Sunday • 更新于 2022-10-19

上一节 03：挂载与更新 05：运行时核心… 下一节

04：h 函数 与 render 函数

不知道大家还记不记得，我们在 **第二章** 讲解 **运行时** 的概念时，曾经使用过这样的一个案例：

代码块

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <title>Document</title>
  <script src="https://unpkg.com/vue@3.2.36/dist/vue.global.js"></script>
</head>

<body>
  <div id="app"></div>
</body>

<script>
  const { render, h } = Vue
  // 生成 VNode
  const vnode = h('div', {
    class: 'test'
  }, 'hello render')

  // 承载的容器
  const container = document.querySelector('#app')

  // 渲染函数
  render(vnode, container)
</script>

</html>
```

当时，我们说：“有些同学可能看不懂当前代码是什么意思，没有关系，这不重要，后面我们会详细去讲”，那么现在就是讲解这个的时候了。

根据前两个小节的介绍其实我们已经知道了， `vue` 的渲染分为：挂载和更新。两个步骤。

而无论是挂载还是更新，都是借助 `VNode` 来进行实现的。

在以上代码中，我们知道主要涉及到了两个函数：

1. `h` 函数
2. `render` 函数

那么下面我们一个一个来去说

h 函数

根据以上代码可知，我们可以通过 `h 函数` 得到一个 `vnode`：

代码块

```
  const vnode = h('div', {
```

打印当前的 `vnode`，可以得到一下内容：

代码块

```
{
    "__v_isVNode": true,
    "__v_skip": true,
    "type": "div",
    "props": { "class": "test" },
    "key": null,
    "ref": null,
    "scopeId": null,
    "slotScopeIds": null,
    "children": "hello render",
    "component": null,
    "suspense": null,
    "ssContent": null,
    "ssFallback": null,
    "dirs": null,
    "transition": null,
    "el": null,
    "anchor": null,
    "target": null,
    "targetAnchor": null,
    "staticCount": 0,
    "shapeFlag": 9,
    "patchFlag": 0,
    "dynamicProps": null,
    "dynamicChildren": null,
    "appContext": null
}
```

以上内容，我们剔除掉无用的内容之后，得到一个精简的 `json`：

代码块

```
{
  // 是否是一个 VNode 对象
    "__v_isVNode": true,
  // 当前节点类型
    "type": "div",
  // 当前节点的属性
    "props": { "class": "test" }
  // 它的子节点
    "children": "hello render"
}
```

那么由此可知 `h` 函数本质上其实就是一个：**用来生成 `VNode` 的函数。**

h 函数 最多可以接收三个参数：

1. `type: string | Component`： 既可以是一个字符串 (用于原生元素) 也可以是一个 Vue 组件定义。
2. `props?: object | null`： 要传递的 prop
3. `children?: Children | Slot | Slots`： 子节点。

官方示例描述了它的详细使用方式，这个就不写在文档中了。

render 函数

那么在了解了 `h` 函数的作用之后，下面我们来看 render 函数。

代码块

```
    render(vnode, container)
```

从以上代码中我们可知， `render` 函数主要接收了两个参数：

1. `vnode`： 虚拟节点树 或者叫做 虚拟 `DOM` 树，两种叫法皆可

通过 `render` 函数，我们可以：**使用编程式地方式，创建虚拟 DOM 树对应的真实 `DOM` 树，到指定位置。**

总结

这小节，我们知道了 `h` 函数和 `render` 函数的作用，这两个函数是整个运行时的关键函数，后面我们实现运行时的代码，核心就是实现这两个函数。

03：挂载与更新 ◂ 上一节　　下一节 ▸ 05：运行时核心设计原则

我要提出意见反馈