教育是用生命影响生命的过程

Broadview
www.broadview.com.cn

真正的零起步，
两个小时教会你
写程序！

Java 就该这样学

我用了**8年时间**教授Java这门课，尝试着让初中毕业生、被传统教育淘汰的人，或是退休在家的老人掌握编程技术，**大约15000名**各种基础和理解能力的学生轻松地掌握了这项技术。令人欣慰的是，所有的学[redacted]终都**将这个学习过程当成一场游戏**，[redacted]程这个工作发展成自己的兴趣。在本书中，将阐述我与众不同的做法，希望学习者能够在一开始便建立起新的学习思想，**这样我们才能一起玩代码**。

王洋 编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

金牌讲师为您开启编程的快乐之旅

Java
就该这样学

王洋 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书基于建构主义教育思想，通过大量循序渐进的案例，让学生在体验中掌握 Java 语句，同时获得编程能力、排错能力和学习能力。本书多次使用陷阱式教学法，帮助学生深刻理解所学知识，理解面向对象编程思想。本书详细地介绍了 Java 程序设计的开发环境、概念和方法。内容分为四个阶段：小案例阶段、小案例推动大项目阶段、重点建立复杂编程逻辑阶段和综合大项目阶段，用于巩固面向对象编程思想，并且弥补即时聊天项目在数据库应用上的不足。

本书的内容和组织形式立足于高校教学教材的要求，适用于从职业院校到重点本科院校的课程和学生群体，可以作为 Java 语言入门教材，或者面向就业的实习实训教材，同时也可以作为计算机技术的培训教材，读者完全可以通过本书自学 Java 技术。

图书在版编目（CIP）数据

Java 就该这样学 / 王洋编著. —北京：电子工业出版社，2013.6
ISBN 978-7-121-20222-3

Ⅰ. ①J… Ⅱ. ①王… Ⅲ. ①JAVA 语言－程序设计 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2013）第 081247 号

策划编辑：孙学瑛
责任编辑：葛 娜
印 刷：三河市鑫金马印装有限公司
装 订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本：787×1092 1/16 印张：18.5 字数：472 千字
印 次：2013 年 6 月第 1 次印刷
印 数：4000 册 定价：49.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

关于学习

一直以来人们都认为教师和书籍是知识的载体，教学过程就是将知识传递给学生，于是书上写满了正确的知识，学生看书就可以迅速掌握知识，理论上这是高效率的系统。事实上只有极少数人能够适应这样的系统，因为这些知识也是有人经过一个过程得到的，忽略了发现知识的过程，而直接将结果传递给学生，似乎高效率，但是学生却常常无所适从，因为学习是发现知识的过程，而不是记住知识的过程。

好在建构主义教育思想指明了更加适合学生的教学过程。在这一思想下，书和教师从正确知识的传递者转变成探索知识的引领者，带领着学生去体验、去感觉、去发现属于学生自己的知识。优秀学生通过自己的努力在到达学习目标的道路上，不断地调整，将错误的理解剔除。问题是大多数学生无法完成这个过程，要么陷入错误的包围中，最终放弃了探索；要么通过死记硬背来达到学习目的，结果是学生能够通过考核，却没有运用知识的能力。这些学生的学习目标离正确的轨道越来越远，甚至很多人迷失了学习的真正目标。将记住知识作为唯一的目标，是很多教育者正苦苦探索的正确的教育途径，在教育理论研究中早已经被发现，那就是建构主义教育。

建构主义教育思想从来不认为掌握知识是学习的最终目的。我认为学习的目标是认知、能力和精神。认知和知识是不同的，知识停留在人的头脑中，而认知是能够被熟练使用的知识；能力在不同的领域是不同的。在本书的范畴内，一个优秀的 Java 程序员，需要有编程的能力、排除错误的能力、探索新技术的能力。如果学习的目的仅仅是为了掌握一项技术，那么人终将会被新的技术手段所替代。任何学习过程都需要是生命价值的提升。一个程序员需要有严谨的态度、专注的品质、探索的精神和创新的意识。这些学习目标不是一节课或一个章节的任务，而是需要通过整个教学过程来建构。

学生的学习动机始终是教育理论界热衷讨论的话题。我认为学生学习的动机有三个方面：一是为了获得喜悦，二是为了消除恐惧，三是自我效能。好的成绩可以获得家长、老师的表扬，可

