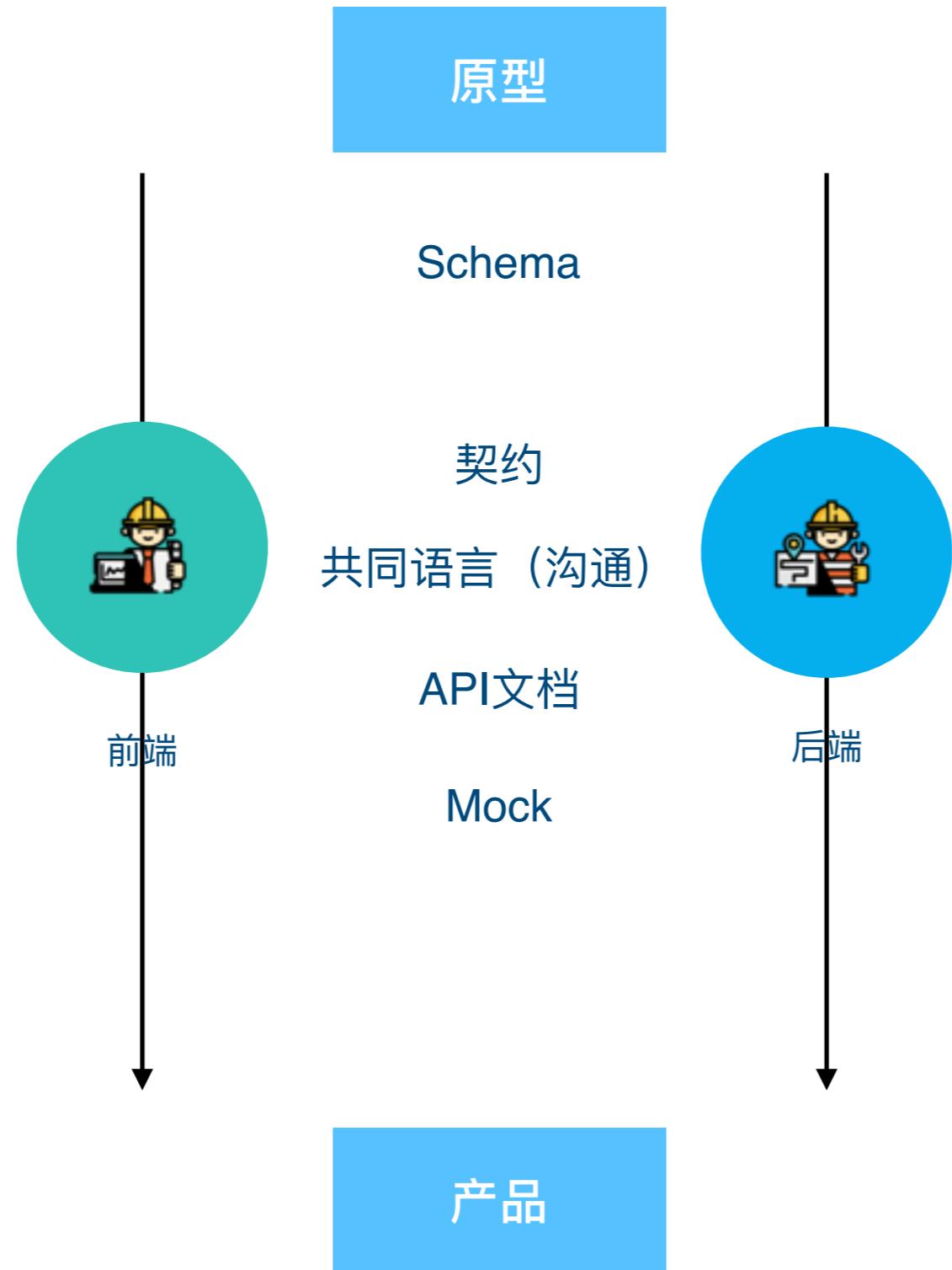
BAO ZUN

# Graphql



BAO ZUN

## Graphql - response 结构



BAO ZUN

## Graphql - response 结构

共同目标 + 相互理解 + 没有埋怨 + 相互独立



BAO ZUN

BAO ZUN

# GraphQL in 前端

- **Relay**
- **Lokka**
- **Apollo**
- ....
- **http post**



```
const Product = ({ product }) => (
  <div>
    <h1>{product.name}</h1>
  </div>
)
```

```
Product.fragment = gql`
```

```
fragment _ on Product {
  name
  description
}
```

