




业界    移动开发    云计算    软件研发    程序员    极客头条    专题

封面秀    特别策划    管理实践    技术架构

CSDN首页 > 程序员杂志

 订阅程序员杂志RSS

## Vue.js：轻量高效的前端组件化方案

发表于 2015-08-11 16:24 | 8866次阅读 | 来源 程序员杂志 | 18 条评论 | 作者 尤雨溪

前端开发    前端框架    Vue.js

**摘要：**Vue.js通过简洁的API提供高效的数据绑定和灵活的组件系统。在前端纷繁复杂的生态中，Vue.js有幸受到一定程度的关注，目前在GitHub上已经有5000+的star。本文将从各方面对Vue.js做一个深入的介绍。

Vue.js 是我在2014年2月开源的一个前端开发库，通过简洁的 API 提供高效的数据绑定和灵活的组件系统。在前端纷繁复杂的生态中，Vue.js有幸受到一定程度的关注，目前在 GitHub上已经有5000+的star。本文将从各方面对Vue.js做一个深入的介绍。

### 开发初衷

2013年末，我还在Google Creative Lab工作。当时在项目中使用了一段时间的Angular，在感叹数据

绑定带来生产力提升的同时，我也感到Angular的API设计过于繁琐，使得学习曲线颇为陡峭。出于对Angular数据绑定原理的好奇，我开始“造轮子”，自己实现了一个非常粗糙的、基于依赖收集的数据绑定库。这就是Vue.js的前身。同时在实际开发中，我发现用户界面完全可以用嵌套的组件树来描述，而一个组件恰恰可以对应MVVM中的ViewModel。于是我决定将我的数据绑定实验改进成一个真正的开源项目，其核心思想便是“数据驱动的组件系统”。

MVVM 数据绑定

MVVM的本质是通过数据绑定链接View和Model，让数据的变化自动映射为视图的更新。Vue.js在数据绑定的API设计上借鉴了Angular的指令机制：用户可以通过具有特殊前缀的HTML 属性来实现数据绑定，也可以使用常见的花括号模板插值，或是在表单元素上使用双向绑定：

```
<!-- 指令 -->
<span v-text="msg"></span>
<!-- 插值 -->
<span>{{msg}}</span>
<!-- 双向绑定 -->
<input v-model="msg">
```

插值本质上也是指令，只是为了方便模板的书写。在模板的编译过程中，Vue.js会为每一处需要动态更新的DOM节点创建一个指令对象。每当一个指令对象观测的数据变化时，它便会对所绑定的目标节点执行相应的DOM操作。基于指令的数据绑定使得具体的DOM操作都被合理地封装在指令定义中，业务代码只需要涉及模板和对数据状态的操作即可，这使得应用的开发效率和可维护性都大大提升。

