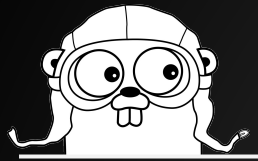
http://golang.org

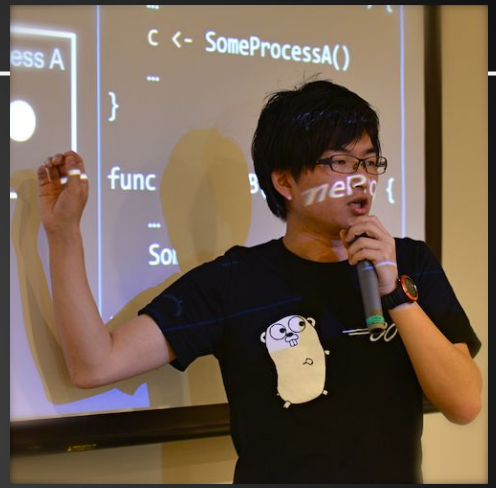
Go 1.5での新機能

Yoshifumi Yamaguchi

Aug. 11, 2015 (Tue) Go言語勉強会



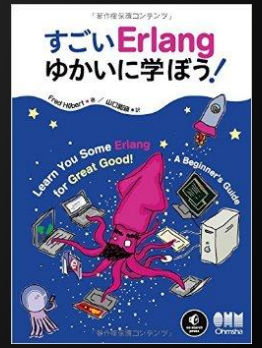
お前だれよ



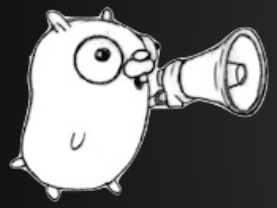
- 山口能迪（やまぐちよしふみ）
- @ymotongpoo
  とんぷーと呼ばれています

- Developer Advocate, Google
  - 新しいバージョンやサービスと連携してくれる企業の支援
  - Android Wear, Chromecast, Google Now, YouTube…

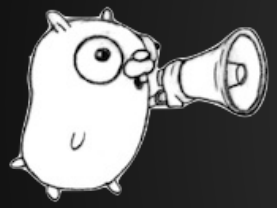
- イカの本を翻訳しました →→→→→→→→→



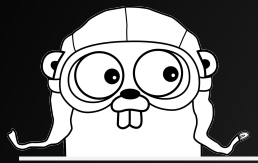

貸　切　車
8501
8501
PASMO
ワンマン
大人 160円
80円



GopherCon 2015の資料読んできた人いますか？



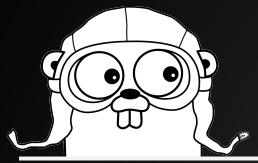
読んできた人は
一人ジャンケンを
楽しんでてください



Go 1.5

- 2015/08/10現在で 1.5 rc1
  - https://github.com/golang/go/milestones

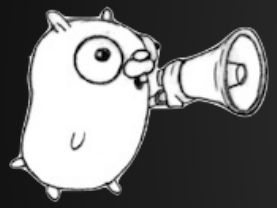
- 公式にはまだドキュメントはでてない
  - https://golang.org/doc/devel/release.html
  - なのでtip版をみましょう
    - https://tip.golang.org/doc/go1.5
    - https://youtu.be/1rZ-JorHJEY（おまけ）



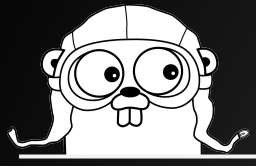
Talks/docs about Go 1.5 so far

- http://talks.golang.org/2015/state-of-go.slide
- http://talks.golang.org/2015/state-of-go-may.slide
  - by Andrew Gerrand; Go 1.5のステータス報告（2月、5月）

- http://talks.golang.org/2015/gogo.slide
  - by Rob Pike; C実装をなくした話



Highlights



これまでのリリースノート

Introduction to Go 1.4

The latest Go release, version 1.4, arrives as scheduled six months after 1.3.

