МИРОВОЙ КОМПЬЮТЕРНЫЙ БЕСТСЕЛЛЕР

Head First Java

2nd Edition

Kathy Sierra,
Bert Bates

O'REILLY®

Beijing • Cambridge • Köln • Sebastopol • Taipei • Tokyo

Создатели серии Head First

Кэти Сьерра (Kathy Sierra) заинтересовалась теорией обучения, еще когда была разработчиком игровых приложений (для студий Virgin, MGM и Amblin). Основную концепцию оформления серии Head First она придумала, когда преподавала предмет «Разработки новых подходов в сфере распространения информации» в рамках курса «Развлекательные СМИ» в Калифорнийском университете. Позже Кэти стала главным тренером компании Sun Microsystems, где обучала других тренеров технике преподавания современных технологий, связанных с Java. В то же время она была ведущим разработчиком нескольких сертификационных экзаменов для Java-программистов. Вместе с Бертом Бейтсом активно использовала методику изложения материала, представленную в этой книге, чтобы обучить сотни преподавателей, разработчиков и обычных людей, которые не имеют отношения к программированию. Кроме того, Кэти — основатель одного из крупнейших в мире сайтов для Java-сообщества javaranch.com и блога headrush.typepad.com.

Кэти также соавтор изданий из серии Head First, посвященных сервлетам, технологии EJB и шаблонам проектирования.

В свободное от работы время Кэти занимается верховой ездой на своей исландской лошади, катается на лыжах и бегает, а также пытается преодолеть скорость света.

kathy@wickedlysmart.com

Берт Бейтс (Bert Bates) — разработчик программного обеспечения. Когда он трудился над проблемами искусственного интеллекта (около десяти лет), то заинтересовался теорией обучения и современными методиками преподавания. С тех пор он сам преподает программирование своим клиентам. Недавно Берт стал членом команды, разрабатывающей сертификационные экзамены для компании Sun.

Первые десять лет своей карьеры в качестве программиста Берт провел в разъездах, сотрудничал с такими вещательными компаниями, как Radio New Zealand, Weather Channel и Arts & Entertainment Network (A & E). Один из его самых любимых проектов заключался в построении полноценного симулятора сети железных дорог для компании Union Pacific Railroad.

Берт — заядлый игрок в го, и уже долгое время работает над соответствующей программой. Он прекрасный гитарист и сейчас пробует свои силы в игре на банджо. Любит кататься на лыжах, занимается бегом и тренирует свою исландскую лошадь Энди (хотя еще неизвестно, кто кого тренирует).

Берт пишет книги в соавторстве с Кэти. Они уже корпят над новой серией изданий (следите за обновлениями в блоге).

Иногда его можно застать на сервере IGC для игры в го (под псевдонимом *jackStraw*).

terrapin@wickedlysmart.com

Кэти и Берт пытаются не оставлять ни одного письма без ответа, но, учитывая количество посланий и напряженный график командировок, сделать это довольно сложно. Самый лучший (и быстрый) способ получить техническую помощь по этой книге — обратиться на очень оживленный форум для новичков в Java: javaranch.com.

Содержание (краткое)

Содержание (подробное)

Введение

Ваш мозг по отношению к Java. Когда *вы* стараетесь что-то изучить, ваш *мозг* пытается оказать вам услугу, убеждая, что все это не имеет никакого значения. Он думает: «Лучше сосредоточиться на том, как избежать встречи со свирепым хищником или что обнаженный сноубордист — это плохая идея». Как же *убедить* мозг в том, что от знания Java зависит ваша жизнь?

1 Погружаемся

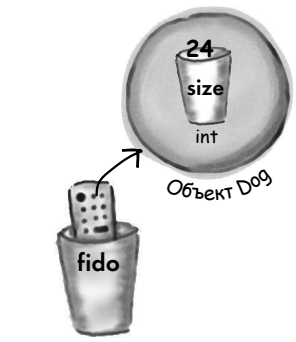
Java открывает новые возможности. Еще во времена первой (и довольно скромной) открытой версии 1.02 этот язык покорил разработчиков своим дружественным синтаксисом, объектно ориентированной направленностью, управлением памятью и, что важнее всего, перспективой переносимости на разные платформы. Соблазн написать код один раз и запускать его везде оказался слишком велик. Однако, по мере того как программистам приходилось бороться с ошибками, ограничениями и чрезвычайной медлительностью Java, первоначальный интерес к языку остывал. Но это было давно. Сегодня Java достаточно мощный и работает быстрее.