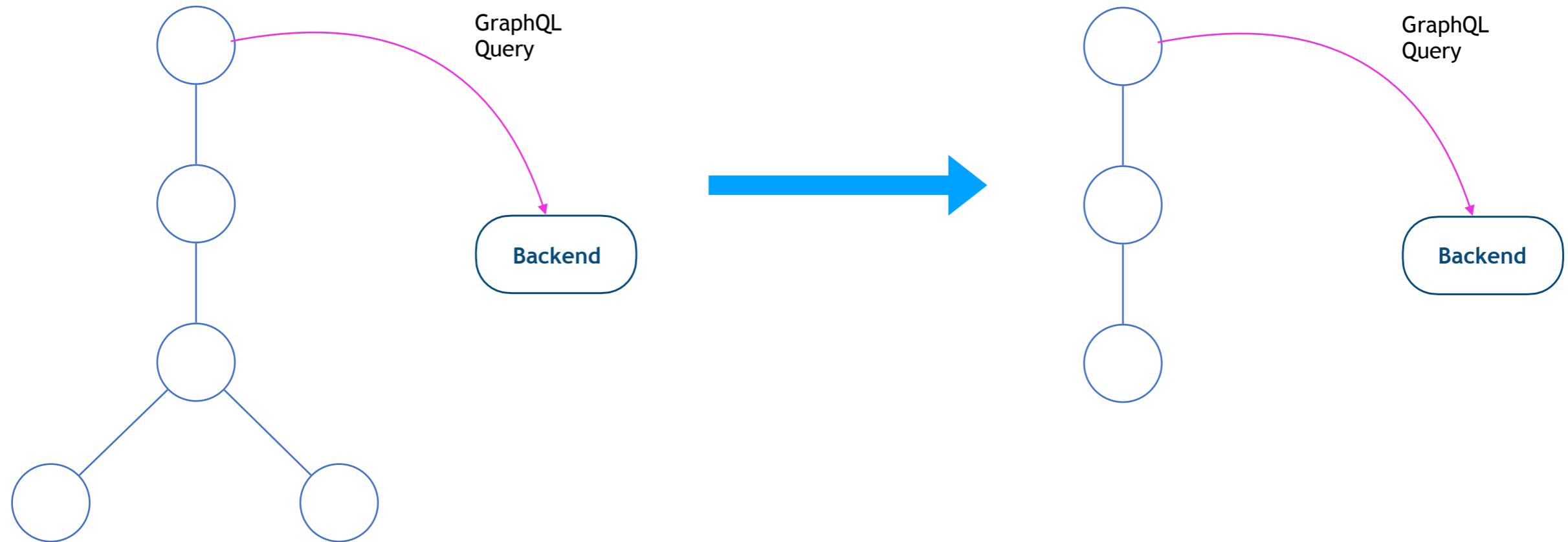
```
,
```

```
const ProductList = ({ products }) => (
  <div>
    products.map(product =>{
      return <Product product={product} />
    })
  </div>
)
```

```
ProductList.fragment = gql`
```

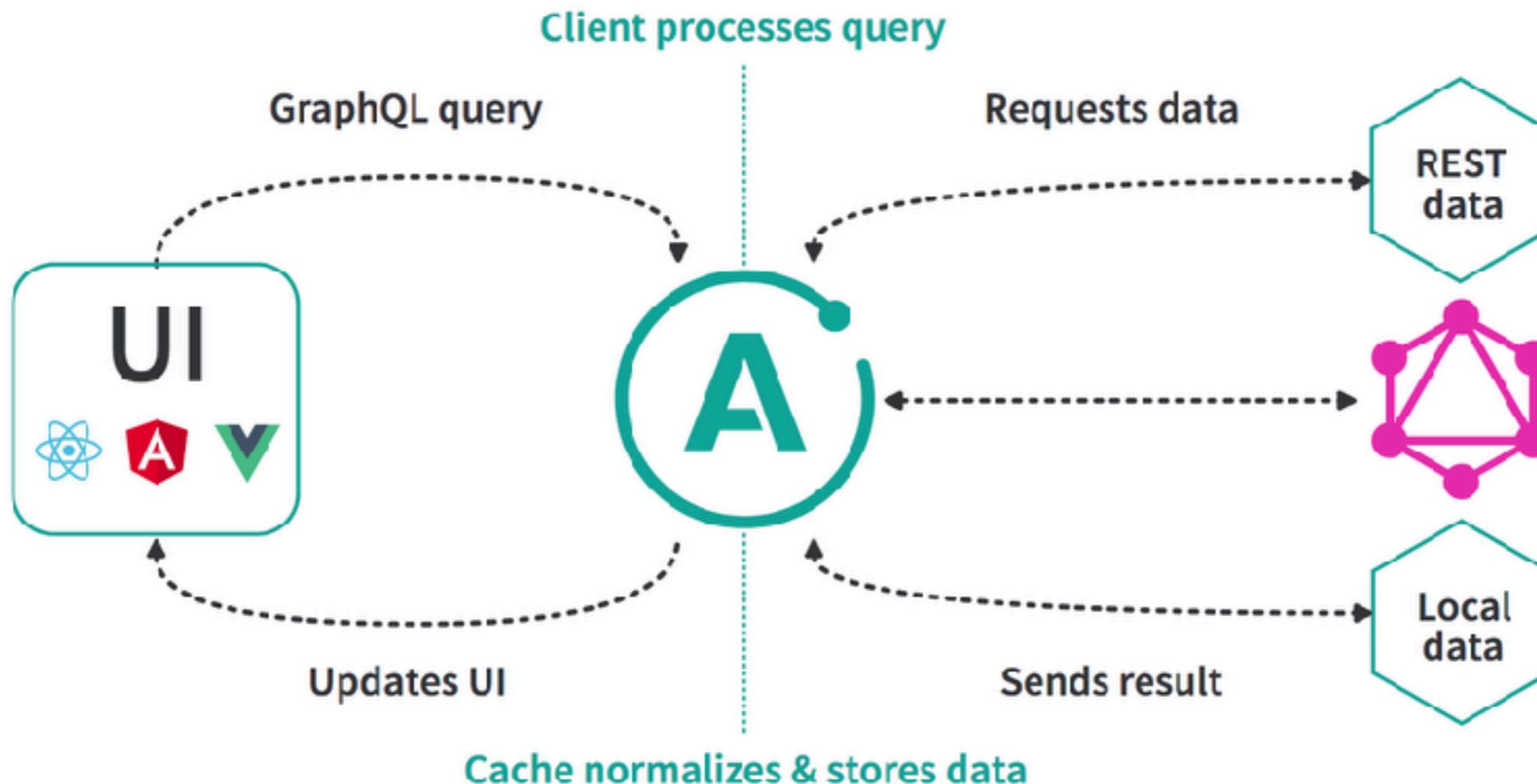
```
fragment _ on ProductEdge {
  node {
    ${Product.fragment}
  }
}
```

