如何用 Golang 幫 Ruby 加速

12 Sep 2015

tka

self.inspect

tka

Web / RoR / Ruby / Golang / Linux

twitter: tkalu

github: tka

blog: blog.tka.lu

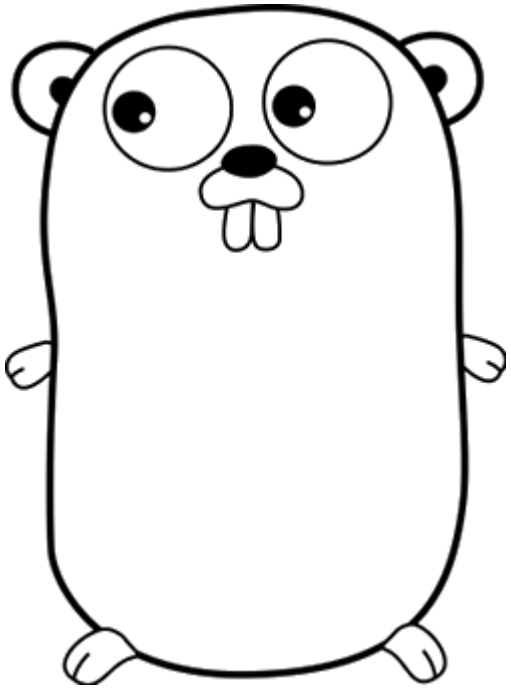
Topic

1. golang shared library 的簡介
2. 使用 ruby-ffi 呼叫 golang 的方法
3. 比較貼近實際使用的例子

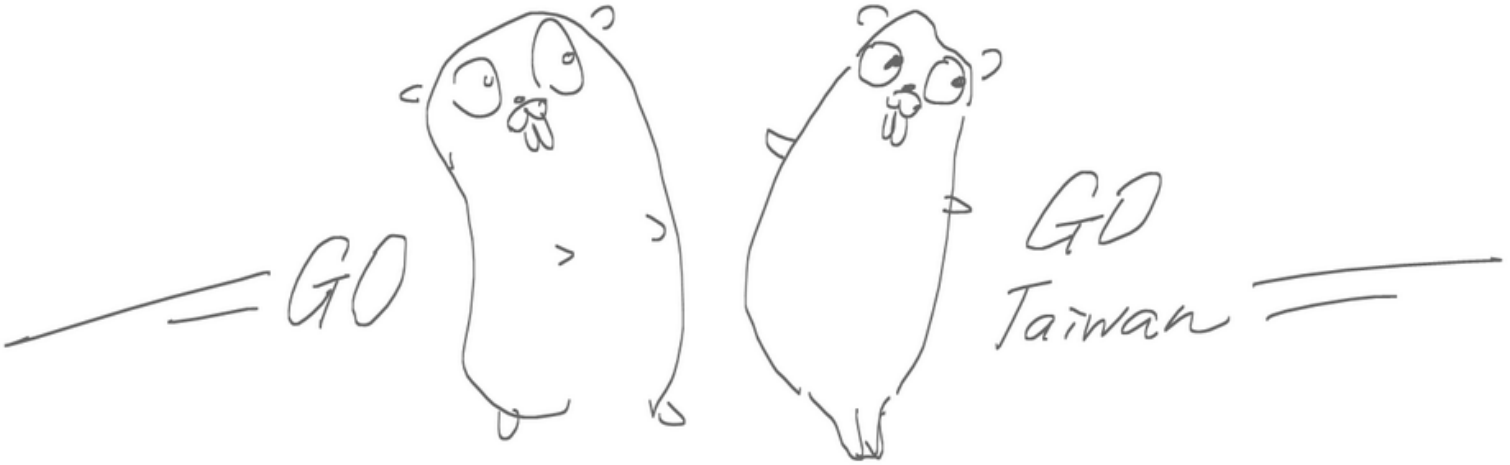
Golang

1. Static
2. Strong Typing
3. Compile
4. GC

Gopher



Gopher



by @ETBlue

Now

- Golang 1.5 2015/08/19
- Golang 1.5.1 2015/09/08
- Download

Shared Library

```
go help build

-buildmode=c-shared

Build the listed main packages, plus all packages that they

import, into C shared libraries. The only callable symbols will

be those functions marked as exported. Non-main packages are

ignored.
```

