Содержание

Предисловие

Ум — это не сосуд, который нужно наполнить, а факел, который нужно зажечь.

Плутарх

Что должен знать каждый Java-программист? Зависит от обстоятельств. От того, зачем, кого и когда вы спрашиваете. Ответов по крайней мере столько же, сколько точек зрения. В языке, платформе, экосистеме и сообществе, которые влияют на программное обеспечение (ПО) и жизни многих людей и делали так из двадцатого века в двадцать первый, от одного ядра ко многим, от мегабайт к гигабайтам, ответ зависит от большего количества факторов, чем возможно охватить в одной книге единственным автором.

Вместо этого в данной книге мы опираемся на некоторые из этих многочисленных точек зрения, чтобы дать вам поперечный срез и представление о типах мышления во вселенной Java. Это не *все* мнения, но 97 из них от 73 участников. Процитируем предисловие «97 вещей, которые должен знать каждый программист» (O'Reilly):

> Когда так много нужно знать, так много нужно сделать и есть так много способов сделать это, ни один человек или один источник не может утверждать, что его путь — «единственный истинный». Идеи не совпадают, как модульные части, и никто к этому не стремится — пожалуй, даже наоборот. Ценность каждого вклада проистекает из его уникальности. Ценность коллекции заключается в том, как материалы дополняют, подтверждают и даже противоречат друг другу. Здесь нет всеобъемлющего повествования: вы должны прочувствовать идеи, поразмыслить над ними и связать воедино, примеряя их к своей ситуации, знаниям и опыту.

Что должен знать каждый Java-программист? 97 тем, которые мы выбрали в качестве ответа, охватывают язык, JVM, методы тестирования, JDK, сообщество, историю, гибкое мышление, ноу-хау реализации, профессионализм, стиль, суть, парадигмы программирования, программистов как людей, архитектуру ПО, навыки за пределами программирования, инструментарий, механику GC, языки JVM, отличные от Java… и многое другое.

Разрешения

В духе первой книги из серии «97 вещей…» каждый интеллектуальный вклад в данный том внесен на основе неограничивающей модели с открытым исходным кодом. Каждый вклад автора лицензируется в соответствии с Creative Commons Attribution 4.0 license (*https://oreil.ly/zPsKK*). Многие материалы также впервые появились в публикации «97 вещей…» на странице Medium (*https://medium.com/97-things*).

Все это — топливо и огонь для ваших мыслей и кода.

Онлайн-обучение O'Reilly

Более сорока лет *O'Reilly Media* помогает компаниям добиваться успеха, предоставляя им технологии и бизнес-тренинги, сведения и наработки.

Наше уникальное сообщество экспертов и новаторов делится знаниями и опытом с помощью книг, статей, конференций и нашей платформы онлайн-обучения O'Reilly. Последняя предоставляет вам доступ по запросу к практическим подготовительным курсам, методам углубленного обучения, интерактивным средам программирования и обширной коллекции текстов и видео от O'Reilly и более 200 других издателей. Чтобы узнать подробности, посетите *http://oreilly.com*.

Как с нами связаться

Вы можете получить доступ к веб-странице этой книги, где найдете примеры, обнаруженные неточности и прочую дополнительную информацию, по адресу *https://oreil.ly/97Tejpsk*.

Чтобы прокомментировать или задать технические вопросы об этой книге, напишите письмо на *bookquestions@oreilly.com*.

Для получения новостей и информации о наших книгах и курсах посетите *http://oreilly.com*.

Присоединяйтесь к нам в Twitter *http://twitter.com/oreillymedia*, а также посмотрите *http://twitter.com/97_Things*.

Смотрите нас на YouTube: *http://youtube.com/oreillymedia*.

Благодарности

Многие люди вложили свои знания и время, как прямо, так и косвенно, в проект «97 вещей, которые должен знать каждый Java-программист». Все они заслуживают похвалы.

Мы хотели бы поблагодарить всех тех, кто потратил время и силы, чтобы внести свой вклад в эту книгу. Мы также благодарны Брайану Гетцу за дополнительные отзывы, комментарии и предложения.

Спасибо O'Reilly за поддержку, которую они оказали этому проекту, в том числе Зан Маккуэйд и Корбину Коллинзу за их руководство и заботу об участниках и содержании, а также Рэйчел Румелиотис, Сьюзан Конант и Майку Лукидесу за их вклад в проделанную работу.

