双11返场进行中！优惠Go on，快来捡漏！ 02 天 14 时 32 分 58 秒 Go >>>

Sunday • 更新于 2022-10-19

上一节 15：框架实现：... 01：前言 下一节

16：总结

那么到这里我们就已经完成了 diff 算法的逻辑。

整个的 diff 就分成 5 种场景，前四种场景相对而言，还比较简单。最复杂的就是第五种，乱序的场景。

整个 diff 中，涉及到了很多的算法，可能还有很多我们之前不太了解的一些概念，比如：最长递增子序列。

那么 diff 完成之后，也标记着我们整个的 runtime 就已经全部讲解完成了。

那么在这里我们把整个 runtime 总结一下，对于 runtime 而言：

1. 首先我们讲解了 dom 、节点、节点树和虚拟 DOM ，虚拟节点之间的概念。
2. 然后我们说明了 render 函数和 h 函数各自的作用。我们知道 h 函数可以得到一个 vnode ，而 render 函数可以渲染一个 vnode
3. 接下来我们讲解了挂载、更新、卸载，这三组概念。也知道了针对于不同的 vnode 节点，那么他们的挂载、更新、卸载方式也都是不同的。
4. 下面我们讲解了组件，我们知道组件本质上是一个对象（或函数），组件的挂载本质上是 render 函数的挂载。
5. 组件内部维护了一个 effect 对象，以达到响应性渲染的效果。
6. 而针对于 setup 函数而言，它在实现上反而更加简单，因为我们不需要改变 this 指向了。
7. 结合所学，新旧节点的所有挂载和更新情况，可以被分为九种场景：
    1. 旧节点为纯文本时：
        1. 新节点为纯文本：执行文本替换操作
        2. 新节点为空：删除旧节点
        3. 新节点为数组：清空文本，添加多个新节点
    2. 旧节点为空时：
        1. 新节点为纯文本：添加新节点
        2. 新节点为空：不做任何事情
        3. 新节点为数组时：添加多个新节点
    3. 旧节点是数组时：
        1. 新节点为纯文本：删除所有旧节点，添加新节点
        2. 新节点为空：删除所有旧节点
        3. 新节点为数组时：进行 diff 操作
8. 最后的 diff 分为 5 种场景，最后一种场景还是比较复杂的。

那么从下一章开始，我们就要进入到 **编译器** compiler 模块的学习了，大家准备好了~~~~

我要提出意见反馈

Copyright © 2022 imooc.com All Rights Reserved | 京ICP备 12003892号-11 京公网安备11010802030151号