#11058 should work on windows, Milestone Golang1.6

References

- BUILDING PYTHON MODULES WITH GO 1.5
- Go 1.5: Calling Go shared libraries from Firefox addons
- Go and Ruby-FFI
- Ruby and Go Sitting in a Tree

Golang Shared Library Example

```
package main

import ( "C" )

// 這邊要特別指定
//export add
func add(x int, y int) int {
    return x + y
}

func main() { }
```

Compile

```
go build -buildmode=c-shared -o libgo_example_1.so libgo_example_1.go
```

libgo_example_1.h

```
typedef signed char GoInt8;
typedef unsigned char GoUint8;
typedef short GoInt16;
typedef unsigned short GoUint16;
typedef int GoInt32;
typedef unsigned int GoUint32;
typedef long long GoInt64;
typedef unsigned long long GoUint64;
typedef GoInt64 GoInt;
typedef GoUint64 GoUint;
...
typedef struct { char *p; GoInt n; } GoString;
...
extern GoInt add(GoInt p0, GoInt p1);
```

Fiddle & Ruby-FFI

Fiddle

http://ruby-doc.org/stdlib-2.2.3/libdoc/fiddle/rdoc/Fiddle.html

Fiddle is an extension to translate a foreign function interface (FFI) with ruby. It wraps libffi, a popular C library which provides a portable interface that allows code written in one language to clal code written in another language.

Ruby-FFI

https://github.com/ffi/ffi

Ruby-FFI is a ruby extension for programmatically loading dynamic libraries, binding functions within them, and calling those functions from Ruby code. Moreover, a Ruby-FFI extension works without changes on Ruby and JRuby.

Benchmark

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/ruby-fiddle-vs-ffi.rb

```
LIB_PATH= "./libgo_example_1.so"

module FFILib
 extend FFI::Library
 ffi_lib LIB_PATH
 attach_function :add, [ :int, :int ], :int
end

FFILib.add(1,1)

lib = Fiddle.dlopen(LIB_PATH)
fiddle_add = Fiddle::Function.new(
 lib['add'],  [Fiddle::TYPE_INT, Fiddle::TYPE_INT],  Fiddle::TYPE_INT
)
fiddle_add.call(1,1)
```

Benchmark

```
                user     system      total        real
fiddle      1.860000   0.280000   2.140000 (  2.131172)
ruby-ffi    0.970000   0.260000   1.230000 (  1.227431)
```

選 Ruby-FFI 就對了

1. 速度快
   a. 快了約 45 %
2. 文件完整
   a. https://github.com/ffi/ffi/wiki Ruby-FFI wiki

Ruby-FFI Example

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_1.rb

```
require 'ffi'
module Go
 extend FFI::Library
 ffi_lib "./libgo_example_1.so"
 attach_function :add, [ :int, :int ], :int
end

n=1000000
Benchmark.bm do |x|
 x.report("golang") { n.times{ Go.add(1,1) }}
 x.report("ruby"){ n.times{ 1+1 }}
end
```

