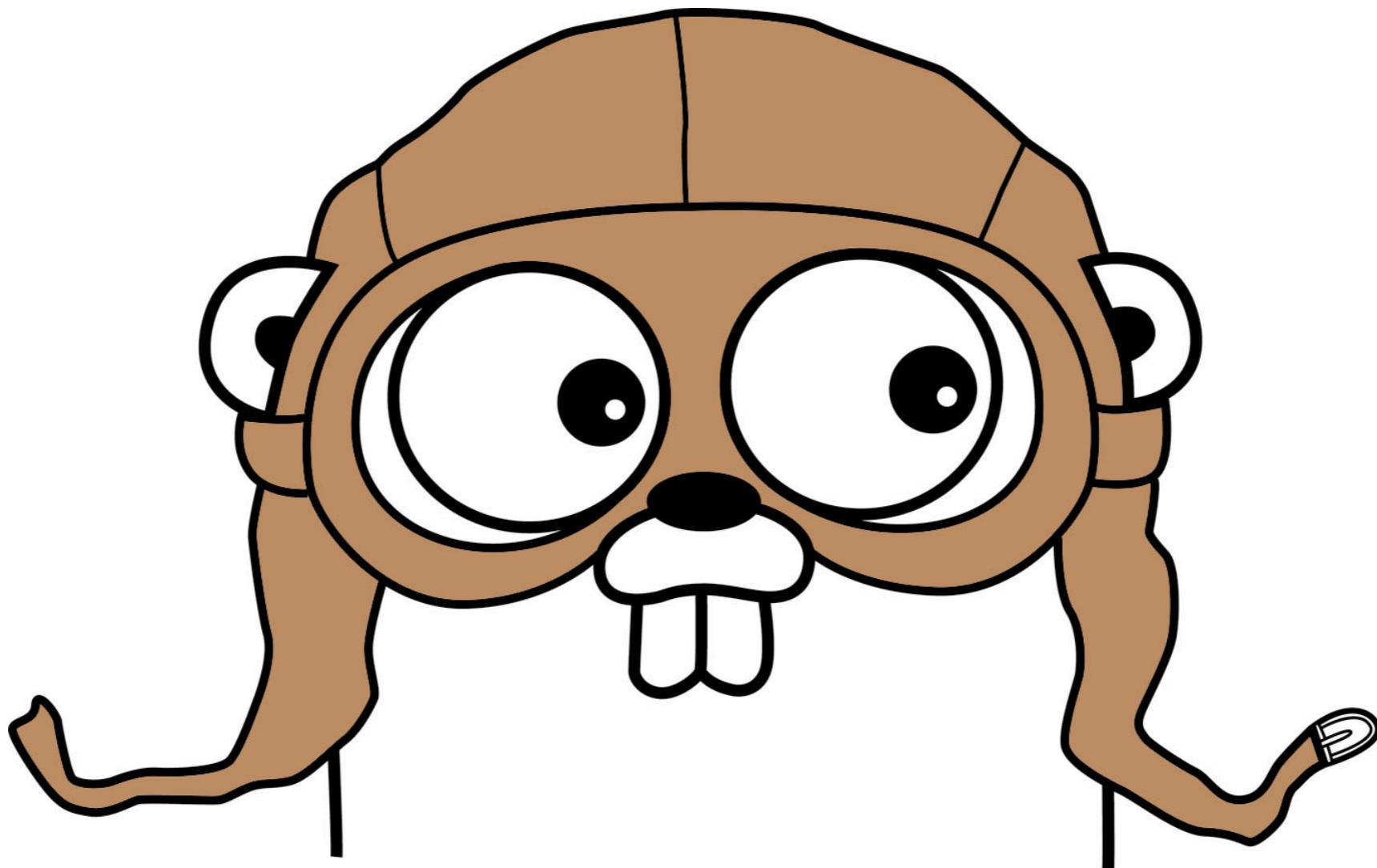


GET GO-ING WITH A NEW LANGUAGE



KAT ZIEŃ (@KASIAZIEN)
SCOTLAND PHP 6 OCTOBER 2018

