

全部开发者教程

Vue 3 源码分析

03: 如何让你的程序变得更加“聪明”？

04: vue 2 的响应性核心 API: Object.defineProperty

05: Object.defineProperty 在设计层的缺陷

06: vue3的响应性核心 API: proxy

07: proxy的最佳拍档: Reflect—拦截 js 对象操作

08: 总结

第五章：响应系统 - 初见 reactivity 模块

01: 前言

02: 源码阅读: reactive 的响应性, 跟踪 Vue 3 源码实现逻辑

03: 框架实现: 构建 reactive 函数, 获取 proxy 实例



Sunday • 更新于 2022-10-19

上一节 04: vue 2 的响... 06: vue3的响应... 下一节

## 05: Object.defineProperty 在设计层的缺陷

vue2 使用 `Object.defineProperty` 作为响应性的核心 API，但是在 vue3 的时候却放弃了这种方式，转而使用 `Proxy`（后面会详细讲解，现在只需要知道即可）实现，为什么会这样呢？

这是因为：`Object.defineProperty` 存在一个致命的缺陷

在 vue 官网中存在这样的一段描述：

由于 JavaScript 的限制，Vue 不能检测数组和对象的变化。尽管如此我们还是有一些办法来回避这些限制并保证它们的响应性。

他说：由于 JavaScript 的限制，Vue 不能检测数组和对象的变化 这是什么意思呢？

我们来看下面的这个例子：

<> 代码块

```
1  <template>
2    <div id="app">
3      <ul>
4        <li v-for="(val, key, index) in obj" :key="index">
5          {{ key }} - {{ val }}
6        </li>
7      </ul>
8      <button @click="addObjKey">为对象增加属性</button>
9    </div>
10   <ul>
11     <li v-for="(item, index) in arr" :key="index">
12       {{ item }}
13     </li>
14   </ul>
15   <button @click="addArrItem">为数组添加元素</button>
16 </div>
17 </template>
18
19 <script>
20 export default {
21   name: 'App',
22   data() {
23     return {
24       obj: {
25         name: '张三',
26         age: 30
27       },
28       arr: ['张三', '李四']
29     }
30   },
31   methods: {
32     addObjKey() {
33       this.obj.gender = '男'
34       console.log(this.obj) // 通过打印可以发现，obj 中存在 gender 属性，但是视图中并没有更新
35     },
36     addArrItem() {
37       this.arr[2] = '王五'
```

索引目录

05: Object.defir



意见反馈

收藏教程

标记书签

现

```
39         }  
40     }  
41 }  
42 </script>
```

在上面的例子中，我们呈现了 vue2 中响应性的限制：

1. 当为 **对象** 新增一个没有在 `data` 中声明的属性时，新增的属性 **不是响应性** 的
2. 当为 **数组** 通过下标的形式新增一个元素时，新增的元素 **不是响应性** 的

那么为什么会这样呢？

想要搞明白这个原因，那就需要明白官网所说的 **由于 JavaScript 的限制** 指的是什么意思。

我们知道

1. vue 2 是以 `Object.defineProperty` 作为核心 API 实现的响应性
2. `Object.defineProperty` 只能监听 **指定对象的指定属性的 getter 和 setter**
3. 被监听了 `getter` 和 `setter` 的属性，就被叫做 **该属性具备了响应性**

那么这就意味着：我们 **必须要知道指定对象中存在该属性**，才可以为该属性指定响应性。

但是 **由于 JavaScript 的限制**，我们没有办法监听到 **指定对象新增了一个属性**，所以新增的属性就没有办法通过 `Object.defineProperty` 来监听 `getter` 和 `setter`，所以 **新增的属性将失去响应性**

那么如果想要增加具备响应性的新属性，那么可以通过 `Vue.set` 方法实现

那么此时，我们已经知道了这些 vue2 中的“缺陷”，那么 vue3 是如何解决这些缺陷的呢？我们继续来往下看~~~~

04: vue 2 的响应性核心 API: Object.defineProperty... < 上一节      下一节 > 06: vue3的响应性核心 API: proxy

 我要提出意见反馈

