Язык программирования Golang

21 октября 2016

Владимир Ананьев
АО "Программный Регион"

Что такое Golang?

"Go is an open source programming language that makes it easy to build simple, reliable, and efficient software."

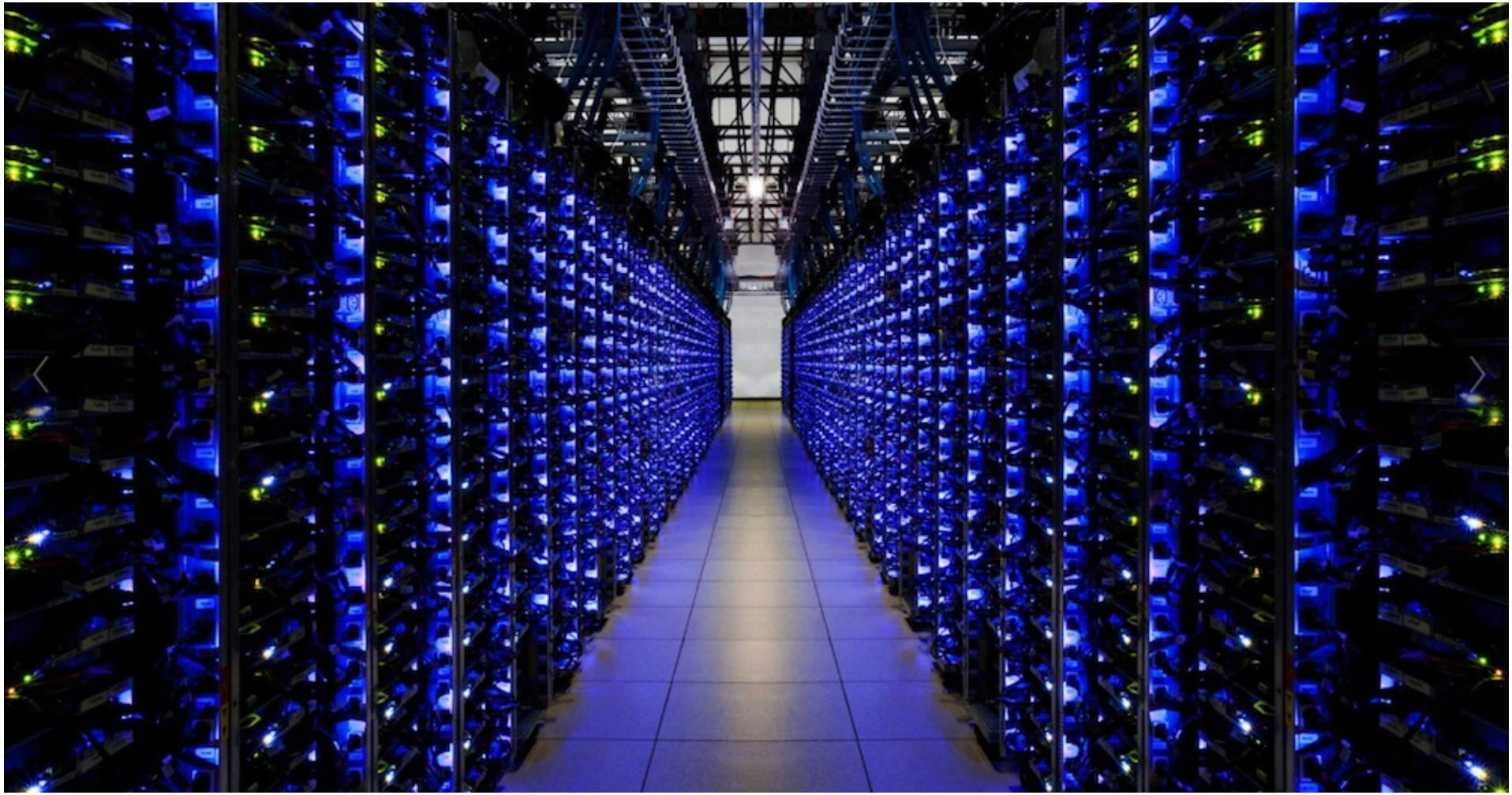
golang.org/ (https://golang.org/)

