ECUG Con 十周年盛会

Vue.js 2.0 后端渲染实践

阴明

@kalasoo

掘金创始人、前端开发

Agenda

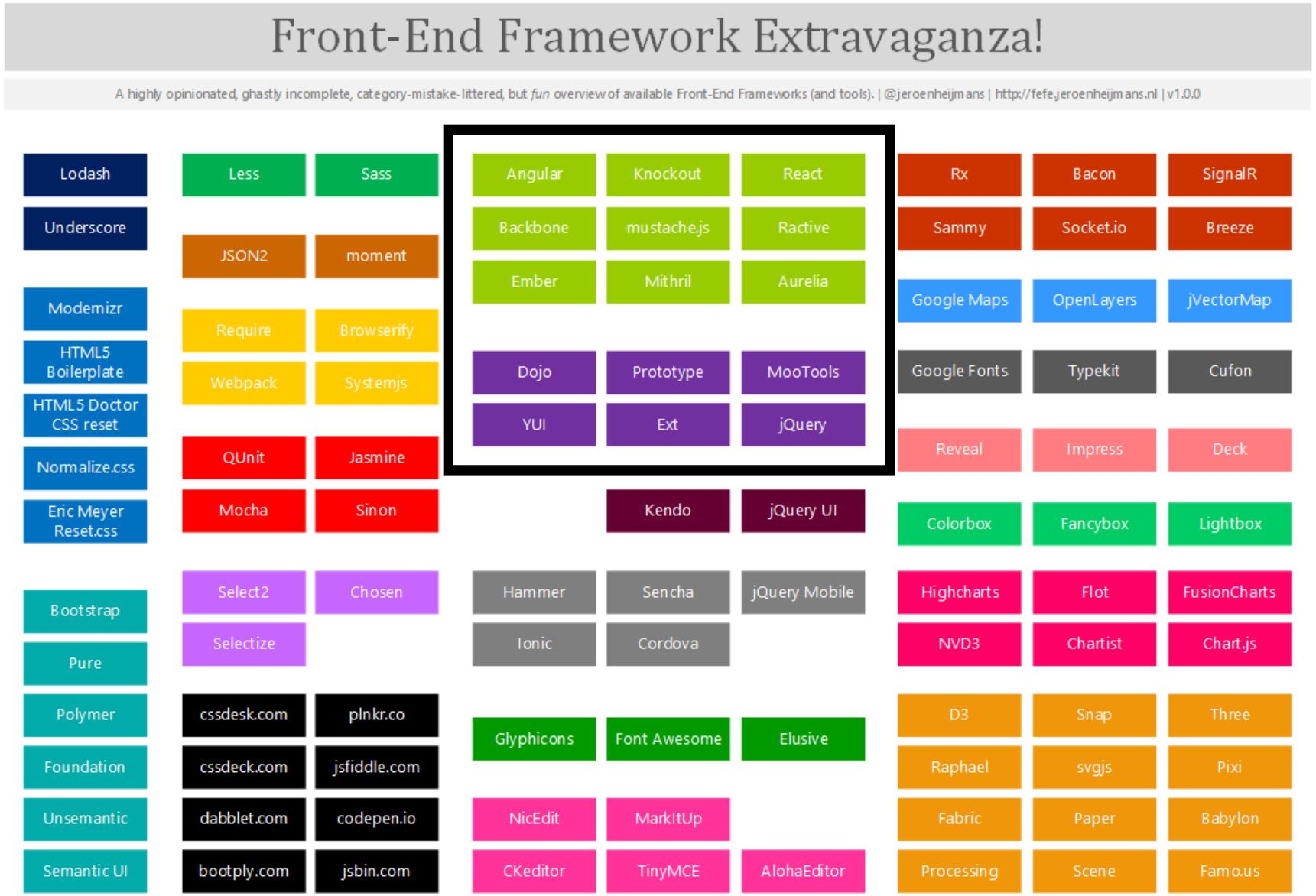
- 前端框架的繁荣及成熟
- Why Vue.js
- 纯前端应用的弊端
- Vue.js 2.0 后端渲染
- 后端渲染 Nuxt.js 的开发实践

