Содержание

Предисловие

Когда я услышал, что Мэтт Фарина и Мэтт Батчер начали работу над новой книгой о языке Go, это меня очень взволновало. Они оба много лет были важнейшими действующими лицами в экосистеме Go, обладают большим опытом работы и способны добавить в аромат содержания этой книги запахи специй прошлых учебных пособий. Эта книга должна стать преемницей книги «Go in Action», развивающей заложенные там основы и переводящей их в практическое русло.

Книга разбита на четыре простые в освоении части, каждая из которых имеет собственную направленность. Часть 1 освежает в памяти основные идеи языка Go. Если вы спешите и уже обладаете навыками, позволяющими уверенно писать код на языке Go, можете смело пропустить этот раздел, хотя я не рекомендую этого делать. Знакомясь с окончательным вариантом рукописи, я обнаружил в ней самородки такого размера, что, полагаю, главы этой части будут полезны всем читателям.

Часть 2 погружает читателя в недра механики управления Go-приложениями. Глава об ошибках является одним из лучших описаний Go-ошибок из всех, прочитанных мной прежде, а глава, посвященная отладке и тестированию, содержит массу полезной информации об этом важном этапе разработки, помогающем поднять приложение с уровня, требующего доказательства идеи, до уровня надежной производственной системы.

В третьей части вы узнаете о способах создания пользовательских интерфейсов. Глава по шаблонам является отличным руководством по самой сложной, как многие полагают, части экосистемы Go. Она знакомит с практическими приемами многократного использования шаблонов и создания выразительных веб-интерфейсов. Приведенные примеры соответствуют уровню книги, поскольку трудно найти примеры использования шаблонов, легко переносимые в реальные приложения. Затем вы увидите, как создавать и использовать REST API, и познакомитесь с хитростями управления версиями этого API.

Заключительная часть книги посвящена теме функциональной совместимости, необходимой практически любому современному приложению. Она позволяет глубоко погрузиться в облачную инфраструктуру и увидеть, как язык Go вписывается в модель облачных

вычислений. Заканчивается эта часть широким обзором микрослужб и методов взаимодействий между службами.

Кем бы вы ни были, новичком, только что познакомившимся с языком Go, или профессионалом с многолетним опытом, эта книга даст вам жизненно необходимые знания, которые помогут вам поднять ваши приложения на новый уровень. Авторы проделали большую работу по представлению сложной информации в согласованной манере, позволяющей ее легко усвоить. Я искренне рад публикации этой книги и тому вкладу, которое она привнесет в Go-сообщество. Я надеюсь, что вы получите то же удовольствие от ее прочтения, что и я.

— *Брайан Кетелсен* (*Brian Ketelsen*),
один их авторов книги «Go in action»,
один из основателей «Gopher academy»

Введение

При первом знакомстве с языком Go мы сразу оценили его большой потенциал. У нас появилось желание писать на нем приложения. Но это был новый язык, а во многих компаниях опасаются использовать новые языки программирования.

Это особенно касается тех компаний, где потенциал языка Go нашел бы широкое применение. Новым языкам приходится добиваться доверия, признания и принятия. Сотни тысяч разработчиков заняты в бизнесе, лидеры которого долго колеблются, перед тем как попробовать новый язык, а разработчикам, чтобы понять его выгоды и применить в разработке приложений, необходимо достаточно хорошо изучить язык.

Продвижению нового языка способствуют проекты с открытым исходным кодом, конференции, курсы и книги. Цель этой книги – помочь в изучении языка Go всем желающими, внести наш вклад в развитие Go-сообщества и разъяснить перспективы применения языка Go руководству компаний, занятых в том числе разработкой программного обеспечения.

Начиная работу над книгой, мы представляли ее целиком посвященной применению Go в разработке облачных приложений. Мы уже несколько лет работаем в сфере облачных вычислений, а Go – это язык, специально созданный для нее. Начав сотрудничать с издательством Manning, мы сочли возможным выйти за пределы облачных технологий и дополнительно охватить некоторые полезные и практичные шаблоны программирования. В результате центр внимания книги сместился из сферы облачных вычислений в сферу практического применения Go. Тем не менее корнями она по-прежнему уходит в облачные технологии.