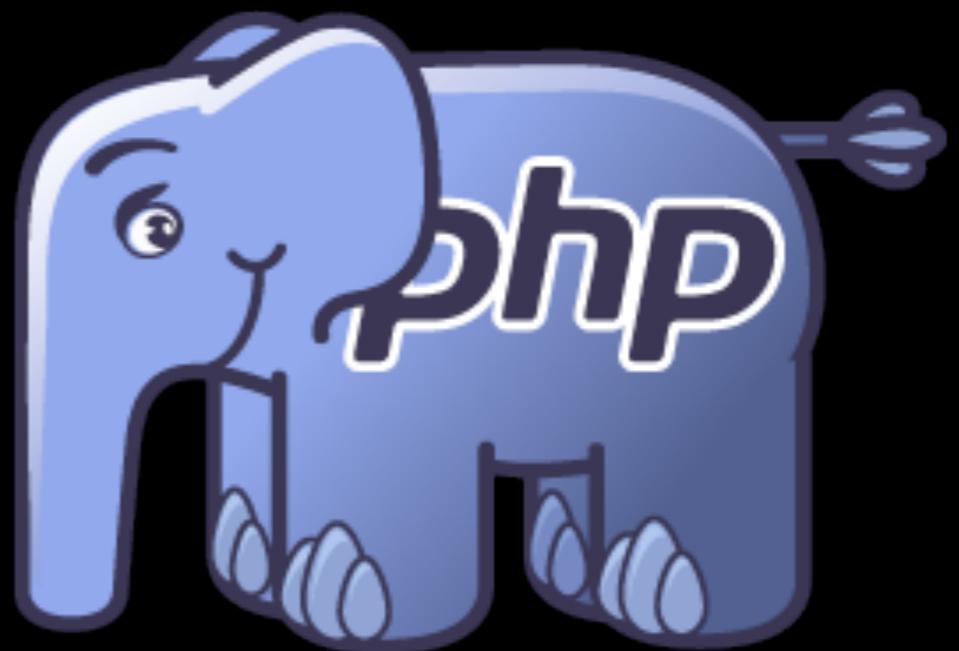
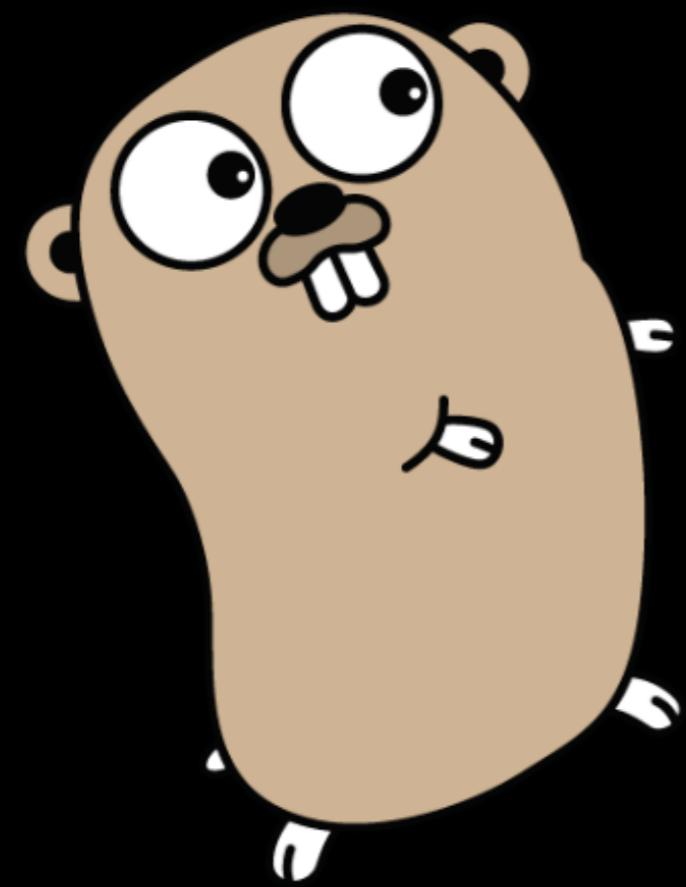


MADE BY IAN BAKER

GO(LANG)



