Язык Go. Основные возможности и применение

Работа
студента 411 группы И. И. Королёва
и студента 451 группы Д. А. Емелина

Саратовский государственный университет
им. Н. Г. Чернышевского

Кафедра математической кибернетики
и компьютерных наук

Научный руководитель: доцент Иванова А. С.

2024

Введение

Go — относительно молодой, но популярный язык программирования.

Цель, которая стояла перед создателями Go — разработать простой и эффективный язык программирования, который мог бы использоваться для создания качественного программного обеспечения.

Golang получил третье место в рейтинге языков программирования, которые хотели бы освоить разработчики.

Мотивация

- Первый процессор Pentium 4 с тактовой частотой 3,7 ГГц был представлен еще в 2005 году корпорацией Intel.
- Macbook Pro 2023 имеет тактовую частоту 4 ГГц.

Итог — почти за два десятилетия мощности не слишком-то изменились.

Для увеличения производительности решили:

- добавлять в процессоры все больше и больше ядер;
- ввести гиперпоточность (концепция параллельного выполнения задач);
- добавить больше кэша.

Мотивация

Если же нельзя положиться на усовершенствование оборудования, единственный выход — более эффективное ПО для повышения производительности.

По итогу выходит, что разрабатываемое нами ПО и языки программирования должны поддерживать параллелизм и быть расширяемыми в условиях постоянного увеличения количества ядер.

Большинство современных языков программирования (таких как Java, Python) поддерживают многопоточность. Но настоящая проблема связана с одновременным исполнением, блокировкой потоков, состоянием гонки и взаимоблокировками.

Мотивация

Возьмем, к примеру, Java.

Каждый канал потребляет около 1 Мб объема памяти, и, в конце концов, если вы задействуете тысячи потоков, все может закончиться нехваткой памяти. Кроме того, взаимодействие между двумя или несколькими потоками – это тоже непросто.

Язык Go (он же Golang) появился в 2009 году, когда уже были многоядерные процессоры. Это позволило не реализовывать параллельные механики поверх уже имеющейся спецификации языка, а разработать язык сразу с учетом параллелизма.

Преимущества языка. Простота

1. Простота

У Go достаточно простой синтаксис (с определенными допущениями), поэтому приложения можно разрабатывать быстрее, чем на некоторых других языках.

Golang достаточно быстро может изучить как полный новичок в программировании, так и уже «сформировавшийся программист», тот, кто уже знает один или несколько языков.

Преимущества языка. Простота

Go специально не учитывает многие особенности современных языков ООП в угоду простоте кода и удобству его поддержки:

1. Нет классов. Все разделяется на пакеты. Go работает со структурами, а не с классами.
2. Не поддерживает наследование. Это упрощает изменение кода. Без наследования Go также становится легко читаемым: нет суперклассов, которые следует изучать особо тщательно.
3. Нет аннотаций (метаданных, прикрепляемых к коду для различных целей).
4. Нет конструкторов.
5. Нет исключений.

Преимущества языка. Простота

Вышеизложенные изменения делают Golang отличным от других языком, а программирование на Go – предельно простым.

В отличие от прочих новых языков, таких как Swift, синтаксис Go стабилен. Он остался прежним с первого выпуска версии 1.0, что состоялся в 2012 году. Это делает его обратно совместимым.

Преимущества языка. Горутины

2. Горутины

У Go есть goroutine вместо потоков. Они потребляют только 2 Кб памяти. Таким образом, можно в любой момент активировать миллионы горутин.

Преимущества языка. Горутины

Горутины — это функции, которые могут работать параллельно, то есть программа выполняет несколько строк практически одновременно. Чтобы сделать из функции горутину, надо просто написать перед ней ключевое слово go.

```
func server(i int) {
  for {
    print(i)
    time.Sleep(10)
  }
}
go server(1)
go server(2)
```

Преимущества языка. Горутины

Результат — практически одновременный вызов, несмотря на задержку time.Sleep(10), обеих горутин. Конечно, в небольшой программе это делать практически бессмысленно, а вот при вызове множества функций — очень даже оправданно. Экономится время, и ресурсы процессора используются равномерно.

