链滴

React Vue Web 前端主流技术栈比较

作者：yanjoy

原文链接：https://ld246.com/article/1641802340800

来源网站：链滴

在底层的前端框架领域中，最早是jquery称霸互联网，近几年MVVM类型的框架慢慢成为主流，Vue React和Angular三大框架并驾齐驱。可以说，目前这三种是开发者用的最多使用最广的底层框架，由此衍生了大量基于这些框架的l库。

这些年不断有新的框架冒出来，又不断有旧的框架被淘汰，在这里重点总结一下Vue、React各自的缺点以及区别。

前端框架解决了什么问题?

框架其实就解决了使用声明语法，描述组件对象的嵌套关系，并自动生成与**DOM** 对象的对应关系的题：

- DOM对象以及他们的从属（同时是传递关系）关系，是通过html自动生成的，然而开发者把“组”抽象为js对象，由框架实现子组件的自动创建，自动销毁，自动数据传递，自动render，自动事件听等功能；
- 解决把js组件对象存在它应该在的地方，并且rerender的时候能把js组件对象和DOM节点对应起来过程；
- 组件复用的问题；
- 组件重渲染之后，commit到DOM对象上。

在解决Web应用开发这些问题时，现代前端框架采用了几种不同的方案：

- **Vue**： 是基于数据的watch 的，组件级别通过Object.defineProperty监听对象属性的变化，重写组的api 监听数组元素的变化，之后进行dom 的更新。
- **React**： 不检查数据变化，React不会直接渲染数据到 dom，而是中间加了一层虚拟 dom，每次渲染成这个虚拟 dom，然后 diff 下渲染出的虚拟 dom 是否变了，变了的话就去更新对应的 dom 所以React是需要通过this.setState()去手动更新数据，它只关心当前组建状态。
- **Angualr**： 基于脏检查，在每个可能改变数据的逻辑之后都对比下数据是否变了，变了的话就去更dom。

运行效率问题

Vue

vue 是组件级别的数据 watch，当组件内部监听数据变化的地方特别多的时候，一次更新可能计算量别大，计算量大了就可能会导致丢帧，也就是渲染的卡顿。所以当Vue组件特别庞大或数据特别多时渲染性能就会明显下降。

解决方案：优化方式就是把大组件拆成小组件，这样每个数据就不会有太多的watcher。

React

React并不监听数据的变化，而是渲染出整个虚拟dom，然后diff。但是当应用的组件树特别大的时，只是 shouldComponentUpdate 跳过部分组件渲染，依然可能计算量特别大。计算量大了同样可导致渲染的卡顿。

解决方案：通过链表记录下组件路径，通过链表的循环遍历按照时间片分段，让vdom 的生成不再阻页面渲染，这就像操作系统对多个进程的分时调度一样。这个通过把组件树改成链表，把vdom 的生从递归改循环的功能就是react fiber。

逻辑、组件复用问题

Vue

vue 的组件是操作对象的方式，那么逻辑复用方式很自然可以想到通过对象属性的 mixin，vue2 的件内逻辑复用方案就是 mixin，但是 mixin 很难区分混入的属性、方法的来源，比较乱，代码维护性。

解决方案：通过 mixin 的方式来复用逻辑，也有组件太大的问题，在 vue3 中也采用了hooks（Vue3. function based API）方法。

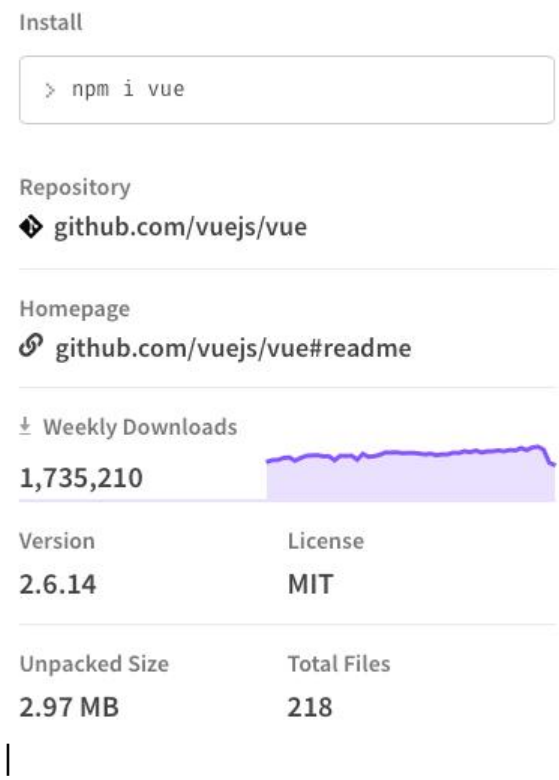
React

React 的组件是 class 和 function 两种形式，那么类似高阶函数的高阶组件（high order componen）的方式就比较自然，也就是组件套组件，在父组件里面执行一部分逻辑，然后渲染子组件。 除了多一层组件的 HOC 方式以外，没有逻辑的部分可以直接把那部分 jsx 作为 props 传入另一个组件来复，也就是render props。但是 HOC 的逻辑复用方式最终导致了组件嵌套太深，而且 class内部生命期比较多，逻辑都放在一起导致了组件比较大问题。

