

# 探索 React 生态圈

郭达峰 @dfguo  
Strikingly.com CTO



促进软件开发领域知识与创新的传播



# 实践第一 案例为主

时间：2015年12月18-19日 / 地点：北京·国际会议中心

欢迎您参加ArchSummit北京2015，技术因你而不同



ArchSummit北京二维码



【北京站】

2016年04月21日-23日



关注InfoQ官方信息  
及时获取QCon演讲视频信息

# Geekbang>

极客邦科技

全球领先的技术人学习和交流平台

**InfoQ**<sup>ueue</sup>

专注中高端技术  
人员的社区媒体

**EGO** EXTRA GEEKS' ORGANIZATION  
NETWORKS

高端技术人员  
学习型社交网络

**StuQ**<sup>ueue</sup>

实践驱动的IT职业  
学习和服务平台

扫我，码上开启新世界



# Geekbang>

InfoQ! | EGO NETWORKS | StuQ!

*strikingly*

THE EASIEST  
EDITOR POSSIBLE

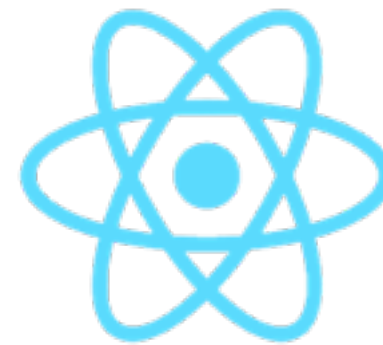


BEAUTIFUL  
MOBILE SITES





ANGULARJS



React

GraphQL **Relay**



webpack  
MODULE BUNDLER

*BABEL*



你们城里人真会玩  
you city people play well

本质原因是什么？

前端正面临着前所未有的  
工程化挑战



# React 生态圈

- 目标平台
- 数据处理
- 语言

# 目标平台

React 生态圈

# Brief History

- 服务器渲染
- 客户端渲染

# 目标平台

- 浏览器作为唯一目标平台
  - 首次打开速度
  - SEO
  - Twitter 11 年时甚至重回服务器渲染方案

目标平台：前端+后端（同构）

Virtual DOM = UI虚拟机



目标平台： iOS & Android

React Native

# 目标平台： iOS & Android

- Facebook Ads Manager
  - 85% 代码重用

目标平台：TUI

Left			Right		
Name	Size	Modify time	Name	Size	Modify time
../	340	Jun 05 15:45	../	340	Jun 05 15:45
/assets	136	Jun 24 15:20	/assets	136	Jun 24 15:20
/bin	102	Jun 05 12:13	/bin	102	Jun 05 12:13
/client	170	Jun 24 15:20	/client	170	Jun 24 15:20
/lib	170	Jun 24 15:20	/lib	170	Jun 24 15:20
/node_modules	1870	Jun 24 15:44	/node_modules	1870	Jun 24 15:44
/server	238	Jun 24 15:20	/server	238	Jun 24 15:20
/shared	136	Jun 24 15:20	/shared	136	Jun 24 15:20
AUTHORS	36	May 19 19:07	AUTHORS	36	May 19 19:07
CHANGELOG.md	35	May 19 19:07	<b>CHANGELOG.md</b>	<b>35</b>	<b>May 19 19:07</b>
CONTRIBUTION.md	1141	May 19 19:07	CONTRIBUTION.md	1141	May 19 19:07
LICENSE	1101	May 19 19:07	LICENSE	1101	May 19 19:07
package.json	2495	Jun 24 15:44	package.json	2495	Jun 24 15:44
README.md	1279	Jun 24 15:20	README.md	1279	Jun 24 15:20
webpack.config.dev.js	316	Jun 24 15:20	webpack.config.dev.js	316	Jun 24 15:20

1 Help   2 Menu   3 View   4 Edit   5 Copy   6 RenMov   7 Mkdir   8 Delete   9 PullDn   10 Quit



The screenshot shows a web browser window at localhost:8181 with a terminal window open in the background. The terminal displays a file explorer view with columns for Name, Size, and Modify time. The browser's developer tools are open, showing a list of CSS properties on the left and a corresponding code snippet on the right. The code snippet includes attributes like style, data-reactid, and data-reactid.

