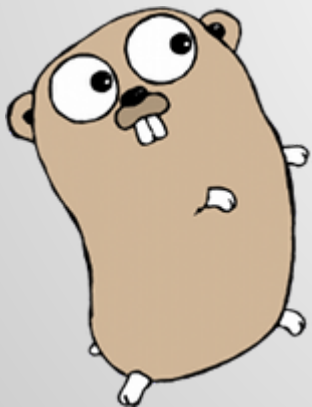


Betabeers

Go: El lenguaje de Google

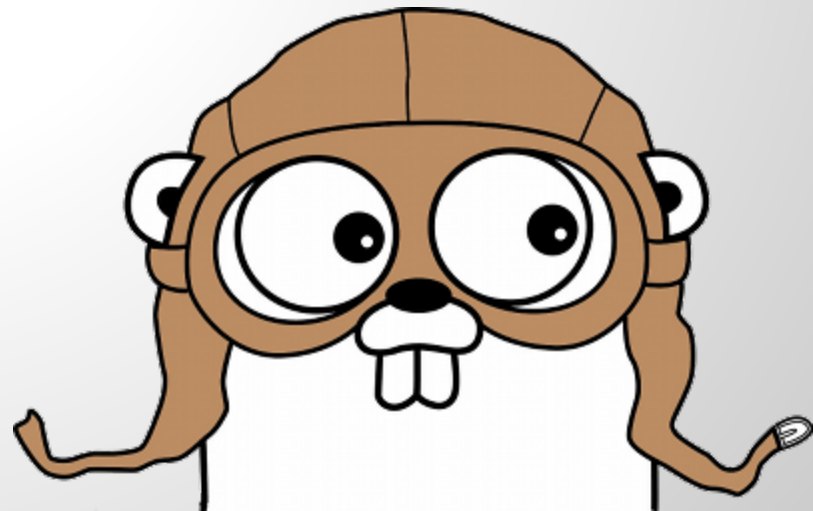
La flexibilidad de Python con la rapidez del
código máquina



@jmroble

README.txt


- ¿Por qué?
- Primeros pasos, *Go go go!*
- Vistazo rápido
- Paquetes para todo
- Marchando un *webserver!*
- ¿Quién usa esto?
- No todo es tan bonito
- ¡Me lo quedo!
- The 6W



¿Por qué?

