OTUS

Специальная цена

Разработчик Golang

Разработка сетевых приложений и микросервисов на Go

Длительность курса: 166 академических часов

1 Начало работы с Go

1 **Начало работы с Go**

* Установка рабочего окружение Go
* Основные синтаксические конструкции
* Публичные и приватные идентификаторы
* Импорт пакетов
* Алиасы и импорт без использования
* init функции и порядок инициализации пакетов
* internal пакеты
* Рекомендованная структура проекта

Домашние задания

1 Hello now()

Завести Go репозиторий на GitHub, написать программу печатающую текущее время / точное время с использованием библиотеки NTP. Программа должна корректно обрабатывать ошибки библиотеки: распечатывать их в STDERR и возвращать ненулевой код выхода.

---

2 **Инструментарий Go, тестирование**

* Подробнее про GOROOT и GOPATH
* Сборка модулей и установка программ: go get, go build, go install
* Кросс-компиляция
* Модули и зависимости: go mod
* Форматирование кода: go fmt, imports
* Линтеры: go vet, golint, металинтеры
* Тестирование Go программ
* Пакет testing
* Паттерны и анти-паттерны unit-тестирования

---

3 **Элементарные типы данных в Go**

* Элементарные типы
* Zero value
* Константы
* Указатели
* Строки и руны и массивы байтов
* Стандартный функции для работы со строками и Unicode
* Преобразование и присвоение типов
* Передача по ссылке и значению

Домашние задания

1 Распаковка строки

Создать Go функцию, осуществляющую примитивную распаковку строки, содержащую повторяющиеся символы / руны, например:

* "a4bc2d5e" => "aaaabccddddde"
* "abcd" => "abcd"
* "45" => "" (некорректная строка)

Дополнительное задание: поддержка escape - последовательности
* `qwe\4\5` => `qwe45` (*)
* `qwe\45` => `qwe44444` (*)
* `qwe\\5` => `qwe\\\\\` (*)

4 **Слайсы и словари**

* Отличия массивов и слайсов
* Внутренняя структура слайсов и словарей
* Создание и работа со слайсами и словарями
* Отличие длинны и емкости (length vs capacity)
* Различные способы итерации
* Сортировка
* Распространные ошибки и затруднения

Домашние задания

1 Частотный анализ

Написать функцию, которая получает на вход текст и возвращает
10 самых часто встречающихся слов без учета словоформ

---

5 **Функции и обработка ошибок**

* Функции и области видимости
* Замыкания
* Функции с переменным числом аргументов
* Типы ошибок
* panic, recover и defer
* Стек-трейсы
* Best practice обработки ошибок

---

6 **Структуры**

* Определение структур
* Вложенные и анонимные структуры
* Тэги элементов структуры
* Использование тэгов для JSON и работы с СУБД
* Публичные структуры и публичные элементы структур
* Методы типов

Домашние задания

1 Двусвязный список

Цель:
https://en.wikipedia.org/wiki/Doubly_linked_list

Ожидаемые типы (псевдокод):

```
List // тип контейнер
Len() // длинна списка
First() // первый Item
Last() // последний Item
PushFront(v interface{}) // добавить значение в начало
PushBack(v interface{}) // добавить значение в конец
Remove(i Item) // удалить элемент

Item // элемент списка
Value() interface{} // возвращает значение
Nex() *Item // следующий Item
Prev() *Item // предыдущий
```

Реализовать двусвязанный список на языке Go

---

7 **Интерфейсы**

* Определение и реализация интерфейсов
* Внутренняя структура интерфейсов
* Интерфейсы как "универсальный" тип
* Определение типа значения интерфейса
* Опасный и безопасный type cast
* Конструкций switch
* Слайсы и словари с интерфейсами
* Где мои generic-и?

---

8 **Горутины и каналы**

* Запуск горутин
* Каналы, внутреняя структура канала
* Буферизованные и небуферизованные каналы
* Операции работы с каналами
* Использование каналов для передачи данных и синхронизации
* Конструкция select
* Таймеры в Go

Домашние задания

1 Параллельное исполнение

Написать функцию для параллельного выполнения N заданий (т.е. в N параллельных горутинах).