Книга «Go на практике» – это наша попытка помочь разработчикам познакомиться с языком для продуктивного его использования, а также помочь развитию сообщества и способствовать улучшению разрабатываемого программного обеспечения.

Благодарности

На написание этой книги мы потратили около двух лет, но ее завершение стало бы невозможным без поддержки наших семей. Они поддерживали нас с раннего утра до позднего вечера и в выходные дни, когда мы целиком уходили в работу. Они были рядом, когда мы не писали, но были поглощены разрешением связанных с книгой проблем.

Хорошие программы не пишутся в вакууме. Мы также признательны членам Go-сообщества, столь щедро отдавшим свое время созданию языка, его библиотек и процветанию его экосистемы. Увлекательно быть частью такого разнообразного и динамичного сообщества разработчиков. Мы, в частности, благодарим, Роба Пайка (Rob Pike), Брайана Кетелсена (Brian Ketelsen) и Дэйва Чейни (Dave Cheney), которые помогли нам на начальном этапе изучения Go. Они отлично справляются с ролью эмиссаров языка. Особая благодарность Брайану за предисловие к книге и одобрение нашей работы.

Мы ценим помощь множества людей, которые пожертвовали своим временем и приложили усилия к созданию этой книги. Это был трудный путь. Мы благодарны многим внимательным читателям, в том числе и нашим рецензентам, обнаружившим многочисленные ошибки, что позволило их своевременно исправить.

Мы хотели бы поблагодарить всех сотрудников издательства Manning, особенно нашего редактора Сусанну Клайн (Susanna Kline), научных редакторов Айвана Киркпатрика (Ivan Kirkpatrick), Кима Шириера (Kim Shrier), Гленна Бернсайда (Glenn Burnside) и Алена Коуниот (Alain Couniot), технического корректора Джеймса Фрасче (James Frasché), а также всех, кто трудился над нашей книгой. Большое спасибо рецензентам, нашедшим время на чтение нашей рукописи на различных стадиях ее готовности и написание отзывов, оказавших нам неоценимую помощь: Энтони Крампу (Anthony Cramp), Остину Риендо (Austin Riendeau), Брэндону Титусу (Brandon Titus), Дугу Спарлингу (Doug Sparling), Фердинандо Сантакосе (Ferdinando Santacroce), Гари А. Стаффорду (Gary A. Stafford), Джиму Амрхайну (Jim Amrhein), Кевину Мартину (Kevin Martin), Натану Дэвису (Nathan Davies), Квинтину Смиту (Quintin Smith), Сэму Зайделу (Sam Zaydel) и Уэсу Шэддиксу (Wes Shaddix).

Наконец, мы благодарны сообществу Glide, сформировавшемуся за время работы над созданием диспетчера пакетов верхнего уровня для Go. Благодарим вас за вашу поддержку.

Я начал писать эту книгу, работая в компании Revolv, продолжил после приобретения ее компанией Google/Nest и закончил в компании Deis. Спасибо им всем за поддержку в создании этой книги. Благодарю Брайана Хардока (Brian Hardock), Кристиана Кавалли (Cristian Cavalli), Ланна Мартина (Lann Martin) и Криса Чинга (Chris Ching), всех, с кем я советовался. Мэтт Боерсма (Matt Boersma) написал полезные отзывы о нескольких главах. Кент Rancourt (Kent Rancourt) и Аарон Шлезингер (Aaron Schlesinger) подали идею для нескольких примеров в книге. Мэтт Фишер (Matt Fisher), Сиварам Мозики (Sivaram Mothiki), Кирзан Мала (Keerthan Mala), Хельги Порбьёрнссон (Helgi Þorbjörnsson) (да, Хельги, я скопировал и вставил это), Гейб Монрой (Gabe Monroy), Крис Армстронг (Chris Armstrong), Сэм Бойер (Sam Boyer), Джефф Блейел (Jeff Bleiel), Джошуа Андерсон (Joshua Anderson), Римас Мосевикиус (Rimas Mocevicius), Джек Фрэнсис (Jack Francis) и Джош Лэйн (Josh Lane) – все вы (вольно или невольно) оказали влияние на определенные фрагменты этой книги. Нельзя недооценивать влияние Мишеля Ноорали (Michelle Noorali) и Адама Риза (Adam Reese). Я многому научился, наблюдая за работой нескольких Ruby-разработчиков, знатоков Go. И спасибо Энджи (Angie), Аннабель (Annabelle), Клэр (Claire) и Кэтрин (Katherine) за их постоянную поддержку и понимание.