It contains only one tiny langu

Introduction to Go 1.3

The latest Go release, version 1.3, arrives six months after 1.2, and contains no language changes. It focuses
refactoring of the compiler too
nance improvements across th

Introduction to Go 1.2

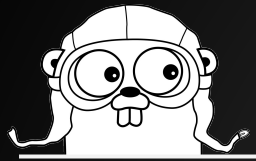
Since the release of Go version 1.1 in April, 2013, the release schedule has been shortened to make the release
process more eff
while 1.1 took ov

Introduction to Go 1.1

The release of Go version 1 (Go 1 or Go 1.0 for short) in March of 2012 introduced a new period of stability in the
sers and systems
issued. These point

Introduction to Go 1

Go version 1, Go 1 for short, defines a language and a set of core libraries that provide a stable foundation for creating reliable products, projects, and publications.

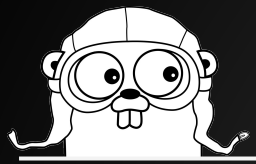


Preface of Go 1.5 Release Note

“A significant release”

Introduction to Go 1.5

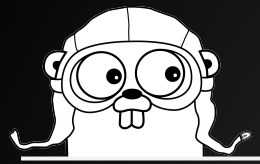
The latest Go release, version 1.5, is a significant release, includin
implementation. Despite that, we expect almost all Go programs to
because the release still maintains the Go 1 promise of compatibil



Major updates

大きく取り上げてたもの

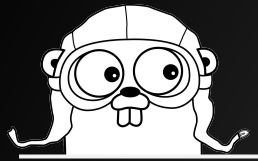
- gcとランタイムをGo自身で実装
  - C言語の実装がなくなった
- コンカレントGC
- デフォルトで GOMAXPROCS がCPU数
- internalパッケージ
- vendoringのサポート
- go tool trace
- go docでCLIむけ新機能



Major updates

リリースノートにはさりげなく書いてあったもの

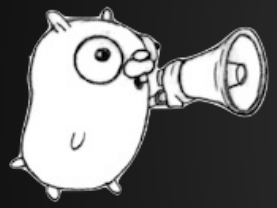
- 共有ライブラリの生成
  - `-buildmode=shared/c-shared`
- 共有ライブラリのリンク
  - `-linkshared` オプション



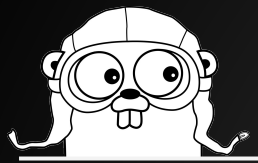
golang.org/x/mobile

- Go Mobile project
  - `golang.org/x/mobile`
  - `gomobile` コマンドからAndroid/iOSアプリの生成
    - ネイティブアプリ、ライブラリ両方の用途に（一応）使える
  - Go 1.5で共有ライブラリが作れるようになって進展

- @tenntenn が詳しい記事を書いています
  - http://klabgames.tech.blog.jp.klab.com/archives/1034818110.html
  - http://klabgames.tech.blog.jp.klab.com/archives/1036291237.html



Topics

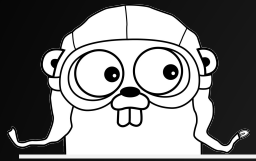


セルフホスティング

- Go 1.5のビルドにGo 1.4が必要（Go 1.4.2）

```
$ git clone https://go.googlesource.com/go go1.5
$ cd go1.5/src
$ GOROOT_BOOTSTRAP=/path/to/go1.4.2/ ./make.bash
```

- ビルド済みバイナリを使ったほうがいい
  - https://golang.org/dl/#go1.5rc1

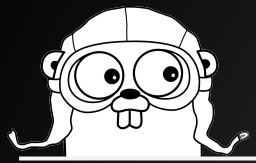


コンカレントGC

- https://goo.gl/k4mZxm
  - Goのコンパイラを書いているAustin Clementsによる資料
  - コンカレントGCのペーシングの説明