Краткая история создания языка

- Разработка началась в **сентябре 2007 года**
- Непосредственно проектированием занимались **Роберт Гризмер**, **Роб Пайк** и **Кен Томпсон**
- Официально язык представлен в **ноябре 2009 года**
- На данный момент поддерживаются **FreeBSD**, **OpenBSD**, **Linux**, **Mac OS**, **Windows**
- Начиная с версии **1.3** экспериментальная поддержка **DragonFly**, **Plan 9** и **Solaris**
- Начиная с версии **1.4** поддержка **Android**
- Текущая версия **1.7**

Почему Go?

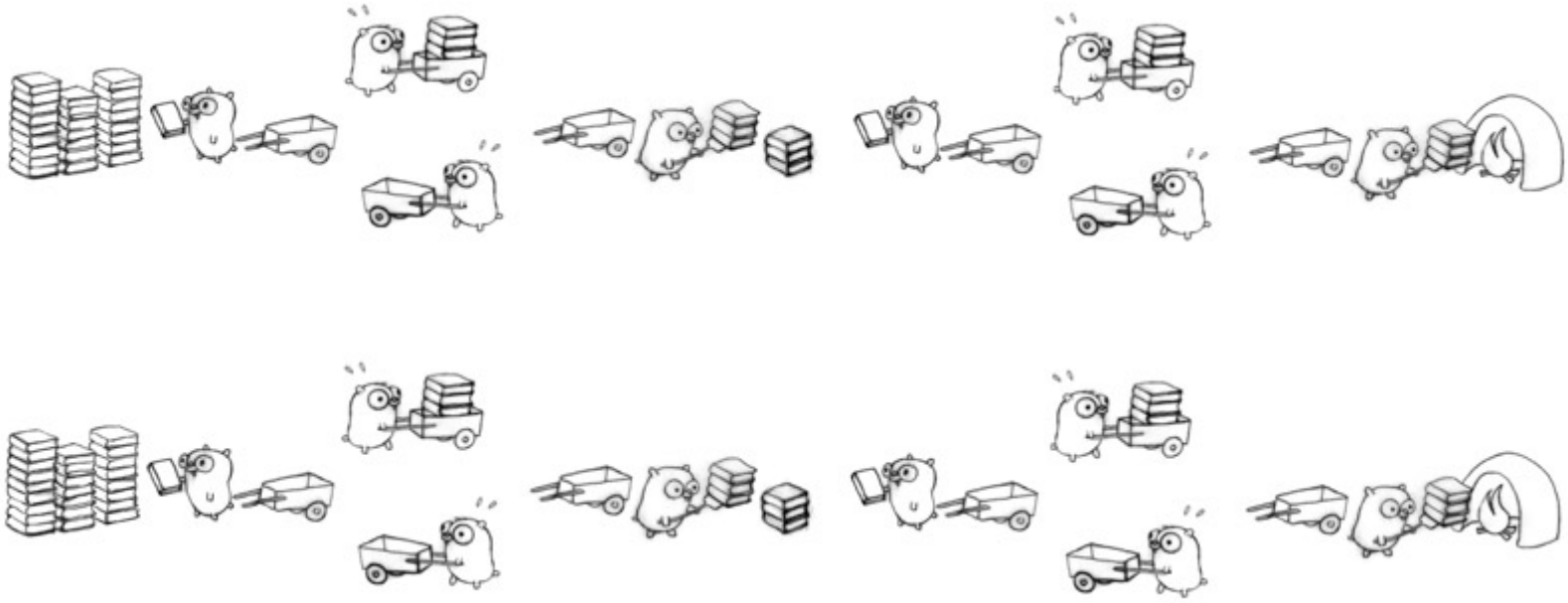
Go - это ответ на проблемы масштабирования в Google



Масштаб системы

- Предназначен для масштабирования на 10^{6+} машинах
- Ежедневные задачи на тысячах машин
- Координация задач, взаимодействие с другими задачами внутри системы
- Много вешей должны происходить сразу

Решение: качественная поддержка многопоточности



Вторая проблема: Масштаб персонала

В 2011:

- 5000+ разработчиков в 40+ офисах
- 20+ изменений в минуту
- 50% наработок кода меняется каждый месяц
- 50 миллионов тестов каждый день

Решение: создать простой язык, с помощью которого создавать большую базу для повторного использования

Кто использует Go в Google?

