Русский

Урок Спринт 3.2

Пример кода 2

Тестирование инкрементации

Предположим, у нас есть функция, которая инкрементирует общий счётчик:

```
// counter.go

package counter

import "sync"

var mu sync.Mutex
var counter int

func Increment() {
    mu.Lock()
    defer mu.Unlock()
    counter++
}

func GetCounter() int {
    mu.Lock()
    defer mu.Unlock()
    return counter
}
```

Теперь давайте напишем тесты для этого кода.

```
// counter_test.go

package counter

import (
    "sync"
    "testing"
)

func TestIncrement(t *testing.T) {
    // Сбросим счётчик
    counter = 0

    // Для ожидания горутин
    var wg sync.WaitGroup

    // Будем делать инкремент столько раз
    numIncrements := 1000

    // Для ожидания всех запущенных горутин
    wg.Add(numIncrements)

    // Увеличиваем значение счётчика конкурентно
    for i := 0; i < numIncrements; i++ {
        go func() {
            defer wg.Done()
            Increment()
        }()
    }

    // Подождём все горутины
```

```
    wg.Wait()

    // Проверим, получили ли ожидаемое значение
    expectedCounter := numIncrements
    actualCounter := GetCounter()

    if actualCounter != expectedCounter {
        t.Errorf("Expected counter value: %d, Actual counter value: %d", expectedCounter, actualCounter)
    }
}
```

Запуск теста:

```
go test
```

Этот тест создает 1000 горутин, каждая из которых вызывает функцию `Increment`. После завершения всех горутин, тест проверяет, что значение счётчика соответствует ожидаемому.