01 开篇词：为什么要研究 React 内部运行机制与设计原理

更新时间：2020-08-12 14:13:39

> 机会不会上门来找人，只有人去找机会。——狄更斯

开篇词—为什么要研究 React 内部运行机制与设计原理

你好，我是上古鹏，目前在一家一线互联网公司做资深前端工程师。在多年的前端开发经历中，我先后使用过 jQuery，Backbone.js，Vue 与 React 等前端框架。记得当年使用 Backbone.js 框架开发前端项目时，感觉在前端使用 MVC 的开发模式大大提高了开发效率，并降低了程序的维护成本。直到后来接触 React，React 带来的数据驱动更新的思想，让我感觉到 React 的出现在前端技术发展历史中具有里程碑式的意义，它不但提高了我们的开发效率，提升了应用程序的执行性能，更是改变了我们的编程模式。

现阶段，React 作为前端开发工程师面试必问、工作必用的内容，它正变得越来越重要。尤其对于大型公司，求职者是否对 React 的运行原理有一定的了解甚至能直接决定他是否被录用。本专栏主要讲 React 的内部运行机制与设计原理，期望能给读者带来 React 更多深层次的内容。

下面，我们来简单说一说为什么要研究 React 内部运行机制与设计原理。

能力提升的核心体现之一

多年前，我在网易实习时的一位职业导师也和我谈过如何在技术这条路走的最快最稳。他说，对于一般人，毕业后 1-3 年是夯实基础的阶段，3-5 年是能力进一步提升的阶段。那么我们该如何看待这两个阶段呢？

夯实基础：能够熟练掌握前端领域的常用知识点，如 JavaScript 继承原理，异步编程方式及原理，熟练使用 React、Vue 等前端框架等。

能力提升：深入了解常用框架原理，能够根据自身的知识储备解决某一类问题，如开发效率问题，代码质量问题，多端兼容问题等。

React 技术体系在前端应用中占据了大半壁江山，一名优秀的前端工程师对 React 的掌握不应该仅停留在应用层面，而是应该熟知其内部工作机制与设计原理。

入职互联网一线大厂的必备技能

由于 React 被广泛应用于大、中、小型互联网公司，国内一线互联网公司对前端工程师岗位招聘中对 React 框架原理提出了明确要求。

某跳动公司对前端开发工程师招聘职位的要求中有：

> 对主流前端框架（React/Vue/Angular等）有一定了解，并至少对其中一种有深入了解；

某巴巴公司对前端开发工程师招聘职位的要求中有：

> 熟悉（React/Vue/Angular）中一种框架，且有实际项目应用经验，具备独立项目开发能力；

某鹅厂对前端开发工程师招聘职位的要求中有：

> 对（React/Vue/Angular）框架非常熟悉，并有项目经验；

如果你还不知道`setState`的工作原理，甚至没有听说过 React Fiber 架构，那么你距离互联网一线大厂的要求还有很大一段距离。

相对于 **Vue**，大厂更喜欢 **React**

React 和 Vue 是当下前端开发领域中最受欢迎的两个优秀框架。事实上，**在大厂的业务代码中 React 却占据了绝对的主导地位**。这是为什么呢？

React 是由 Facebook 公司来更新和维护，它是大量世界级优秀程序员的思想结晶，它拥有着最多的开发者和技术社区。**React 的流行不仅仅局限于普通开发工程师对它的认可，更多的是其他框架借鉴它的思想**。Vue 框架设计之初，有很多的灵感来自 Angular 和 React，包括 Vue 3 的很多新特性（如 Function Based API）也是借鉴和学习了 React。**React 可以说是前端开发领域的先驱者，它总是会引领整个个前端的潮流**。

高级前端工程师必须要学习 React 以及它的内部运行机制与设计原理，因为你可以了解到 React 如何以最优雅的方式在组件上面实现「数据驱动更新」思想，你也可以了解到 React 如何让你的程序不直接操作 DOM 的情况下还能更新 DOM … 同时，你也会了解到 React 使用元素来描述 DOM 具有非常好的创新性意义，因为它实现了以最小对象来描述页面结构，节省了大量的数据成本，它的这种做法是提升页面渲染性能的关键因素之一。

事实上，React 相对于 Vue 的学习门槛要高，因为 React 不仅需要开发者掌握 HTML 和 CSS，还需要拥有良好的 JavaScript 基础（尤其是 ES6）。这个门槛也是衡量开发者水平的标准之一，如果你一直使用 Vue 而没有接触 React，那么你的技术能力可能很难得到大多数人的认可。

