

累计销量

10万册

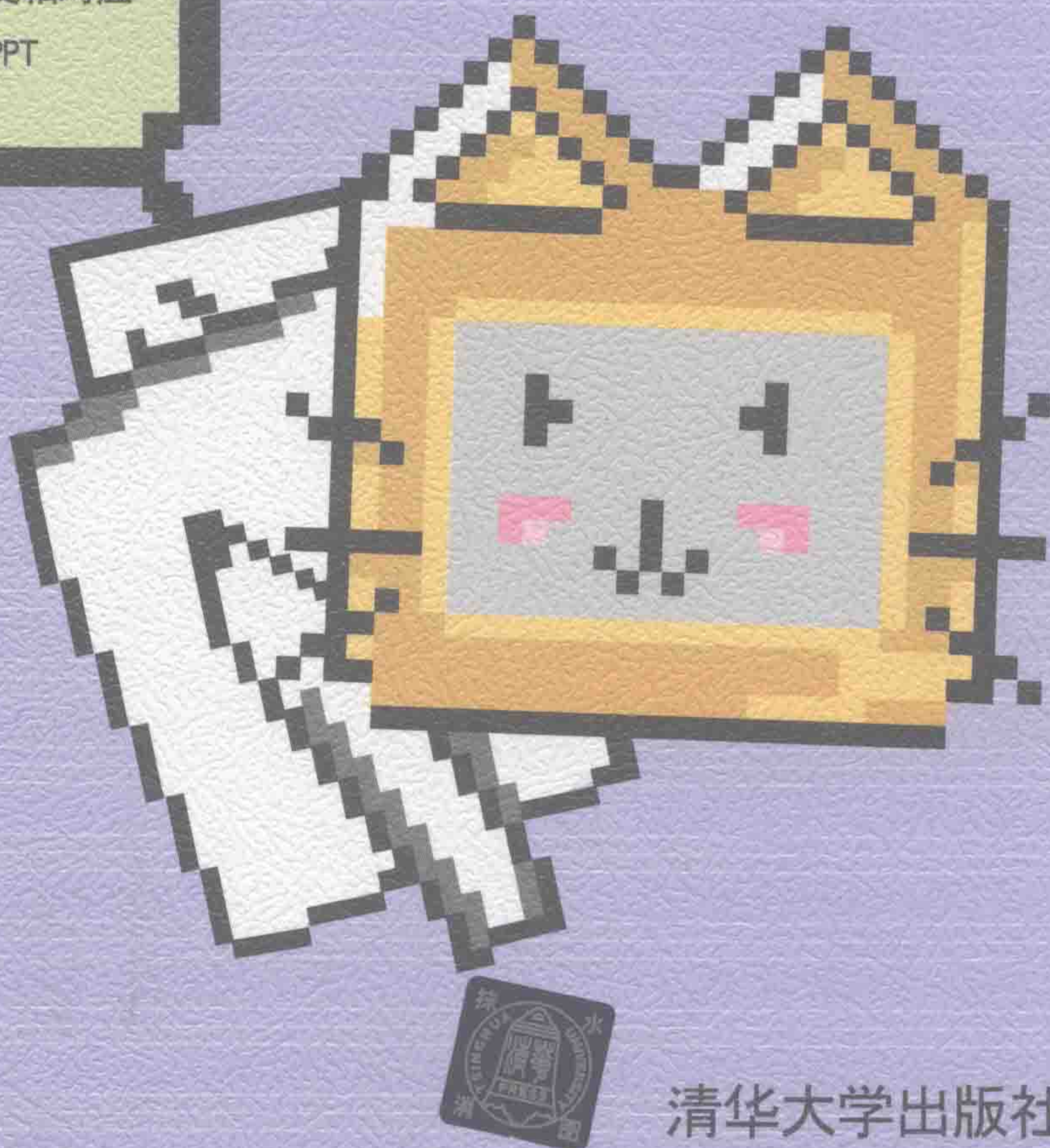
JWorld@TW技术论坛版主
Java权威技术顾问与专业讲师

全新改版!

