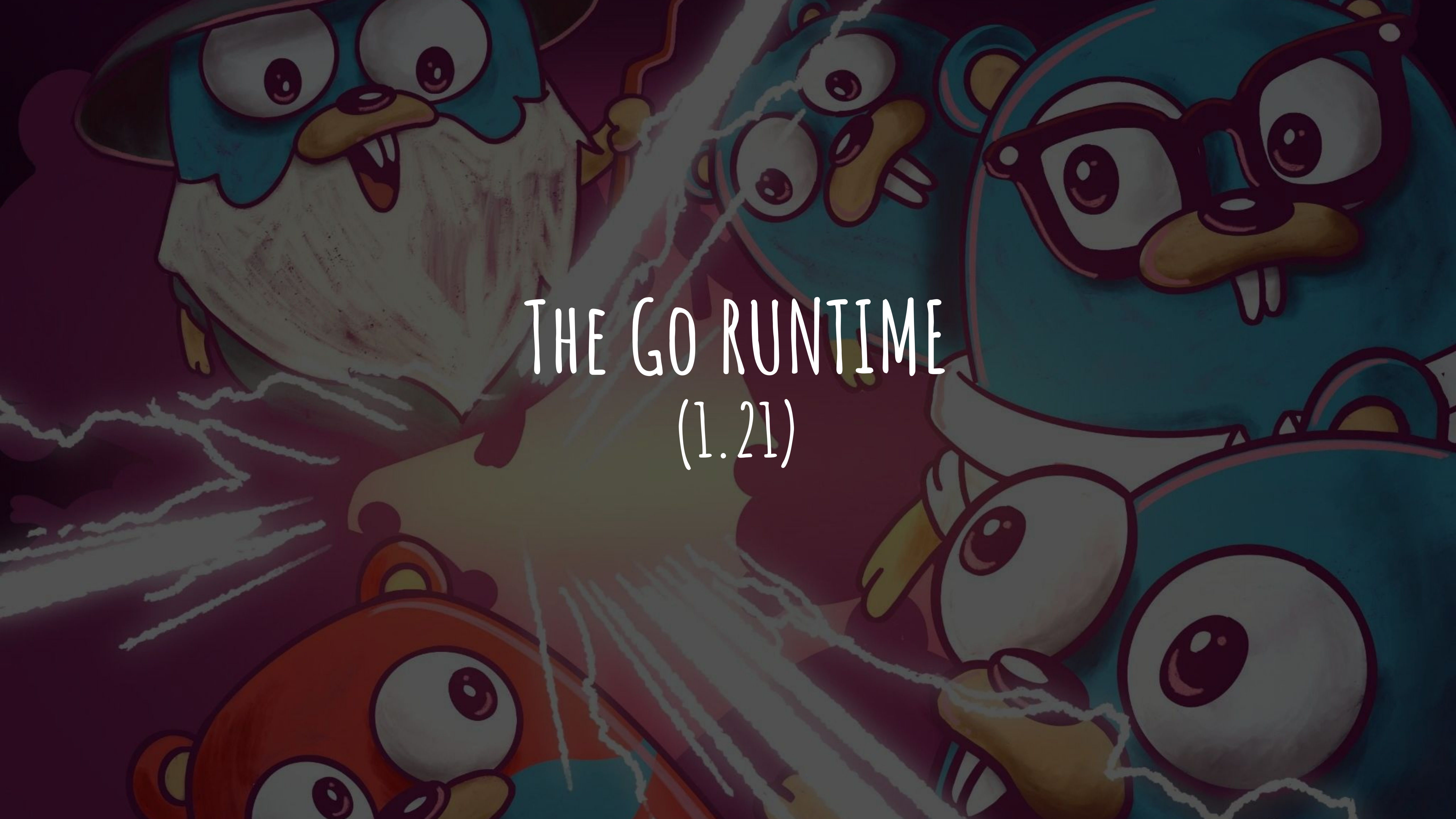
UNDERSTANDING THE GO RUNTIME

JESÚS ESPINO - SOFTWARE ENGINEER @ MATTERMOST

Introduction

Disclaimer



The Go runtime

(1.21)

The Compiler

The Go Compiler

The Go Compiler

How fast is Go?

vs

How fast is Go?

```go
package main
func main() {}
```

VS

```c
int main() {}
```

How fast is Go?

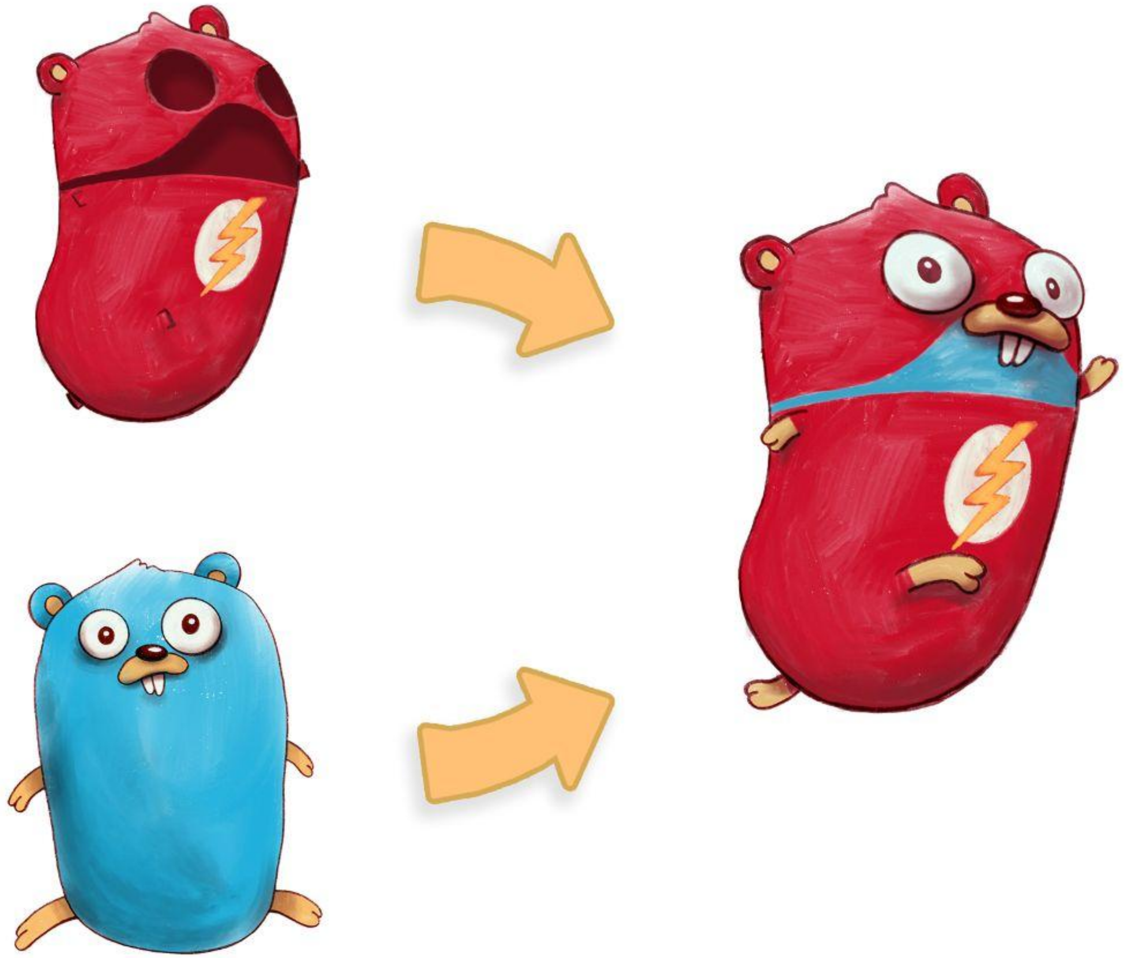
1.2 MBs ~1 second/1000 executions	VS	16 KBs ~0.3 second/1000 executions

How fast is Go?

75 times binary size
3 times execution time

Compiler ❤️ Runtime

The compiler and the go runtime tandem



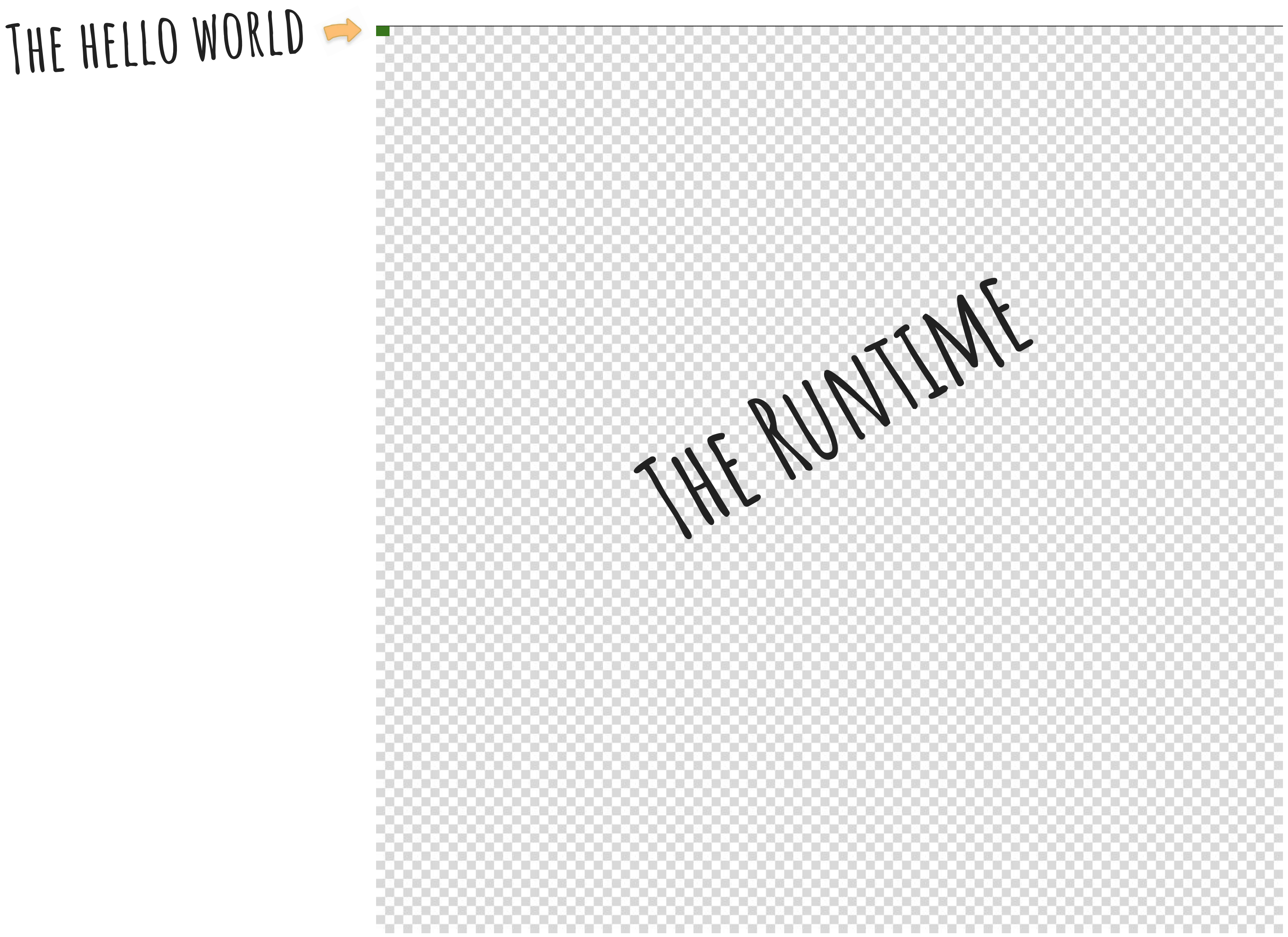
Hello world example

```go
package main


import "fmt"


func main() {
    fmt.Println("hello world!")
}
```

Our compiled hello world



Hello World Assembly

```
0x0000 00000 (.../main.go:5)      TEXT     main.main(SB), ABIInternal, $64-0
0x0000 00000 (.../main.go:5)      CMPQ     SP, 16(R14)
0x0004 00004 (.../main.go:5)      PCDATA   $0, $-2
0x0004 00004 (.../main.go:5)      JLS      82
0x0006 00006 (.../main.go:5)      PCDATA   $0, $-1
0x0006 00006 (.../main.go:5)      PUSHQ    BP
0x0007 00007 (.../main.go:5)      MOVQ     SP, BP
0x000a 00010 (.../main.go:5)      SUBQ     $56, SP
0x000e 00014 (.../main.go:5)      FUNCDATA $0, gclocals·g2BeySu+wFnoycgXfElmcg==(SB)
0x000e 00014 (.../main.go:5)      FUNCDATA $1, gclocals·EaPwxsZ75yY1hHMVZLmk6g==(SB)
0x000e 00014 (.../main.go:5)      FUNCDATA $2, main.main.stkobj(SB)
0x000e 00014 (.../main.go:6)      MOVUPS   X15, main..autotmp_8+40(SP)
0x0014 00020 (.../main.go:6)      LEAQ     type:string(SB), DX
0x001b 00027 (.../main.go:6)      MOVQ     DX, main..autotmp_8+40(SP)
0x0020 00032 (.../main.go:6)      LEAQ     main..stmp_0(SB), DX
0x0027 00039 (.../main.go:6)      MOVQ     DX, main..autotmp_8+48(SP)
0x002c 00044 (/usr/local/go/src/fmt/print.go:314) MOVQ     os.Stdout(SB), BX
0x0033 00051 (<unknown line number>)      NOP
0x0033 00051 (/usr/local/go/src/fmt/print.go:314) LEAQ     go:itab.*os.File,io.Writer(SB), AX
0x003a 00058 (/usr/local/go/src/fmt/print.go:314) LEAQ     main..autotmp_8+40(SP), CX
0x003f 00063 (/usr/local/go/src/fmt/print.go:314) MOVL     $1, DI
0x0044 00068 (/usr/local/go/src/fmt/print.go:314) MOVQ     DI, SI
0x0047 00071 (/usr/local/go/src/fmt/print.go:314) PCDATA   $1, $0
0x0047 00071 (/usr/local/go/src/fmt/print.go:314) CALL     fmt.Fprintln(SB)
0x004c 00076 (.../main.go:7)      ADDQ     $56, SP
0x0050 00080 (.../main.go:7)      POPQ     BP
0x0051 00081 (.../main.go:7)      RET
0x0052 00082 (.../main.go:7)      NOP
0x0052 00082 (.../main.go:5)      PCDATA   $1, $-1
0x0052 00082 (.../main.go:5)      PCDATA   $0, $-2
0x0052 00082 (.../main.go:5)      CALL     runtime.morestack_noctxt(SB)
0x0057 00087 (.../main.go:5)      PCDATA   $0, $-1
0x0057 00087 (.../main.go:5)      JMP      0
```

Generated with: `go build -gcflags=-S .`

Hello World Assembly

```
0x0000 00000 (.../main.go:5)      TEXT     main.main(SB), ABIInternal, $64-0
0x0000 00000 (.../main.go:5)      CMPQ     SP, 16(R14)
0x0004 00004 (.../main.go:5)      PCDATA   $0, $-2
0x0004 00004 (.../main.go:5)      JLS      82
0x0006 00006 (.../main.go:5)      PCDATA   $0, $-1
0x0006 00006 (.../main.go:5)      PUSHQ    BP
0x0007 00007 (.../main.go:5)      MOVQ     SP, BP
0x000a 00010 (.../main.go:5)      SUBQ     $56, SP
0x000e 00014 (.../main.go:5)      FUNCDATA $0, gclocals·g2BeySu+wFnoycgXfElmcg==(SB)
0x000e 00014 (.../main.go:5)      FUNCDATA $1, gclocals·EaPwxsZ75yY1hHMVZLmk6g==(SB)
0x000e 00014 (.../main.go:5)      FUNCDATA $2, main.main.stkobj(SB)
0x000e 00014 (.../main.go:6)      MOVUPS   X15, main..autotmp_8+40(SP)
0x0014 00020 (.../main.go:6)      LEAQ     type:string(SB), DX
0x001b 00027 (.../main.go:6)      MOVQ     DX, main..autotmp_8+40(SP)
0x0020 00032 (.../main.go:6)      LEAQ     main..stmp_0(SB), DX
0x0027 00039 (.../main.go:6)      MOVQ     DX, main..autotmp_8+48(SP)
0x002c 00044 (/usr/local/go/src/fmt/print.go:314) MOVQ     os.Stdout(SB), BX
0x0033 00051 (<unknown line number>)      NOP
0x0033 00051 (/usr/local/go/src/fmt/print.go:314) LEAQ     go:itab.*os.File,io.Writer(SB), AX
0x003a 00058 (/usr/local/go/src/fmt/print.go:314) LEAQ     main..autotmp_8+40(SP), CX
0x003f 00063 (/usr/local/go/src/fmt/print.go:314) MOVL     $1, DI
0x0044 00068 (/usr/local/go/src/fmt/print.go:314) MOVQ     DI, SI
0x0047 00071 (/usr/local/go/src/fmt/print.go:314) PCDATA   $1, $0
0x0047 00071 (/usr/local/go/src/fmt/print.go:314) CALL     fmt.Fprintln(SB)
0x004c 00076 (.../main.go:7)      ADDQ     $56, SP
0x0050 00080 (.../main.go:7)      POPQ     BP
0x0051 00081 (.../main.go:7)      RET
0x0052 00082 (.../main.go:7)      NOP
0x0052 00082 (.../main.go:5)      PCDATA   $1, $-1
0x0052 00082 (.../main.go:5)      PCDATA   $0, $-2
0x0052 00082 (.../main.go:5)      CALL     runtime.morestack_noctxt(SB)
0x0057 00087 (.../main.go:5)      PCDATA   $0, $-1
0x0057 00087 (.../main.go:5)      JMP      0
```

Generated with: `go build -gcflags=-S .`

Slices, Maps And Channels

```go
package main

func main() {
    sampleSlice := []int{1, 2, 3, 4, 5}
    sampleSlice = append(sampleSlice, 6)

    sampleMap := map[string]int{"a": 1, "b": 2}
    sampleMap["c"] = 3

    sampleChannel := make(chan int)

    for _, value := range sampleSlice {
        sampleChannel <- value
    }
    for _, value := range sampleMap {
        sampleChannel <- value
    }
}
```

Slices, Maps And Channels Assembly

```
0x0077 00119 (.../main.go:5)  CALL runtime.growslice(SB)
0x00e3 00227 (.../main.go:7)  CALL runtime.fastrand(SB)
0x010a 00266 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0131 00305 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0158 00344 (.../main.go:8)  CALL runtime.mapassign_faststr(SB)
0x016d 00365 (.../main.go:10) CALL runtime.makechan(SB)
0x0199 00409 (.../main.go:13) CALL runtime.chansend1(SB)
0x01ea 00490 (.../main.go:16) CALL runtime.mapiterinit(SB)
0x020e 00526 (.../main.go:17) CALL runtime.chansend1(SB)
0x0220 00544 (.../main.go:16) CALL runtime.mapiternext(SB)
0x0239 00569 (.../main.go:3)  CALL runtime.morestack_noctxt(SB)
```

Generated with: `go build -gcflags=-S .`

Slices, Maps And Channels

```go
package main

func main() {
    sampleSlice := []int{1, 2, 3, 4, 5}
    sampleSlice = append(sampleSlice, 6)

    sampleMap := map[string]int{"a": 1, "b": 2}
    sampleMap["c"] = 3

    sampleChannel := make(chan int)

    for _, value := range sampleSlice {
        sampleChannel <- value
    }
    for _, value := range sampleMap {
        sampleChannel <- value
    }
}
```

Slices, Maps And Channels Assembly

```
0x0077 00119 (.../main.go:5)  CALL runtime.growslice(SB)
0x00e3 00227 (.../main.go:7)  CALL runtime.fastrand(SB)
0x010a 00266 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0131 00305 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0158 00344 (.../main.go:8)  CALL runtime.mapassign_faststr(SB)
0x016d 00365 (.../main.go:10) CALL runtime.makechan(SB)
0x0199 00409 (.../main.go:13) CALL runtime.chansend1(SB)
0x01ea 00490 (.../main.go:16) CALL runtime.mapiterinit(SB)
0x020e 00526 (.../main.go:17) CALL runtime.chansend1(SB)
0x0220 00544 (.../main.go:16) CALL runtime.mapiternext(SB)
0x0239 00569 (.../main.go:3)  CALL runtime.morestack_noctxt(SB)
```

Generated with: go build -gcflags=-S .