解决方案：HOC 和 render props 是 react 的 class 组件支持的两种逻辑复用方案。通过 function 件的 ooks api 解决了 class 组件的逻辑复用方案Class组件太大的问题。

生态活跃度的区别

| --- | --- |

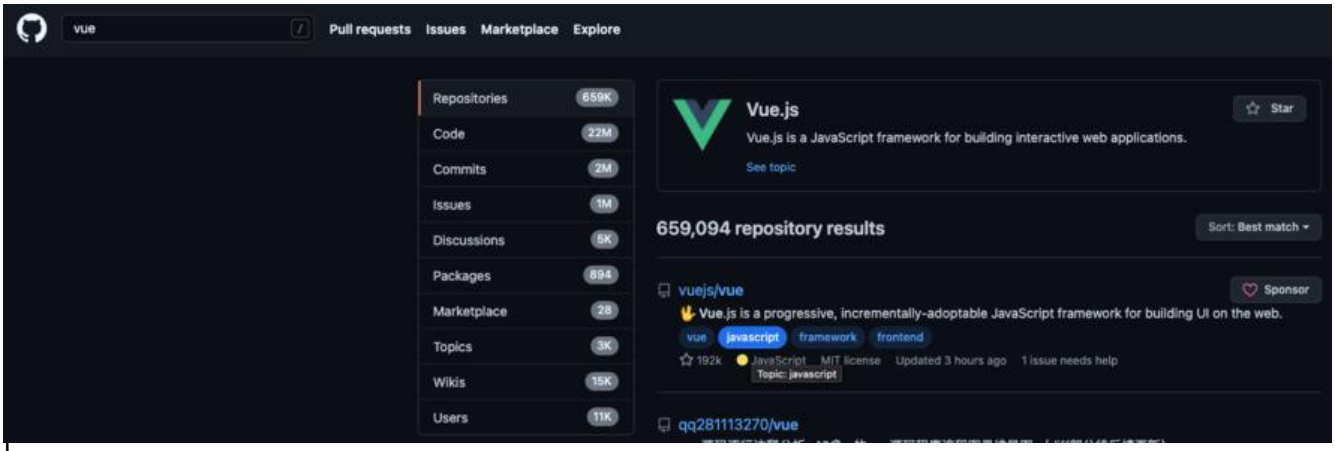
| 
 |

原文链接：React Vue Web 前端主流技术栈比较

|

从npmjs.com上分别看了下，React和Vue之间存在着巨大的安装使用差距，这就意味着世界上大多团队/开发者在选取前端技术栈时，首先选用的是React。

| --- |

| 

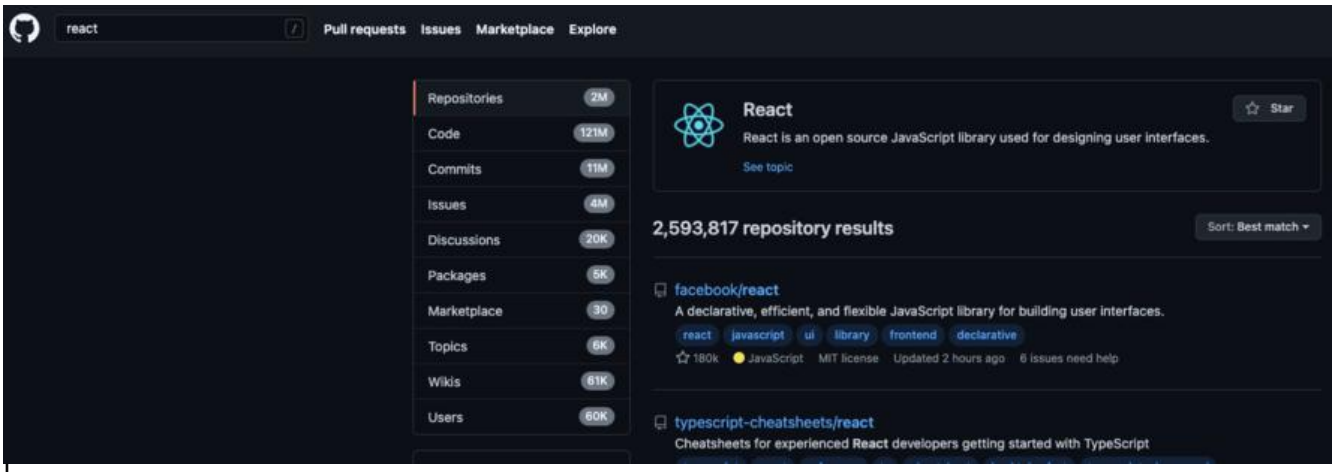
|

| Vue 相关库与项目 |

| --- |

|

原文链接：React Vue Web 前端主流技术栈比较



| React 相关库与项目 |

这就意味着世界上大多数团队/开发者在选取前端技术栈时，首先选用的是React。

在知名程序员社区Github中搜索两个库可以发现：

● Vue生态支持相对React偏弱，最常见的问题就是 React 那边有甚至受欢迎的插件Vue没有。React很多十分受欢迎的开源库，而Vue相对匮乏。

● Vue在TypeScript语言支持弱，现在虽然3.x版本进行了改善，但是由于开发者对于框架的思维惯性因还是没有多少人认可。

关键特点对比

非技术性特点	Vue	React
对于初学者的学习曲线*	入门2-3天可以尝试开发简单功能；6-8周自熟练完成项目需求；对项目框架融会贯通，能够独立解决复杂问题则需要4-6个月甚至更长时间	门1-2周可以尝试开发简单功能；6-10周左右独自熟练完成项目需求；对项目框架融会贯通，能够独解决复杂问题则需要4-6个月甚至更长时间
库大小	轻巧	较大
支持团队	个人/社区爱好者	Face ook公司官方支持
设计哲学	易用、灵活、高效	声式、组件化、一次学习随处编写
适合什么项目*	轻巧，易学且易于编写。由于其熟悉的模板语法和件的使用，将现有项目集成或迁移到Vue变得更快，更流畅。因此，Vue非常适合初创企业，但也可在大型应用程序中使用。	公司的支持庞大的社区和更大的生态系统React非常适合构建复杂的企业级应用程序。因为它的成熟性和社区性，或多或少地保证了寻求帮助获得及时详细的答案

*对于初学者的学习曲线： “初学者”是指有基本的计算机基础知识，JavaScript、html、css、浏览运行原理等web开发基础知识扎实。因为个体差异，这里仅为一般情况。对于初学者，React学习曲相对陡峭，但是一旦入门之后的学习路线两者没有本质差别。另外阿里目前 Antd Pro V5版本的Reac脚手架已经大大降低了学习成本。

