Sunday • 更新于 2022-10-19

上一节 04：框架实现：... 06：热更新的开... 下一节

05：框架实现：createGetter && createSetter

那么当我们搞明白了 `WeakMap` 的特性之后，那么接下来我们就来看看 `mutableHandlers` 如何处理。

我们知道对于 `Proxy` 而言，它的 `handler` 可以监听 **代理对象** 的 `getter` 和 `setter`，那么此时的 `mutableHandlers` 就是监听 **代理对象** `getter` 和 `setter` 的核心部分。

所以接下来我们需要创建对应的 `get` 和 `set` 监听：

代码块

```
/**
 * 响应性的 handler
 */
export const mutableHandlers: ProxyHandler<object> = {
    get,
    set
}
```

getter

代码块

```
/**
 * getter 回调方法
 */
const get = createGetter()

/**
 * 创建 getter 回调方法
 */
function createGetter() {
    return function get(target: object, key: string | symbol, receiver: object) {
        // 利用 Reflect 得到返回值
        const res = Reflect.get(target, key, receiver)
        // 收集依赖
        track(target, key)
        return res
    }
}
```

setter

代码块

```
/**
 * setter 回调方法
 */
const set = createSetter()

/**
 * 创建 setter 回调方法
 */
function createSetter() {
    return function set(
        target: object,
```

```
            receiver: object
        ) {
            // 利用 Reflect.set 设置新值
            const result = Reflect.set(target, key, value, receiver)
            // 触发依赖
            trigger(target, key, value)
            return result
        }
    }
```

track && trigger

在 `getter` 和 `setter` 中分别调用了 `track && trigger` 方法，所以我们需要分别创建对应方法：

1. 创建 `packages/reactivity/src/effect.ts`：

代码块

```
/**
 * 用于收集依赖的方法
 * @param target WeakMap 的 key
 * @param key 代理对象的 key，当依赖被触发时，需要根据该 key 获取
 */
export function track(target: object, key: unknown) {
    console.log('track: 收集依赖')
}

/**
 * 触发依赖的方法
 * @param target WeakMap 的 key
 * @param key 代理对象的 key，当依赖被触发时，需要根据该 key 获取
 * @param newValue 指定 key 的最新值
 * @param oldValue 指定 key 的旧值
 */
export function trigger(
    target: object,
    key?: unknown,
    newValue?: unknown
) {
    console.log('trigger: 触发依赖')
}
```

那么至此我们就可以：

1. 在 `getter` 时，调用 `track` 收集依赖
2. 在 `setter` 时，调用 `trigger` 触发依赖

我们可以在两个方法中分别进行一下打印，看看是否可以成功回调。

测试

在 `packages/vue/examples/reactivity/reactive.html` 中：

代码块

```
const { reactive } = Vue

const obj = reactive({
  name: '张三'
})
console.log(obj.name); // 此时应该触发 track
obj.name = '李四' // 此时应该触发 trigger
```

重新打包 项目，运行以上测试。