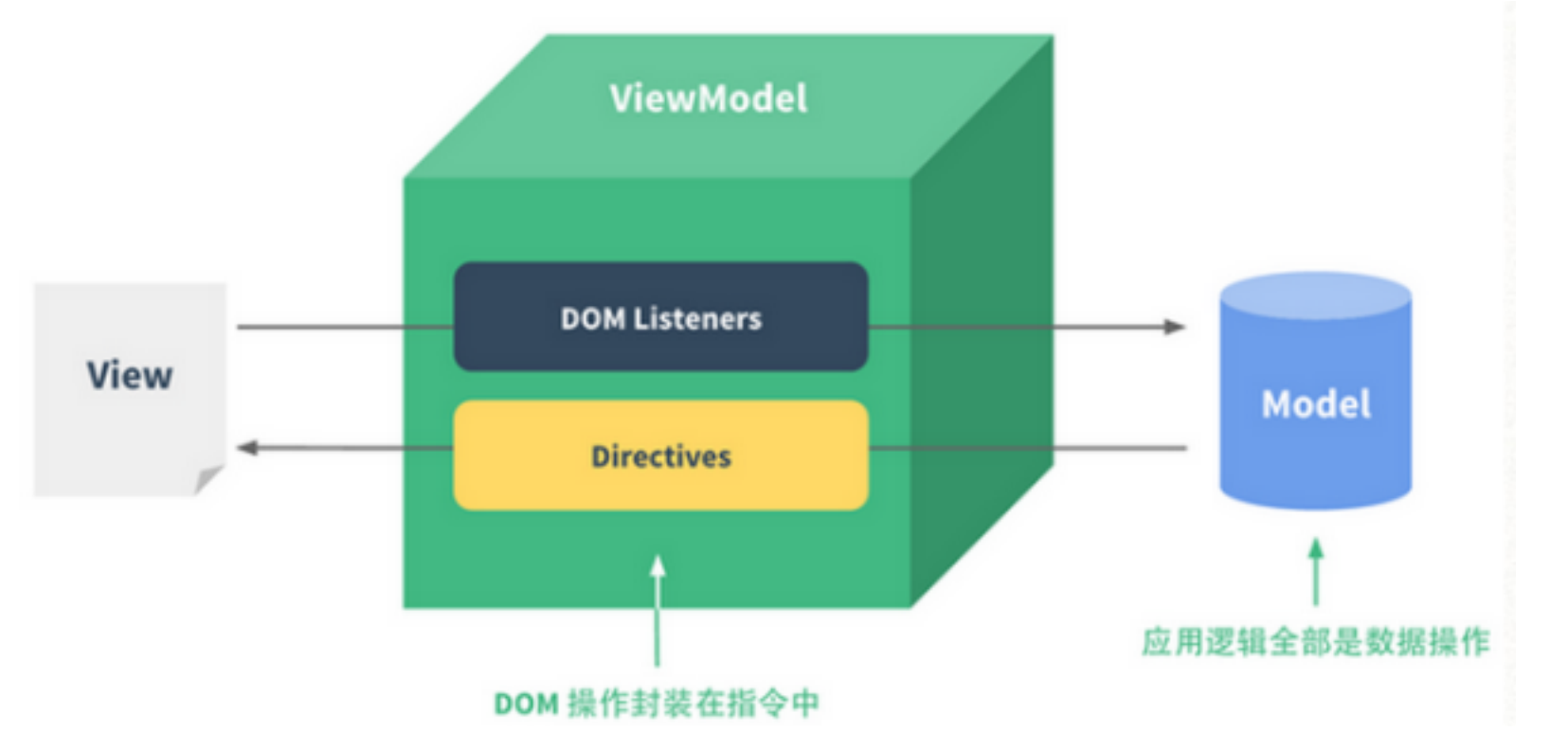


图1 Vue.js的MVVM架构

与Angular不同的是，Vue.js的API里并没有繁杂的module、controller、scope、factory、service等概念，一切都是以“ViewModel 实例”为基本单位：

```
<!-- 模板 -->
<div id="app">
  {{msg}}
</div>
```

```
// 原生对象即数据
var data = {
  msg: 'hello!'
}
// 创建一个 ViewModel 实例
var vm = new Vue({
  // 选择目标元素
  el: '#app',
  // 提供初始数据
  data: data
})
```

渲染结果：

```
<div id="app">
  hello!
</div>
```



</div>

在渲染的同时，Vue.js也已经完成了数据的动态绑定：此时如果改动data.msg的值，DOM将自动更新。是不是非常简单易懂呢？除此之外，Vue.js对自定义指令、过滤器的API也做了大幅的简化，如果你有Angular的开发经验，上手会非常迅速。

### 数据观测的实现

Vue.js的数据观测实现原理和Angular有着本质的不同。了解Angular的读者可能知道，Angular的数据观测采用的是脏检查（dirty checking）机制。每一个指令都会有一个对应的用来观测数据的对象，叫做watcher；一个作用域中会有很多个watcher。每当界面需要更新时，Angular会遍历当前作用域里的所有watcher，对它们一一求值，然后和之前保存的旧值进行比较。如果求值的结果变化了，就触发对应的更新，这个过程叫做digest cycle。脏检查有两个问题：

- 1. 任何数据变动都意味着当前作用域的每一个watcher需要被重新求值，因此当watcher的数量庞大时，应用的性能就不可避免地受到影响，并且很难优化。
- 2. 当数据变动时，框架并不能主动侦测到变化的发生，需要手动触发digest cycle才能触发相应的DOM 更新。Angular通过在DOM事件处理函数中自动触发digest cycle部分规避了这个问题，但还是有很多情况需要用户手动进行触发。

