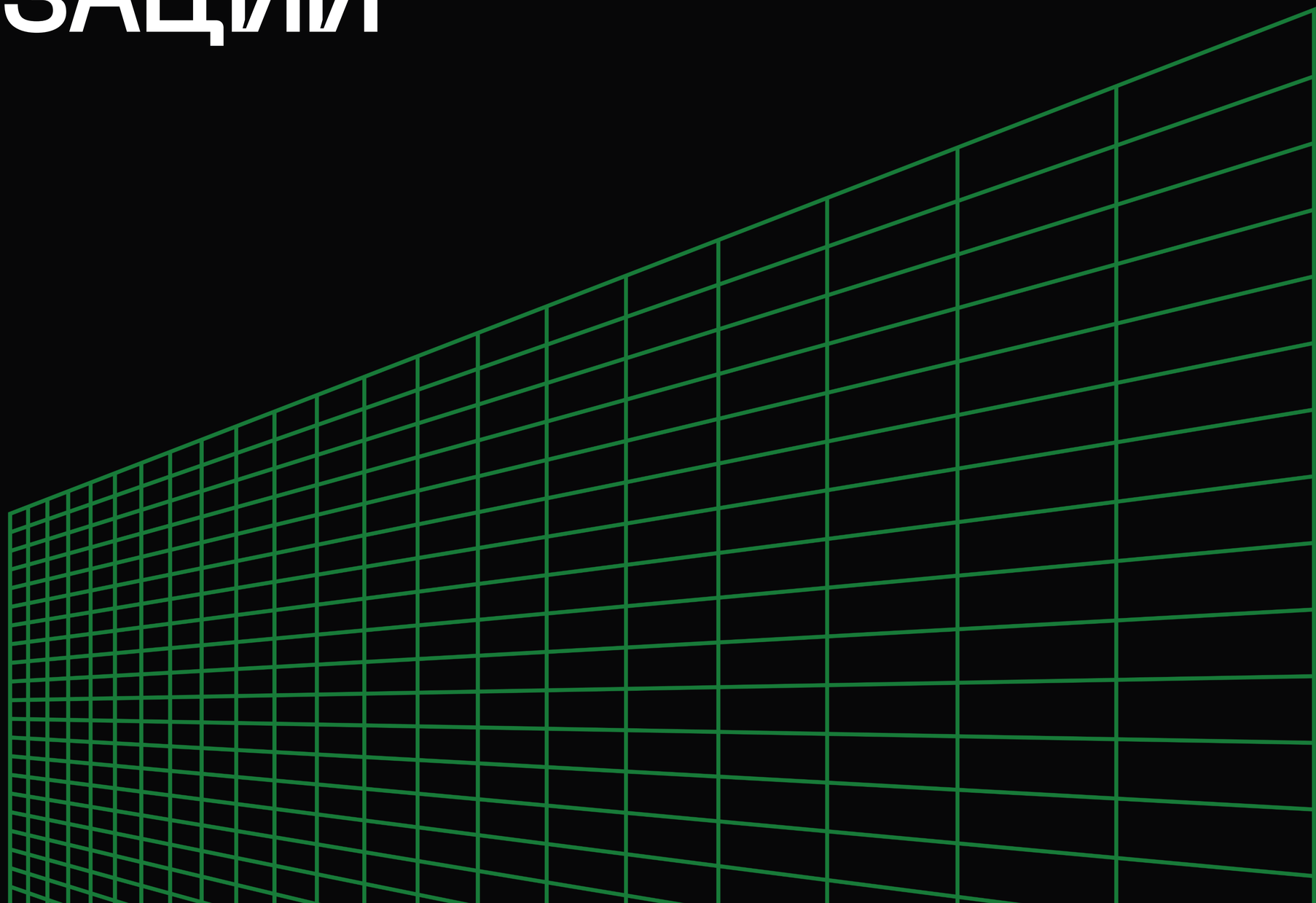
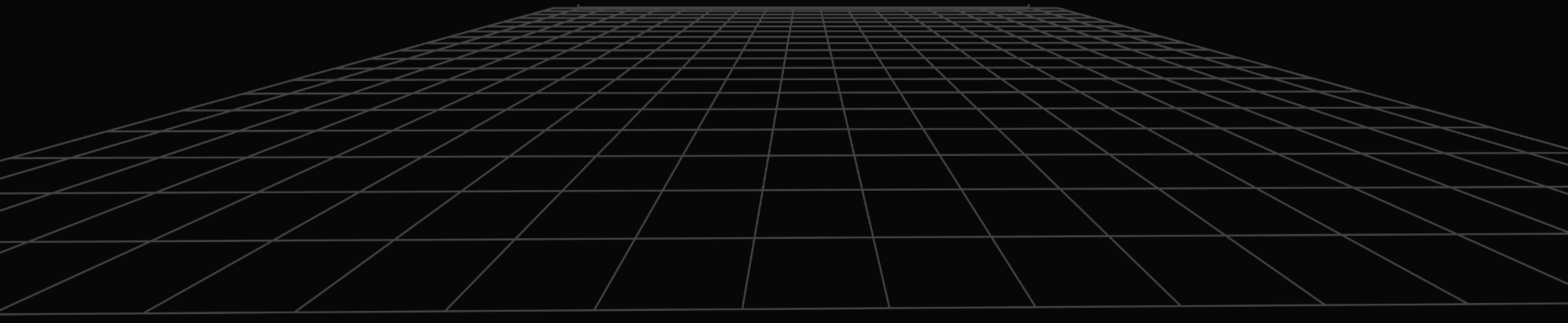
ОПТИМИЗАЦИИ В GO



