Backend Golang
Урок 2

Инструментарий (build, test, fmt), управление зависимостями (go mod)

Формат занятий

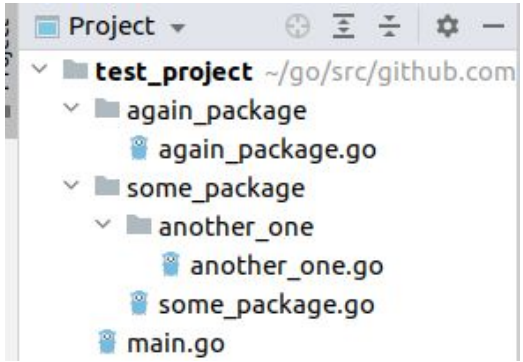
Все ключаем камеры.

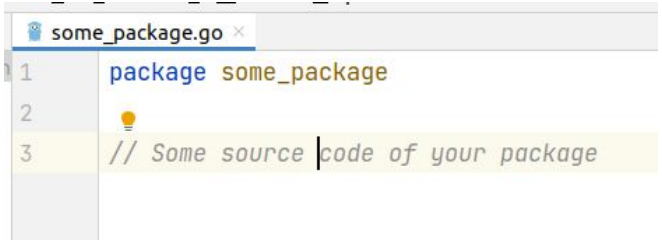
Занятие состоит 3-х уроков по 40 минут:

1. Разбор ДЗ, теория по новой теме;
2. Теория, практика;
3. Практика, закрепление материала.

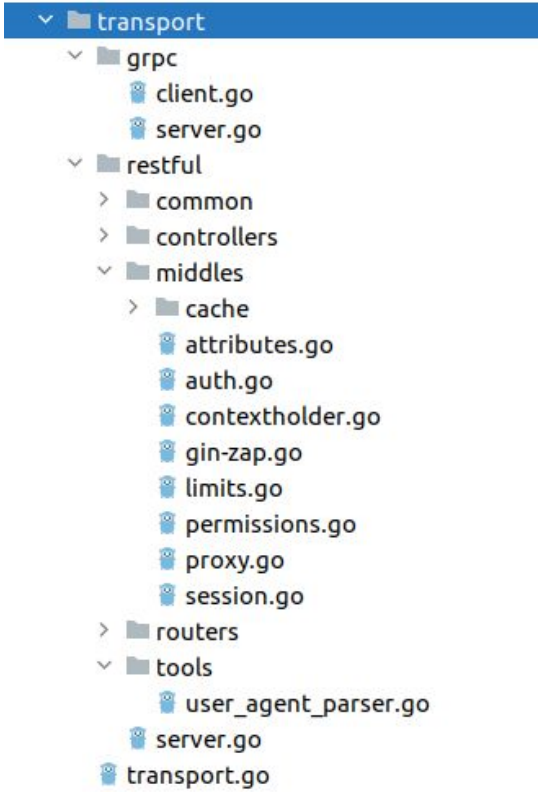
Перерыв между уроками — 15 минут.

Строительный блок — package





Боевой пример



Программы на Go делятся на 2 типа:

1. Исполняемые
2. Библиотеки

Исполняемая программа содержит package main и func main()

```go
package main

import (
  "log"

  "github.com/vrischmann/envconfig"
)

func main() {
  var conf struct {
     Name string `envconfig:"myName"`
  }
  if err := envconfig.Init(&conf); err != nil {
     log.Fatalf("could not init envconfig: %s", err)
  }
}
```

Библиотека не содержит package main

```
package greeter

func Greet(name string) string {
  return "Hello, " + name
}
```

Импортирование пакетов

```
package main

import (
  "fmt"

  "github.com/alikhanmurzayev/test_project/greeter"
)

func main() {
  fmt.Println(greeter.Greet("Alikhan"))
}
```

One-shot запуск

```go
package main

import (
  "fmt"
  "github.com/alikhanmurzayev/test_project/greeter"
)

func main() {
  fmt.Println(greeter.Greet("Alikhan"))
}
```

```
$ go run main.go
```

```
alikhan@cerebro:~/go/src/github.com/alikhanmurzayev/test_project$ go run main.go
Hello, Alikhan
```