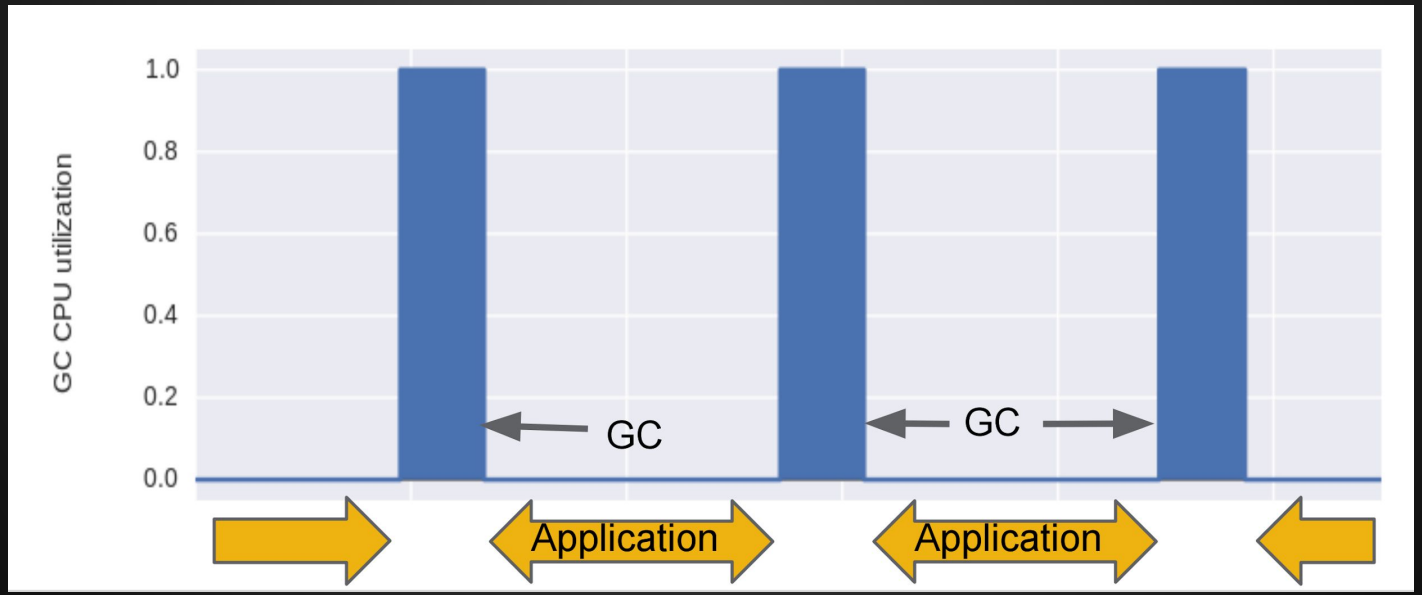
- https://talks.golang.org/2015/go-gc.pdf
  - GopherCon 2015でのRick Hudsonによる発表
  - Go 1.4と比較してのGCのパフォーマンスの評価

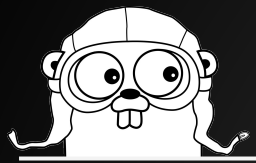
Austin Clements, Rick Hudson, Russ Coxはキューブメイト→