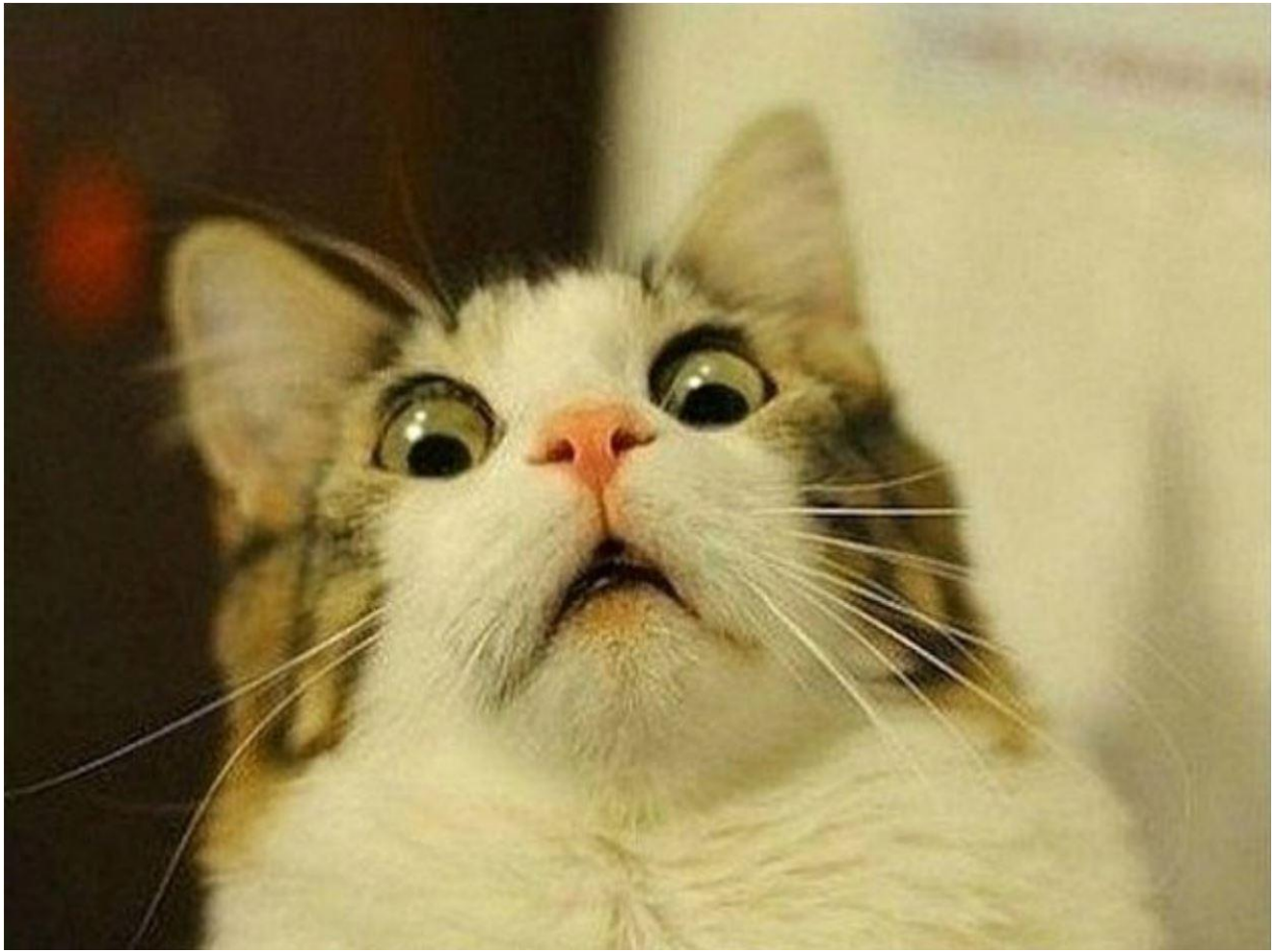
Off Topic: Python

```
import ctypes
lib = ctypes.CDLL("././libgo_example_1.so")
lib.add(1, 1)
```

Benchmark Result

```
            user     system      total        real
golang   1.130000   0.270000   1.400000 (  1.394132)
ruby     0.050000   0.000000   0.050000 (  0.050941)
```

Ruby 25x faster than Ruby-FFI+Golang



from http://imgur.com/gallery/vBRPf

Fibonacci numbers 費氏級數

0,1,1,2,3,5,8,13,21,34……

費氏級數 Ruby

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_2.rb

```ruby
module LibGo
 extend FFI::Library
 ffi_lib './libgo_example_2.so'
 attach_function :fib, [:int], :int
end

def fib(i)
 return i if i < 2
 return fib(i-2)+fib(i-1)
end

Benchmark.bm(15) do |x|
 (2..14).step(2).each do |num|
   x.report("fib #{num} golang "){ n.times{ LibGo.fib(num) }}
   x.report("fib #{num} ruby"){ n.times{ fib(num); }}
 end
end
```

費氏級數 Golang

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_2.go

```
package main

import ("C")

//export fib
func fib(n int) int {
     if n < 2 {
          return n
     }
     return fib(n-2) + fib(n-1)
}

func main() {}
```

費氏級數 Benchmark Result

```
                    user     system      total        real
-----------------------------------------------------------
fib 2 golang      0.000000   0.000000   0.000000 (  0.012477)
fib 2 ruby        0.000000   0.000000   0.000000 (  0.002036)
-----------------------------------------------------------
fib 4 golang      0.020000   0.000000   0.020000 (  0.012580)
fib 4 ruby        0.000000   0.000000   0.000000 (  0.005213)
-----------------------------------------------------------
fib 6 golang      0.010000   0.010000   0.020000 (  0.012965)
fib 6 ruby        0.020000   0.000000   0.020000 (  0.013494)
-----------------------------------------------------------
fib 8 golang      0.010000   0.000000   0.010000 (  0.014471)
fib 8 ruby        0.040000   0.000000   0.040000 (  0.034710)
-----------------------------------------------------------
fib 10 golang     0.010000   0.000000   0.010000 (  0.016872)
fib 10 ruby       0.090000   0.000000   0.090000 (  0.090370)
```

