От издателя

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.

Скачивание исходного кода примеров

Скачать файлы с дополнительной информацией для книг издательства «ДМК Пресс» можно на сайте www.dmkpress.com на странице с описанием соответствующей книги.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг – возможно, ошибку в основном тексте или программном коде, – мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу dmkpress@gmail.com, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Apress очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Я бы хотел посвятить эту книгу бабочкам, которые порхают на ветру.

Хотя постойте, это слишком просто.

Я бы хотел посвятить эту книгу игрокам, которые, как выражаются британцы, «a flutter on the horses» (перевод: «делают ставки на лошадей [во время скачек]»).

Да, вообще-то это и есть реальное использование слова flutter [прим. пер.: одно из значений flutter – «делать небольшие ставки»], но это тоже слишком просто.

Нет, я бы хотел посвятить эту книгу всему неизведанному, что человечеству только предстоит открыть или даже создать.

По своей натуре я пессимист, но я борюсь с этим каждый день, поскольку я признаю, что вселенная – это удивительнейшее место, и несмотря на все, что нам говорят в вечерних новостях, человечество способно эту великую и чудесную красоту создать.

И с моей заявленной целью – быть бессмертным, потому что смерть – это не я, это то, что было до меня, я просто хочу идти дальше и пропустить это, – я с нетерпением жду, чтобы увидеть всё на свете!

Оглавление

Об авторе

Фрэнк Заметти – автор ряда популярных технических книг, был программистом около 40 лет, 25 из которых занимался этим профессионально, надо же ему было что-то кушать. В те времена на его визитке значилось *архитектор*, хотя он продолжал писать тупой код и каждый день крутился как мог. Фрэнк – первоклассный гик: если он не заставляет свой компьютер выполнять приказы (скорее всего, дьявольские), то занят просмотром, чтением или написанием научной фантастики, моделированием рельсотрона, катушки Теслы или любой другой штуковины, способной прикончить его в любой момент; он любит без причины цитировать *«Вавилон 5»*, *«Властелин колец»*, *«Хроники Риддика»* или *«Настоящих гениев»*, а также играть в компьютерные игры. Еще Фрэнк – рок-музыкант (клавишник) и заядлый любитель пиццы и других углеводов. У него есть жена, собака и несколько детей. Если подвести итог его крутости, то он тот, кто всегда готов воскликнуть «С тобой мой меч!» (ну да, обычно гики цитируют *«Властелин колец»* без очевидной причины).

О техническом рецензенте (обозревателе)

Герман ван Росмален работает разработчиком и архитектором программного обеспечения для De Nederlandsche Bank NV, центрального банка в Нидерландах. За его плечами более 30 лет опыта разработки приложений на разных языках программирования. Герман был вовлечен в создание приложений для мейнфреймов, десктопов, серверов, веб-браузеров и смартфонов. Последние 4 года он занимается в основном разработкой на .NET/C# и Angular после 15 лет работы с Java.

Герман живет в маленьком городке Пейнаккер в Нидерландах со своей женой Лизбет и детьми Барбарой, Леони и Рамоном. Наравне с разработкой софта в свободное время Фрэнк тренирует женскую футбольную команду на протяжении последних 10 лет.

И конечно же, он болеет за Фейеноорд (футбольный клуб из города Роттердам в Нидерландах, который считается одним из ведущих клубов страны)!

О переводе

Данная книга была переведена специалистами компании Binwell, а также действующими преподавателями Binwell University. Надеемся, что наши книги и переводы позволят вам легко освоить новые технологии, а также построить успешную карьеру в сфере информационных технологий.

Над переводом книги «Flutter на практике» работали:

Артем Тищенко, переводчик – руководитель направления Mobile в компании Binwell, специалист в разработке нативных (iOS Swift/Objective C), а также кросс-платформенных (Xamarin и Flutter) мобильных приложений. Ментор Binwell University, соавтор ряда популярных статей для Microsoft Developer Blogs и Хабрахабры.