– *Мэтт Батчер* (*Matt Butcher*)

Я хотел бы поблагодарить мою красивую и удивительную жену Кристину, а также наших прекрасных дочерей, Изабеллу и Обри, за их любовь и поддержку.

Я писал эту книгу, работая в компании Hewlett Packard Enterprise, ранее называющейся Hewlett-Packard. Работая в компании Hewlett Packard Enterprise, я многому научился, общаясь с теми, кто был гораздо умнее меня. В частности, я должен поблагодарить Раджива Пандей (Rajeev Pandey), Брайана Акера (Brian Aker), Стива Маклеллана (Steve McLellan), Эрин Handgen (Erin Handgen), Эрика Густафсона (Eric Gustafson), Майка Хагедорна (Mike Hagedorn), Сьюзан Балле (Susan Balle), Дэвида Гравеса

(David Graves) и многих других. Они учили меня писать приложения и управлять ими, и это, безусловно, отразилось на данной книге.

Многие другие также оказали влияние на определенные фрагменты этой книги, иногда даже сами не осознавая этого. Благодарю Тима Плетчера (Tim Pletcher), Джейсона Буберела (Jason Buberel), Сэм Бойера (Sam Boyer), Ларри Гарфилда (Larry Garfield) и всех тех, кого я мог забыть.

Наконец, я хочу поблагодарить Мэтта Батчера. Я никогда не задумывался над тем, чтобы стать автором книги, пока ты не предложил мне это. Спасибо!

– *Мэтт Фарина* (*Matt Farina*)

О книге

Книга «Go на практике» описывает практические приемы программирования на языке Go. Разработчики, знакомые с основами Go, найдут в ней шаблоны и методики создания Go-приложений. Каждая глава затрагивает определенную тему (как, например, глава 10 «Взаимодействие облачных служб») и исследует различные технологии, относящиеся к ней.

Как организована книга

Одиннадцать глав разделены на четыре части.

Часть I «Основные понятия и принципы» закладывает фундамент для будущих приложений. Глава 1 описывает основы языка Go и будет полезна всем, кто еще не знаком с ним или хотел бы узнать о нем больше. В главе 2 рассказывается о создании консольных приложений и серверов, а в главе 3 – об организации параллельных вычислений в Go.

Часть II «Надежные приложения» включает главы 4 и 5, охватывающие темы ошибок, аварий, отладки и тестирования. Цель этого раздела – рассказать о создании надежных приложений, способных автоматически справляться с возникающими проблемами.

Часть III «Интерфейсы приложений» содержит три главы и охватывает диапазон тем, от генерации разметки HTML и обслуживания ресурсов до реализации различных API и работы с ними. Многие Go-приложения поддерживают взаимодействие через веб-интерфейс и REST API. Эти главы охватывают приемы, помогающие в их конструировании.

Часть IV «Облачные приложения» содержит остальные главы, посвященные облачным вычислениям и генерации кода. Язык Go разрабатывался с учетом нужд облачных технологий. В этой части демонстрируются приемы, позволяющие работать с облачными службами и приложениями и даже с микрослужбами в них. Кроме того, она охватывает приемы генерации кода и метапрограммирование.

В книге описывается 70 приемов, и в каждом случае сначала ставится задача, затем дается решение и в заключение обсуждаются причины выбора тех или иных подходов.