C String & Golang String

```
typedef struct { char *p; GoInt n; } GoString;
```

從 Accept-Language 比對適合的語系

- RFC2616 #14.4 Accept-Language
- Golang package
  - https://godoc.org/golang.org/x/text/language
- Ruby Gem http_accept_language
  - https://github.com/iain/http_accept_language

從 Accept-Language 比對適合的語系 Ruby

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_3.rb

```ruby
module LibGo
 class GoString < FFI::Struct
   layout :p, :pointer,
     :n, :int
 end
 extend FFI::Library
 ffi_lib './libgo_example_3.so'
 attach_function :preferredLanguageFrom, [GoString.by_ref], GoString.by_ref

 attach_function :preferredLanguageFromUseCString, [:string], :string
end
```

從 Accept-Language 比對適合的語系 Ruby

```ruby
def use_go
 x = LibGo::GoString.new
 x[:p] = FFI::MemoryPointer.from_string(@lang)
 x[:n] = @lang.length
 result =  LibGo.preferredLanguageFrom( x )
 result[:p].get_string(0, result[:n])
end

def use_go_cstring
 LibGo.preferredLanguageFromUseCString( @lang )
end

def use_ruby
 parser = HttpAcceptLanguage::Parser.new(@lang)
 parser.preferred_language_from( @availables )
end
```

從 Accept-Language 比對適合的語系 Golang

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_3.go

```
//export preferredLanguageFrom
func preferredLanguageFrom(httpAcceptLanguage *string) *string {
    tag, _, _ := language.ParseAcceptLanguage(*httpAcceptLanguage)
    t, _, _ := mather.Match(tag...)
    l := t.String()
    return &l
}
```

從 Accept-Language 比對適合的語系 Golang

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_3.go

```
//export preferredLanguageFromUseCString
func preferredLanguageFromUseCString(cHttpAcceptLanguage *C.char)
*C.char {

    httpAcceptLanguage := C.GoString(cHttpAcceptLanguage)
    tag, _, _ := language.ParseAcceptLanguage(httpAcceptLanguage)
    t, _, _ := mather.Match(tag...)
    return C.CString(t.String())
}
```

從 Accept-Language 比對適合的語系 Benchmark Result

```
                user     system      total        real
Ruby        2.550000   0.000000   2.550000 (  2.550428)
GoString    2.160000   0.070000   2.230000 (  2.083522)
CString     1.680000   0.080000   1.760000 (  1.606600)
```

Array

Array - Ruby

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_array.rb

```ruby
module Go
 extend FFI::Library
 ffi_lib "./libgo_example_array.so"
 attach_function :sum, [ :pointer, :long_long ], :long_long
end

def sum_by_go(nums)
 pointer = FFI::MemoryPointer.new :long_long, nums.length
 pointer.write_array_of_long_long(nums)
 Go.sum(pointer, nums.length)
end
```

Array - Ruby

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_array.rb

```
source = Array.new(100){ SecureRandom.random_number(1000)}
runs = 100000

Benchmark.bmbm do |x|
 x.report("go"){ runs.times{ sum_by_go(source)}}
 x.report("ruby"){ runs.times{ source.inject(0, :+) }}
```

Array - Golang

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_array.go

```
import (
     "C"
     "reflect"
     "unsafe"
)

//export sum
func sum(cArray unsafe.Pointer, length int) int {
     hdr := reflect.SliceHeader{Data: uintptr(cArray), Len: length, Cap: length}
     data := *(*[]int)(unsafe.Pointer(&hdr))
     s := 0
     for _, i := range data {
          s = s + i
     }
     return s
}
```

Turning C arrays into Go slices

Array - Benchmark Result

```
          user     system      total        real
go      0.310000   0.030000   0.340000 (  0.340010)
ruby    0.600000   0.010000   0.610000 (  0.595653)
```

Garbage collection

Garbage collection

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_gc.go

```
//export printTime
func printTime(prefix *C.char) {
     ticker := time.NewTicker(time.Millisecond * 500)
     go func() {
          for t := range ticker.C {
               fmt.Println(C.GoString(prefix), "Tick at", t)
          }
     }()
}
```