前端框架的繁荣

Vue.js, React.js, Angular.js

百家争鸣到三足鼎立

这个是前端的状态


Front-End Framework Extravaganza!
A highly opinionated, ghastly incomplete, category-mistake-littered, but fun overview of available Front-End Frameworks (and tools). | @jeroenheijmans | http://fefe.jeroenheijmans.nl | v1.0.0
Lodash
Underscore
Modernizr
HTML5 Boilerplate
HTML5 Doctor CSS reset
Normalize.css
Eric Meyer Reset.css
Bootstrap
Pure
Polymer
Foundation
Unsemantic
Semantic UI
Less
Sass
JSON2
moment
Require
Browserify
Webpack
Systemjs
QUnit
Jasmine
Mocha
Sinon
Select2
Chosen
Selectize
cssdesk.com
plnkr.co
cssdeck.com
jsfiddle.com
dabblet.com
codepen.io
bootply.com
jsbin.com
Angular
Knockout
React
Backbone
mustache.js
Ractive
Ember
Mithril
Aurelia
Dojo
Prototype
MooTools
YUI
Ext
jQuery
Kendo
jQuery UI
Hammer
Sencha
jQuery Mobile
Ionic
Cordova
Glyphicons
Font Awesome
Elusive
NicEdit
MarkItUp
CKeditor
TinyMCE
AlohaEditor
Rx
Bacon
SignalR
Sammy
Socket.io
Breeze
Google Maps
OpenLayers
jVectorMap
Google Fonts
Typekit
Cufon
Reveal
Impress
Deck
Colorbox
Fancybox
Lightbox
Highcharts
Flot
FusionCharts
NVD3
Chartist
Chart.js
D3
Snap
Three
Raphael
svgjs
Pixi
Fabric
Paper
Babylon
Processing
Scene
Famo.us

JavaScript in 2016

> …
>
> 你看其实很简单。用 Typescript 写代码，用 Fetch 发起异步请求，所有代码编译成 ES6 ， 然后用上 Babel 的 stage-3 配置项，把 ES6 转译成 ES5 。所有代码用 SystemJS 加载。如果你用不了 Fetch ， 就加个 polyfill ， 或者用 Bluebird 、 Request 或者 Axios ， 这样你就可以用 await 来处理 Promise 了。
>
> …

来源：https://www.v2ex.com/t/310767?p=2

Web 技术和 JavaScript 到达各个领域

- 后端：Node.js #v8powered
- 移动：Hybrid, React Native, NativeScript, Weex
- 桌面：Electron, nw.js
- VR：WebVR, A-Frame, WebGL
- 硬件：Cylon.js, Tessel, Johnny-Five
- 数据可视化：d3.js, vis.js, HighCharts, Charts
- ……

三足鼎立

Vue.js

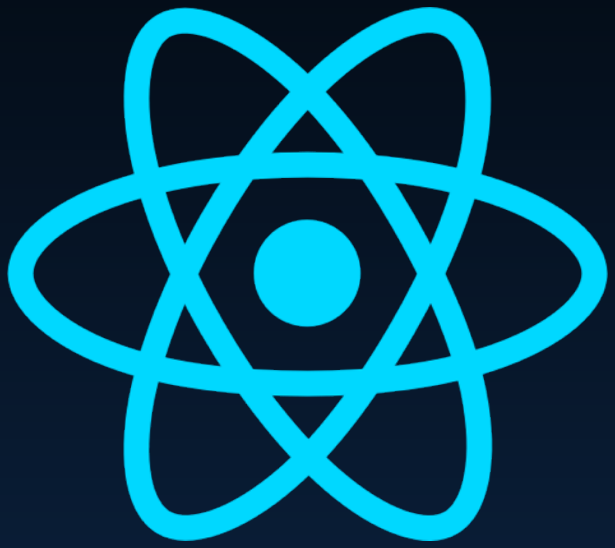
-

Evan You / Alibaba?

