LESSON 3

main.go

```go
for i, c := range cities {
  fmt.Println(i, c)
}
```

# Go

## if · for · switch

Learn it all @MyWebUniversity




A FREE MYWEBUNIVERSITY COURSE

# GO PROGRAMMING

LESSON 3

## Control Flow

if with init, for (the only loop), range, switch and defer

# if, with an Init Statement



if.go

```
if score >= 90 {
    fmt.Println("A")
} else if score >= 75 {
    fmt.Println("B")
}

if n, err := strconv.Atoi(input); err != nil {
    fmt.Println("not a number")
} else {
    fmt.Println(n * 2)
}
```

- Go's if needs no parentheses but always needs braces. It can start with a short statement, …
- This is the classic Go pattern: if err colon equals do something, semicolon, err not equal …

01

# for Is the Only Loop



for.go

```go
for i := 0; i < 3; i++ { fmt.Println(i) }   // classic

n := 1
for n < 100 { n *= 2 }                      // like while

for {                                       // forever
    if done() { break }
}
```

- Go has just one loop, for, in three shapes: the classic counter, a while-style loop with …
- break and continue work as usual, and labels let you break out of nested loops.

02

# range



range.go

```
cities := []string{"Kabul", "Herat", "Balkh"}
for i, c := range cities {
    fmt.Println(i, c)
}
for _, c := range cities { fmt.Println(c) }
for i := range 5 { fmt.Print(i) }              // Go 1.22+: 01234
```

- for with range walks over slices, maps, strings and channels, giving the index and the ...
- Use the blank identifier, an underscore, when you do not need the index.

03

# switch



switch.go

```go
switch day {
case "Friday":
    fmt.Println("weekend")
case "Saturday", "Sunday":
    fmt.Println("work day")
}

switch {                        // no value: if-else chain
case score >= 90: grade = "A"
default:          grade = "B"
}
```

- Go's switch needs no break: each case stops automatically. A case can list several values, …
- Use fallthrough in the rare case you want to continue into the next case.

04

# defer



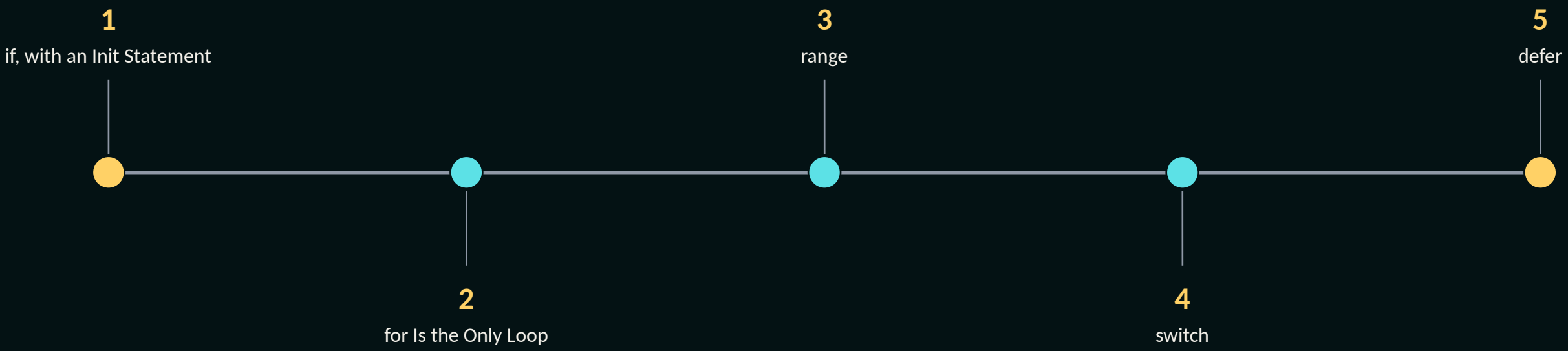
defer.go

```go
func readConfig() error {
    f, err := os.Open("config.json")
    if err != nil {
        return err
    }
    defer f.Close()          // runs when readConfig returns
    // ... read the file ...
    return nil
}
```

- defer schedules a function call to run when the surrounding function returns, no matter how …
- It is perfect for cleanup: open a file, then immediately defer its Close. Several defers …

05

# Lesson Map


1
if, with an Init Statement
2
for Is the Only Loop
3
range
4
switch
5
defer

# Sources and Further Reading



if.go

```go
if score >= 90 {
    fmt.Println("A")
} else if score >= 75 {
    fmt.Println("B")
}

if n, err := strconv.Atoi(input); err != nil {
    fmt.Println("not a number")
} else {
    fmt.Println(n * 2)
}
```

GO PROGRAMMING · 3

range.go

```go
cities := []string{"Kabul", "Herat", "Balkh"}
for i, c := range cities {
    fmt.Println(i, c)
}
for _, c := range cities { fmt.Println(c) }
for i := range 5 { fmt.Print(i) }            // Go 1.22+: 01234
```

Thank you for watching

# MyWebUniversity.com

*Free technical education since 2000 — English and Dari*

@MyWebUniversity · @MyWebUniversity-Dari