Вячеслав Черников, редактор перевода – руководитель разработки в компании Binwell, руководитель Binwell University. Работает в сфере Mobile и разработки ПО с 2005 года. Создавал приложения и игры для iOS, Android, Symbian, Windows Mobile, Meego, Linux и Windows UWP. Имеет богатый опыт разработки нативных (iOS Swift/Objective C) и кросс-платформенных (Xamarin, Qt, PhoneGap/HTML5, Unity) мобильных приложений. Автор книги «Разработка мобильных приложений на C# для iOS и Android» (ДМК Пресс, 2020), а также популярных статей для Хакера, Хабрахабры, Microsoft Developer Blogs, спикер, преподаватель и немного [безумный] ученый.

Также выражаем благодарность **Александру Рыжкову** и **Сергею Селютину** за помощь с корректировками.

Благодарности

Если вы никогда не занимались написанием книги, то я открою вам секрет: написание самой книги – это лишь малая часть большой работы над ней. Иногда, мне кажется, наименьшая часть!

Поэтому я хочу поблагодарить всех, кто усердно работал и помогал с этим проектом (не важно, лично или с редактурой), включая Нэнси Чен, Луиса Корригана, Джеймса Маркхэма, Германа ван Росмалена, Вэлмода Спара и Даниша Кумара. Если вашего имени нет в списке, хотя оно должно здесь быть, я приношу свои искренние извинения и благодарю вас.

Также я хотел бы поблагодарить Ларса Бака и Каспера Ланга за создание Dart, довольно элегантного и очень приятного в использовании языка программирования, лежащего в основе Flutter. Говорю как человек, который создал свой собственный язык и набор инструментов для него много лет назад, я очень-очень ценю то, что вы сделали, ребята. Честь вам и слава!

Работа над книгой по Flutter требует от меня благодарности почти всей команде разработчиков этого фреймворка. Я занимаюсь мобильной разработкой около 20 лет (посмотрите на etherient.com, страницу Products, а конкретно Eliminator – игра, которую я выпустил в 2001 году для платформы Pocket PC; я верю, что это было мое первое мобильное приложение, по крайней мере первая удачная попытка), и тогда, насколько я могу судить, я использовал достаточно много утилит, фреймворков и библиотек. Учитывая весь этот опыт, я могу с уверенностью сказать, что Flutter даже с первой версии был на голову выше всех.

Это поразительно, сколько команда Flutter смогла сделать за такой относительно короткий период времени, и без их тяжелой работы я бы, очевидно, не написал эту книгу! Я с нетерпением жду возможности все больше и больше использовать Flutter, а также мне очень интересно, что будет с Flutter дальше!

Введение

Создание кросс-платформенных приложений, которые выглядят и работают как нативные, – сложная задача даже после многих лет работы над ее решением. Вы можете писать отдельные программы для каждой платформы и пытаться сделать их дизайн максимально похожим. Но фактически это значит написать одно приложение несколько раз. Заказчики, как правило, не готовы за такое платить!

Может, взять HTML-страницу и использовать один и тот же код для всех платформ? Но тогда вы можете остаться в дураках с точки зрения возможностей самого устройства, не говоря уже о том, что производительность часто находится на низком уровне (существуют способы минимизирования проблем, но они никогда не исчезнут полностью).

Я занимаюсь этим уже второе десятилетие (серьезно!), поэтому я видел такое много раз. И если я замечу на горизонте образ единорога, то буду сомневаться до конца. Однако если, подойдя поближе, окажется, что единорог действительно реален?

Итак, я представляю вам единорога, который на самом деле существует, – Flutter!

Благодаря талантливым инженерам из Google Flutter – это платформа, позволяющая писать (более или менее) кросс-платформенный код, который одинаково работает на Android и iOS, при этом обеспечивая производительность, идентичную нативным приложениям. Flutter, созданный с использованием современных инструментов и методов разработки, открывает программистам мир мобильной разработки, в котором, даже не побоюсь сказать, *весело* работать!

В этой книге вы познакомитесь с Flutter, создав два реальных приложения вместо надуманных примеров, предназначенных лишь для демонстрации технологии. На этом пути вы узнаете много реальных тонкостей, включая проблемы, с которыми я столкнулся, и решения, которые я нашел. При этом вы получите практический опыт реального использования Flutter, который подготовит вас к созданию собственных приложений в будущем.