Функция принимает на вход:
- слайс с заданиями `[]func() error`;
- число заданий которые можно выполнять параллельно (`N`);
- максимальное число ошибок после которого нужно приостановить обработку.

Учесть что задания могут выполняться разное время.

---

9 **Примитивы синхронизации**

* Мьютексы
* Условные переменные
* Гарантировано одноразовое выполнение
* Pool и WaitGroup
* Модель памяти в Go
* Race-детектор

2 Стандартные библиотеки и практики

1 **Работа с вводом/выводом**

* Тип Buffer
* Стандартные интерфейсы: Reader, Scanner, Writer, Closer
* Блочные устройства, Seeker
* Форматированный ввод и вывод: fmt

Домашние задания

1 Копирование файлов

Цель: Реализовать утилиту копирования файлов (см man dd).

Выводить в консоль прогресс копирования в %.
Должна быть возможность скачать протестировать и установить программу с помощью go get/test/install
Программа должна проходить проверку go vet и golint

---

2 **Форматирование данных**

* base64 и другие форматы строк, кодировки
* Текстовые форматы: JSON, XML, YAML
* Использование стуктур и интерфейсов для парсинга
* Бинарные форматы: MsgPack и ProtocolBuffers

---

3 **Взаимодействие с OS**

* Обработка аргументов командной строки: flags, cobra
* Работа с переменными окружения
* Запуск внешних программ
* Работа с файловой системой
* Временные файлы
* Обработка сигналов

Домашние задания

1 Утилита envdir

Реализовать утилиту envdir на Go.
Эта утилита позволяет запускать программы получая переменные окружения из определенной директории. См man envdir

---

4 **Файлы конфигурации и логирование**

* Различные варианты конфигурации
* Использование простых форматов: ini, yaml и т.п.
* Библиотеки для работы с конфигурацией: viper и confita
* Стандартная библиотека для логирования
* Расширенное логирование с помощью Zap

Домашние задания

1 Каркас микросервиса

Реализовать "каркас" микросервиса, считывающий конфиг из файла, создающий логгер/логгеры с указанными уровнями детализации.

5	**Рефлексия**	* Использование пакета reflect * Type и Value * Использование рефлексии совместно с type switch * Пакет unsafe и тип unsafe.Pointer * Пример использования рефлексии
6	**Кодогенерация в Go**	* Примеры применения кодогенерации * Команда go generate * Генерация кода по Protobuf спецификациям Домашние задания 1 Простейший календарь Сделать "заготовку" для микросервиса-календаря. Определить структуру определяющую событие, написать методы для добавления/изменения/удаления событий. Хранить события в памяти, без персистентности.
7	**Go Internals - Память**	* Структура памяти в Linux процессе * Особенности памяти в Go приложении * Выделение памяти на стеке * Выделение и освобождение памяти в куче * Механизм сборки мусора в Go

8 **Профилирование и оптимизация Go программ**

* Общая информация о профилировании
* Виды профайлеров
* Профилирование использования CPU
* Профилирование выделения памяти
* Flame диаграммы
* Общие оптмизации в Go
* Оптимизации выделения памяти

Домашние задания

1 Оптимизация производительности

Провести анализ производительности программы из дз 2.6.
Построить Flame диаграммы и выявить наиболее медленные фрагменты.
Попытаться улучшить производительность.

