Оглавление

Содержание

Предисловие

В современной разработке программного обеспечения тестирование управляет проектированием, реализацией, развитием, обеспечением качества и развертыванием программных систем. Чтобы быть эффективным разработчиком, вы должны стать успешным тестировщиком программного обеспечения, и эта книга поможет вам в этом.

Тестирование – это не что иное, как выполнение части программного обеспечения с целью увидеть, соответствует ли его поведение ожидаемому. Но тестирование – непростая задача. Его сложность становится очевидной, стоит только подумать о полном наборе тестов, которые необходимо спроектировать и выполнить. Какие из бесконечного множества возможных тестов следует написать? Достаточно ли полно протестирована система, чтобы ее можно было передать в промышленную эксплуатацию? Какие дополнительные тесты нужны и зачем? И если нужно изменить систему, то как настроить набор тестов, чтобы он способствовал, а не препятствовал будущим изменениям?

Книга не увиливает от таких сложных вопросов. Она охватывает такие ключевые методы тестирования, как проектирование по контрактам, тестирование на основе свойств, граничное тестирование, критерии достаточности тестирования, мутационное тестирование и правильное использование фиктивных объектов. А там, где это уместно, дает ссылки на исследовательские работы по теме.

В то же время эта книга наглядно показывает, что сами тесты и процесс тестирования остаются настолько простыми, насколько это возможно. Эта простота достигается за счет того, что всегда принимается точка зрения разработчика, который фактически разрабатывает и запускает тесты. Книга полна примеров, которые помогут читателю сразу приступить к применению приемов в своих проектах.

Эта книга появилась на основе курса, преподававшегося в Делфтском техническом университете в течение многих лет. Этот курс по тестированию программного обеспечения я ввел в учебную программу

бакалавриата в 2003 году. В 2016-м ко мне присоединился Маурисио Аниче, а в 2019 году он полностью взял этот курс на себя. Маурисио – превосходный лектор, и в 2021 году студенты избрали его учителем года факультета электротехники, математики и компьютерных наук.

В Делфтском техническом университете мы обучаем тестированию на самом первом курсе нашей программы бакалавриата по информатике и вычислительной технике. Было трудно найти книгу, соответствующую нашему представлению о том, что эффективный инженер-программист должен быть эффективным тестировщиком программного обеспечения. Многие академические учебники сосредоточены на результатах исследований. Многие книги, ориентированные на разработчиков, посвящены конкретным инструментам или процессам.

Книга «Эффективное тестирование программного обеспечения» Маурисио Аниче (Maurício Aniche) заполняет этот пробел между теорией и практикой. Она написана для действующих разработчиков и освещает самые современные методы тестирования программного обеспечения. В то же время она идеально подходит для университетских курсов бакалавриата и учит следующее поколение информатиков, как стать эффективными тестировщиками программного обеспечения.

– *Доктор Ари ван Дерсен* (Dr. Arie van Deursen),
профессор факультета программной инженерии,
Делфтский технический университет, Нидерланды

Книга «Эффективное тестирование программного обеспечения» Маурисио Аниче (Maurício Aniche) – это практическое введение, которое поможет разработчикам тестировать свой код. Это компактный обзор основ тестирования программного обеспечения, охватывающий основные темы, о которых должен знать каждый разработчик. Сочетание теории и практики в книге показывает глубину опыта Маурисио как ученого и действующего программиста.

Мой собственный путь в информатике был довольно случайным: несколько курсов программирования в университете, обучение на рабочем месте и, наконец, курс переквалификации, ведущий к получению докторской степени. Это заставило меня завидовать программистам, которые прошли нужные курсы в нужное время и обладали теоретической базой, которой мне не хватало. Я периодически обнаруживал, что та или иная моя идея, обычно с недоработанной реализацией, оказывалась устоявшейся концепцией, о которой я не слышал. Вот почему я считаю важным читать вводные материалы, такие как эта книга.

На протяжении большей части моей профессиональной карьеры я рассматривал тестирование как необходимое зло, которое в основном связано с утомительным выполнением текстовых инструкций вручную. В настоящее время стало совершенно очевидно, что автоматизация тестирования лучше всего выполняется с помощью компью-

теров, но потребовались десятилетия, чтобы это понимание нашло широкое распространение. Вот почему разработка через тестирование, когда я впервые столкнулся с ней, сначала показалась мне безумием, а затем необходимостью.