Вы также узнаете то, как создавать серверные приложения на Node.js и Web Sockets. Эти бонусы являются полезным дополнением к описанию самого фреймворка Flutter и языка Dart.

Кроме того, вы сможете создать дополнительное третье приложение, которое разительно отличается от первых двух, – игру! Да, мы вместе создадим игру на Flutter, чтобы рассмотреть такие возможности, которые редко встречаются в настоящих приложениях, но дают вам взглянуть на фреймворк с разных сторон и получить максимум опыта. Эта игра может быть не совсем «практичной», но игры всегда увлекательны, а немного веселья никому не повредит!

Дочитав до конца, вы получите представление о том, что такое Flutter, и у вас будет отличная возможность создать свое новое крутое приложение на его основе.

Если вы были компьютерным энтузиастом в 80-х годах, то наверняка помните, каково это было – набивать на своем компьютере машинный код из журнальной статьи в 20 страниц мелким шрифтом, чтобы сыграть в игру или запустить приложение для сверки ваших доходов и расходов (да, мы действительно это делали – были даже радиостанции, транслировавшие исходники, которые затем компилировались в готовое приложение с помощью специальной утилиты, подобно тому как сейчас кодируется звук для передачи по телефонной линии).

Итак, прежде чем начать, я предлагаю вам зайти на веб-сайт Apress, найти эту книгу и скачать себе исходные коды примеров. Это позволит обойтись без стирания пальцев в кровь, перепечатывая листинги из книги!

Не забывайте, лучший способ чему-то научиться – это делать, поэтому обязательно скачайте и измените примеры под себя, увидев своими глазами, на что эти изменения повлияют. После прочтения каждой главы, связанной с приложением, заходите в исходные коды и пробуйте добавить одну или две свои функции, и я даже дам вам пару советов. Думаю, вскоре вы поймете, что благодаря возможностям Flutter небольшие изменения могут существенно повлиять на то, что появляется на экране.

Итак, приготовьтесь к приятной и информативной поездке в мир Flutter!

Я надеюсь, что вам понравится эта книга и вы многому научитесь. Это моя главная цель! Так что перекусите, сядьте в кресле поудобнее, подготовьте ноутбук и приступайте. Вас ждут приключения! (Да, я прекрасно понимаю, как банально это звучит.)

Исходные коды примеров вы можете найти в репозитории на GitHub: `https://github.com/Apress/practical-flutter`.

ГЛАВА 1

FLUTTER: ПЛАВНОЕ ПОГРУЖЕНИЕ

Поехали!

Если вы спросите десятерых разных разработчиков мобильных приложений, как они их разрабатывают для Android и iOS, то, скорее всего, получите десять разных ответов. Но это ненадолго, благодаря дебютанту на этой сцене – Flutter.

В первой главе мы рассмотрим разработку для мобильных устройств, то, как Flutter вписывается в эту картину и, возможно, полностью ее меняет. Мы начнем с ним работать, получим базовое представление о языке и фреймворке, а также подготовим почву для создания реальных приложений.

Итак, давайте сразу поговорим о том, что же такое мобильная разработка.

Медитации над бездной

Разработка софта – это непростая задача!

Я не хочу утомлять вас историей, но факт в том, что я начал, так или иначе, программировать с 7 лет, а это означает, что я занимаюсь этим почти 40 лет (около 25 из них профессионально). Я много повидал и много сделал, но главное, что я понял: разработка софта – это непростая задача. Конечно, некоторые отдельные задачи и проекты могут быть простыми, но в целом это довольно сложная работа, которой мы занимаемся!

И это мы еще не говорим о мобильной разработке, которая куда сложнее!

Я начал разработку мобильных приложений примерно два десятилетия назад, еще во времена Windows CE/Pocket PC и Palm Pilot (были и другие платформы, но именно эти две были единственными настоящими игроками на рынке). Тогда все было не так уж и плохо, несмотря на ограниченный набор устройств и возможностей инструментов разработки, которыми мы располагали. Безусловно, использование этих инструментов было не таким приятным, как сегодня, но был всего один способ разработки приложений для Pocket PC, один способ разработки приложений для Palm OS. Звучит не очень, но отсутствие выбора приводит к отсутствию путаницы, что является одной из самых больших трудностей в области разработки софта на сегодняшний день.