以有更好的名次甚至奖学金；而差的成绩会被批评、留级，甚至拿不到毕业证书。我们发现普遍的教学手段是为了推动学习动机的前两个方面，这造成了两个可能的结果：一是有些学生对于奖励或者惩罚麻木了，一旦丧失了学习动机，自然好的成绩就无从谈起；二是在另外一些学生身上，这些手段一直能够起作用。我们会得到所谓的好学生，问题是，这些动机是外界推动的，而非内生的，这些习惯于此的好学生或许一生都在意别人的评价。如果教学过程能够激发学生的自我效能，让学生的学习是基于自己强烈的爱好和成功的喜悦，我们就一定能够培养出优秀的学生，而他们也将一生受益。

问题是，为什么建构主义教育思想如此的好，却很少在教学实践中应用？这是因为建构主义和现有的教学形式相比仍有些弱点。

第一，从理论上讲，建构主义教育的效率比较低。现在我们能够在短时间内将大量的正确知识传递给学生，学生只需要理解记忆就好了，而建构主义教育要呈现知识探索的过程，这样会消耗更多的时间和精力。

第二，建构主义教育的效果不可控。学生是通过体验自己发现整合知识，那么不同的学生或许得到的结论不同、深度不同。

第三，考核困难。我们不能再用知识点来考核学生，因为教学过程中就没有传递经典的知识点。

第四，实施建构主义教育对于教师的要求比较高。它的教学过程的设计建立在对学生深入理解的基础上，教师不仅仅要准备知识。

为了实现上述效果，教师将扮演不同以往的角色，教师不再是知识的载体，教师将陪伴着学生一同探索，带领着学生犯错误，引领着学生思考整合。为了克服建构主义教育思想的弱点，在写本书时，我基于对学生和技术的理解，剔除了大量知识点的讲解，在反复的教学实践中，已经能够获得和传统教学相同的教学效率。另外，我大量总结和研究了学生的学习过程，建立了学生在学习 Java 过程中的学习曲线，依照学习曲线来评估和考核学生的学习效果。

关于本书

本书总结了我多年来在这条道路上的探索，力求提供基于建构主义教育思想的 Java 教学材料，帮助学生轻松地掌握作为 Java 程序员所需要的知识和能力。书中的内容并不是简单的案例堆砌，每个部分的任务都包含了对相关知识的整合，都基于学生的学习曲线特点。

我在 8 年教学探索后才动手写这本书，是因为我一直相信“教育是用生命影响生命的过程”。我无法在一本书中实现和我亲自上课同样的影响过程，课堂上一遍遍地重复代码所传递的严谨态度，无法在书中呈现，加上我对技术、对学生的理解，以及对建构主义教育思想理解的局限，让我清楚地知道，我并没有完成一部让我心满意足的作品，书中不可避免地有很多不足，恳请读者批评指正。

这本书的内容是我所教的数以万计的学生的成果，甚至有很多案例是我的学生在学习过程中发明的。从 8 年前开始的这段探索并不是一挥而就的，我诚挚地感谢我的学生，是他们的忍受、包容和努力帮助我完成了这本书。我要感谢我的家人，我儿子的出生和成长让我开始接触和研究教育理论，给我之前漫无目的的探索指明了方向。为了让这本书通俗易懂，我那学文科的爱人像一名真正的学生一样，通过本书来学习 Java 技术。在她的努力下，这本书具备了更广的覆盖范围，确保读者即使没有任何专业基础，也能够通过本书掌握 Java 技术。同时也要感谢电子工业出版社的老师为本书的出版所付出的辛勤工作。

代码下载

我一直希望这本书的定价尽可能低，希望有更多的人能够没有负担地学习 Java 技术，所以本书不用光盘提供代码，而是将代码放在网上供下载。需要强调的是，请不要直接编译运行，或者复制我所提供的代码，供下载的代码是我的，只有你亲手输入到电脑里的内容才属于你。