Name	Size	Modify time
..	340	Jun 05 15:45
/assets	136	Jun 24 15:20

Terminal Output:

```

ls
node_modules
package.json
README.md
CONTRIBUTION.md
LICENSE

```

Developer Tools - Styles:

- backface-visibility
- background
- background-attachment
- background-blend-mode
- background-clip
- background-color
- background-image
- background-origin
- background-position
- background-position-x
- background-position-y
- background-repeat
- background-repeat-x
- background-repeat-y
- background-size
- baseline-shift
- background-color;

Developer Tools - Sources:

```

style="width: 112px; height:
: 0px 0px 0px; transform:
data-reactid=".0">
ta-reactid=".0.0">_</ul>
data-reactid=".0.0.0">_</ul>
ls" data-reactid=".0.0.1">
nel panel_selected" data-
:leftPanel">
anel_caption" data-reactid=
Panel.0">/Users/azproduction/Pro
-mc</div>
le-list" data-reactid=
anel.1">

```





The screenshot shows a development environment with three main components:

- File Explorer (Top):** Two windows showing a directory listing for `/Users/azproduction/Projects/my/node-mc`. The files listed are:
 

Name	Size	Modify time
..	340	Jun 05 15:45
/assets	136	Jun 24 15:20
- Browser (Middle):** A window titled "mc" at `localhost:8181`. The browser content shows a terminal-like interface with a table of files and a sidebar.
- Code Editor (Bottom):** A code editor showing HTML/JSX code with a DOM inspector on the right. The selected element is:
 

```

      <div class="panel panel_selected" data-reactid=".0.0.1.$leftPanel" style="background-color: red;">
      </div>
      
```

 The DOM tree shows the following structure:
 

```

      <body>
      <div id="app">
      <div class="tui-dom" style="width: 112px; height: 21px; transform-origin: 0px 0px 0px; transform: scaleX(5) scaleY(7);" data-reactid=".0">
      <div class="mc" data-reactid=".0.0">
      <ul class="menu" data-reactid=".0.0.0">_</ul>
      <div class="panels" data-reactid=".0.0.1">
      <div class="panel panel_selected" data-reactid=".0.0.1.$leftPanel" style="background-color: red;">
      <div class="panel_caption" data-reactid=".0.0.1.$leftPanel.0">/Users/azproduction/Projects/my/node-mc</div>
      
```





# 其他平台

- TUI
- React canvas
- React three (3D)
- React art

# 多目标平台

- Virtual DOM = UI虚拟机
- 目标平台不再只是浏览器
  - 解决首次打开速度和 SEO 问题
  - 其他客户端成为可能性



想想都有点小激动  
Excited!

# 数据处理

React 生态圈

# 数据处理

- Flux
- GraphQL / Relay

# GraphQL

What is this?



GraphQL是RESTful API之外的另一个选择，在**某些情况下**，是更好的选择。

# REST API 的问题



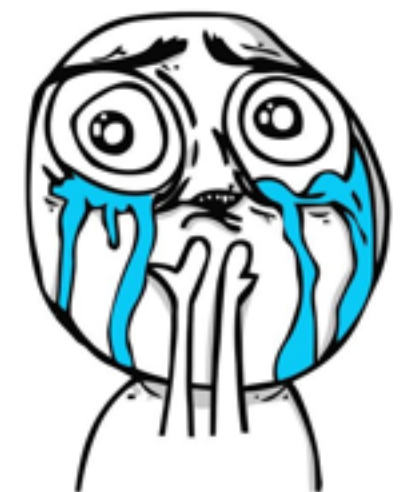
The screenshot shows a Strikingly website management interface. On the left is a light blue square with the word "strikingly" in white. To its right, the site title "Growth Hacking Sharing from Strikingly" is displayed in a large, dark font, with a green "已发布" (Published) badge next to it. Below the title, the URL "growth-dafeng.strikingly.com" is shown in blue, followed by "更新于4天前" (Updated 4 days ago). A row of four buttons is positioned below the URL: a green "编辑" (Edit) button with a pencil icon, a grey "数据" (Data) button with a bar chart icon, a grey "表单" (Forms) button with a list icon, and a grey settings gear icon. On the far right, a white box with a thin border contains the number "287" in a large font, with "访问量 过去7天" (Visits in the last 7 days) written below it in a smaller font.

