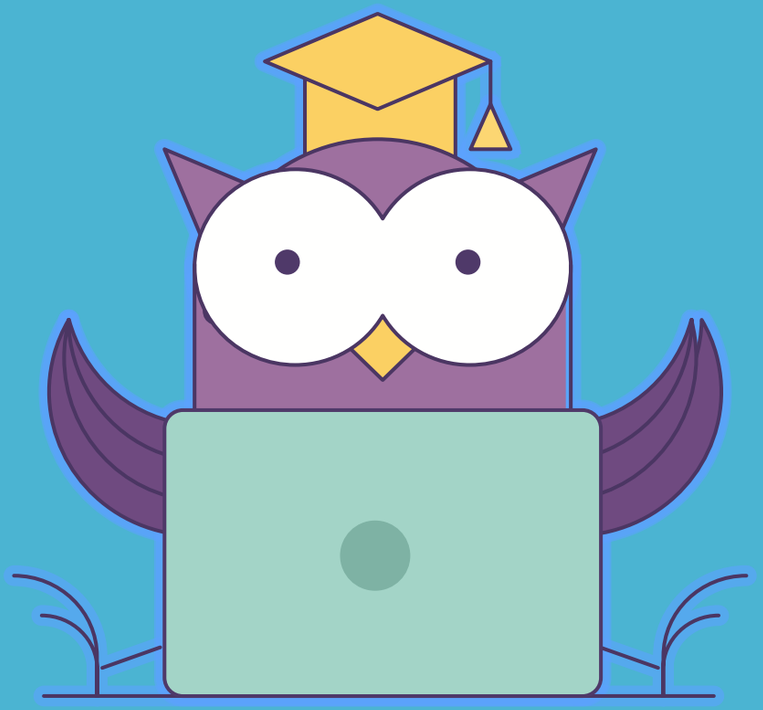
ОНЛАЙН-ОБРАЗОВАНИЕ

Начало работы с Go

Иван Ремень

OTUS

Как меня слышно и видно?



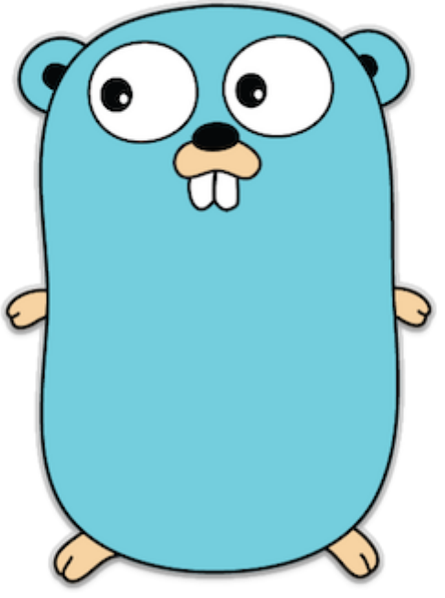
> Напишите в чат

+ если все хорошо

– если есть проблемы со звуком или с видео

Цель занятия

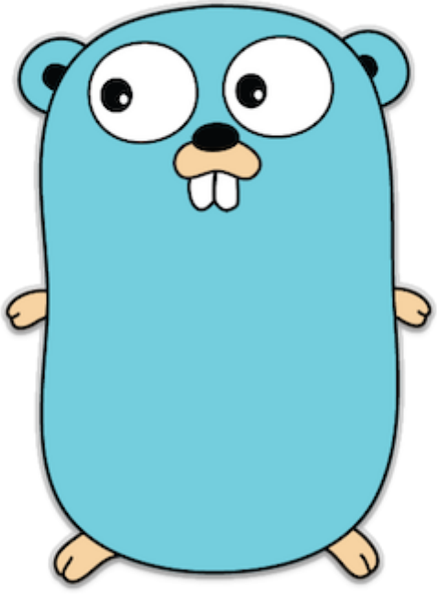
- Понять, почему появился еще один язык
- Изучить основные синтаксические конструкции



Преподаватель

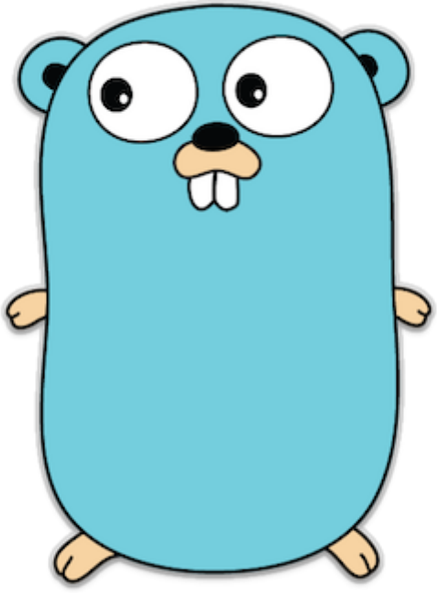
Иван Ремень

- Выпускник МГТУ им. Н.Э. Баумана
- Получил степень магистра в 2016 году
- Начал свою карьеру в Почта@Mail.Ru стажером в 2013 году
- С 2016 года тимлид команды C/C++ разработки
- В окрябре 2018 года перешел в Ситимобил
- С апреля 2019 года - руководитель серверной разработки
- Преподаватель с 2015 года
- Пишу на Go с 2016 года