代码下载地址：http://www.broadview.com.cn/20222

王　洋

于 2012 年 3 月 14 日

学习之前

在翻开这本书的这一刻，你一定对编程是有兴趣的。每年成千上万的人开始学习编程，但是大多数人都没有坚持下来。失败者都在找借口说这门技术太难了，不适合没有基础的人，和最初的想象完全不同，学习编程太枯燥了。坚持下来的人通常有不同凡响的毅力，但很多人也没有真正驾驭这些技术，大概 5 年以后，很多人说：“哦，原来是这么回事”。我用了 8 年时间教授 Java 这门课，尝试着让初中毕业生、被传统教育淘汰的人，或是退休在家的老人掌握编程技术，大约 15000 名各种基础和理解能力的学生轻松地掌握了这项技术。令人欣慰的是，所有的学生从始至终都将这个学习过程当成一场游戏，并将编程这个工作发展成自己的兴趣。下面我将阐述我的做法与众不同的地方，希望学习者能够在一开始便建立起新的学习思想，这样我们才能一起玩代码。

教会你的不是这本书或者我

我上课的第一个要求，就是请所有的学生将学校发的书从窗户扔出去，看一百遍这些愚蠢的书，你也不可能学会这项技术。可是今天我不得不通过书来低成本地帮助大家学习，但是请记住第一个原则：这本书也教不会你，真正能让你学习并一直坚持学下去的是最开始的兴趣，是整个学习过程持续不断的热情。如果这本书的内容能够一直帮助你维护开始的兴趣，你自己就能学会所有的一切。所以请放松下来，这本书不是要教给你什么，而是引领着你一起玩代码。

本书的内容尽力让你的学习是有趣的、有成就感的，但是帮助你的真正力量是你自己的主动投入，属于你的内容才是你的动力所在。我能够提供这些有趣的内容，不是我多么有创造力，所有的案例都是我教过的学生提供的。我相信源于学习者的内容最贴近学习者，他们能够提供这么多充满想象力的案例，是因为我一直鼓励大家自由地创造。在这里我也这样鼓励你，有人跟我学习就是为了能够做一个连连看的游戏，这是他的梦想；有人 6 年多沉迷于网络游戏，当意识到他也能的情况下，他全身心地投入要做一个自己的网络游戏。雷电、坦克大战、环保网站数不胜数，是你的想法，帮助你学会所有的一切，即便想法相当宏大，也会在这本书的推进过程中，解决一个又一个难题，将这本书扔掉的时候，你已经完成了开始梦想的软件。

开始编程序不需要理论基础

书只是传统学习模式罪恶的焦点，真正的问题是大多数人认为学习就意味着要掌握知识，而书里承载着知识，看懂书便吸收了知识，就学会了知识。这个结论根本的错误是，编程序需要的不是知识，而是驾驭代码的能力，是学习新技术的能力，是解决困难的能力，甚至是面对这项工作的精神，如果这些能力和精神没有问题，那么知识就一定没有问题。

中国整个的教育体系建立在行为主义的教育思想上，没有人关心学习者的感受。老师的心思在他所教授的知识体系上，所以课堂上老师滔滔不绝地讲着，而学生在以各种形式昏昏欲睡。没有几个学生提出异议，因为大家都麻木了，想当然地认为学习只能是这个样子，一定要先经历一大堆枯燥的理论学习，才有能力进行实践。其实每次受益终生的学习都不是这个样子的，这种教育模式唯一的效果是熄灭学生开始的学习热情，最终学生记下了大量的知识，却没有编程能力，其实也没有多少人在这样的学习过程中真正记住什么。

我观察到 1 岁的孩子，通过自己的努力艰难地学会站起来走路，这是孩子非常了不起的成就。这个过程别人的帮助十分有限，甚至使用学步车或学步带的帮助都是有害的。是什么支持着孩子掌握如此复杂的技能？是强烈的意愿。我想到如果直立行走成为今天大学的一门课程，会是怎样的一个景象——这个内容会成为一个学期的课程。第一节课想必是直立行走对人类的重要意义，还有直立行走的演变过程，然后一点点地展开，告诉学生直立行走需要动用多少块骨骼、多少块肌肉，小脑所起的作用，它们的名字都是什么，这些内容足够支撑一份期中考试的卷子，下学期会更加深入地讨论，是哪些肌肉收缩带动着哪些骨骼和哪些关节运动。不知道这么多内容一个学期是否能讲完，结果呢，学生会通过这样的过程学会走路吗？这就是我们现在所遭受的教育。这个例子看起来有些调侃，但是我们今天的学习就是这样的状况，所以大家抱怨在学校学的都是理论，没有实践能力，虽然很多人都在维护这个体系，但是我认为这根本就迷失了方向，我们只想学会走路，只想有编程的能力。

