

State of the gopher

Felix Morgner

September 26, 2016

Coredump Lightning Talks 2016

Agenda

What I'll talk about

Disclaimer

interface{}

Overloading

Pointers

Garbage

Summary

Disclaimer

The views and opinions expressed in this ...

~~The views and opinions expressed in this ...~~

DO NOT CODE IN GO

Also, I'm one of those guys...



interface{

interface{} EVERYWHERE

```
package main

import "fmt"

func do_stuff(par interface{}) {
    fmt.Println(par)
}

func main() {
    do_stuff(123)
}
```

What does this remind us of?

What does this remind us of?



```
class Hello {  
    static void do_stuff(Object par) {  
        System.out.println(par);  
    }  
  
    public static void main(String ... args) {  
        do_stuff(123);  
    }  
}
```

Overloading

Function overloads are for the weak

```
for i, p := range args {  
    switch i {  
    case 0:  
        spar, ok = p.(string)  
  
        if !ok {  
            ipar, ok = p.(int)  
            val = ipar  
        } else {  
            val = spar  
        }  
  
        if !ok {  
            panic("1st parameter not type string or int")  
        }  
  
    default:  
        panic("Wrong parameter count.")  
    }  
}  
  
fmt.Println(val)
```

Because this is way too hard

```
#include <string>
#include <array>
#include <iostream>

void do_stuff(int param) {
    std::cout << param << '\n';
}

void do_stuff(std::string param) {
    std::cout << param << '\n';
}

int main() {
    do_stuff(42);
    do_stuff("nogo");
    do_stuff(std::array<char, 4>{{'n', 'o', 'g', 'o'}});
}
```

Pointers

Let's talk unicorns



Sadly, no unicorns in go. But POINTERS EVERYWHERE

```
package main

func take_ptr(par *int) {
    // ...
}

func ret_ptr() *int {
    answer := 42
    return &answer
}
```

Wait! What!?!

```
func ret_ptr() *int {  
    answer := 42  
    return &answer  
}
```

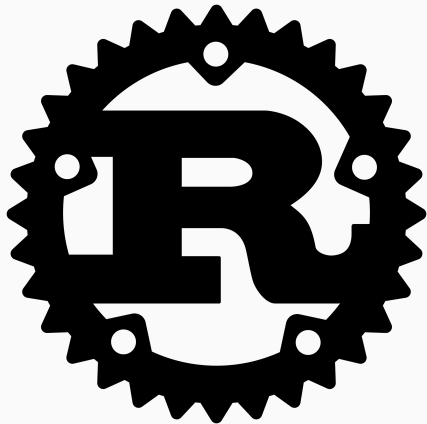
Garbage

This is the only garbage collector

}

Summary

Join the resistance!



C++ Meetup 28.09.2016 @ ETH