第五课:如何编写测试

单元测试、集成测试、基准测试

苏林

回顾上一次公开课的内容

第四课:Cargo包管理

为什么需要包管理器

Cargo new 命令

模块系统

今天公开课内容

1、单元测试

2、集成测试

3、基准测试

4、静态分析(lint)

5、模糊测试(fuzzing)

讨论一下, 为什么要测试?

如何没有单元测试? -> 大规模软件的重构.

rustc —test => 告诉rustc生成一个运行所有单元测试的测试二进制文件

Cargo test

二进制文件 -> 入口函数 -> 生成main函数 函数体内就是这些测试的定义 (cfg(test) #[test] + /tests目录下 + 注释下)

启很多线程并行运行 每个测试.

Cargo test RUST_TEST_THREADS=1

单元测试

```rust
#[cfg(test)]
mod tests {

    #[test]
    fn it_works() {
        assert_eq!(2 + 2, 4);
    }

}
```

分离测试代码

```rust
fn sum(a: i8, b: i8) -> i8 {a + b}

#[cfg(test)]
mod tests {
    fn sum_inputs_outputs() -> Vec<((i8, i8), i8)> {
        vec![((1, 1), 2), ((0, 0), 0), ((2, -2), 0)]
    }
    #[test]
    fn test_sums() {
        for (input : (i8, i8), output : i8) in sum_inputs_outputs() {
            assert_eq!(crate::sum( a: input.0,  b: input.1), output);
        }
    }
}

```

失败测试

`#[should_panic]`

忽略测试

```rust
pub fn silly_loop() {
    for _ in 1..1#[should_panic]};
}
#[cfg(test)]
mod tests {
    #[test]
    #[ignore]
    pub fn test_silly_loop() {
        ::silly_loop();
    }
}
```

集成测试

基准测试

QA环节

加群一起交流Rust & Databend

Databend

Rust语言中文社区

知数堂