Rust 编程语言入门

杨旭，微软 MVP
Rust、Go 开发者

13.3 闭包（3）

- 使用泛型参数和 Fn Trait 来存储闭包

继续解决 13.1 中“运动计划”程序的问题

- 另一种解决方案：
- 创建一个 struct，它持有闭包及其调用结果。
  - 只会在需要结果时才执行该闭包
  - 可缓存结果
- 这个模式通常叫做记忆化（memoization ）或延迟计算（lazy evaluation）

如何让 struct 持有闭包

- struct 的定义需要知道所有字段的类型
  - 需要指明闭包的类型
- 每个闭包实例都有自己唯一的匿名类型，即使两个闭包签名完全一样。
- 所以需要使用：泛型和 Trait Bound（第10章）

Fn Trait

- Fn traits 由标准库提供
- 所有的闭包都至少实现了以下 trait 之一：
  - Fn
  - FnMut
  - FnOnce
- （例子）

使用缓存器（Cacher）实现的限制

1. Cacher 实例假定针对不同的参数 arg，value 方法总会得到同样的值。
   - （例子）
   - 可以使用 HashMap 代替单个值：
     - key：arg 参数
     - value：执行闭包的结果
2. 只能接收一个 u32 类型的参数和 u32 类型的返回值

再见