

全部开发者教程

11: 框架实现：生命回调钩子中访问响应性数据

12: 源码阅读：响应性数据改变，触发组件的响应性变化

13: 框架实现：响应性数据改变，触发组件的响应性变化

14: 源码阅读：composition API，setup 函数挂载逻辑

15: 框架实现：composition API，setup 函数挂载逻辑

16: 总结

第十二章：runtime 运行时 - diff 算法核心实现

01: 前言

02: 前置知识：VNode 虚拟节点 key 属性的作用

03: 源码阅读：场景一：自前向后的 diff 对比

04: 框架实现：场景一：自前



Sunday • 更新于 2022-10-19

◀ 上一节 01: 前言 03: 源码阅读: ... 下一节 ▶

## 02: 前置知识：VNode 虚拟节点 key 属性的作用

再去学习 diff 之前，我们需要先了解一个前置的知识点。

我们知道在 v-for 循环的时候，我们必须指定一个 key 值。那么这个 key 值的作用是什么呢？

我们之前写过一个方法： packages/runtime-core/src/vnode.ts 中的 isSameVNodeType 方法。

该方法是用来判断两个 VNode 是否为相同 VNode 的方法：

<> 代码块

```
1  /**
2   * 根据 key || type 判断是否为相同类型节点
3   */
4   export function isSameVNodeType(n1: VNode, n2: VNode): boolean {
5     return n1.type === n2.type && n1.key === n2.key
6   }
```

在这个代码中，type 我们知道什么意思。但是这个 key 是干什么的呢？

这里的 key 是一个 props，比如我们可以在 vue 源码中，利用 h 函数，构建这样的 VNode：

<> 代码块

```
1   const vnode = h('li', {
2     key: 1,
3   }, '1')
```

VNode 的值为：

<> 代码块

```
1   {
2     "__v_isVNode": true,
3     "type": "li",
4     "props": { "key": 1 },
5     "key": 1,
6     "children": "1",
7     "shapeFlag": 9,
8     ....
9   }
```

此时我们可以发现，VNode 中多出了一个 key 的属性，值为 1。

如果我们通过 isSameVNodeType 方法，判断两个 VNode 是否为 相同的 VNode，则只需要保证两个 VNode 的 type 和 key 相同即可：

<> 代码块

```
1   const vnode = h('li', {
2     key: 1,
3   }, '1')
4   const vnode2 = h('li', {
5     key: 1,
6   }, '2')
7   isSameVNodeType(vnode, vnode2) // true
```

索引目录

02: 前置知识: V

📄

?

📱

💬

这个概念在 `diff` 中非常重要，它决定了 `ELEMENT` 的 **复用** 逻辑。具体怎么复用，我们会在后面进行详细讲解。

对于现在的代码而言，目前还不能增加 `key` 属性，所以我们需要为我们的代码增加 `key` 属性的挂载：

1. 在 `packages/runtime-core/src/vnode.ts` 的 `createBaseVNode` 中，增加 `key` 属性：

<> 代码块

```
1   const vnode = {
2     __v_isVNode: true,
3     type,
4     props,
5     shapeFlag,
6     + key: props?.key || null
7   } as VNode
```

这样，我们的 `vnode` 就可以具备 `key` 属性了。

01: 前言 ◀ 上一节

下一节 ▶ 03: 源码阅读：场景一：自前向后的 `diff` 对比

✎ 我要提出意见反馈

企业服务 网站地图 网站首页 关于我们 联系我们 讲师招募 帮助中心 意见反馈 代码托管

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



✎ 意见反馈

♥ 收藏教程

🔖 标记书签