# Введение

Разработка программного обеспечения и сервисов для сети Интернет в глобальном масштабе стала как никогда доступна. Если только у вас и вашей команды есть интересная новая идея или необычное решение для уже известной проблемы, вся мощь вычислительных облаков Cloud и обеспечиваемый ими легкий доступ к прорывным технологиям, легкость и скорость запуска контейнеров, точная настройка и изоляция их деталей с помощью Docker, и оркестрация работающих в контейнерах микросервисов с помощью Kubernetes даст вам возможность выйти на мировой уровень так, как если бы вы с полной уверенностью показали идеально настроенное демо приложения на вашем ноутбуке.

В этой книге мы взглянем на все с высоты птичьего полета, проанализируем популярную концепцию приложений, созданных работать в облаке (Cloud Native), вспомним как появились технологии виртуализации и масштабирования, разберем что именно принесут нам контейнеры и микросервисы, и увидим, как настройка и оркестрация Kunernetes позволяет развернуть систему любой сложности в любом облаке и мгновенно приспособить ее к растущим нагрузкам, при этом сделав ее надежной и устойчивой к отказам.

## Актуальность и глубина информации. Онлайн-документация

Тема книги и технологии, которые мы изучаем и пробуем в ней, являются одним из самых популярных и динамичных направлений программирования и разработки последних пары лет. Уровень изменений и их скорость очень высока, и то что было актуально и важно полгода назад, может уступить свое место новой разработке или сервису.

Поэтому мы не стараемся максимально глубоко изучить все инструменты которыми пользуемся в данной книге, особенно это касается библиотек и программных сервисов API, предоставляемых известными публичными облаками (такими как Amazon AWS, Google Cloud и Microsoft Azure). Основное это понять процесс, который применяется при разработке в облаках, эффективно использовать базовые и главные возможности контейнеров Docker, и перейти “на ты” с Kubernetes.

Мы не станем перепечатывать массу документации из Интернета, прежде всего с сайтов docker.io и kubernetes.io. Большие компании, Google, Amazon и другие создают целый штаб качественных технических писателей, сопровождающих важные продукты, особенно если дело касается их коммерческих предложений и связанных с ними технологий, прежде всего Kubernetes. Хорошая документация, примеры, онлайн-лаборатории - все это к вашим услугам, и чем лучше качество и скорость начала работы с облаком, тем быстрее и больше оно привлекает клиентов.

На мой взгляд, основное - понять суть происходящего, увидеть “лес за деревьями”, узнать про краткую историю и эволюцию платформ, явлений, экосистем технологий, которые мы стараемся изучить, прежде всего иногда понять даже то, нужно ли нам вообще идти в этом направлении. Именно это очень трудно сделать в разношерстном море ссылок, блогов и статей Интернета, именно это мы и попробуем сделать в книге, уложившись в небольшой размер, и сделав ее путешествием по Cloud Native.

## Аудитория книги

Эта книга прежде всего для программистов, которые на данный момент работают в привычной, не обязательно связанной с облаком среде - к примеру, запускают сервисы или работают с базами данных на собственных серверах или виртуальных машинах AWS, разрабатывают стандартные приложения для операционных систем (с интерфейсами командной строки, классические графические приложения, или вспомогательные приложения), или в основном сконцентрированы на пользовательских интерфейсах web и мобильных приложений. Подразумевается что вы знаете один или несколько языков программирования и основы сетей и протокола HTTP, но не более того.

Книга описывает, как применить классические знания о программировании в новой среде облачных вычислений (Сloud), где вместо ваших собственных настоящих физических серверов или управляемых вручную “тяжелых” мощных виртуальных вычислительных машин есть лишь эфемерная среда единого кластера (cluster), в которой будут исполняться сервисы, но среда эта способна практически мгновенно масштабировать приложение и сервисы для доступа миллионов пользователей, обеспечивать практически неограниченные ресурсы для вычисления и хранения данных, и эффективно обновлять приложения, используя множество полезных инструментов и сервисов Kubernetes, Docker и открытого сообщества вокруг них. При этом количество настроек самого приложения, изменений в нем будет не так велико - большая часть работы уже будет сделана для нас, особенно когда это касается непрерывного управления и масштабирования приложения.

По большому счету, это же относится и к программистам сервисов или больших монолитных приложений, уже работающих на выделенных (dedicated) для этого серверах, виртуальных машинах публичных провайдеров облака, или же в собственных центрах вычислений, как часть частных облачных решений. Даже если большая часть всего, что касается контейнеров, микросервисов, архитектуры приложения как сервисов, уже вам отлично известно, то книга может быть полезна просто посмотреть на то, что дополнительно может предложить вам Kubernetes и общая концепция Cloud Native (концепция приложений, созданных для облака).