BSD License

- Google

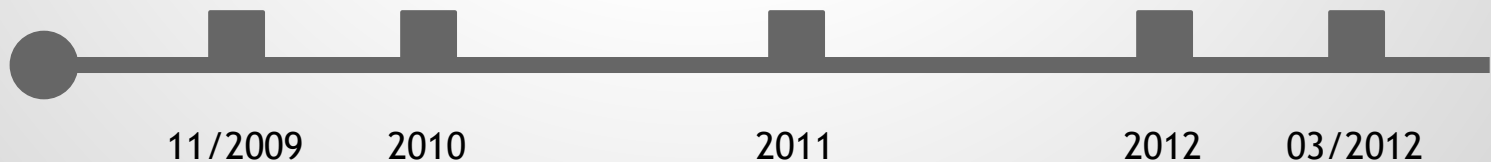
-  =   + 

simplicidad / flexibilidad / RAD

eficiencia

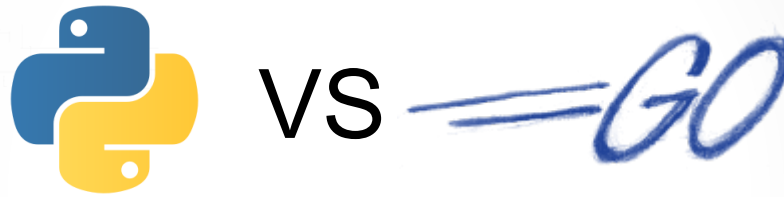
Go 0

Go 1



Primeros pasos, *Go go go!*

- Benchmark* servidor web

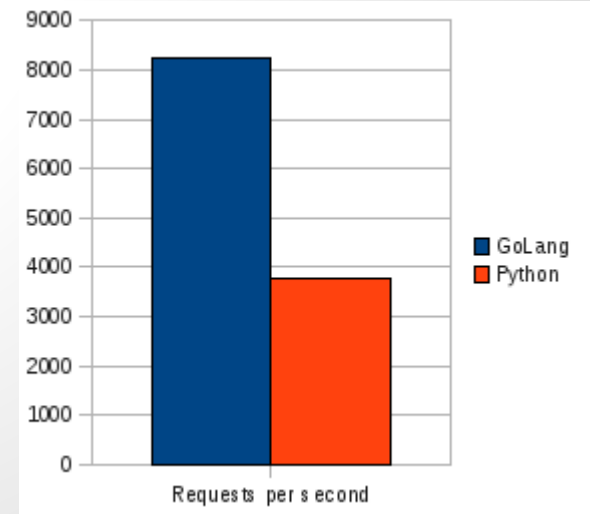


- Apache Benchmark

Go fue un **119%** más rápido que Python

```
% ab -n 40000 -c 8 http://localhost:port/
```

```
1 package main
2 import (
3     "fmt"
4     "net/http"
5 )
6 type Basic struct{}
7
8 func (b Basic) ServeHTTP(w http.ResponseWriter, r* http.Request) {
9     fmt.Fprint(w, "hola")
10 }
11 func main() {
12     var b Basic
13     http.ListenAndServe("localhost:5000", b)
14 }
```



Vistazo rápido

- "Sintaxis conocida"

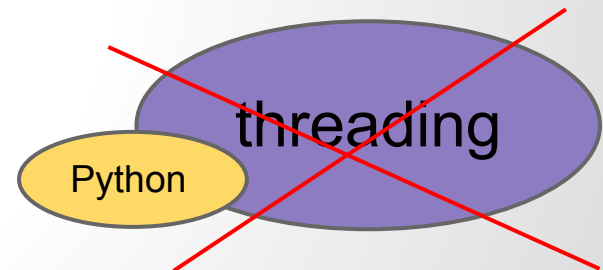
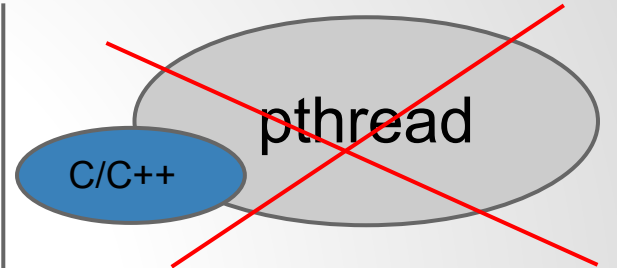
```
if , for, switch, package
```

- ... pero peculiar:

```
var i int
for i = 0; i < n ; i++ {
    fmt.Println(i)
}
```

- Go es concurrente: CSP

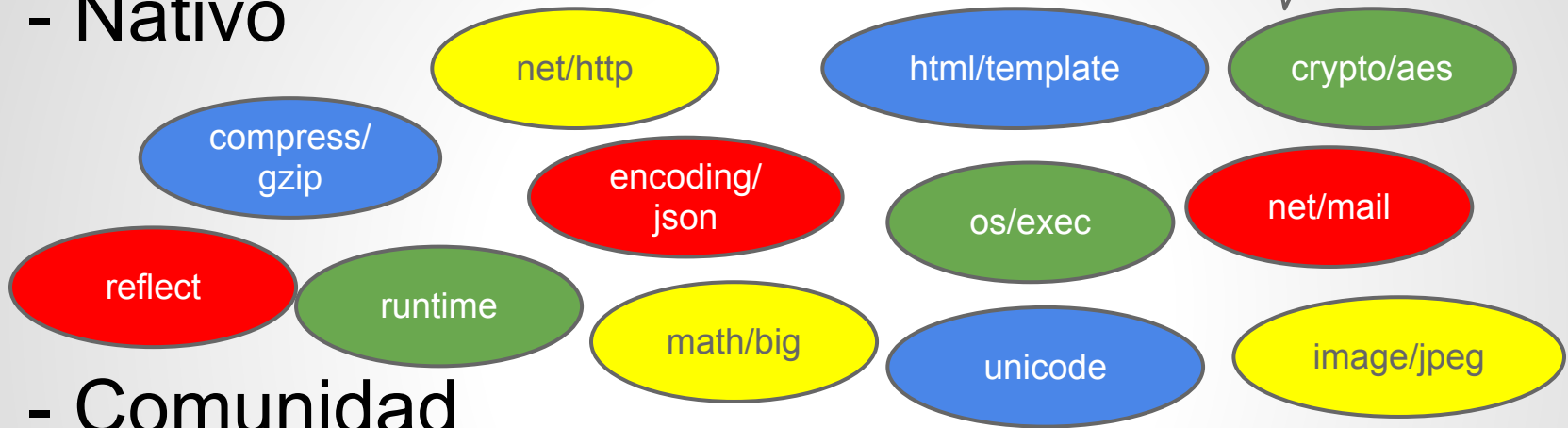
```
miCanal := make(chan string)
go func() {
    miCanal <- "desde el hilo"
}()
fmt.Println(<- miCanal)
```