Slices, Maps And Channels

```go
package main

func main() {
    sampleSlice := []int{1, 2, 3, 4, 5}
    sampleSlice = append(sampleSlice, 6)

    sampleMap := map[string]int{"a": 1, "b": 2}
    sampleMap["c"] = 3

    sampleChannel := make(chan int)

    for _, value := range sampleSlice {
        sampleChannel <- value
    }
    for _, value := range sampleMap {
        sampleChannel <- value
    }
}
```

Slices, Maps And Channels Assembly

```
0x0077 00119 (.../main.go:5)  CALL runtime.growslice(SB)
0x00e3 00227 (.../main.go:7)  CALL runtime.fastrand(SB)
0x010a 00266 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0131 00305 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0158 00344 (.../main.go:8)  CALL runtime.mapassign_faststr(SB)
0x016d 00365 (.../main.go:10) CALL runtime.makechan(SB)
0x0199 00409 (.../main.go:13) CALL runtime.chansend1(SB)
0x01ea 00490 (.../main.go:16) CALL runtime.mapiterinit(SB)
0x020e 00526 (.../main.go:17) CALL runtime.chansend1(SB)
0x0220 00544 (.../main.go:16) CALL runtime.mapiternext(SB)
0x0239 00569 (.../main.go:3)  CALL runtime.morestack_noctxt(SB)
```

Generated with: `go build -gcflags=-S .`

Slices, Maps And Channels

```go
package main

func main() {
    sampleSlice := []int{1, 2, 3, 4, 5}
    sampleSlice = append(sampleSlice, 6)

    sampleMap := map[string]int{"a": 1, "b": 2}
    sampleMap["c"] = 3

    sampleChannel := make(chan int)

    for _, value := range sampleSlice {
        sampleChannel <- value
    }
    for _, value := range sampleMap {
        sampleChannel <- value
    }
}
```

Slices, Maps And Channels Assembly

```
0x0077 00119 (.../main.go:5)  CALL runtime.growslice(SB)
0x00e3 00227 (.../main.go:7)  CALL runtime.fastrand(SB)
0x010a 00266 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0131 00305 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0158 00344 (.../main.go:8)  CALL runtime.mapassign_faststr(SB)
0x016d 00365 (.../main.go:10) CALL runtime.makechan(SB)
0x0199 00409 (.../main.go:13) CALL runtime.chansend1(SB)
0x01ea 00490 (.../main.go:16) CALL runtime.mapiterinit(SB)
0x020e 00526 (.../main.go:17) CALL runtime.chansend1(SB)
0x0220 00544 (.../main.go:16) CALL runtime.mapiternext(SB)
0x0239 00569 (.../main.go:3)  CALL runtime.morestack_noctxt(SB)
```

Generated with: `go build -gcflags=-S .`

Slices, Maps And Channels

```go
package main

func main() {
    sampleSlice := []int{1, 2, 3, 4, 5}
    sampleSlice = append(sampleSlice, 6)

    sampleMap := map[string]int{"a": 1, "b": 2}
    sampleMap["c"] = 3

    sampleChannel := make(chan int)

    for _, value := range sampleSlice {
        sampleChannel <- value
    }
    for _, value := range sampleMap {
        sampleChannel <- value
    }
}
```

Slices, Maps And Channels Assembly

```
0x0077 00119 (.../main.go:5)  CALL runtime.growslice(SB)
0x00e3 00227 (.../main.go:7)  CALL runtime.fastrand(SB)
0x010a 00266 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0131 00305 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0158 00344 (.../main.go:8)  CALL runtime.mapassign_faststr(SB)
0x016d 00365 (.../main.go:10) CALL runtime.makechan(SB)
0x0199 00409 (.../main.go:13) CALL runtime.chansend1(SB)
0x01ea 00490 (.../main.go:16) CALL runtime.mapiterinit(SB)
0x020e 00526 (.../main.go:17) CALL runtime.chansend1(SB)
0x0220 00544 (.../main.go:16) CALL runtime.mapiternext(SB)
0x0239 00569 (.../main.go:3)  CALL runtime.morestack_noctxt(SB)
```

Generated with: go build -gcflags=-S .

Slices, Maps And Channels

```go
package main

func main() {
    sampleSlice := []int{1, 2, 3, 4, 5}
    sampleSlice = append(sampleSlice, 6)

    sampleMap := map[string]int{"a": 1, "b": 2}
    sampleMap["c"] = 3

    sampleChannel := make(chan int)

    for _, value := range sampleSlice {
        sampleChannel <- value
    }
    for _, value := range sampleMap {
        sampleChannel <- value
    }
}
```

Slices, Maps And Channels Assembly

```
0x0077 00119 (.../main.go:5)  CALL runtime.growslice(SB)
0x00e3 00227 (.../main.go:7)  CALL runtime.fastrand(SB)
0x010a 00266 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0131 00305 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0158 00344 (.../main.go:8)  CALL runtime.mapassign_faststr(SB)
0x016d 00365 (.../main.go:10) CALL runtime.makechan(SB)
0x0199 00409 (.../main.go:13) CALL runtime.chansend1(SB)
0x01ea 00490 (.../main.go:16) CALL runtime.mapiterinit(SB)
0x020e 00526 (.../main.go:17) CALL runtime.chansend1(SB)
0x0220 00544 (.../main.go:16) CALL runtime.mapiternext(SB)
0x0239 00569 (.../main.go:3)  CALL runtime.morestack_noctxt(SB)
```

Generated with: go build -gcflags=-S .

Slices, Maps And Channels

```go
package main

func main() {
    sampleSlice := []int{1, 2, 3, 4, 5}
    sampleSlice = append(sampleSlice, 6)

    sampleMap := map[string]int{"a": 1, "b": 2}
    sampleMap["c"] = 3

    sampleChannel := make(chan int)

    for _, value := range sampleSlice {
        sampleChannel <- value
    }
    for _, value := range sampleMap {
        sampleChannel <- value
    }
}
```

Slices, Maps And Channels Assembly

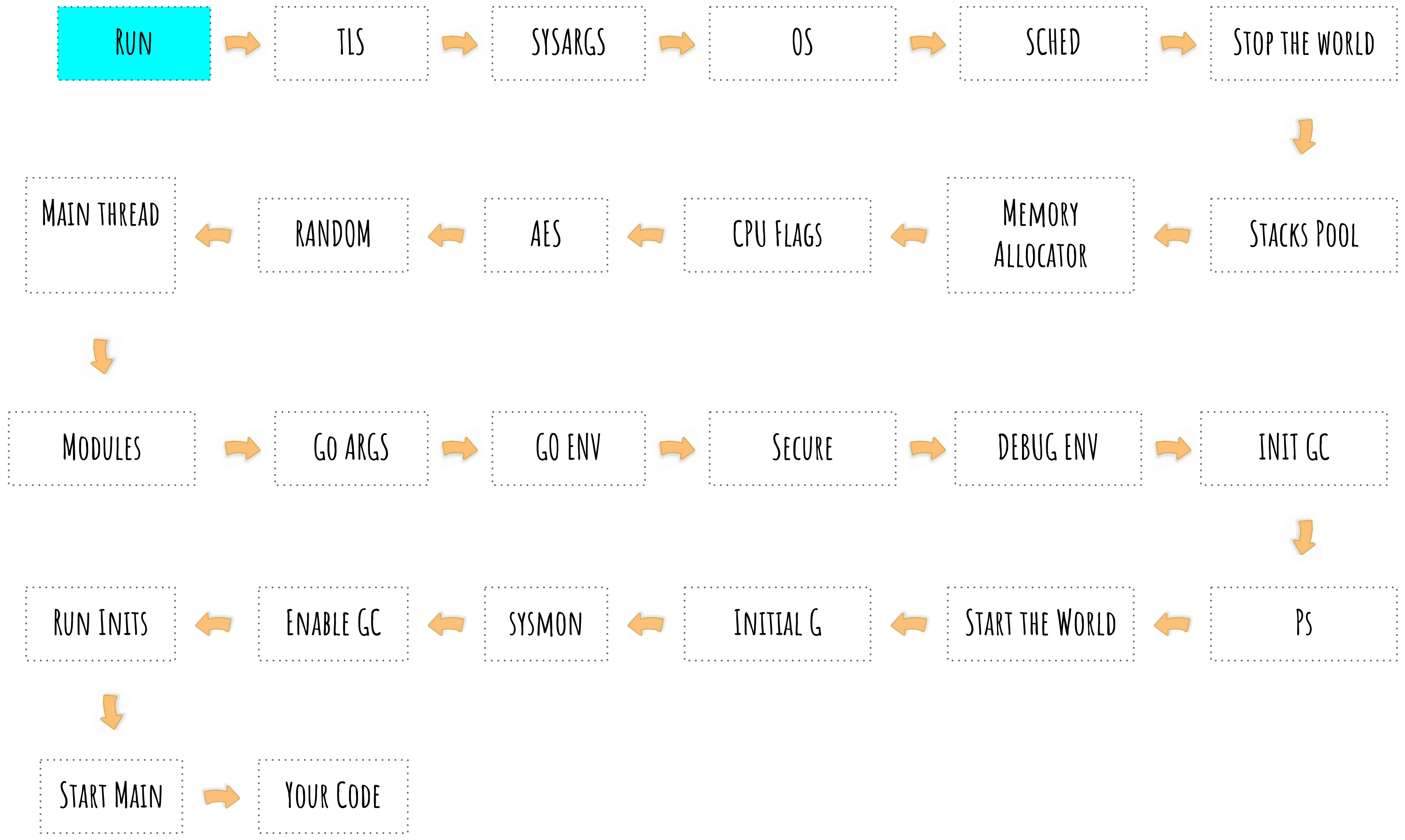
```
0x0077 00119 (.../main.go:5)  CALL runtime.growslice(SB)
0x00e3 00227 (.../main.go:7)  CALL runtime.fastrand(SB)
0x010a 00266 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0131 00305 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0158 00344 (.../main.go:8)  CALL runtime.mapassign_faststr(SB)
0x016d 00365 (.../main.go:10) CALL runtime.makechan(SB)
0x0199 00409 (.../main.go:13) CALL runtime.chansend1(SB)
0x01ea 00490 (.../main.go:16) CALL runtime.mapiterinit(SB)
0x020e 00526 (.../main.go:17) CALL runtime.chansend1(SB)
0x0220 00544 (.../main.go:16) CALL runtime.mapiternext(SB)
0x0239 00569 (.../main.go:3)  CALL runtime.morestack_noctxt(SB)
```

Generated with: go build -gcflags=-S .