*适合什么项目：Vue适合小型项目或初创公司这种说法往往来源于目前许多开发者刻板映像，另外Vu创作者尤雨溪自己也说过： “Vue 从一开始的定位就是尽可能的降低前端开发的门槛，让更多的人能更快地上手开发。 ..... 很多时候我更希望自己做的东西能帮到那些中小型企业和个人开发者。举个

子来说，美国传统行业里有很多 small business，它们不像大公司那样有专门的 IT 团队来信息化整流程，很多只能雇一个普通的 contractor 程序员，有些甚至是老板自己兼职研究代码。我收到过好封这样的感谢信，说因为 Vue 让它们多快好省地做了个内部应用，解决了实际问题，这样的故事是我觉得特别爽的。”，在种种原因影响下，Vue确实不会作为很多公司大型项目优先选择技术栈，从也导致了Vue社区并不如React繁荣。但是这些并不意味着Vue不能拿来构建大型Web应用。

技术特性对比

技术特性	**Vue**	**React**
超大型web应用首屏渲染时间*	较慢	快
提供了相应式和组件化的视图组件	是	是
性能优化	更多依赖开发者自身水平	eact Fiber
服务器渲染	支持不好	支持
数据绑定	双向绑定	单向数据流
TypeScript支持	2.x不支持，3.x版本改善	持良好
组件逻辑复用性	mixin混入，3.x版本后改为hooks	ooks、HOC

综述

技术层面从加载速度、运行时性能来说，两者目前综合各种场景应该说是没有什么质的差别，其实不在某项技术更优，只有更适合问题。我们能够根据项目实际情况选择合适的技术栈更为重要。两种技框架其实均能非常好的解决我们日常研发所遇见问题。

React

React的应用广泛程度、社区活跃度更好，并且React是由Facebook公司独立团队进行更新维护。而国内很多公司都将React作为web端应用的主要技术栈，知名的比如：阿里、天猫、华为、豆瓣、美、滴滴、知乎等。这些大厂也开源了一系列基于React技术栈的框架，其中最为著名的则为Antd团队发的umi、antd

design。

当我们需要研发大型复杂web应用程序时，React强大的社区、丰富的第三方库能够为我们项目提供好的支持。缺点是React的学习曲线可能比较陡峭，编程思维模式较抽象，不利于新人快速掌握。

Vue

Vue从一开始的定位就是尽可能的降低前端开发的门槛，让更多的人能够更快地上手开发，在易用性功能，性能，文档易读上面表现更好。vue3更是把模版的优势发挥到了极致，在大量数据渲染上可以 React 更快。缺点则是社区规模太小，框架本身也仅仅是由个人开发者在进行维护。在大型复杂web用开发中，Vue项目优化比较考验开发者本身技术水平。

最后，使用Vue作者尤雨溪自己的总结：

使用场景上来说，React 配合严格的Flux 架构，适合超大规模多人协作的复杂项目。理论上 Vue 配类似架构也可以胜任这样的用例，但缺少类似 Flux 这样的官方架构。小快灵的项目上，Vue 和 React 的选择更多是开发风格的偏好。对于需要对 DOM 进行很多自定义操作的项目，Vue 的灵活性优于Re

ct。