Angular.js

-

Google

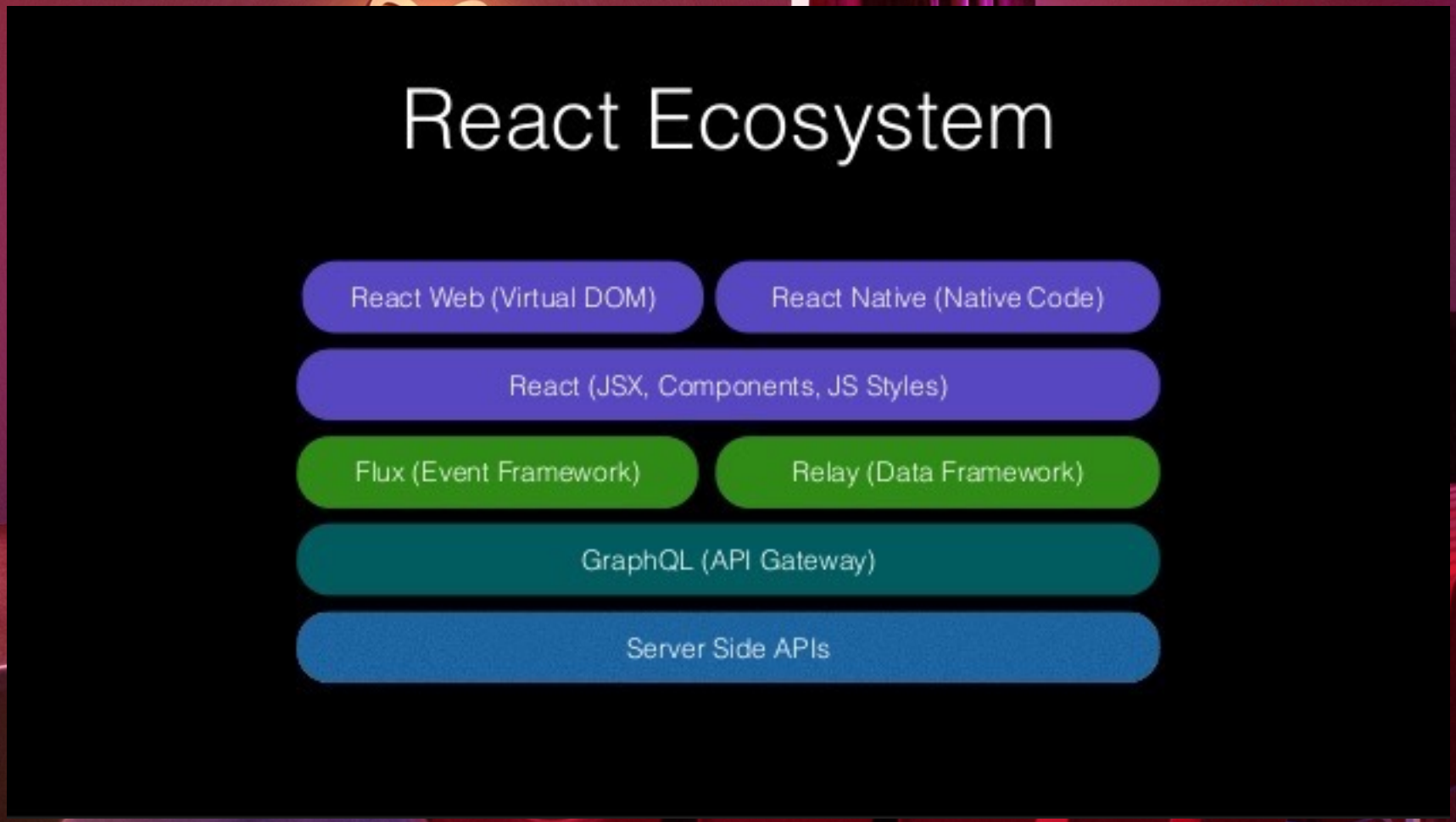


React.js

-

Facebook

每个框架甚至都有了自己的技术生态



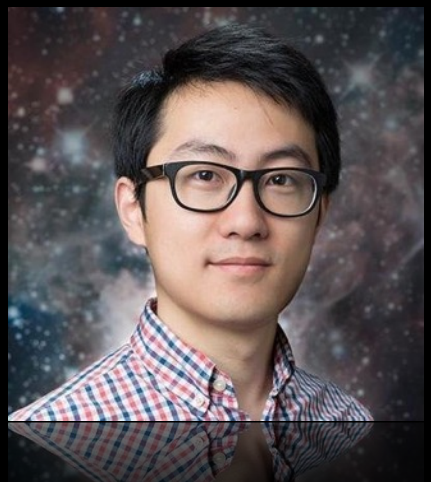
Why Vue.js?

我们很偶然但也很幸运的选择了 Vue.js

0.12.X => 2.X

Vue.js / About

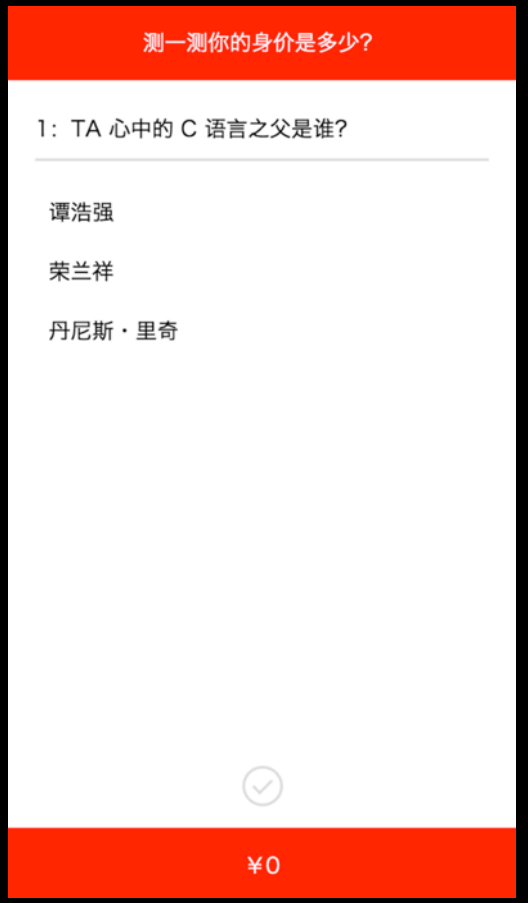
- The Progressive JavaScript Framework
- 由尤雨溪开发并开源
- GitHub 上有超过［3万5千］个 ⭐
- NPM 上每月会有超过［22万］次 ⏬



我用 Vue.js 做的第一个东西

coder-price

http://xitu.github.io/coder-price/



Vue.js 主要是做啥的?