Компиляция в исполняемый файл

$ go build -o main.bin

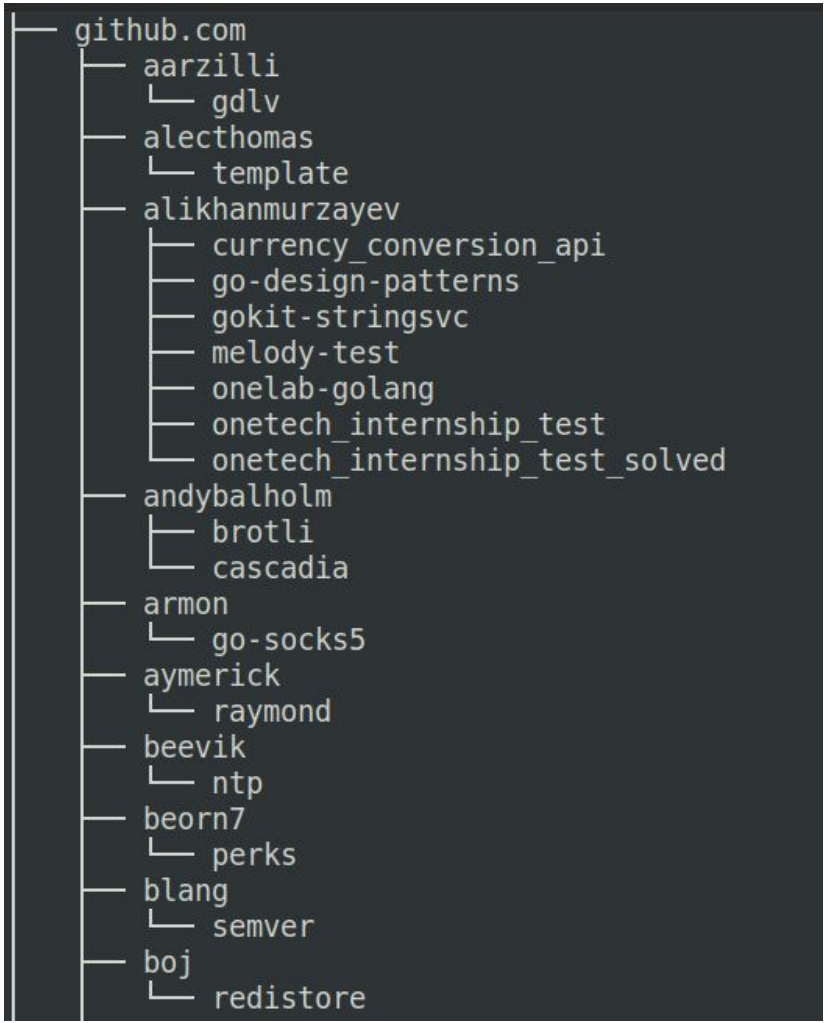
```
alikhan@cerebro:~/go/src/github.com/alikhanmurzayev/test_project$ go build -o main.bin
alikhan@cerebro:~/go/src/github.com/alikhanmurzayev/test_project$ file main.bin
main.bin: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked, Go BuildID=67PSVVP8dSjWFxNkdARE/RV6TMCyEzJVRZcWKWr3G/b00URFCJ_7cS7nAZ9u9s/PKyhWx8RpArR8IwmJkA4, not stri
pped
alikhan@cerebro:~/go/src/github.com/alikhanmurzayev/test_project$
```

Существует 2 способа организации кода

1. Единое дерево кода
2. Go modules

Единое дерево кода

$GOPATH/go/src



Для работы с единым деревом кода нужно:

- Создать проект github.com/username/projectname
- Скачать проект в GOPATH с помощью go get github.com/username/projectname/...
- Изменять, компилировать, и комитить проект из $GOPATH/src/github.com/username/projectname

Преимущества:

- Простота, плоская структура

Недостатки:

- Отсутствие версий

Задание

1. Создать локально проект для библиотеки с функцией Sum:

```
func Sum(a, b int) int {
  return a + b
}
```

2. Создать локально проект для исполняемой программы, которая будет использовать библиотеку

Go modules

Модуль — набор go-пакетов.

В корневой директории модуля находится go.mod файл.

Если в директории есть go.mod и внутри где-то есть go-пакеты, то директория считается модулем.

Создание модуля

```
$ go mod init github.com/username/projectname
```

```
alikhan@cerebro:~/Desktop/test_project$ go mod init github.com/alikhanmurzayev/test_project
go: creating new go.mod: module github.com/alikhanmurzayev/test_project
go: to add module requirements and sums:
        go mod tidy
alikhan@cerebro:~/Desktop/test_project$ cat go.mod
module github.com/alikhanmurzayev/test_project

go 1.16
alikhan@cerebro:~/Desktop/test_project$
```

Добавление зависимостей

```
$ go get github.com/kelseyhightower/envconfig
```

```
alikhan@cerebro:~/Desktop/test_project$ go get github.com/kelseyhightower/envconfig
go get: added github.com/kelseyhightower/envconfig v1.4.0
alikhan@cerebro:~/Desktop/test_project$ cat go.mod
module github.com/alikhanmurzayev/test_project

go 1.16

require github.com/kelseyhightower/envconfig v1.4.0 // indirect
alikhan@cerebro:~/Desktop/test_project$
```