JDK 9 Java 学习笔记

林信良 编著

- 分享作者学习Java心得
- 涵盖OCP/JP(原SCJP)考试范围
- 提供Java 9新功能快速索引
- 深入探讨Java模块平台系统
- JDK基础与IDE操作交相对应
- 提供Lab文档与教学PPT



碁峯

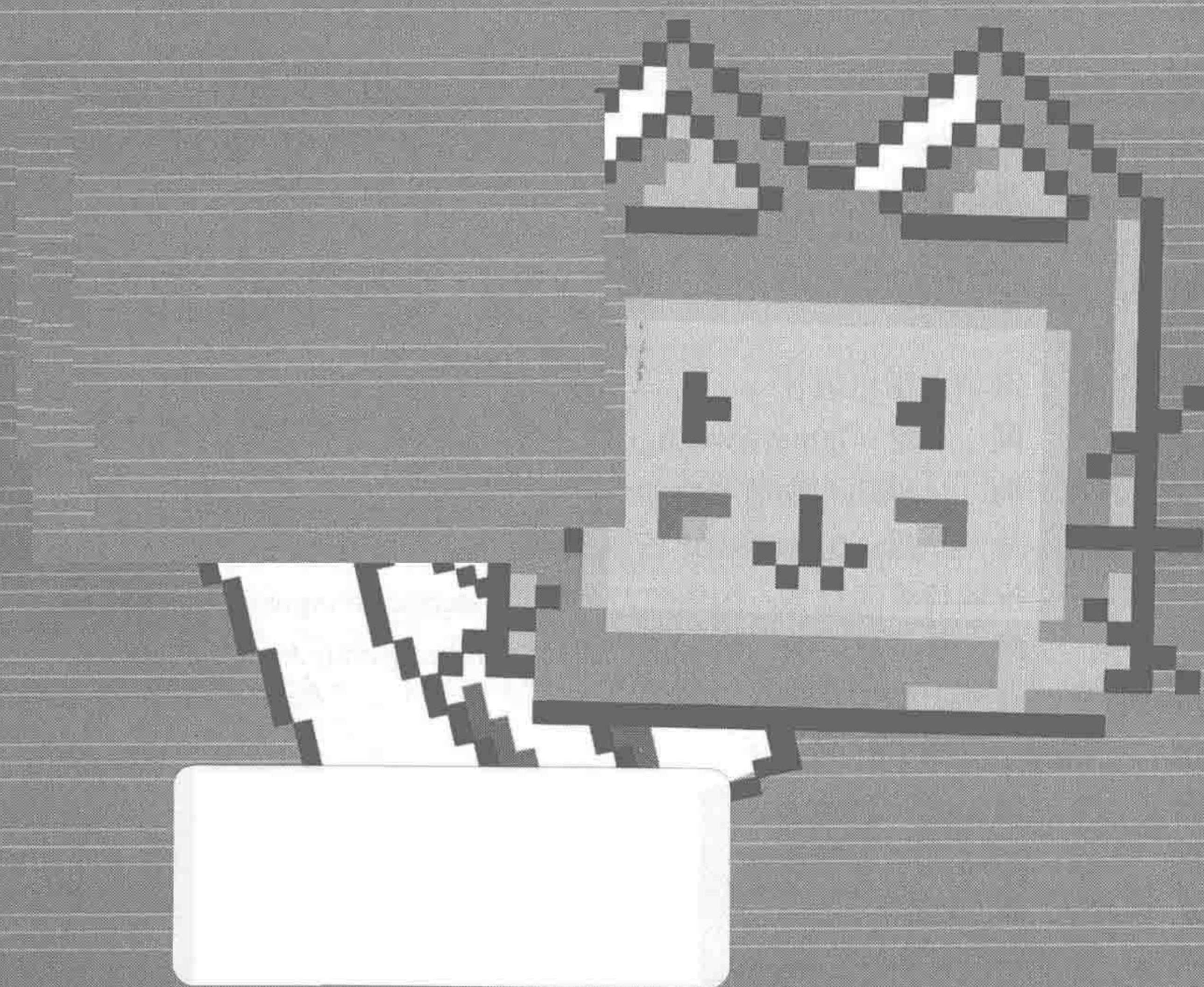
www.gotop.com.tw



清华大学出版社

JDK 9 Java 学习笔记

林信良 编著



清华大学出版社

北 京

内 容 简 介

本书是作者多年来教学实践经验的总结,汇集了学员在学习 Java 或认证考试时遇到的概念、操作、应用等问题及解决方案。

本书针对 Java SE 9 新功能全面改版,无论是章节架构或范例程序代码,都做了重新编写与全面翻新,并详细介绍了 Java 9 的模块化, JVM、JRE、Java SE API、JDK 与 IDE 之间的对照关系。必要时可从 Java SE API 的源代码分析,了解各种语法在 Java SE API 中如何应用。对于建议练习的范例提供了 Lab 文档,以突出练习重点。此外,本书还将 IDE 操作纳为教学内容之一,让读者能与实践相结合,轻松快速掌握 Java 编程技巧。