コンカレントGC

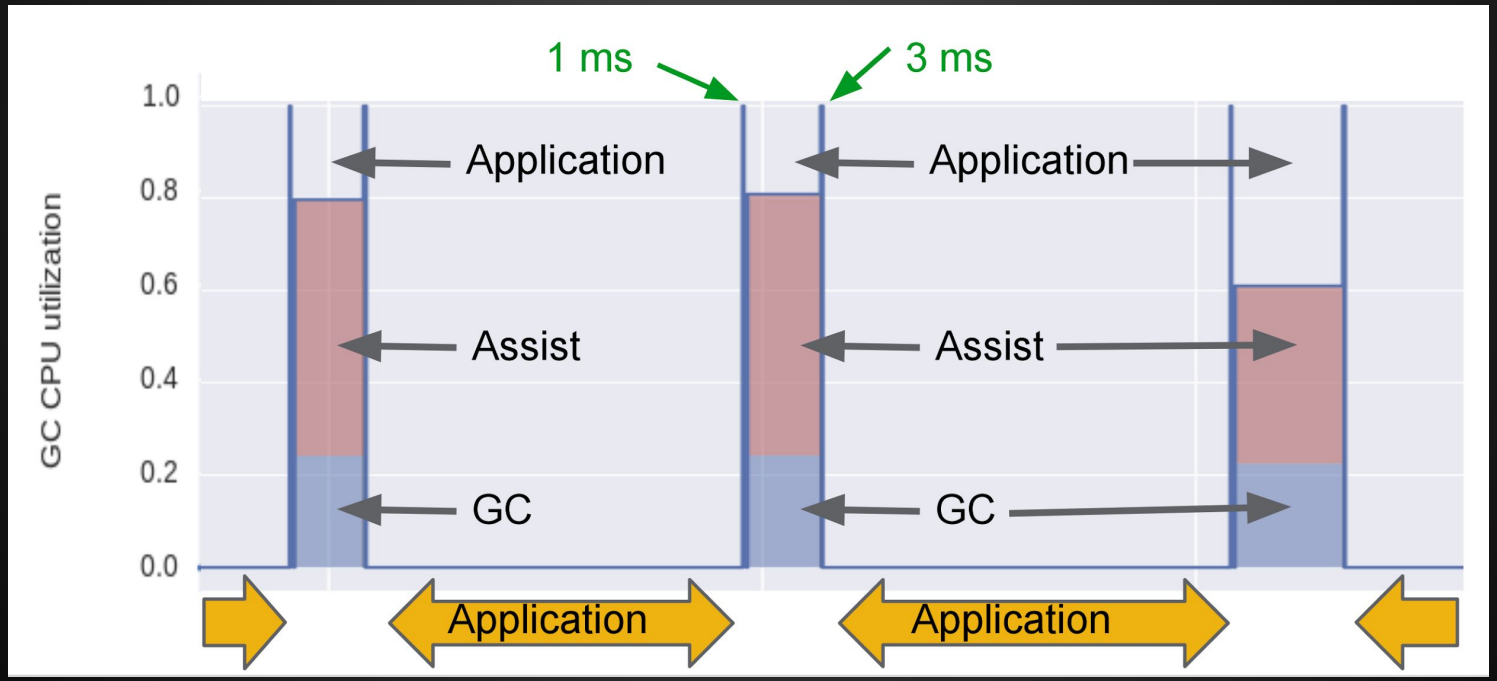
- おさらい：GoのGCは基本 mark & sweep
- Go 1.4 : stop the world