А еще, хотя сегодня это считается непопулярным, тогда не было понятия кросс-платформенной разработки. Раньше приходилось писать код дважды, чтобы приложение работало на обеих платформах. Учитывая различия между ними, это было не так-то просто.

С тех пор индустрия мобильных устройств и приложений претерпела немало эволюционных изменений, подъемов и падений. Долгое время у нас было много платформ для поддержки: Android, iOS, webOS, Tizen, Windows Mobile и несколько других, которые я даже не помню. Все это время перенос

приложений между платформами был нормой, поскольку не было хорошего кросс-платформенного подхода, по крайней мере без существенных компромиссов. Да, со временем стало проще, потому что улучшился инструментарий для нативной разработки. Apple выпустила свой SDK для iOS в 2008 году, а Google выпустил свой Android SDK год спустя – в 2009-м. Нам приходилось разрабатывать приложения для каждой платформы, поскольку разработка iOS основана на языке Objective-C (сегодня чаще на языке Swift), в то время как разработка Android основана преимущественно на языке Java (теперь чаще на Kotlin).

В конце концов, количество платформ стало сокращаться. На сегодняшний день это гонка Android и iOS, хотя есть и другие платформы, которые чаще всего используются *только* при решении специфических задач. В связи с этим применение кросс-платформенных инструментов становится более привлекательным.

Наша прогрессивная эпоха интернета предлагает создавать приложения с помощью технологий, лежащих в его основе, и как результат получить приложение, которое выглядит и работает примерно одинаково на обеих платформах (теоретически и на других тоже). Это сопровождается компромиссами, которые со временем минимизируются, но все еще существуют. Такие вещи, как производительность и прямой доступ к возможностям «железа», все еще сложно совместить с веб-технологиями.

Однако, помимо веб-технологий, в последние несколько лет мы наблюдали рождение и других кросс-платформенных инструментов, которые позволяют нам написать приложение один раз и работать с ним примерно одинаково в разных операционных системах. Популярные варианты – Corona SDK (в первую очередь для игр, но не обязательно), Xamarin, PhoneGap (просто веб-технологии, умно завернутые в собственный компонент WebView), Titanium и Sencha Touch (опять же, на основе веб-технологий, но с хорошим слоем абстракции над ним), может, еще несколько. Так что сейчас нам доступно множество различных вариантов, каждый со своими плюсами и минусами.

А теперь внимание! На арену выходит новый конкурент, жаждущий убить остальных и показать единственный верный путь написания кросс-платформенных мобильных приложений: Flutter.

Да, это немного глупое название… но знаете, мы можем закрыть на это глаза, потому что преимуществ у него выше крыши!

Что за (глупое) название?

Flutter – это продукт Google – ну, вы знаете, корпорации, которая основательно контролирует интернет, хорошо это или плохо (в случае с Flutter я думаю, что хорошо). Изначально этот фреймворк родился под именем Sky в 2015 году на саммите разработчиков Dart (не забудьте это слово, Dart, мы к нему скоро вернемся). Сначала он работал только на операционной системе Android от са-

мого Google, но вскоре был портирован и на iOS, так что сегодня он поддерживает две ведущие мобильные операционные системы.

Первые версии Flutter были выпущены сразу после его анонса, а кульминацией стал выпуск стабильной версии «Flutter 1.0» 4 декабря 2018 года. После этого Flutter был готов к прайм-тайму, и пришло время для разработчиков запрыгивать на борт! Популярность Flutter можно было бы охарактеризовать как метеорическую, и на это есть веские причины.

Одна из них заключается в следующем: первоначально заявленная цель Flutter или, по крайней мере, одна из основных, заключалась в отрисовке пользовательского интерфейса со скоростью 120 кадров в секунду. Google осознавал, что отзывчивый интерфейс приведет пользователей в восторг, поэтому эта функциональность и легла в основу Flutter. Это благородная цель, которой достигают лишь немногие кросс-платформенные фреймворки (даже нативные инструменты – и те часто испытывают трудности со скоростью отрисовки сложного интерфейса).