The Runtime

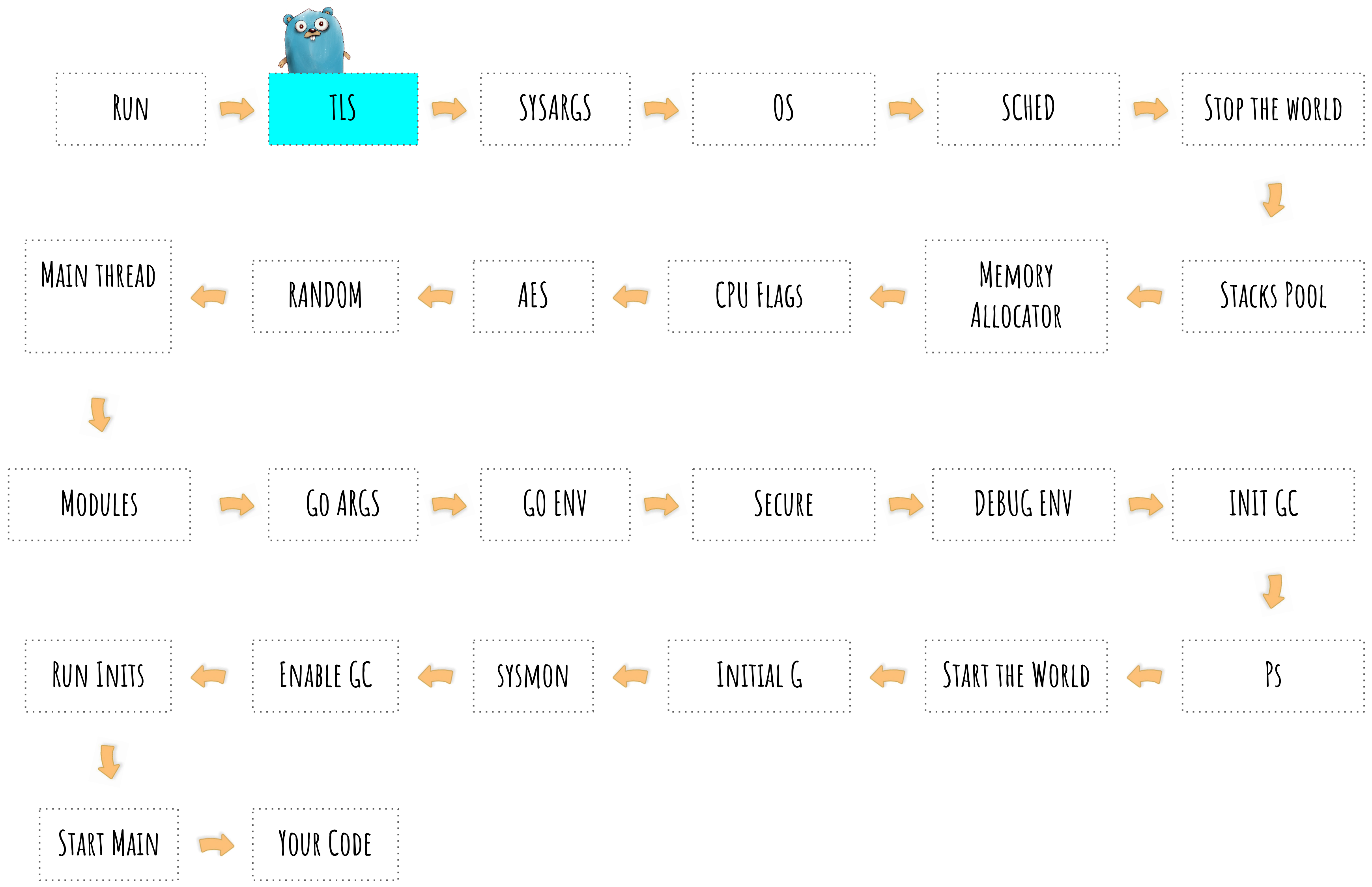
The runtime

The bootstrap process



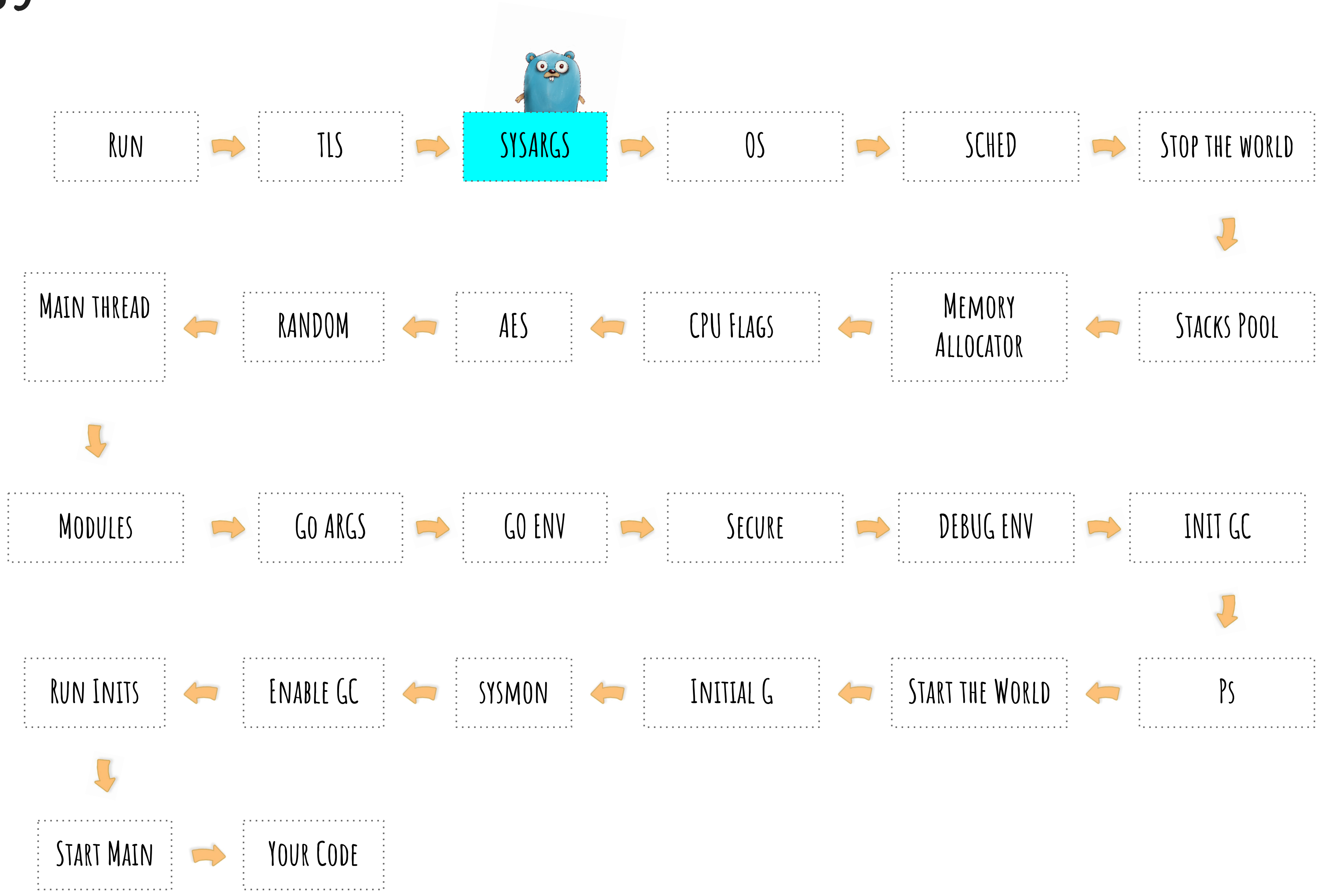
runtime/rt0_linux_amd64.s:7

The Thread Local Storage



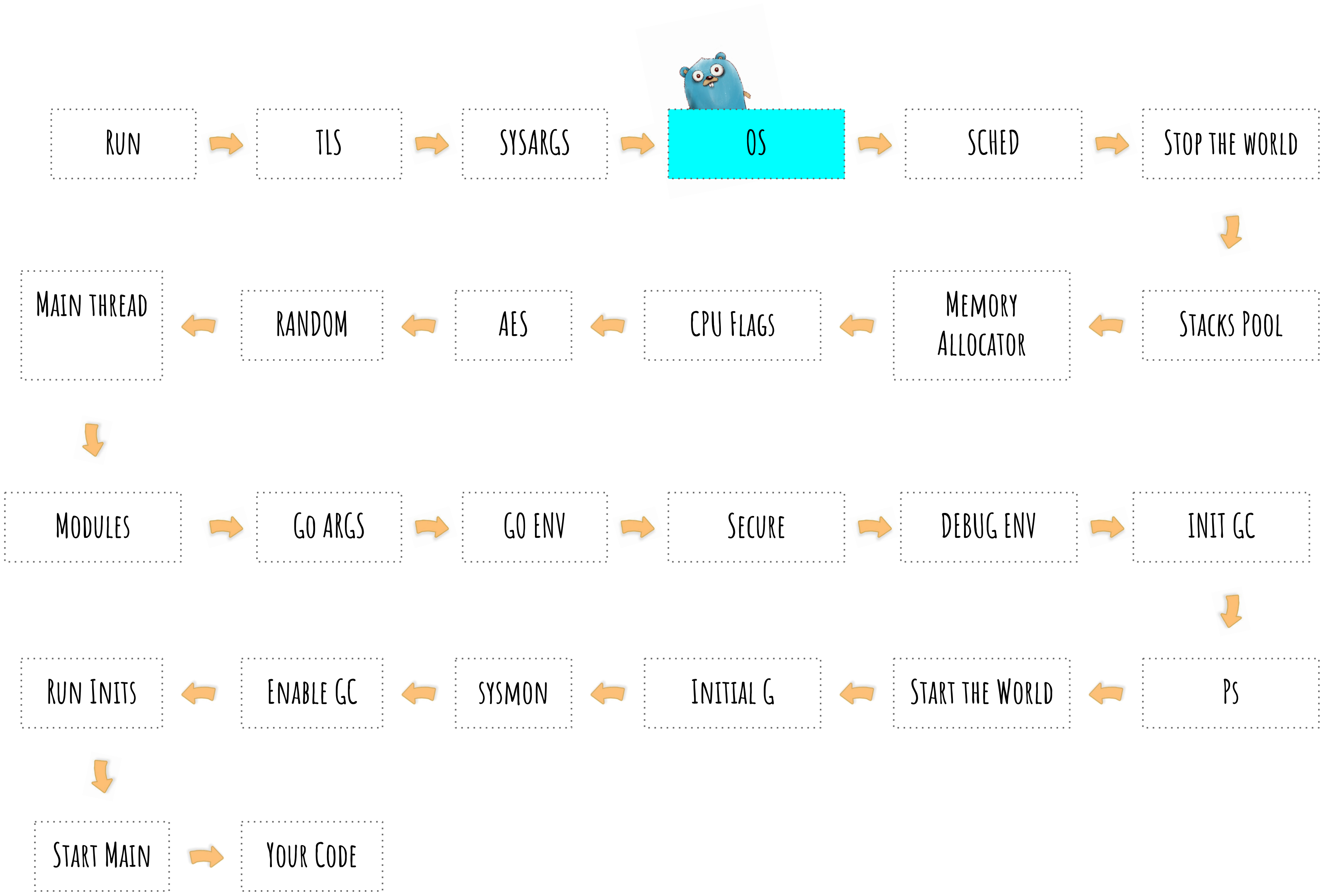
runtime/sys_linux_amd64.s:637

Sysargs



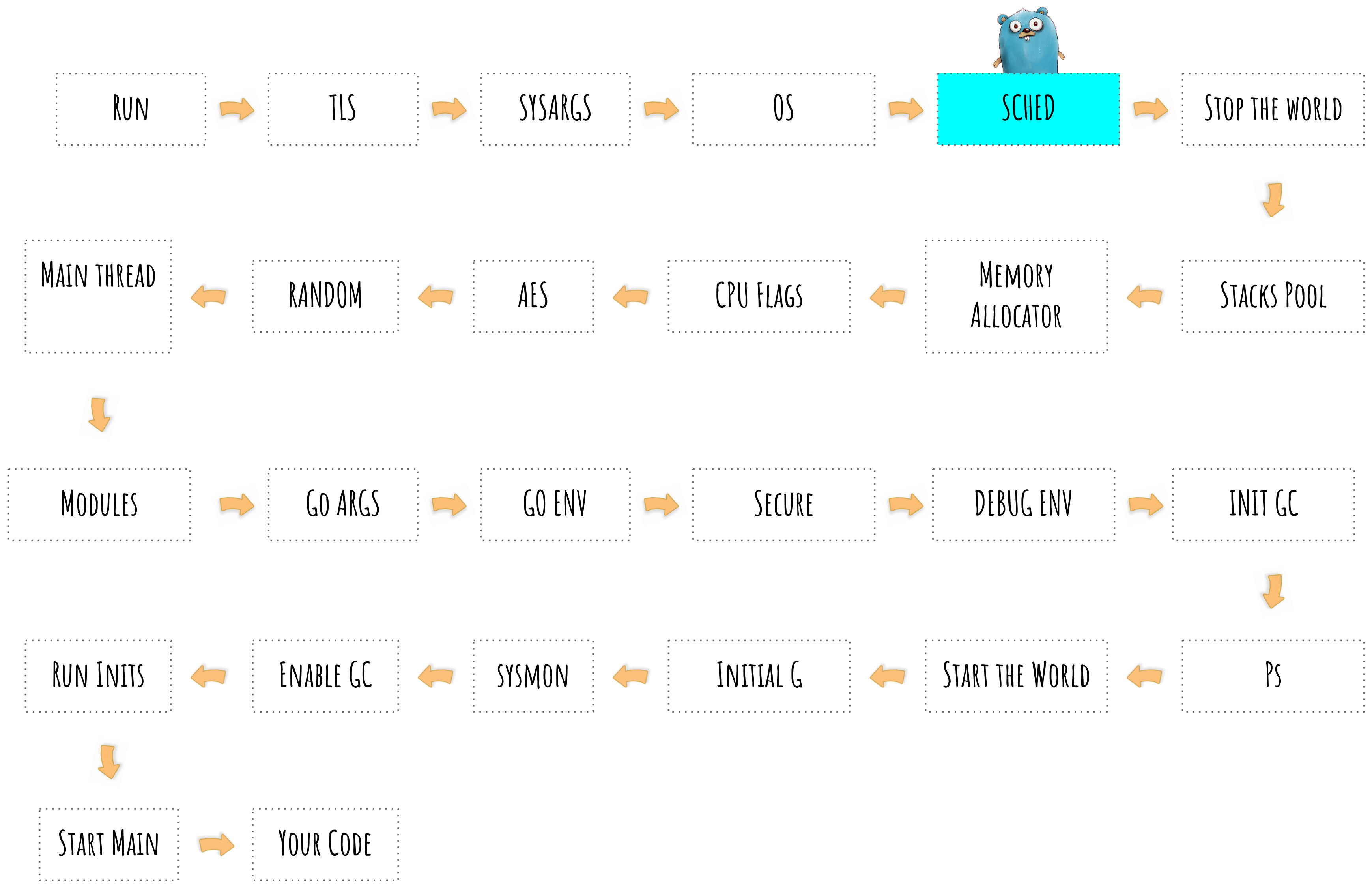
runtime/os_linux.go:232

OS



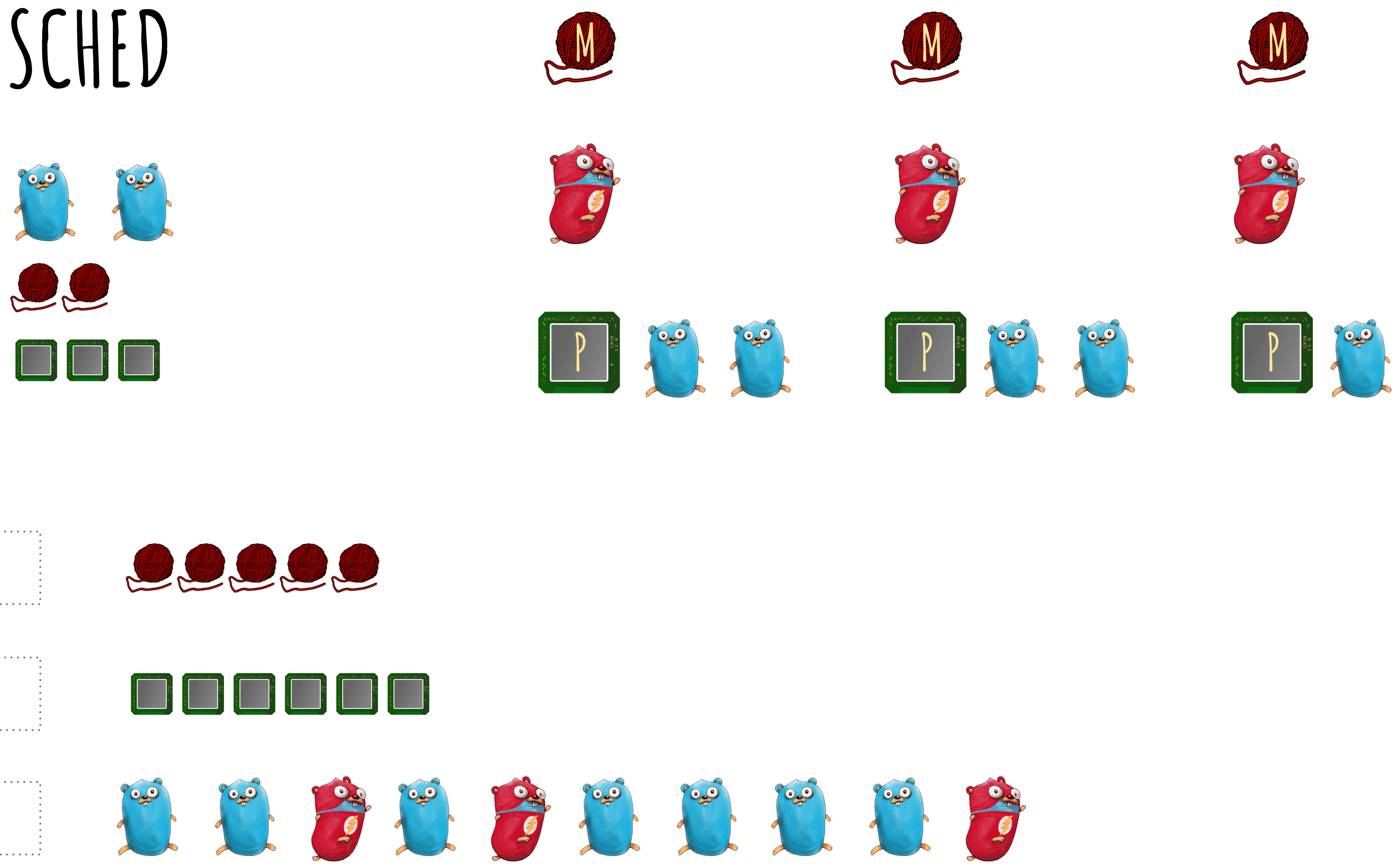
runtime/os_linux.go:346

The Scheduler

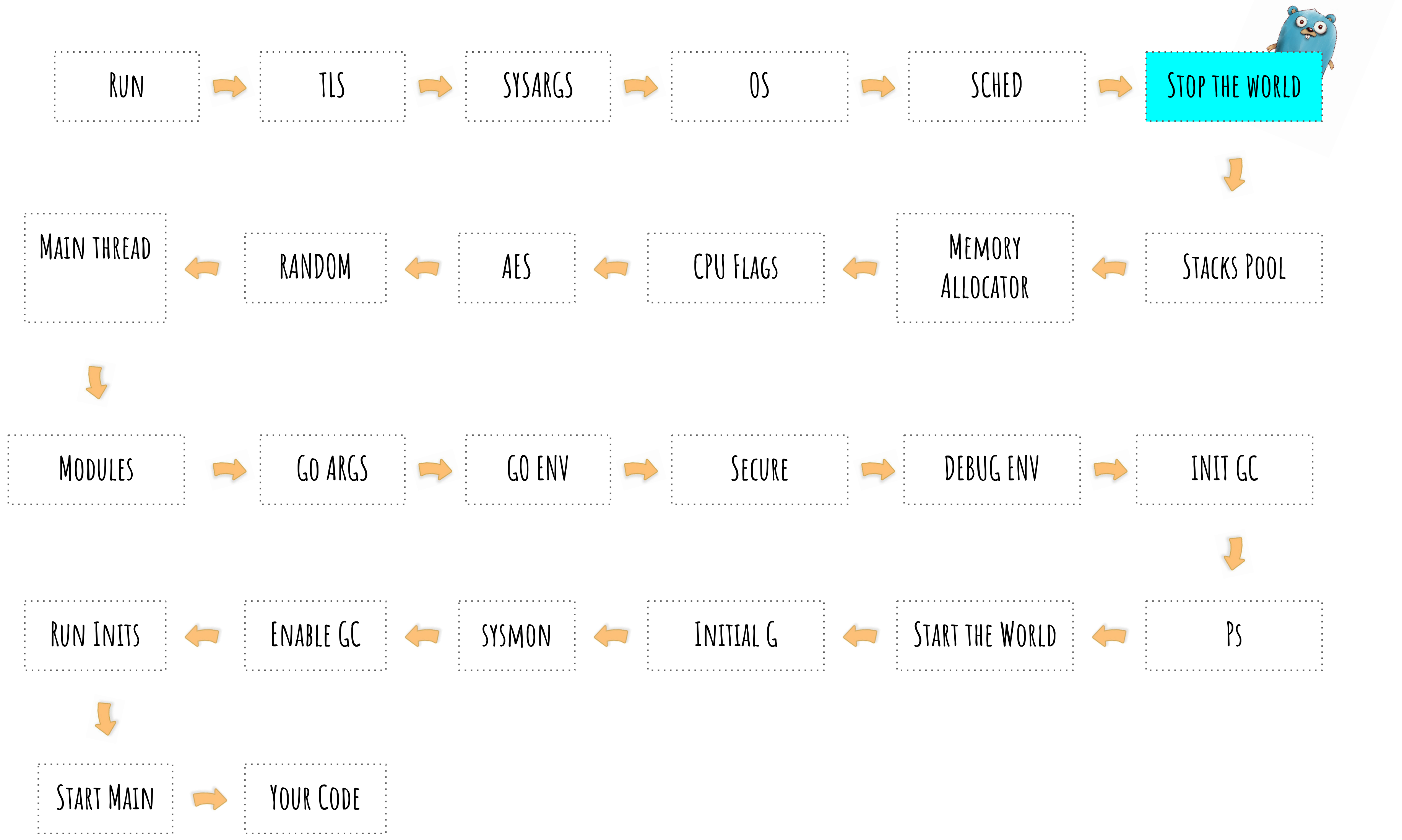


runtime/proc.go:694

The Scheduler

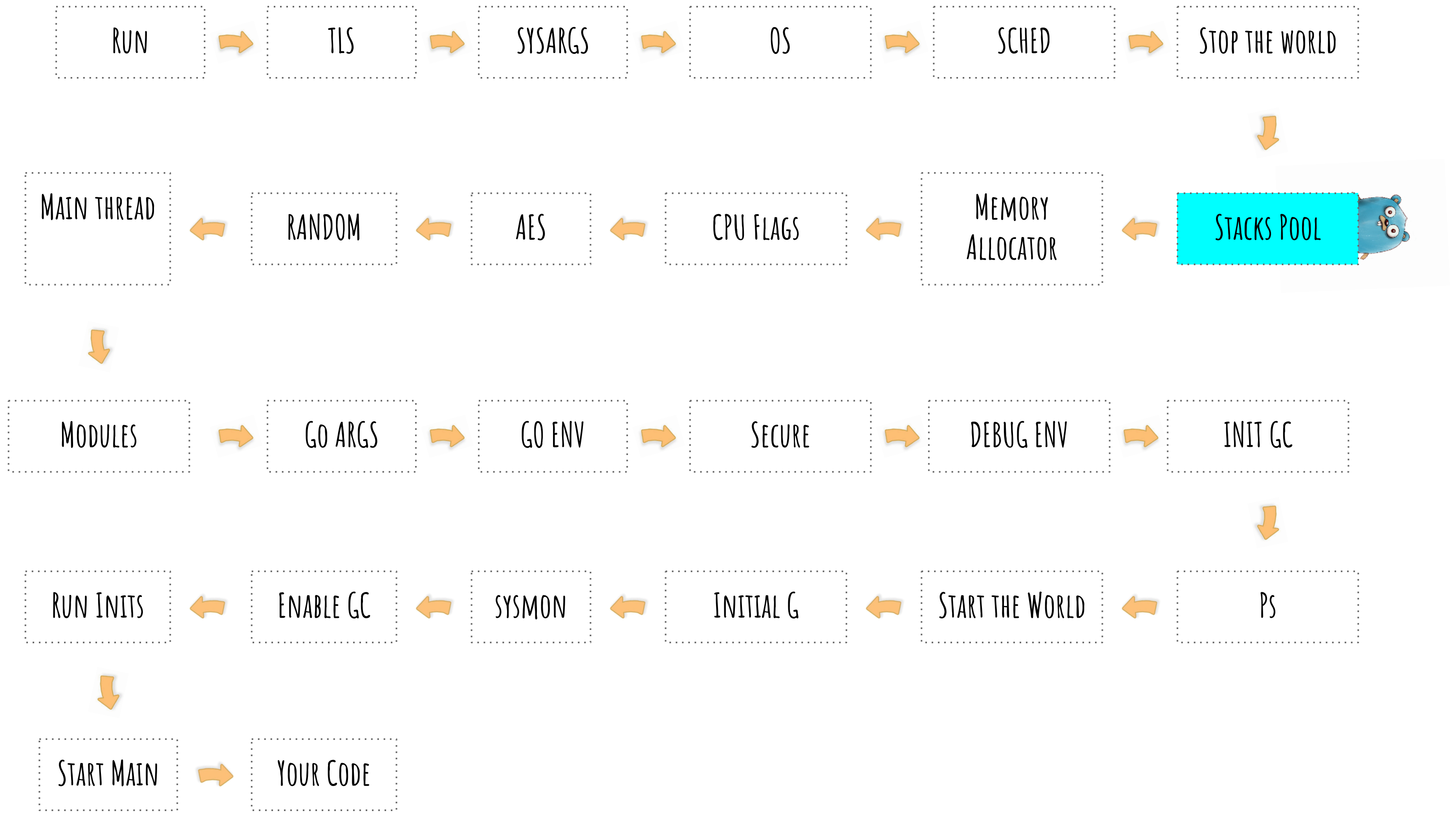


Stop The World



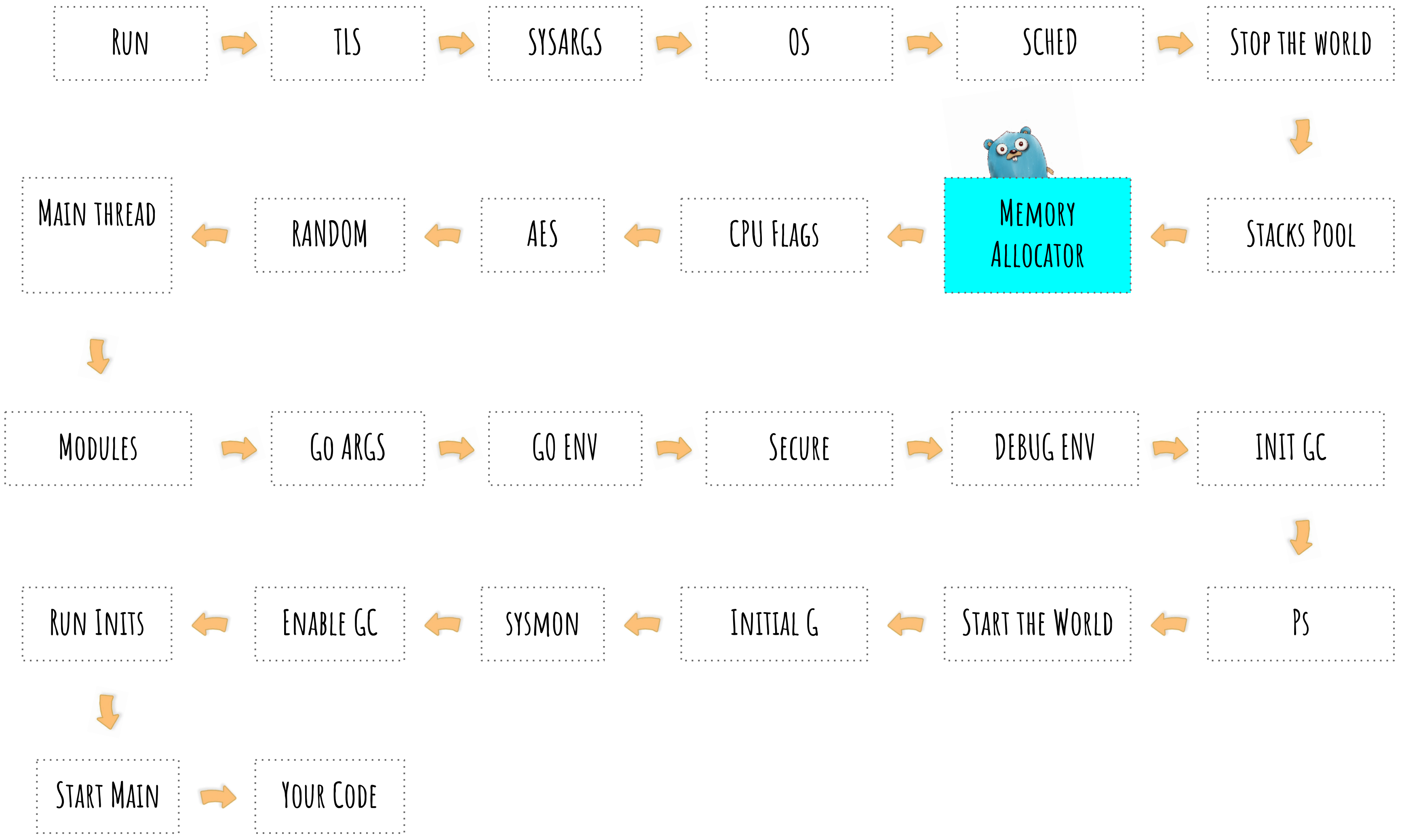
runtime/lockrank_on.go:309

Stacks Pool



runtime/stack.go:167

Memory Allocator



runtime/malloc.go:375

Memory Allocator

```go
package main

func BenchmarkAlloc(b *testing.B) {
    slice := []int{}
    for 1 := 0; i < b.N; i++ {
        slice = make([]int, 1024)
    }
    _ = slice
}
```

```
int = 8 bytes, 8*1024 = 8192
8192 B/op        1 allocs/op
```

```go
package main

func BenchmarkAlloc(b *testing.B) {
    slice := []int{}
    for 1 := 0; i < b.N; i++ {
        slice = make([]int, 1025)
    }
    _ = slice
}
```

```
int = 8 bytes, 8*1025 = 8200
???? B/op        1 allocs/op
```

Memory Allocator

```
package main

func BenchmarkAlloc(b *testing.B) {
    slice := []int{}
    for 1 := 0; i < b.N; i++ {
        slice = make([]int, 1024)
    }
    _ = slice
}
```

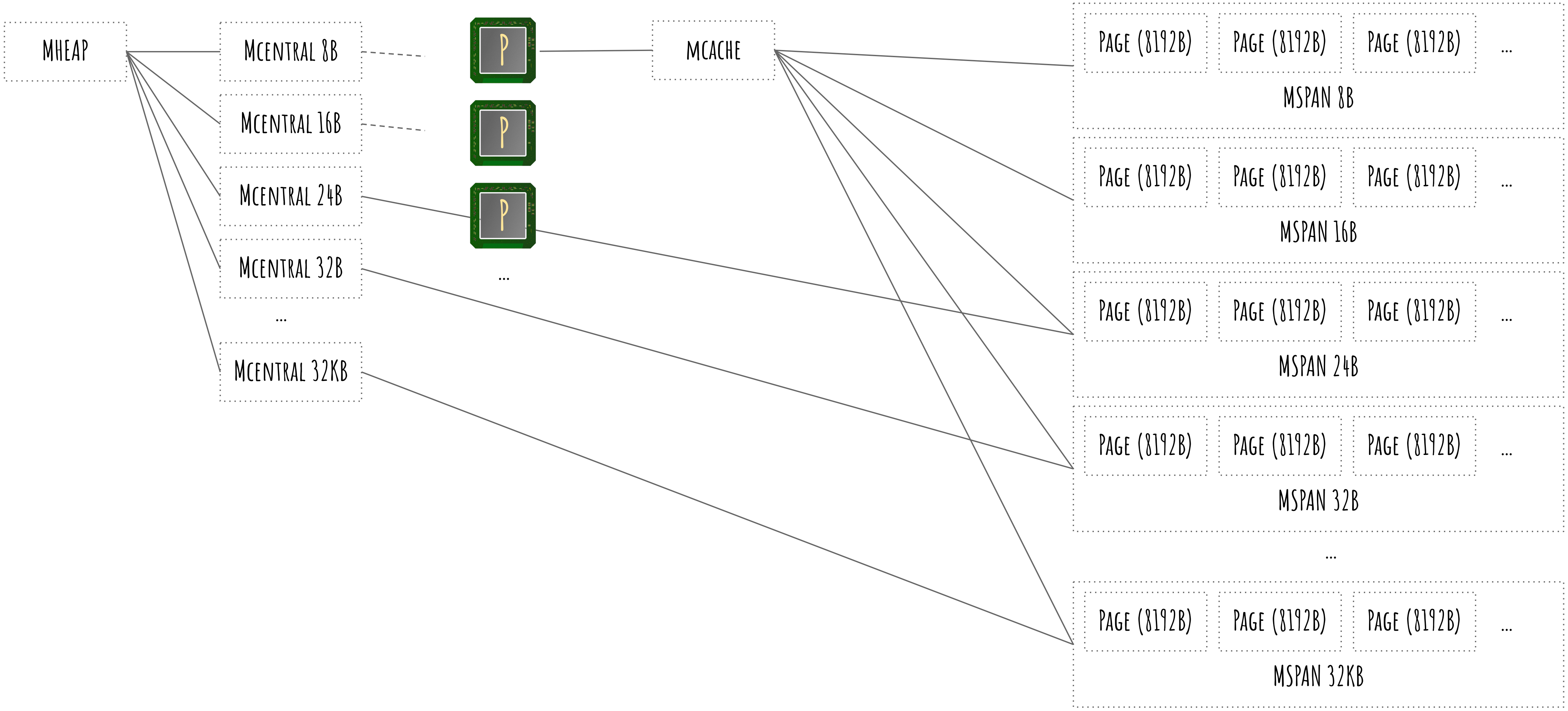
```