DIFFERENCES



DIFFERENCES



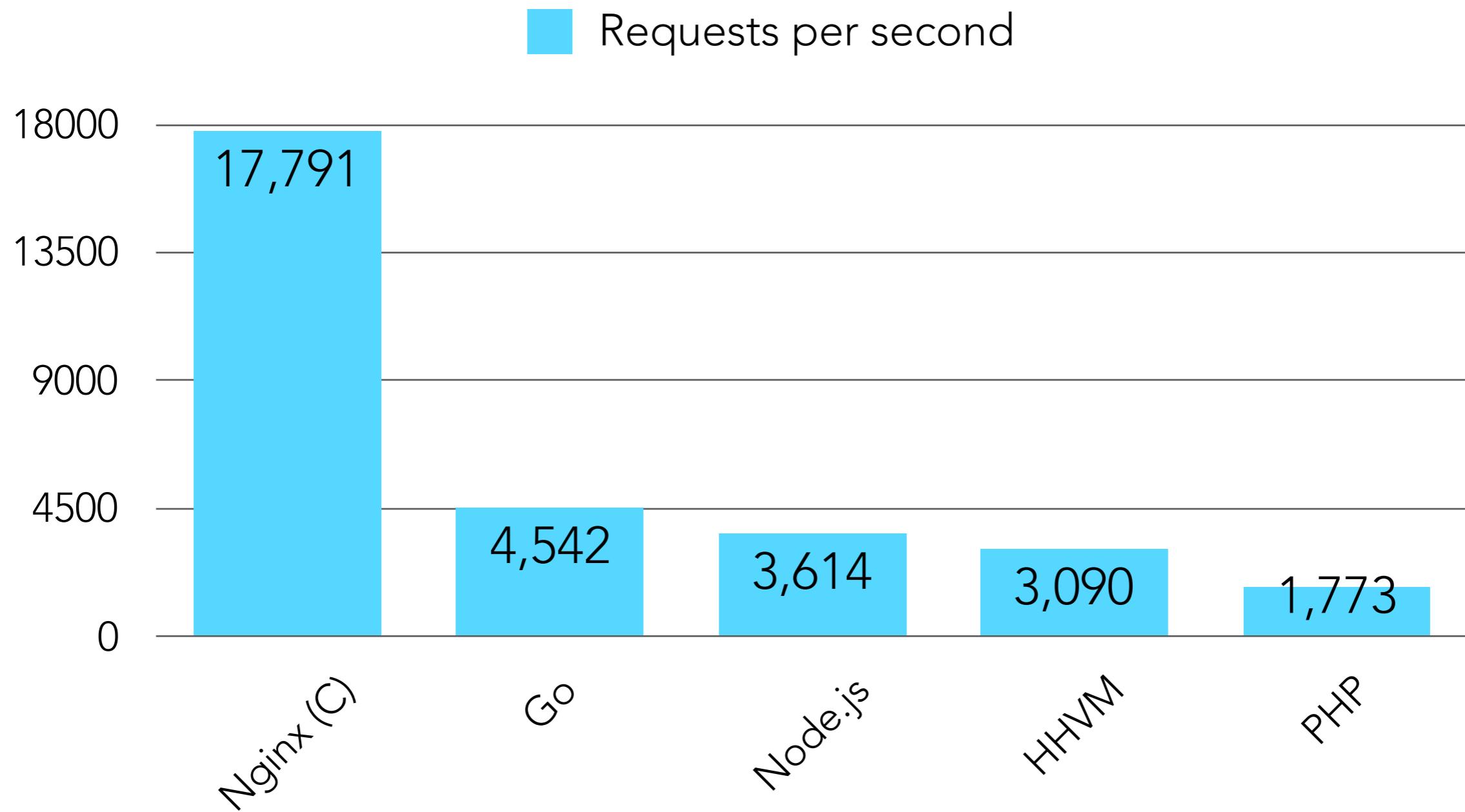
DIFFERENCES



INTERPRETED VS COMPILED



BENCHMARKS



```
package main

import (
    "fmt"
    "math/rand"
)

func main() {
    var greeting string = "Hello! Is it %s you're looking for?\n"
    words := [2]string{"me", "tea"}

    rand.Seed(42)

    for i := 0; i < 5; i++ {
        fmt.Printf(greeting, words[rand.Intn(len(words))])
    }
}
```

```
package main
```

```
import (
    "fmt"
    "math/rand"
)

func main() {
    var greeting string = "Hello! Is it %s you're looking for?\n"
    words := [2]string{"me", "tea"}

    rand.Seed(42)

    for i := 0; i < 5; i++ {
        fmt.Printf(greeting, words[rand.Intn(len(words))])
    }
}
```

```
package main

import (
    "fmt"
    "math/rand"
)

func main() {
    var greeting string = "Hello! Is it %s you're looking for?\n"
    words := [2]string{"me", "tea"}

    rand.Seed(42)

    for i := 0; i < 5; i++ {
        fmt.Printf(greeting, words[rand.Intn(len(words))])
    }
}
```

```
package main

import (
    "fmt"
    "math/rand"
)

func main() {
    var greeting string = "Hello! Is it %s you're looking for?\n"
    words := [2]string{"me", "tea"}

    rand.Seed(42)

    for i := 0; i < 5; i++ {
        fmt.Printf(greeting, words[rand.Intn(len(words))])
    }
}
```

```
package main

import (
    "fmt"
    "math/rand"
)

func main() {
    var greeting string = "Hello! Is it %s you're looking for?\n"
    words := [2]string{"me", "tea"}

    rand.Seed(42)

    for i := 0; i < 5; i++ {
        fmt.Printf(greeting, words[rand.Intn(len(words))])
    }
}
```

```
package main

import (
    "fmt"
    "math/rand"
)

func main() {
    var greeting string = "Hello! Is it %s you're looking for?\n"
    words := [2]string{"me", "tea"}
    rand.Seed(42)

    for i := 0; i < 5; i++ {
        fmt.Printf(greeting, words[rand.Intn(len(words))])
    }
}
```

```
package main

import (
    "fmt"
    "math/rand"
)

func main() {
    var greeting string = "Hello! Is it %s you're looking for?\n"
    words := [2]string{"me", "tea"}

    rand.Seed(42)

    for i := 0; i < 5; i++ {
        fmt.Printf(greeting, words[rand.Intn(len(words))])
    }
}
```

```
$ go run lionel.go
```