В своей практике мне приходилось видеть много малопонятного тестового кода. Это особенно заметно задним числом, когда отсутствует давление сроков или когда модель предметной области устоялась. Я считаю, что этот тестовый код можно было бы улучшить, если бы методы структурирования и анализа задач, описанные в этой книге, имели большее распространение среди программистов. Это не означает, что все мы должны стать учеными, однако простота применения концепций может иметь большое значение. Например, я считаю проектирование по контрактам полезным при работе с компонентами, сохраняющими состояние. Возможно, я не всегда добавляю явные предварительные и заключительные проверки в свой код, но знание концепций помогает мне думать или обсуждать, что должен делать код.

Очевидно, что тестирование программного обеспечения – важная тема для разработчиков, и эта книга поможет им начать ее осваивать. Для тех из нас, кто занимается этой темой немного дольше, книга послужит хорошим напоминанием о методах, которыми мы пренебрегли или, возможно, упустили в первый раз. В книге также имеются замечательные разделы, посвященные тестированию программного обеспечения как практике, в частности краткое введение в крупномасштабное тестирование и, что мне особенно нравится, обсуждение вопросов поддержания высокого качества тестового кода. В реальном мире многие наборы тестов превращаются в источник разочарования, потому что они не поддерживаются.

Опыт Маурисио проявляется в практических рекомендациях и эвристиках, которые он включает в объяснение каждой методики. Он тщательно описывает инструменты, но позволяет читателю найти свой собственный путь (хотя обычно лучший выбор – последовать его совету). И конечно же, содержание самой книги было тщательно протестировано, так как изначально она разрабатывалась на основе его курса в Делфтском техническом университете.

Я сам часто встречался с Маурисио, когда читал лекции для его курса, после которых мы заходили в палатку в историческом центре города, чтобы отведать маринованной сельди (имеющей уникальный вкус для гурманов Северной Европы). Мы обсуждали методы программирования и тестирования, а также жизнь в Нидерландах. Я был впечатлен его стремлением дать все самое лучшее своим студентам и его идеями для исследований. Я с нетерпением жду, когда снова смогу сесть на поезд до Делфта.

– *Доктор Стив Фриман* (Dr. Steve Freeman),
автор книги «Growing Object-Oriented Software,
Guided by Tests» (Addison-Wesley Professional)

Вступление

Каждый разработчик программного обеспечения помнит конкретную ошибку, повлиявшую на его карьеру. Позвольте мне рассказать вам о моей ошибке. В 2006 году я занимал должность технического руководителя небольшой группы разработчиков, работавшей над созданием приложения для контроля платежей на заправочных станциях. В то время я заканчивал бакалавриат по информатике и начинал свою карьеру как разработчик программного обеспечения. До этого мне приходилось работать только над двумя серьезными веб-приложениями. И, как ведущий разработчик, я очень серьезно относился к своим обязанностям.

Система должна была напрямую связываться с подающими насосами. Как только клиент заканчивал заправку, бензоколонка уведомляла нашу систему и приложение запускало свой процесс: сбор информации о покупке (тип топлива, количество в литрах), расчет конечной стоимости, проведение пользователя через процесс оплаты и сохранение информации для будущей отчетности.

Программная система должна была работать на выделенном устройстве с процессором 200 МГц, 2 Мбайт ОЗУ и несколькими мегабайтами постоянной памяти. Это была первая попытка внедрения устройства в бизнес. То есть не было никакого предшествующего проекта, опыт разработки которого мы могли бы изучить или позаимствовать из него код. Мы также не могли повторно использовать какие-либо внешние библиотеки, и нам даже пришлось реализовать свою упрощенную базу данных.

Система обслуживала заправочные станции, поэтому моделирование их оборудования стало жизненно важной частью нашего процесса разработки. Мы писали новую функцию, запускали систему, запускали симулятор, моделировали несколько покупок топлива и вручную проверяли, правильно ли реагирует система.