int = 8 bytes, 8*1024 = 8192
8192 B/op       1 allocs/op
```

```
package main

func BenchmarkAlloc(b *testing.B) {
    slice := []int{}
    for 1 := 0; i < b.N; i++ {
        slice = make([]int, 1025)
    }
    _ = slice
}
```

```
int = 8 bytes, 8*1025 = 8200
9472 B/op       1 allocs/op
```

Memory Allocator



Memory Allocator

```
package main

func BenchmarkAlloc(b *testing.B) {
    slice := []int{}
    for 1 := 0; i < b.N; i++ {
        slice = make([]int, 1024*1024)
    }
    _ = slice
}
```

```
int = 8 bytes, 8*1024*1024 = 88388608
8388619 B/op        1 allocs/op
```

```
package main

func BenchmarkAlloc(b *testing.B) {
    slice := []int{}
    for 1 := 0; i < b.N; i++ {
        slice = make([]int, 1024*1024+1)
    }
    _ = slice
}
```

```
int = 8 bytes, 8*1024*1024+1 = 88388616
??????? B/op        1 allocs/op
```

Memory Allocator

```go
package main

func BenchmarkAlloc(b *testing.B) {
    slice := []int{}
    for 1 := 0; i < b.N; i++ {
        slice = make([]int, 1024*1024)
    }
    _ = slice
}
```

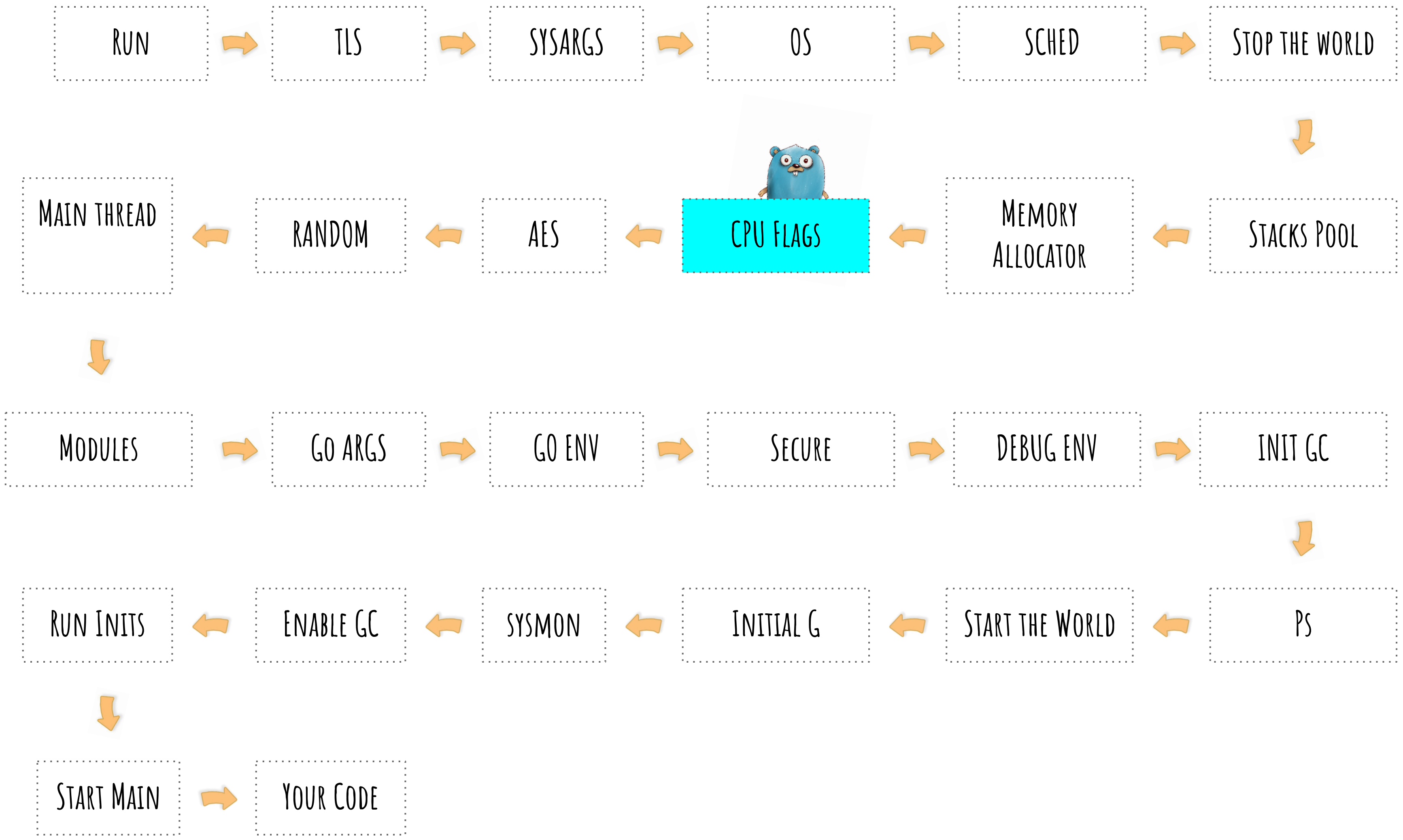
```
int = 8 bytes, 8*1024*1024 = 88388608
8388619 B/op          1 allocs/op
```

```go
package main

func BenchmarkAlloc(b *testing.B) {
    slice := []int{}
    for 1 := 0; i < b.N; i++ {
        slice = make([]int, 1024*1024+1)
    }
    _ = slice
}
```

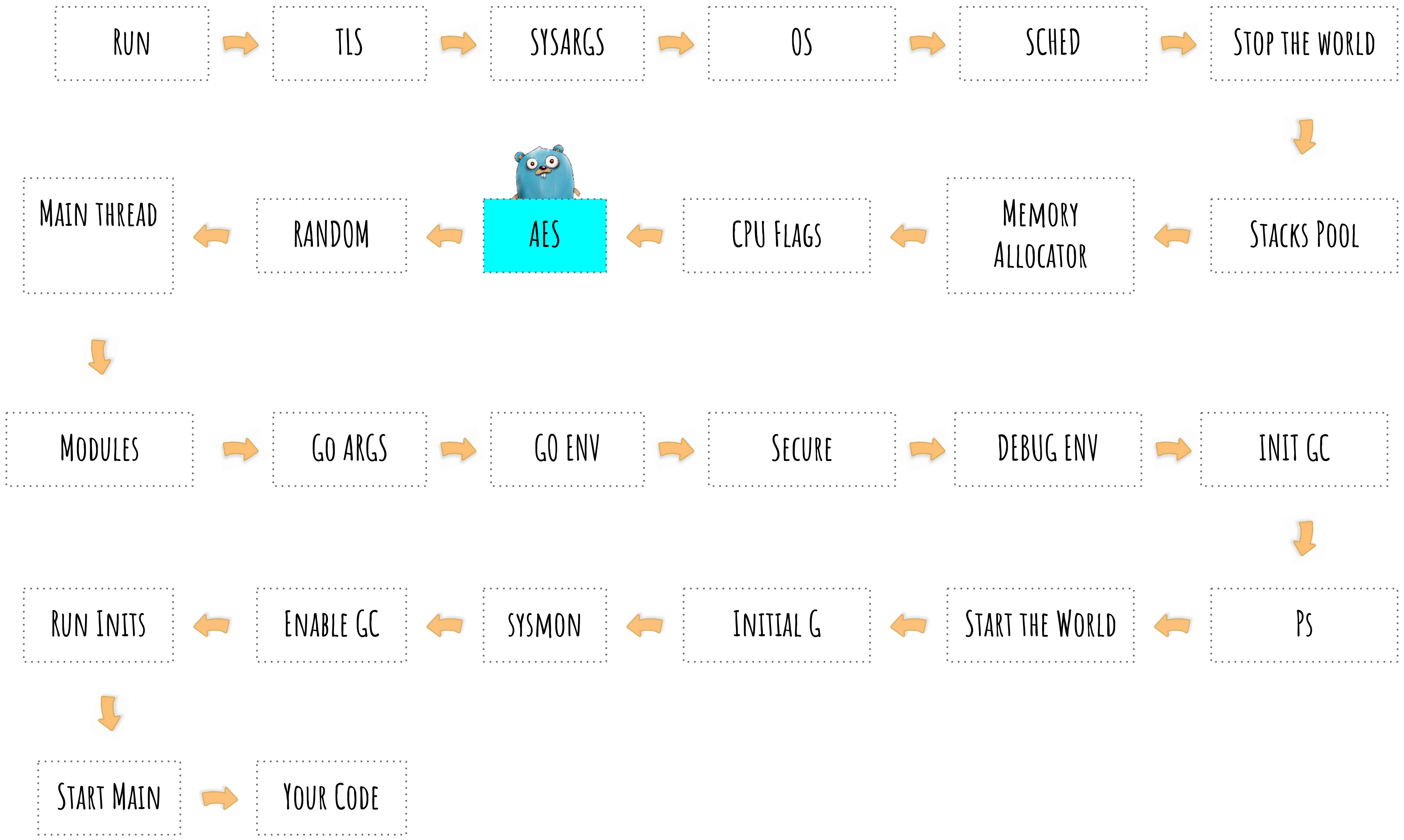
```
int = 8 bytes, 8*1024*1024+1 = 88388616
8396811 B/op          1 allocs/op
```