1.0 它是一个前端 MVVM 框架

Model / View / ModelView

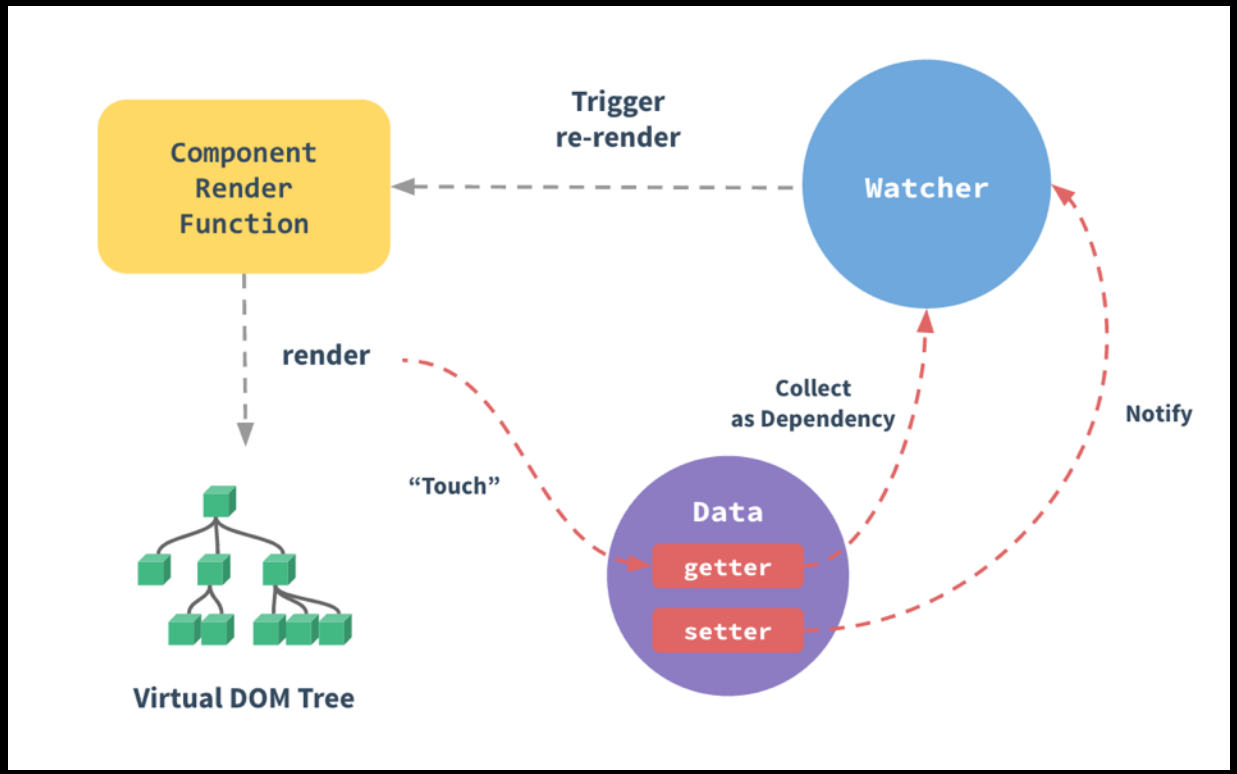
```
UI = VM(State)
```

Vue.js 主要是做啥的?