本书适合 Java 的初、中级读者以及广大 Java 应用开发人员阅读。

本书资料可通过 <http://www.tupwk.com.cn/downpage> 免费下载。

北京市版权局著作权合同登记号 图字: 01-2018-3785

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话: 010-62782989 13701121933

图书在版编目(CIP)数据

Java JDK 9 学习笔记 / 林信良编著. —北京: 清华大学出版社, 2018
ISBN 978-7-302-50118-3

I. ①J… II. ①林… III. ①JAVA 语言—程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2018)第 100148 号

责任编辑: 王 定

封面设计: 牛艳敏

版式设计: 思创景点

责任校对: 孔祥峰

责任印制: 李红英

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座

邮 编: 100084

社 总 机: 010-62770175

邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者: 清华大学印刷厂

经 销: 全国新华书店

开 本: 185mm×260mm

印 张: 36.75

字 数: 941 千字

版 次: 2018 年 6 月第 1 版

印 次: 2018 年 6 月第 1 次印刷

定 价: 98.00 元

产品编号: 079536-01

序

当你拿起这本书，翻开这篇序，我便有了机会问你一个问题：“为什么想翻开这本书？”

“这个梗上一版的序用过啦！”

好吧！那学 Java 目的是什么呢？从事程序设计？那么我就有个新梗了：“什么是‘设计’呢？”

唔！好难回答的问题，来拜一下 Google 大神吧！搜索“什么是设计”，看完答案之后就更不懂了，在没有一个具体目标之前，得到的可能都是近乎空洞的答案吧！

试着在身边找个东西，比如键盘，你觉得设计得好吗？不好？哪里觉得不好？触感！什么样的不好？反馈的力道！反馈是来自哪里？键轴！键轴是由哪些成分组成的呢？底座、弹簧、轴心、轴帽！造成反馈差异性的主要来源是什么？弹簧和轴心！喔？弹簧啊？它的圈数是多少呢？……

当你逐一挖掘出其中的元素之后，面对一个觉得设计不错的键盘，或许你就能知道其中有哪些“设计”了。

很多时候，当谈到一件东西设计得好或不好时，并不会明确地知道自己在讲什么，只是综合了各种感觉而得到的模糊结论。当然，在平时生活中，并不用每件事物都得探究到底，只是当某个事物是喜爱的、想赖以维生的，或者是两者综合，以为喜欢某个事物，因而想要进一步赖以维生，这个时候就不能只靠个模糊结论来搪塞下去了。

当你拿起这本书，表示选择了 Java 这门程序设计语言，为什么呢？因为 Java 可以写程序。可以写程序的语言很多啊！因为可以写手机 APP？那为什么不选 Objective-C 或 Swift 呢？因为听说业界很缺？喔！缺的是哪个工作性质的职位？手机……好吧！再问下去，可能有人只是被说服参加了三个月的 APP 补习课程，只好硬着头皮继续学下去了……

Java 本身是门程序设计语言，本身就有设计的成分在里头，基于面向对象典范，后来有了一些函数式典范的影子。Java 不是门简洁的语言，然而为了解决这方面问题，近来在语言设计上有了不少简化语法，把 Java 用在许多层面，Java 面对了为模块化制订标准的需求……

然而解决问题并不能只靠程序语言，面对不同的问题，需要设计各种流程，也有不同的数据结构设计。对于更复杂的问题，得靠更有效率的算法。当项目有一定规模，有弹性的架构设计是必要的。为了掌握程序的行为，就得设计可测试的程序；想要避免被入侵，必须将安全上的设计纳入考虑；当庞大的数据迎面而来，平行化方案的设计可能就得出现了……

设计是用来解决问题的！你喜欢或讨厌一种设计，代表它在解决问题上是否顺你的心、得你的意，从而认定一个设计是否优雅。

既然已经选择了 Java，那表示你得接受它的设计了。那么接下来的问题，就在于如何使用 Java 来表达你的设计了。不过，这也得真的有能够表达的设计，你有能表达的算法设计吗？数据结构设计？测试上的设计？架构上的设计？安全上的设计？平行处理的设计？……