CPU Flags



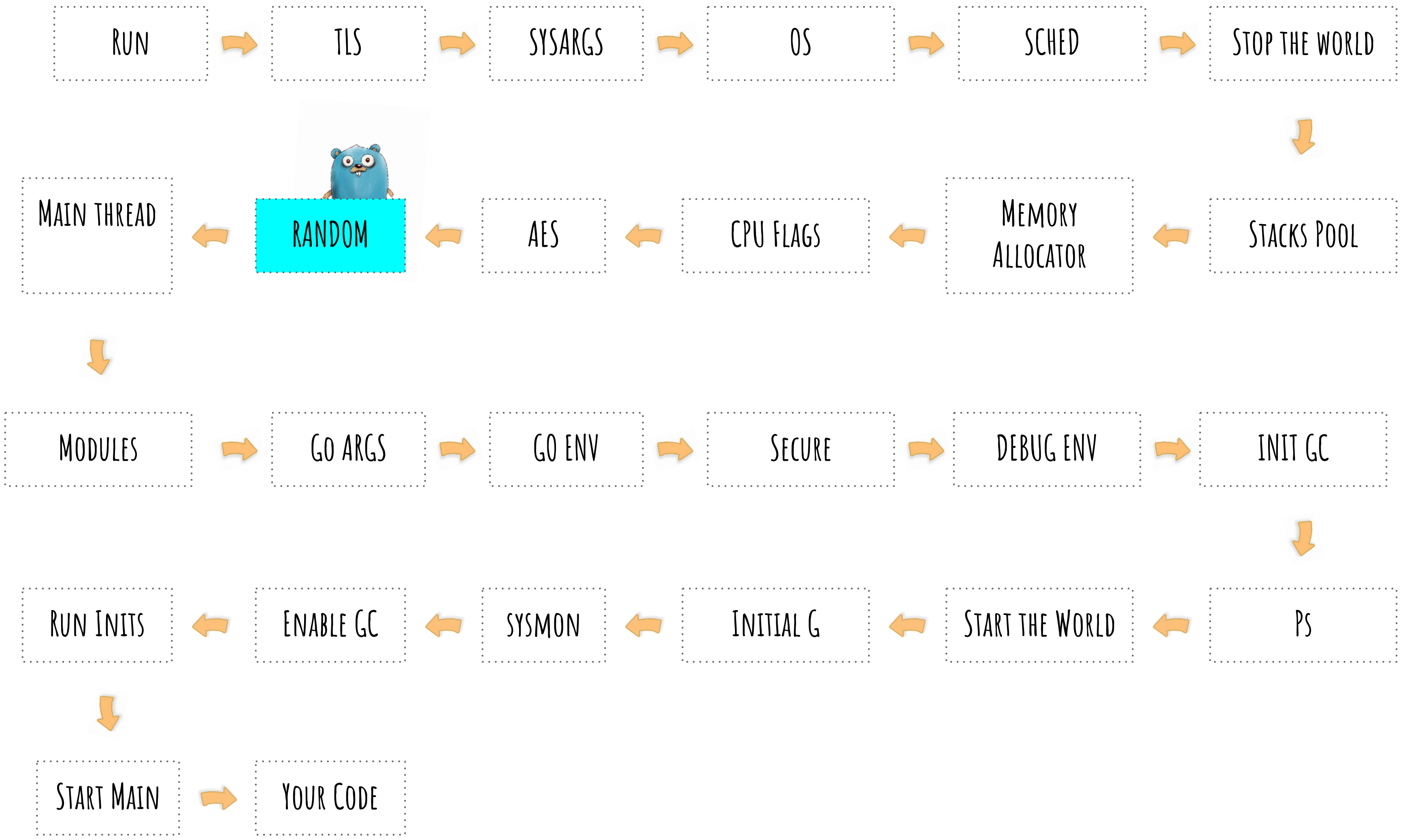
runtime/proc.go:634

AES



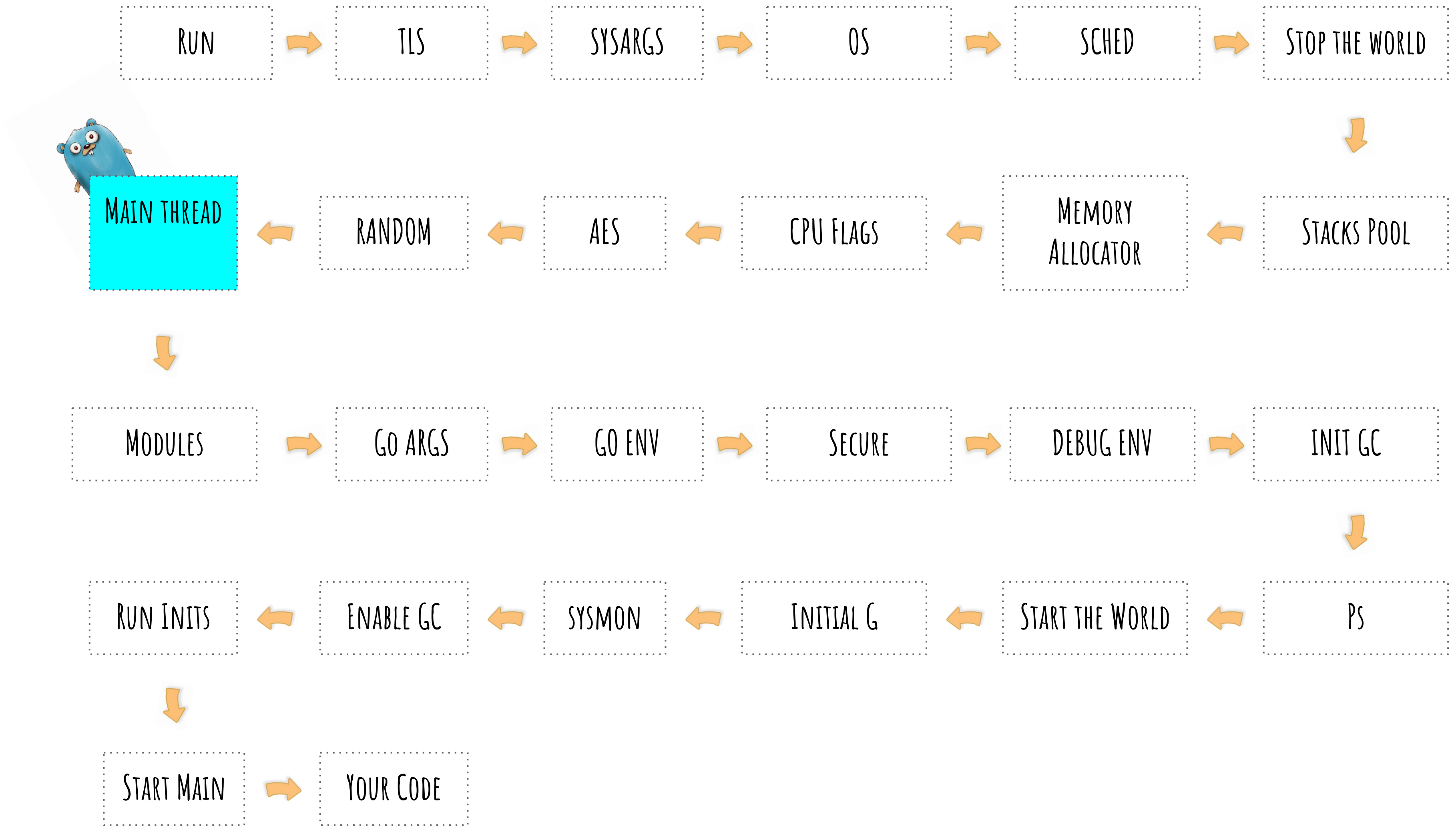
runtime/alg.go:312

Random



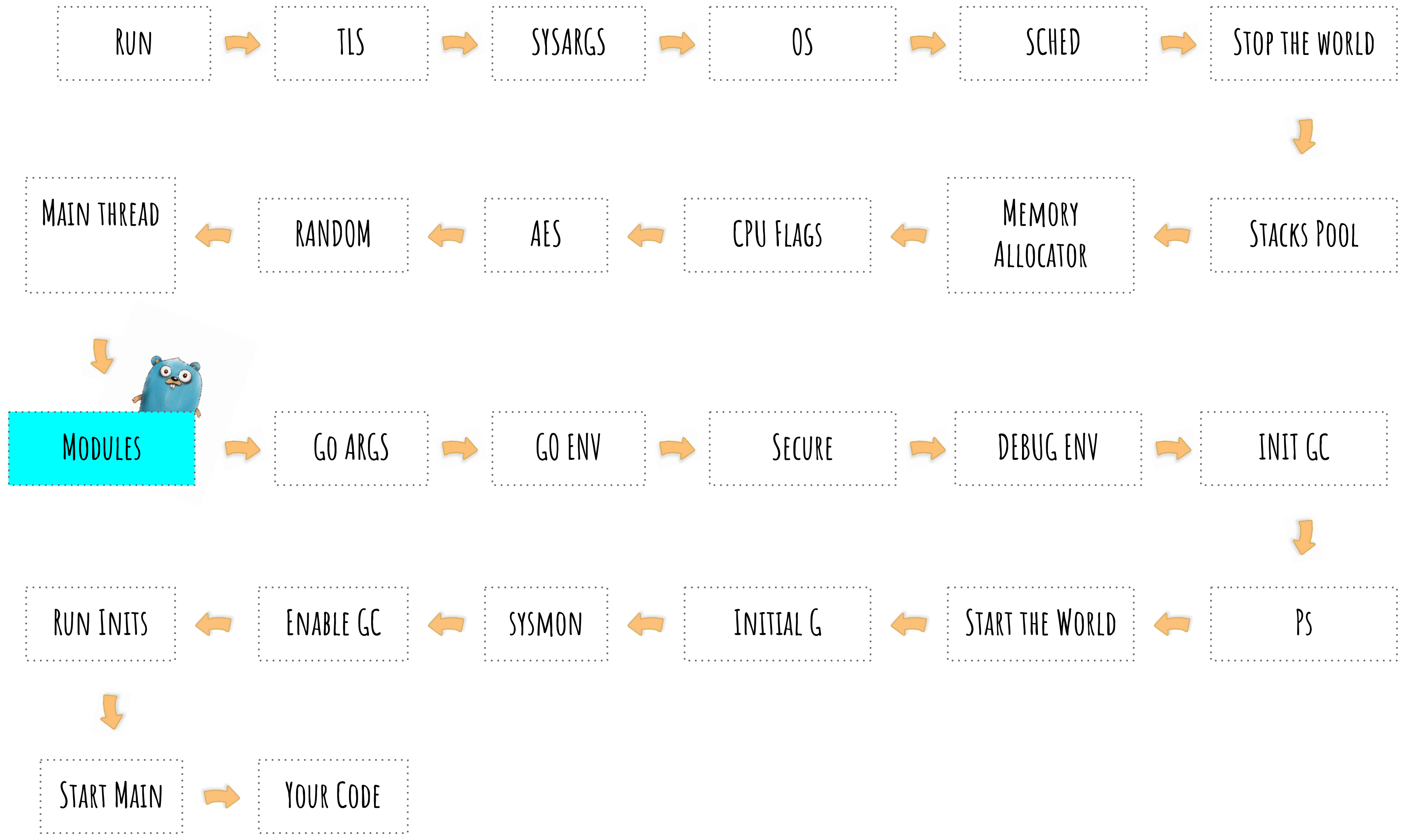
runtime/proc.go:884

Main Thread



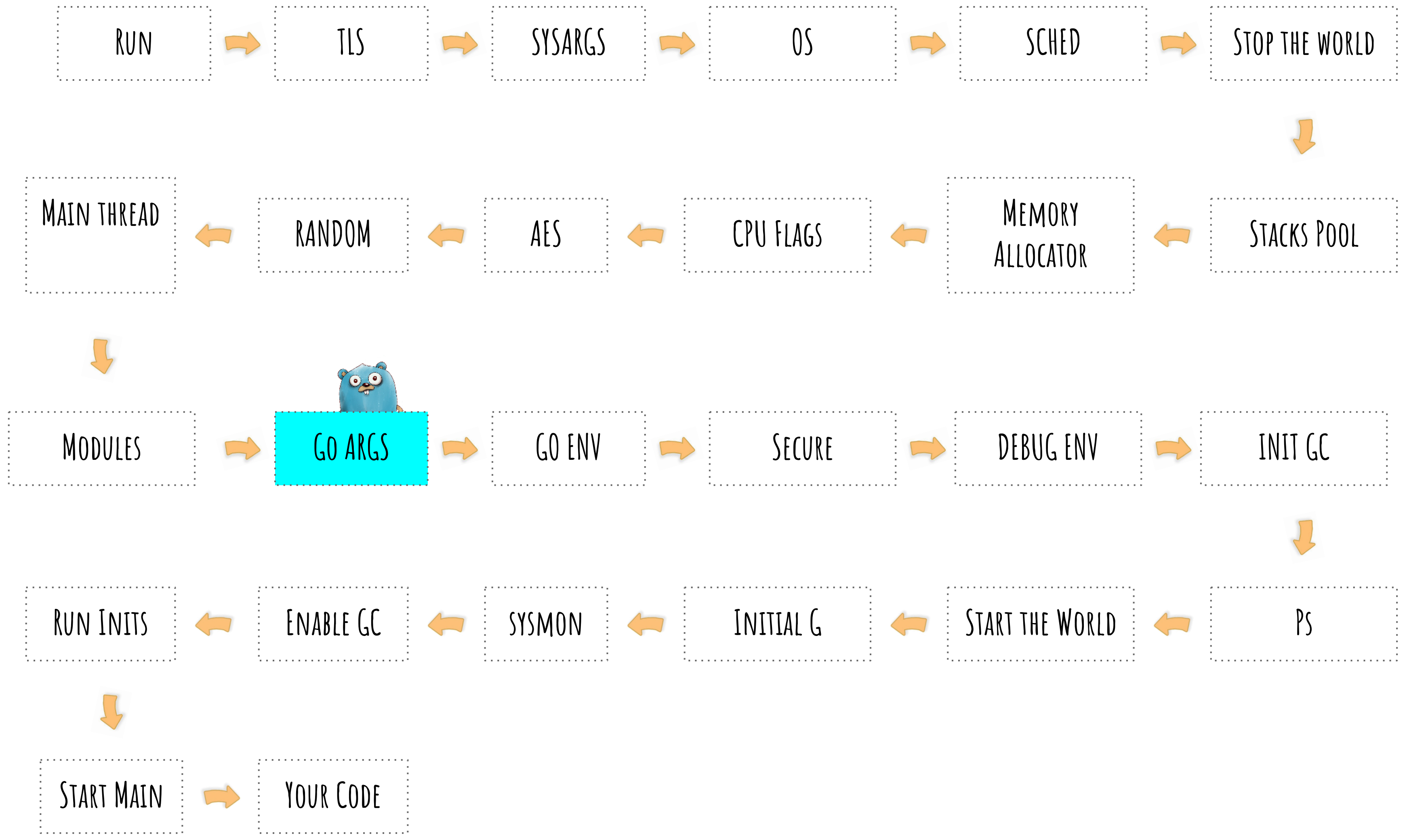
runtime/proc.go:823

Modules



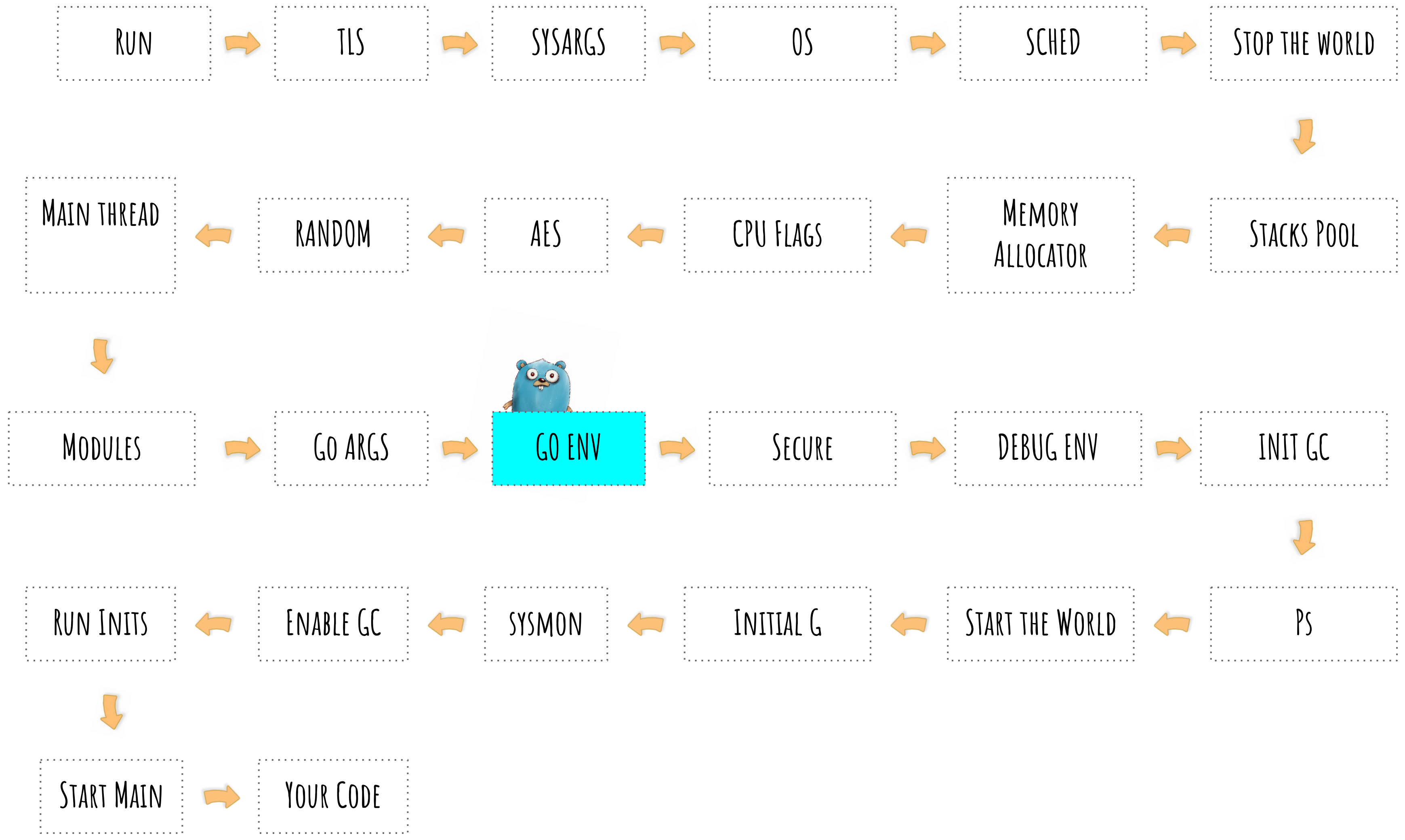
runtime/symtab.go:452

Go Args



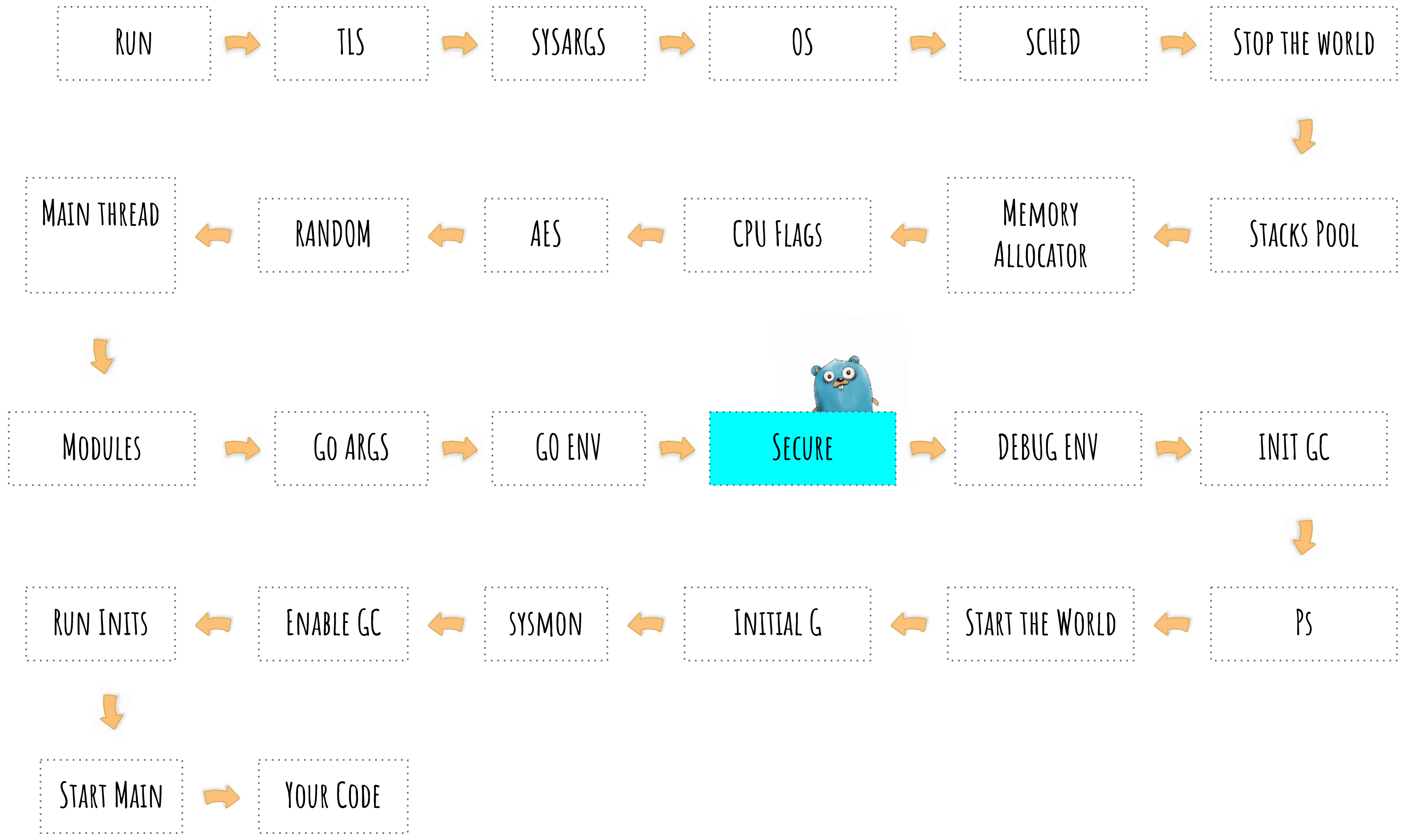
runtime/runtime1.go:72

GO ENV



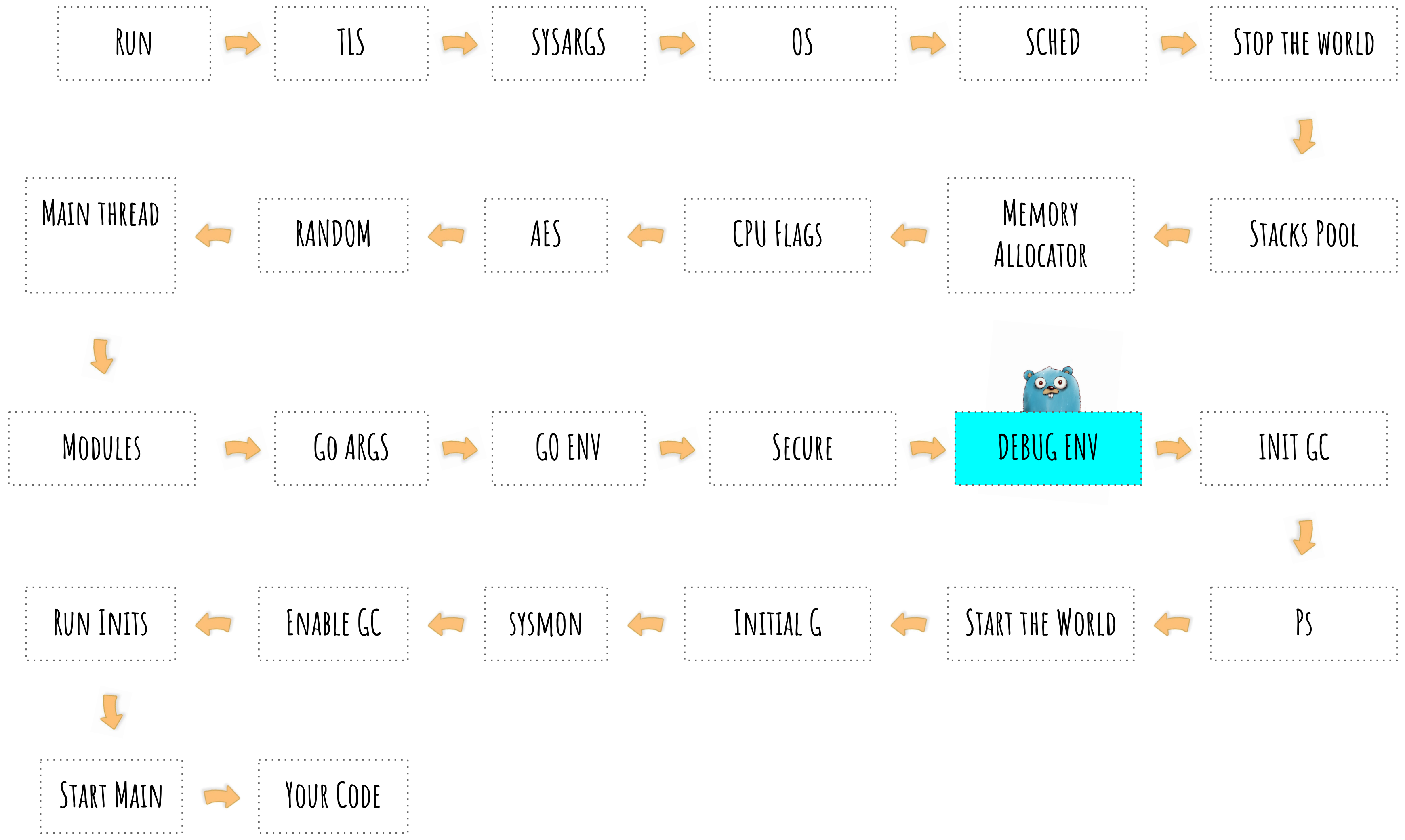
runtime/runtime1.go:82

Secure



runtime/security_unix.go:9

