GO-01 01: Введение и подготовка окружения

Описание:

Вначале мы расскажем вам о том, для чего был создан язык Go, каковы сферы его применения, а также научимся устанавливать Go, чтобы вы смогли создать свою первую программу на новом языке.
Введение и подготовка окружения
В то время, когда ООП (объектно-ориентированное программирование) как способ организации кода прочно закрепилось в практике программирования еще с конца 70-х годов прошлого века, современное программирование от этого не стало проще. Огромная скорость развития автоматизации всевозможных отраслей экономики предъявляет серьезные требования всему ИТ-сообществу. Основные вызовы, которые брошены программированию как отрасли, это:

- увеличение скорости процесса разработки прикладных решений;
- уменьшение времени вывода готовых решений в боевую среду;
- оптимизация быстродействия для работы под большими нагрузками и утилизации «железных» ресурсов.

От скорости разработки зависит то, как быстро мы сможем реализовать идею в высококонкурентной бизнес-среде. От скорости разработки, совместно со временем ввода решения в эксплуатацию, зависит то, насколько качественной будет поддержка нашего решения (исправление багов, доработка новой функциональности). От оптимизации созданного нами решения зависит то, как наш продукт будет работать в условиях высоких нагрузок и масштабирования.
Поиски решения для первых двух задач дали понять, что те огромные монолитные проекты, которые появились благодаря удобству и структуризации ООП, не могут соответствовать современным требованиям. Все потому, что каждая новая доработка или каждое исправление ошибки порождает новый каскад связанных задач. Это случается из-за того, что очень часто сроки, в которые надо было реализовать функционал, не позволяли как следует продумать решение и написать код, следуя хорошим практикам. Или перед крупным релизом пропустили на code-review задачу, где закрыли глаза на небольшое дублирование кода, а сами в своем релизном функционале не успели написать тесты. Согласитесь, что это не самые редкие случаи в жизни программиста. Таким образом, со временем может оказаться, что разросшаяся за годы функциональность превратила наш проект в огромного и неповоротливого монстра, который отнимает все больше ресурсов на поддержку.
Решением, которое стало вполне естественным и было хорошо принято сообществом, явилась декомпозиция огромной функциональности на микросервисы с четко определенной зоной ответственности.

Анализ проблемы быстродействия показал, что нет инструмента, который бы максимально эффективно использовал многопроцессорные системы, начавшие стремительно развиваться в начале XXI века.
Итогом работы над решением задач по созданию такого инструмента стал язык Go, который появился в компании Google в 2009 году.
Сферы применения языка Go обусловлены идеей его происхождения: быстрый, простой инструмент для параллельного программирования. Именно поэтому он наиболее применим при создании сетевых приложений, консольных утилит, систем мониторинга и, конечно, в web-программировании для разработки back-end. По причине того, что web-разработка находится на стыке системного, сетевого и прикладного программирования, Go здесь очень хорошо прижился.
На сегодняшний день есть примеры успешной реализации больших и сложных систем, написанных на Go:

- Kubernetes - система для развертывания, масштабирования и управления контейнерами;
- Prometheus - современная система мониторинга.

Кроме этого, являясь языком общего назначения, он становится популярной заменой языкам с динамической типизацией (например, PHP).
Все потому, что сложность систем возрастает и преимущества динамической типизации нивелируются проблемами, которые возникают в боевой среде из-за потери контроля над типами данных.
Также стоит упомянуть, что Go обладает высокой скоростью компиляции - это не только повышает эффективность работы программиста, но и сокращает время развертывания приложения при переносе в боевую среду.
Установка
Рассмотрим способы установки Go на разных системах. Далее будут приведены команды, которые необходимо выполнить в терминале ОС.
Ubuntu
Для 18.04 LTS потребуется добавить PPA репозиторий, после чего мы сможем установить Go версии 1.14. Если устанавливать из стандартного репозитория, то максимальная доступная версия будет 1.10.

```
sudo add-apt-repository ppa:longsleep/golang-backports
sudo apt update
sudo apt install golang-go
```

Для Ubuntu 20.04 LTS версия 1.14 доступна из стандартного репозитория, поэтому можно сразу выполнить установку.

```
sudo apt install golang-1.14-go
```

Mac OS X
Самым удобным способом установки будет менеджер пакетов brew.

```
$ brew update
$ brew install go
```