Соглашения и загрузка примеров кода

Весь исходный код в книге оформлен моноширинным шрифтом, `как этот текст`, чтобы выделить его в окружающем тексте. Большинство листингов сопровождаются указателями на ключевые понятия или пронумерованными маркерами для ссылки на них в тексте с пояснениями к коду.

Исходный код примеров можно загрузить на сайте издательства, по адресу: www.manning.com/books/go-in-practice, или из репозитория GitHub: github.com/Masterminds/go-in-practice.

Авторский интернет-форум

Приобретая книгу «Go на практике», вы получаете свободный доступ к закрытому веб-форуму издательства Manning Publications, где можно оставить отзыв о книге, задать технические вопросы и получить помощь авторов или других пользователей. Чтобы получить доступ к форуму и зарегистрироваться на нем, перейдите на страницу www.manning.com/books/go-in-practice. Здесь вы найдете информацию о том, как пользоваться форумом после регистрации, какого рода помощь можно получить и правила поведения на форуме.

Издательство Manning, идя навстречу пожеланиям своих читателей, предоставляет площадку для конструктивного диалога между читателями, а также между читателями и авторами. Это не обязывает авторов к участию в нем – любое их участие является добровольным (и никак не оплачивается). Задавайте авторам сложные вопросы, чтобы заинтересовать их!

Авторский интернет-форум и архивы предыдущих дискуссий будут доступны на веб-сайте издателя, пока книга продолжает печататься.

Об авторах

Мэтт Батчер – архитектор программного обеспечения в компании Deis, где занимается проектами с открытым исходным кодом. Написал несколько книг и множество статей. Имеет докторскую степень и преподает на факультете информационных технологий в университете Лойола (Чикаго, США). Мэтт увлечен созданием сильных команд и разработкой элегантных решений сложных проблем.

Мэтт Фарина работает в должности ведущего инженера группы передовых технологий в компании Hewlett Packard Enterprise. Технический писатель, лектор и регулярно вносит свой вклад в проекты с открытым исходным кодом. Занимается разработкой программного обеспечения уже более четверти века. Любит решать проблемы, применяя не только новейшие, но и самые обычные технологии, о которых часто забывают.

Об изображении на обложке

Рисунок на обложке книги «Go на практике» называется «Типичная одежда русской крестьянки, 1768 год». Иллюстрация взята из книги «Collection of the Dresses of Different Nations, Ancient and Modern» (Коллекция костюмов разных народов, античных и современных) Томаса Джефферис (Thomas Jefferys), опубликованной в Лондоне между 1757 и 1772 годом. В подписи к странице было указано, что она представляет выполненную вручную каллиграфическую цветную гравюру, обработанную аравийской камедью.

Томас Джефферис (1719–1771) носил звание «Географ короля Георга III». Английский картограф, был ведущим поставщиком карт того времени. Он выгравировал и напечатал множество карт для нужд правительства, других официальных органов и широкий спектр коммерческих карт и атласов, в частности Северной Америки. Будучи картографом, интересовался местной одеждой народов, населяющих разные земли, и собрал блестящую коллекцию различных платьев в четырех томах.

Очарование далеких земель и дальних путешествий для удовольствия было относительно новым явлением в конце XVIII века, и коллекции, такие как эта, были весьма популярны, знакомя с внешним видом жителей других стран. Разнообразие рисунков, собранных Джефферисом, свидетельствует о проявлении народами мира около 200 лет яркой индивидуальности и уникальности. С тех пор стиль одежды сильно изменился, и исчезло разнообразие, характеризующее различные области и страны. Теперь трудно отличить по одежде даже жителей разных континентов. Если взглянуть на это с оптимистической точки зрения, мы пожертвовали культурным и внешним разнообразием в угоду разнообразию личной жизни, или в угоду более разнообразной и интересной интеллектуальной и технической деятельности.

В наше время, когда трудно отличить одну техническую книгу от другой, издательство Manning проявило инициативу и деловую сметку, украшая обложки книг изображениями, основанными на богатом разнообразии жизненного уклада народов двухвековой давности, придав новую жизнь рисункам Джефферис.