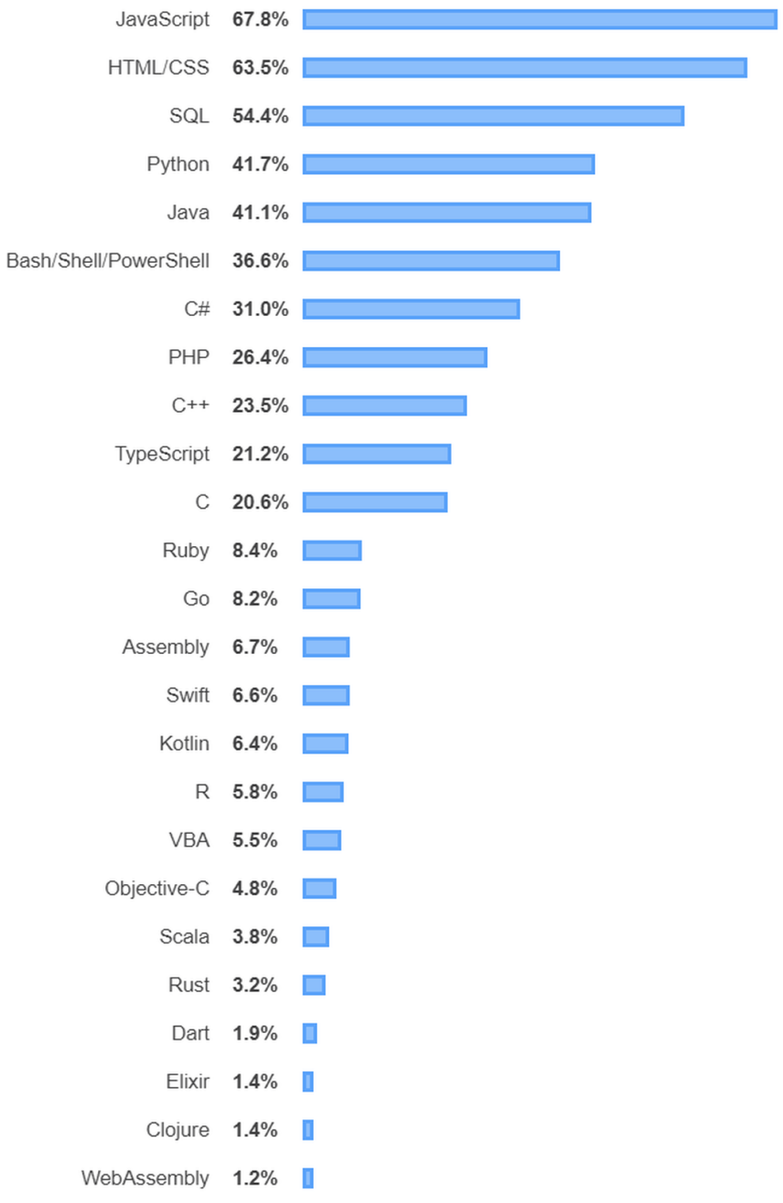


Кто уже пишет на Go?

- По работе?
- Для себя?