## Программирование и архитектура. Концепция Cloud Native

О чем именно эта книга? Мы попытаемся узнать большое количество связанных технологий, которые прекрасно сочетаются в единое целое, когда приложение или просто набор сервисов начинает работать в облаке. В общем и целом связаны технологии, которые мы будем рассматривать, *концепцией разработки приложений, созданных и приспособленных для работы в современном облаке (cloud native applications)*. Концепция эта больше высокого уровня, можно назвать это архитектурой или дизайном (будем считать, как и многие эксперты, что эти два слова означают для программистов примерно одно и то же). В общем случае приложение, созданное для облака, способно быстро запускаться на любом облаке с поддержкой основных технологий концепции (Docker и Kubernetes), и при правильном подходе быстро масштабироваться при пиковой нагрузке, непрерывно обслуживать пользователей даже во время обновлений и фатальных ошибок, и позволять программистам использовать любые нужные им технологии и инструменты.

На словах концепция звучит прекрасно, однако между архитектурой (дизайном) и непосредственной разработкой большая работа и множество деталей, в которых, как известно, заключается иногда самая большая неприятность. В книге будет некоторое количество намеренно очень простых примеров, главная цель которых состоит в том, чтобы за минимальное количество шагов и настройки соединить теорию и историю изучаемого вопроса с практикой - непосредственным использованием довольно сложного рабочего инструмента (особенно когда речь идет о Kubernetes). После этого, уже зная основные детали, поняв их смысл, можно строить на этом уже реальную, сложную функциональность. Цельных, больших приложений “промышленного” уровня в книге мы рассматривать не будем - но вы всегда сможете найти их на GitHub, конференциях KubeCon, и разнообразных блогах.

### Масштабирование на практике

Одним из основных преимуществ концепции Cloud Native и оркестратора Kubernetes является возможность автоматически масштабировать ваше приложение при пиковых нагрузках, и масштабировать лишь те его части, которые в этом нуждаются, если приложение разработано как набор микросервисов. На маленьких примерах это понять непросто, и лучше всего, если у вас в процессе чтения появится идея приложения или сервиса, которому понадобится быть в облаке, использовать впечатляющую мощь облачных библиотек и быть наготове для масштабирования и доступа для миллионов пользователей. В этом случае вся информация книги будет идти постепенно и в идеальном порядке. Если вы при этом еще будете использовать новый для себя язык программирования и небольшое руководство по нему, процесс станет особенно приятным и полезным.

## Русскоязычные термины

В данный момент компьютерные технологии практически полностью находятся в зоне английского языка, как, впрочем, практически весь глобальный Интернет. Ничего плохого в этом нет, единый язык, который достаточно просто начать изучать, помогает мгновенному распространению информации, участию в конференциях программистов и архитекторов со всех уголков планеты, полезности таких сайтов как StackOverflow, и многому другому.

Тем не менее, нет ничего более легкого для усвоения, чем структурированная и правильно поданная информация на родном языке, который просто “работает” на уровне глубокого подсознания, не требуя ни минуты задержки, впитываясь в память и сознание. Эта книга постарается максимально структурировать и постепенно ввести вас в мир облачных технологий на родном языке, и мы будет использовать максимально близкие по смыслу русскоязычные аналоги английских названий технологий. Правда, многие библиотеки, алгоритмы, компании, сайты, паттерны проектирования, процессы разработки настолько влились в нашу жизнь на английском языке, что чтение их названий на русском языке будет даже мешать. В этом случае мы будем использовать максимально нейтральное название и включать в скобках английское название, чтобы не перемешивать языки и шрифты, и не коверкать книгу и предложения не слишком звучным транслитом (Докер? Кубернетес? А может быть Кюбэрнэтис? “Нода” и “поды”? В любом случае подобное выглядит просто неприятно - мы будем писать английский термин и использовать наиболее верный перевод, без излишней смысловой нагрузки).

## Пользовательские интерфейсы

В этой книге мы в основном рассматриваем технологии для облака, особенно подходящие для быстрой разработки, от идеи до минимального работающего продукта (minimal viable product, MVP), и его последующей миграции в полноценное, масштабируемое до глобального мирового сервиса приложение или системы. Пользовательский интерфейс для системы это отдельная тема, чрезвычайно динамичная, вызывающая всегда горячие споры и как все что свойственно вкусу и внешности, то, что вызывает восторг у одних, кажется совершенно ужасным другим. Конечно у приложения будет web интерфейс, приложения для iOS и Android, возможно консоль администратора в дополнение ко всем диагностическим инструментам что мы затронем в этой книге. Тема чрезвычайно обширная и бесконечная и изучения требует в отдельном разговоре. Будем считать что отличный интерфейс у нас есть, и уделять внимания в данной книге ему не будем.