如果我告诉你 Java 有 8 种基本的数据类型，然后将它们都写在黑板上，一个个地讲解它们的定义、特点和范围，我相信你一定昏昏欲睡。当我讲到第 8 种数据类型的时候，你一定忘掉了前 6 种，这根本和你的专注力或者学习能力无关，这样的课程内容和你的需求无关，你凭什么能注意力集中地听课。我们所期望的学习过程不是这样的，在上课和玩游戏之间，越来越多的人选择玩游戏。

到底游戏和上课之间的区别是什么？我总结了三点不同。其一，游戏一直有成就感激励着你，无论是打死了鬼怪让经验值提升，还是一关关的挑战让人有成就感，甚至成为高手是可以炫耀的。我们经历的上课会有成就感吗？有人学会一个什么东西会跑去找人炫耀吗？其二，游戏需要玩家一直主动地参与，几乎所有的游戏都需要不停地操作，所以你停不下来，你会克服困倦，可以整夜地玩。而上课呢？学生在绝大多数时间里都是被动地听，即便走神了，即便睡着了，也不会造成失败。其三，游戏被精巧地设计，整个玩的过程既不容易也不难，太容易的游戏会让玩家睡着，而太难的游戏会让人放弃。可是大多数课堂要么太容易，一直在讲每个人都知道或是无

聊的事情，要么太难，所以学生要么睡着了，要么放弃了。

这本书的目标就是这三点，给学习者一直带来成就感，即便没有基础，你也只需要 2 个小时便能够写出可以炫耀的王八，每一步的内容都是基于你的成就感设计的；你一直需要动手操作，没有人告诉你真正的答案，完全要你自己发现；内容被设计得既不容易也不难。但是如果上我的课，我有大量的手段来确保这三点，而对于书这个有限的载体需要你按照我的原则和指示来干，所以不能像大多数书一样一口气看下去，我会提示你在哪里停下来完成你的任务，有的时候欲速则不达。正确的学习体验是，我提示你，你尝试，经过失败、反思和努力，你找到解决的办法，并体会其中的感觉，直到你会了，或许你没法将你会的东西告诉别人，但是你会的已经是你自己的能力。

大家都有共同的学习曲线

上面的讨论揭示了一个问题：好的老师不是精通知识的人，而是能够理解学生的人。需要理解的是什么？是我这里提到的学习曲线。什么是学习曲线？就是一个人从接触一个新学科开始，每个阶段学习的特点和关注点。我们注意到一个普遍现象：老师明明讲了一个内容，所有的学生像是完全没有听到一样。老师气愤，觉得学生没有注意听讲；学生无辜地看着老师，却不敢辩解，确实没听到。问题就在于在那一刻学生的学习曲线恰好不在老师讲授的点上，因为学生的注意力不在此，所以老师在做无用功。

比如一个编程的初学者，第一个星期，他的关注点是单词，告诉他再多的东西他都听不到，因为单词总也拼不对；第二个星期，学生的学习曲线到了关注语句了；第一个月，学生关注三、五行语句之间的逻辑关系；第二个月，学生开始关注复杂的逻辑，以及一个函数里的三、五十行代码；从第四个月开始，学生有能力注意到不同代码段之间的关系；第五个月，学生开始有能力理解面向对象。那些一上来就讲面向对象的书，完全是在浪费纸张。

你会发现学习编程的学习曲线和代码规模有关系。当然学习曲线没有一个对每个人都适用的准确标准，不过每个学生的学习过程是相同的，不同之处在于经历每个阶段的时间。一些因素影响着个体的学习曲线，主动练习的量、基础、逻辑能力等因素都影响着时间进程。

不要尝试从别人那里获得答案

有问题问老师几乎是好学生的标志，可是我的课堂上是禁止提问的。为什么要提问题？因为想得到答案。从掌握知识的角度看，向老师提问是最高效率的做法，但是我前面已经讨论了，我们的目的不是要学知识，而是获得能力，其中最重要的一个能力是学习能力。获得答案的办法有很多，尤其是编程这门专业，因为这个专业所需的学习成本最低，一台电脑、适当的软件足够了。有了这个条件，你就可以任意地尝试。在我的教学过程中，我发现绝大多数的问题，自己用电脑试一下就知道答案了。还有一些问题是为了理解所学的知识，原则是如果你不理解，就是这