Через несколько месяцев мы реализовали несколько важных функций. Наши тесты (выполнявшиеся вручную), включая тесты, проведенные компанией, прошли успешно. У нас появилась версия, которую можно было протестировать в реальных условиях! Но испытания в реальных условиях оказались непростыми: команде инженеров при-

шлось внести физические изменения на заправочной станции, чтобы насосы могли взаимодействовать с нашим программным обеспечением. К моему удивлению, компания решила запланировать первый пуск в Доминиканской Республике. Я был рад не только увидеть, как реализуется мой проект, но и посетить такую прекрасную страну.

Я был единственным разработчиком, который ездил в Доминиканскую Республику, поэтому я отвечал за исправление всех ошибок, обнаруженных на месте. Следил за установкой и за первым запуском программного обеспечения. Весь день наблюдал за системой, и все выглядело нормально.

А вечером мы пошли праздновать. Пиво было холодным, и гордость переполняла меня. Я лег спать рано, чтобы утром успеть подготовиться к встрече с заинтересованными сторонами и обсуждению следующих шагов проекта. Но в 6 утра в моем номере зазвонил телефон. Это был владелец опытной АЗС: «По всей видимости, ночью произошел сбой программного обеспечения. Ночная смена не знала, что делать, а бензоколонки не выдавали ни капли топлива, поэтому за всю ночь станция ничего не смогла продать!» Я был потрясен. Как такое могло случиться?

Я отправился прямо на станцию и начал отладку системы. Ошибка была вызвана непроверенной ситуацией, когда количество заправок превышало возможности системы. Мы знали, что используем встроенное устройство с ограниченным объемом памяти, поэтому приняли меры предосторожности. Но мы ни разу не проверили, что произойдет, если предел будет достигнут, – и это была наша ошибка!

Все наши тесты проводились вручную: для имитации заправки мы подходили к симулятору, нажимали кнопку, имитирующую включение насоса, качающего топливо, ждали несколько секунд (считалось, что чем дольше мы ждали, тем больше литров топлива «купили»), а затем останавливали процесс заправки. Чтобы сымитировать 100 заправок, нужно было 100 раз нажать кнопку на симуляторе. Это довольно медленная и утомительная процедура. Поэтому во время разработки мы пробовали выполнить по две-три заправки кряду. Однажды мы протестировали механизм обработки исключений, но этого было недостаточно.

Первая программная система, над которой я работал в роли ведущего разработчика, не проработала даже суток! Что я мог сделать, чтобы предотвратить ошибку? Пришло время изменить способ создания программного обеспечения, и это привело меня к тому, что я больше узнал о тестировании программного обеспечения. Конечно, в колледже нас познакомили с разными методами тестирования и рассказывали о важности тестирования программного обеспечения, но мы часто начинаем понимать ценность некоторых вещей, только когда они становятся нужны.

Сегодня я не могу представить создание системы без набора автоматизированных тестов. Такой набор может сказать мне за считанные секунды, правильный или неправильный код я написал, поэтому я работаю намного продуктивнее. Эта книга – моя попытка помочь разработчикам избежать моих ошибок.

Благодарности

Это не первая моя техническая книга, но первая, в которую я вложил всю свою душу. И это стало возможным только благодаря помощи и поддержке многих людей.

В первую очередь я хочу отметить самого важного человека, подтолкнувшего меня к написанию этой книги, – профессора, доктора Ари ван Дерсена (Arie van Deursen). Ари был моим научным руководителем, а затем – коллегой в Исследовательской группе разработки программного обеспечения (Software Engineering Research Group, SERG) в Делфтском техническом университете. В 2017 году он пригласил меня преподавать его курс по тестированию программного обеспечения для первокурсников факультета информатики (да, в Делфте обучают тестированию программного обеспечения с самого первого курса!). Преподавая вместе с ним, я многое узнал о его взглядах на теоретическое и практическое тестирование программного обеспечения. Страсть Ари к обучению людей этой теме вдохновила меня, и я продолжаю работать над совершенствованием этого курса (теперь он стал полностью моим). Эта книга – естественный результат интереса, который он пробудил во мне много лет назад.

