慕课网首页 免费课 实战课 体系课 慕课教程 专栏 手记 企业服务 Ē 我的课程



Q

从所有教程的词条中查询…

07: 框架实现: ... 下一节 ▶

Sunday • 更新于 2022-10-19

## 全部开发者教程 Ξ

04:源码阅读: h 函数,跟踪 ELEMENT + ARRAY\_CHILDREN 场景下的 源码实现

05: 框架实现: 构建 h 函数, 处理 ELEMENT + ARRAY\_CHILDREN 场景

06: 源码阅读: h 函数, 组件 的本质与对应的 VNode

07: 框架实现: 处理组件的 VNode

08: 源码阅读: h 函数, 跟踪 Text 、 Comment、 Fragment 场景

09: 框架实现: 实现剩余场景 Text09: 框架实现: 实现剩余 场景 Text 、 Comment、 Fragment

10: 源码阅读: 对 class 和 style 的增强处理

06: 源码阅读: h 函数, 组件的本质与对应的 VNode

首页 > 慕课教程 > Vue3源码分析与构建方案 > 06:源码阅读: h 函数,组件的本质与对应的 V...

组件是 vue 中非常重要的一个概念,这一小节我们就来看一下 组件 生成 VNode 的情况。

◆ 上一节 05: 框架实现: ...

在 vue 中,组件本质上是 一个对象或一个函数 (Function Component)

我们这里 不考虑 组件是函数的情况,因为这个比较少见。

我们可以直接利用 h 函数 + render 函数渲染出一个基本的组件:

1. 创建 packages/vue/examples/imooc/runtime/h-component.html

```
<> 代码块
     <script>
       const { h, render } = Vue
 3
 4
       const component = {
 5
        render() {
           const vnode1 = h('div', '这是一个 component')
 6
 7
           console.log(vnode1);
 8
           return vnode1
 9
10
       const vnode2 = h(component)
11
12
       console.log(vnode2);
13
       render(vnode2, document.querySelector('#app'))
14
15
     </script>
```

- 2. 在当前代码中共触发了两次 n 函数, 我们来查看两次打印的结果:
  - 1. vnode 2:

```
{_v_isVNode: true, _v_skip: true, type: {_}, props: null, key: null, _, } @
anchor: null
appContext: null
children: null
> component: {uid: 0, vnode: {_}, type: {_}, parent: null, appContext: {__}, _, }
distant null
    dirs: nuct
dynamicChildren: null
dynamicProns: null
dynamicChildrdynamicProps:
▶el: div
key: null
patchFlag: 0
props: null
ref: null
scopeId: null
shaneFlag: 4
  scopeId: null
shapeFlag: 4
slotScopeIds: null
ssContent: null
ssFallback: null
staticCount: 0
    suspense: null
target: null
targetAnchor: null
transition: null
        Feturn v
}

▶[[Prototype]]: Object
_v_isVNode: true
_v_skip: true
```

▶ 意见反馈

♡ 收藏教程

□ 标记书签

索引目录

06: 源码阅读: h

?

 $\odot$ 

- 1. shapeFlag: 这个是当前的类型表示, 4 表示为一个组件
- 2. type: 是一个对象,它的值包含了一个 render 函数,这个就是 component 的 真实 渲染 内容
- 3. \_\_v\_isVNode: VNode 标记
- 2. vnode1:与 ELEMENT + TEXT\_CHILDREN 相同

```
1 {
2    __v_isVNode: true,
3    type: "div",
4    children: "这是一个 component",
5    shapeFlag: 9
6  }
```

那么由此可知,对于组件而言,它的一个渲染,与之前不同的地方主要有两个:

- 1. shapeFlag === 4
- 2. type: 是一个 对象 (组件实例) , 并且包含 render 函数

仅此而已,那么依据这样的概念,我们可以通过如下代码,完成同样的渲染:

```
<> 代码块
const component = {
2 render() {
       return {
3
          "__v_isVNode": true,
 4
       "__v_+...
"type": "div",
 5
         "children": "这是一个 component",
 6
         "shapeFlag": 9
   }
       }
 8
9
   }
10
11
12 render({
13 "__v_isVNode": true,
      "type": component,
14
15
      "shapeFlag": 4
16 }, document.querySelector('#app'))
```

✓ 我要提出意见反馈

?

0

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

\* **%** 6 **.** 

Copyright © 2022 imooc.com All Rights Reserved | 京ICP备 12003892号-11 京公网安备11010802030151号