Sunday • 更新于 2022-10-19

上一节 08：源码阅读：… 10：框架实现：… 下一节

09：框架实现：处理新旧节点不同元素时，ELEMENT 节点的更新操作

根据上一小节我们知道，当新旧节点不同元素时，执行的是一个 **先删除、后挂载** 。依据此思路，我们可以直接进行对应的实现。

1. 在 `packages/runtime-core/src/renderer.ts` 的 `patch` 方法中增加 `type` 判断：

代码块

```
    /**
     * 判断是否为相同类型节点
     */
    if (oldVNode && !isSameVNodeType(oldVNode, newVNode)) {
        unmount(oldVNode)
        oldVNode = null
    }
```

2. 在 `packages/runtime-core/src/vnode.ts` 中，创建 `isSameVNodeType` 方法：

代码块

```
    /**
     * VNode
     */
    export interface VNode {
        key: any
        ...
    }

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

3. 在 `packages/runtime-core/src/renderer.ts` 实现 `unmount` 方法：

代码块

```
    export interface RendererOptions {
      /**
         * 卸载指定dom
         */
        remove(el): void
    }

        /**
         * 解构 options，获取所有的兼容性方法
         */
        const {
            ...
            remove: hostRemove
        } = options

    const unmount = (vnode) => {
```

4. 在 `packages/runtime-dom/src/nodeOps.ts` 中，实现 `remove` 方法：

代码块

```
  /**
  * 删除指定元素
  */
  remove: (child) => {
    const parent = child.parentNode
    if (parent) {
      parent.removeChild(child)
    }
  }
```

此时代码完成。

创建对应测试实例 `packages/vue/examples/runtime/render-element-update-2.html`：

代码块

```
<script>
  const { h, render } = Vue

  const vnode = h('div', {
    class: 'test'
  }, 'hello render')


  render(vnode, document.querySelector('#app'))

  // 延迟两秒，生成新的 vnode，进行更新操作
  setTimeout(() => {
    const vnode2 = h('h1', {
      class: 'active'
    }, 'update')
    render(vnode2, document.querySelector('#app'))
  }, 2000);
</script>
```

测试成功

08：源码阅读：新旧节点不同元素时，ELEM... ◂ 上一节　　下一节 ▸ 10：框架实现：删除元素，ELEMENT 节点...

我要提出意见反馈