过去、现在或未来，你写的程序中，真的有“设计”的成分在吗？

林信良
2017 年 12 月

导 读

这份导读让你可以更了解如何使用本书。

新旧版差异

就目录上来说，你可以看出的差异是，上一版为 18 个章节，新版为 19 个章节，第 19 章“深入模块化”是新的章节，也是 JDK9 最重要的新增功能。然而，认识模块化最好的方式是从实际的例子着手，因此第 1~18 章，全部的范例都是采用模块项目构建。而在各章说明时若有需要，也适时地带入了常用的模块化概念。第 19 章一开始则是整理前 18 个章节遇到过的模块化介绍，然后紧接着深入探讨模块化。

当然，照例要谈一些 JDK9 的其他新增功能，散落在各章节中适当的地方介绍。如果发现页侧有  图标，表示提及 JDK9 新功能，本书还提供了 JDK9 新功能快速查询目录。

虽然程序常用来处理计算，然而许多开发者对数字处理其实认识不多，因而第 15 章通用 API 增加了数字处理的内容；为了认识 JDK9 Stack-Walking API，读者有必要先认识如何取得并运用 `StackTraceElement` 来进行堆栈追踪，因而在第 15 章还增加了一个小节来介绍堆栈追踪，对于读者了解应用程序行为，或者是处理 Bug，应该会有所帮助。

在第 2 章介绍了模块化之后，范例项目便基于模块化设计了，并使用了 JDK9 的语法增强或改进部分程序代码；在 9.1.6 节介绍 Lambda 之后，为了提高可读性，使用 Lambda 相关语法或 API 来操作程序范例。

旧版有个附录 B“窗口程序设计”，在新版中删掉了，这表示了 Java 在窗口程序这块的地位。当然，Java 有 Java FX 这项技术，如果仍希望使用 Java 进行窗口程序设计，可以寻找 Java FX 的专书。附录 B 虽然不在了，不过范例项目的程序代码留着作为第 19 章的练习，读者可以自行研读原始码，并试着将之模块化。如果一定需要点说明，可以参考旧版《Java SE 6 技术手册》第 19 章说明：

- github.com/JustinSDK/JavaSE6Tutorial/blob/master/docs/CH19.md

字型

本书正文中与程序代码相关的文字，都用固定字体来加以呈现，以与一般名词相区别。例如，JDK 是一般名词，而 `String` 则是程序代码相关文字，所以使用了固定字体以区分。

程序范例

读者可以在以下网址下载本书的范例：

- <http://www.tupwk.com.cn/downpage>
- http://books.gotop.com.tw/v_ACL052100

本书许多范例都使用完整程序操作来展现，如看到以下程序代码示范时：

ClassObject Guess.java

```
package cc.openhome;

import java.util.Scanner; ← ① 告诉编译程序接下来想偷懒
import static java.lang.System.out;

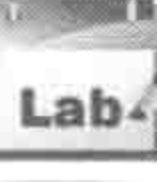
public class Guess {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in); ← ② 建立 Scanner 实例
        int number = (int) (Math.random() * 10);
        int guess;

        do {
            System.out.print("猜数字 (0 ~ 9) :");
            guess = scanner.nextInt(); ← ③ 取得下一个整数
        } while(guess != number);

        out.println("猜中了...XD");
    }
}
```

范例开始的左边名称为 ClassObject 表示可以在范例文件的 samples 文件夹的各章节文件夹中找到对应的 ClassObject 项目；而右边名称为 Guess.java 表示可以在项目中找到 Guess.java 文件。如果程序代码中出现标号与提示文字，表示在后续的正文中，会有对应于标号及提示的更详细说明。

原则上，建议读者每个项目范例都能亲自动作撰写，但出于教学时间或操作时间上的考虑，本书有建议进行的练习。如果在范例开始前有个  图标，例如：

 Game1 SwordsMan.java

```
package cc.openhome;

public class SwordsMan extends Role {
    public void fight() {
        System.out.println("挥剑攻击");
    }
}
```

表示建议范例动手操作，而且在范例文件的 labs 文件夹中提供了练习项目的基础，打开项目后，完成项目中遗漏或必须补齐的程序代码或设定即可。

如果使用以下程序代码呈现，则表示它是一个完整的程序内容，而不是项目的一部分，这主要用来展现一个完整文档的撰写方法。


```
public class Hello {  
    public static void main(String[] args) {  
        System.out.println("Hello!World!");  
    }  
}
```