Hello! Is it tea you're looking for?

Hello! Is it tea you're looking for?

Hello! Is it me you're looking for?

Hello! Is it me you're looking for?

Hello! Is it tea you're looking for?

```
$ go build lionel.go
```

```
$ ls
```

```
lionel lionel.go
```

```
$ ./lionel
```

```
Hello! Is it tea you're looking for?  
Hello! Is it tea you're looking for?  
Hello! Is it me you're looking for?  
Hello! Is it me you're looking for?  
Hello! Is it tea you're looking for?
```

TOOLS

OUT OF THE BOX



ReactionGIFS.me



IMAGE FROM LEGO.COM

GOPATH

`$G0PATH/src/<vendor>/<project-name>`

e.g.

`/Users/Kat/go-code/src/gitlab.com/katzien/hello`

`/Users/Kat/go-code/src/github.com/golang/go`

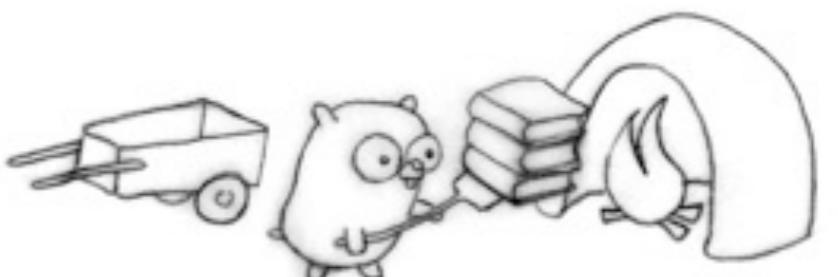
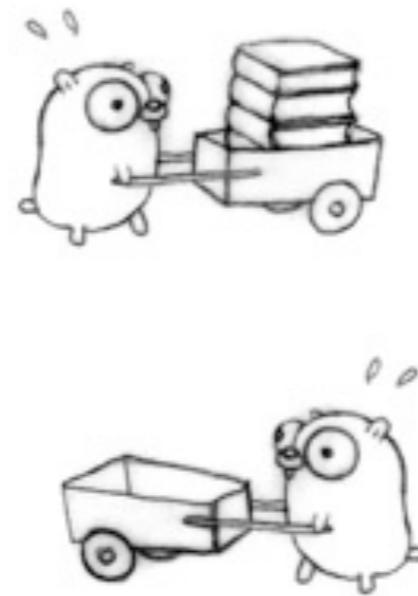
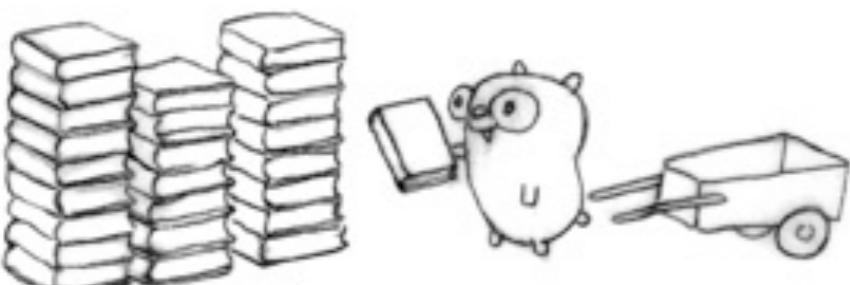
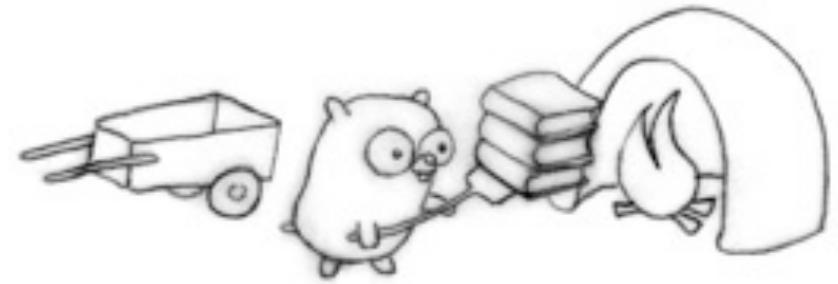
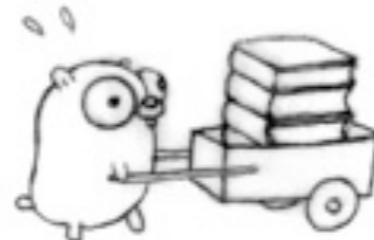
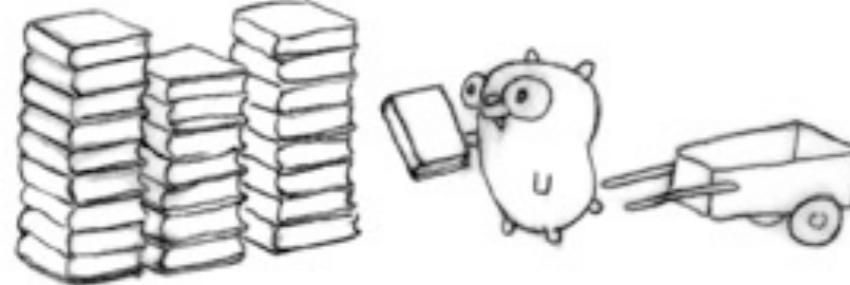