Сотни проектов. Тысячи программистов. Миллионы строк Go кода.

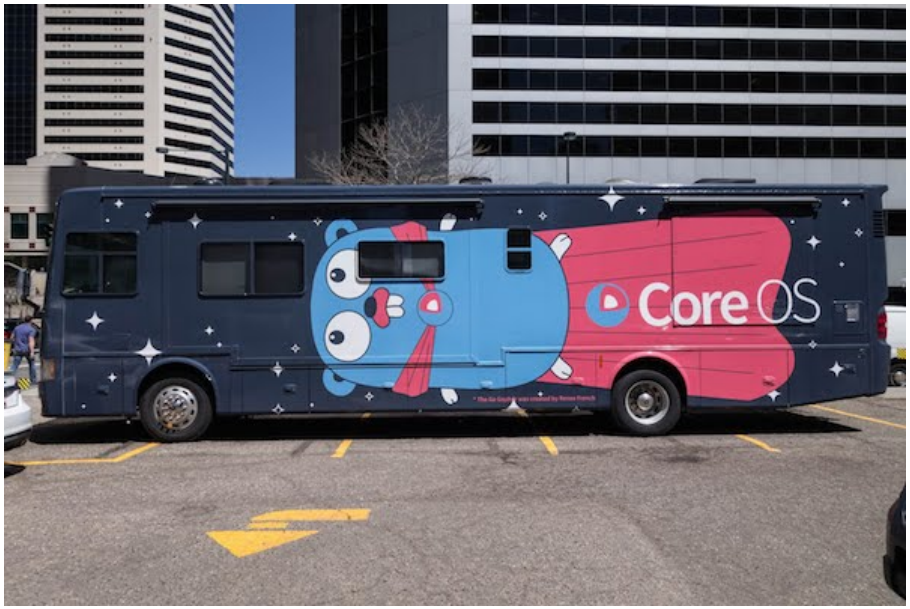
Примеры:

- **Flywheel**: SPDY proxy for Chrome on mobile devices
- **dl.google.com**: Download server for Chrome, ChromeOS, Android SDK, Earth, etc.
- **Vitess**: YouTube MySQL balancer
- **Seesaw**: Linux Virtual Server (LVS) based load balancer
- **Lingo**: Logs analysis in Go, migrated from Sawzall

Кто использует Go за пределами Google?