Vue.js采用的则是基于依赖收集的观测机制。从原理上来说，和老牌MVVM框架Knockout是一样的。依赖收集的基本原理是：

- 1. 将原生的数据改造成“可观察对象”。一个可观察对象可以被取值，也可以被赋值。
- 2. 在watcher的求值过程中，每一个被取值的可观察对象都会将当前的watcher注册为自己的一个订阅者，并成为当前watcher的一个依赖。
- 3. 当一个被依赖的可观察对象被赋值时，它会通知所有订阅自己的watcher重新求值，并触发相应的更新。
- 4. 依赖收集的优点在于可以精确、主动地追踪数据的变化，不存在上述提到的脏检查的两个问题。但传统的依赖收集实现，比如Knockout，通常需要包裹原生数据来制造可观察对象，在取值和赋值时需要采用函数调用的形式，在进行数据操作时写法繁琐，不够直观；同时，对复杂嵌套结构的对象支持也不理想。

Vue.js利用了ES5的Object.defineProperty方法，直接将原生数据对象的属性改造为getter和setter，在这两个函数内部实现依赖的收集和触发，而且完美支持嵌套的对象结构。对于数组，则通过包裹数组的可变方法（比如push）来监听数组的变化。这使得操作Vue.js的数据和操作原生对象几乎没有差别[注:在添加/删除属性，或是修改数组特定位置元素时，需要调用特定的函数，如obj.\$add(key, value)才能触发更新。这是受ES5的语言特性所限。]，数据操作的逻辑更为清晰流畅，和第三方数据同步方案的整合也更为方便。