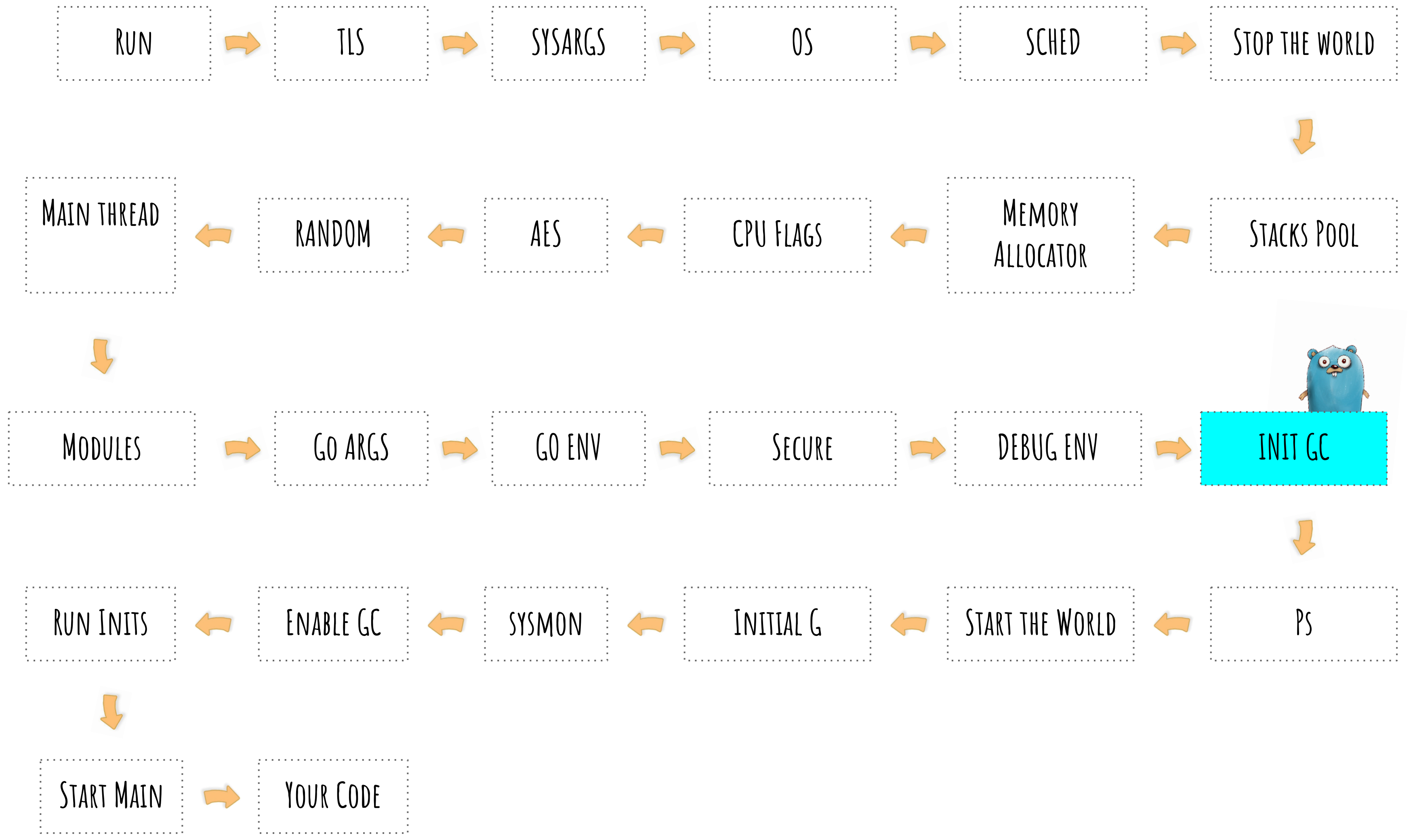
Debug Env



runtime/runtime1.go:367

Too much info? Take a break.
Look... here is a Kitten

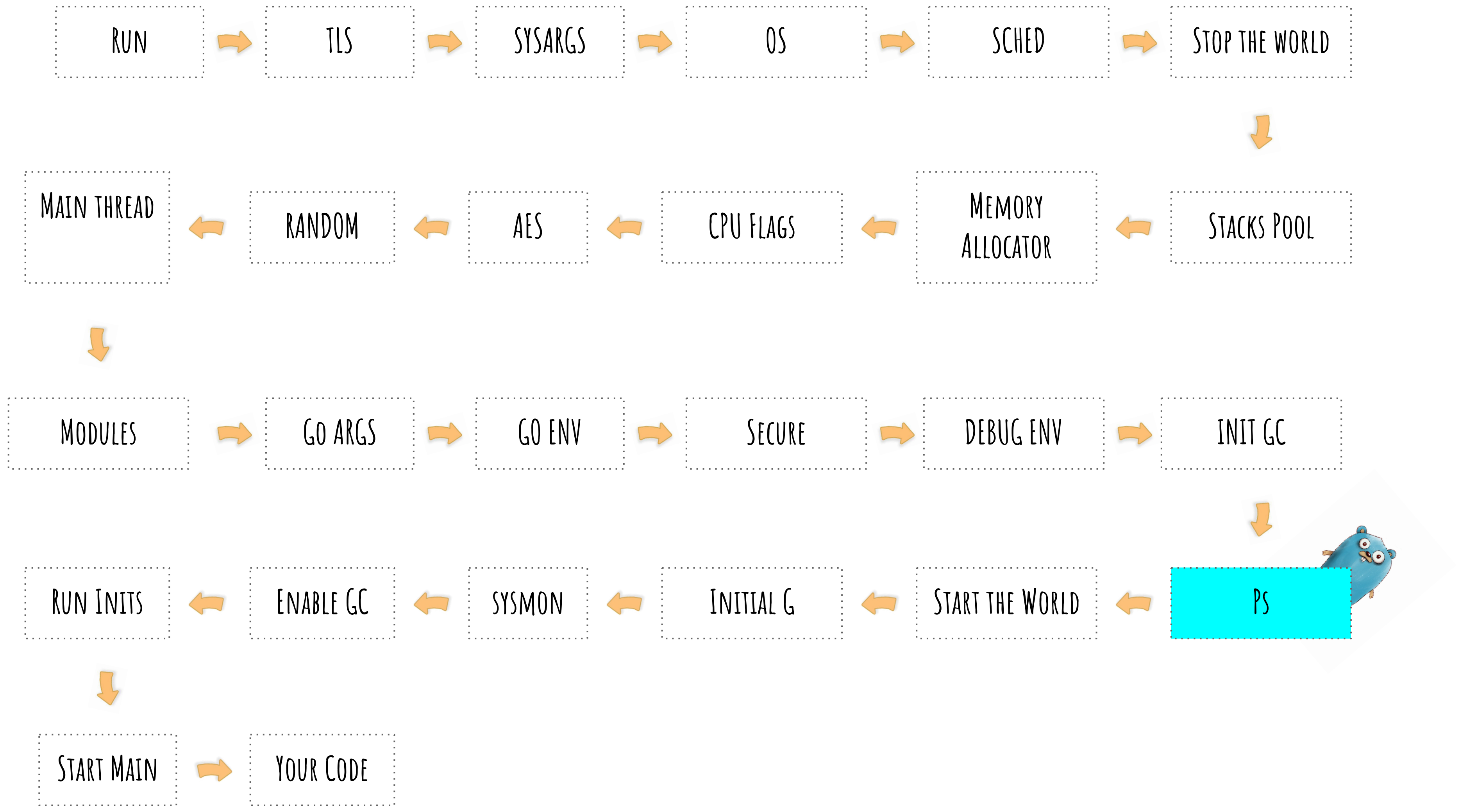
Init GC



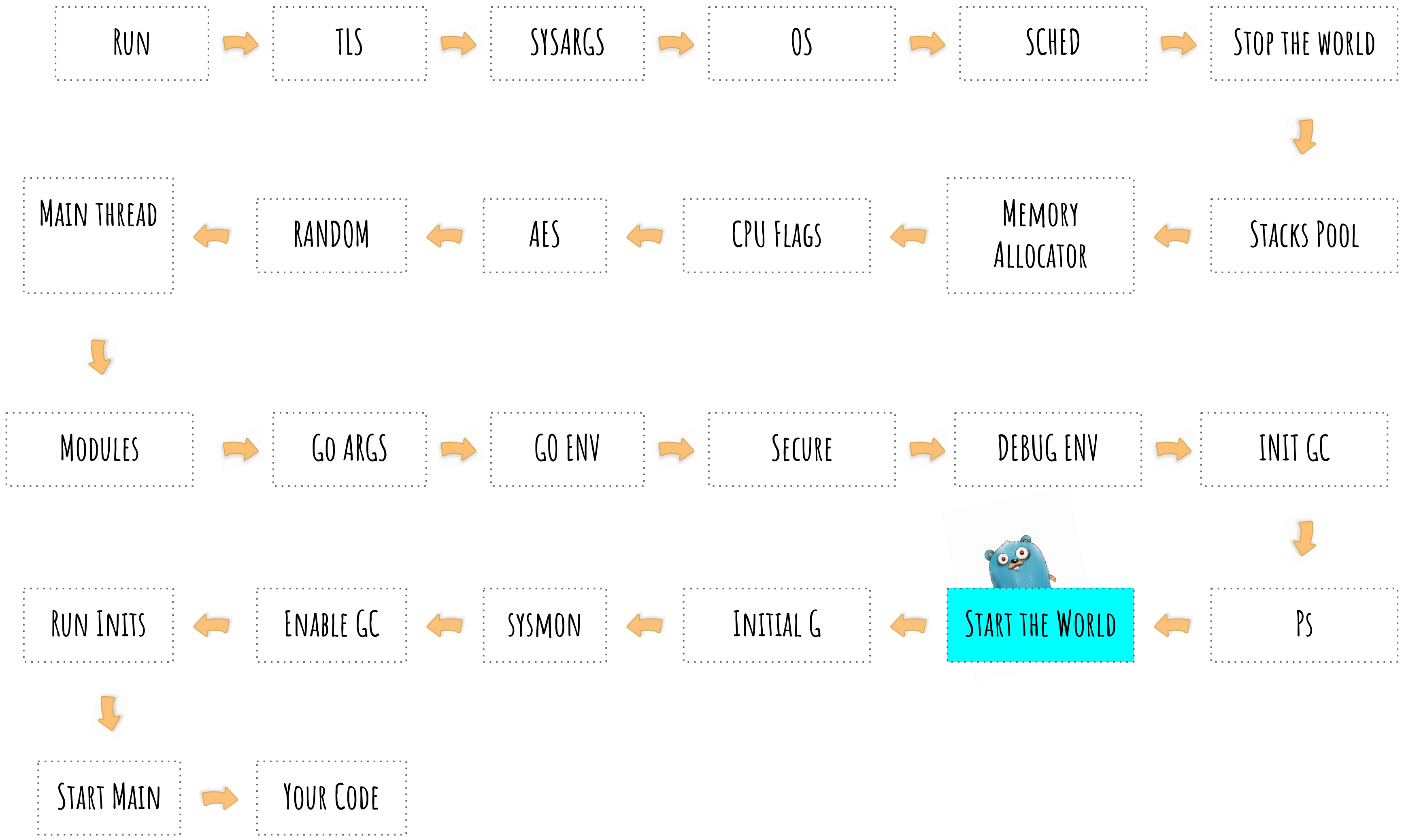
runtime/mgc.go:184

Init GC

Ps

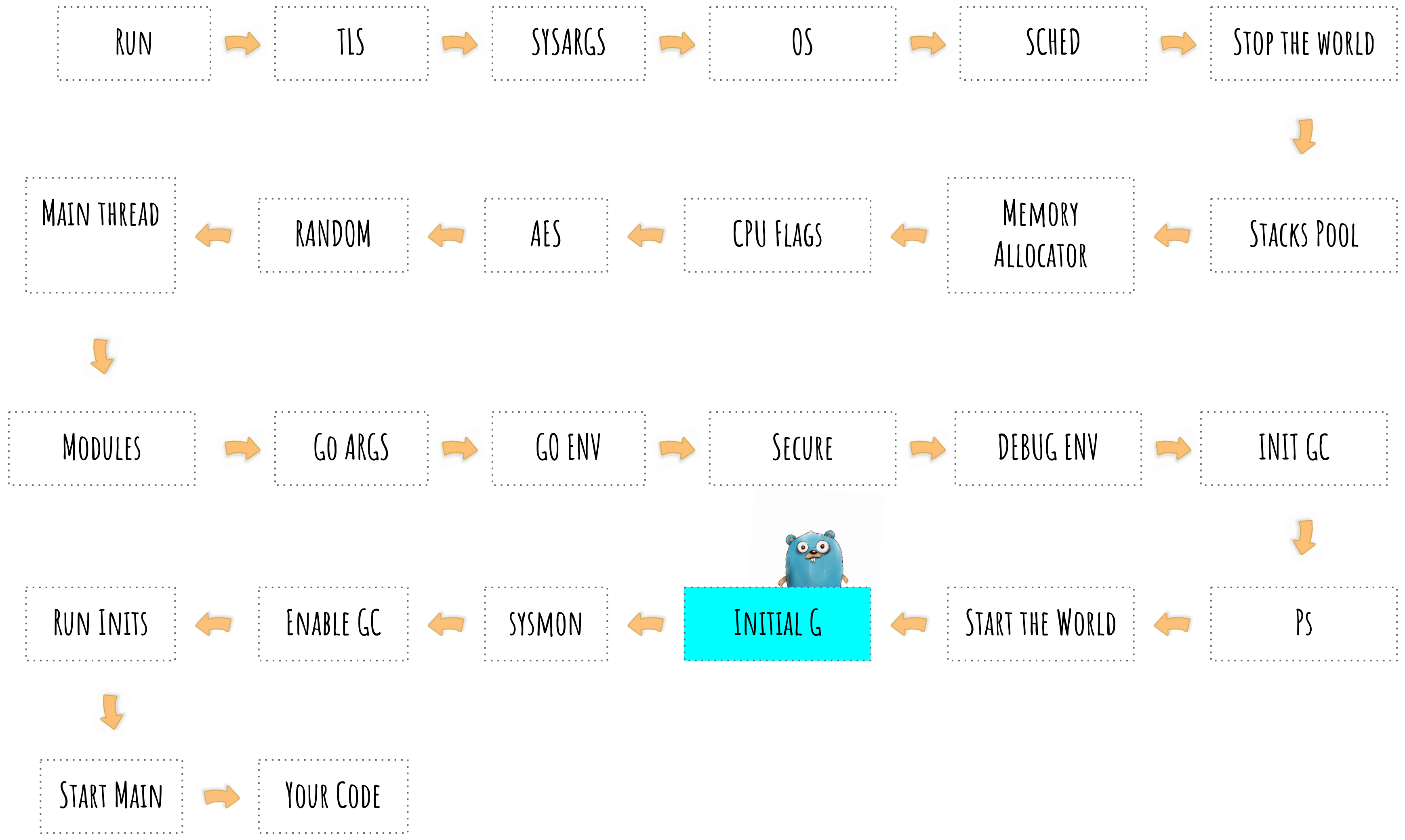

Run
TLS
sysargs
OS
Sched
Stop the world
Stacks Pool
Memory Allocator
CPU Flags
AES
Random
Main thread
Modules
Go args
Go env
Secure
Debug env
Init GC
Ps
Start the World
Initial G
Sysmon
Enable GC
Run Inits
Start Main
Your Code

