

Melhorando qualidade de código com Fuzzy Test

#IDTech

unico

Quem sou eu?

Tiago Temporin

SRE na Unico

 @_ttemporin

 AprendaGolang

 AprendaGolang

 AprendaGolang

www.aprendagolang.com.br



Go 1.18 - A grande release

Any

Novo tipo de dado que é um alias para `interface{}`.



Generics

A feature mais aguardada desde 2009.



Workspace

Diga adeus aos replaces no `go.mod`.



Fuzzy Test

Chega de testes somente com valores previstos.



O que é Fuzzy Test?

A técnica de fuzzing consiste em gerar inputs automaticamente com base em um padrão previamente informado.

Esses inputs serão utilizados para testar as funções que escrevemos, porém com algumas variações que podemos não estar preparados.

Diferença entre tests

Test comum

```
func TestReverse(t *testing.T) {  
    testcases := []string{"Tiago Temporin", " ", "!33241"}  
    for _, orig := range testcases {  
        rev, err1 := Reverse(orig)  
        if err1 != nil {  
            return  
        }  
  
        rev2, err2 := Reverse(rev)  
        if err2 != nil {  
            return  
        }  
  
        if orig != rev2 {  
            t.Errorf("esperado: %q, obtido: %q", orig, rev2)  
        }  
    }  
}
```

Diferença entre tests

Fuzzy Test

```
func FuzzReverse(f *testing.F) {
    testcases := []string{"Tiago Temporin", " ", "!33241"}
    for _, c := range testcases {
        f.Add(c)
    }

    f.Fuzz(func(t *testing.T, orig string) {
        rev, err1 := Reverse(orig)
        if err1 != nil {
            return
        }

        rev2, err2 := Reverse(rev)
        if err2 != nil {
            return
        }

        if orig != rev2 {
            t.Errorf("esperado: %q, obtido: %q", orig, rev2)
        }
    })
}
```

Fuzz Test

Preparação

Adicionar casos de uso
com o método

```
*testing.F.Add(args ...any)
```

Execução

```
*testing.F.Fuzz(ff any) onde  
ff deve ser uma função  
func(t *testing.T, args ...any)
```

args ...any?

[]byte, string, bool, byte,
rune, float32, float64, int,
int8, int16, int32, int64,
uint, uint8, uint16, uint32,
uint64.

Fuzz Func

Start

```
func (f *F) Fuzz(ff any) {
    if f.fuzzCalled {
        panic("testing: F.Fuzz called more than once")
    }
    f.fuzzCalled = true
    if f.failed {
        return
    }
    f.Helper()

    // ff should be in the form func(*testing.T, ...interface{})
    fn := reflect.ValueOf(ff)
    fnType := fn.Type()
    if fnType.Kind() != reflect.Func {
        panic("testing: F.Fuzz must receive a function")
    }
    if fnType.NumIn() < 2 || fnType.In(0) != reflect.TypeOf((*T)(nil)) {
        panic("testing: fuzz target must receive at least two arguments, where the first argument is a *T")
    }
    if fnType.NumOut() != 0 {
        panic("testing: fuzz target must not return a value")
    }
}
```


Fuzz Func

Type Validation

```
// supportedTypes represents all of the supported types which can be fuzzed.
var supportedTypes = map[reflect.Type]bool{
    reflect.TypeOf([]byte)(""): true,
    reflect.TypeOf(string)(""): true,
    reflect.TypeOf(bool)(false): true,
    reflect.TypeOf(byte)(0): true,
    reflect.TypeOf(rune)(0): true,
    reflect.TypeOf(float32)(0): true,
    reflect.TypeOf(float64)(0): true,
    reflect.TypeOf(int)(0): true,
    reflect.TypeOf(int8)(0): true,
    reflect.TypeOf(int16)(0): true,
    reflect.TypeOf(int32)(0): true,
    reflect.TypeOf(int64)(0): true,
    reflect.TypeOf(uint)(0): true,
    reflect.TypeOf(uint8)(0): true,
    reflect.TypeOf(uint16)(0): true,
    reflect.TypeOf(uint32)(0): true,
    reflect.TypeOf(uint64)(0): true,
}
```

```
// Save the types of the function to compare against the corpus.
var types []reflect.Type
for i := 1; i < fnType.NumIn(); i++ {
    t := fnType.In(i)
    if !supportedTypes[t] {
        panic(fmt.Sprintf("testing: unsupported type for fuzzing %v", t))
    }
    types = append(types, t)
}
```

Demo Time



```
→ fuzzy go test -v ./...
=== RUN    TestReverse
reverse_test.go:22: esperado: "Tiago Temporin", obtido: "Tiago Temporin"
reverse_test.go:22: esperado: " ", obtido: " "
reverse_test.go:22: esperado: "!33241", obtido: "!33241"
--- FAIL: TestReverse (0.00s)
=== RUN    FuzzReverse
=== RUN    FuzzReverse/seed#0
=== RUN    FuzzReverse/seed#1
=== RUN    FuzzReverse/seed#2
--- PASS: FuzzReverse (0.00s)
    --- PASS: FuzzReverse/seed#0 (0.00s)
    --- PASS: FuzzReverse/seed#1 (0.00s)
    --- PASS: FuzzReverse/seed#2 (0.00s)
FAIL
FAIL    github.com/aprendagolang/fuzzy 0.193s
FAIL
→ fuzzy
```




Obrigadx! :)

Tiago Temporin

tiago.temporin@unico.io

unico