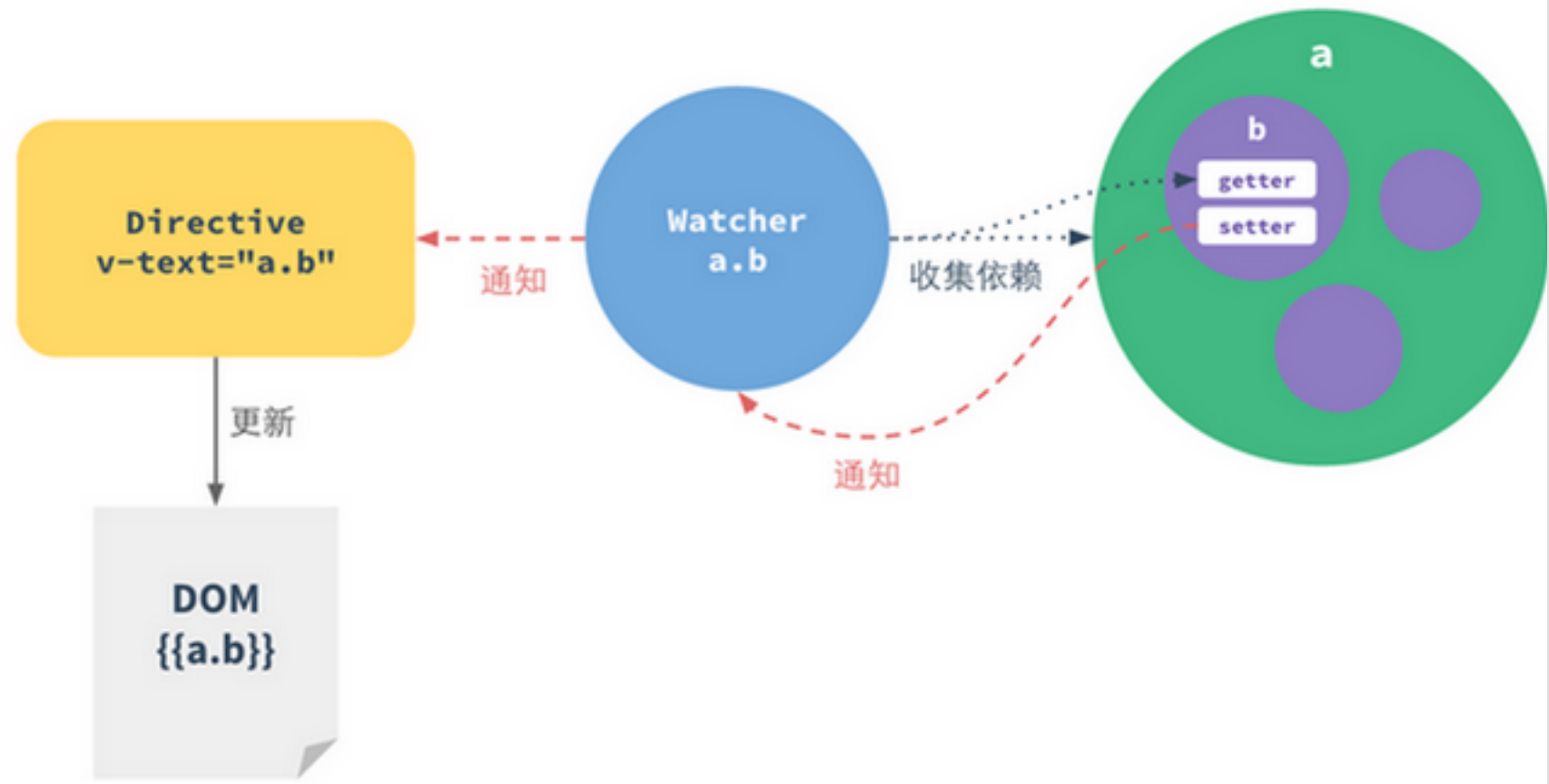


图2 Vue.js的数据观测和数据绑定实现图解

### 组件系统

在大型的应用中，为了分工、复用和可维护性，我们不可避免地需要将应用抽象为多个相对独立的模块。在较为传统的开发模式中，我们只有在考虑复用时才会将某一部分做成组件；但实际上，应用类UI 完全可以看作是全部由组件树构成的：

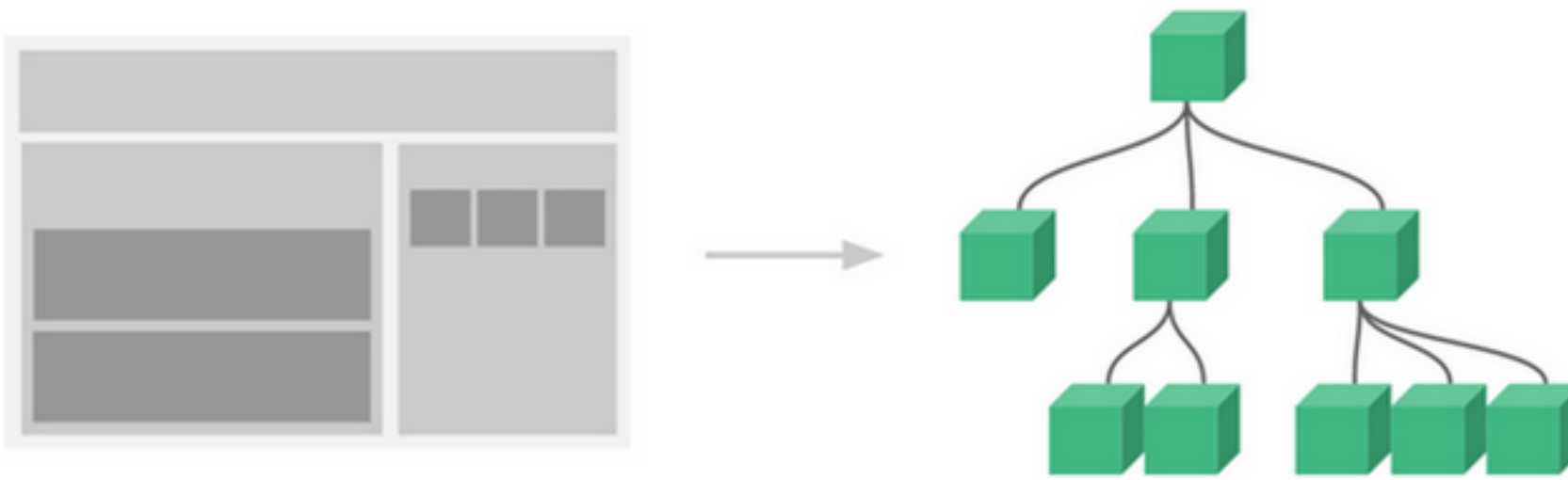


图3 UI = 组件树

因此，在Vue.js的设计中将组件作为一个核心概念。可以说，每一个Vue.js应用都是围绕着组件来开发的。

注册一个Vue.js组件十分简单：

```
Vue.component('my-component', {
  // 模板
  template: '<div>{{msg}} {{privateMsg}}</div>',
  // 接受参数
  props: {
    msg: String<br>

  },
  // 私有数据，需要在函数中返回以避免多个实例共享一个对象
  data: function () {
    return {
      privateMsg: 'component!'
    }
  }
})
```

注册之后即可在父组件模板中以自定义元素的形式调用一个子组件：

```
<my-component msg="hello"></my-component>
```

