100 ОШИБОК JAVA
и как их избежать

Тагир
Валеев

БОМБОРА
ИЗДАТЕЛЬСТВО
Москва

УДК 004.42
ББК 32.973.2-018.1
В15

Tagir Valeev

100 JAVA MISTAKES AND HOW TO AVOID THEM

Валеев, Тагир.

В15 100 ошибок Java и как их избежать / Тагир Валеев ; [перевод с английского В. В. Краснянской, Ю. В. Войтко]. — Москва : Эксмо, 2025. — 480 с. : ил. — (Manning: профессиональные книги для ИТ-специалистов).

ISBN 978-5-04-207668-8

Эта книга научит создавать высококачественный и надежный код на Java, используя инструменты статистического анализа. Вы узнаете, как обнаруживать и предотвращать распространенные ошибки программирования, оптимизировать процесс тестирования и отладки, а также повысить качество ПО еще на этапе разработки. Полезная информация представлена живым языком со множеством примеров и практических рекомендаций.

УДК 004.42
ББК 32.973.2-018.1

ISBN 978-5-04-207668-8

Моей матери, Наде

Оглавление

Предисловие

В августе 2023-го мне на почту упало вот что: «Здравствуйте, Кей! Меня зовут Тагир Валеев, возможно, вы помните меня по кое-каким обсуждениям в групповых рассылках. И еще мы лично встречались на одной конференции…»

Письмо показалось мне странным. Разумеется, я знал Тагира — и не только по тем «обсуждениям в групповых рассылках». Я автор классической книги «Java. Библиотека профессионала»[1], цель которой со времен Java 1.0 заключается в том, чтобы разъяснять этот язык квалифицированным программистам. Пока я обновляю текст (для 13-го издания), мне нужно рыться во множестве малоизвестных и каверзных аспектов самого ЯП и его библиотеки. Периодически я захожу в тупик с каким-нибудь изощренным вопросом, запускаю мою любимую поисковую систему и нахожу вдумчивый разбор, проведенный не кем иным, как Тагиром. Этот парень по-настоящему разбирается в Java. Ему точно нужно написать книгу…

О чем и было его сообщение. «Я работаю над своей самой первой книгой», — извещал Тагир и просил меня написать предисловие. Тему он преподносил так: «В тексте я акцентирую внимание на типичных и регулярно возникающих ошибках, которые разработчики допускают в программах на Java, и стараюсь дать советы о том, как их избежать».

Подобная идея не нова. Много лет назад мне пришлось научиться программировать на C, и меня ужасало, что люди вообще используют столь очевидно опасный язык, но я ничего не мог поделать. Только в этом ЯП имелись привязки к библиотекам, которые мне требовалось применить. Я совершил все распространенные ошибки и лишь позднее наткнулся на чудесную книгу Эндрю

[1] Хорстманн К. Java. Библиотека профессионала. Том 1. Основы. М.: Диалектика, 2020.

Кенига «Ловушки и подводные камни C»[1]. Моя жизнь протекала бы гораздо безмятежнее, если бы я тогда опирался на его текст, а не набивал шишки, тыкаясь наугад.

Конечно же, Java — это не C, а весьма тщательно разработанный ЯП, и его библиотека тоже отлично продумана. Правда, если Java 1.0 была несложной, то текущая версия возлежит на слоях хитросплетений, которые накапливались четверть века. И мы за минувшие годы тоже кое-чему научились… В общем, не все из того, что двадцать пять лет назад выглядело как хорошая мысль, выдержало проверку временем.

Соответственно, можно с легкостью рассмотреть сотню «подводных камней» в языке и программном интерфейсе Java, чем Тагир и занялся в своей книге. Каждый ее подраздел основан на примере кода, увиденного автором на практике. А кода он повидал немало, поскольку трудится в JetBrains — компании, создавшей IntelliJ IDEA, самую популярную (с большим отрывом) интегрированную среду разработки для Java.