Flutter также предоставляет свои готовые компоненты пользовательского интерфейса – в отличие, например, от Xamarin и ReactNative, он не использует нативные контролы. Другими словами, когда вы хотите, чтобы Flutter отобразил кнопку, он рисует ее сам, а не просит об этом операционную систему, как делают другие фреймворки. Именно это и отличает Flutter от остальных и позволяет приложениям быть одинаковыми на разных платформах. Важно то, что новые компоненты пользовательского интерфейса, или *виджеты* (это слово тоже запомните, потому что, как и Dart, оно тоже скоро встретится), могут быть добавлены во Flutter быстро и легко, не беспокоясь о том, поддерживает ли их сама операционная система.

Это также позволяет Flutter предоставлять специфические наборы виджетов в стилистиках Material и Cupertino. Первый реализует Material Design от самой Google – стиль Android по умолчанию. Последний реализует стиль iOS от Apple.

Flutter можно разделить на четыре основные части, включая Dart. Я собираюсь оставить это до следующего раздела, так что давайте перейдем ко второму компоненту – основному движку Flutter. Этот движок в основном написан на C++ и использует библиотеку Skia, так что производительность отрисовки сравнима с нативной. Skia – это небольшая графическая библиотека с открытым исходным кодом, которая также написана на C++ и имеет очень высокую производительность на всех поддерживаемых платформах.

В качестве третьего основного компонента Flutter предоставляет унифицированный доступ к возможностям поддерживаемых операционных систем. Другими словами, код для запуска камеры на iOS и Android будет единым, а для этого можно использовать готовые методы Flutter.

Последний компонент – это виджеты, но, как и Dart, они тоже заслуживают собственного раздела, так что вернемся к ним позднее.

Если коротко, то Flutter состоит из этих четырех модулей, поэтому не так уж и много нужно знать, чтобы начать на нем разрабатывать. Тем не менее я ду-

маю, что немного углубленное изучение инструментов никогда не будет лишним. Надеюсь, вы согласны!

Теперь давайте более детально разберем Dart и виджеты, о которых говорили ранее.

Dart: язык богов?

Когда Google начал работать над Flutter, им предстояло ответить на главный вопрос: какой язык программирования выбрать? Может быть, язык веб-разработки, такой как JavaScript? Или же Java, язык Android? Или ради поддержки iOS выбрать Swift (в конце концов, Swift является языком с открытым исходным кодом)? Возможно, что-то вроде Go или Ruby было бы хорошим вариантом. Как насчет «старой школы», C/C++? А может, попробовать C # от Microsoft (у него тоже открытый исходный код)?

Я уверен, что было много вариантов, но в конце концов Google решил (не без причины!) использовать язык, который они создали несколько лет назад: Dart.

Вся следующая глава посвящена Dart, поэтому сейчас я воздержусь от деталей, но приведу небольшой пример:

```
import "dart:math" as math;

class Point {
   final num x, y;
   Point(this.x, this.y);
   Point.origin() : x = 0, y = 0;
   num distanceTo(Point other) {
      var dx = x - other.x;
      var dy = y - other.y;
      return math.sqrt(dx * dx + dy * dy);
   }

   Point operator +(Point other) => Point(x + other.x, y + other.y);
}

void main() {
   var p1 = Point(10, 10);
   var p2 = Point.origin();
   var distance = p1.distanceTo(p2);
   print(distance);
}
```

Не обязательно сразу детально понимать всё, что вы здесь видите. Тем не менее если вы работали раньше с любым C-подобным языком, например Java или JavaScript, то готов поспорить, что вы без проблем во всем разберетесь. В этом и заключается главное преимущество Dart: большинство современных разработчиков смогут написать и понять подобный код довольно легко.

ПРИМЕЧАНИЕ. Интересно, что мы называем языки C-подобными, но сам C – потомок гораздо более старого языка ALGOL. Я думаю, что ALGOL никогда не получит заслуженного уважения, так что этой ремаркой я выражаю всю любовь к нему!