Кроме того, Кевлин хотел бы поблагодарить свою жену Каролин за то, что она понимает смысл его бреда, и своих сыновей Стефана и Янника за то, что они понимают своих родителей.

Триша желала бы добавить благодарность своему мужу Исре (за то, что он подсказал ей: пребывание в стрессе из-за того, что она делает слишком мало, мешает ей делать вообще что-либо) и своим дочерям Эви и Эми за бескорыстную любовь и обнимашки.

Мы надеемся, что эта книга станет для вас информативной, поучительной и вдохновляющей. Наслаждайтесь!

Всё, что вам нужно, — это Java*

Андерс Норас

Работая над первой крупной редакцией Visual Studio, команда Microsoft представила миру трех персонажей-разработчиков: Морта, Элвиса и Эйнштейна.

Морт, приспосабливающийся работник, быстро исправлял и придумывал что-то по ходу дела. Элвис, прагматичный сотрудник, создавал решения на века, обучаясь на рабочем месте. Эйнштейн, программист-параноик, как одержимый разрабатывал наиболее эффективное решение и старался во всем разобраться, прежде чем писать код.

Языки программирования (ЯП) разделены, как религиозные касты. Мы со стороны Java смеялись над Мортами и хотели быть Эйнштейнами, которые создают фреймворки, чтобы вынудить Элвисов писать свой код «правильно».

Это происходило на заре эры фреймворков, и, если вы не владели новейшим, лучшим объектно-реляционным мэппером и фреймворком с инверсией управления, вы не были настоящим Java-программистом. Библиотеки превратились в фреймворки с установленной архитектурой. И по мере того, как они становились технологическими экосистемами, многие из нас забыли о маленьком, но очень способном языке — Java.

Java — отличный ЯП, и в его библиотеке классов есть инструменты на все случаи жизни. Нужно работать с файлами? `java.nio` тебя прикроет. Базы данных? `java.sql` — вам сюда. Почти каждый дистрибутив Java даже оснащен полноценным HTTP-сервером, однако вам придется перейти от модуля с именем Java на `com.sun.net.httpserver`.

По мере того как наши приложения переходят к бессерверным архитектурам, где единицами развертывания могут быть отдельные функции, преимущества, которые мы получаем от фреймворков, уменьшаются. Это связано с тем,

* В оригинале All You Need Is Java — отсылка к All You Need Is Love, классическому хиту Beatles 1967 года. — *Прим. ред.*

что мы, вероятно, будем тратить не так много времени на решение технических и инфраструктурных проблем, сосредоточив усилия по программированию на бизнес-возможностях, которые реализуют наши приложения.

Как выразился Брюс Джойс, автор книг про образование и педагогику:

> *Нам приходится время от времени изобретать колесо, и не потому, что нам нужно много колес, а потому, что нам нужно много изобретателей.*

Многие задались целью создания универсальных фреймворков для бизнес-логики, чтобы максимально использовать их повторно. Большинство потерпели неудачу, поскольку на самом деле нет никаких универсальных бизнес-проблем. Чем один бизнес отличается от другого? Тем, что делает нечто особенное уникальным, специфичным для него образом. Вот почему вы гарантированно будете писать бизнес-логику практически для каждого проекта. Возможно, у вас возникнет соблазн ввести механизм правил или что-то подобное ради того, чтобы создать нечто универсальное и многоразовое. Но в конечном счете настройка механизма правил — это программирование, часто на языке, уступающем Java. Почему бы вместо этого не попробовать просто написать на Java? Вы с удивлением обнаружите, что конечный результат легко читаем, а это, в свою очередь, упрощает поддержку кода даже для тех программистов, которые на Java не работают.

Довольно часто можно заметить, что библиотека классов Java немного ограниченна, и в таких случаях требуется нечто, способное сделать чуть более комфортной работу с датами, сетевое взаимодействие или что-то еще. Это прекрасно. Задействуйте любую библиотеку. Разница в том, что теперь вы будете применять ее, потому что возникла конкретная необходимость, а не потому, что она входила в стек, который вы всегда использовали.

В следующий раз, когда вам в голову придет идея небольшой программы, выведите свои знания о библиотеке классов Java из спящего режима, вместо того чтобы лезть в дебри JHipster. Хипстеризм — это уже не комильфо; жить простой жизнью — вот что актуально сейчас. Бьюсь об заклад, Морту нравилась простая жизнь.