За выполнением горутин в Go следит специальная библиотека времени исполнения: она распределяет между ними ядра процессора, может ограничивать число доступных ядер. Библиотека помогает запускать огромное количество горутин — намного больше, чем позволяет операционная система, — и не требует от программиста заниматься распараллеливанием вручную.

Преимущества языка. Каналы

Это что-то вроде общего хранилища данных. Каналы передаются как аргументы горутин и помогают им общаться между собой и обмениваться данными. В каналах есть очередь и блокировка — чтобы разные горутины не смогли одновременно закинуть туда разные данные. Особенность каналов: они позволяют записывать и считывать только один тип данных. Например, int — целые числа.

```
func main() {
    channel := make(chan float32)

    fmt.Printf("type of 'c' is %T\n", channel)
    fmt.Printf("value of 'c' is %v\n", channel)
}
```

Преимущества языка. Горутины

Другие преимущества:

1. Горутины используют больше памяти только тогда, когда это необходимо.
2. Также они запускаются быстрее, чем потоки.
3. Более того, они идут вместе со встроенными примитивами, чтобы безопасно обмениваться данными.
4. В условиях одновременного использовании структур данных не придется прибегать к блокировке мьютексов.
5. 1 горутина может свободно работать на нескольких потоках. Горутины мультиплексируются в небольшое количество потоков ОС.

Преимущества языка. Работа с памятью

3. Работа с памятью и компиляция

Одним из наиболее значительных преимуществ языков C и C++ над другими современными языками, такими как Java/Python, является их производительность. Дело в том, что C и C++ вместо интерпретации компилируются.

Как и языки низкого уровня, Go является компилируемым. Это означает, что производительность почти такая же высокая, как и в низкоуровневых языках. А еще он использует сборщик мусора для выделения и удаления объекта.

4. Чистота кода и статическая типизация

Компилятор Go позволяет держать код «чистым». К примеру, неиспользуемые переменные считаются ошибкой компиляции. В Go решается большая часть проблем форматирования. Это делается, к примеру, при помощи программы gofmt при сохранении или компиляции. Форматирование правится автоматически.

Язык программирования Go имеет статическую, строгую типизацию. Это означает, что типы данных определяются на этапе компиляции и проверяются на соответствие во время выполнения программы.

Преимущества языка. Не нужны фреймворки

5. Не нужны фреймворки

В Go отсутствует традиционное понятие фреймворков, которые часто используются в других языках программирования по типу Python, Ruby, JavaScript и т. д. Вместо этого в Гоу для создания приложений применяются модули и библиотеки.

Разработчики используют библиотеки и инструменты, чтобы строить мощные и надежные Go-приложения, на 100% подходящие для решения задач, поставленных перед ними.

Преимущества языка. Не нужны фреймворки

Почему отсутствие фреймворков — это плюс для разработчика?

- Гибкость.
  Вместо принудительного использования фреймворков, Go предоставляет разработчикам свободу выбора библиотек и инструментов в зависимости от каждого проекта. Это позволяет строить приложения, оптимизированные под решение конкретных задач.
- Модульность.
  Программирование на Go — модульный подход к разработке. Этот подход позволяет импортировать сторонние библиотеки и использовать только те функции, которые действительно нужны в процессе разработчик. То есть модульность минимизирует зависимости.

Преимущества языка. Не нужны фреймворки

Почему отсутствие фреймворков — это плюс для разработчика?

- Множество библиотек.
  Сообщество Golang активно разрабатывает и поддерживает много полезных библиотек и инструментов, которые можно использовать в проектах без необходимости использования фреймворков. Эти библиотеки позволяют реализовать функциональные возможности по типу маршрутизации HTTP, работы с БД, обработки форм и т. п.
- Производительность.
  Go изначально разрабатывался с акцентом на высокую скорость работы. Отсутствие больших и тяжеловесных фреймворков способствует более низкому потреблению ресурсов рабочей машины и обеспечивает более быстрый запуск приложений.