如果使用以下程序代码，则表示它是个代码段，主要展现程序撰写时需要特别注意的片段：

```
SwordsMan swordsMan = new SwordsMan();  
...  
System.out.printf("剑士 (%s, %d, %d)%n", swordsMan.getName(),  
    swordsMan.getLevel(), swordsMan.getBlood());  
Magician magician = new Magician();  
...  
System.out.printf("魔法师 (%s, %d, %d)%n", magician.getName(),  
    magician.getLevel(), magician.getBlood());
```

提示框

在本书中会出现以下提示框：

- 提示 >>>

针对课程中所提到的观点，提供了一些额外的资源或思考方向，暂时忽略这些提示对课程的影响，但有时间的话，针对这些提示做阅读、思考或讨论是有帮助的。
- 注意 >>>

针对课程中所提到的观点，以提示框方式特别呈现出必须注意的一些使用方式、陷阱或避开问题的方法，看到这个提示框时请读者集中精神阅读。

附录

范例文件包括本书中的所有范例，提供 NetBeans 范例项目，附录 A 用于说明如何使用这些范例项目。

联系作者

若有本书勘误反馈等相关书籍问题，可通过网站与作者联系。网址如下：
<http://openhome.cc>

目 录

Chapter 1	Java 平台概论	1
1.1	Java 不只是语言	2
1.1.1	前世今生	2
1.1.2	三大平台	5
1.1.3	JCP 与 JSR	6
1.1.4	Oracle JDK 与 OpenJDK	7
1.1.5	建议的学习路径	8
1.2	JVM/JRE/JDK	12
1.2.1	什么是 JVM	12
1.2.2	区分 JRE 与 JDK	14
1.2.3	下载、安装 JDK	15
1.2.4	认识 JDK 安装内容	18
1.3	重点复习	19
Chapter 2	从 JDK 到 IDE	21
2.1	从 Hello World 开始	22
2.1.1	撰写 Java 原始码	22
2.1.2	PATH 是什么	24
2.1.3	JVM(java)与 CLASSPATH	27
2.1.4	编译程序(javac)与 CLASSPATH	29
2.2	管理原始码与位码文档	30
2.2.1	编译程序(javac)与 SOURCEPATH	30
2.2.2	使用 package 管理类	32
2.2.3	使用 import 偷懒	34
2.3	初识模块平台系统	36
2.3.1	JVM(java)与 module- path	37
2.3.2	编译程序(javac)与 module-path	39

2.3.3	编译程序(javac)与 module-source-path	40
2.4	使用 IDE	41
2.4.1	IDE 项目管理基础	41
2.4.2	使用了哪个 JRE	47
2.4.3	类文档版本	49
2.5	重点复习	51
Chapter 3	基础语法	53
3.1	类型、变量与运算符	54
3.1.1	类型	54
3.1.2	变量	57
3.1.3	运算符	60
3.1.4	类型转换	65
3.2	流程控制	68
3.2.1	if...else 条件式	68
3.2.2	switch 条件式	70
3.2.3	for 循环	72
3.2.4	while 循环	73
3.2.5	break、continue	74
3.3	重点复习	76
3.4	课后练习	77
Chapter 4	认识对象	78
4.1	类与对象	79
4.1.1	定义类	79
4.1.2	使用标准类	81
4.1.3	对象指定与相等性	84
4.2	基本类型打包器	86
4.2.1	打包基本类型	86
4.2.2	自动装箱、拆箱	87
4.2.3	自动装箱、拆箱的内幕	88
4.3	数组对象	90

4.3.1	数组基础	91
4.3.2	操作数组对象	93
4.3.3	数组复制	98
4.4	字符串对象	100
4.4.1	字符串基础	101
4.4.2	字符串特性	103
4.4.3	字符串编码	107
4.5	查询 Java API 文件	108
4.6	重点复习	113
4.7	课后练习	114
Chapter 5	对象封装	116
5.1	何谓封装	117
5.1.1	封装对象初始流程	117
5.1.2	封装对象操作流程	119
5.1.3	封装对象内部数据	121
5.2	类语法细节	123
5.2.1	public 权限修饰	123
5.2.2	关于构造函数	125
5.2.3	构造函数与方法重载	126
5.2.4	使用 this	128
5.2.5	static 类成员	130
5.2.6	不定长度自变量	135
5.2.7	内部类	136
5.2.8	传值调用	138
5.3	重点复习	140
5.4	课后练习	141
Chapter 6	继承与多态	142
6.1	何谓继承	143
6.1.1	继承共同行为	143
6.1.2	多态与 is-a	147
6.1.3	重新定义行为	150
6.1.4	抽象方法、抽象类	153
6.2	继承语法细节	154
6.2.1	protected 成员	154
6.2.2	重新定义的细节	156
6.2.3	再看构造函数	157
6.2.4	再看 final 关键字	159