Рейтинг популярности языков



Где используется Go

- Веб (backend)
- Системные утилиты
- Devops
- Сетевое программирование

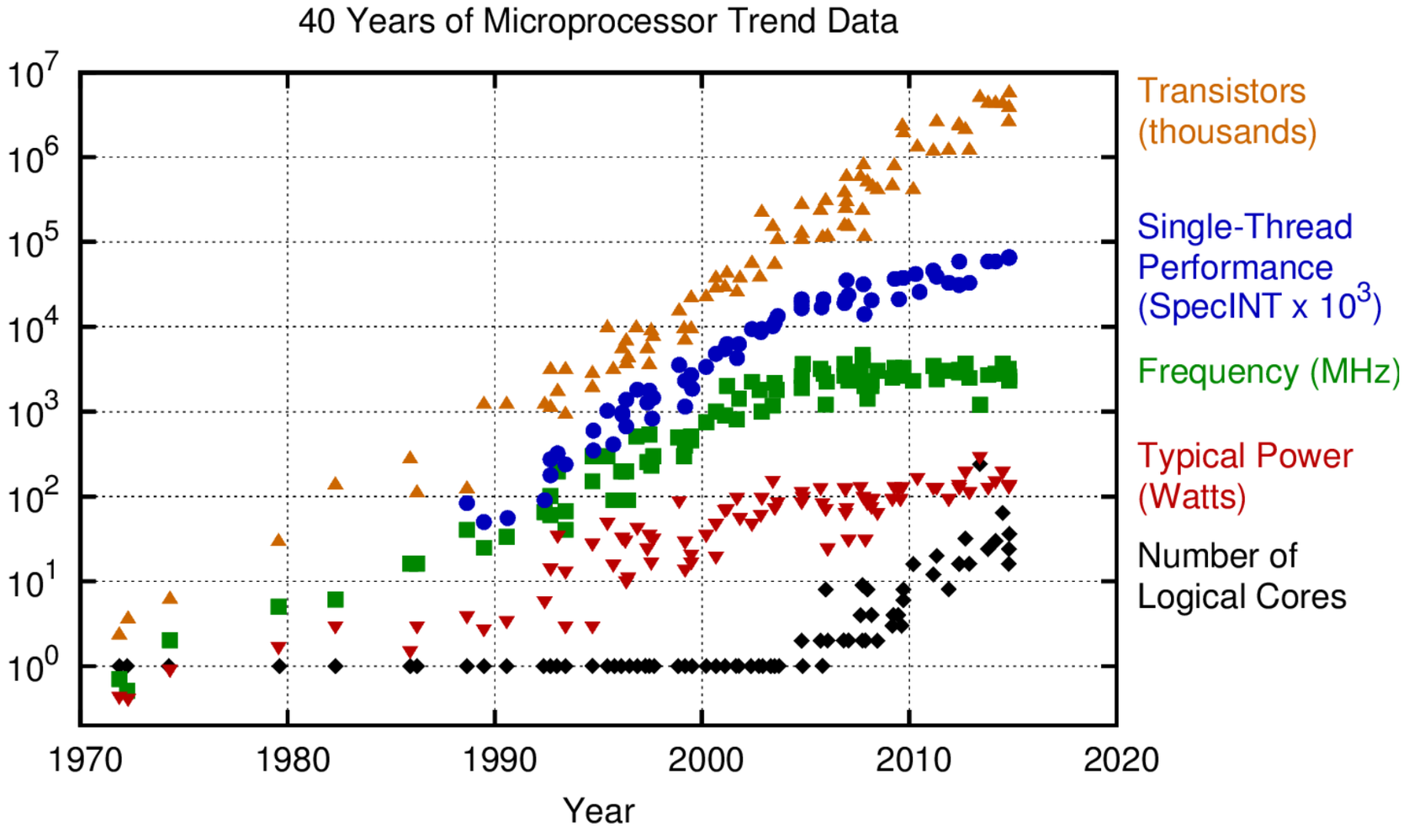
Где НЕ используется Go

- Разработка игр
- Научные вычисления
- Машинное обучение
- Встраевыемые устройства

- Grafana
- Docker
- consul
- Kubernetes

Немного истории

1956 — 1958 LISP
1959 Cobol
1964 Basic
1970 Pascal
1970 C
1978 SQL
1983 C++
1991 Python
1995 Java
1995 PHP
2009 Go
2010 Rust



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

Go (часто также Golang) — компилируемый многопоточный язык программирования, разработанный внутри компании Google. Разработка Go началась в сентябре 2007 года, его непосредственным проектированием занимались Роберт Гризмер, Роб Пайк и Кен Томпсон, занимавшиеся до этого проектом разработки операционной системы Inferno. Официально язык был представлен в ноябре 2009 года

Проблемы, которые решали в google

- Медленная сборка
- Неконтролируемые зависимости
- Каждый программист использует свое подмножество языка
- Сложности деплоя

Характеристики Go

Императивный
Компилируемый в нативный код
Статическая типизация
Нет классов, но есть структуры с методами
Есть интерфейсы
Нет наследования, но есть встраивание
Функции - объекты первого класса
Есть замыкания
Функции могут возвращать больше 1 значения
Есть указатели, но нет адресной арифметики
Обширные возможности для конкурентности
Сборка в 1 бинарный файл
Набор стандартный инструментов

Заповеди Роба Пайка

Don't communicate by sharing memory, share memory by communicating.
Concurrency is not parallelism.
The bigger the interface, the weaker the abstraction.
Make the zero value useful.
interface{} says nothing.
Gofmt's style is no one's favorite, yet gofmt is everyone's favorite.
A little copying is better than a little dependency.
Syscall must always be guarded with build tags.
Clear is better than clever.
Reflection is never clear.
Errors are values.
Don't just check errors, handle them gracefully.
Documentation is for users.
Don't panic.