Часть I

ОСНОВНЫЕ ПОНЯТИЯ И ПРИНЦИПЫ

Эта часть книги содержит базовые сведения о языке Go и описание фундаментальных принципов разработки приложений на нем. Глава 1 начинается с обзора языка Go для тех, кто еще с ним не знаком.

Главы 2 и 3 охватывают основные компоненты приложений. Глава 2 описывает основы разработки приложений, включая консольные и серверные, и приемы их настройки. Глава 3 рассматривает использование сопрограмм Go. Сопрограммы являются одним из наиболее мощных и полезных элементов языка Go. Они широко используются в Go-приложениях, и вы часто будете сталкиваться с ними на протяжении всей книги.

Глава 1

Введение в язык Go

В этой главе рассматриваются следующие темы:

- *введение в язык Go;*
- *место языка Go в ландшафте языков программирования;*
- *подготовка к работе с языком Go.*

С течением времени меняется подход к разработке и запуску программного обеспечения. Инновации трансформируют понятие о вычислительной среде, где выполняются программы. Чтобы полностью использовать преимущества нововведений, необходимы языки и инструменты, изначально их поддерживающие.

Во времена, когда создавалось большинство популярных ныне языков программирования и поддерживающих их инструментов, существовали только одноядерные процессоры, соответственно, они проектировались с прицелом на работу в однопроцессорной среде. В настоящее время настольные компьютеры, серверы и даже телефоны оснащены многоядерными процессорами. На любом из них можно выполнять программы с параллельными операциями.

Меняются инструменты разработки приложений. Увеличение сложности программного обеспечения требует окружений, способных быстро собирать код и эффективно его выполнять. Тестирование больших и сложных приложений должно выполняться быстро, чтобы не тормозить процесс разработки. Многие приложения используют библиотеки. Благодаря решению проблемы нехватки дискового пространства появилась возможность поддерживать несколько версий библиотек.

Меняется подход к предоставлению инфраструктуры и программного обеспечения. Использование групп серверов, размещаемых в одном месте, и простое выделение виртуальных частных серверов стали нормой. Прежде масштабирование служб, как правило, озна-

чало необходимость инвестиций в собственное оборудование, включая средства балансировки нагрузки, серверы и хранилища. Закупка всего перечисленного, установка и ввод в эксплуатацию занимали от нескольких недель до нескольких месяцев. Теперь все это можно получить в облаке за несколько секунд или минут.

Эта глава служит введением в язык программирования Go для тех, кто не сталкивался с ним раньше. Она содержит сведения о языке, средствах поддержки, его месте в ряду других языков, установке и начале работы с ним.

1.1. Что представляет собой язык Go?

Язык Go, который часто называют *golang*, чтобы упростить поиск сведений о нем в Интернете, – это статически типизированный и компилируемый язык программирования с открытым исходным кодом, разработку которого начинала компания Google. Роберт Грисемер (Robert Griesemer), Роб Пайк (Rob Pike) и Кен Томпсон (Ken Thompson) сделали попытку создать язык для современных программных систем, способных решать проблемы, с которыми они столкнулись при масштабировании больших систем.

Вместо попытки достичь теоретического совершенства создатели языка Go отталкивались от ситуаций, часто возникающих на практике. Их вдохновлял опыт ведущих языков, созданных прежде, таких как C, Pascal, Smalltalk, Newsqueak, C#, JavaScript, Python, Java и других.

Go – не обычный статически типизированный и компилируемый язык. Его статическая типизация имеет черты, делающие ее похожей на динамическую, а скомпилированные двоичные файлы включают среду выполнения со встроенным сборщиком мусора. На структуру языка наложили отпечаток проекты, для которых он должен был использоваться в Google: большие проекты, поддерживающие масштабирование и разрабатываемые многочисленными группами разработчиков.

По сути, Go – это язык программирования, определяемый спецификациями, которые можно реализовать в любом компиляторе. Их реализация по умолчанию распространяется в виде инструмента `go`. Но Go – это не просто язык программирования. На рис. 1.1 изображена его многослойная структура.