React 在多端统一方面的巨大潜能

2019 年京东前端团队发布了一套遵循 React 语法规范的 **多端开发** 解决方案—Taro。

Taro 遵循 React 语法规范，它采用与 React 一致的组件化思想，组件生命周期与 React 保持一致，同时支持使用 JSX 语法，让代码具有更丰富的表现力，使用 **Taro** 进行开发可以获得和 React 一致的开发体验。**Taro 的成功实现离不开作者们对 React 内部工作原理的深刻认知。**

开发 React 应用更加得心应手

如果你是一位 React 的忠实开发者，如果你有过因为 React 应用程序报错而不能轻松定位到问题原因的经历，如果还不知道 React 组件和元素的区别，如果你对 React 框架没有自己的深刻认知 … 那么，你应该学习本课程。通过本课程的学习，了解 React 的内部工作机制与设计原理后再写 React 程序时会像打通了任督二脉一样，异常清爽，得心应手。

课程是如何设计的？

上面说的这些，其实也正是咱们这个课程的核心设计理念。接下来，我就说说这门课具体是怎么设计的。

为了能让你更快的掌握 React 内部运行机制与设计原理，同时也照顾一下那些没有基础的同学，我将专栏内容作了如下安排。

本专栏围绕 **React 内部运行机制与设计原理** 核心技术展开，共划分为 6 章 30 个小节。

第一章，为本专栏的学习入口， 主要介绍怎么才能顺利开始研究 React 内部运行机制与设计原理，详细介绍了应用程序首次渲染的入口—`ReactDOM.render` 函数。

第二章，为本专栏的学习基础， 主要介绍 React 世界中的一些重要的基础概念。包括组件的定义与设计思想、深入理解组件的生命周期函数、组件实例的创建与作用、元素的设计原理以及更新与更新队列的定义等。

第三章，主要介绍 React Fiber 架构， 由浅入深的帮助大家对 Fiber 有个清晰的认识。

第四章，主要介绍 React 任务体系， 了解 React 对任务的定义及调度逻辑。

第五章，以应用程序的运行为主线，详细介绍其在首次渲染过程中的整体执行流程。 React 应用程序首次渲染流程包括了构建 fiberRoot 对象，构建 workInProgress 树，收集 Effect List 以及将内容更新到屏幕等过程。

第六章，继续以应用程序的运行为主线，详细介绍其在更新渲染过程中的整体执行流程。 React 应用程序更新渲染流程中的重要工作有为需要更新的 Fiber 结点标记 effectTag，收集 Effect List，Fiber 结点的 diff 处理等。

在最后，我再说两句，React 是一个非常优秀的前端框架，无论你现在的工作中正在使用其他框架还是使用 React 框架，对 React 进行深入学习研究，对我们今后的工作都会有指导、启发意义。通过对本专栏的学习，你将收获到**源码学习方法，React 内部运行原理，React 核心设计思想，面试高级技能，JavaScript 基础知识点回顾 …**

通往知识财富的路上充满了很多的疑惑与不解，所以我们需要更多的耐心与毅力，拨开层层迷雾后才能到达知识的殿堂。学习《 前端高手必学课-React》这门课的过程中，难免会感觉到枯燥乏味，请不要担心，**每一篇文章都是由浅入深讲述相关内容，每一段代码都有详细的注释与说明**。如果你遇到困惑或者不理解的地方请不要担心，每一节容易引起不理解的内容都会有说明在后面文章中进行详细介绍。你也可以在留言区进行留言，我也会在最快的速度回复你。

《前端高手必学课-React》专栏是我多年来研究 React 源码及设计思想的一个总结课程。专栏中的文章结合了各种技术社区对 React 相关知识点的介绍以及个人理解，并非官方文档。由于个人水平有限，专栏中难免有错误与不当之处，也希望你在阅读的过程中发表自己的看法，让我们一起沟通，一起进步。

书山有路勤为径，学海无涯苦作舟。今天学的东西，看起来是那样的平平淡淡，甚至枯燥无味，但也许有一天你会发现，读书学习原来是那样的弥足珍贵。学习是一个大浪淘沙的过程，是一个化平淡为神奇的过程，更是一个不断积累财富的过程。**只有不断的积累，才能取得最终的成功！**

}

02 研究 React 应用程序的内部运行机制要找到切入点