



Introduction to Go

Kevin W. Gisi

Twin Cities Code Camp 8—April 10, 2010

Summary

Go is a brand-spanking-new systems language that Google released in November, 2009. Every wonder how awesome C would be if it was garbage-collected, concurrent, and didn't take a few weeks to compile? Wake up; it's here! We'll take a look at this new language that steals some of the dynamic flexibility of Python and Ruby, the performance of C, and a compile time that you'll miss if you blink.

Why Go?

- It's a systems language
- It's fun, like dynamic languages

We Already Have a Systems Language!

Like C

```

void      (int m, int t, int
c) {
    ((t / m) <= 1) ?
primes(m,t+1,c) : !(t % m) ?
primes(m,t+1, t % m) :
    ((t % m)==(t / m) && !c) ?
    (printf("%d\t", (t / m)),
primes(m,t+1,c)) :
    ((t % m)> 1 && (t % m) < (t /
m)) ? primes(m,t+1,c + !((t /
m) % (t % m))) :
    (t < m * m) ? primes(m,t+1,c)
: 0;
}

```

code/c.c

We Already Have Fun Languages!

```

module
    def      (      *      )
        args.      ? cmd.      :
    "#{cmd} #{args.      (" ")}"
    end
end

puts this is terrible code

```

code/ruby.rb

Hello, world

```

package main

import "fmt"

func      () {
    fmt.Printf("Hello, world\n")
}

```

code/hello_world.go

Specifications

- Compiled
- Imperative, structured
- Concurrent
- Strongly typed (explicit or inferred)

Variables & Types

- int, float
- int8, int32, float64
- uint, ufloat
- string

Variables: Pointers and Arrays

Pointers

- [TODO]

Arrays

- NOT pointers
- Referenced via slices

Variables: Slices and Maps

Slices

Maps

```
package main

import "fmt"

func      () {
    m := map[string] int{}
    m["price"] = 5
}
```

code/maps.go

Variable Declaration

```
// Declare a variable
var s string = "";

// Go infers the type
var s = "";

// Syntactic shorthand -
// initializing declaration
s := "";
```

code/declaring_variables.go

Methods

- Pass by value
- Multiple return values

Concurrency



Goroutines

Channels