- 两个分开的请求
  - /api/v1/site/{id}
  - /api/v1/site/{id}/analytics

# REST API 的问题

- Custom endpoint: `/api/v1/site_info/{id}`
  - 各种资源的组合, 不好维护
- Over-fetching
  - iOS 客户端只需要部分信息 (粒度不同)
- 客户端版本问题

每次需求的变更  
都需要后端修改



“The REST interface is designed to be efficient for *large-grain* hypermedia data transfer...”

- Dr Roy T Fielding, Author of REST

REST更适用于大粒度数据场景

不同的UI设计和需求导致了对数据有更小粒度控制的需求。





GraphQL是RESTful API  
之外的另一个选择，在对  
数据有更小粒度控制需求  
的时候，是更好的选择。



# GraphQL at Facebook

- 3 years
- 26 billion request / day

# GraphQL Client-side

- GraphQL as a unified query interface - only 1 endpoint
- Describe the shape of the data

# GraphQL Query

Code

```
{
  user(id: 3500401) {
    id,
    name,
    isViewerFriend,
    profilePicture(size: 50) {
      uri,
      width,
      height
    }
  }
}
```

# JSON Response

Code

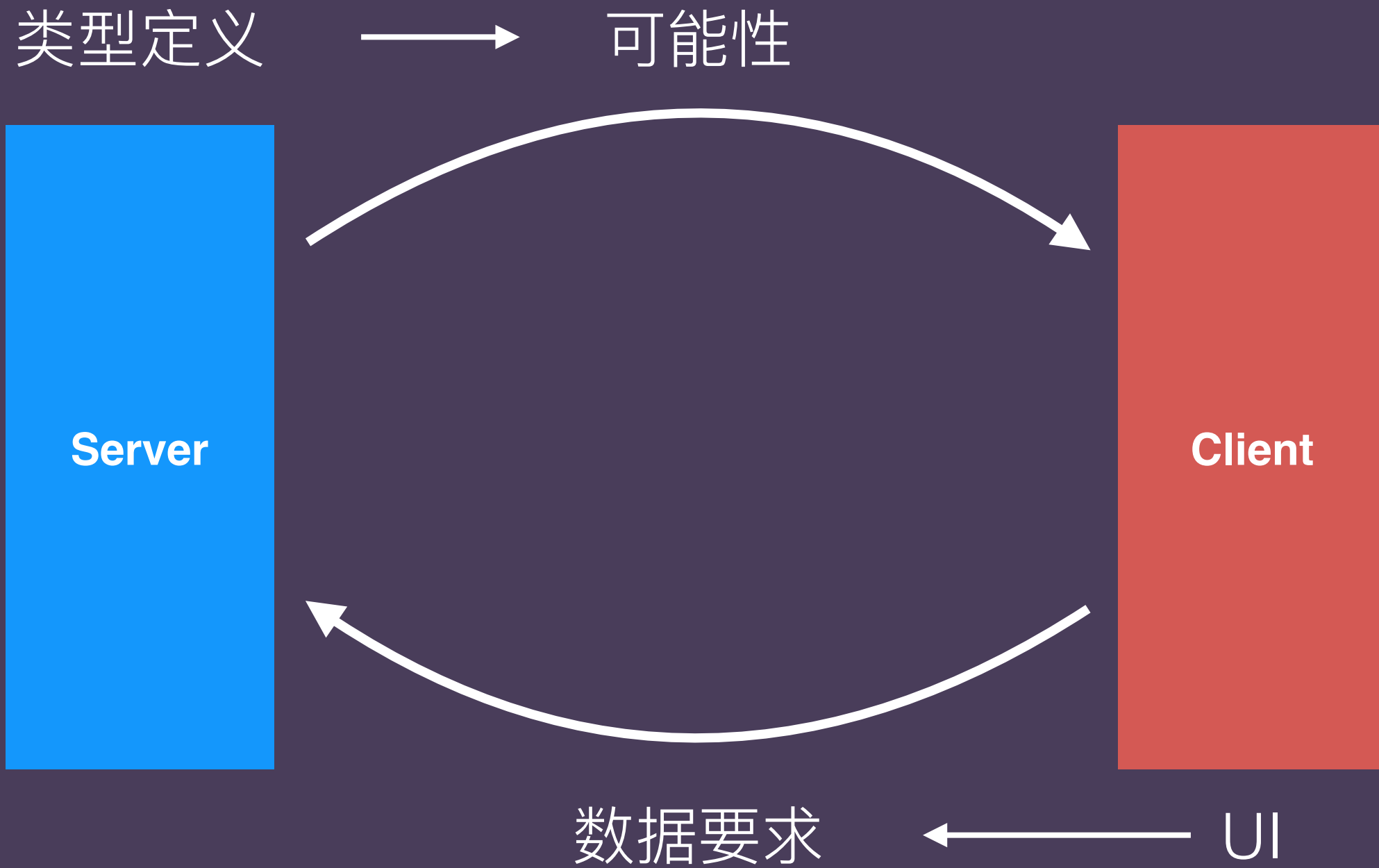
```
{
  "user" : {
    "id": 3500401,
    "name": "Jing Chen",
    "isViewerFriend": true,
    "profilePicture": {
      "uri": "http://someurl.cdn/pic.jpg",
      "width": 50,
      "height": 50
    }
  }
}
```