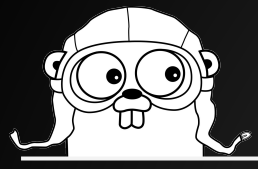




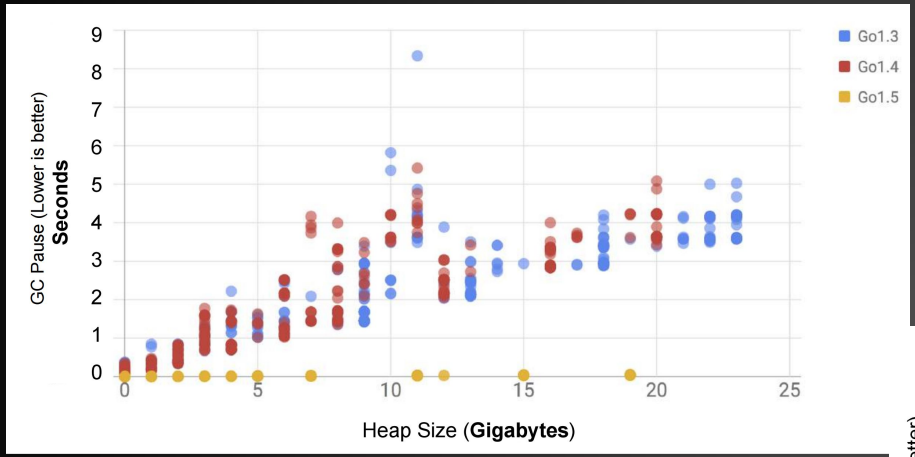
コンカレントGC

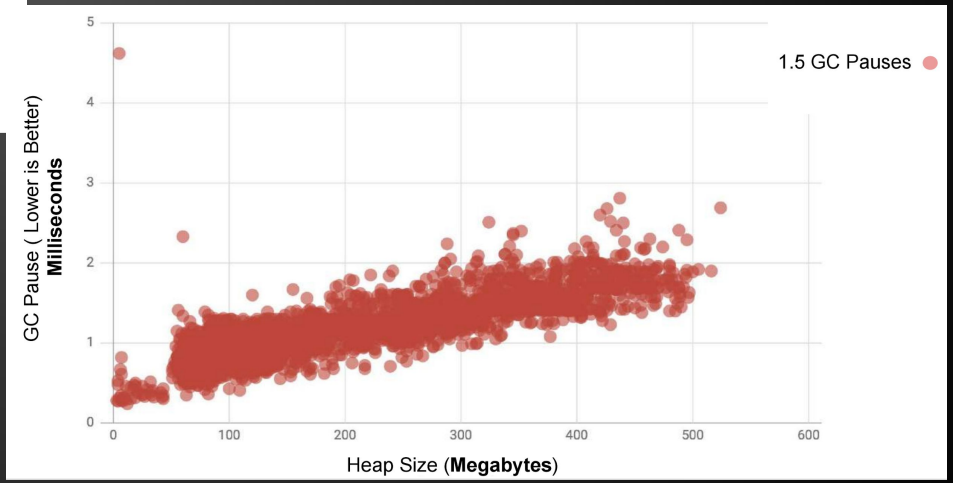
- Go 1.5 : Concurrent GC

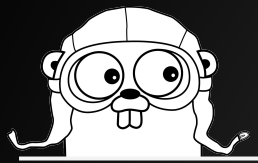




GCのパフォーマンス


GC Pause (Lower is better)
Seconds
9
8
7
6
5
4
3
2
1
0
0
5
10
15
20
25
Heap Size (Gigabytes)
Go1.3
Go1.4
Go1.5


GC Pause (Lower is Better)
Milliseconds
5
4
3
2
1
0
0
100
200
300
400
500
600
Heap Size (Megabytes)
1.5 GC Pauses



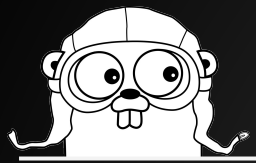
共有ライブラリの生成

C形式とGo形式の2つがある

- C形式：`-buildmode=c-shared`
- Go形式：`-buildmode=shared`

資料: https://goo.gl/COEkIJ

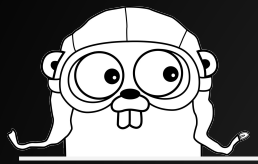
- Ian Lance TaylorによるGoで生成したバイナリの実行形式に関する資料



共有ライブラリの生成

2015/08/10時点でのGo 1.5での対応予定

	linux/amd64	linux/arm darwin/amd64 darwin/arm
c-shared	○	○
shared	○	×



C形式の共有ライブラリ

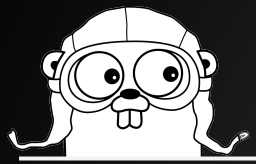
サンプル: https://goo.gl/0b5AuH

- mainパッケージのみ
- C パッケージのimport
- //export <関数名> のコメント
  - <関数名> はGoでの宣言と一致させる
- 空でいいのでmain関数を宣言

main.go

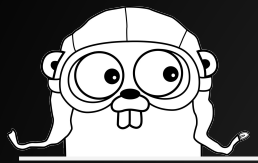
```go
package main

import (
	"C"
	"log"
	"net/http"
)

//export server
func server() {
	http.HandleFunc("/", handler)
	log.Fatal(http.ListenAndServe(":8080", nil))
}

func handler(w http.ResponseWriter, r
*http.Request) {
	w.Write([]byte("Hello, Go c-shared lib"))
}

func init() {
	log.Println("Go HTTP server is loaded")
}

// mainは必須
func main() {}
```

```
$ go build -buildmode=c-shared -o libgomain.so main.go
```



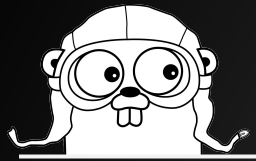
Go形式の共有ライブラリ

- `-buildmode=shared`
  - `go build -linkshared` でリンク
- Go 1.5ではlinux/amd64でのみサポート予定
  - rev:546836 では動かない