CONCURRENCY VS PARALLELISM

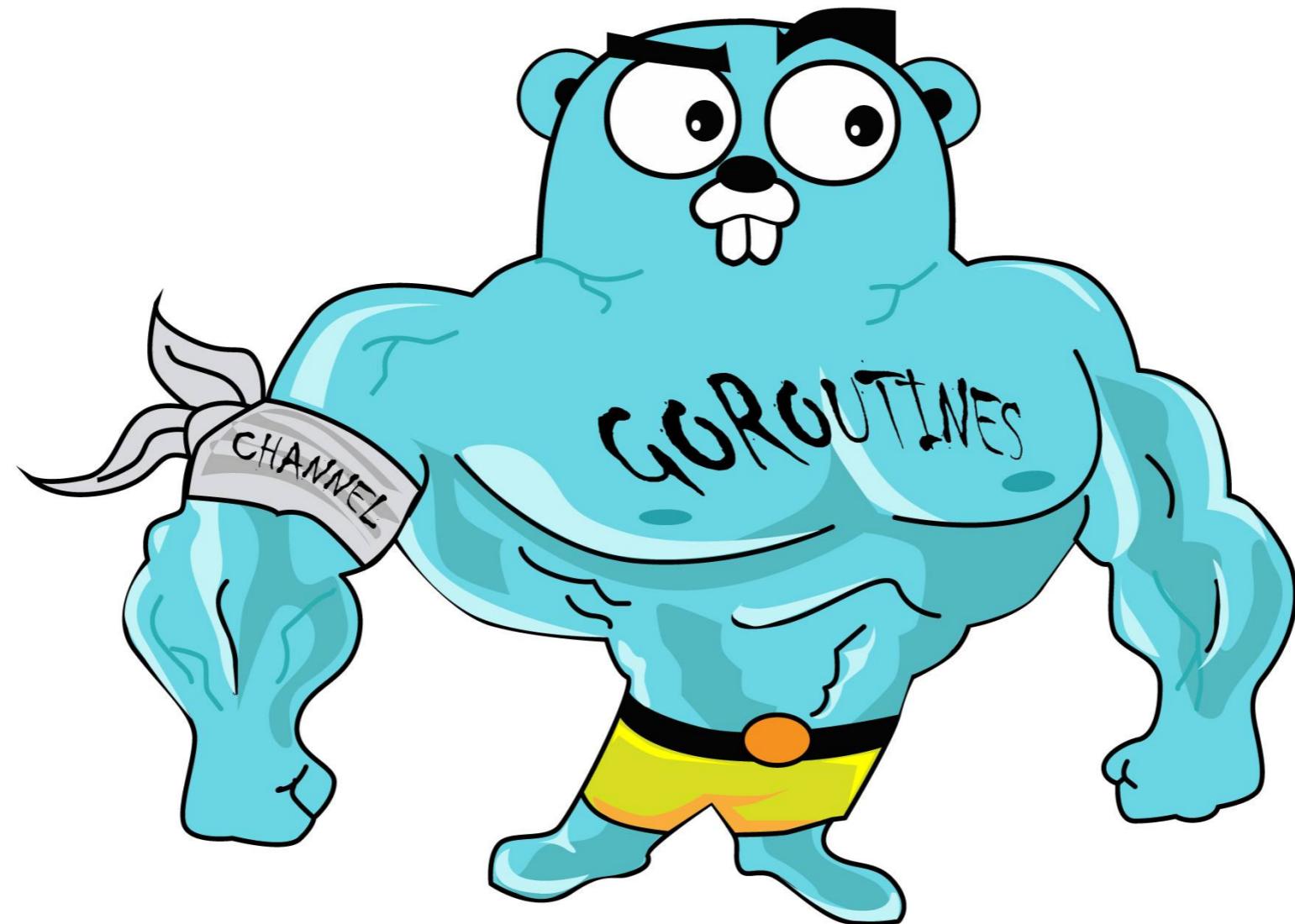
Concurrency: two threads are making progress