个阶段你无法理解，写的代码多了，感受到了，自然而然就理解了。那个时候的理解，和现在有人告诉你完全不同，自己顿悟的结果是真正属于你的，而不是别人灌输给你的。

问老师问题揭示了一个现象：长期以来老师都是学生的依赖，是知识权威，所以听到正确答案似乎理所应当。这个现象不仅仅表现在计算机的相关专业上，只是在编程相关专业上的危害太大了。这个行业需要想象力和创新精神，所有的权威都妨碍了学生在这个方面的发展。没有权威，你也只能依赖自己的思考、尝试和体验。

笔记是没有意义的学习工具

我的学生不允许记笔记，我同样要求学生抛弃这个经典的学习工具。回想一下从小学到现在，你在课堂上记了多少本笔记，那么又有多少笔记你回头看过，很多人只在考试前突击看一下。可是你为什么要记笔记？理由不外乎要记住重点，或是一时消化不了的内容。不管理由是什么，不回头看就将这些丢给了一个本子，它们并没有进入你的头脑。想象一下，如果没有笔记怎么办？你只能尽力地记住更多的内容。所以丢掉这个拐棍，能够帮你掌握得更多。

问题是，还是有两个理由支持笔记这样的工具。有人说可以通过整理笔记来总结整理所学的知识。好了，再也不提知识了，如果需要总结整理，就写代码吧，因为这是最终的目的。还有一个理由就是必然会有一时掌握不了的内容，记下了回头消化。暂且不论是否能回头看，即便是你有这样的学习能力，你也根本不需要回头看。还是那个原则：如果你现在理解不了，那么就是理解不了，就丢掉好了，理解不了的东西不属于你，属于你的东西就把它变成你身体的一部分。

我不赞成记笔记，但是我希望学生能够每天写学习日志，这和记笔记有着本质的区别。反对记笔记，因为这是完全从知识角度出发的工作，而且往往还是被动的，大多数人的笔记是某种形式的复述；学生日志的内容要比笔记广阔多了，可以是所学知识，还可以是学习心得、喜悦与困惑，甚至描述生活感悟或对未来的憧憬，写什么随你，即便是记录所学到的知识，也是你所掌握的知识，就是一段学习生活的记录，它将激励你、鼓舞你、鞭策你，如果能够因此养成写日志的习惯，这将成为一生的财富。

20 遍地敲代码

我的很多学生开始觉得跟着我学习，真是太幸福的一件事情了，不需要书，不需要记笔记，不需要背任何东西。但是我也有让人痛苦的要求，所有经历的代码都要敲 20 遍。事实上大多数学生没有做到 20 遍，但是所有做到 20 遍的人对编程的理解远远超过其他人。我甚至得出一个结论：在我教过的学生中，对编程的理解程度和他敲过的代码遍数成正比。

为什么要敲 20 遍代码呢？我理解了不就行了吗？我们需要的不是理解，而是能力，作为一个程序员需要建立手上的感觉。在软件企业里，最好的软件工程师不仅仅会技术，而且可以不思

考、靠下意识就能完成大多数编码。这样的人才有精力去思考用户需求，设计软件的框架。更何况在学习的过程中，我们认为的理解不见得真的理解了，就好像来到一个陌生的环境住下，我带着你走了一遍新住所门前的马路，那么你当然知道这条马路的存在了，但是你对这条马路是没有感觉的，当你走了 10 遍后，你开始忽略这条马路，注意点放在了左边有个小卖部，右边有棵歪脖树，前面有个岔路上，你开始能够感觉到这条马路的细节；如果走了 20 遍会怎么样？你甚至能够在伸手不见五指的夜晚摸着回来。代码也一样，不在乎学得有多快，而是每学一点，就建立感觉，让这些内容成为你身体的一部分，坚持这个原则，没有人会学习失败。

20 遍还有一个附加的好处，一些一时理解不了的内容，你会发现大量的重复练习后，问题自然而然地解决了，这是熟读唐诗三百首的道理。很多人担心这样做的效率，会不会学得很慢。我的经验：不会慢，而是快得惊人，因为这样学习一点扎扎实实地掌握一点，向前推进过程中不需要返工之前所学。

所有的内容都有内在的关联，是一个整体

这本书包含了大量的内容，如 Java、数据库编程、数据结构和算法。我没有将这些内容分开，我此前的教学实践也从不划分科目，它们本来就是一个整体。我甚至不用不同的学科来设计这本书的结构，这本书唯一要遵循的是学习者的学习曲线。遇到数据库，我们就讨论数据库，所以或许你能看到数据库的内容散落在书的不同地方。划分科目本来就是一些无聊专家的游戏，与我们没有任何关系。