Start The World



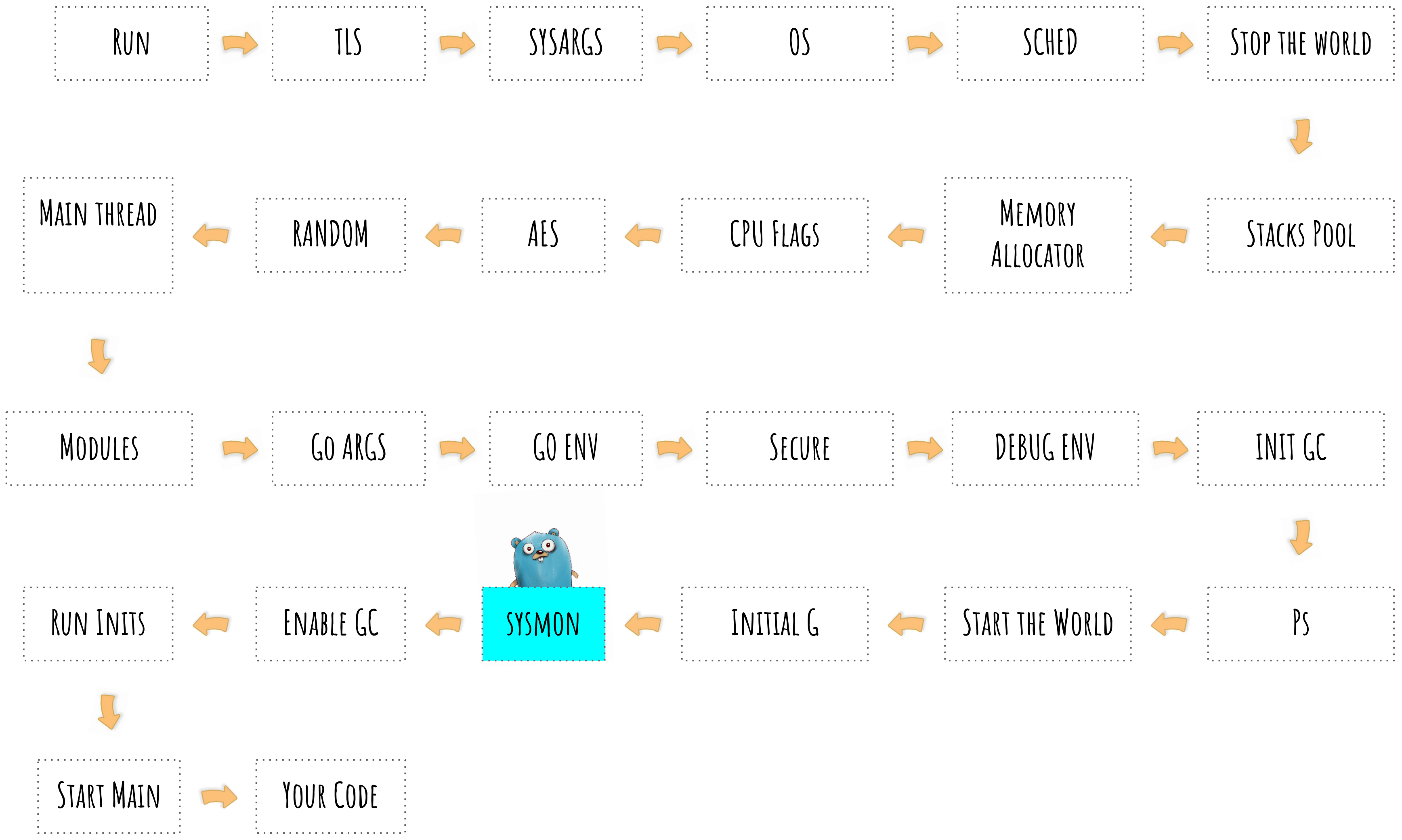
runtime/proc.go:1280

Initial G



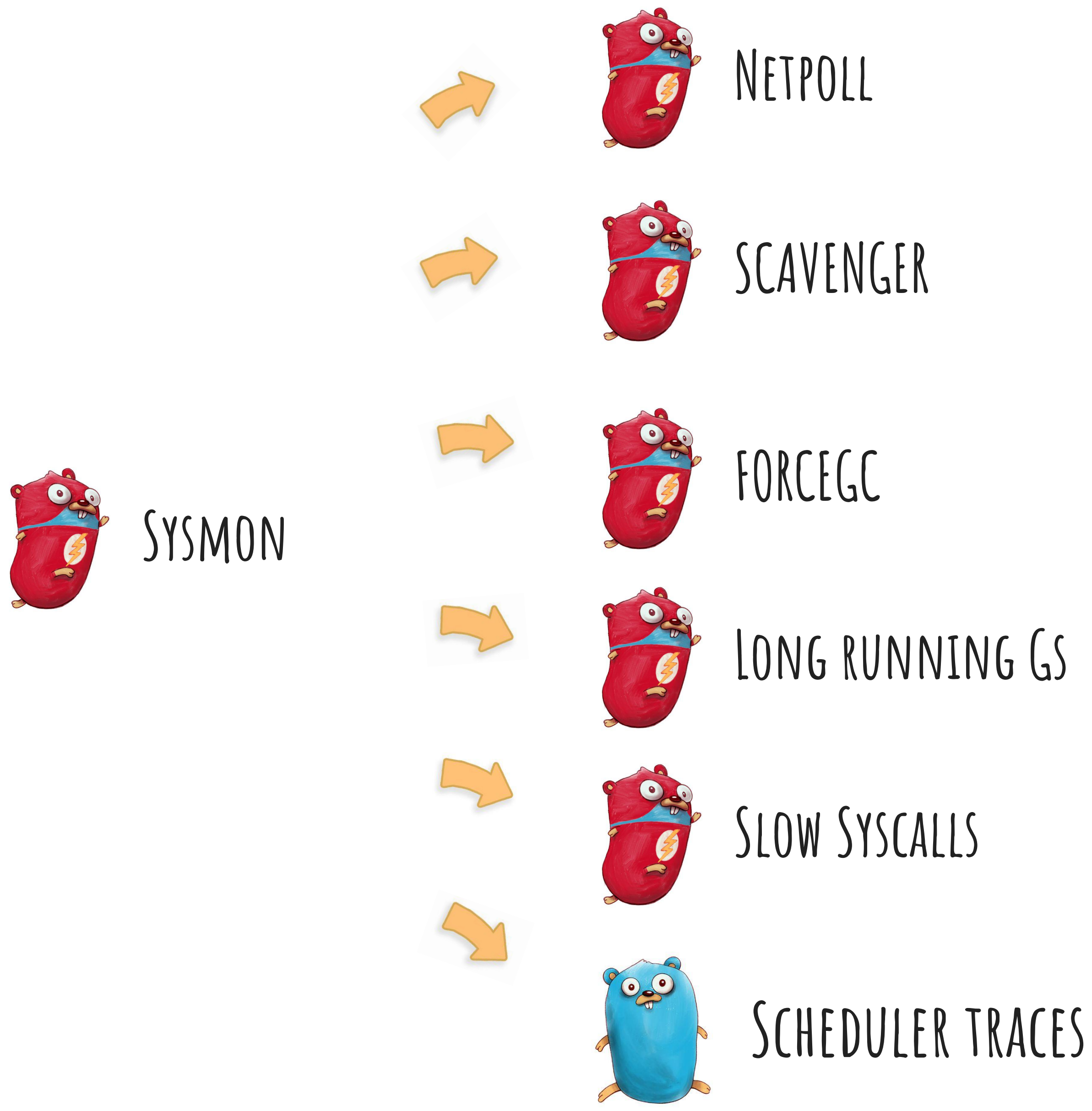
runtime/proc.go:144

SYSMON

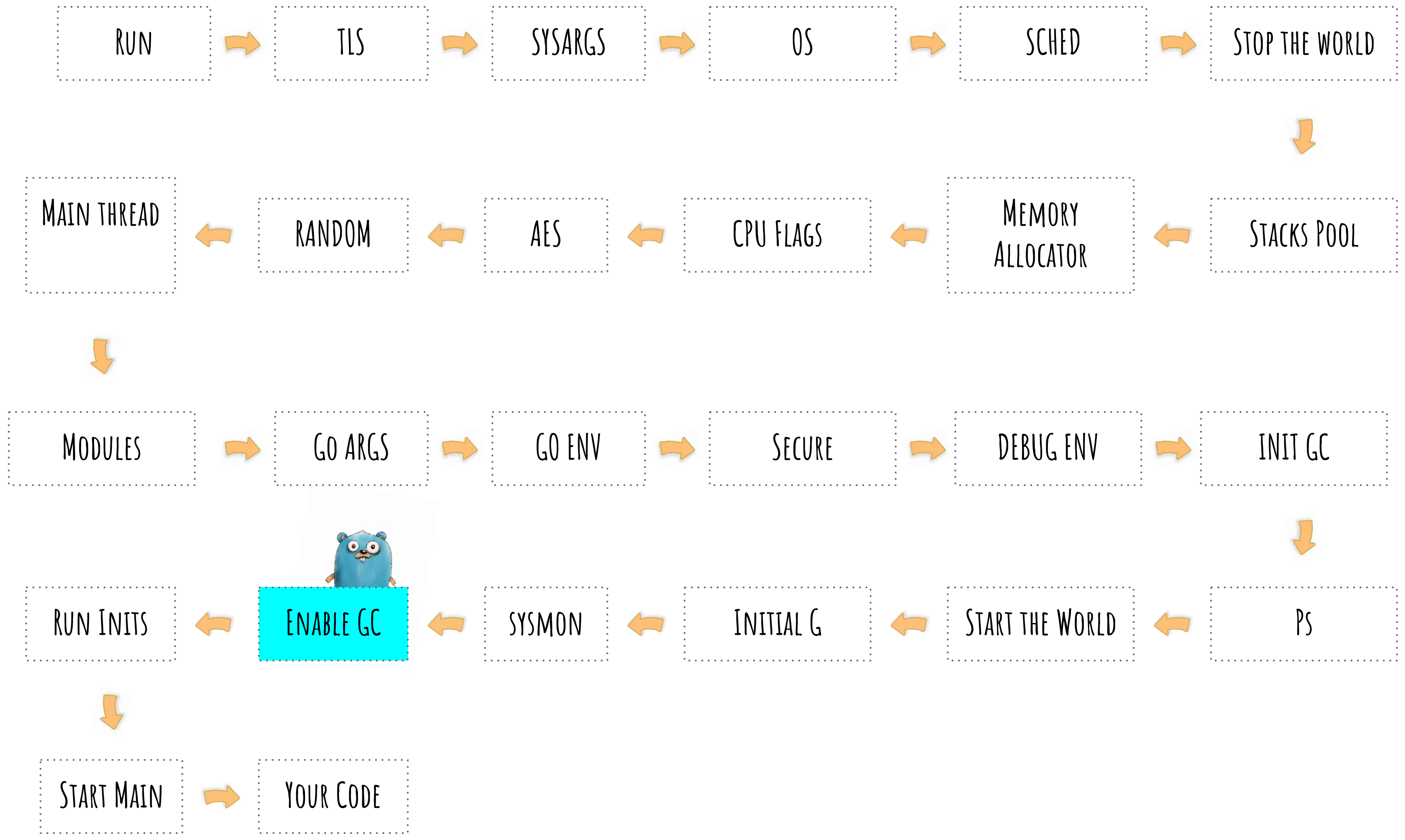


runtime/proc.go:5515

SYSMON

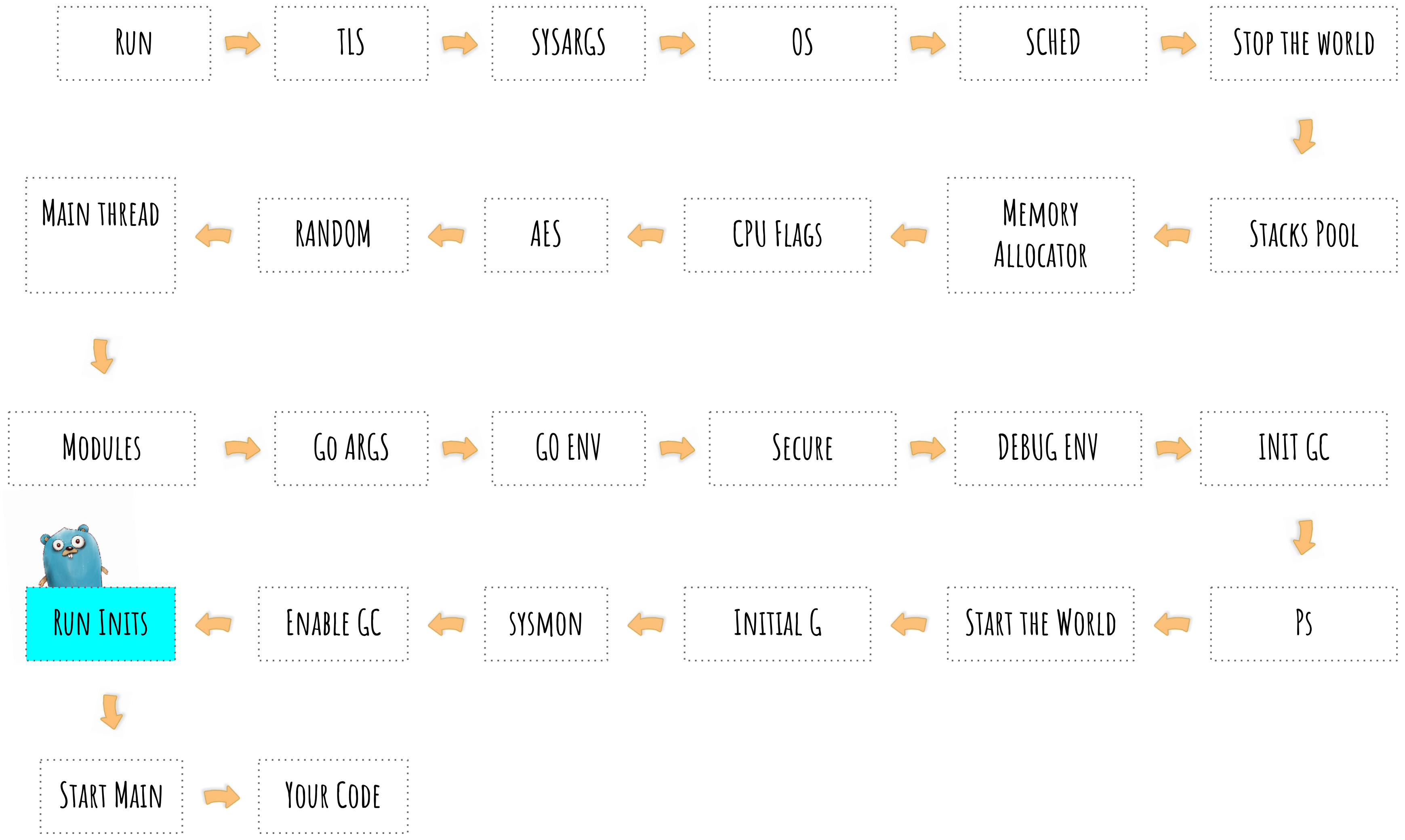

Sysmon
Netpoll
Scavenger
ForceGC
Long running Gs
Slow Syscalls
Scheduler traces

Enable GC



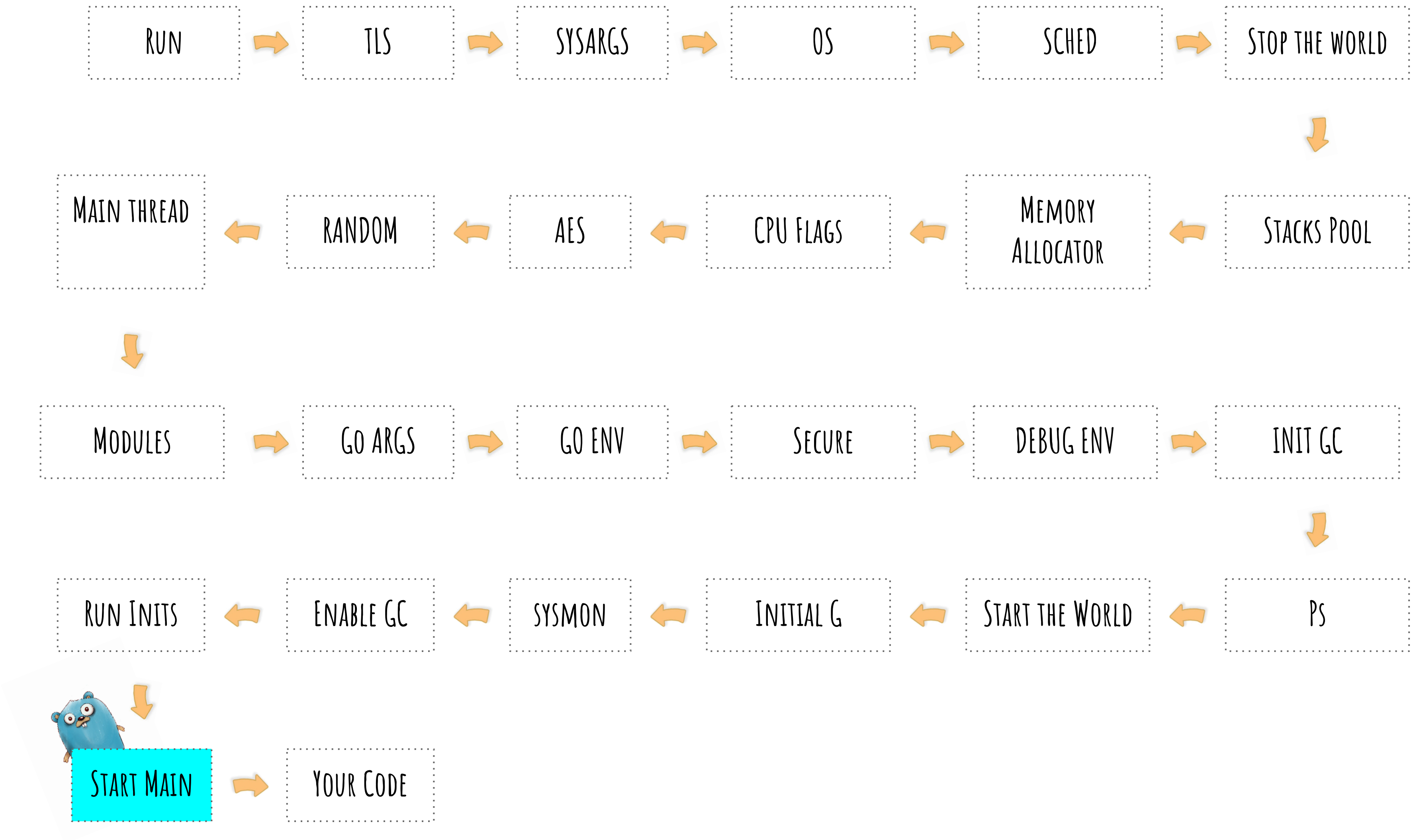
runtime/mgc.go:197

Run Inits



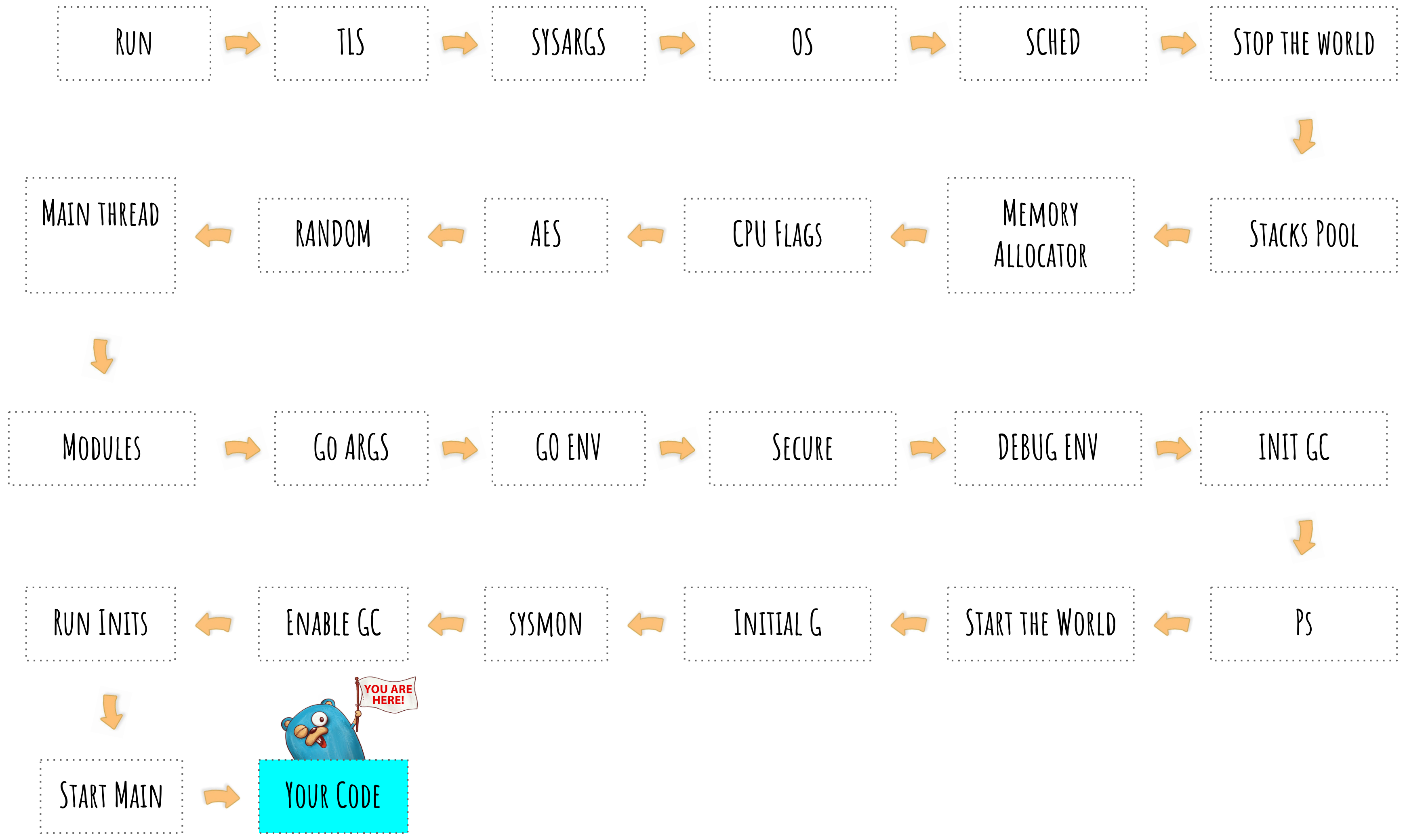
runtime/proc.go:248

Start Main



runtime/proc.go:266

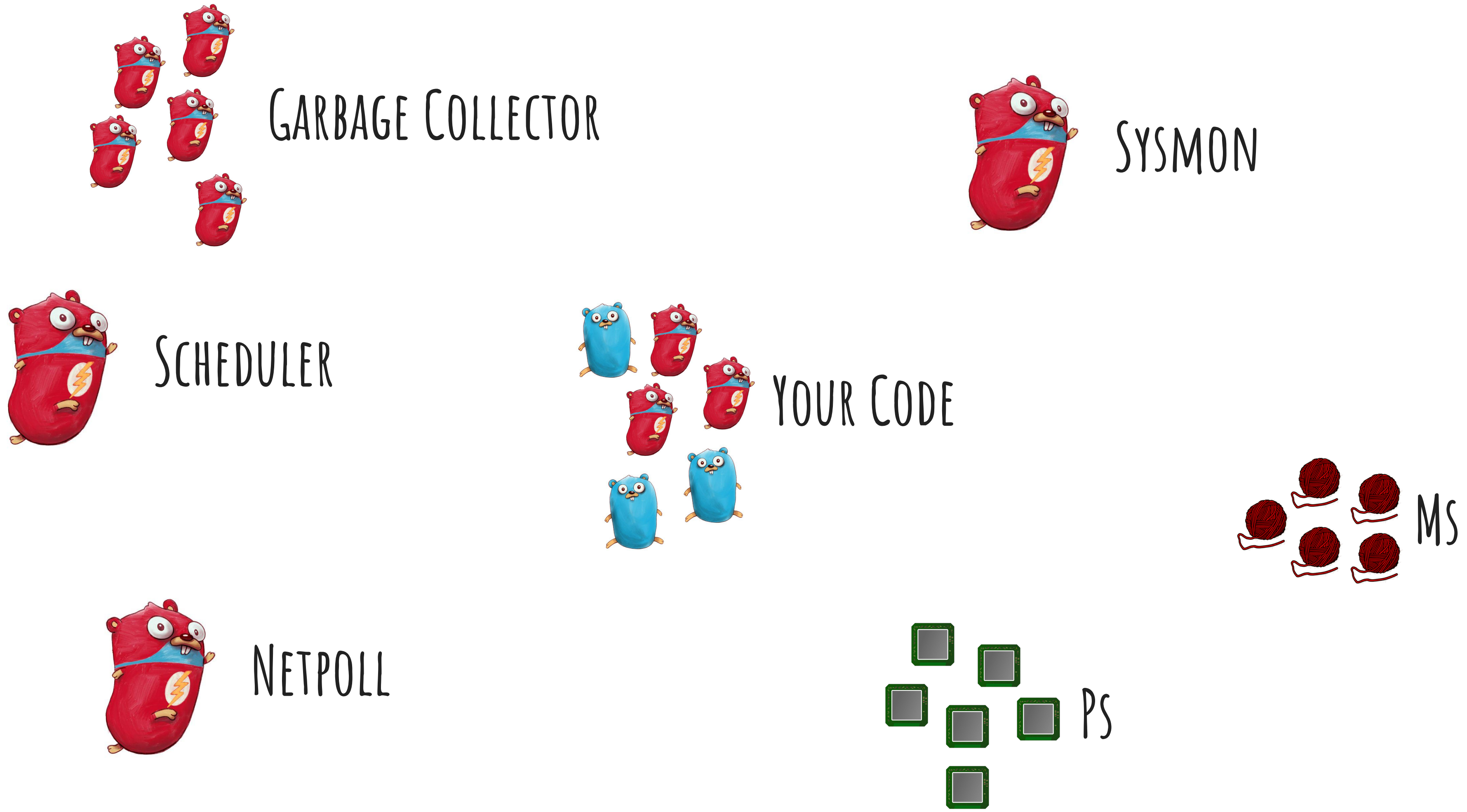
Run Inits


Run
TLS
SYSARGS
OS
SCHED
Stop the world
Stacks Pool
Memory Allocator
CPU Flags
AES
Random
Main thread
Modules
Go ARGS
GO ENV
Secure
DEBUG ENV
INIT GC
Ps
Start the World
Initial G
SYSMON
Enable GC
Run Inits
Start Main
Your Code
YOU ARE HERE!

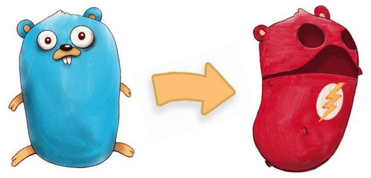
During Runtime

During running

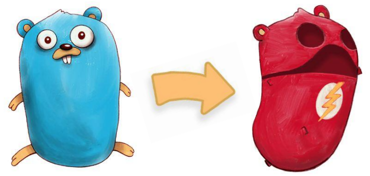
During running



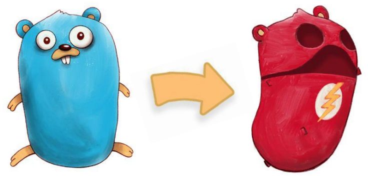
During running



Slices, maps and channels operations



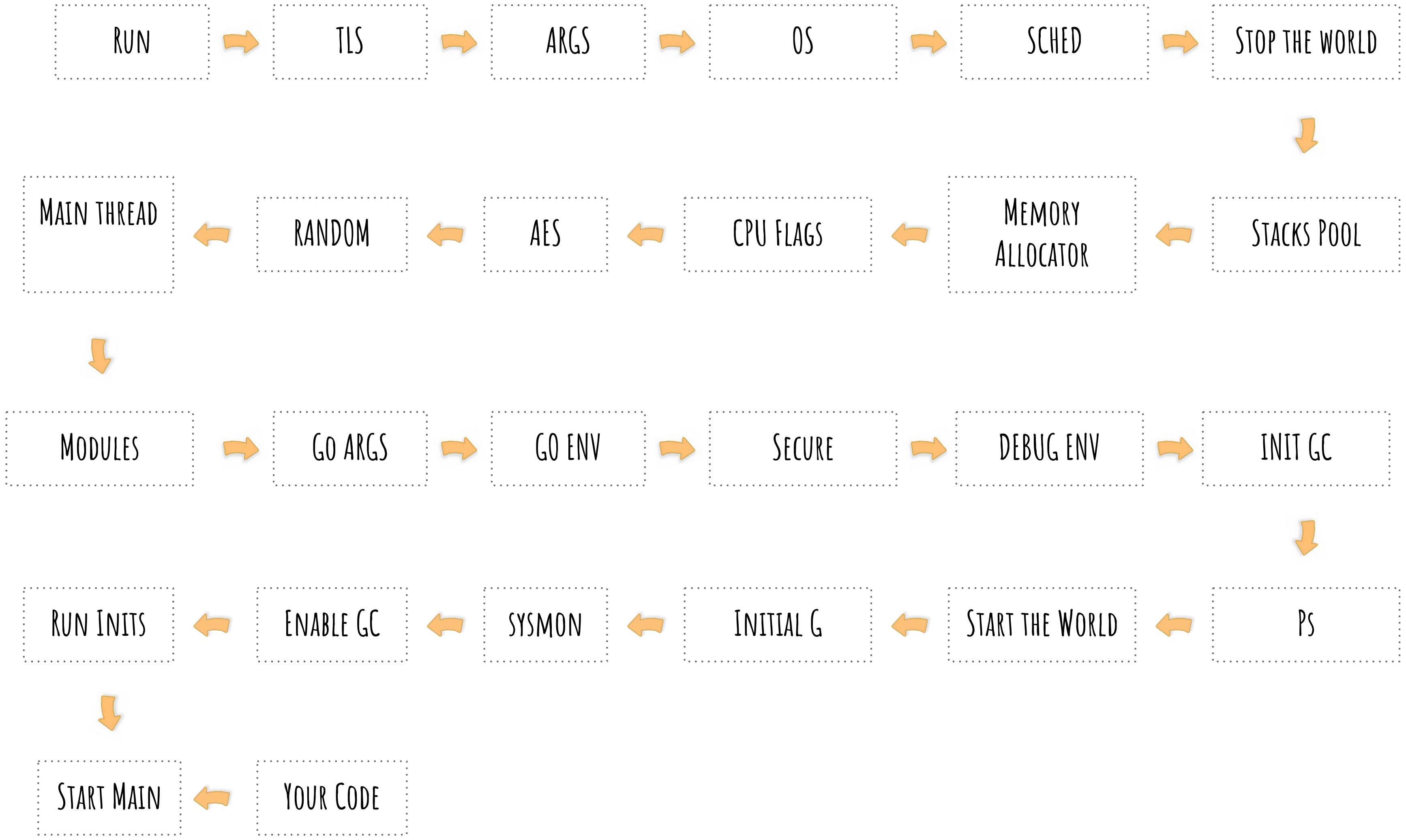
Spawn goroutine



Alloc Memory

Summary

Summary


Run
TLS
Args
OS
Sched
Stop the world
Stacks Pool
Memory Allocator
CPU Flags
AES
Random
Main thread
Modules
Go Args
Go Env
Secure
Debug Env
Init GC
Ps
Start the World
Initial G
Sysmon
Enable GC
Run Inits
Start Main
Your Code

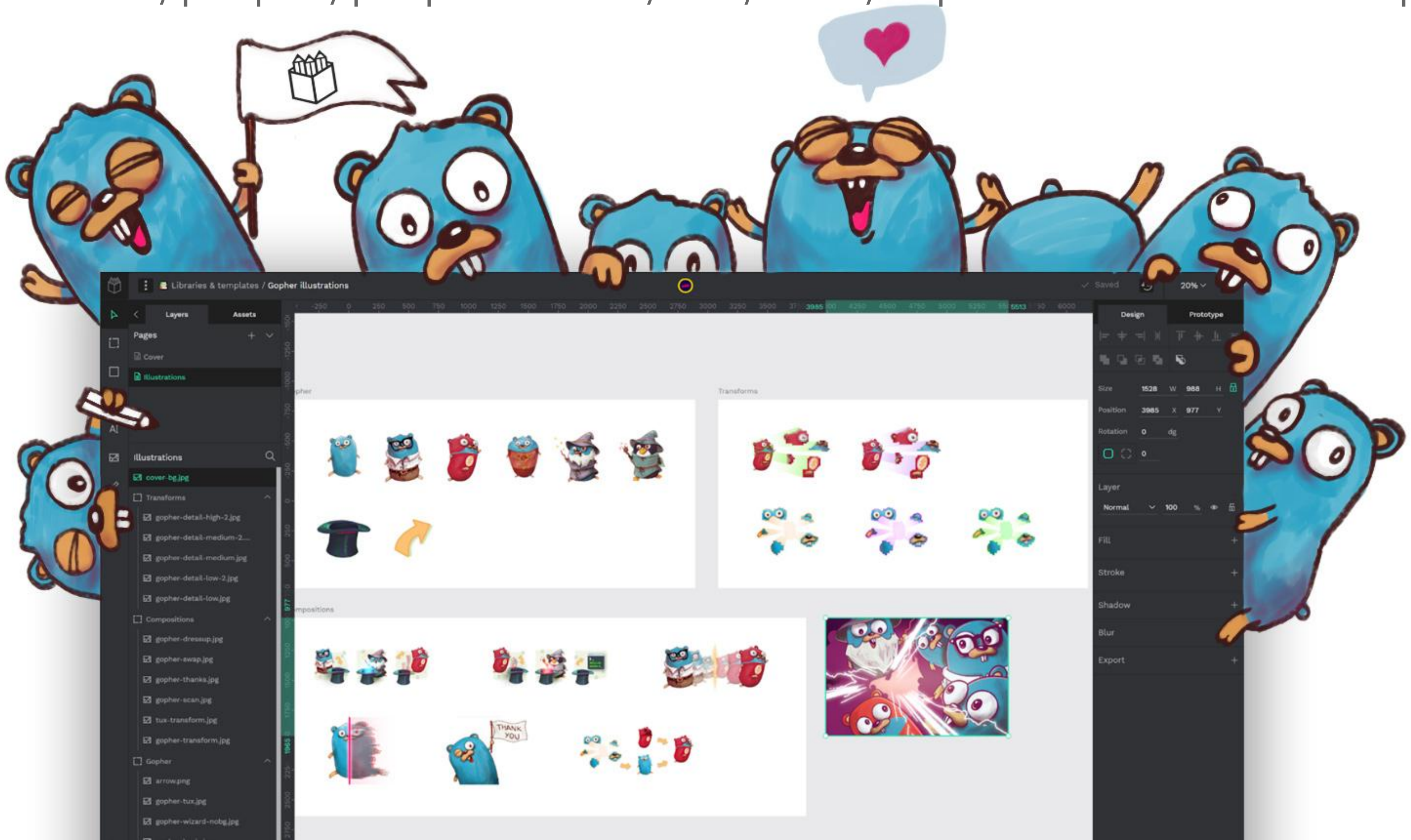
Summary

```
0x0077 00119 (.../main.go:5)  CALL runtime.growslice(SB)
0x00e3 00227 (.../main.go:7)  CALL runtime.fastrand(SB)
0x010a 00266 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0131 00305 (.../main.go:7)  CALL runtime.mapassign_faststr(SB)