Balun.Courses

ПРОВЕРЬ ЗАПИСЬ

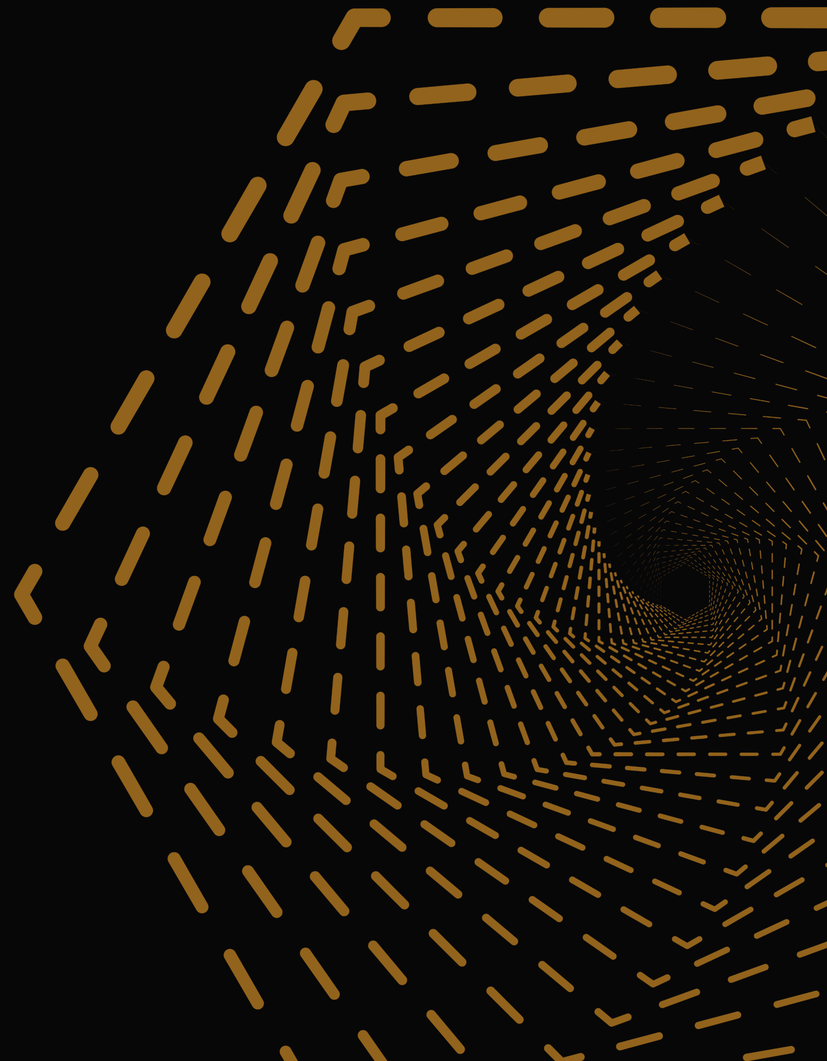


ПРАВИЛА ЗАНЯТИЯ

1. вопросы в чате можно задавать
   в любое время

2. вопросы голосом задаем
   по поднятой руке в Zoom

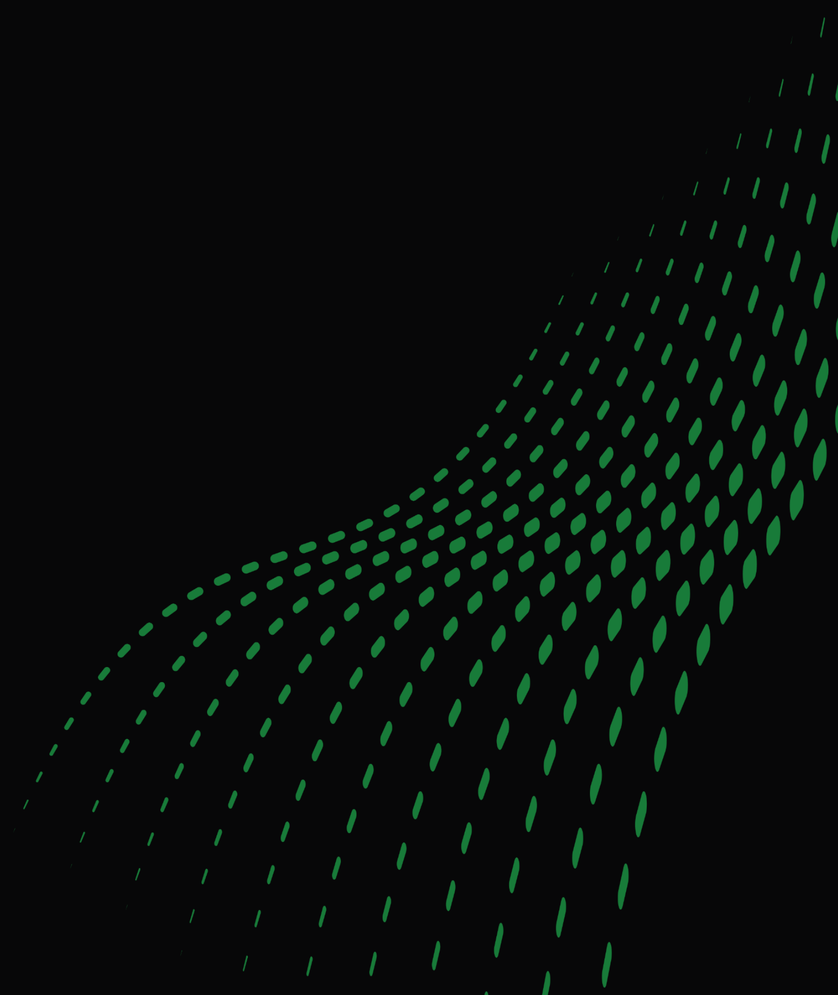
3. ответы на вопросы будут
   в запланированных местах



Balun.Courses

ПЛАН ЛЕКЦИИ

1. Управление рантаймом Go
2. Escape analysis
3. Bounds check elimination
4. Reflection



УПРАВЛЕНИЕ СБОРКОЙ МУСОРА

```go
// SetGCPercent returns the previous setting.
// The initial setting is the value of the GOGC environment variable
// at startup, or 100 if the variable is not set.
// This setting may be effectively reduced in order to maintain a memory
// limit.
// A negative percentage effectively disables garbage collection, unless
// the memory limit is reached.
// See SetMemoryLimit for more details.
func SetGCPercent(percent int) int {
    return int(setGCPercent(int32(percent)))
}
```

```go
// GC runs a garbage collection and blocks the caller until the
// garbage collection is complete. It may also block the entire
// program.
func GC() {
    // We consider a cycle to be: sweep termination, mark, mark
    // termination, and sweep. This function shouldn't return
    // until a full cycle has been completed, from beginning to
    // end. Hence, we always want to finish up the current cycle
    // and start a new one. That means:
    //
```