---

9 **Go Internals - Планировщик**

* Основные структуры планировщика: P, M, G
* Механизм переключения горутин
* Системные вызовы в Go
* Сетевые вызовы в Go

3 Сетевое взаимодействие

1 **Низкоуровневые протоколы TCP, UDP, DNS**

* Стандартные интерфейсы - Dialer, Conn
* Возможные сетевые проблемы: потеря пакетов, подвисшие соединения, недоступность
* Таймауты и контекст
* Отладка сетевых проблем

Домашние задания

1 Telnet

Реализовать telnet клиент (см man telnet)

---

2 **HTTP библиотека**

* Использование HTTP клиента
* Создание простого HTTP сервера
* Декораторы и middleware
* HTTP/1.1 и HTTP/2.0
* REST
* Протокол S3

---

3 **GRPC**

* Описание API с помощью Protobuf
* Генерация кода для GRPC клиента и сервера
* Реализация API
* Прямая и обратная совместимость API
* Представление о Clean Architecture

Домашние задания

1 GRPC сервис

Создать GRPC спецификацию для сервиса-календаря из дз 2.7.
Сгенерировать GRPC сервер и клиент, проверить работу сервиса.
Отделить модель данных от Protobuf структур.

---

4 **Работа с SQL**

* Стандартные интерфейсы sql.DB, sql.Rows, sql.Tx
* Подключение к СУБД и настройка пула подключений
* Выполнение запросов и получение результатов
* Простейшие SQL запросы

Домашние задания

1 Работа с базами данных

Изменить код сервиса-календаря, так что бы события хранились в базе данных.

---

5 **NoSQL базы данных**

* Встроенные базы данных, BoltDB
* Использование Redis для кеширования данных
* Выбор базы данных для проекта

---

6 **Clean Architecture**

---

7 **Очереди сообщений**

* Использование RabbitMQ, Kafka
* Возможные проблемы с очередями: перегрузка, падение обработчиков, сбойные сообщения
* Запуск пула обработчиков с помощью systemd или supervisor

Домашние задания

1 Работа с очередями

Реализовать "напоминания" о событиях с помощью RabbitMQ.
Создать процесс, который периодически сканирует основную базу данных, выбирая события о которых нужно напомнить. Создать процесс, который читает сообщения из очереди и шлет уведомления.

---

8 **Web-Sockets**

4 Микросервисы

1 **Монолит и микросервисы**

* Отличия микросервисов от монолитной архитектуры
* Преимущества микросервисов
* 12 factor applications
* Архитектура одного микросервиса
* Clean Architecture

Домашние задания

1 Доработка сервиса

Доработать дз 3.8, разделив код на отдельные сервисы

---

2 **Docker**

* Основные команды docker
* Представление о контейнер и образе
* Сборка собственных Docker контейнеров
* Запуск и управление контейнерами
* Передача конфигурации и получение логов
* Использование Docker Registry

Домашние задания

1 Докеризация сервиса

Подготовить Docker файлы для каждого из сервисов из дз 3.8.
Собрать образы и загрузить в DockerHub

---

3 **Тестирование микросервисов**

* Использование docker-compose
* Интеграционные тесты
* Behaviour Driven Development
* Описание тестов на языке Gherkin
* Использование godog для запуска тестов

Домашние задания

1 Интеграционное тестирование

Создать docker-compose файл, поднимающий все сервисы проекта, включая базу и очередь. Написать простейший интеграционный тест для проекта.

---

4 **Мониторинг**

* Проверка работоспособности сервисов
* Health Check в Docker
* Системные метрики: LA, CPU, MEM, IO
* Количественный мониторинг, prometheus

Домашние задания

1 Мониторинг сервиса

Обеспечить простейший мониторинг проекта с помощью prometheus

---

5 **Использование CI**

* GitLab pipelines

Домашние задания

1 Continuous integration

Автоматизировать тестирование и выкатку новых версий с помощью CI

---

6 **Kubernetes**

* Docker Swarm
* Puppet
* Ansible
* Как мы решаем проблему запуска сервсисов на более 1 машины

Домашние задания

1 Разворачивание в kubernetes

Развернуть проект в Kubernetes. Создать описания pod-ов и service-ов.
Избавиться от hard-code IP в конфигах проектов.

5 Проект

1	**Проектная работа**	Домашние задания 1Проект
2	**Консультация по проектам**	
3	**Презентация проектов**	