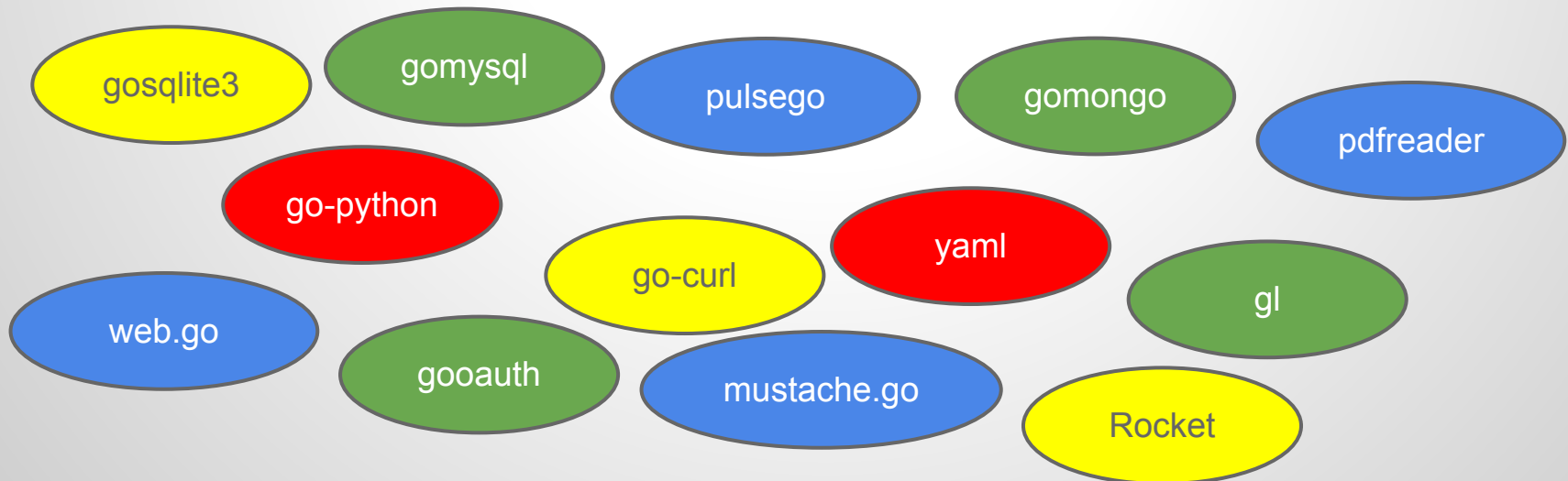
Paquetes para todo



- Nativo



- Comunidad



Marchando un *webserver*!

```
1 package main
2
3 import (
4     "fmt"
5     "encoding/json"
6     "github.com/hoisie/web"
7     "github.com/hoisie/mustache"
8     "database/sql"
9     _ "github.com/mattn/go-sqlite3"
10 )
11 type Person struct {
12     ID      int
13     Name    string
14     Cars   []string
15 }
16
17 var friend Person = Person {
18     ID:1,
19     Name:"Pepe",
20     Cars: []string{"Clio","Focus"} }
21 func handlerIndex(val string) string {
22     return mustache.RenderFile("index.html", map[string]string{"title":"Prueba"})
23 }
24 func handlerJson(ctx* web.Context) string {
25
26     ctx.ContentType("json")
27     dump, _ := json.Marshal(friend)
28     return string(dump)
29 }
30 func main() {
31     db, _ := sql.Open("sqlite3","db.dat")
32     rows, err := db.Query("SELECT version FROM version")
33     if err != nil {
34         fmt.Println(err)
35         return
36     }
37     defer rows.Close()
38     for rows.Next() {
39         var version string
40         rows.Scan(&version)
41         fmt.Println("Version: " + version)
42     }
43     web.Get("/json",handlerJson)
44     web.Get("/(.*)",handlerIndex)
45     web.Run("0.0.0.0:5000")
46 }
```

Pasos:

1. Instalar Go
(descomprimir un zip)
2. Establecer variables de entorno GOROOT y PATH
3. Abrir shell y copiar el ejemplo
4. Obtener dependencias
% go get
5. Compilar
% go build
6. ¡Voilà!

¡46 líneas!

¿Quién usa esto?

- Google

- iPubs[®]

- Stat | Hat

- Airbrake

- Canonical

- Novartis

- ...

