

全部开发者教程

09: 框架实现：初步实现 watch 数据监听器

10: 问题分析：watch 下的依赖收集

11: 框架实现：完成 watch 数据监听器的依赖收集

12: 总结：watch 数据侦听器

13: 总结

第八章：runtime 运行时 - 运行时核心设计原则

01: 前言

02: HTML DOM 节点树与虚拟 DOM 树

03: 挂载与更新

04: h 函数与 render 函数

05: 运行时核心设计原则

06: 总结

第九章：runtime 运行时 - 构建 h 函数，生成 Vnode

Sunday • 更新于 2022-10-19

13: 总结02: HTML DO... 下一节

01：前言

从本章开始我们将要进入到 **渲染系统** 的学习，也就是 **运行时 runtime**。

在之前我们明确过什么是 **运行时**，看下面的代码（第二章运行时使用过该代码）：

<> 代码块

```
1 <body>
2   <div id="app"></div>
3 </body>
4 <script>
5   const { render, h } = Vue
6   // 生成 VNode
7   const vnode = h('div', {
8     class: 'test'
9   }, 'hello render')
10
11   // 承载的容器
12   const container = document.querySelector('#app')
13
14   // 渲染函数
15   render(vnode, container)
16 </script>
```

以上代码代表了一个基本的 **运行时**。即：把 **VNode** **渲染到页面中**。所以大家可以简单的把运行时理解为 **渲染系统**。

根据以上代码我们可以知道，整个 **runtime**，包含了两个主要的环节：

1. h 函数：生成 VNode

2. render 函数：渲染 VNode

这也是 **正式讲解 runtime** 的时候主要要去讲解的两大块内容。

但是在正式讲解 **runtime** 之前，我们还需要了解一些 **runtime** 的基本概念和设计原则，比如：**VNode** 是什么？它在干嘛？为什么需要它？里面传递的参数又是干什么的？为什么要传递它们？等等...

而这些就是我们本章所需要讲解的主要内容。

13: 总结 13: 总结 下一节 02: HTML DOM 节点树与虚拟 DOM 树

我要提出意见反馈

索引目录

01: 前言