До Go 1.19 использовали балласты (например, twitch) или кастомные воркеры (например, Uber Go GC Tunner)

УПРАВЛЕНИЕ СБОРКОЙ МУСОРА

```
// The initial setting is math.MaxInt64 unless the GOMEMLIMIT
// environment variable is set, in which case it provides the initial
// setting. GOMEMLIMIT is a numeric value in bytes with an optional
// unit suffix. The supported suffixes include B, KiB, MiB, GiB, and
// TiB. These suffixes represent quantities of bytes as defined by
// the IEC 80000-13 standard. That is, they are based on powers of
// two: KiB means 2^10 bytes, MiB means 2^20 bytes, and so on.
//
// SetMemoryLimit returns the previously set memory limit.
// A negative input does not adjust the limit, and allows for
// retrieval of the currently set memory limit.
func SetMemoryLimit(limit int64) int64 {
    return setMemoryLimit(limit)
}
```

Go 1.19

ПАКЕТ RUNTIME

```
runtime.NumCPU()
runtime.Gosched()
runtime.NumGoroutine()
runtime.FuncForPC( pc: 0)
runtime.GOMAXPROCS( n: 0)
// + profiling
```

Также есть функции для профилирования и работы с внешними указателями

ПАКЕТ RUNTIME

```
// For example, consider a finalizer that inspects a mutable field in x
// that is modified from time to time in the main program before x
// becomes unreachable and the finalizer is invoked.
// The modifications in the main program and the inspection in the finalizer
// need to use appropriate synchronization, such as mutexes or atomic updates,
// to avoid read-write races.
func SetFinalizer(obj any, finalizer any) {
```

```
        if err := syscall.SetNonblock(fd, nonblocking: false); err ==
    }


    runtime.SetFinalizer(f.file, (*file).close)
    return f
}
```

ESCAPE ANALYSIS

```
b.Run( name: "indirection assignment", func(b *testing.B) {
    b.ReportAllocs()

    for i := 0; i < b.N; i++ {
        var number int
        data := &Data{}
        data.pointer = &number
        _ = data
    }
})
```


```
b.Run( name: "indirection assignment", func(b *testing.B) {
    b.ReportAllocs()

    for i := 0; i < b.N; i++ {
        var number int
        data := &Data{}
        data.pointer = &number
        _ = data
    }
})
```

BOUNDS CHECK ELIMINATION

```
_ = out[31] // bounds check elimination hint
binary.LittleEndian.PutUint32(out[0:4], x0)
binary.LittleEndian.PutUint32(out[4:8], x1)
binary.LittleEndian.PutUint32(out[8:12], x2)
binary.LittleEndian.PutUint32(out[12:16], x3)
binary.LittleEndian.PutUint32(out[16:20], x12)
binary.LittleEndian.PutUint32(out[20:24], x13)
binary.LittleEndian.PutUint32(out[24:28], x14)
binary.LittleEndian.PutUint32(out[28:32], x15)
```

REFLECTION

Unmarshaling

lib	json size	MB/s	allocs/op	B/op
standard	regular	22	218	10229
standard	small	9.7	14	720
easyjson	regular	125	128	9794
easyjson	small	67	3	128

Почему стандартная сериализация медленнее?

REFLECTION

```
BenchmarkSerialization
BenchmarkSerialization/convertThroughSerialization
BenchmarkSerialization/convertThroughSerialization-12              2318         531515 ns/op      265098 B/op       3937 allocs/op
BenchmarkSerialization/convertThroughSerializationNew
BenchmarkSerialization/convertThroughSerializationNew-12           4245         274356 ns/op      282184 B/op       4389 allocs/op
BenchmarkSerialization/easyjsonSerialization
BenchmarkSerialization/easyjsonSerialization-12                   20594          57407 ns/op       53259 B/op        499 allocs/op
PASS
```

Почему стандартная сериализация медленнее?