2 Путешествие в Объектвилль

Мне говорили, что там будут объекты. В главе 1 вы размещали весь свой код в методе main(), но это не совсем объектно ориентированный подход. По сути, он не имеет ничего общего с объектами. Да, вы использовали некоторые объекты, например строковые массивы для генератора фраз, но не создавали свои собственные типы объектов. Теперь вы оставите позади мир процедур, выберетесь из тесного метода main() и сами начнете разрабатывать объекты. Вы узнаете, чем же так удобно объектно ориентированное программирование (ООП) на языке Java, и почувствуете разницу между классом и объектом.

3 Свои переменные нужно знать в лицо

Переменные делятся на два вида: примитивы (простые типы) и ссылки. В жизни должно быть что-то помимо чисел, строк и массивов. Как быть с объектом PetOwner с переменной экземпляра типа Dog? Или Cat с переменной экземпляра Engine? В этой главе мы приоткроем завесу тайны над типами в языке Java и вы узнаете, что именно можно *объявлять* в качестве переменных, какие значения *присваивать* им и как вообще с ними *работать*.



Ссылка типа Dog

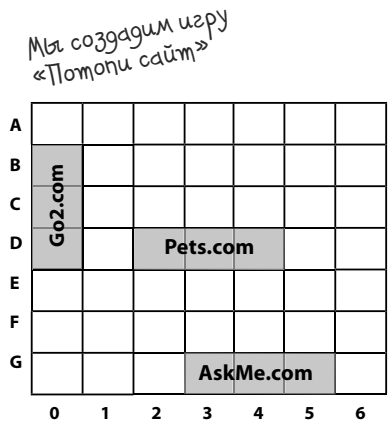
---

4 Как себя ведут объекты

Состояние влияет на поведение, а поведение — на состояние. Вы знаете, что объекты характеризуются **состоянием** и **поведением**, которые представлены **переменными экземпляра** и **методами**. До этого момента вопрос о связи между состоянием и поведением не поднимался. Вам уже известно, что каждый экземпляр класса (каждый объект определенного типа) может иметь уникальные значения для своих переменных экземпляра. Суть объектов заключается в том, что их поведение зависит от состояния. Иными словами, ***методы используют значения переменных экземпляра***. В этой главе вы узнаете, как изменить состояние объекта.

Особо мощные методы

Сделаем методы еще более мощными. Вы уже поработали с переменными, несколькими объектами и написали небольшой код. Но для полноценной работы требуется гораздо больше инструментов, например **операторы** и **циклы**. Почему бы вам во время учебы не *создать* что-нибудь настоящее, чтобы собственными глазами увидеть, как с нуля пишутся (и тестируются) программы. **Может быть, это будет игра** вроде «Морского боя».

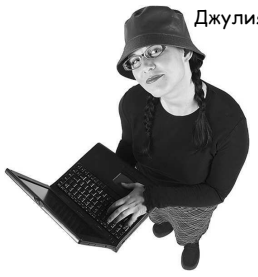


Использование библиотеки Java

Вместе с Java поставляются сотни готовых классов. Можете не тратить время на изобретение собственного велосипеда, если знаете, как отыскать нужное в библиотеке Java, называемой еще **Java API**. *Думаем, у вас найдутся дела поважнее.* При написании кода сосредоточьтесь на той части, которая уникальна для вашего приложения. Стандартная библиотека Java представляет собой гигантский набор классов, готовых к применению в качестве строительных блоков.

Хорошо, когда известно, что ArrayList находится в пакете java.util. Но как бы я смогла додуматься до этого самостоятельно?

Джулия, 31 год, модель

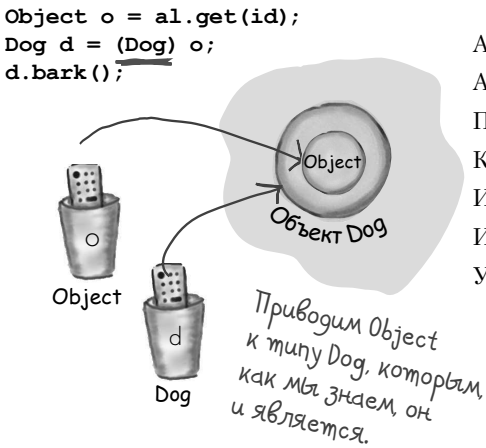


7 Прекрасная жизнь в Объективилле

Планируйте свои программы с прицелом на будущее. Как вы оцените способ создания Java-приложений, при котором у вас останется много свободного времени? Заинтересует ли вас создание гибкого кода, которому не страшны досадные изменения в техническом задании, возникающие в последний момент? Поверьте, вы можете получить все это лишь за три простых подхода по 60 минут каждый. Изучив принципы полиморфизма, вы узнаете о пяти шагах грамотного проектирования классов, трех приемах для достижения полиморфизма и восьми способах создания гибкого кода.