Другие коллеги из Делфтского университета тоже оказали на меня значительное влияние. Фрэнк Малдер (Frank Mulder), который сейчас вместе со мной преподает тестирование программного обеспечения, является очень опытным разработчиком программного обеспечения и не боится столкнуться с любыми сложностями в разработке программного обеспечения. Я потерял счет нашим дискуссиям о различных практиках за эти годы. Мы также проводим наши обсуждения в аудитории, и студенты получают почти такое же удовольствие, как и мы, когда мы излагаем свои взгляды. Многие практические дискуссии в этой книге начались с бесед с Фрэнком.

Я благодарю Воутера Полета (Wouter Polet). Воутер был моим ассистентом в течение многих лет. Когда началась пандемия Covid, я сказал Воутеру, что мы должны сделать конспекты лекций доступными для студентов, которые не могут посещать занятия. Он воспринял это

как задание и быстро создал веб-сайт, содержащий стенограммы видеороликов, снятых несколькими годами ранее.

Эти стенограммы стали моими конспектами лекций, которые позже превратились в книгу. Без поддержки Воутера эта книга едва ли появилась бы на свет. Я также благодарю Сару Юхошову (Sára Juhošová), присоединившуюся к нам в роли главного ассистента преподавателя и сыгравшую важную роль в организации поддержки курса практическими инструментами. Не знаю, прочтет ли кто-нибудь еще эту книгу так же внимательно, как она. Сара потратила много времени на доводку моих плохо написанных предложений – без ее помощи книга не была бы такой, какой она стала. Наконец, я благодарю Надин Куо (Nadine Kuo) и десятки ассистентов преподавателей, которые на протяжении многих лет помогали мне улучшать материал курса. Многие люди помогали мне (их так много, чтобы было невозможно перечислить их всех здесь), и все они вложили частичку себя в эту книгу.

Спасибо профессору, доктору Энди Зайдману (Andy Zaidman) и доктору Аннибале Паничелле (Annibale Panichella). Энди много лет был моим коллегой, а до этого служил образцом для подражания. Я читал его статьи с увлечением и интересом. Любовь Энди к эмпирическому тестированию программного обеспечения вдохновила меня приехать в Делфт для постдокторантуры. Аннибале много лет был моим коллегой по офису и является лучшим исследователем в области разработки программного обеспечения, которого я знаю. Аннибале – эксперт мирового уровня по тестированию программного обеспечения на основе поиска, и я многое узнал от него по этой теме (и немалую часть из этого за кружкой пива). Я мало рассказываю об этом в книге, но Аннибале показал мне, как далеко может зайти искусственный интеллект в тестировании программного обеспечения, и заставил меня задуматься о том, что должны делать разработчики (люди).

Но не только профессорско-преподавательский коллектив Делфтского университета оказал на меня влияние и способствовал появлению этой книги. Во-первых, я хочу поблагодарить Альберто Соузу (Alberto Souza). Альберто – один из моих лучших друзей и самых прагматичных разработчиков, которых я знаю. Когда я решил начать работу над книгой, мне нужна была положительная поддержка, и Альберто дал ее мне. Я не уверен, что без его постоянных положительных отзывов смог бы закончить книгу.

Также хочу поблагодарить Стива Фримана (Steve Freeman). Стив – один из авторов известной книги «Growing Object-Oriented Systems, Guided by Tests» (Addison-Wesley Professional, 2009). Стив был основным докладчиком на семинаре по разработке через тестирование (Test-Driven Development, TDD) в 2011 году, на котором я выступил со своим первым академическим докладом. В настоящее время Стив каждый год читает лекцию в рамках моего курса тестирования как приглашенный гость. Я большой поклонник Стива и горячо поддерживаю его видение разработки программного обеспечения, а его книгу считаю одной из самых весомых из всех, что я когда-либо читал. Мне

также нравится обсуждать с ним темы разработки программного обеспечения, потому что он страстный и уверенный. Мои главы о TDD и фиктивных объектах не отражают образа мыслей Стива, и все же он определенно повлиял на мои взгляды на тестирование.