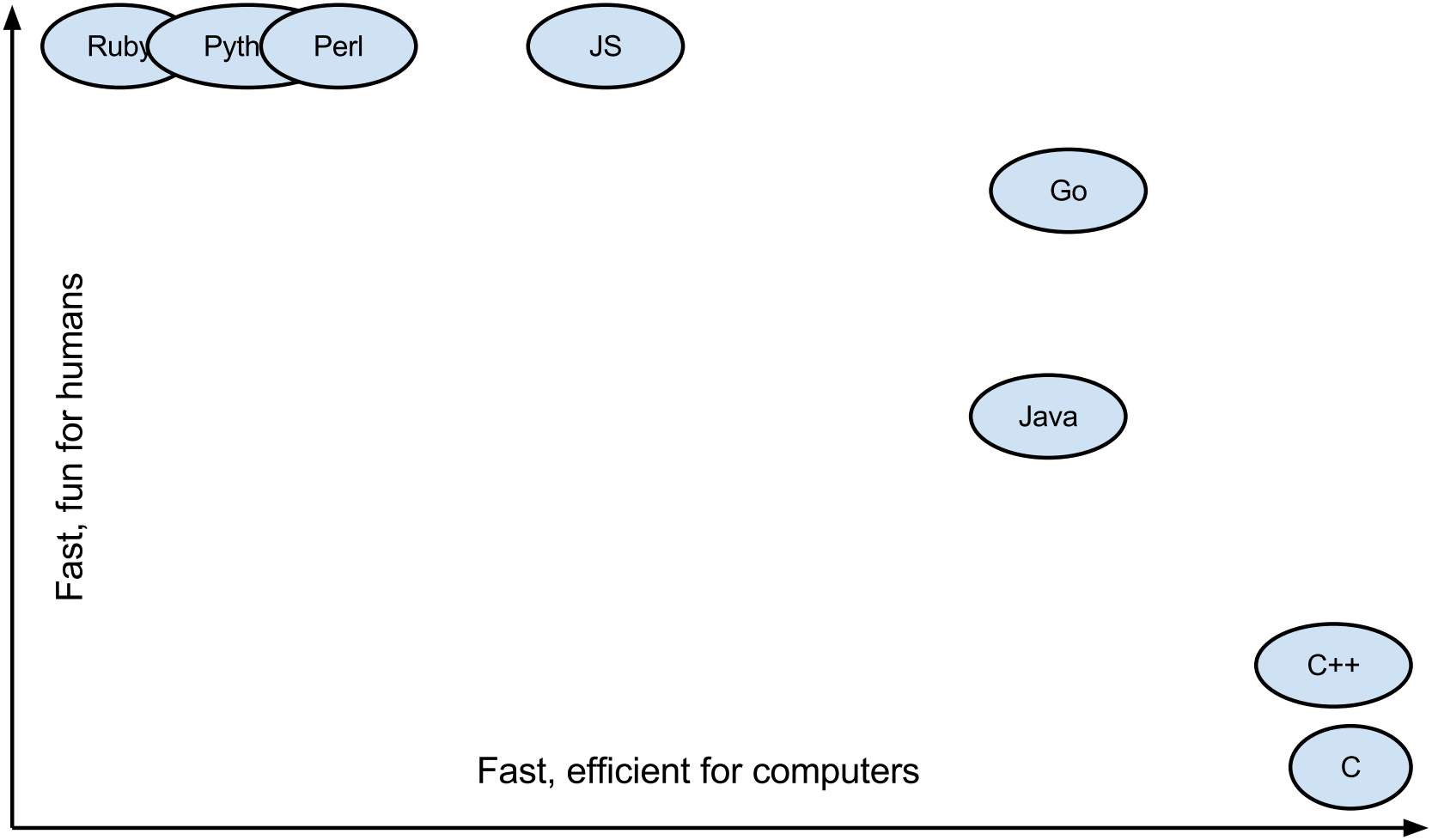
golang.org/wiki/GoUsers (http://golang.org/wiki/GoUsers)

Aerospike, BBC Worldwide, Bitbucket, Booking.com, Core OS, Datadog, Digital Ocean, Docker, Dropbox, Facebook, Getty Images, GitHub, GOV.UK, Heroku, IBM, Intel, InfluxDB, Iron.io, Kubernetes, Medium, MongoDB, Mozilla services, Netflix, New York Times, pool.ntp.org, Rackspace, Shutterfly, SmugMug, SoundCloud, SpaceX, Square, Stack Exchange, Thomson Reuters Eikon, Tumblr, Twitch, Twitter, Uber, VMWare ...



Сравнение с другими языками программирования

"Go: 90% Perfect, 100% of the time" - Brad Fitzpatrick, GoCon Tokyo, May 2014



Go имеет много сходств с Java

- С-подобный синтаксис
- Статически типизированный
- Имеет сборщик мусора (garbage collector)
- Безопасное использование памяти (memory safe)
- Переменные всегда инициализированы (0/nil/false)
- Есть методы
- Есть поддержка интерфейсов
- Приведение типов
- Механизм reflection

Но также имеет и различия

Fast, efficient for computers:

- Программы компилируются в машинный код, нет интерпретатора
- Контроль заполнения памяти (**memory layout**)

Fun, fast for humans:

- Простой синтаксис
- Статически слинкованные бинарные файлы
- Возвращаемые значения функций и замыкания
- **UTF-8** по умолчанию
- Встроенные типы **map** и **array/slice**
- Встроенная многопоточность (**concurrency**) (www.youtube.com/watch?v=cN_DpYBzKso (https://www.youtube.com/watch?v=cN_DpYBzKso))

В Go нет:

- Классов (есть структуры)
- Наследования
- Конструкторов
- `final`
- Exceptions
- Generics

Почему нет?