0x0158 00344 (.../main.go:8)  CALL runtime.mapassign_faststr(SB)
0x016d 00365 (.../main.go:10) CALL runtime.makechan(SB)
0x0199 00409 (.../main.go:13) CALL runtime.chansend1(SB)
0x01ea 00490 (.../main.go:16) CALL runtime.mapiterinit(SB)
0x020e 00526 (.../main.go:17) CALL runtime.chansend1(SB)
0x0220 00544 (.../main.go:16) CALL runtime.mapiternext(SB)
0x0239 00569 (.../main.go:3)  CALL runtime.morestack_noctxt(SB)
```

Generated with: go build -gcflags=-S .

The Illustrations of the Talk

- Made by Juan de la Cruz for this talk
- Creative Commons 0 (Use it however you want)
- Downloadable in Penpot (Open Source Design tool) format
- https://github.com/penpot/penpot-files/raw/main/Gopher-illustrations.penpot



A Gift from Mattermost

References

- Memory Allocator: https://medium.com/@ankur_anand/a-visual-guide-to-golang-memory-allocator-from-ground-up-e132258453ed
- The Garbage collector (Maya Rosecrance): https://youtu.be/gPxFOMuhnUU?si=O9pn99sLiqptgyw3
- The GC Pacer (Madhav Jivrajani): https://youtu.be/We-8RSk4eZA?si=QNXxqq2xVEoh9At9
- The memory allocator (Andre Carvalho): https://youtu.be/3CR4UNMK_Is?si=B0bUKHohbNq73t7V
- The netpoll (Cindy Sridharan): https://youtu.be/xwlo3xigknI?si=dmTrK_CH_fa0Bs51
- The scheduler (Madhav Jivrajani): https://youtu.be/wQpC99Xu1U4?si=uOu0RiLyMpNXKYa0
- Other related talks from myself:
  - The go compiler: https://youtu.be/qnmoAA0WRgE?si=ANt-Mvm4hpR9Vydx
  - About goroutines: https://youtu.be/MYtUOOizITs?si=FVGFtez2z3fNCjx7

CONCLUSIONS

THANK
YOU

Let's Keep in touch

jespinog

jespino

jesus-espino