Parallelism: two threads are executing simultaneously

CONCURRENCY != PARALLELISM



BUILT-IN CONCURRENCY



GOROUTINES

```
go myFunction()
```

```
func say(word string) {
    for i := 0; i < 5; i++ {
        time.Sleep(10 * time.Millisecond)
        fmt.Println(word)
    }
}

func main() {
    go say("world")
    say("hello")
}
```

```
func say(word string) {
    for i := 0; i < 5; i++ {
        time.Sleep(10 * time.Millisecond)
        fmt.Println(word)
    }
}
```

```
func main() {
    go say("world")
    say("hello")
}
```

```
func say(word string) {
    for i := 0; i < 5; i++ {
        time.Sleep(10 * time.Millisecond)
        fmt.Println(word)
    }
}

func main() {
    go say("world")
    say("hello")
}
```

```
$ go run goroutines.go
```

```
world  
hello  
world  
hello  
hello  
world  
world  
hello  
world  
hello
```

```
$ go run goroutines.go
```

world	world	hello	world
hello	hello	world	hello
world	hello	hello	world
hello	world	world	hello
hello	hello	world	world
world	world	hello	hello
world	hello	hello	hello
hello	world	world	world
world	world	hello	world
hello	hello	world	hello

```
$ go run goroutines.go
```

world	world	hello	world
hello	hello	world	hello
world	hello	hello	world
hello	world	world	hello
hello	hello	world	world
world	world	hello	hello
world	hello	hello	hello
hello	world	world	world
world	world	hello	world
hello	hello	world	hello

```
func main() {
    wg := sync.WaitGroup{}
    wg.Add(1)

    go func() {
        defer wg.Done()
        say("world")
    }()
}

say("hello")

wg.Wait()
fmt.Println("Done!")
}
```

```
func main() {
    wg := sync.WaitGroup{}
    wg.Add(1)

    go func() {
        defer wg.Done()
        say("world")
    }()
}

say("hello")

wg.Wait()
fmt.Println("Done!")
}
```

```
func main() {
    wg := sync.WaitGroup{}
    wg.Add(1)

    go func() {
        defer wg.Done()
        say("world")
    }()
}

say("hello")

wg.Wait()
fmt.Println("Done!")
}
```

CHANNELS



CHANNELS

```
var messenger chan string  
messenger = make(chan string)
```

// Send value to channel
messenger <- "Ohai!"

// Receive from channel and assign to variable
message := <-messenger

```
jobs := make(chan Task)
```

```
for n := limit; n > 0; n-- {
    go func() {
        for task := range jobs {
            do(task)
        }
    }()
}
```

```
for _, task := range workSlice {
    jobs <- task
}
```

WE CAN SPEED THINGS UP



NEITHER IS BETTER OR WORSE:
THEY'RE JUST DIFFERENT



FAVOURITE THINGS

PHP

- ▶ quick to get things done
- ▶ mature frameworks and libraries
- ▶ PHP 7
- ▶ types are coming 

GO

- ▶ strong types
- ▶ keeping things simple
- ▶ error checking policy
- ▶ multiple return values
- ▶ interfaces
- ▶ built-in tools
- ▶ easy to run
- ▶ concurrency
- ▶ speed

NOT IDEAL

PHP

- ▶ inconsistency
- ▶ type juggling
- ▶ no dead code checking
- ▶ bit verbose?

GO

- ▶ less mature ecosystem
- ▶ dependency management
- ▶ “Ugh, do I really have to write it from scratch?”