Windows
Необходимо перейти по ссылке golang.org и скачать MSI-дистрибутив для установки. После этого запустить и следовать инструкциям. По умолчанию Go будет установлен в C:\Go.
Следующим шагом следует провести настройку GOPATH. GOPATH - это путь к вашему рабочему пространству, в котором будут располагаться скомпилированные бинарные файлы, исходный код проектов и вообще все, с чем вы будете работать.

```
echo "export GOPATH=~/go" >> ~/.profile
```

Вместо файла .profile могут быть .bash_profile, .zshrc или любой другой файл настройки терминала.
Чтобы переменная GOPATH стала доступна, нужно перечитать файл настроек терминала. Например, так:

```
source ~/.zshrc
```

Далее рекомендуется создать директории рабочего пространства.

```
mkdir -p $GOPATH $GOPATH/src $GOPATH/pkg $GOPATH/bin
```

- /src - в этой директории будет находиться исходный код проектов;
- /pkg - здесь будут располагаться скомпилированные внешние зависимости;
- /bin - в этой директории будут лежать скомпилированные бинарные файлы.

Также очень полезно будет добавить $GOPATH/bin в переменную PATH, так как в процессе работы может потребоваться воспользоваться какими-то go-программами. Поэтому лучше сразу настроить среду, чтобы в процессе работы нас ничего не отвлекало. Добавим в файл настройки терминала команду и перечитаем его.

```
echo "export PATH=$PATH:$GOPATH/bin" >> ~/.profile
```

Следует также упомянуть о GOROOT. GOROOT - это переменная, в которой содержится путь к директории установки Go. Эта переменная устанавливается автоматически при установке Go посредством установщика или менеджера пакетов. Однако, если вы устанавливаете Go вручную из архива или выполняете сборку из исходного кода, то эту переменную следует установить самостоятельно.
До версии Go 1.10 эта переменная требовалась при сборке. В более новых версиях необходимость в настройке этой переменной отпала, так как это значение устанавливается при установке Go. Однако знание о назначении переменной GOROOT может понадобиться при настройке IDE на работу с Go.
Далее в практикуме будет описана работа с модулями и необходимость в строгом следовании правилам работы с GOPATH станет необязательна. Но эти знания необходимы, чтобы иметь базовое понимание о работе с Go.
Проверка установки и первая программа
Теперь проверим нашу установку. Выполним в терминале команду.

```
go version
```

В выводе команды мы должны увидеть версию Go.

```
go version go1.14.2 linux/amd64
```

Для компиляции и запуска программ есть две команды:

- go run - скомпилирует программу во временном хранилище и запустит;
- go build - скомплирует бинарный файл.

Создадим файл $GOPATH/src/hello.go со следующим содержимым:

```
package main

import "fmt"

func main() {
     fmt.Println("Hello from Rebrain!")
}
```

Откроем в терминале $GOPATH/src и выполним команду:

```
go run hello.go
```

После этого в консоли вы сможете увидеть результат работы вашей первой программы на Go.

IDE для разработки

Стоит упомянуть об инструментах для разработки на Go. На рынке программного обеспечения представлено много разных инструментов: удобных и не очень, платных и бесплатных. Но в процессе выбора инструмента следует уделить внимание тем удобствам и возможностям, которые предоставляет IDE. На сегодня большую популярность имеют два инструмента:

- Visual Studio Code — бесплатный редактор от компании Microsoft, который очень удобно и быстро расширяет свои возможности с помощью большого числа плагинов, размещенных в стандартном маркетплейсе;
- GoLand — коммерческая IDE от компании JetBrains, которая предназначена исключительно для разработки на Go и в стандартной поставке имеет все необходимые инструменты для работы.

Помимо озвученных инструментов есть немалое количество других редакторов, которые предлагают в той или иной степени сравнимые возможности, поэтому вы свободно можете выбирать любой.

Полезные ссылки:

- golang code style
- Go in Visual Studio Code
- GoLand documentation

Задание:

1. Установите Go и настройте рабочее пространство.
2. Ознакомьтесь со стилем написания кода на Go (code style) и утилитой для форматирования gofmt.
3. Создайте файл main.go и напишите код, который выводит приветствие и текущие дату и время в виде строки ДД.ММ.ГГГГ ЧЧ:ММ.
    - Для вывода строки нужно воспользоваться стандартным пакетом fmt.
    - Для получения даты и форматирования потребуется функция из стандартного пакета time.
4. Подробно концепция пакетов и работа с ними будут рассмотрены позже. На данном этапе следует ознакомиться с функциональностью указанных пакетов.
5. Скомпилируйте ваш код и запустите полученный бинарный файл. В качестве ответа пришлите исходный код программы.