

🔖 标记书签

```

12 // 新节点的 shapeFlag
13 const { shapeFlag } = newVNode
14
15 // 新子节点为 TEXT_CHILDREN
16 if (shapeFlag & ShapeFlags.TEXT_CHILDREN) {
17   // 旧子节点为 ARRAY_CHILDREN
18   if (prevShapeFlag & ShapeFlags.ARRAY_CHILDREN) {
19     // TODO: 卸载旧子节点
20   }
21   // 新旧子节点不同
22   if (c2 !== c1) {
23     // 挂载新子节点的文本
24     hostSetElementText(container, c2 as string)
25   }
26 } else {
27   // 旧子节点为 ARRAY_CHILDREN
28   if (prevShapeFlag & ShapeFlags.ARRAY_CHILDREN) {
29     // 新子节点也为 ARRAY_CHILDREN
30     if (shapeFlag & ShapeFlags.ARRAY_CHILDREN) {
31       // TODO: 这里要进行 diff 运算
32     }
33     // 新子节点不为 ARRAY_CHILDREN, 则直接卸载旧子节点
34     else {
35       // TODO: 卸载
36     }
37   } else {
38     // 旧子节点为 TEXT_CHILDREN
39     if (prevShapeFlag & ShapeFlags.TEXT_CHILDREN) {
40       // 删除旧的文本
41       hostSetElementText(container, '')
42     }
43     // 新子节点为 ARRAY_CHILDREN
44     if (shapeFlag & ShapeFlags.ARRAY_CHILDREN) {
45       // TODO: 单独挂载新子节点操作
46     }
47   }
48 }
49 }

```

#### 4. 创建 patchProps 方法:

<> 代码块

```

1  /**
2   * 为 props 打补丁
3   */
4  const patchProps = (el: Element, vnode, oldProps, newProps) => {
5    // 新旧 props 不相同时才进行处理
6    if (oldProps !== newProps) {
7      // 遍历新的 props, 依次触发 hostPatchProp, 赋值新属性
8      for (const key in newProps) {
9        const next = newProps[key]
10       const prev = oldProps[key]
11       if (next !== prev) {
12         hostPatchProp(el, key, prev, next)
13       }
14     }
15     // 存在旧的 props 时
16     if (oldProps !== EMPTY_OBJ) {
17       // 遍历旧的 props, 依次触发 hostPatchProp, 删除不存在于新props 中的旧属性
18       for (const key in oldProps) {
19         if (!(key in newProps)) {
20           hostPatchProp(el, key, oldProps[key], null)
21         }
22       }
23     }
24   }
25 }

```

至此, 更新操作完成。

[意见反馈](#)

[收藏教程](#)

[标记书签](#)



<> 代码块

```
1  <script>
2    const { h, render } = Vue
3
4    const vnode = h('div', {
5      class: 'test'
6    }, 'hello render')
7
8
9    render(vnode, document.querySelector('#app'))
10
11    // 延迟两秒，生成新的 vnode，进行更新操作
12    setTimeout(() => {
13      const vnode2 = h('div', {
14        class: 'active'
15      }, 'update')
16      render(vnode2, document.querySelector('#app'))
17    }, 2000);
18  </script>
```

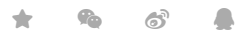
测试更新成功。

06: 源码阅读：渲染更新，ELEMENT 节点... < 上一节 下一节 > 08: 源码阅读：新旧节点不同元素时，ELEM...

 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