Sunday • 更新于 2022-10-19

上一节 14：reactive 函… 01：前言 下一节

15：总结

在本章，我们初次解除了 `reactivity` 模块，并且在该模块中构建了 `reactive` 响应性函数。

对于 `reactive` 的响应性函数而言，我们知道它：

1. 是通过 `proxy` 的 `setter` 和 `getter` 来实现的数据监听
2. 需要配合 `effect` 函数进行使用
3. 基于 `WeakMap` 完成的依赖收集和处理
4. 可以存在一对多的依赖关系

同时我们也了解了 `reactive` 函数的不足：

1. `reactive` 只能对 **复杂数据** 类型进行使用
2. `reactive` 的响应性数据，不可以进行解构

因为 `reactive` 的不足，所以 `vue 3` 又为我们提供了 `ref` 函数构建响应性，那么：

1. `ref` 函数的内容是如何进行实现的呢？
2. `ref` 可以构建简单数据类型的响应性吗？
3. 为什么 `ref` 类型的数据，必须要通过 `.value` 访问值呢？

带着以上三个问题，我们来看下一章 `ref` **的响应性** ~~~~

14：reactive 函数的局限性 上一节 下一节 01：前言

我要提出意见反馈