2.0 一个渐进式前端解决方案

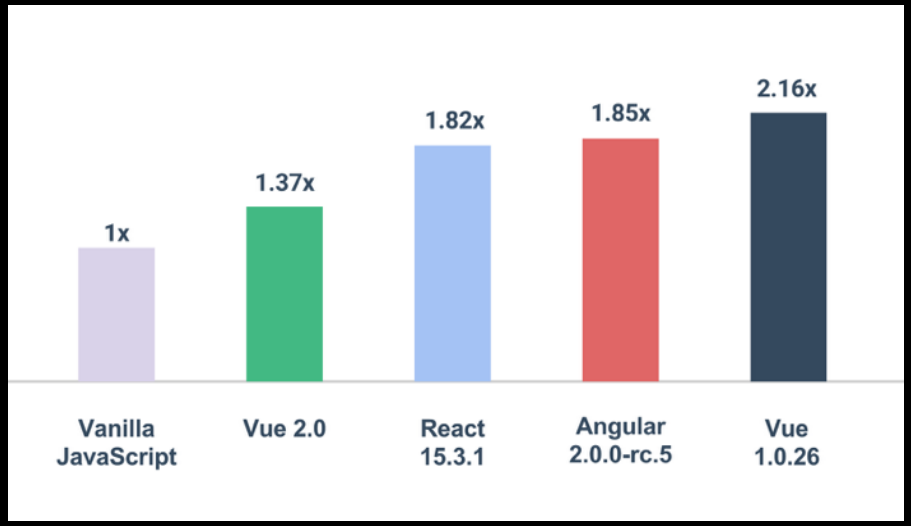
Vue.js 原理

- `Object.defineProperty`

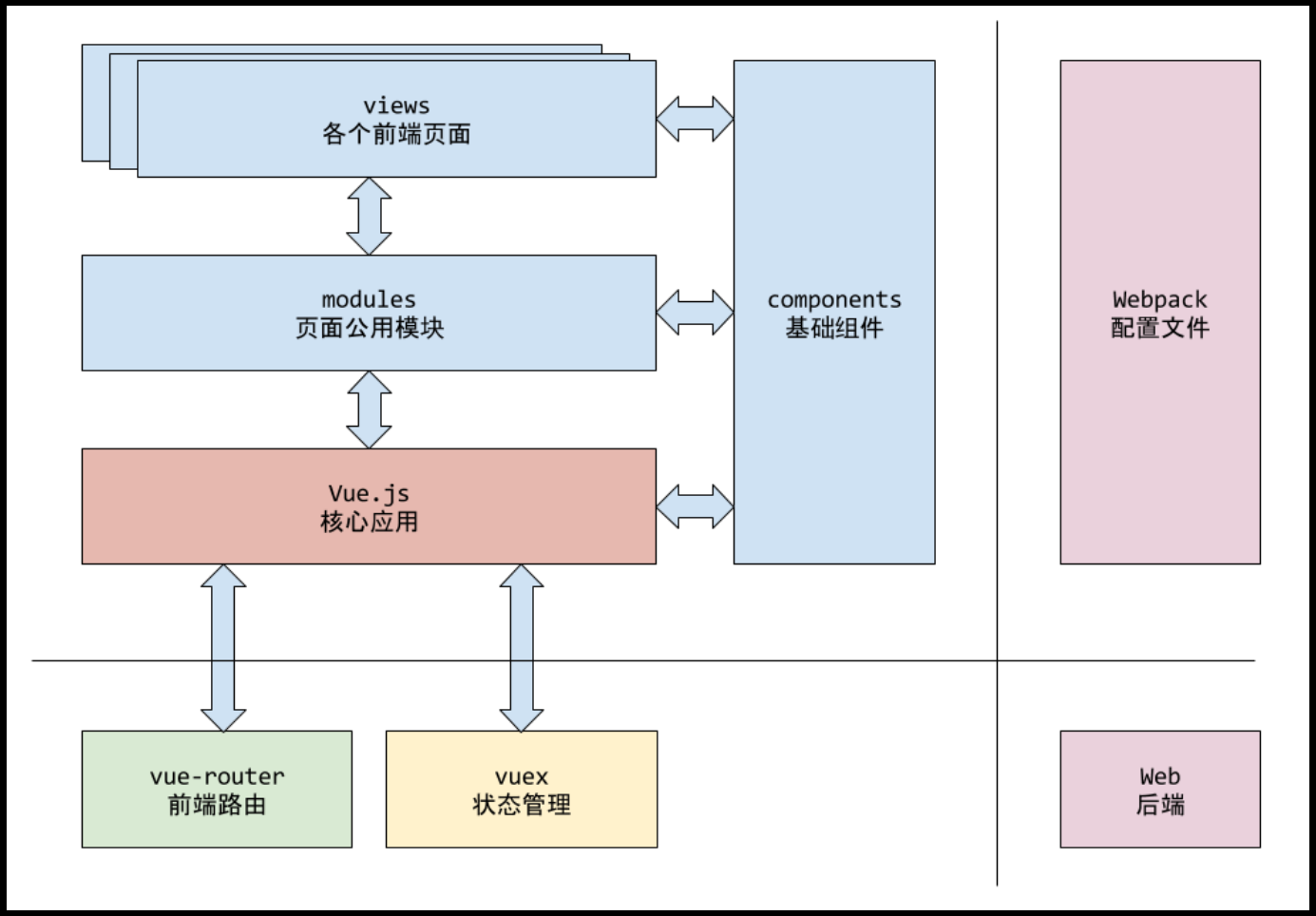


选择 Vue.js 框架

- 开发效率：Declarative Rendering
- 代码维护：组件化 vue-loader
- 速度性能：要能满足需求



掘金 Vue.js 架构


views
各个前端页面
modules
页面公用模块
Vue.js
核心应用
components
基础组件
Webpack
配置文件
vue-router
前端路由
vuex
状态管理
Web
后端