渲染结果：

```
<div>hello component!</div>
```

Vue.js的组件可以理解为预先定义好了行为的ViewModel类。一个组件可以预定义很多选项， 但最核心的是以下几个：

- 模板（template）： 模板声明了数据和最终展现给用户的DOM之间的映射关系。
- 初始数据（data）： 一个组件的初始数据状态。对于可复用的组件来说，这通常是私有的状态。
- 接受的外部参数(props)： 组件之间通过参数来进行数据的传递和共享。参数默认是单向绑定（由上至下），但也可以显式地声明为双向绑定。
- 方法（methods）： 对数据的改动操作一般都在组件的方法内进行。可以通过v-on指令将用户输入事件和组件方法进行绑定。
- 生命周期钩子函数（lifecycle hooks）： 一个组件会触发多个生命周期钩子函数，比如created，attached，destroyed等等。在这些钩子函数中，我们可以封装一些自定义的逻辑。和传统的MVC相比，可以理解为 Controller的逻辑被分散到了这些钩子函数中。
- 私有资源（assets）： Vue.js当中将用户自定义的指令、过滤器、组件等统称为资源。由于全局注册资源容易导致命名冲突，一个组件可以声明自己的私有资源。私有资源只有该组件和它的子组件可以调用。

除此之外，同一颗组件树之内的组件之间还可以通过内建的事件API来进行通信。Vue.js提供了完善的定义、复用和嵌套组件的API，让开发者可以像搭积木一样用组件拼出整个应用的界面。这个思路的可行性在Facebook开源的React当中也得到了印证。

基于构建工具的单文件组件格式

Vue.js的核心库只提供基本的API，本身在如何组织应用的文件结构上并不做太多约束。但在构建大

型应用时，推荐使用Webpack+vue-loader这个组合以使针对组件的开发更高效。

Webpack是由Tobias Koppers开发的一个开源前端模块构建工具。它的基本功能是将以模块格式书写的多个JavaScript文件打包成一个文件，同时支持CommonJS和AMD格式。但让它与众不同的是，它提供了强大的loader API来定义对不同文件格式的预处理逻辑，从而让我们可以将CSS、模板，甚至是自定义的文件格式当做JavaScript模块来使用。Webpack 基于loader还可以实现大量高级功能，比如自动分块打包并按需加载、对图片资源引用的自动定位、根据图片大小决定是否用base64内联、开发时的模块热替换等等，可以说是目前前端构建领域最有竞争力的解决方案之一。

我在Webpack的loader API基础上开发了vue-loader插件，从而让我们可以用这样的单文件格式 (\*.vue) 来书写Vue组件：

```
<style>
.my-component h2 {
  color: red;
}
</style>

<template>
  <div class="my-component">
    <h2>Hello from {{msg}}</h2>
    <other-component></other-component>
  </div>
</template>

<script>
// 遵循 CommonJS 模块格式
var otherComponent = require('./other-component')

// 导出组件定义
module.exports = {
  data: function () {
    return {
      msg: 'vue-loader'
    }
  },
  components: {
    'other-component': otherComponent
  }
}
</script>
```

同时，还可以在\*.vue文件中使用其他预处理器，只需要安装对应的Webpack loader即可：

```
<style lang="stylus">
.my-component h2
  color red
</style>

<template lang="jade">
div.my-component
  h2 Hello from {{msg}}
</template>

<script lang="babel">
// 利用 Babel 编译 ES2015
export default {
  data () {
    return {
      msg: 'Hello from Babel!'
    }
  }
}
</script>
```



这样的组件格式，把一个组件的模板、样式、逻辑三要素整合在同一个文件中，即方便开发，也方便复用和维护。另外，Vue.js本身支持对组件的异步加载，配合Webpack的分块打包功能，可以极其轻松地实现组件的异步按需加载。

### 其他特性

Vue.js还有几个值得一提的特性：

1. 异步批量DOM更新：当大量数据变动时，所有受到影响的watcher会被推送到一个队列中，并且每个watcher只会推进队列一次。这个队列会在进程的下一个“tick”异步执行。这个机制可以避免同一个数据多次变动产生的多余DOM操作，也可以保证所有的DOM写操作在一起执行，避免DOM读写切换可能导致的layout。
2. 动画系统：Vue.js提供了简单却强大的动画系统，当一个元素的可见性变化时，用户不仅可以很简单地定义对应的CSS Transition或Animation效果，还可以利用丰富的JavaScript钩子函数进行更底层的动画处理。
3. 可扩展性：除了自定义指令、过滤器和组件，Vue.js还提供了灵活的mixin机制，让用户可以在多个组件中复用共同的特性。