Garbage collection

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_gc.rb

```
module Go
 extend FFI::Library
 ffi_lib "./libgo_example_gc.so"
 attach_function :printTime, [ :string ], :void, :blocking=>false
end

def start_tick(tick_prefix)
 Go.printTime(tick_prefix)
end

start_tick("GC TEST PREFIX")
sleep(1)
puts "create garbage\n"
20000.times{|i| "i"*i }
sleep(1)
```

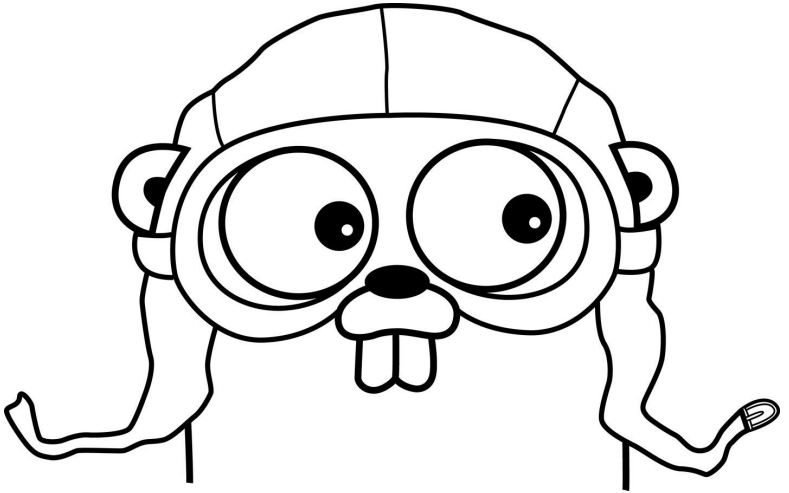
Garbage collection result

```
GC TEST PREFIX Tick at 2015-09-11 13:53:35.273543677 +0800 CST

create garbage

GC TEST PREFIX Tick at 2015-09-11 13:53:35.773574752 +0800 CST

i Tick at 2015-09-11 13:53:36.273540868 +0800 CST

i Tick at 2015-09-11 13:53:36.773521298 +0800 CST
```