因为这样的设计思路，在跟着这本书学习完以后，你或许要去看其他的书籍，那些依照知识体系写成的书，可能会描述一些我没有提及的知识。也就是说，这是一本带领你学习的书，而不是一个知识的词典，所以我建议你在看这本书的时候，暂时不要看其他的书，那或许会破坏我精心设计的学习节奏。

关于高端技术

Java 语言如此强大，是因为这个技术属于整个产业界。由于有大量的企业和开发者的支持，在 Java 语言的基础上已经产生了大量的扩展，对于 Java 的发展还在继续着。这些发展有来自于官方的，也有第三方的扩展，同样的问题会找到侧重点不同的解决方案，经过市场的筛选，有些技术就会成为流行技术。招聘企业可能会问求职者是否掌握了一些流行技术，以便判断这位求职者发展到了什么程度，于是便有人想到走捷径，直接学习流行技术，甚至直接背面试题来求职，很多学校也急功近利地推动着这个现象，这是对于学生、软件企业和学校都无法持续发展的策略。

因为所谓的流行技术都是快速发展变化的，今天大家会说 SSH、AJAX 甚至 Android，明天大家会用什么没人知道。追逐这些技术的基础是你有强大的学习能力，而不是一直能够找到带领

你的老师。在编程技术中，学习能力意味着扎实的基础、逻辑能力、清晰的概念和探索精神，对于 Java 来说，你需要在 Java SE 的学习阶段就获得这些能力，所以我一直重视基础技术的学习。基础扎实的学生学习 Struts、Spring 或 Hibernate 只需一个半天就能够摸到一个技术的门路，然后在实践中能够迅速完善相关的知识；而很多人学习一个星期都稀里糊涂。所以我建议你用 60% 的时间去彻底掌握基础知识，而不是用主要的时间去学习流行技术。

我一直在使用流行技术，而不是高端技术。我认为在技术领域并没有什么高端或低端的区分。很多人会误认为，掌握所谓的高端技术的人会比使用相对基础技术的人更厉害。实际上在我教过的学生中，目前发展最好的学生主要使用的语言是 JavaScript。确实有些技术是另外技术的基础，但是打好基础可能会更强大。

考虑到即便限定在 Java 的范畴内，要学的技术也非常多，我将技术分成了三种类型：第一种是必须熟练掌握的，再过 100 年都不会变化的；第二种是通常会用得上，你需要知道怎么做；第三种是知道这个能做什么就好了，你根本不需要有能力去做，在未来需要的时候，你至少知道看那个方面的书，到网络上搜索就可以。

基于上面的分类，我将学习深度分为知道能做什么、知道怎么做、能够熟练使用这三个程度，每一个程度都是下一个程度的前提条件，我们的学习也将遵循这样的规律。我就见过有学生用了大量的时间和精力来学习 Struts，结果在工作中却不使用 Struts，后来我发现，虽然他会照葫芦画瓢地写 Struts 的代码，但是不知道在什么地方用 Struts 技术，这样的话等于完全没学。这本书涉及的大多是需要熟练使用的内容。

学习本书的几点建议

（1）在我提问的地方，停顿一下，思考一下再向后看。

（2）处理开始的代码，在其他代码前，我都会描述是做什么的，请看完描述，不要直接看或练习我提供的代码，而是放下书，尝试着自己实现，即便是失败了，再看我的代码，你也是主动学习，而不是被动接受。

（3）我会在需要的地方提示你练习，或者建议你停下来明天再学，这些都是我长期教学经验的积累，是学习曲线所提供的节奏，希望你能遵从我的这些建议。

（4）我所用的软件环境是 JDK、Eclipse、MySQL、Tomcat，这些软件都是免费的，开始只需要安装 JDK 和 Eclipse 就可以了，其他软件可以在学到的时候安装。我不提倡在学习期间过度依赖工具，所以 Eclipse 在我看来只是一个编辑工具，我甚至都不会涉及单步调试的使用，也不会用到 Eclipse 中任何的向导，因此你可以安装更加小巧的工具。之所以不建议使用记事本来学习，是因为记事本完全没有提示功能，而提示功能能够帮助你去探索。我不认为搭建开发环境是多么重要的内容，所以将环境搭建内容放到附录里，如果你有能力自己搭建的话，完全可以不看附录，或者找个懂的人帮你搭建开发环境，我也不认为有什么不可以的。

