Sunday • 更新于 2022-10-19

上一节 06：框架实现：... 08：框架实现：... 下一节

07：源码阅读：场景三：新节点多余旧节点时的 diff 比对

以上两种场景，新节点数量和旧节点数量都是完全一致的。

但是我们也知道一旦产生更新，那么新旧节点的数量是可能会存在不一致的情况，具体的不一致情况会分为两种：

1. 新节点的数量多于旧节点的数量
2. 旧节点的数量多于新节点的数量

那么针对于这两种情况怎么进行处理呢？这就是我们接下来需要说的。

本小节，我们先来看一下 **新节点的数量多于旧节点的数量** 的场景，一旦出现这种场景，那么我们就需要 **挂载多余的新节点。**

但是新节点的数量多于旧节点的数量的场景下，依然可以被细分为两种具体的场景：

1. 多出的新节点位于 **尾部**
2. 多出的新节点位于 **头部**

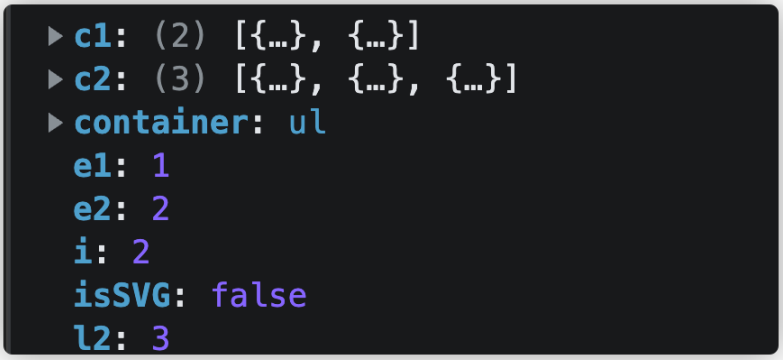
明确好了以上内容之后，我们来看如下测试实例 `packages/vue/examples/imooc/runtime/render-element-diff-3.html`：

代码块

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

  const vnode = h('ul', [
    h('li', {
      key: 1
    }, 'a'),
    h('li', {
      key: 2
    }, 'b'),
  ])
  // 挂载
  render(vnode, document.querySelector('#app'))

  // 延迟两秒，生成新的 vnode，进行更新操作
  setTimeout(() => {
    const vnode2 = h('ul', [
      h('li', {
        key: 1
      }, 'a'),
      h('li', {
        key: 2
      }, 'b'),
      h('li', {
        key: 3
      }, 'c')
    ])
    render(vnode2, document.querySelector('#app'))
  }, 2000);
</script>
```

根据以上代码进入 `debugger`，忽略掉前两种场景，直接从第三种场景开始：

1. 代码进入场景三 `3. common sequence + mount`，此时各参数的值为：



2. 满足 `if (i > e1)` 和 `if (i <= e2)` 的场景，进入 `if`

    1. 执行：

    ```
    const nextPos = e2 + 1
    const anchor = nextPos < l2 ? (c2[nextPos] as VNode).el : parentAnchor
    ```

        1. 这两行代码的目的是为了计算出 `anchor`，也就是**新增节点插入的 锚点（位置）**。
        2. 当前场景下 `anchor = parentAnchor`，即：使用父级的 `anchor`

    2. 执行 `while (i <= e2)`，进入循环

    3. 每次循环时，执行 `patch` 方法，新增节点即可。

    4. 最后执行 `i++`

以上逻辑为：多出的新节点位于 **尾部** 的场景。

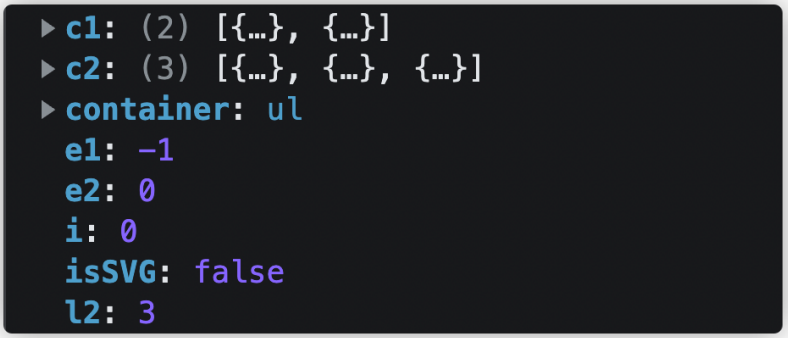
那么接下来我们来看：多出的新节点位于 **头部** 的场景：

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

  const vnode = h('ul', [
    h('li', {
      key: 1
    }, 'a'),
    h('li', {
      key: 2
    }, 'b'),
  ])
  // 挂载
  render(vnode, document.querySelector('#app'))

  // 延迟两秒，生成新的 vnode，进行更新操作
  setTimeout(() => {
    const vnode2 = h('ul', [
      h('li', {
        key: 3
      }, 'c'),
      h('li', {
        key: 1
      }, 'a'),
      h('li', {
        key: 2
      }, 'b')
    ])
```

```
29      }, 2000);
30    </script>
```

根据以上代码，再次进入情景三：

1. 代码进入场景三 `3. common sequence + mount`，此时各参数的值为：



2. 代码执行：

代码块

```
1     const nextPos = e2 + 1
2     const anchor = nextPos < l2 ? (c2[nextPos] as VNode).el : parentAnchor
```

    1. 此时 `nextPos > l2`，所以 `anchor` 的值为 `c2[1]`，即：新增的节点位于 `h('li', { key: 1 }, 'a')` **之前**

3. 后续的代码执行略过

由以上代码可知：

1. 对于 **新节点多余旧节点** 的场景具体可以在细分为两种情况：
    1. 多出的新节点位于 **尾部**
    2. 多出的新节点位于 **头部**
2. 这两种情况下的区别在于：**插入的位置不同**
3. 明确好插入的位置之后，直接通过 `patch` 进行打补丁即可。

06：框架实现：场景二：自后向前的 diff 对比 ‹ 上一节　　下一节 › 08：框架实现：场景三：新节点多余旧节点...

我要提出意见反馈