- **Ясность** - самое главное.
- Должно быть понятно при прочтении кода, что конкретно программа делает.
- Должно быть понятно при написании программы, как заставить программу делать необходимое.
- Less is exponentially more (Pike, 2012) (http://commandcenter.blogspot.com/2012/06/less-is-exponentially-more.html)
- Go at Google: Language Design in the Service of Software Engineering (Pike, 2012) (http://talks.golang.org/2012/splash.article)

Установка Golang

golang.org/dl/ (https://golang.org/dl/)

- Скачать архив
- Скопировать в **/usr/local**
- Создать директорию **~/go**
- Обновить файлик **~/.profile**

```
export PATH=$PATH:/usr/local/go/bin
export GOPATH=$HOME/go
export PATH=$PATH:$GOPATH/bin
```

Hello, world!

Чтобы убедиться, что все хорошо, создаем файлик **hello.go**:

```
package main

import "fmt"

func main() {
    fmt.Printf("Hello, world!\n")
}
```

Run

и запускаем

```
go run hello.go
```

Что за инстументы у нас есть

- Утилита **go**, которая делает много всего
- Набор переменных окружения для конфигурации
- Стандартная библиотека
- 3 директории:
- **src** - исходные коды, огранизованные в виде пакетов (package)
- **pkg** - package objects
- **bin** - скомпилированные бинарники

$ go

```
build       compile packages and dependencies
clean       remove object files
doc         show documentation for package or symbol
env         print Go environment information
fix         run go tool fix on packages
fmt         run gofmt on package sources
generate    generate Go files by processing source
get         download and install packages and dependencies
install     compile and install packages and dependencies
list        list packages
run         compile and run Go program
test        test packages
tool        run specified go tool
version     print Go version
vet         run go tool vet on packages
```

golang.org/cmd/go/ (http://golang.org/cmd/go/)

Переменные окружения

```
$ go env

GOARCH="amd64"
GOBIN=""
GOEXE=""
GOHOSTARCH="amd64"
GOHOSTOS="linux"
GOOS="linux"
GOPATH="/home/user/go"
GORACE=""
GOROOT="/usr/lib/go"
GOTOOLDIR="/usr/lib/go/pkg/tool/linux_amd64"
CC="gcc"
GOGCCFLAGS="-fPIC -m64 -pthread -fmessage-length=0"
CXX="g++"
CGO_ENABLED="1"
```

workspace

```
bin/
    hello                          # binary executable
    outyet                         # binary executable
pkg/
    linux_amd64/
        github.com/golang/example/
            stringutil.a           # package object
src/
    github.com/golang/example/
        .git/                      # Git repository metadata
        hello/
            hello.go               # command source
        outyet/
            main.go                # command source
            main_test.go           # test source
    golang.org/x/image/
        .git/                      # Git repository metadata
        bmp/
            reader.go              # package source
            writer.go              # package source
    ...
```

golang.org/doc/code.html#Workspaces (http://golang.org/doc/code.html#Workspaces)

Стандартная библиотека

archive tar zip bufio builtin bytes compress bzip2 flate gzip lzw zlib
container heap list ring context crypto aes cipher des dsa ecdsa elliptic hmac
md5 rand rc4 rsa sha1 sha256 sha512 subtle tls x509 pkix database sql
driver debug dwarf elf gosym macho pe plan9obj encoding ascii85 asn1
base32 base64 binary csv gob hex json pem xml errors expvar flag fmt
go ast build constant doc format importer parser printer scanner token
types hash adler32 crc32 crc64 fnv html template image color palette
draw gif jpeg png index suffixarray io ioutil log syslog math big cmplx
rand mime multipart quotedprintable net http cgi cookiejar fcgi httptest
httptrace httputil pprof mail rpc jsonrpc smtp textproto url os exec signal user
path filepath reflect regexp syntax runtime cgo debug msan pprof race trace
sort strconv strings sync atomic syscall testing iotest quick text scanner
tabwriter template parse time unicode utf16 utf8 unsafe

+ Огромное количество пакетов (официальных и неофициальных) на Github

Что дальше?