### 与Web Components的异同

对Web Components有了解的读者看到这里可能会产生疑问：Vue.js的组件和Web Components的区别在哪里呢？这里简要地做一下分析。

Web Components是一套底层规范，本身并不带有数据绑定、动画系统等上层功能，因此更合适的比较对象可能是Polymer。Polymer在API和功能上和Vue.js比较相似，但它对Web Components的硬性依赖使得它在浏览器支持方面有一定的问题——在不支持Web Components规范的浏览器中，需要加载庞大的polyfill，不仅在性能上会有影响，并且有些功能，比如ShadowDOM，polyfill并没有办法完美支持。同时，Web Components规范本身尚未定稿，一些具体设计上仍存在不小的分歧。相比之下，Vue.js在支持的浏览器中（IE9+）没有任何依赖。

除此之外，在支持Web Components的环境中，我们也可以很简单地利用Web Components底层API将一个Vue.js组件封装在一个真正的自定义元素中，从而实现Vue.js组件和其他框架的无缝整合。

### 总结

在发布之初，Vue.js原本是着眼于轻量的嵌入式使用场景。在今天，Vue.js也依然适用于这样的场景。由于其轻量（22kb min+gzip）、高性能的特点，对于移动场景也有很好的契合度。更重要的是，设计完备的组件系统和配套的构建工具、插件，使得Vue.js在保留了其简洁API的同时，也已经完全有能力担当起复杂的大型应用的开发。

从诞生起到现在的一年半历程中，Vue.js经历了一次彻底的重构，多次API的设计改进，目前已经趋于稳定，测试覆盖率长期保持在100%，GitHub Bug数量长期保持在个位数，并在世界各地都已经有公司/项目将Vue.js应用到生产环境中。在2015年早些时候，Vue.js将发布1.0版本，敬请期待。

#### 【参考链接】

Vue.js官方网站：<http://vuejs.org>

Vue.js GitHub仓库：<https://github.com/yyx990803/vue>

Webpack官方网站：<http://webpack.github.io>

vue-loader单页组件示例：<https://github.com/vuejs/vue-loader-example>

本文选自程序员电子版2015年8月A刊，该期更多文章请查看[这里](#)。2000年创刊至今所有文章目录请查看[程序员封面秀](#)。欢迎[订阅程序员电子版](#)（含iPad版、Android版、PDF版）。

欢迎加入CSDN前端交流群2：465281214，进行前端技术交流。

你也可以扫描下面二维码，加入CSDN前端大讲堂微信群，享受高含金量在线公开课的同时，还可与专家讲师在线切磋交流。（如果群满受限，你可加“Rachel\_qq”为好友，并注明“参加CSDN前端大讲堂”，申请入群。）



近期CSDN前端大讲堂课程预告：

分享主题简介：React 带来的革命性创新是前端世界过去几年最激动人心的变化。自从接触 React 以来，我们深信 React 会彻底改变客户端开发者（包括前端、iOS 和 Android）的开发体验。本公开课中，将从四个大的方向：目标平台（targets）、数据处理（data）、工具（tools）和新的挑战，分享 React 生态系统和社区的进展和未来趋势。同时也将在线解答前端开发者在使用React过程中所面临的难点。

主讲人介绍：郭达峰，Strikingly.com CTO、联合创始人，具有多年前端开发经验。2010年开发了三款Facebook平台应用，获取了超过千万的用户。2012年创立了建站平台Strikingly，成为第一家进入YC孵化器的华人团队。Strikingly.com已使用React重写了大部分应用，是国内使用React比较多的公司之一。

课前温习：[不仅用于UI构建：Facebook React完全解析](#)

本文为CSDN原创文章，未经允许不得转载，如需转载请联系market#csdn.net(#换成@)