go doc

- godoc ではありません！
- godoc よりも簡潔で使いやすいCLIコマンド
- godoc はこれまでどおりHTMLドキュメントなどの生成に



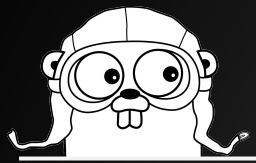
go doc

godoc

```
 % godoc net/http HandleFunc
func HandleFunc(pattern string, handler func(ResponseWriter, *Request))
    HandleFunc registers the handler function for the given pattern in the
    DefaultServeMux. The documentation for ServeMux explains how patterns
    are matched.

type ServeMux struct {
    // contains filtered or unexported fields
}
    ServeMux is an HTTP request multiplexer. It matches the URL of each
    incoming request against a list of registered patterns and calls the
    handler for the pattern that most closely matches the URL.

    Patterns name fixed, rooted paths, like "/favicon.ico", or rooted
    subtrees, like "/images/" (note the trailing slash). Longer patterns
    take precedence over shorter ones, so that if there are handlers
    registered for both "/images/" and "/images/thumbnails/", the latter
    handler will be called for paths beginning "/images/thumbnails/" and the
    former will receive requests for any other paths in the "/images/"
    subtree.

    Note that since a pattern ending in a slash names a rooted subtree, the
    pattern "/" matches all paths not matched by other registered patterns,
    not just the URL with Path == "/".

    Patterns may optionally begin with a host name, restricting matches to
    URLs on that host only. Host-specific patterns take precedence over
    general patterns, so that a handler might register for the two patterns
    "/codesearch" and "codesearch.google.com/" without also taking over
    requests for "http://www.google.com/".

    ServeMux also takes care of sanitizing the URL request path, redirecting
    any request containing . or .. elements to an equivalent .- and ..-free
```

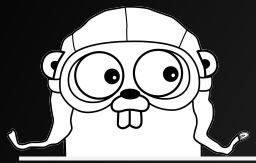
go doc

```
 % go doc net/http.HandleFunc
func HandleFunc(pattern string, handler func(ResponseWriter, *Request))

    HandleFunc registers the handler function for the given pattern in the
    DefaultServeMux. The documentation for ServeMux explains how patterns are
    matched.
```



internal

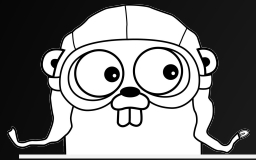
- パッケージ内に `internal` というディレクトリがあると、その中は公開しない形になる
- Go 1.4では標準ライブラリでは使えていた
- Go 1.5で3rd partyでも使えるようになった
  - 嬉しい嬉しい！！！



vendoring

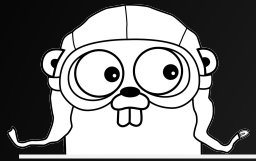
- 「Goではバージョンが固定できないんですか！？」
- 多くの議論がなされてきた
- https://golang.org/s/go15vendor
  - 開発者用ML（golang-dev）の議論を元にRuss Coxが提案
  - vendor という名前のディレクトリの下に依存するライブラリのソースを丸ごと追加して公開
  - vendor 以下がVCSクローンよりも優先される
  - 現状は実験的機能なので GO15VENDOREXPERIMENT=1 で有効化

- http://deeeet.com/writing/2015/06/26/golang-dependency-vendoring/



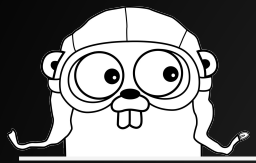
vendoring

```
$GOPATH
|    src/
|    |
github.com/constabulary/example-gsftp/
|    |    |    cmd/
|    |    |    |    gsftp/
|    |    |    |    |    main.go
|    |    |    vendor/
|    |    |    |    github.com/pkg/sftp/
|    |    |    |    golang.org/x/crypto/ssh/
|    |    |    |    |    agent/
```



Go Mobile

- https://github.com/golang/mobile
- Go 1.5が必要
- Android と iOS 向けのアプリの作成
  - all-Go app
  - SDK
- gomobileコマンドにより実施
  - tenntennの記事を読むのが一番早い
    - http://klabgames.tech.blog.jp.klab.com/archives/1034818110.html
    - http://klabgames.tech.blog.jp.klab.com/archives/1036291237.html



おしらせ

- @golangjp
- Google+ “Golang JP” コミュニティ
- https://gophers.slack.com/
  - #japan チャンネル
  - ここから招待してもらってください