## Примеры на Go и Java

Go как нельзя лучше подходит для микросервисов, особенно в качестве очень коротких, простых и быстрых процедур обработки данных. Этот язык специально был создан простым по синтаксису, в нем по сути не присутствует полноценная система объектно-ориентированного программирования, и он во всем приглашает к простому, короткому, быстрому коду.

Крайне полезно то, что Go компилируется в машинный код для целевой операционной системы, то есть работает так эффективно, как это возможно без специальных оптимизаций. Работая в облаке, где каждую минуту исполнения, мегабайт оперативной памяти, использованную энергию необходимо будет оплатить, любые промежуточные этапы, использующие ресурсы, могут быть чуть дороже, чем интерпретируемые языки, например Python, или языки на основе виртуальных машин Node.js и Java, даже если они быстро компилируются (just in time, JIT) в машинный код в процессе работы. Это “чуть”, помноженное на масштаб и количество пользователей (что в конечном итоге основной показатель успеха вашей идеи, стратегии и качества реализации), может вылиться в порядочные траты. Тем более что некоторые сложные участки программ на основе виртуальных машин оптимизировать крайне тяжело и они будут исполняться в разы медленнее.

Конечно, компиляция и сборка в машинный код всегда представляет собой проблему переносимости на конкретную платформу - что именно вы ожидаете на сервере в облаке, конкретный тип Linux? А если облачного провайдера нужно будет поменять или вы решите перейти на частное облако или свой собственный кластер? Это было бы огромным препятствием, но благодаря границам контейнеров Docker вы можете практически полностью изолировать детали реализации, настройку системы и ее пакетов и библиотек, от непосредственной операционной системы серверов в облаке или кластере. По сути, контейнеры полностью избавляют вас от зависимости от деталей облака и операционных систем, лишь бы Docker или подобная система контейнеров поддерживалась на серверах, что сейчас просто подразумевается как само собой разумеющееся требование к провайдеру облачных услуг.

Язык и платформа Java чрезвычайно популярны для классических корпоративных приложений, практически идеально переносятся между платформами и операционными системами, а количество фреймворков для веб-приложений, сервисов и микросервисов поражает воображение. Начать разработку может быть не так быстро и эффектно как на Go, но если у вас изощренный и сложный домен приложения (то есть описание области, в которой ваше приложение будет действовать), мощь объектного подхода, продуманный дизайн и стабильность языка, хорошо известные шаблоны проектирования и практики работы, и сотни разнообразных инструментов Java могут быть как нельзя кстати.

Вдобавок, главные игроки экосистемы Java и платформ для этого языка крайне заинтересованы оставить этот язык в лидерах разработки даже в мире микросервисов. Идет большая работа по уменьшению требований к оперативной памяти, размера виртуальной машины JRE, и разбиения платформы на модули, позволяющего исключить из приложения ненужные мегабайты библиотек JAR. Посмотрите на GraalVM от Oracle, оптимизирующей запуск программы и компилируемый код в бинарный, или библиотеки Quarkus и Microprofile, создающие минимальные микросервисы.

## Сторонние библиотеки и инструменты

Все разработчики рано или поздно сталкиваются с похожими задачами и частыми проблемами. Так как хороший разработчик (и человек разумный в целом) как правило ленив, он пытается сразу же решить задачу так, чтобы никогда больше с ней не сталкиваться в будущем. Количество же инструментов, библиотек, блогов и стартапов в такой популярной и растущей стремительными темпами области как облачные вычисления просто поражает воображение. Многие задачи решены, решаются, а затем немедленно заново решаются с различной степенью эффективности, качеством документации и уровнем поддержки готовых решений.

Мы будем использовать многие из них, но это не совет на будущее и не “лучший инструменты или библиотека их всех”. Это лишь способ быстрее научиться главному, не изобретая уже много раз решенные вещи. Вполне может быть, что упомянутая в книге библиотека или инструмент выйдет из моды, обнаружится критические просчеты или просто будет заменен более удачным конкурентов. В это случае лишь поиск по основным тематическим сайтам поможет вам заменить их. Основы же останутся прежними.

## Основные провайдеры облачных услуг - Amazon, Google, Microsoft

Основные технологии (контейнеризация, Docker, оркестрация сервисов Kubernetes) концепции приложений, созданных для облака Cloud Native, уже практически гарантированно работают на всех провайдерах облака, будь это три кита облачных вычислений Amazon Web Services (AWS), Google Cloud Platform (GCP), или Microsoft Azure, или провайдеры меньших размеров, часто также предлагающих интересные цены и услуги (российский #CloudMTS, Digital Ocean). Для нас же это означает, что на данный момент концепция стала стандартом, который очень широко поддерживается, и будет поддерживаться на многие годы вперед, а значит изучение ее основ, и технологий в ней, даст нам ключи к эффективной работе с любыми облаками. Созданные же единожды приложения можно будет и переносить между облаками, и запускать в гибридном облаке, работающем на нескольких провайдерах одновременно.