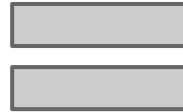
MapReduce



No todo es tan bonito

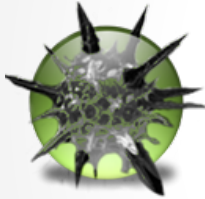
- Binarios gordos

```
1 package main
2 import (
3     "fmt"
4     "net/http"
5 )
6 type Basic struct{}
7
8 func (b Basic) ServeHTTP(w http.ResponseWriter, r* http.Request) {
9     fmt.Fprint(w, "hola")
10 }
11 func main() {
12     var b Basic
13     http.ListenAndServe("localhost:5000",b)
14 }
```



3,4 MB

- "Alergia"



- Garbage collector

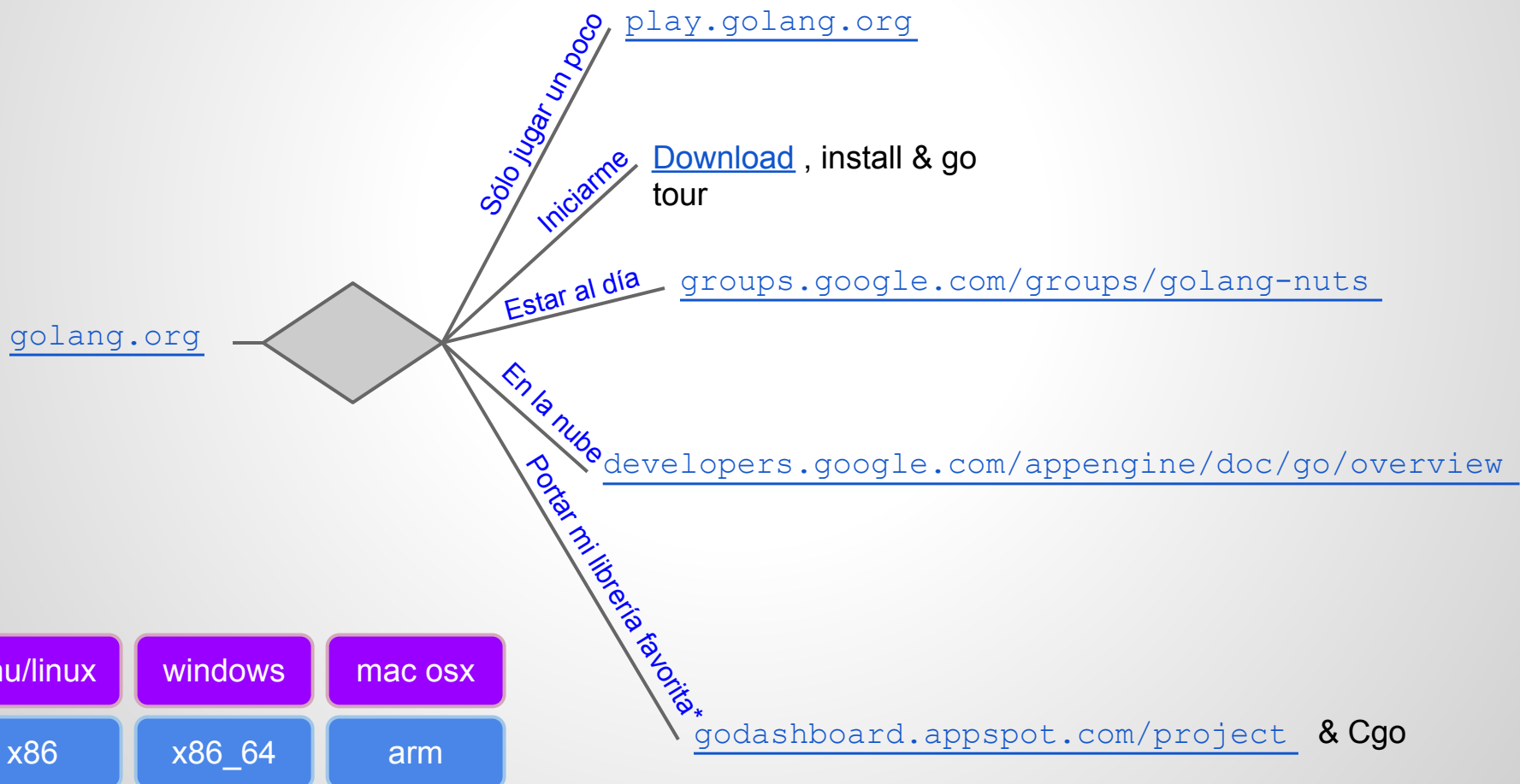


- Generics

```
interface{ }
```

¡Me lo quedo!

- Quiero empezar con Go



The 6W

- ¿Qué?

Lenguaje flexible como Python con la eficiencia de C

- ¿Quién?

Google y la comunidad de SL

- ¿Donde?

`golang.org`

- ¿Cómo?

Compilando y optimizando el código

- ¿Cuándo?

Go 1, Marzo 2012

- ¿Por qué?

Desarrollar más rápido, ejecutar con menos recursos

¡Muchas gracias a todxs!



+ Info:

@jmrobles

robleshermoso.wordpress.com

Presentación QR-disponible



Betabeers