Структура проекта

```
MBP-Remen:lesson-1-1 bhychik$ echo $GOPATH
/Users/bhychik/gopath/
MBP-Remen:lesson-1-1 bhychik$ tree -L 2 /Users/bhychik/gopath/
/Users/bhychik/gopath/
├── bin
|   ├── bee
|   ├── bolt
|   ├── deadcode
|   ├── dupl
|   ├── easyjson
|   └── varcheck
├── pkg
|   ├── darwin_amd64
|   └── mod
└── src
    ├── coursera
    ├── github.com
    ├── gitlab.com
    ├── go.uber.org
    ├── golang.org
    ├── google.golang.org
    ├── gopkg.in
    └── web
```

Hello world

```
package main

func main() {
    println("Hello, world")
}
```

https://play.golang.org/p/T9DUX6PSBG9

Импортирование пакетов

```go
package main

import (
    "fmt"
)

func main() {
    fmt.Println("Hello, world!")
}
```

https://play.golang.org/p/AH8TPSfGkFd

Импортирование пакетов

```go
package main

import (
    "fmt"
)

func main() {
    fmt.Println("Hello, world!")
}
```

https://play.golang.org/p/AH8TPSfGkFd

Объявление переменных

```go
package main

import (
    "fmt"
)

func main() {
    var a int = 5
    var b = 6
    c := 7
    fmt.Printf("%d, %d, %d\n", a, b, c)

    c, d := 9, 132
    b = 15

    fmt.Printf("%d, %d, %d, %d\n", a, b, c, d)
}
```

https://play.golang.org/p/NScjfQ0MDqf

```go
package main

import (
    "fmt"
)

func main() {
    const (
        a = 1 // a == 1
        b = 2 // b == 2
        c     // c == 2
        d     // d == 2
    )

    fmt.Printf("%d %d %d %d", a, b, c, d)
}
```

https://play.golang.org/p/7t0xxlXfRfI

```go
package main

import (
    "fmt"
)

func main() {
    const (
        a = iota // a == 0
        b = iota // b == 1
        c = iota // c == 2
        d        // d == 3 (implicitely d = iota)
    )
    fmt.Printf("%d %d %d %d", a, b, c, d)

}
```

https://play.golang.org/p/BaQfGzGrYnZ
Можно использовать формулы

Условный оператор

```
package main

func main() {
    a := 8
    b := 7

    if a > b {
        println("yes")
    }
}
```

https://play.golang.org/p/cJAoA5XXXzO

```
package main

func main() {
    a := 8
    b := 7

    if a > b {
        println("yes")
    } else {
        println("no")
    }
}
```

https://play.golang.org/p/Evz0eqq-DjJ

Присвоение внутри блока if

```
package main

func main() {
    if a, b := 5, 7; a > b {
        println("yes")
    } else {
        println("no")
    }
}
```

https://play.golang.org/p/RHfUJSzSq-s

```
package main

func main() {
    for i := 0; i < 10; i++ {
        println(i)
    }
}
```

https://play.golang.org/p/As_cwgGaJkH

```
package main

func main() {
        i := 0
    for i < 10 {
        println(i)
        i++
    }
}
```

https://play.golang.org/p/KxdBiaP5QlC

Вечный цикл

```
package main

func main() {
    i := 0
    for {
        println(i)
        i++
        if i > 10 {
            break
        }
    }
}
```

Итерация по массиву

```
package main

func main() {
    a := [...]int{81, 54, 43, 66}
    for i, v := range a {
        println(i, v)
    }
}
```

https://play.golang.org/p/JlnxPkg3OD4

```go
package main

func main() {
    a := 15
    switch a {
    case 10:
        println("One")
    case 15:
        println("Two")
    case 30:
        println("Three")
    default:
        println("DEFAULT!")
    }
}
```

https://play.golang.org/p/wWii1tobB2Z

switch (fallthrough)

```go
package main

func main() {
    a := 15
    switch a {
    case 10:
        println("One")
        fallthrough
    case 15:
        println("Two")
        fallthrough
    case 30:
        println("Three")
        fallthrough
```

Сложный switch

```
package main

func main() {
    a := 10
    switch {
    case a < 10:
        println("One")
    case a < 30 && a >= 10:
        println("Two")
    case a > 3:
        println("Three")
    default:
        println("DEFAULT!")
    }
}
```

https://play.golang.org/p/SHpa3b3C_zq

- Поняли, почему появился еще один язык
- Изучили основные синтаксические конструкции

Вопросы?

Не заполните заполнить опрос. Ссылка на опрос будет в слаке.

Спасибо за внимание!