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

Q 🃜 💄 我的课程

Q

从所有教程的词条中查询…

首页 > 慕课教程 > Vue3源码分析与构建方案 > 04:框架实现: computed 的响应性:初见调...

全部开发者教程 :≡

97. 心知. 151 同平双油大土 响应性

08: 总结

第七章:响应系统 computed && watch

01: 开篇

02: 源码阅读: computed 的响应

03: 框架实现: 构建 ComputedRefImpl , 读取计 算属性的值

04: 框架实现: computed 的响应性: 初见调度器,处理脏的状态

05: 框架实现: computed 的 缓存

06: 总结: computed 计算属 性

07:源码阅读:响应性的数据 监听器 watch, 跟踪源码实现 逻辑

Sunday・更新于 2022-10-19

◆ 上一节 03: 框架实现: ...

05: 框架实现: ... 下一节 ▶

04: 框架实现: c

索引目录

调度器 _dirty 脏 实现

04: 框架实现: computed 的响应性: 初见调度器, 处理脏的状态

根据之前的代码可知,如果我们想要实现 **响应性**,那么必须具备两个条件:

1. 收集依赖: 该操作我们目前已经在 get value 中进行。

2. 触发依赖: 该操作我们目前尚未完成, 而这个也是我们本小节主要需要做的事情。

那么根据第二小节的源码可知,这部分代码是写在 ReactiveEffect 第二个参数上的:

这个参数是一个匿名函数, 被叫做 scheduler 调度器。

该匿名函数中,又涉及到了一个 _dirty 变量,该变量我们把它叫做 脏。

那么想要实现 computed 的响应性, 就必须要搞明白这两个东西的概念:

调度器

调度器 scheduler 是一个相对比较复杂的概念,它在 computed 和 watch 中都有涉及,但是在当前的 computed 实现中,它的作用还算比较清晰。

所以根据我们秉承的: **没有使用就当做不存在**的理念,我们只需要搞清楚,它在当前的作用即可。

根据我们在第二小节的源码阅读,我们可以知道,此时的 scheduler 就相当于一个回调函数。

在 triggerEffect 只要 effect 存在 scheduler,则就会执行该函数。

_dirty 脏

对于 dirty 而言, 相对比较简单了。

它只是一个变量,我们只需要知道: 它为 false 时,表示需要触发依赖。为 true 时表示需要重新执行 run 方法,获取数据。即可。

实现

那么明确好了以上两个概念之后,接下来我们就来进行下 computed 的响应性实现:

1. 在 packages/reactivity/src/computed.ts 中, 处理脏状态和 scheduler:

② []

0

```
9
      constructor(getter) {
10
          this.effect = new ReactiveEffect(getter, () => {
11
             // 判断当前脏的状态,如果为 false,表示需要《触发依赖》
12
              if (!this._dirty) {
                 // 将脏置为 true,表示
13
                 this._dirty = true
14
                 triggerRefValue(this)
15
              }
17
           })
18
           this.effect.computed = this
19
       }
20
     get value() {
21
        // 触发依赖
22
23
           trackRefValue(this)
24
          // 判断当前脏的状态,如果为 true ,则表示需要重新执行 run,获取最新数据
          if (this._dirty) {
25
              this._dirty = false
26
27
              // 执行 run 函数
              this._value = this.effect.run()!
28
29
30
          // 返回计算之后的真实值
31
32
          return this._value
33
      }
   }
34
```

2. 在 packages/reactivity/src/effect.ts 中, 添加 scheduler 的处理:

```
<> 代码块
    export type EffectScheduler = (...args: any[]) => any
1
3
4
     * 响应性触发依赖时的执行类
5
 6
7
    export class ReactiveEffect<T = any> {
8
       * 存在该属性,则表示当前的 effect 为计算属性的 effect
9
10
11
      computed?: ComputedRefImpl<T>
12
13
      constructor(
        public fn: () => T,
14
          public scheduler: EffectScheduler | null = null
15
      ) {}
16
17
   }
18
```

?

0

3. 最后不要忘记,触发调度器函数

```
<> 代码块
1 /**
2 * 触发指定依赖
3
4
    export function triggerEffect(effect: ReactiveEffect) {
     // 存在调度器就执行调度函数
if (effect.scheduler) {
7
      effect.scheduler()
8
    // 否则直接执行 run 函数即可
9
    else {
10
11
      effect.run()
12
    }
13 }
```

此时, 重新执行测试实例, 则发现 computed 以具备响应性。

▶ 我要提出意见反馈

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 A

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

∷

?

.

0