Не вдаваясь во все мельчайшие подробности (для этого предназначена следующая глава), я думаю, что даже небольшой справки по Dart будет достаточно. Google создал Dart еще в 2011 году, и изначально он был представлен на конференции GOTO в Орхусе, Дания. Первый релиз 1.0 состоялся в ноябре 2013 года, примерно за два года до выпуска Flutter. За Dart стоит благодарить Ларса Бака (который разработал ещё и JavaScript-движок V8, используемый в Chrome и Node.js) и Каспера Лунда.

Dart – это лаконичный язык, который быстро набирает обороты, в основном благодаря Flutter. Поскольку он считается языком общего назначения, его широко используют для создания веб-приложений, серверного кода и приложений IoT (Internet of Things, «интернет вещей»). Пока я писал эту главу, вышел опрос о том, какие языки программирования вызывают наибольший интерес разработчиков в 2019 году, опубликованный JAXenter: `https://jaxenter.com/poll-results-dart-word-2019-154779.html`. В результате два языка заметно опередили остальные: Dart и Python, Dart вырвался вперед. Dart испытал наибольший рост в 2018 году. И хотя Flutter – почти наверняка один из самых популярных вариантов его использования, но, несмотря на это, Dart развивается во всех направлениях. Так что будьте уверены, Dart не обделен вниманием.

Так что же такое Dart? Предыдущий пример кода демонстрирует главные ключевые моменты, о которых я хотел сказать:

- Dart полностью объектно-ориентирован;
- инфраструктура языка включает в себя сборщик мусора, поэтому нет необходимости следить за памятью;
- его синтаксис основан на C, который подойдет большинству разработчиков (тем не менее, как и у любого другого языка с похожим синтаксисом, у него есть ряд особенностей, которые поначалу могут сбить вас с толку);
- он поддерживает общие языковые конструкции, такие как интерфейсы, наследование, абстрактные классы, шаблонные классы (generics, или «дженерики») и статическую типизацию;
- Dart включает проверку соответствия типов. Это позволяет использовать алгоритм для контроля правильности вашего кода;
- он поддерживает многопоточность, так что одновременно может исполняться несколько отдельных процессов, обеспечивая высокую производительность;
- Dart может компилироваться в нативный код для повышения производительности. Он не только компилируется в код для процессоров ARM и x86 в режиме Ahead Of Time (при сборке приложения), но и может транслиро-

ваться в JavaScript, а также поддерживает динамическую компиляцию во время исполнения (Just In Time);

- Dart позволяет использовать большой набор репозиториев с готовыми библиотеками, которые обеспечивают дополнительную функциональность для всего, что может понадобиться разработчикам;
- поддержка популярных сред разработки, включая Visual Studio Code и IntelliJ IDEA;
- ядро Dart поддерживает создание «слепков» (snapshots), которые позволяют упаковать весь исполняемый код (не только ваш код, но и библиотеки) в единый двоичный файл, что ускоряет запуск приложения.

Dart также зарегистрирован в качестве международного стандарта ECMA-408, а его актуальную спецификацию всегда можно получить на сайте www.dartlang.org/guides/language/spec.

Как я уже отмечал ранее, вся вторая глава будет посвящена Dart, а пока мы перейдем к следующей важной теме.

Виджеты окружают!

Давайте вернемся к разговору о главном госте нашего шоу, Flutter, и концепции, которая лежит в его основе, – виджетах.

Flutter – это и есть виджет. Когда я говорю, что *он и есть* виджет, я имею в виду... ну... я имею в виду, что *почти* все в нем является виджетом (гораздо сложнее найти во Flutter то, что виджетом *не является*!).

Что же такое виджет, спросите вы? Это части вашего пользовательского интерфейса (хотя и не все виджеты явно отображаются на экране). Виджет также представляет собой фрагмент кода, например:

```
Text("Hello!")
```

и это также виджет...

```
RaisedButton(
   onPress : function() {
      // Сделай что-нибудь.
   },
   child : Text("Click me!")
)
```

и это тоже виджет...

```
ListView.builder(
   itemCount : cars.length,
   itemBuilder : (inContext, inNum) {
      return new CarDescriptionCard(card[inNum]);
   }
)
```