6. Маппинги

Маппинги (Maps) в языке программирования Go представляют собой встроенную структуру данных, которая представляет собой набор пар ключ-значение, где каждый ключ уникален в рамках маппинга. Они известны также как ассоциативные массивы или словари в других языках программирования.

```
Database struct {
    URL      string `yaml:"url"`
    DBName   string `yaml:"dbname"`
    Username string `yaml:"username"`
    Password string `yaml:"password"`
}
```

Преимущества языка. Маппинги

Основные характеристики маппингов в Go

1. Уникальные ключи: каждый ключ в маппинге уникален, что означает, что в маппинге не может быть двух элементов с одинаковыми ключами.
2. Неупорядоченность: элементы в маппинге не хранятся в определенном порядке. Это означает, что порядок элементов при обходе маппинга не гарантирован и может меняться между разными запусками программы.
3. Динамическое изменение размера: маппинги в Go могут динамически увеличиваться или уменьшаться по мере добавления или удаления элементов.

Что можно написать на Go?

На Go можно написать практически все, за исключением некоторых моментов (например, разработки, связанные с машинным обучением — здесь больше подходит все же Python с низкоуровневыми оптимизациями на C/C++ и CUDA). Основная специализация:

- Веб-приложения: Go имеет нативную поддержку для создания веб-серверов и веб-приложений. С помощью популярных фреймворков, таких как Gin, Echo, или net/http, можно легко создавать веб-сервисы, API, сайты и другие веб-приложения.
- Микросервисы: Go является популярным выбором для разработки микросервисов благодаря своей производительности, эффективности и поддержке конкурентности. Многие крупные компании используют Go для создания своей микросервисной архитектуры.

Недостатки языка Go

- Ограниченная область применения. Язык больше подходит для сетевых и серверных приложений, чем для десктопных. Также он не имеет поддержки для создания графических интерфейсов.
- Чрезмерная простота синтаксиса. Эта простота — и плюс, и минус. Так как некоторые сложные задачи могут потребовать написание большего кода в Go, если сравнивать его с другими языками программирования.
- Средняя распространенность. Несмотря на рост популярности, Go остается нишевым языком. Количество вакансий, где работодатель требует знания Go, меньше, чем для других популярных языков программирования по типу Java, Python или C++.
- Низкий порог входа. Легкость изучения Go приводит к тому, что очень многие программисты осваивают его, из-за чего наблюдается рост конкуренции.

Востребованность

Хотя Go и нишевый язык, на рынке иногда наблюдаются всплески его популярности. На основе данных из опроса на GitHub, в 2021 году Golang попал в ТОП-5 самых востребованных языков и даже опередил C# и PHP. А в первой половине 2023 года Go 10 место в этом же топе.

Если смотреть глобально и мыслить объективно, Golang все равно востребован на рынке. Правда, его востребованность может сильно варьироваться в зависимости от региона, отрасли и конкретной компании.

Популярные проекты на Go

Docker. Это одна из самых известных и широко используемых платформ для контейнеризации приложений. Основная часть Docker (в том числе Engine) написана именно на Go.

Kubernetes. Оркестратор контейнеров с открытым исходным кодом, разработанный для управления и автоматизации контейнеризированных приложений. Kubelet и Kubectl — компоненты Kubernetes, написанные на Go.

Etcd. На Go написано распределенное хранилище ключ-значение в памяти Etcd, используемое для хранения конфигураций и данных в Kubernetes.

Prometheus. Система мониторинга с открытым исходным кодом, предназначенная для сбора и анализа метрик приложений и инфраструктуры. Основная и серверная часть Prometheus написаны Go-программистами.

Заключение

Go – мощный, безопасный, достаточно простой и очень востребованный язык программирования. Он хорошо подходит для создания высокопроизводительных систем, поскольку способен повысить производительность программы в 5–10 раз без каких-либо оптимизаций.

А благодаря гибкости и способности решать разнообразные задачи, использовать его можно в самых разных областях – от сетевого программирования до криптовалютных приложений.

Список использованных источников

Seguin, K.

The Little Go Book / K. Seguin.

США, 2018. С. 154–174.

An Introduction to Programming in Go

[Электронный ресурс] / Go Docs / Go Docs.

2024.

URL: https://www.golang-book.com/ (Дата обращения 25.01.2024). Загл. с экр. Яз. англ.

Go in Action

[Электронный ресурс] / Go In Action.

2024.

URL: https://www.manning.com/books/go-in-action (Дата обращения 22.01.2024). Загл. с экр. Яз. англ.

Спасибо за внимание!