目　录

第 1 章　认识 Java 程序……1

1.1　写代码前的准备……1

1.1.1　程序的入口……2

1.1.2　初步理解类和对象……4

1.2　画王八……5

1.2.1　运行 Java 程序……8

1.2.2　绘图……11

1.3　满天星星……19

1.4　飞行的小球……24

1.4.1　使用线程……25

1.4.2　线程的生命周期……29

1.5　小球撞墙……31

1.6　下大雪……35

1.7　键盘控制小球……40

1.8　打字母的游戏……48

1.9　鼠标控制小球……59

1.10　第一阶段总结……60

第 2 章　开始一个项目……63

2.1　聊天界面……63

2.1.1　任务描述……63

2.1.2　做按钮……64

2.1.3　Java 的布局思想……66

2.1.4　登录界面……71

2.1.5 主界面 ····· 72
2.2 响应用户输入 ····· 75
2.2.1 任务描述 ····· 75
2.2.2 事件响应 ····· 76
2.2.3 关于字符串内容的比较 ····· 79
2.2.4 取得用户名和密码 ····· 81
2.2.5 用面向对象的思想重写 ····· 82
2.2.6 上溯和下溯的讨论 ····· 84
2.3 IO 流 ····· 86
2.3.1 任务描述 ····· 86
2.3.2 读一个字符 ····· 87
2.3.3 读整个文件 ····· 91
2.3.4 复制文件 ····· 93
2.3.5 复制大文件 ····· 94
2.3.6 文件的加密/解密 ····· 97
2.3.7 异常的干扰 ····· 101
2.3.8 字符流 ····· 103
2.3.9 实现聊天记录 ····· 107
2.4 建立网络通信 ····· 110
2.4.1 什么是网络 ····· 110
2.4.2 在网络上传消息 ····· 114
2.4.3 到服务器验证用户名和密码 ····· 119
2.4.4 将聊天信息发送到服务器端 ····· 123
2.5 数据库访问 ····· 130
2.5.1 接触 MySQL ····· 133
2.5.2 创建和删除数据库 ····· 136
2.5.3 创建、修改和删除表 ····· 137
2.5.4 关于数据库设计 ····· 139
2.5.5 学习添加、删除和修改数据 ····· 142
2.5.6 查询数据 ····· 143
2.5.7 SQL 复习 ····· 149
2.5.8 用 Java 访问数据库 ····· 149
2.5.9 用户身份验证 ····· 158
2.5.10 将代码融入项目中 ····· 161
2.5.11 讨论反射 ····· 162
2.6 应对多用户访问 ····· 169

第 3 章　获得逻辑能力 …… 172
3.1　用数组实现的记事本 …… 172
3.2　使用链表的记事本 …… 178
3.3　让 Java 系统库帮助你 …… 184
3.4　思考面向对象和面向过程的不同 …… 187
3.5　深入学习 ArrayList 和 LinkedList …… 188
3.6　Set 集合 …… 192
3.7　试试二分查找法，理解二叉树 …… 199
3.8　复制一个目录的内容 …… 201
3.9　Map …… 211
3.10　保存用户的 Socket …… 211
3.11　同步用户名 …… 215
3.12　多用户转发逻辑 …… 219
第 4 章　理解面向对象 …… 230
4.1　用面向对象的思想重写聊天程序 …… 230
4.2　做一个数据库的管理工具 …… 238
4.3　驾驭 JTable …… 243
4.4　有更好的方法驾驭 JTable …… 247
4.5　用面向对象的方法驾驭 JTable …… 248
4.6　完成资源管理器 …… 256
4.7　有没有更好的参数传递方式 …… 274
附录 A　准备编程环境 …… 276

第1章
认识 Java 程序

学习曲线

关注点：单词和简单逻辑

代码量：500 行

1.1 写代码前的准备

好了，我们开始吧！一开始是一段艰难的过程，因为我想尽快开始写程序，但是 Java 不是一个很初级的技术，我们不得不理解一些概念，当写了第一个程序以后，你会发现剩下的都没那么困难，别忘了，如果理解不了，就丢掉这些讨论，只要你能将代码敲上 20 遍，该理解的就理解了，鼓足勇气越过这个坎吧。