# Graphql in 前端



BAO ZUN

问题





不是银弹

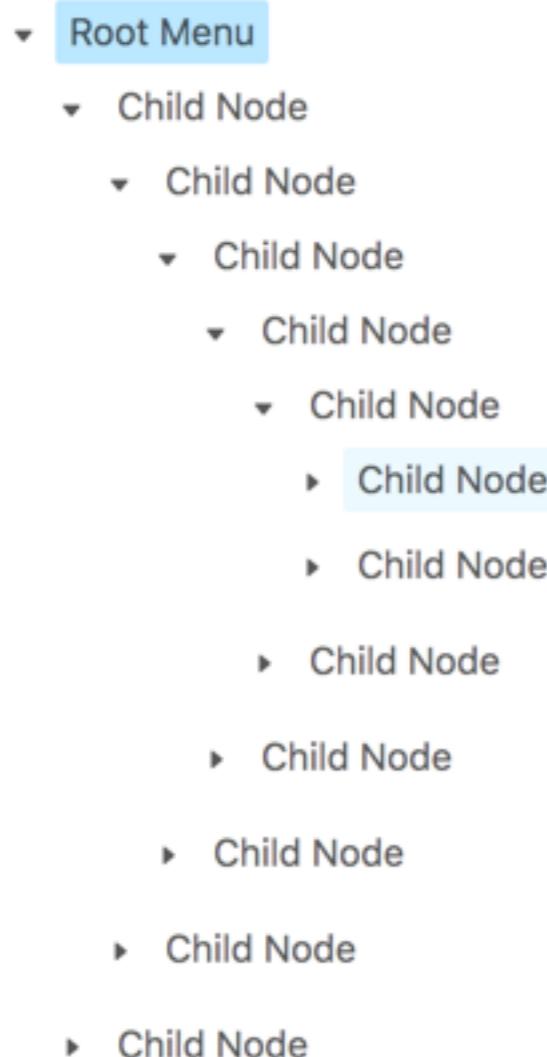


BAO ZUN



问题

## One query get unknown depth tree



```
query menus {  
    shop {  
        menus {  
            name  
            url  
            children {  
                name  
                url  
                children {  
                    name  
                    url  
                    children {  
                        name  
                        url  
                        children {  
                            name  
                            url  
                            ...  
                        }  
                    }  
                }  
            }  
        }  
    }  
}
```

BAO ZUN



问题

文件上传



BAO ZUN

# True or False



# GraphQL 是针对图数据库设计的么？

BAO ZUN



# GraphQL 是针对图数据库设计的么？

不是

BAO ZUN



# GraphQL 与编程语言无关的么？

BAO ZUN



# GraphQL 与编程语言无关的么？

是

BAO ZUN



**GraphQL 可以返回客户端  
希望的数据结构？**



GraphQL 可以返回客户端  
希望的数据结构？

是



# Facebook Relay 是唯一的 GraphQL 客户端么



Facebook Relay 是唯一的  
GraphQL 客户端么

不是



# GraphQL 当道 REST 已死？

# GraphQL 当道 REST 已死？

# 你怎么看？

谢谢