Возвращаемся обратно к примеру *Hello, world!*:

```
package main

import "fmt"

func main() {
    fmt.Printf("Hello, world!\n")
}
```

и сфокусируемся на строчке:

```
package main
```

Особенности использования пакетов

- Один файл может принадлежать только одному пакету
- В одном пакете может быть несколько файлов
- Вся стандартная библиотека состоит из пакетов
- По соглашению, одна директория содержит только один пакет
- Пакет компилируется отдельно (**as a standalone unit**)
- Пакет компилируется один раз
- Неиспользуемые импорты вызывают ошибку компиляции
- Один пакет **main**

Видимость внутри пакета

- Экспортируемые (**like public**)
- Неэкспортируемые (**like private**)

Демо

Типы данных

- bool
- uint, int
- float32, float64
- complex64, complex128
- string
- array
- struct
- channel

Еще типы данных

- int8
- int16
- int32
- int64
- uint8
- uint16
- uint32
- uint64

Демо, объявление и использование переменных

Функция init()

```
package main

import "fmt"

var subject string

func init() {
    subject = "world"
}

func main() {
    fmt.Printf("Hello, %s!\n", subject)
}
```

iota

```
const (
    Monday = iota
    Tuesday
    Wednesday
    Thursday
    Friday
    Partyday
    Sunday
)
```

Wikipedia: Iota is the ninth letter of the Greek alphabet. (http://en.wikipedia.org/wiki/Iota)

if

```
if age < 13 {
    ...
} else if age >= 13 && age < 20 {
    ...
} else {
    ...
}
```

for

- Нет **while**
- Нет **foreach**
- Для циклов есть только **for**

```
for i := 0; i < 20; i++ {
    ...
}

for {
    ...
}

for i, v := range slice {
    ...
}

for k, v := range map {
    ...
}
```

switch

```
switch {
case age < 13:
    ...
case age >= 13 && age < 20:
    ...
default:
    ...
}
```

goto

Материалы

- Tour of Go (tour.golang.org (https://tour.golang.org))
- Golang wiki (golang.org/wiki/Learn (http://golang.org/wiki/Learn))
- Go by Example (gobyexample.com (https://gobyexample.com))
- Effective Go (golang.org/doc/effective_go.html (https://golang.org/doc/effective_go.html))
- Program your next server in Go (talks.golang.org/2016/applicative.slide (https://talks.golang.org/2016/applicative.slide))

Задание

- Просмотреть "Tour of Go" (tour.golang.org (https://tour.golang.org))
- Установить Go (golang.org/dl/ (https://golang.org/dl/), последнюю версию)
- Установить LiteIDE (sourceforge.net/projects/liteide/files/ (https://sourceforge.net/projects/liteide/files/), последнюю версию)
- Написать приложение **track-server**:
- **Веб-приложение** с использованием стандартной библиотеки **net/http** (golang.org/pkg/net/http/ (https://golang.org/pkg/net/http/))
- Обрабатывает запросы на порту **8085**
- Формат входящего запроса **http://127.0.0.1:8085/{code}**
- **{code}** - base64-шифр от url адреса, например, **d3d3LnlhbmRleC5ydQ==** означает **www.yandex.ru** (base64.ru/ (http://base64.ru/))
- golang.org/pkg/encoding/base64/ (https://golang.org/pkg/encoding/base64/)
- Алгоритм работы: приходит запрос, парсится url, происходит редирект на требуемый url

Thank you

21 октября 2016

Владимир Ананьев
АО "Программный Регион"
vladimir.ananyev@regium.com (mailto:vladimir.ananyev@regium.com)