纯前端应用弊端及问题

🔒兼容问题🔒

😭百度搜索😭

⚡速度性能⚡

兼容性问题

 IE ... 8 

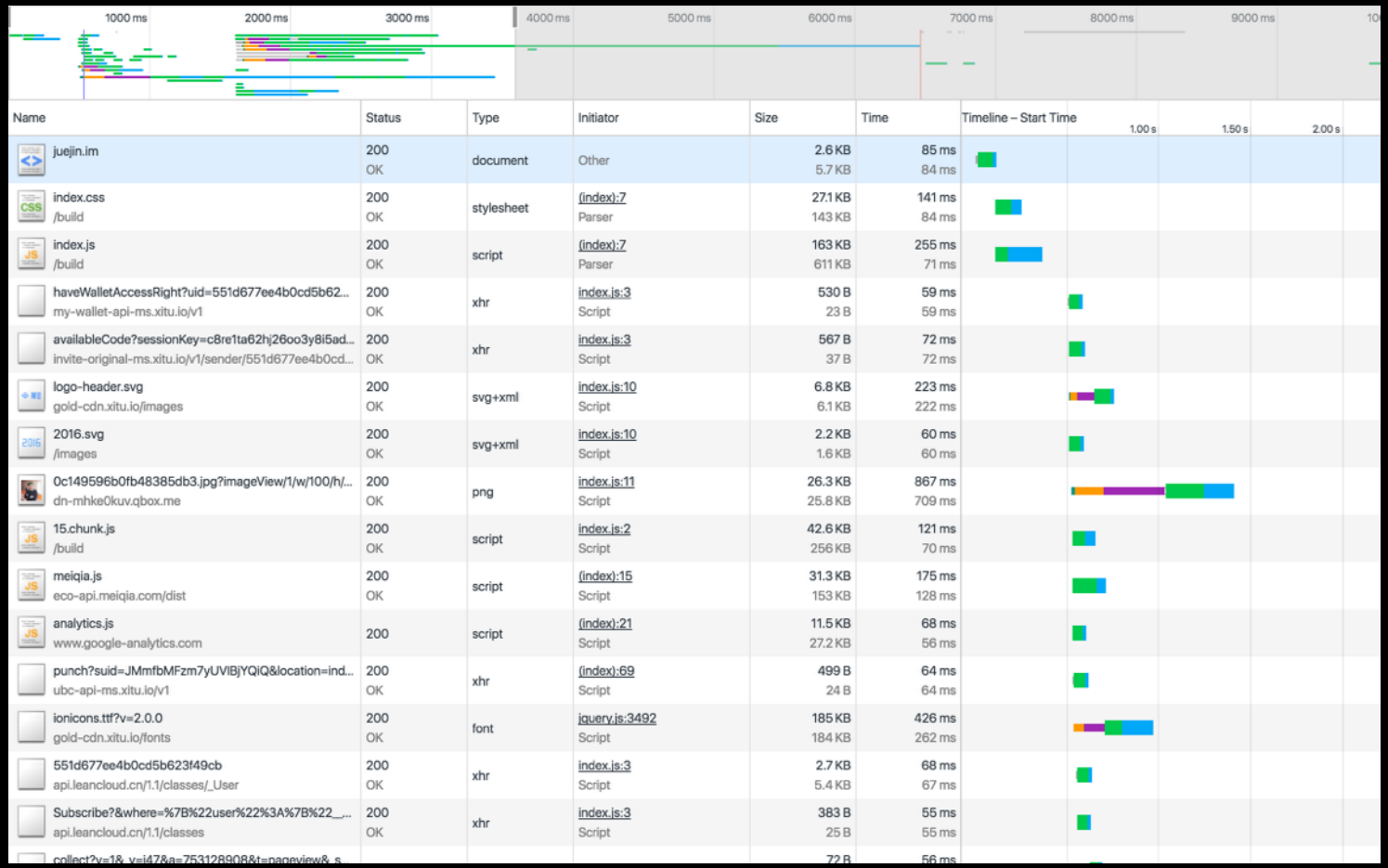
Vue.js, React.js, Angular.js: IE9+

SEO

Google 👍

Baidu 👎

Initial Loading



URL <=> Content Cache

- 并不是每一个资源获取都有固定 URL
- 每一个页面不一定都有 URL
- 主流的 Cache URL 模式很难执行

Vue.js 2.0 后端渲染

Render Function

Stream

A Simple Vue.js Program

```
<ul :id='id'>
  <li v-for:'item in items'>{{item}}</li>
<ul>
// A simple vue program bound to <ul>
var vue =  new Vue({
  el: 'ul',
  data: {
    id: 'myId',
    items: ['Item 1', 'Item 2']
  }
})
```