Могу вас заверить, что вы многое узнаете, читая об этих ловушках: поймете, почему они опасны, и научитесь обходить их. Я программировал на Java со времен версии 1.0 и, пока изучал книгу, не раз ловил себя на том, что согласно киваю. В иных случаях у меня вырывалось: «Ого, я пользовался Java со времен версии 1.0, но о таком и не подозревал!» Негативные моменты, выбранные Тагиром, действительно важны. Они встречаются в реальном коде, поэтому всем, кто работает с Java, необходимо держать в уме такие проблемы и то, как с ними справляться.

Лучший путь к успеху — учиться на неудачах, а наименее затратный путь к успеху — учиться на неудачах других. Чтобы стать более умелым и уверенным в себе программистом на Java, идите вслед за Тагиром по дороге, проложенной через подобные ошибки, и извлекайте пользу из преподанных им уроков! О более сведущем провожатом и мечтать не нужно.

Кей Хорстманн,
почетный профессор Университета штата Калифорния в Сан-Хосе

---

[1] Koenig A. C Traps and Pitfalls. Addison-Wesley, 1989.

Вступление

Я уже около двадцати лет создаю коммерческое ПО, прежде всего на Java, хотя у меня есть опыт работы и с другими языками. Мной написаны сотни тысяч строк кода, а пока я изучал программы, составленные другими людьми, то прочел даже больше. Если на протяжении моей карьеры подтвердилась хотя бы одна азбучная истина, то звучит она так: «Люди неидеальны». Любой программист допускает ошибки. Самые умные разработчики ПО тоже способны внести баги, со стороны кажущиеся очевидными.

За ошибками же всегда следуют трудности. Программа не ведет себя так, как ожидалось, сбоит и выбрасывает исключения, пользователи жалуются, а вы тратите часы на то, чтобы воспроизвести, проанализировать и решить проблему. Даже для того, чтобы отыскать крошечный промах вроде опечатки, могут потребоваться серьезные усилия.

Примерно десять лет назад я обнаружил статический анализатор Java под названием FindBugs, что стало для меня настоящим откровением. В нашей компании раньше не применяли такие инструменты, и он за пару минут отыскал десятки реальных ошибок в проекте, над которым я тогда работал. FindBugs точно указывал на неправильный фрагмент кода, поэтому мне не пришлось ни ждать, пока мой промах затронет пользователей, ни выявлять причины проблемы. Я смог просто исправить ее.

Так я стал завзятым фанатом статического анализа. Однако же вскоре выяснилось, что возможности FindBugs не безграничны: он часто выдает ложные срабатывания и указывает на довольно мелкие недостатки, из-за чего затягивается изучение отчета. С другой стороны, он не сообщает о многих настоящих багах, которые следовало бы учесть. Я начал участвовать в работе над FindBugs, стараясь улучшить его, а через пару лет примкнул к потрясающей команде создателей IntelliJ IDEA — среды с великолепным статическим

анализатором. С тех пор моя основная задача состоит в том, чтобы помогать разработчикам уменьшить количество багов.

Примерно тогда же я изменил свой подход к исправлению ошибок. Я уже не просто решал проблемы, а постоянно спрашивал себя: «Почему они возникли? Возможно ли было избежать их в принципе? Есть ли риск появления схожих багов в будущем? Что способна сделать моя команда, чтобы нам больше не встречались такие трудности?»

Я выяснил, что не существует панацеи от ошибок. Иногда вам может помочь статический анализ, а порой лучше положиться на тестирование или использование утверждений. В некоторых случаях нужно усовершенствовать ваш стиль программирования, автоматизировать создание кода или же применять методы из библиотек, вместо того чтобы подготавливать что-либо вручную. Впрочем, многие недочеты встречаются регулярно, а значит, поддаются классификации, и есть стандартные способы их предотвратить. Вот почему я решил написать книгу, чтобы обобщить свой опыт и перечислить наиболее типичные ошибки, появляющиеся в программах на Java. Надеюсь, что другие разработчики, прочитав ее, будут готовы решать проблемы, с которыми еще не сталкивались лично.