# GraphQL Server-side

GraphQL核心

类型定义

# Big Picture 逼格图



# REST API 的问题都被解决了

- Custom endpoint
- Over-fetching
- 客户端版本问题

每次需求的变更  
不需要后端修改





# Relay

Using GraphQL in React

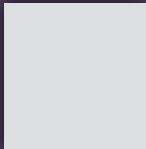
# Component

- Component is at the core of React
- Reusing any component, we are reusing
  - HTML - React
  - Javascript - React
  - CSS - Maybe React?
  - Data - Relay

# Relay

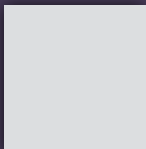
Comments (3)

---



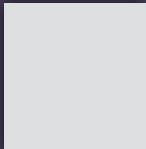
**Nate**  
This is a great speech!

Reply



**Kevin**  
This is a great speech!

Reply



**Bood**  
This is a great speech!

Reply

Leave a comment...

Submit

```
<CommentBox>  
  <Header /> GraphQL Query  
  <CommentList /> GraphQL Query  
  <SubmitForm />  
</CommentBox>
```

# Relay

- Co-locate data definition with React component
- Using GraphQL to describe the data
- Automatically batch and fetch the data for you

# Other solutions

- Relay + GraphQL
- Falcor + JSONGraph
- Om Next
- Meteor

# 语言

React 生态圈

# 语言

- Node.js
  - CommonJS
  - NPM
- ES6

Immutable.js



```
1 var obj1 = {
2   val1: 1,
3   val2: 2
4 }
5 var obj2 = obj1
6 obj2.val1 = 2
7
8 obj1 === obj2 // true
```

```
1 var obj1 = Immutable.fromJS({
2   val1: 1,
3   val2: 2
4 })
5 var obj2 = obj1.set('val1', 2)
6
7 obj1 === obj2 // false
```

# Immutable.js

- Immutable - changes create new copy on every change
- Persistent - old copies will be kept
- Structural sharing - new/old objects shares memory

# Immutable.js

- Om
  - a clojure wrapper of React.js
  - faster than React.js?! Why?