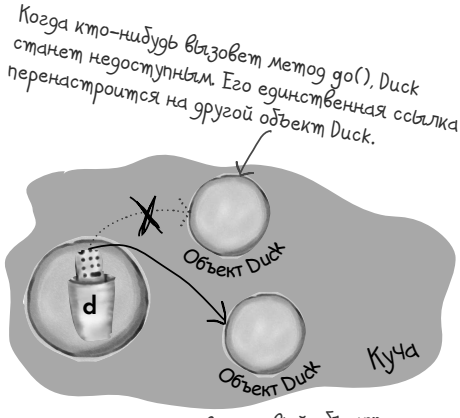
8 Серьезный полиморфизм

Наследование — это только начало. Чтобы задействовать полиморфизм, нужны интерфейсы. Новый уровень гибкости и масштабируемости может обеспечить только архитектура, основанная на интерфейсах. Мы уверены, что вы захотите их использовать. Вы удивитесь, как могли жить без них раньше. Что такое интерфейс? Это на 100 % абстрактный класс. Что такое абстрактный класс? Это класс, для которого нельзя создать экземпляр. Зачем это нужно? Читайте главу...



9 Жизнь и смерть объектов

Объекты рождаются и умирают. Вы управляете их жизненным циклом. Вы решаете, когда и как *создавать* их. И вы решаете, когда их *уничтожать*. Но на самом деле вы не уничтожаете их, а просто делаете брошенными и недоступными. Уже после этого безжалостный **сборщик мусора** может аннулировать объекты, освобождая используемую память. В этой главе мы рассмотрим, как создаются объекты, где размещаются, как эффективно использовать их и делать недоступными.



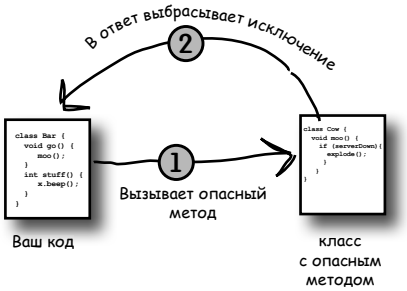
10 Числа имеют значение

Поговорим о математике. В Java API есть множество удобных и простых в использовании методов для работы с числами. Поскольку большинство из них статические, то сначала разберемся, какими особенностями обладают статические переменные и методы, включая константы, которые в Java считаются *статическими* финализированными переменными.

Статические переменные универсальны для всех экземпляров класса.

11 Опасное поведение

Случается всякое. То файл пропадает, то сервер падает. Не важно, насколько хорошо вы программируете, ведь невозможно контролировать *все*. Что-то может пойти не так. При создании опасного метода вам понадобится код, который будет обрабатывать возможные нестандартные ситуации. Но как узнать, опасен ли метод? И куда *поместить* код для обработки ***непредвиденной*** ситуации? В *этой* главе мы разработаем музыкальный MIDI-проигрыватель, использующий опасный JavaSound API.



12 Очень графическая история

Соберитесь, вам предстоит придумать GUI (graphical user interface — графический пользовательский интерфейс). Даже если вы уверены, что всю оставшуюся жизнь будете писать код для серверных программ, где работающий на клиентской стороне пользовательский интерфейс представляет собой веб-страницу, рано или поздно вам придется создавать инструменты и вы захотите применить графический интерфейс. Работе над GUI посвящены две главы, из которых вы узнаете ключевые особенности языка Java, в том числе такие, как **обработка событий** и **внутренние классы**. В этой главе мы расскажем, как поместить на экран кнопку и заставить ее реагировать на нажатие. Кроме того, вы научитесь рисовать на экране, добавлять изображение в формате JPEG и даже создавать анимацию.

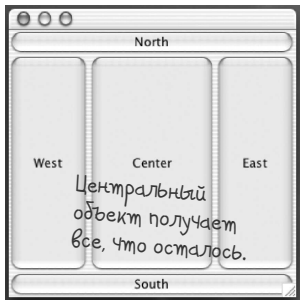
```
class MyOuter  {

   class MyInner {
      void go() {
       }
    }

}
```

13 Улучшай свои навыки

Swing — это просто. Код для работы со Swing выглядит просто, но, скомпилировав его, запустив и посмотрев на экран, вы подумаете: «Эй, этот объект должен быть в другом месте». Инструмент, который *упрощает* создание *кода*, одновременно усложняет управление им — это **диспетчер компоновки**. Но, приложив небольшие усилия, вы можете подчинить диспетчер компоновки. В этой главе мы будем работать со Swing и ближе познакомимся с виджетами.