顶  
5

踩  
5



推荐阅读相关主题： 前端 api设计 开源项目 开发经验 解决方案 开发模式

相关文章 最新报道

Vue.js：轻量高效的前端组件化方案

Twitter在用哪些Javascript框架？

程序员2015年8月A：前端框架

让开发者高效编程的10个新框架

Amaze UI 2.3.0版本发布 整合多个第三方插件

Foundation 5发布：号称最快版本，响应式用户体验更完善

已有18条评论

还可以再输入500个字



有什么感想，你也来说说吧！

idea丁一鸣 欢迎您！

发表评论

最新评论 最热评论

	组件化方案 帽子是不是有点大?	▲▼ 回复
	五号 2015-08-25 23:15 学习	▲▼ 回复
	NightThink 2015-08-25 10:24 小巧，精致，易上手，才好用	▲▼ 回复
	serberina99 2015-08-24 16:12 感觉类似angularJs	▲▼ 回复
	jenmario1983 2015-08-14 20:54 未来几年前端开发将成为web应用的突破口。	▲▼ 回复
	Rodger-lai 2015-08-14 19:10 学习	▲▼ 回复
	lijianhua1205 2015-08-14 16:17 要坚持不断的造新轮子	▲▼ 回复
	张兴隆 2015-08-14 15:13 呵呵呵	▲▼ 回复
	rufi6853 2015-08-14 12:43 代码看着比React的舒服，数据双向绑定好呢还是Flux模式好？	▲▼ 回复
	jackie_gp 2015-08-13 16:05 好用吗？	▲▼ 回复
	超度逗比 2015-08-13 14:33 难怪一目了然，甩以往一些介绍Vue.js的文章几条街，原来是Vue作者本人介绍的。Vue的确是个小强！	▲▼ 回复
	prowayne 2015-08-13 09:37 早就开始用了，简单易懂	▲▼ 回复
	路大大 2015-08-13 09:04 说白了 就是放弃了一大波IE版本	▲▼ 回复
	HongheWu 2015-09-22 23:33 angularjs1.3+也是不对IE8及以下支持，取舍也是一种方法	▲▼ 回复
	过了即是客 2015-08-13 00:50 等出了1。0的时候就开始学习使用	▲▼ 回复
	zzcoffeebean 2015-08-12 21:41	



mvvm大家也可以看看

avalonjs

▲▼

回复

目睹了整个事件的阳哥

2015-08-12 19:33

早就听说了..正在使用..

▲▼

回复

idea丁一鸣

2015-08-12 17:45

我看这是个不错的简单地框架

▲▼

回复

共1页

首页

上一页

1

下一页

末页

请您注意

- 自觉遵守：爱国、守法、自律、真实、文明的原则
- 尊重网上道德，遵守《全国人大常委会关于维护互联网安全的决定》及中华人民共和国其他各项有关法律法规
- 严禁发表危害国家安全，破坏民族团结、国家宗教政策和社会稳定，含侮辱、诽谤、教唆、淫秽等内容的作品
- 承担一切因您的行为而直接或间接导致的民事或刑事法律责任
- 您在CSDN新闻评论发表的作品，CSDN有权在网站内保留、转载、引用或者删除
- 参与本评论即表明您已经阅读并接受上述条款

近期活动

更多

第七届中国云计算大会

为了更好地推进云计算大数据的技术创新，展示国内外云计算大数据的产业成果，交流云计算大数据的应用经验，第七届中国云计算大会将于2015年6月3-5日北京国家会议中心举办。此次会议将承续前六届大会的成功经验，邀请国内外知名专家出席会议并作演讲。以更加前瞻性视野，分享国内外云计算大数据的技术趋势和实践经验，推动云计算大数据的发展和进步。

微博关注

程序员编辑部

北京 东城区

+ 加关注

程序员移动端订阅下载

相关热门文章

- 《程序员11月A刊：云计算开源核心技术变迁》...
- Kubernetes容器管理技术变迁
- 微信支付开发中的“坑”与解决之道
- 《程序员11月B刊：微信·生态》火热上市！
- 微信生态中企业应用的创新与创业机会
- LBS应用的路径引导方法

热门标签

- 技术架构
- 程序员
- 编程语言
- 前端开发
- 产品设计
- 大数据
- 智能算法
- 移动
- 云计算
- 图书
- 开源
- 管理实践

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

网站客服 杂志客服 微博客服 webmaster@csdn.net 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 | 江苏乐知网络技术有限公司 提供商务支持  
京 ICP 证 070598 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

网站客服 杂志客服 微博客服 webmaster@csdn.net 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 | 江苏乐知网络技术有限公司 提供商务支持  
京 ICP 证 070598 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