Virtual DOM

- Vue.js 2.0 在 DOM 渲染上完全适用 Virtual DOM

```
<ul id='myId'>
  <li>Item 1</li>
  <li>Item 2</li>
<ul>
```

```
// Pseudo-code of a DOM node
represented as Javascript
Let domNode = {
  tag: 'ul'
  attributes: { id: 'myId' }
  children: [
    // where the LI's would go
  ]
};
```

```
domNode.children.push('<ul>Item 3</ul>');
sync(originalDomNode, domNode);
```

Render Function

- 用 Render Function 来生成 DOM

```
new Vue({
  el: '#app',
  data: {
   message: 'hello world'
  },
  render() {
    var node = this.$createElement;
    return node(
      'div',
      { attrs: { id: 'myId' } },
      this.message
    );
  }
});
```

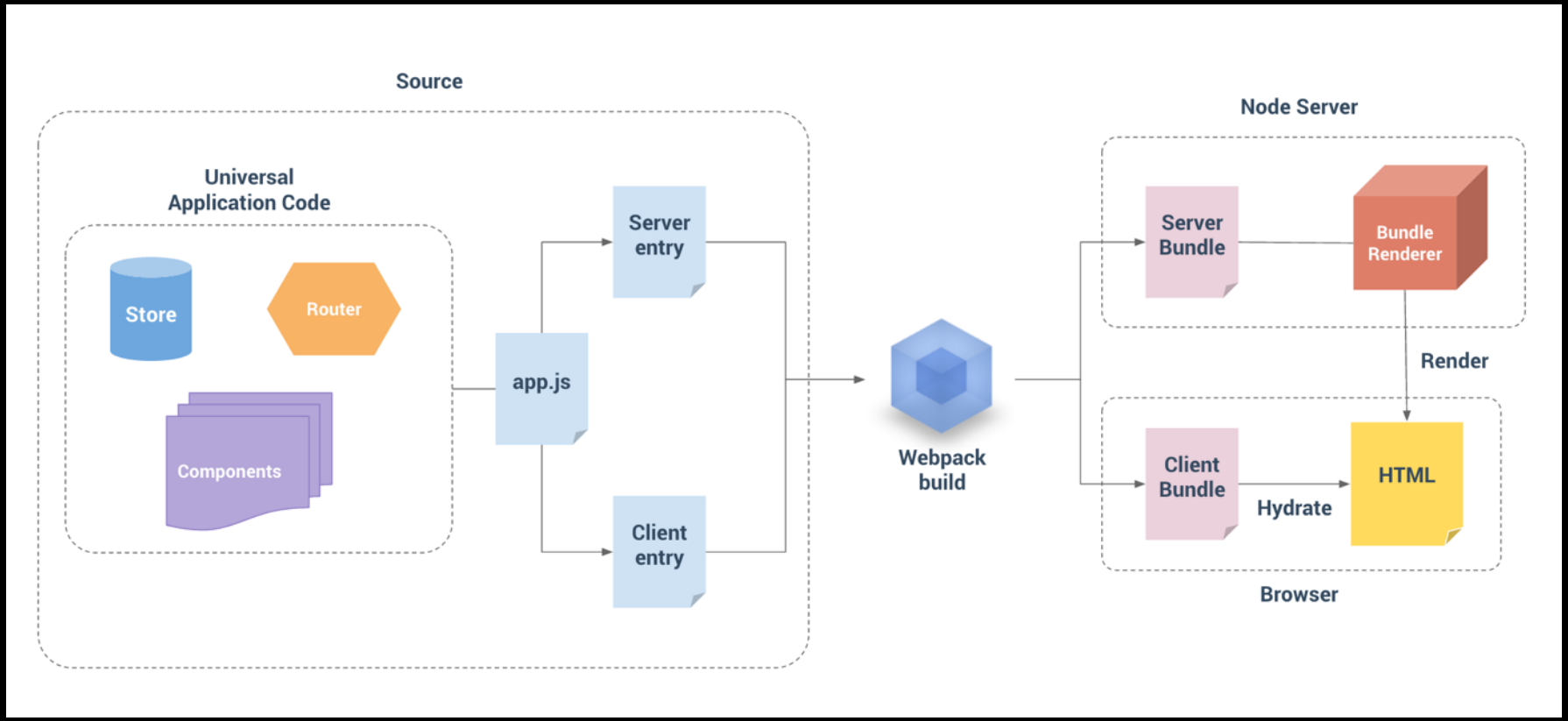
```
<div id='app'>
  <div id='myId'>
    hello world
  </div>
</div>
```