FROM PHP TO GO AND BACK:

DID ANYTHING CHANGE?



STRICT TYPES

```
declare(strict_types=1);
```

SHORTER, SENSIBLE NAMES

namespace Controllers;

class HomePageController { ... }

Noise

```
public static <I, O> ListenableFuture<O> chain
(ListenableFuture<I> input, Function<? super I, ? extends ListenableFuture<? extends O>> function)
dear god make it stop
```

- a recently observed chat status

HANDLE ERRORS FIRST

```
if (!empty($name)) {  
    if (!is_string($name)) {  
        return false;  
    }  
    return true;  
}  
else {  
    return false;  
}
```

```
if (empty($name)) {  
    return false;  
}  
if (!is_string($name)) {  
    return false;  
}  
return true;
```

INTERFACES

Go's implicit interfaces:

has a, rather than is a

If a struct **has** methods A, B, C, then it implements interface X.

If an object **is** of type X, then it has methods A, B, C.

INTERFACES

```
type Shape interface {  
    Area() float64  
}
```

GO

INTERFACES

```
type Shape interface {
    Area() float64
}
type Circle struct {
    radius float64
}

func (c Circle) Area() float64 {
    return math.Pi * c.radius * c.radius
}
```

GO

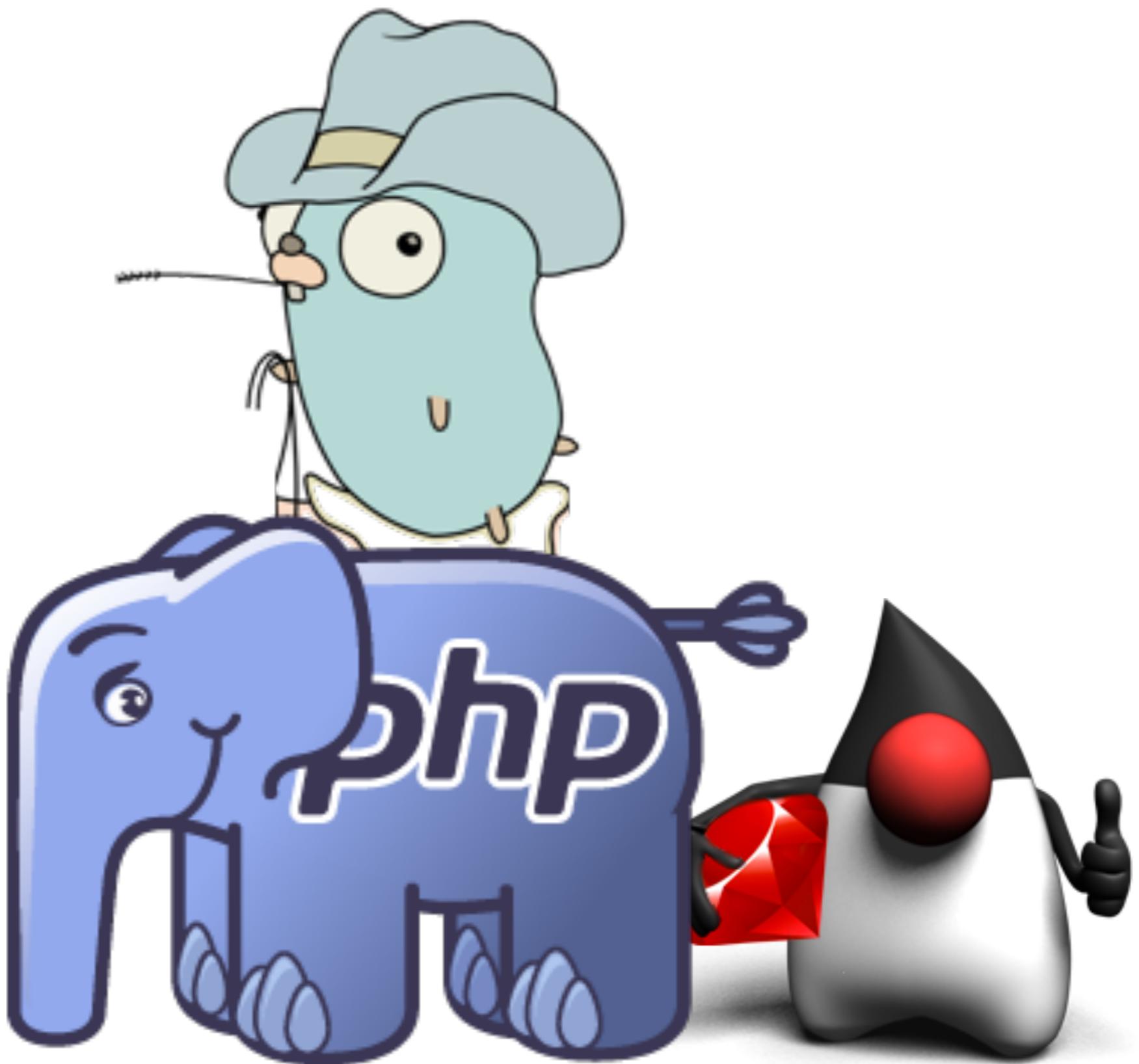
INTERFACES

```
type Shape interface {  
    Area() float64  
}  
type Circle struct {  
    radius float64  
}  
  
func (c Circle) Area() float64 {  
    return math.Pi * c.radius * c.radius  
}
```