Я благодарен сотрудникам Manning Publications. Они помогли мне сформировать идеи и воплотить их в книгу – окончательная версия книги сильно отличается (в лучшую сторону) от первоначального замысла. Спасибо вам, Кристен Уоттерсон (Kristen Watterson), Тиффани Тейлор (Tiffany Taylor), Тони Арритола (Toni Arritola), Ребекка Райнхарт (Rebecca Rinehart), Мелисса Айс (Melissa Ice), Иван Мартинович (Ivan Martinovic), Пол Уэллс (Paul Wells), Кристофер Кауфманн (Christopher Kaufmann), Энди Маринкович (Andy Marinkovich), Айра Дучич (Aira Ducic), Джейсон Эверетт (Jason Everett), Азра Дедич (Paul Wells) и Майкл Стивенс (Michael Stephens). Я также благодарю Фрэнсис Буонтемпо (Frances Buontempo) – разработчика, которому было поручено проверить все примеры в этой книге от начала до конца. Ее своевременные и содержательные отзывы помогли существенно улучшить книгу.

Спасибо всем рецензентам: Амит Ламба (Amit Lamba), Атул С Хот (Atul S Khot), Дэвид Кабреро Соуто (David Cabrero Souto), Франческо Базиле (Francesco Basile), Джеймс Лю (James Liu), Джеймс МакКин Вуд (James McKean Wood), Джереми Аллен (Jereme Allen), Джоэл Холмс (Joel Holmes), Кевин Орр (Kevin Orr), Маттео Баттиста (Matteo Battista), Майкл Холмс (Michael Holmes), Нельсон Х. Феррари (Nelson H. Ferrari), Прабхути Пракаш (Prabhuti Prakash), Роберт Эдвардс (Robert Edwards), Шон Лэм (Shawn Lam), Стивен Бирн (Stephen Byrne), Тимоти Вулдридж (Timothy Wooldridge) и Том Мэдден (Tom Madden) – ваши отзывы и предложения помогли сделать эту книгу лучше.

Наконец, я благодарю свою любимую жену Лауру. Я подписал контракт с издательством Manning за несколько недель до рождения нашего ребенка. Она была невероятно терпеливой и поддерживала меня все это время. Без нее я не смог бы написать книгу (да и сделать многое другое в этой жизни). Нашему ребенку сейчас семь месяцев, и, хотя он еще мало что знает о тестировании, он – главная причина, почему я хочу сделать мир лучше.

О книге

Как и многое другое в мире разработки, тестирование программного обеспечения – это искусство. За последнее десятилетие мы узнали, что автоматизированные тесты – лучший способ тестирования программного обеспечения. Компьютеры могут выполнять сотни тестов за доли секунды, и такие наборы тестов позволяют компаниям уверенно обновлять и развертывать программное обеспечение десятки раз в день.

Автоматизации тестов посвящено огромное количество ресурсов (книг, руководств и онлайн-курсов). Независимо от используемого языка или вида разрабатываемого программного обеспечения вы сможете найти информацию, руководствуясь которой сумеете выбрать правильный инструмент. Но нам не хватает ресурсов, связанных с разработкой эффективных тестов. Автоматизация выполняет тесты, написанные разработчиком. Если они некачественные или не охватывают какие-то части кода, где могут скрываться ошибки, то такой набор тестов мало полезен.

Сообщество разработчиков относится к тестированию программного обеспечения как к виду искусства, когда вдохновленные разработчики с творческим подходом создают более эффективные наборы тестов, чем все остальные. Но в этой книге я бросаю вызов такому отношению и показываю, что тестирование программного обеспечения не обязательно должно зависеть от знаний, опыта или творческих способностей: его в значительной мере можно систематизировать.

Следуя эффективному систематическому подходу к тестированию программного обеспечения, мы больше не полагаемся на умение опытных разработчиков писать эффективные тесты. И если мы найдем способы автоматизировать большую часть процесса тестирования, то это позволит нам сосредоточиться на тестах, требующих творческого подхода.

Кому адресована книга

Эта книга написана для разработчиков, желающих больше узнать о тестировании или отточить свои навыки. Если вы имеете многолет-

ний опыт разработки программного обеспечения и написали множество автоматических тестов, но при этом всегда полагаетесь только на свою интуицию, решая, каким должен быть следующий тест, то эта книга поможет вам структурировать ваш мыслительный процесс.

Книга будет полезна разработчикам с разным уровнем знаний: начинающие смогут следовать всем примерам кода и методам, которые я представляю; опытные познакомятся с методами, которые им, возможно, еще не знакомы, и в каждой главе будут учиться на обсуждениях практических приемов.