- $createElement
- jsx

```
// @returns {VNode}
createElement(
  // {String | Object | Function}
  // An HTML tag name, component options, or function
  // returning one of these. Required.
  'div',
  // {Object}
  // A data object corresponding to the attributes
  // you would use in a template. Optional.
  {
    // (see details in the next section below)
  },
  // {String | Array}
  // Children VNodes. Optional.
  [
    createElement('h1', 'hello world'),
    createElement(MyComponent, {
      props: {
        someProp: 'foo'
      }
    }),
    'bar'
  ]
)
```

服务器端渲染

- Render Function 可以在后端 Node 环境运行



- 服务器端渲染

```
// Step 1: Create a Vue instance
var Vue = require('vue')
var app = new Vue({
  render: function (h) {
    return h('p', 'hello world')
  }
})

// Step 2: Create a renderer
var renderer = require('vue-server-renderer')
               .createRenderer()

// Step 3: Render the Vue instance to HTML
renderer.renderToString(app, function (error, html) {
  if (error) throw error
  console.log(html)
  // => <p server-rendered="true">hello world</p>
})
```

Stream

- 只要在 Vue 业务包在一个 function call 中并连接上 `window context`
- 服务器 `renderer` 拿到相关业务 js 文件吐出内容
- Vue.js 2.0 支持 Stream 后端流式数据
  - 在 HTML 完整生成之前向前端吐数据
  - `renderer.renderToStream( … )`

Nuxt.js 开发实践

❤️有人造轮子❤️

开源支持

- 脚手架：vue-cli
- 组件化：vue-loader
- 路由：vue-router
- 状态管理：vuex
- Ajax：vue-resource / vue-axios
- 前端开发工具：vue-devtools
- 前端组件库：vux / vue-mdl / mint-ui / element
- 后端渲染?

Nuxt.js

- 一个类似于 Next.js (React) 的开源后端渲染库
- 但同时支持 code-splitting, generation 等
- github.com/nuxt/nuxt.js

```
$ npm install -g nuxt
$ nuxt dev
$ nuxt build
$ nuxt start
$ nuxt generate
```

Nuxt.js 文件结构

```
/assets                 // LESS, SCSS, ES6, Coffee…
/components             // Vue Components
/layouts                // Application Layouts
/pages                  // Vue Pages/Routes
/plugins                // 3rd party / Vue Plugins
/static                 // 静态文件
/store                  // Vuex State Management
nuxt.config.js          // nuxt config file
package.json
```

nuxt.config.js

```
module.exports = {
  head: {
    title: 'starter',
    meta: [
      { charset: 'utf-8' },
      { name: 'viewport',
        content: 'width=device-width, initial-scale=1' },
      { hid: 'description', content: 'Nuxt.js project' }
    ],
    link: [
      { rel: 'icon', type: 'image/x-icon', href: 'favicon.ico' }
    ]
  },
  css: ['~assets/css/main.css'],
  loading: { color: '#3B8070' }
}
```

Pages

```
<template> … </template>
<script>
export default {
  data () { … },
  fetch () { … },
  layout: '…', transition: '…', scrollToTop: '…',
  validate ({params}) { … }, middleware () { … },
  head: {
    title: '…', …
  }
}
</script>
```

Async Data

```
<script>
export default {
  data ({params}) {
    return axios.get(`https://api/${params.id}`)
    .then((res) => {
      return { title: res.data.title }
    })
  }
}
</script>
```

Vuex / Fetch

```
<script>
export default {
  fetch ({params}) {
    return axios.get(`https://api/${params.id}`)
    .then((res) => {
      store.commit('setTitle', res.data.title)
    })
  }
}
</script>
```

Vuex / nuxtServerInit

```
// store/index.js
actions: {
  nuxtServerInit ({ commit }, { req }) {
    if (req.authUser) {
      commit('user', req.authUser)
    }
  }
}
```

- 后端加载数据并交予前端

努力的方向

Any application that can be written in JavaScript, will eventually be written in JavaScript.

— Atwood's Law 2007

总结

- 前端框架虽好，但是还是需要后端渲染
- Vue.js 2.0 后端渲染技术层已算成熟
- Nuxt.js 等库优化了后端渲染的实现效率

- 交互型产品适合前端应用
- 内容型产品适合后端应用

ECUG Con 十周年盛会

谢谢

好技术内容，都在**掘金**

juejin.im

2017-01-15