Преимущества

- Версионирование
- Возможна работа вне GOPATH
- Защита от подмены модулей

go get ведет себя по разному:

- При вызове в модуле скачивает в

  $GOPATH/pkg/mod

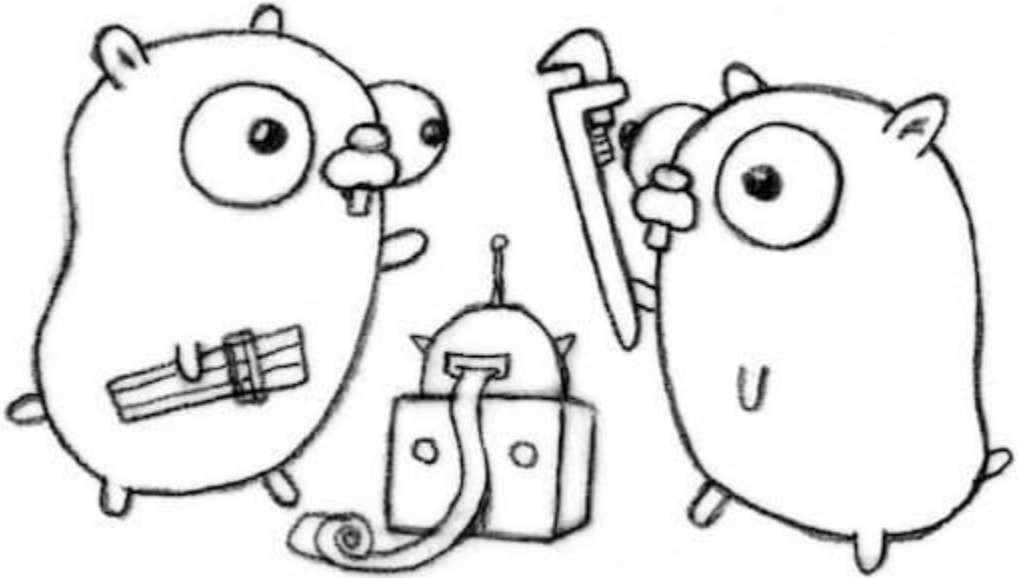
- При вызове вне-модуля скачивает в

  $GOPATH/src

Задание

Внедрить модули в ранее написанные программы (в библиотеку и исполняемую программу)

Утилиты



Получение справки

```
$ go help
```

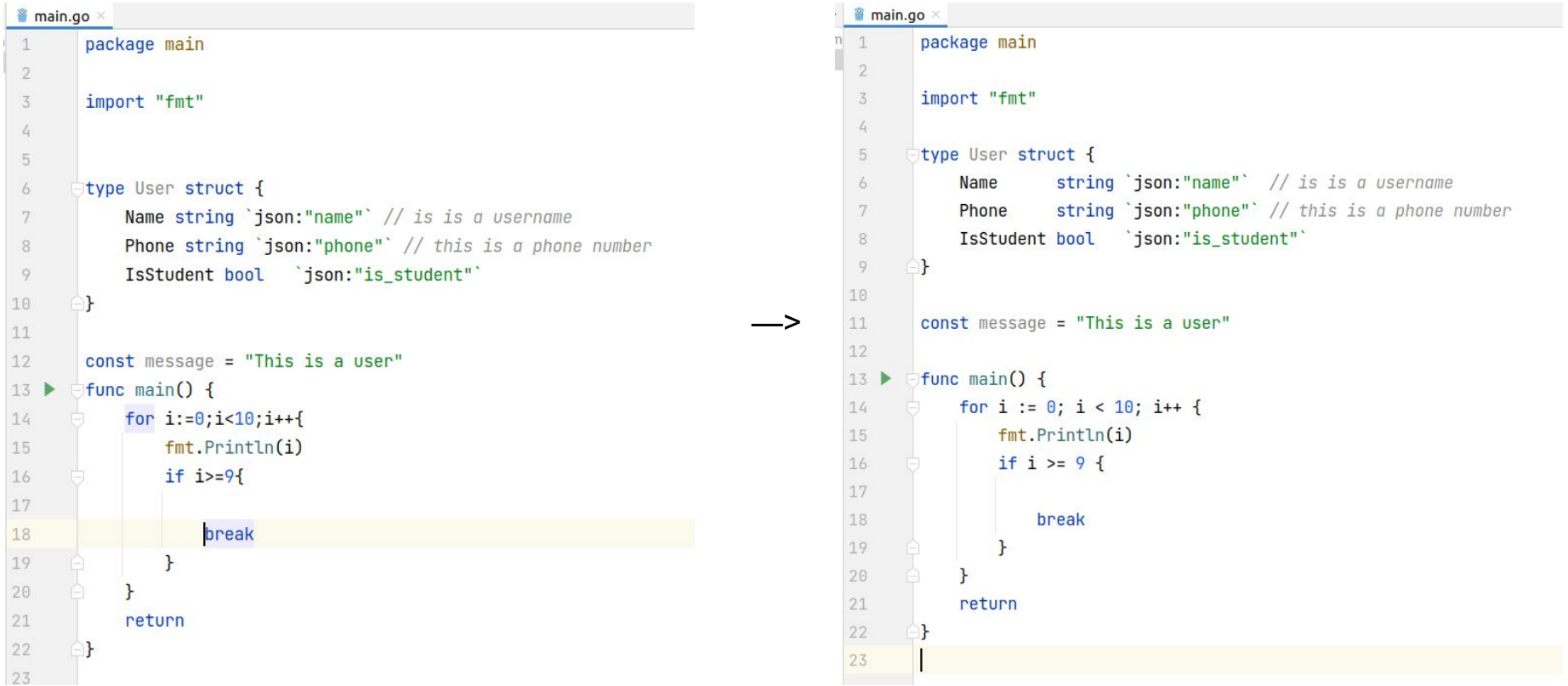
```
alikhan@cerebro:~/go/src/github.com/alikhanmurzayev/test_project$ go help run
usage: go run [build flags] [-exec xprog] package [arguments...]

Run compiles and runs the named main Go package.
Typically the package is specified as a list of .go source files from a single directory,
but it may also be an import path, file system path, or pattern
matching a single known package, as in 'go run .' or 'go run my/cmd'.

By default, 'go run' runs the compiled binary directly: 'a.out arguments...'.
If the -exec flag is given, 'go run' invokes the binary using xprog:
        'xprog a.out arguments...'.
If the -exec flag is not given, GOOS or GOARCH is different from the system
default, and a program named go_$GOOS_$GOARCH_exec can be found
```