Методы тестирования, которые я описываю, предназначены для разработчиков, пишущих код. Тестировщики программного обеспечения, преданные своему делу, которые рассматривают программы как черные ящики, тоже могут читать эту книгу, но должны иметь в виду, что она написана с точки зрения разработчика, написавшего тестируемый код.

Примеры в этой книге написаны на Java, но я сделал все возможное, чтобы избежать причудливых конструкций, которые могут быть незнакомы разработчикам, использующим другие языки программирования. Я также постарался обобщить приемы, чтобы, даже если код не переводится непосредственно в ваш контекст, вы могли заимствовать хотя бы идеи.

В главе 7 я обсуждаю проектирование систем с учетом простоты их тестирования. Эти идеи в большей степени ориентированы на разработку объектно ориентированных программных систем, чем функциональных. Однако это единственная глава, которая может не иметь прямого применения к функциональному программированию.

Организация книги

Эта книга состоит из 11 глав. В главе 1 я привожу аргументы в пользу систематического и эффективного тестирования программного обеспечения. Здесь вы познакомитесь с примером, когда два разработчика реализуют одну и ту же функцию – один небрежно, а другой систематически, – подчеркивающим различия между их подходами. Затем мы обсудим различия между модульными, интеграционными и системными тестами и я обосную, почему разработчики должны в первую очередь сосредоточиться на быстрых модульных и интеграционных тестах (согласно известной пирамиде тестирования).

Глава 2 знакомит с тестированием предметной области. Эта практика тестирования фокусируется на инженерных тестах, основанных на требованиях. Когда речь идет о требованиях, команды разработчиков программного обеспечения используют разные методы для тестирования – пользовательские истории, унифицированный язык моделирования (Unified Modeling Language, UML) или внутренние форматы. Каждый сеанс тестирования должен начинаться с формулирования требований к разрабатываемой функции.

В главе 3 показано, как использовать исходный код и структуру программы для расширения тестов, которые мы разрабатываем с приме-

нением методов тестирования предметной области. Мы можем запускать инструменты оценки покрытия кода тестами и, анализируя результаты, выявлять и исследовать те части кода, которые не были охвачены первоначальным набором тестов. Некоторые разработчики не считают оценку покрытия кода полезным показателем, но в этой главе я надеюсь убедить вас, что при правильном применении эта оценка может быть важнейшей частью процесса тестирования.

В главе 4 мы обсудим идею о том, что качество кода зависит не только от полноты тестирования, но также от организации кода и надежности классов и методов, на которые могут опираться другие классы и методы системы. Проектирование по контрактам позволяет сделать пред- и постусловия в коде явными, благодаря чему, если что-то пойдет не так, программа остановится, не вызывая других проблем.

Глава 5 знакомит с тестированием на основе свойств. Согласно этой методике тесты пишутся не на одном конкретном примере, а проверяют все свойства программы, причем за создание входных данных, соответствующих свойствам, отвечает платформа тестирования. Освоить эту технику непросто, потому что порой довольно сложно выразить свойства, и для этого требуется практика. Кроме того, тестирование на основе свойств подходит не для всех фрагментов кода, однако в этой главе вы найдете множество примеров, демонстрирующих данную идею, которые помогут вам лучше понять ее.

В главе 6 обсуждаются практические вопросы, выходящие за рамки разработки хороших тестов. В сложных системах одни классы зависят от других, и разработка тестов может стать весьма утомительным занятием. Здесь я познакомлю вас с фиктивными объектами и заглушками, которые позволят игнорировать некоторые зависимости во время тестирования. Мы также обсудим важный компромисс: фиктивные объекты упрощают тестирование, но они делают тесты более тесно привязанными к производственному коду, что может усложнить дальнейшее их развитие. В главе обсуждаются плюсы и минусы фиктивных объектов, а также перечисляются ситуации, когда их можно или нельзя использовать.

В главе 7 я объясняю разницу между системами, разработанными с учетом и без учета поддержки тестирования. Мы обсудим несколько простых шаблонов проектирования, которые помогут вам писать легко контролируемый и простой для наблюдения код (мечта любого разработчика, когда дело доходит до тестирования). Эта глава посвящена разработке программного обеспечения, а также тестированию – как вы увидите, между ними существует тесная связь.

