Sunday • 更新于 2022-10-19

上一节 16：源码阅读：… 18：框架实现：… 下一节

17：深入事件更新：vue event invokers

那么这一小节我们来看下 ** `invokers` 对象 和 `invoker` 函数 （即：vue event invokers。简称： `vei` ）** 在事件处理中存在的作用。

在上一小节中，我们进行了两次挂载和一次卸载操作，本质上并没有对事件进行 **更新** 操作。而 `vei` 的作用是在更新中，才可以体现的。

我们知道， `vue` 对事件的处理是通过：

1. `el.addEventListener`
2. `el.removeEventListener`

来完成的。

那么现在我们来思考一下：

> 如果一个 `button` 最初的 `click` 事件，点击之后打印 `hello`
> 两秒之后，更新打印 `你好`

那么这样的一个更新操作，如果让我们通过 `el.addEventListener` 和 `el.removeEventListener` 来实现，那么我们会怎么做？

可能有同学说，这还不简单吗？很轻松的写出如下代码：

代码块

```
<script>
  const btnEle = document.querySelector('button')
  // 设置初始点击行为
  const invoker = () => {
    alert('hello')
  }
  btnEle.addEventListener('click', invoker)

  // 两秒之后，更新点击事件
  setTimeout(() => {
    // 先删除
    btnEle.removeEventListener('click', invoker)
    // 再添加
    btnEle.addEventListener('click', () => {
      alert('你好')
    })
  }, 2000);

</script>
```

但是我们知道如果频繁的删除、新增事件是非常消耗性能的，那么有没有更好的方案呢？

肯定是有的，这个方案就是 `vei` ，我们来看下面这段代码：

代码块

```
<script>
```

```
    const invoker = () => {
      invoker.value()
    }
    // 为 invoker 指定了 value 属性，对应的值是《事件点击行为》
    invoker.value = () => {
      alert('hello')
    }
    // 把 invoker 作为回调函数，invoker 内部通过触发 value，来触发真正的点击行为
    btnEle.addEventListener('click', invoker)

    // 两秒之后更新
    setTimeout(() => {
      // 因为真正的事件点击行为其实是 invoker.value，所以我们想要更新事件，就不需要再次触发 addE
      invoker.value = () => {
        alert('你好')
      }
    }, 2000);
  </script>
```

vue 就是通过这样一种方式，来完成的事件更新操作。具体更新代码比较简单，可以看 packages/runtime-dom/src/modules/events.ts 第 77 行：

代码块

```
    if (nextValue && existingInvoker) {
      // patch
      existingInvoker.value = nextValue
    }
```

至于 invokers 则充当了一个事件缓存器，把所有的事件：**以事件名为 key，以事件行为为 value 。保存到 el._vei 中。**

那么这样我们搞明白了 vei 它在事件处理中所存在的意义。

明白了这个之后，下面我们就可以实现 event 事件的挂载和更新操作了。

16：源码阅读：ELEMENT 节点下，事件的... 上一节 下一节 18：框架实现：ELEMENT 节点下，事件的...

我要提出意见反馈