# Immutable.js

## Code

```
shouldComponentUpdate: function(nextProps, nextState) {  
  return nextProps.id !== this.props.id;  
}
```

- high performant React.js application
- compiler friendlier
- Immutability might become part of JavaScript language in the future

# flow

Static typed JavaScript (opt-in)

CSS

CSS

IS

AWESOME

# CSS Problems

1. Global namespace
2. Dependencies
3. Dead code elimination
4. Minification
5. Sharing constants
6. Non-deterministic resolution
7. Isolation



# CSS in JavaScript

```
/* button.js */
```

```
var styles = {  
  container: {  
    background: '#f6f7f8 url(/images/button/background.png) repeat-x',  
    border: '1px solid #cdced0',  
    borderRadius: 2,  
    boxShadow: '0 1px 1px rgba(0, 0, 0, 0.05)',  
  },  
  depressed: {  
    backgroundColor: '#4e69a2',  
    borderColor: '#c6c7ca',  
    color: '#5890ff',  
  },  
};
```

```
<div style={styles.container}> JSX
```

# CSS in JavaScript

1. Global namespace - solved
2. Dependencies - solved
3. Dead code elimination - solved
4. Minification - solved
5. Sharing constants - solved
6. Non-deterministic resolution - solved
7. Isolation - solved

# CSS in JavaScript

- Not without its own problems
  - media queries
  - hover state
- Ongoing work: Radium, react-style, css-module