https://github.com/ffi/ffi/wiki/Core-Concepts#memory-management

Ruby FFI Memory Management

Golang Taiwan



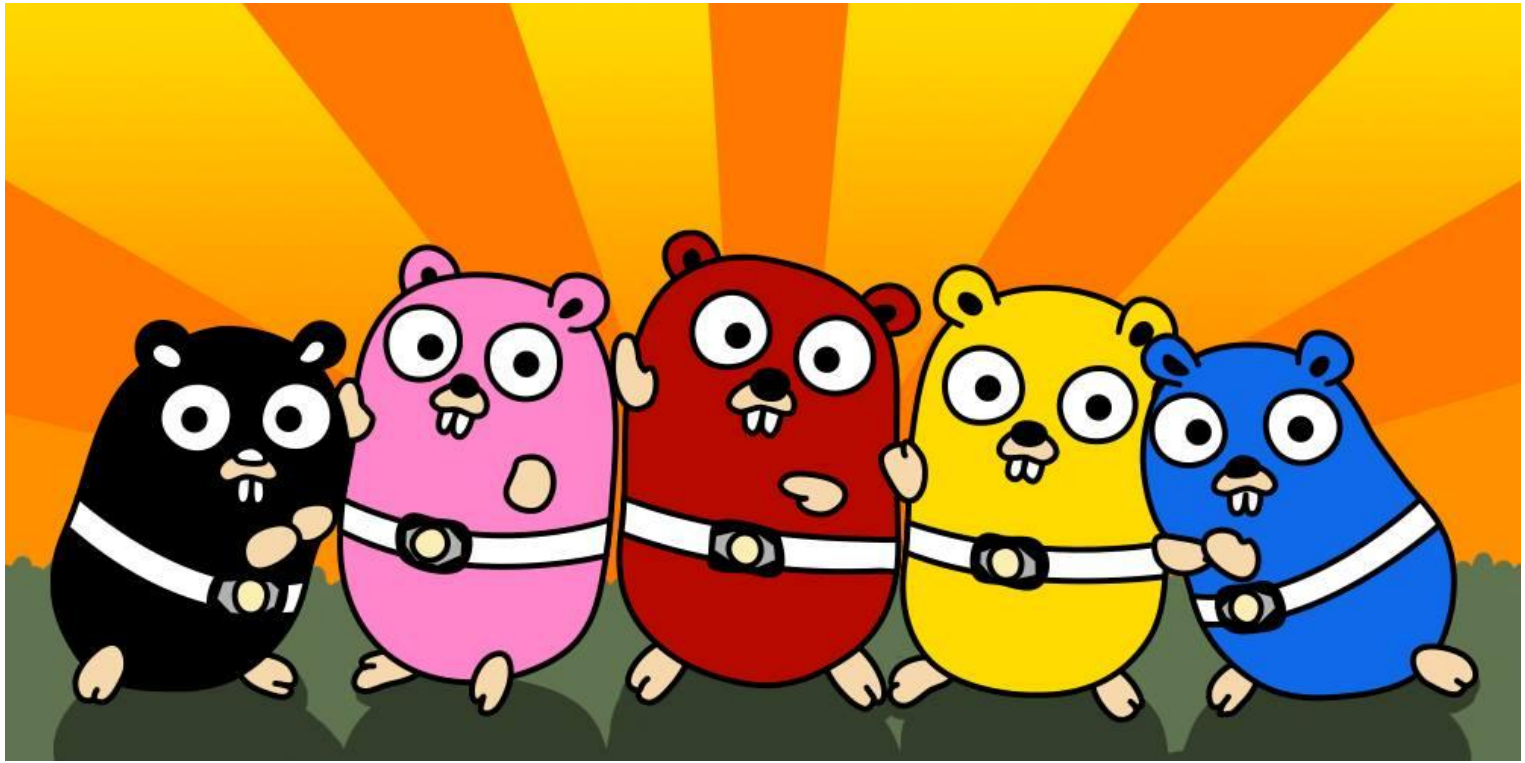
Homepage
http://golang.tw/

Twitter
https://twitter.com/golangtw

Slack
http://golangtw-slackin.herokuapp.com/

2015/10/01 **Golang Taipei Gathering #15**

Thank You



Go Go Power Rangers! art by @miau715

Bonus

Markdown

kramdown, pure ruby

github-markdown, C extenstion

redcarpet, C extenstion

blackfriday, Golang

github_flavored_markdown, Golang

Markdown - Golang

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_gfm.go

```
import (
     "C"
     "github.com/russross/blackfriday"
     "github.com/shurcooL/github_flavored_markdown"
)
//export gfm
func gfm(text *C.char) *C.char {
     mdString := C.GoString(text)
     result := github_flavored_markdown.Markdown([]byte(mdString))
     return C.CString(string(result))
}
//export md
func md(text *C.char) *C.char {
     mdString := C.GoString(text)
     result := blackfriday.MarkdownCommon([]byte(mdString))
     return C.CString(string(result))
}
```

Markdown - Ruby

https://github.com/tka/rubyconf-tw-2015-ruby-go/blob/master/libgo_example_gfm.rb

```
module Go
 extend FFI::Library
 ffi_lib "./libgo_example_gfm.so"
 attach_function :gfm, [ :string ], :string
 attach_function :md, [ :string ], :string
end

results = Benchmark.bmbm do |b|
 b.report("kramdown #{Kramdown::VERSION}") {
     RUNS.times { Kramdown::Document.new(data, input:'GFM' ).to_html }
 }
 b.report("golang-blackfriday") { RUNS.times { Go.md(data) } }
 b.report("golang-gfm") { RUNS.times { Go.gfm(data) } }
 b.report("redcarpet #{Redcarpet::VERSION}") { RUNS.times { Redcarpet::Markdown.
new(Redcarpet::Render::HTML).render(data) } }
 b.report("github-markdown"){ RUNS.times{ GitHub::Markdown.render(data) } }
end
```

Markdown - result

```
                        user     system      total        real
kramdown 1.8.0      6.810000   0.010000   6.820000 (  6.811989)
golang-blackfriday  0.260000   0.010000   0.270000 (  0.220802)
golang-gfm          1.000000   0.020000   1.020000 (  0.914868)
redcarpet 3.3.2     0.050000   0.000000   0.050000 (  0.053187)
github-markdown     0.050000   0.000000   0.050000 (  0.050833)

Real time of X divided by real time of kramdown
golang-blackfriday 0.0324
golang-gfm         0.1343
redcarpet          0.0078
github-markdown    0.0075
```