GO

```
class Circle implements Shape { ... }
```

PHP



WHERE DO I START?

<https://golang.org/>

TALKS

<https://play.golang.org/>

Concurrency is not parallelism

<https://tour.golang.org/>

<https://blog.golang.org/concurrency-is-not-parallelism>

<https://blog.golang.org/>

Simplicity is complicated

https://www.youtube.com/watch?v=rFejpH_tAHM

<https://gobyexample.com>

Building containers in Go

<https://www.youtube.com/watch?v=HPuvDm8lC-4>

@KASIAZIEN

Debuggers From Scratch

<https://www.youtube.com/watch?v=ZrpkrMKYvqQ>

SUCCE

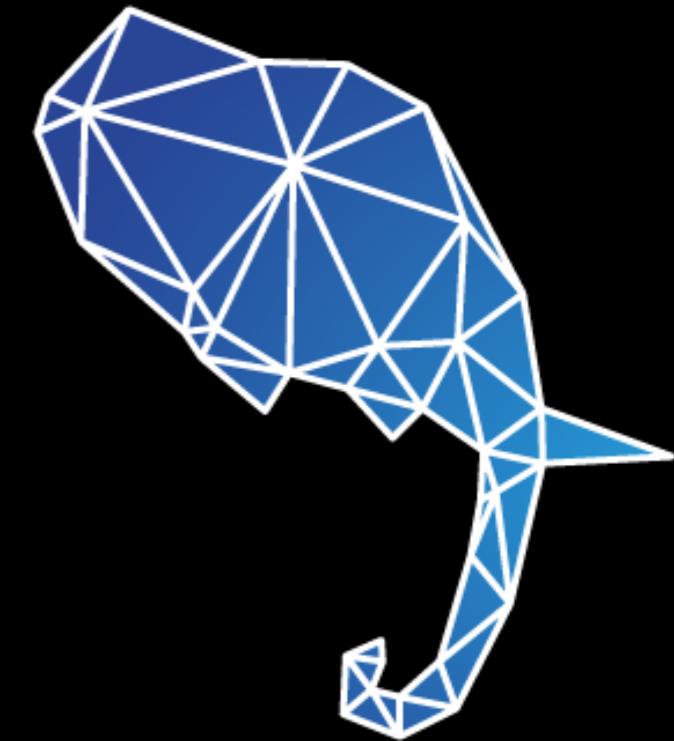
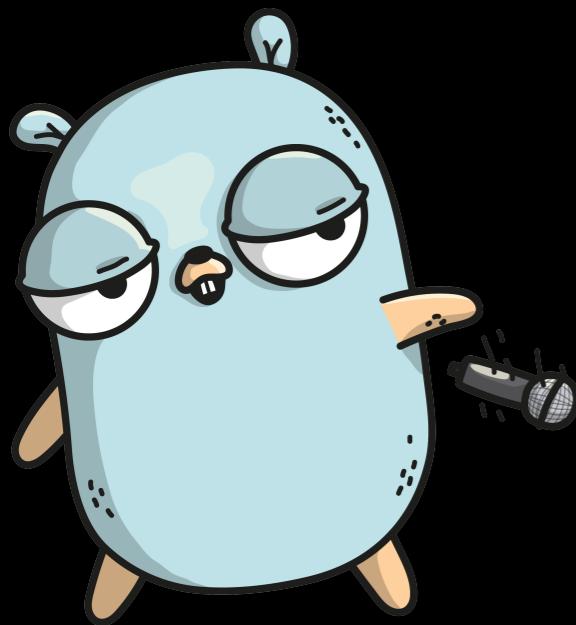
CAPTURE

COMMUNITY



IMAGE FROM @JMIKOLA

THANKS FOR LISTENING!



@KASIAZIEN