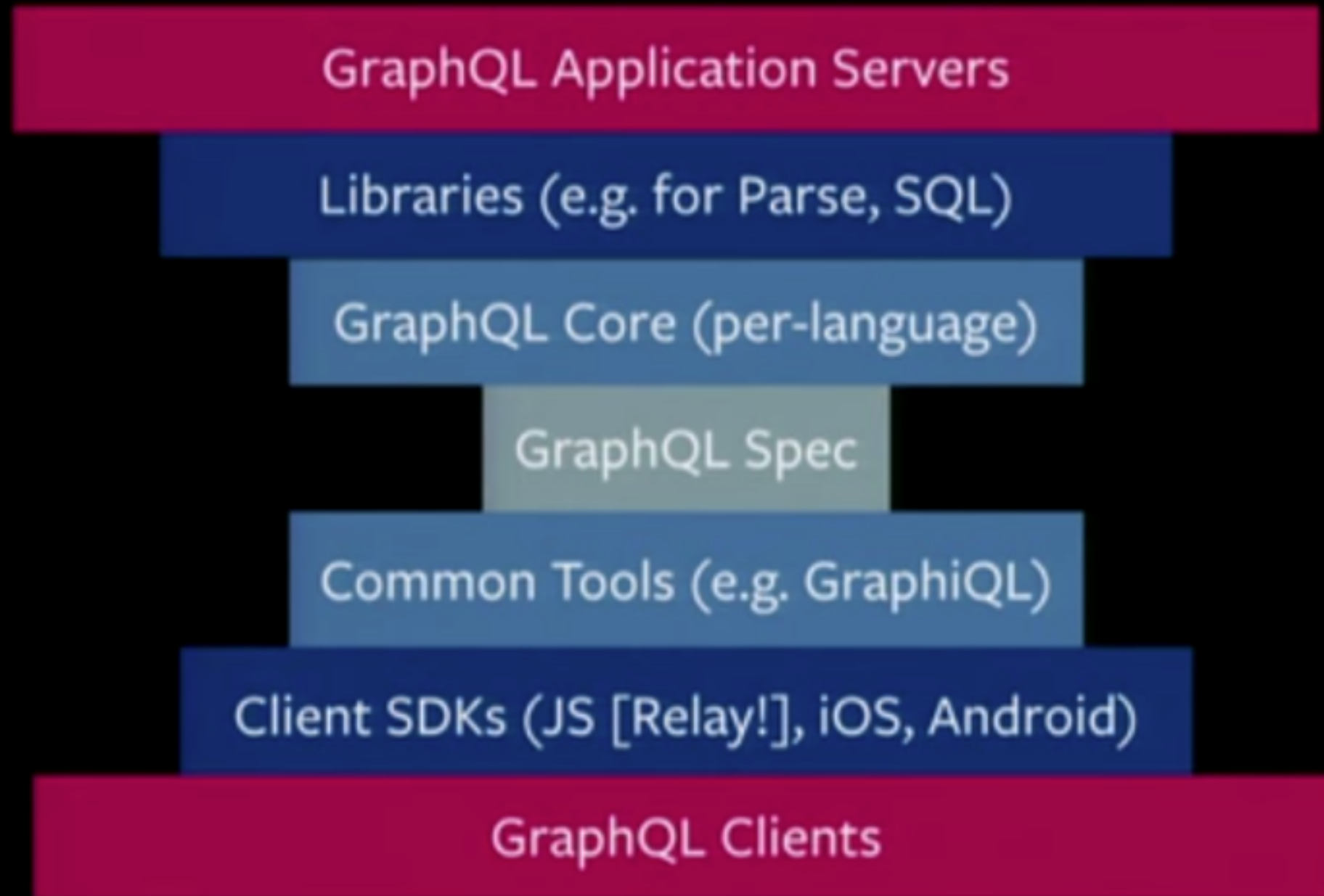
# 探索 React 生态圈

- 目标平台
  - 前后端同构
  - Virtual UI Machine
- 语言
  - ES6
  - Immutable.js
  - Flow
  - CSS in JavaScript
- 数据处理
  - Flux
  - GraphQL / Relay 解决 REST API 的问题

# Q&A



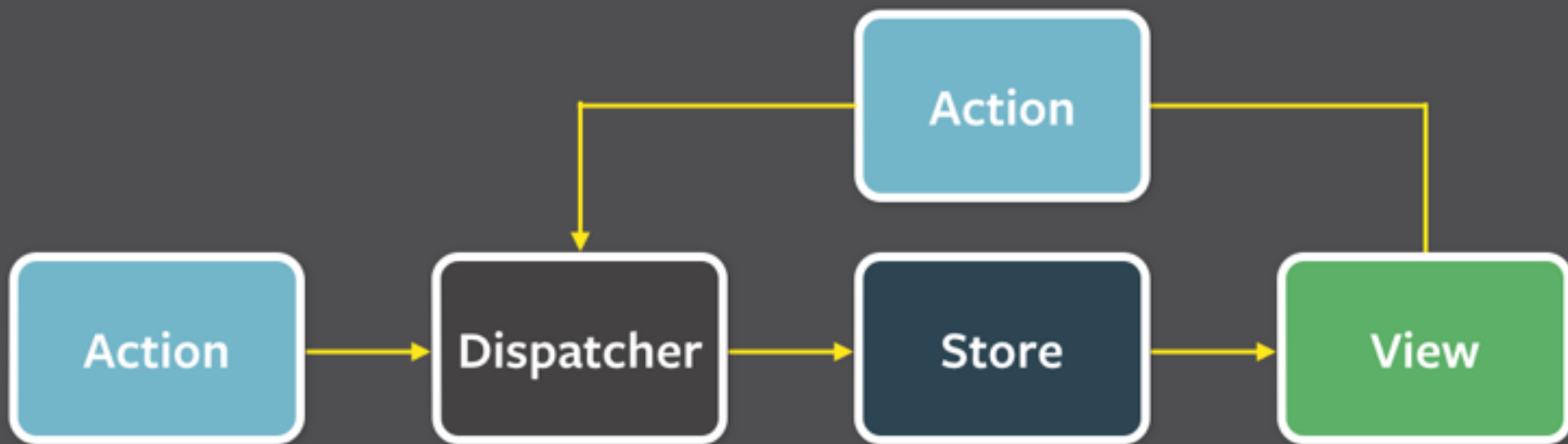
# GraphQL Ecosystem



# 数据处理

- Flux
  - unidirectional data flow
  - single source of truth

# Flux





# 工具

React 生态圈

- Webpack
- Babel

Dev tool

React hot load

# 工具

React 生态圈

# Webpack

- webpack 是一款强大的前端模块管理和打包工具。这里列出他的一些特性：
  - 支持 CommonJS / AMD
  - 灵活和可扩展的 loader（加载器）机制
  - 支持对 CSS，图片等其他资源进行打包
  - 内置强大的 Code splitting 功能可以拆分并动态加载包
  - 开发模式支持 hot module replacement 模式，提高开发效率