Компоненты в областях east и west вытягиваются по ширине.

Компоненты в областях north и south вытягиваются в длину.



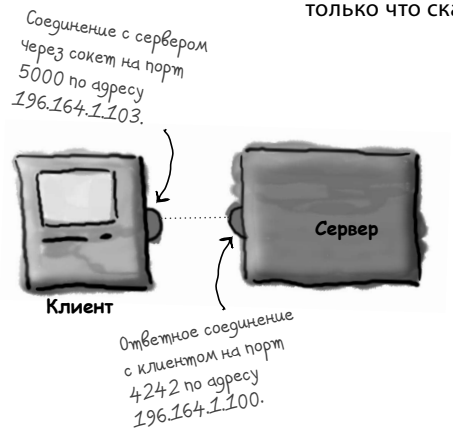
14 Сохранение объектов

Объекты могут быть сплющенными и восстановленными. Они характеризуются состоянием и поведением. Поведение содержится в классе, а *состояние* определяется каждым *объектом* в отдельности. Что же происходит при сохранении состояния объекта? Если вы создаете игру, вам понадобится функция сохранения/восстановления игрового процесса. Если вы пишете приложение, которое рисует графики, вам также необходима функция сохранения/восстановления. **Вы можете сделать это легким объектно ориентированным способом** — нужно просто сублимировать/сплющить/сохранить/опустошить сам объект, а затем реконструировать/надуть/восстановить/наполнить его, чтобы получить снова.

Есть вопросы?

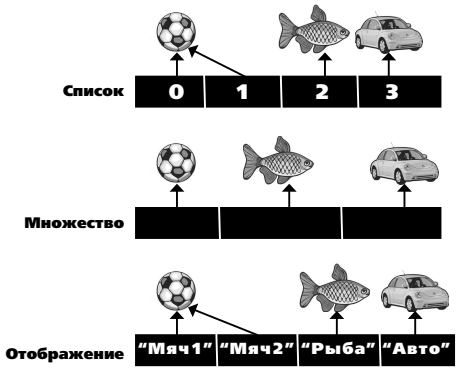
15 Устанавливаем соединение

Свяжитесь с внешним миром. Это просто. Обо всех низкоуровневых сетевых компонентах заботятся классы из библиотеки java.net. В Java отправка и получение данных по Сети — это обычный ввод/вывод, разве что со слегка измененным соединительным потоком в конце цепочки. Получив класс BufferedReader, вы можете считывать данные. Ему все равно, откуда они приходят — из файла или по сетевому кабелю. В этой главе вы установите связь с внешним миром с помощью сокетов. Вы создадите клиентские и серверные сокеты. В итоге у вас будут клиенты и серверы. И вы сделаете так, чтобы они смогли общаться между собой. К концу этой главы вы получите полноценный многопоточный клиент для чатов. Ой, мы только что сказали «*многопоточный*»?



16 Структуры данных

Сортировка в Java — проще простого. У вас уже есть все необходимые инструменты для сбора данных и управления ими, поэтому не нужно писать собственные алгоритмы для сортировки. Фреймворк для работы с коллекциями в Java (Java collections framework) содержит структуры данных, которые должны подойти практически для всего, что вам может понадобиться. Хотите получить список, в который можно добавлять элементы? Вам необходимо найти что-нибудь по имени? Хотите создать список, который автоматически убирает все дубликаты? Желаете отсортировать своих сослуживцев по количеству ударов, которые они нанесли вам в спину, или домашних любимцев по количеству трюков, которые они выучили? Читайте главу…



17 Выпусти свой код

Пришло время его отпустить. Вы написали код. Вы его протестировали и откорректировали. Вы рассказали всем знакомым, что больше не желаете видеть ни единой его строки. По большому счету вы создали произведение искусства. Это то, что действительно работает! Но что дальше? В последних двух главах мы расскажем, как организовывать, упаковывать и развертывать (внедрять или доставлять) код на языке Java. Мы рассмотрим локальный, полулокальный и удаленный варианты развертывания, включая исполняемые Java-архивы (JAR), Java Web Start, RMI и сервлеты. Большую часть этой главы мы посвятим организации и упаковыванию вашего кода — это то, что нужно знать вне зависимости от выбранного варианта доставки.