В главе 8 обсуждается методика разработки через тестирование (Test-Driven Development, TDD), согласно которой тесты пишутся до основного кода, который они тестируют. TDD – чрезвычайно популярный метод, особенно среди практикующих методики гибкой разработки. Я рекомендую прочитать эту главу, даже если вы уже знакомы с TDD, потому что у меня несколько необычный взгляд на то, как

следует применять TDD, и, в частности, на случаи, в которых, как мне кажется, TDD не играет большой роли.

В главе 9 я выйду за рамки модульных тестов и уделю внимание интеграционным и системным тестам. Вы увидите, как методы, обсуждавшиеся в предыдущих главах (например, предметное и структурное тестирование), могут непосредственно применяться в этих тестах. Для написания интеграционных и системных тестов требуется гораздо больше кода, поэтому, если не организовать код должным образом, может получиться весьма сложный набор тестов. В этой главе будет представлено несколько передовых методов разработки надежных и простых в сопровождении наборов тестов.

В главе 10 обсуждается передовой опыт работы с тестовым кодом. Написание тестов в автоматическом режиме является фундаментальной частью нашего процесса. Для нас также желательно, чтобы тестовый код было легко понять и поддерживать. В этой главе будут представлены лучшие (чего мы хотим от наших тестов) и худшие практики (чего мы не хотим от наших тестов).

В главе 11 я вернусь к некоторым понятиям, затронутым в книге, повторю некоторые важные темы и дам несколько последних советов о том, что делать дальше.

О чем не рассказывается в книге

В этой книге не рассматривается тестирование программного обеспечения с использованием конкретных технологий и окружений, например здесь вы не найдете обсуждения выбора фреймворка или способов тестирования мобильных приложений, приложений React или распределенных систем.

Я уверен, что все практики и методы, о которых я рассказываю, применимы к любой программной системе. Эта книга может служить основой для любого вида тестирования. Однако в каждой области есть свои методы и инструменты тестирования; поэтому после прочтения вам следует поискать дополнительные ресурсы, касающиеся конкретного типа приложения, которое вы создаете.

Эта книга посвящена функциональному тестированию (здесь вы не найдете приемов тестирования производительности, масштабируемости и безопасности). Если ваше приложение требует такого типа тестирования, я предлагаю вам поискать специальные ресурсы по этой теме.

О примерах программного кода

Все идеи и концепции в этой книге иллюстрируются программным кодом на Java. Однако код написан так, чтобы разработчики на других языках могли читать его и понимать предлагаемые методы.

Ради экономии места листинги кода не включают все необходимые инструкции импорта. Однако вы можете найти полный исходный код

на веб-сайте книги (www.manning.com/books/Effective-software-testing) и на GitHub (https://github.com/efficient-software-testing/code). Код был протестирован с Java 11, и я не думаю, что возникнут проблемы с более новыми версиями Java.

У меня также есть специальный веб-сайт для этой книги по адресу www.efficient-software-testing.com, где я делюсь свежей информацией по тестированию программного обеспечения. Кроме того, вы можете подписаться на мою бесплатную рассылку новостей.

Живое обсуждение книги

Приобретая книгу «Эффективное тестирование программного обеспечения», вы получаете бесплатный доступ к частному веб-форуму, организованному издательством Manning Publications, где можно оставлять комментарии о книге, задавать технические вопросы, а также получать помощь от автора и других пользователей. Чтобы получить доступ к форуму и зарегистрироваться на нем, откройте в веб-браузере страницу https://livebook.manning.com/book/effective-software-testing/discussion. Узнать больше о форумах Manning и познакомиться с правилами поведения можно по адресу https://livebook.manning.com/discussion.

Издательство Manning обязуется предоставить своим читателям место встречи, где может состояться содержательный диалог между отдельными читателями и между читателями и автором. Но со стороны авторов отсутствуют какие-либо обязательства уделять форуму какое-то определенное внимание – их присутствие на форуме остается добровольным (и неоплачиваемым). Мы предлагаем задавать авторам стимулирующие вопросы, чтобы их интерес не угасал! Форум и архив с предыдущими обсуждениями остается доступным на сайте издательства, пока книга продолжает издаваться.

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге, – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.