

全部开发者教程

三

载

14: 源码阅读：ELEMENT 节点下，style 属性的挂载和更新

15: 框架实现：ELEMENT 节点下，style 属性的挂载和更新

16: 源码阅读：ELEMENT 节点下，事件的挂载和更新

17: 深入事件更新：vue event invokers

18: 框架实现：ELEMENT 节点下，事件的挂载和更新

19: 局部总结：ELEMENT 节点的挂载、更新、props 打补丁等行为总结

20: 源码阅读：renderer 渲染器下，Text 节点的挂载、更新行为

21: 框架实现：renderer 渲染器下，Text 节点的挂载、更新行为



Sunday • 更新于 2022-10-19

上一节 17: 深入事件更... 19: 局部总结：... 下一节 >

18: 框架实现：ELEMENT 节点下，事件的挂载和更新

本小节我们就来实现事件的更新和挂载操作。

1. 在 packages/runtime-dom/src/patchProp.ts 中，增加 patchEvent 事件处理

<> 代码块

```
1    } else if (isOn(key)) {
2      // 事件
3      patchEvent(el, key, prevValue, nextValue)
4    }
```

2. 在 packages/runtime-dom/src/modules/events.ts 中，增加 patchEvent、parseName、createInvoker 方法：

<> 代码块

```
1  /**
2   * 为 event 事件进行打补丁
3   */
4  export function patchEvent(
5    el: Element & { _vei?: object },
6    rawName: string,
7    prevValue,
8    nextValue
9  ) {
10    // vei = vue event invokers
11    const invokers = el._vei || (el._vei = {})
12    // 是否存在缓存事件
13    const existingInvoker = invokers[rawName]
14    // 如果当前事件存在缓存，并且存在新的事件行为，则判定为更新操作。直接更新 invoker 的 value
15    if (nextValue && existingInvoker) {
16      // patch
17      existingInvoker.value = nextValue
18    } else {
19      // 获取用于 addEventListener || removeEventListener 的事件名
20      const name = parseName(rawName)
21      if (nextValue) {
22        // add
23        const invoker = (invokers[rawName] = createInvoker(nextValue))
24        el.addEventListener(name, invoker)
25      } else if (existingInvoker) {
26        // remove
27        el.removeEventListener(name, existingInvoker)
28        // 删除缓存
29        invokers[rawName] = undefined
30      }
31    }
32  }
33
34  /**
35   * 直接返回剔除 on，其余转化为小写的事件名即可
36   */
37  function parseName(name: string) {
38    return name.slice(2).toLowerCase()
```

索引目录

18: 框架实现：E



意见反馈

收藏教程

标记书签

```
41  /**
42   * 生成 invoker 函数
43   */
44   function createInvoker(initialValue) {
45     const invoker = (e: Event) => {
46       invoker.value && invoker.value()
47     }
48     // value 为真实的事件行为
49     invoker.value = initialValue
50     return invoker
51   }
```

支持事件的打补丁处理完成。

可以创建如下测试实例 `packages/vue/examples/runtime/render-element-event.html`：

<> 代码块

```
1  <script>
2    const { h, render } = Vue
3
4    const vnode = h('button', {
5      onClick() {
6        alert('点击')
7      },
8    }, '点击')
9    // 挂载
10   render(vnode, document.querySelector('#app'))
11
12   setTimeout(() => {
13     const vnode2 = h('button', {
14       // onClick() {
15       //   alert('click')
16       // },
17       onDblclick() {
18         alert('双击')
19       },
20     }, '双击')
21     // 挂载
22     render(vnode2, document.querySelector('#app'))
23   }, 2000);
24 </script>
```

测试成功

17: 深入事件更新: vue event invokers ◀ 上一节 下一节 ▶ 19: 局部总结: ELEMENT 节点的挂载、更...

 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