Для разработки приложений нужен не только язык программирования, а также средства тестирования, документирования и форматирования исходного кода. Инструмент `go`, обычно используемый для

компиляции приложений, поддерживает также все вышеперечисленное. Это целый комплект инструментов для разработки приложений. Одним из наиболее важных аспектов этого комплекта является поддержка управления пакетами. Встроенная система управления пакетами, вместе с общими инструментами разработки, позволила сформировать вокруг языка программирования целую экосистему.

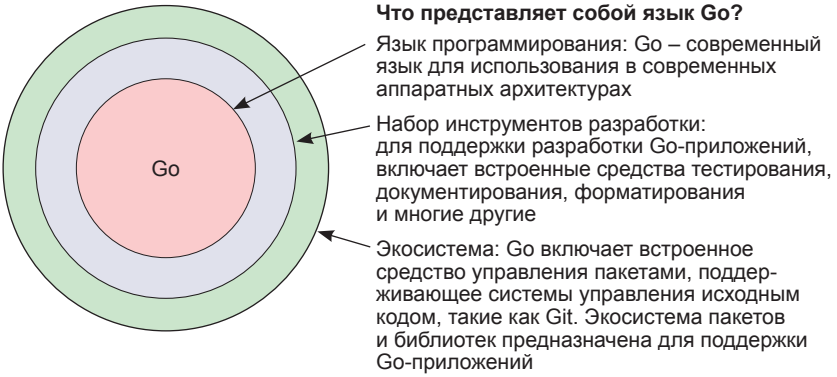

Рис. 1.1 ❖ *Слои языка Go*

Одной из определяющих особенностей Go является его простота. Когда Гисемер, Пайк и Томпсон начинали проектирование языка, новые функциональные возможности не включались в язык, пока все трое не приходили к согласию об их необходимости. Такой стиль принятия решений, а также их многолетний опыт привел к созданию простого и мощного языка. Он прост в изучении, но достаточно мощный для широкого спектра программного обеспечения.

Философию языка можно проиллюстрировать примером синтаксиса объявления переменной:

```
var i int = 2
```

Здесь создается целочисленная переменная, и ей присваивается значение 2. Поскольку имеется присваивание начального значения, определение можно сократить до:

```
var i = 2
```

При наличии начального значения компилятор автоматически определяет по нему тип. В данном случае компилятор обнаружит значение 2 и поймет, что переменная должна иметь целочисленный тип.

Но язык Go не останавливается на этом. А нужно ли само ключевое слово `var`? Нет, потому что Go поддерживает *краткое объявление переменных*:

```
i := 2
```

Это краткий эквивалент первой инструкции определения переменной. Он более чем вдвое короче, легко читается, и все это за счет автоматического определения компилятором недостающих частей.

Простота Go означает, что он поддерживает не все возможности, имеющиеся в других языках. Например, в языке Go отсутствуют тернарный оператор (обычно это `?:`) и обобщенные типы. Не содержит он и некоторых других функциональных возможностей, присутствующих в современных языках, что служит поводом для его критики, но это не должно служить поводом для отказа от использования языка Go. В мире программирования одна и та же задача часто может быть решена множеством способов. Даже если в Go отсутствуют какие-то возможности, имеющиеся в других языках, вместо них он предоставляет иные пути решения проблем, ничуть не хуже.

Несмотря на простоту ядра языка Go, встроенная система управления пакетами позволяет добавлять в него дополнительные возможности. Многие из недостающих элементов можно подключить как сторонние пакеты и включить в приложение.

Минимальный размер и сложность дают свои преимущества. Язык можно быстро освоить, и он хорошо запоминается. Это важное преимущество, когда требуется быстро исследовать чужой код.

1.2. Примечательные особенности языка Go

Поскольку язык Go разрабатывался, исходя из практических нужд, он обладает рядом особенностей, достойных особого упоминания. Все вместе эти полезные характеристики образуют строительные блоки, из которых конструируются Go-приложения.

1.2.1. Возврат нескольких значений

Одна из первых особенностей, которую вы узнаете, начав изучать Go, – функции и методы могут возвращать несколько значений. Большинство языков программирования поддерживает возврат из