Go basics

About Me

Siddhartha Varma

- SDE Intern - backend, Core Team
- Cinephile (Currently watching: Stranger Things S4)

BRO3886

sidv.dev

Less is (exponentially) more

Rob Pike

Why Go

- Created by Rob Pike, Robert Griesemer, Ken Thompson
- Replacement of C++, at Google
- Plagued by slow compilation, not impressed by unecessary features
- need of *concurrency* on fingertips (imagine scale of Google)
- Less boilerplate code (not exactly object-oriented)
- Less is (exponentially) more

Be more expressive with less

Only 25 keywords (+1 after go1.18)

Go

- Fast (compiled to native code)
- Concurrency
- Simple
- Easy to learn
- Easy to write "good" code which needs to be maintained by large teams (set of default rules)

Basics of Go

go.dev/play

main.go

```go
package main

func main() {
}
```

Imports and fmt

```go
package main

import "fmt"

func main() {
    fmt.Println("Hello, World!")
}
```

Factored import statement and exported variables

```go
package main

import (
    "fmt"
    "math"
)

func main() {
    fmt.Println("I got %d problems", math.Sqrt(9801))
    fmt.Println("value of pi: ", math.pi) //wrong, should be math.Pi
}
```

Functions

```go
func add(x int, y int) int {
    return x + y
}

func subtract(x, y int) int {
    return x - y
}

func swap(x, y string) (string, string) {
    return y, x
}

func main() {
    a, b := swap("hello", "world")
    fmt.Println(a, b)
}
```

Variables

```go
func main() {
    var b string
    b = "hello"

    // type inference, shorthand
    a := 10
}
```

Zero values

```
var a bool // false

var b int // 0

var c float64 // 0

var c string // ""
```

Loops

```go
sum := 0
for i := 0; i < 10; i++ {
    sum += i
}
fmt.Println(sum)

sum := 1
for sum < 1000 {
    sum += sum
}
fmt.Println(sum)

for  {
    // do something
    // runs indefinitely
}
```

If-else

```go
if 7%2 == 0 {
    fmt.Println("7 is even")
} else {
    fmt.Println("7 is odd")
}
```

Switch statements

```go
package main

import(
    "fmt"
    "time"
)

func main() {
    i := 2
    fmt.Print("Write ", i, " as ")
    switch i {
    case 1:
        fmt.Println("one")
    case 2:
        fmt.Println("two")
    case 3:
        fmt.Println("three")
    } // prints "Write 2 as two"

    switch time.Now().Weekday() {
    case time.Saturday, time.Sunday:
        fmt.Println("It's the weekend")
    default:
        fmt.Println("It's a weekday")
    }
}
```

Pointers

```
package main

import "fmt"

func main() {
    i, j := 42, 2701

    p := &i         // point to i
    fmt.Println(*p) // read i through the pointer
    *p = 21         // set i through the pointer
    fmt.Println(i)  // see the new value of i

    p = &j         // point to j
    *p = *p / 37   // divide j through the pointer
    fmt.Println(j) // see the new value of j
}
```

Structs

Remeber C?

```go
package main

import "fmt"

type Person struct {
    Firstname string
    Lastname string
    Age int
}

func main() {
    p := Person{"Siddhartha", "Varma", 22}
    fmt.Println(p.Firstname) // prints "Siddhartha"
}
```

Arrays and slices

```go
package main

import "fmt"

func main() {
    var a [2]string
    a[0] = "Hello"
    a[1] = "World"
    fmt.Println(a[0], a[1])
    fmt.Println(a)

    var primes []int
    primes = append(primes, 2, 3, 5, 7, 11)
    fmt.Println(primes)
    primes = append(primes, 13, 17, 19, 23, 29)
    fmt.Println(primes)
}
```

Questions?

Do explore

concurrency, channels, interface, and goroutines

Resources

- go.dev/doc/
- go.dev/blog/
- dave.cheney.net/
- github.com/avelino/awesome-go
- youtube.com/c/GolangDojo
- youtube.com/c/Tutorialedge/
- Creating web applications with Go - Mike Van Sickle
- Go - The Complete Developer’s Guide - Stephen Grider

These slides are available online at

talks.sidv.dev/2022/go-basics

Thank You!