React 进阶实践指南 - 我不是外星人 - 掘金课程

juejin.cn/book/6945998773818490884/section/6948353204413268001

为什么学习React?

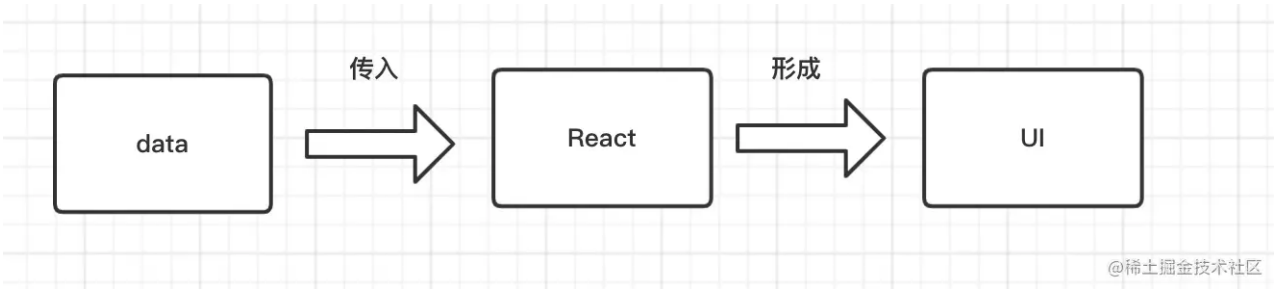
React 是当前非常流行的用于构建用户界面的 JavaScript 库，也是目前最受欢迎的 Web 界面开发工具之一。

这主要是得益于它精妙的设计思想，以及多年的更新迭代沉淀而来的经验。

首先，**React** 的出现让创建交互式 **UI** 变得轻而易举。 它不仅可以为应用的每一个状态设计出简洁的视图。而且，当数据变动时，React 还能高效更新并渲染合适的组件。

这是因为，在 React 的世界中，函数和类就是 UI 的载体。我们甚至可以理解为，将数据传入 React 的类和函数中，返回的就是 UI 界面。

同时，这种灵活性使得开发者在开发 React 应用的时候，更注重逻辑的处理，所以在 React 中，可以运用多种设计模式，更有效地培养编程能力。



其次，**React** 把组件化的思想发挥得淋漓尽致。 在 React 应用中，一切皆组件，每个组件像机器零件一样，开发者把每一个组件组合在一起，将 React 应用运转起来。

最后，**React** 还具有跨平台能力。 React 支持 Node 进行服务器渲染，还可以用 React Native 进行原生移动应用的开发，随着跨平台构建工具的兴起，比如 Taro，开发者可以写一套 React 代码，适用于多个平台。

因此，学好 React，能增强我们自身的职业竞争力。

跟着小册学习React

想要系统学习 React，我建议你跟着小册学。这里，我准备了几个学习中的常见问题，我就结合它们来说说小册优势。

① “看会”等于“学会”吗？

我认为看会不等于学会的。俗话说“好记性不如烂笔头”，前端开发者学习重心还是要放到 **coding** 上来。

因此，《React进阶实践指南》这本小册，在讲解 React api 高阶用法，和一些核心模块原理的同时，也会列举出很多实践 Demo 去强化知识点。那么，小册的最佳学习方式就是：读者可以结合小册每一章节中的知识点，去亲自体验每一个高阶玩法，亲自尝试实现每一个 Demo。

② 有必要掌握小册中的源码吗？

这本小册有很多原理源码，我们是否有必要花费大量时间去研究它们呢？这也是很多人在学习 React 的时候比较关心的问题。我想，虽然我们没必要纠结源码中的一些细枝末节，但还是有必要掌握一些核心原理的（可以不看源码，但需要掌握原理）。原因有两点：

第一，现在前端圈子内卷严重，面试官在面试中为了对比候选人，就会问一些原理/源码层面上的问题。因此，如果应聘者不懂原理/源码，就会很吃亏。

比如应聘者在简历上写了用过 mobx 和 redux，那么面试官就很可能会问两者区别。如果这个时候应聘者的答案只是停留在两者使用层面上的区别，肯定是很难让人满意。

第二，更深的理解方可更好的使用。开发者对框架原理的深入理解可以让其在工作中，更容易发现问题、定位问题、解决问题。就算是面对一些复杂困难的技术场景，也能提供出合理的解决方案。

③ 一定要按顺序学习吗？跳着看可以吗？

本小册的难度是由浅入深的，内容是承上启下的。所以我希望每一个读者能够按照章节顺序阅读，不要跳跃式阅读。

React里程碑

在正式学习 React 之前，首先看一下 React 发展史中一些重要的里程碑（从 `React16` 开始），《React进阶实践指南》这本小册中，会围绕这些里程碑中的内容展开讨论。

- `v16.0` ： 为了解决之前大型 React 应用一次更新遍历大量虚拟 DOM 带来个卡顿问题，React 重写了核心模块 Reconciler ，启用了 Fiber 架构；为了在让节点渲染到指定容器内，更好的实现弹窗功能，推出 createPortal API；为了捕获渲染中的异常，引入 componentDidCatch 钩子，划分了错误边界。
- `v16.2` ： 推出 Fragment ，解决数组元素问题。
- `v16.3` ： 增加 React.createRef() API，可以通过 React.createRef 取得 Ref 对象。增加 React.forwardRef() API，解决高阶组件 ref 传递问题；推出新版本 context api，迎接Provider / Consumer 时代；增加 getDerivedStateFromProps 和 getSnapshotBeforeUpdate 生命周期 。
- `v16.6` ： 增加 React.memo() API，用于控制子组件渲染；增加 React.lazy() API 实现代码分割；增加 contextType 让类组件更便捷的使用context；增加生命周期 getDerivedStateFromError 代替 componentDidCatch 。

- `v16.8` ：全新 React-Hooks 支持，使函数组件也能做类组件的一切事情。
- `v17` ： 事件绑定由 document 变成 container ，移除事件池等。

阅读前的声明

- 本小册涉及的所有 React 源码版本前期为 `v16.13.1` - `v17` ，后期改为最新的 `v18.0.2` ，为了用最精炼的内容把事情讲明白，本小册涉及的源码均为精简后的，会和真正的源码有出入，敬请谅解。
- 本小册各个章节是承上启下的，所以请按照目录，渐进式阅读。
- 所有的实践 Demo 项目，笔者已经整理到 GitHub上，地址为 《React进阶实践指南》——Demo 项目和代码片段，持续更新中。

留言

全部评论（134）

查看全部 134 条回复