Форматирование кода:

```
$ go fmt main.go
```



Обновление импортов

```go
package main

func main() {
        fmt.Println(strings.Index("hello", "l"))
}
```

```go
package main

import (
        "fmt"
        "strings"
)

func main() {
        fmt.Println(strings.Index("hello", "l"))
}
```

Скачаем и установим утилиту ручками

1. Необходимо обновить переменную PATH в ~/.bashrc:

```
export PATH=$PATH:$GOPATH/bin
```

Скачаем и установим утилиту ручками

2. Скачаем библиотеку в $GOPATH/src:

$ go get -d golang.org/x/perf/cmd/benchstat

С флагом -d библиотека только скачается

```
alikhan@cerebro:~/go/src/golang.org/x/tools/cmd$ go get -d golang.org/x/perf/cmd/benchstat
go get: added golang.org/x/perf v0.0.0-20210220033136-40a54f11e909
alikhan@cerebro:~/go/src/golang.org/x/tools/cmd$
```

Скачаем и установим утилиту ручками

3. Установим бинарник в $GOPATH/bin

$ go install golang.org/x/perf/cmd/benchstat

Скачаем и установим утилиту ручками

4. Проверка

```
alikhan@cerebro:~$ benchstat
usage: benchstat [options] old.txt [new.txt] [more.txt ...]
options:
  -alpha α
        consider change significant if p < α (default 0.05)
  -csv
        print results in CSV form
  -delta-test test
        significance test to apply to delta: utest, ttest, or none (d
st")
  -geomean
        print the geometric mean of each file
  -html
        print results as an HTML table
  -norange
        suppress range columns (CSV only)
  -sort order
        sort by order: [-]delta, [-]name, none (default "none")
  -split labels
        split benchmarks by labels (default "pkg,goos,goarch")
alikhan@cerebro:~$
```

ДЗ

1. Почитать про семантическое версионирование (https://semver.org/, https://www.geeksforgeeks.org/introduction-semantic-versioning/)

2. Прочитать серию статей https://blog.golang.org/using-go-modules

3. Повторить примеры из туториала https://golang.org/doc/tutorial/create-module

ДЗ — продолжение

4. Написать библиотеку и программу:

Библиотека:

- v1.0.0 - функция для смены регистра символов (upper to lower, lower to upper)
- v1.0.1 - добавить функцию для нахождения корней квадратного уравнения
- v1.0.2 - добавить функцию для получения UUID (использовать любую библиотеку из списка https://github.com/avelino/awesome-go#uuid)

Исполняемая программа:

- v1.0.0 - использовать функцию для смены регистра
- v1.0.1 - использовать функцию для нахождения корней квадратного уравнения
- v1.0.2 - использовать функцию для получения UUID