Go concurrency

WASA: Web and Software Architecture

Enrico Bassetti

Goroutines

A common error

Concurrency is not Parallelism

“Concurrency is about *dealing with* lots of things at once. Parallelism is about *doing* lots of things at once.”

“[...] The goal of concurrency is good structure.”

Concurrency in Go

Go makes it simple to create concurrency in programs.

It *might* execute things in parallel.

Concurrency in Go

```go
func main() {
  var j = 0
  for j < 10 {
    fmt.Println(j)
    j++
  }
}
```

Concurrency in Go

```go
func main() {
  go func() {
    var i = 0
    for i < 10 {
      fmt.Println(i)
      i++
    }
  }()

  var j = 0
  for j < 10 {
    fmt.Println(j)
    j++
  }
}
```

Channels

```go
func main() {
  var channel = make(chan int)
  go func() {
    var i = 0
    for i < 10 {
      channel <- i
      i++
    }
  }()
  var j = 0
  for j < 10 {
    fmt.Println(<-channel)
    j++
  }
}
```

Buffered Channels

```go
func main() {
  var channel = make(chan int, 2)
  go func() {
    var i = 0
    for i < 10 {
      channel <- i
      i++
    }
  }()
  var j = 0
  for j < 10 {
    fmt.Println(<-channel)
    j++
  }
}
```

Select

```go
func main() {
  var chan1 = make(chan int, 2)
  var chan2 = make(chan int, 2)
  var chan3 = make(chan int, 2)
  // ...
  select {
    case v1 := <-chan1:
      fmt.Printf("Received %v from channel 1\n", v1)
    case v2 := <-chan2:
      fmt.Printf("Received %v from channel 2\n", v1)
    case chan3 <- 1:
      fmt.Printf("Sent value to channel 3\n")
    default:
      fmt.Printf("No one is ready to communicate\n", v1)
  }
}
```

Timeout

```go
func main() {
  var chan1 = make(chan int, 2)
  var timeout = time.After(5 * time.Second)

  // ...
  select {
    case v1 := <-chan1:
      fmt.Printf("Received %v from channel 1\n", v1)
    case <-timeout::
      fmt.Printf("Timeout\n")
  }
}
```

sync.Mutex

```go
func main() {
  var mu sync.Mutex

  mu.Lock()
  mu.Unlock()
}
```

sync.Mutex

```go
var mu sync.Mutex
var idx int

func Increment() {
  mu.Lock()
  defer mu.Unlock()
  idx++
}
```

Links

- https://go.dev/blog/waza-talk