ПАКЕТ REFLECT

```
func main() {
    var x float64 = 3.4
    fmt.Println( a...: "type:", reflect.TypeOf(x))
}
```

ПАКЕТ REFLECT

```go
func main() {
    var x float64 = 3.4
    v := reflect.ValueOf(x)
    fmt.Println( a...: "type:", v.Type())
    fmt.Println( a...: "kind is float64:", v.Kind() == reflect.Float64)
    fmt.Println( a...: "value:", v.Float())
}
```

```
/Users/igorwalther/Library/Caches/JetBrains/GoLand2024.1/tmp/GoLand/___go_build_lection7_typeof
type: float64
kind is float64: true
value: 3.4
```

ПАКЕТ REFLECT

```go
func main() {
    var x uint8 = 'x'
    v := reflect.ValueOf(x)
    fmt.Println( a...: "type:", v.Type())                    // uint8.
    fmt.Println( a...: "kind is uint8: ", v.Kind() == reflect.Uint8) // true.
    x = uint8(v.Uint())                                      // v.Uint returns a uint64.
}
```

```
/Users/igorwalther/Library/Caches/JetBrains/GoLand2024.1/tmp/GoLand/___go_build_lection7_typeof
type: uint8
kind is uint8:  true
```

ПАКЕТ REFLECT

```go
func main() {
    var x float64 = 3.4
    p := reflect.ValueOf(&x)
    fmt.Println( a...: "type of p:", p.Type())
    fmt.Println( a...: "settability of p:", p.CanSet())
    fmt.Println( a...: "settability of p:", p.Elem().CanSet())
}
```

```
/Users/igorwalther/Library/Caches/JetBrains/GoLand2024.1/tmp/GoLand/___go_build_lection7_typeof
type of p: *float64
settability of p: false
settability of p: true
```

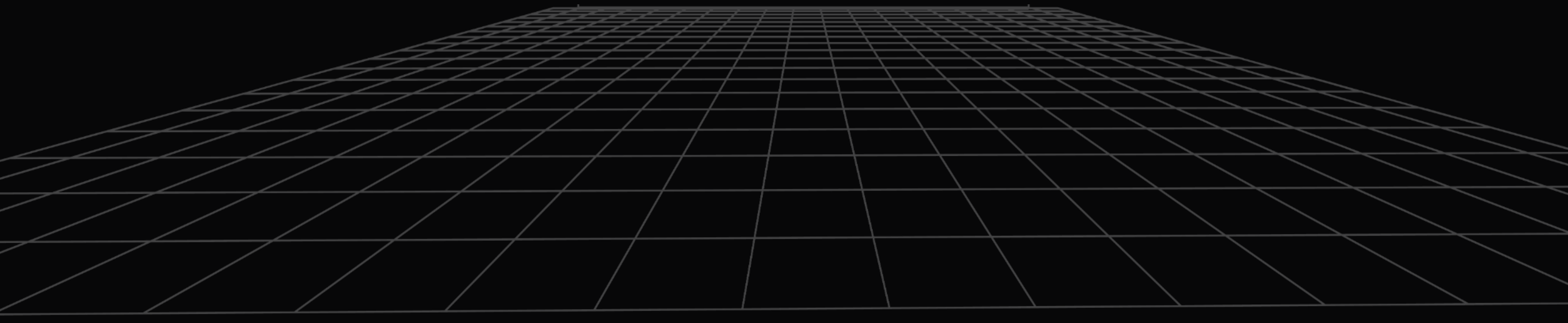
ПАКЕТ REFLECT

```go
type T struct {
    A int
    B string
}

t := T{A: 23, B: "go course"}
s := reflect.ValueOf(&t).Elem()
```

```go
typeOfT := s.Type()
for i := 0; i < s.NumField(); i++ {
    f := s.Field(i)
    fmt.Printf(format: "%d: %s %s = %v\n", i,
        typeOfT.Field(i).Name, f.Type(), f.Interface())
}

s.Field(i: 0).SetInt(x: 77)
s.Field(i: 1).SetString(x: "Sunset Strip")
fmt.Println(a...: "t is now", t)
```

```
/Users/igorwalther/Library/Caches/JetBrains/GoLand2024.1/tmp/GoLand/___go_build_lection7_typeof
0: A int = 23
1: B string = go course
t is now {77 Sunset Strip}
```

DEMO



ИТОГИ И ВЫВОДЫ

ЧТО БУДЕТ ДАЛЬШЕ