Sunday • 更新于 2022-10-19

上一节 05：授人以鱼：… 07：开始搭建自… 下一节

06：授人以渔：如何阅读源码

那么在上一小节中我们已经知道了如何对 `vue` 的源代码进行 `debugger` ，但是如果想要学习或者了解 `vue` 的代码执行，那么光靠 `debugger` 是不够的，除此之外我们还需要掌握另外一个能力，那么 **如何阅读源代码**

阅读源代码的误区

很多同学在去阅读源代码的时候，都会面临一个 **误区**，那就是：**我需要把源代码中每一行代码都读明白。**

这是一个非常不对的行为，很容易让我们 **事倍功半**

所以在这里我们需要先给大家明确一点：**阅读源码绝对不是要读明白其中每一行代码的意思，而是在众多的业务代码中寻找到主线，跟随这个主线来进行阅读**

阅读源码的正确姿势

想要快速、轻松的阅读源码，正确的姿势非常重要，主要有两点：

1. 摒弃边缘情况
2. 跟随一条主线

摒弃边缘情况

在大型项目的源码中，都会充斥着非常多的业务代码，这些业务代码是用来处理很多 **边缘情况** 的，如果我们过分深究这些业务代码则会让我们陷入到一个 **代码泥潭** 中，在繁琐的业务中找不到方向。

所以，我们在阅读源码之前，必须要明确好一点，那就是：**仅阅读核心逻辑**

跟随一条主线

对于像 `vue` 这种量级的项目来说，哪怕我们只去阅读它的核心代码，你也会发现也是非常困难的。我们之前说过， `vue` 的核心大致可以分为三块：

1. 响应性
2. 运行时
3. 编译器

每一大块的内部又分为了很多的业务分支。所以哪怕仅阅读核心代码已然是一个浩大的工作量。

所以说我们还需要另外一个方式，那就是：**跟随一条主线**

> 举个例子：我们以前面的 `packages/vue/examples/imooc/reactive.html` 为例

在该代码中，我们通过 `reactive` 声明了一个响应式数据：

代码块

```
1    // 声明响应式数据 obj
2    const obj = reactive({
3      name: '张三'
```

那么我们就可以以改代码为主线，来去查看 `reactive` 方法的主线逻辑：

1. 首先在 `reactive` 方法中进行了一个判断逻辑，判断 `target` 是否为只读的，如果是只读的就直接返回 `target`，意思是：传的是啥返回啥。
2. 如果不是只读的，则触发 `createReactiveObject` 方法：

```
90  export function reactive(target: object) {
91    // if trying to observe a readonly proxy, return the readonly version.
92    if (isReadonly(target)) {
93      return target
94    }
95    return createReactiveObject(
96      target,
97      false,
98      mutableHandlers,
99      mutableCollectionHandlers,
100     reactiveMap
101   )
102 }
```

3. 在 `createReactiveObject` 方法中，又进行了一堆判断，最后返回了 `proxy` 实例对象，所以我们得到的 `obj` 应该就是一个 `proxy` 实例

```
211   }
212   const proxy = new Proxy(   proxy = Proxy {name: '张三'}
213     target,   target = {name: '张三'}
214     targetType === TargetType.COLLECTION ? collectionHandlers : baseHan
215   )
216   proxyMap.set(target, proxy)
217   return proxy
218 }
```

4. 打印 `obj` 你会发现确实如此

这样的一个简单的例子，就是告诉大家应该如何来通过 `debugger` **配合** 正确姿势 来快速的阅读源代码。

总结

这一小节我们讲解了如何阅读源代码，以上方式不光可以应用到 `vue` 中，也可以应用到其他的框架之中，所以我们把这一小节叫做 **授人以渔。**

当然，我们这里只是通过一个简单的方式来进行了举例，在大家实际阅读的过程之中，肯定还是会遇到很多的困难的，不过好在，在这个过程中，我们会一起来进行阅读~~

05：授人以鱼：如何针对源码进行 debugger ◂ 上一节　　下一节 ▸ 07：开始搭建自己的框架：创建 vue-next-...

我要提出意见反馈