既然要学编程，就要先弄明白什么是程序。（不要忘记，在我提问的时候停顿一下，自己想想答案）有的教科书上有简单的定义：程序是数据结构加算法。无论是这个答案还是百度上的定义，都让人崩溃，本来一个还能蒙一下的词，变成一堆都没法猜的词。我来说什么是程序：“我想让你站起来，走到门口，下楼梯，到公交车站，坐车回家”，就这么一系列的事情，我跟着你说一句，你来做一句，那么我说的叫**命令**。大多数情况下，我们操作电脑，就是在不断地命令电脑，如果我将这些话写在一张纸上，你照着这张纸去做，那么纸上写的就叫**程序**。

可是现在照着纸去做的不是你，是一台比傻子还傻的电脑，它看不懂人话，人话太复杂了，电脑的语言是 0 和 1，但是让人用 0 和 1 跟电脑说话也太折磨人了，所以人们找到了一个办法，将人类的语言简化，简化到一个极致，产生了一门新的语言，我们管它叫做 **Java 语言**，当然还会产生其他语言，比如 C、C++、C#等，这些语言都位于复杂的人类语言和计算机能看懂的 0、1 之间。

人如何能够用 Java 语言说话？当然是通过学习了。那么计算机如何能够看懂 Java 语言呢？要通过一个软件 JDK，JDK 的意思是“Java 开发工具”，JDK 将写好的 Java 程序翻译成计算机所能够认识的 0 和 1，这个翻译过程我们暂时不深究，以后有兴趣可以看一下其他书。

也就是说，我们要学习一个新的、简化的 Java 语言，以便跟计算机说话，而 JDK 会将我所说的话翻译给计算机看，所以需要在计算机上安装 JDK。JDK 是免费的软件，如果你不知道如何获得或者安装，到书后面的附录里看 JDK 安装的内容。

现在我们打开一张纸，在计算机中所谓的打开一张纸就是用记事本创建一个新的文本文件。不是有 IDE（集成开发环境）吗？没错，现在有很多流行的集成开发环境，如 Eclipse、JBuilder 之类的工具，它们提供了很多辅助的工具，简化了程序员的劳动，但是这些方便对于初学者来说是有害的，尤其是学到后期，工具提供的向导通过简单的点击便可以产生大量的代码，你以为自己学会了这些知识，其实只不过学会了如何使用工具，一旦换成其他工具，或者在生成代码的过程中发生了问题，学习者就会无所适从。我们要学会的不是工具，而是 Java 语言本身。

本书并不介绍任何集成开发工具的使用，使用记事本即可学习所有的代码，如果你觉得记事本过于简陋，推荐你使用 EditPlus。

1.1.1 程序的入口

如果是写在纸上的一段话，人们通常会从最上边开始看，这是人类的阅读习惯，或者说是一个约定。但是计算机不是从第一行开始看 Java 程序的，计算机从一个特定的位置开始读，这个位置叫做主函数，还有人说那是程序的入口。

我们又碰到了一个新的名词：函数。什么是函数？其实我们生活中充满了函数，早晨起床时，你喊了一嗓子：“妈！给我做早饭。”这是典型的函数运用，你并不知道早饭是如何做出来的，你就喊了一下，我们管这个叫做**函数调用**，妈妈提供了具体的实现，也就是说，妈妈提供了做早饭的函数，你调用了这个函数。是不是我们的生活中充满了函数和函数调用？上饭店点菜是函数调用，坐出租车是函数调用。

那么回来看所谓的主函数，大多数的计算机语言都将主函数写成 main(){}，程序从这个地方开始运行你写的语句。我们先来看一下这个主函数的各个部分，main 是函数的名字，计算机会找这个名字。这是一个怎样的过程呢？我们单击 Windows 的“开始”菜单，在运行的框里输入 cmd，然后回车，你会看见一个黑窗体，这是我们常说的 DOS 窗体（见图 1-1）。事实上我们看到的图形界面只是为了能够更方便操作而设计的，计算机的操作系统本质上就是这个黑窗体。

图 1-1

现在我们的手离开计算机，想一下，计算机现在是忙着的还是闲着的？（请想一下再往后看）计算机是忙着的，它在忙着让那个白色的小横线闪烁，你或许觉得这真不是个事，但是从计