6.2.5	java.lang.Object	160
6.2.6	关于垃圾收集	165
6.2.7	再看抽象类	167
6.3	重点复习	169
6.4	课后练习	170
Chapter 7	接口与多态	171
7.1	何谓接口	172
7.1.1	接口定义行为	172
7.1.2	行为的多态	175
7.1.3	解决需求变化	178
7.2	接口语法细节	183
7.2.1	接口的默认	183
7.2.2	匿名内部类	187
7.2.3	使用 enum 枚举常数	190
7.3	重点复习	192
7.4	课后练习	193
Chapter 8	异常处理	194
8.1	语法与继承架构	195
8.1.1	使用 try、catch	195
8.1.2	异常继承架构	197
8.1.3	要抓还是要抛	202
8.1.4	贴心还是造成麻烦	205
8.1.5	认识堆栈追踪	206
8.1.6	关于 assert	210
8.2	异常与资源管理	213
8.2.1	使用 finally	213
8.2.2	自动尝试关闭资源	215
8.2.3	java.lang.AutoCloseable 接口	217
8.3	重点复习	221
8.4	课后练习	222
Chapter 9	Collection 与 Map	223
9.1	使用 Collection 收集对象	224
9.1.1	认识 Collection 架构	224
9.1.2	具有索引的 List	225
9.1.3	内容不重复的 Set	228

9.1.4	支持队列操作的 Queue	232
9.1.5	使用泛型	234
9.1.6	简介 Lambda 表达式	238
9.1.7	Iterable 与 Iterator	240
9.1.8	Comparable 与 Comparator	243
9.2	键值对应的 Map	248
9.2.1	常用 Map 操作类	249
9.2.2	访问 Map 键值	252
9.3	不可变的 Collection 与 Map	255
9.3.1	浅谈不可变特性	255
9.3.2	Collections 的 unmodifiableXXX() 方法	256
9.3.3	List、Set、Map 的 of() 方法	258
9.4	重点复习	260
9.5	课后练习	262
Chapter 10	输入/输出	263
10.1	InputStream 与 OutputStream	264
10.1.1	串流设计的概念	264
10.1.2	串流继承架构	266
10.1.3	串流处理装饰器	269
10.2	字符处理类	273
10.2.1	Reader 与 Writer 继承 架构	274
10.2.2	字符处理装饰器	275
10.3	重点复习	277
10.4	课后练习	278
Chapter 11	线程与并行 API	279
11.1	线程	280
11.1.1	线程简介	280
11.1.2	Thread 与 Runnable	282
11.1.3	线程生命周期	284

11.1.4	关于 ThreadGroup	290
11.1.5	synchronized 与 volatile	292
11.1.6	等待与通知	301
11.2	并行 API	305
11.2.1	Lock、ReadWriteLock 与 Condition	305
11.2.2	使用 Executor	313
11.2.3	并行 Collection 简介	323
11.3	重点复习	326
11.4	课后练习	327
Chapter 12	Lambda	328
12.1	认识 Lambda 语法	329
12.1.1	Lambda 语法概览	329
12.1.2	Lambda 表达式与函数 接口	332
12.1.3	Lambda 遇上 this 与 final	334
12.1.4	方法与构造函数参考	336
12.1.5	接口默认方法	338
12.2	Functional 与 Stream API	343
12.2.1	使用 Optional 取代 null	343
12.2.2	标准 API 的函数接口	345
12.2.3	使用 Stream 进行管道 操作	348
12.2.4	进行 Stream 的 reduce 与 collect	351
12.2.5	关于 flatMap() 方法	356
12.2.6	Stream 相关 API	359
12.2.7	JDK9 Optional 与 Stream 增强	360
12.3	Lambda、平行化与异步 处理	362
12.3.1	Stream 与平行化	362
12.3.2	Arrays 与平行化	366

12.3.3	CompletableFuture 非同步处理	367
12.3.4	JDK9 CompletableFuture 增强	369
12.4	重点复习	370
12.5	课后练习	371
Chapter 13	时间与日期	372
13.1	认识时间与日期	373
13.1.1	时间的度量	373
13.1.2	年历简介	374
13.1.3	认识时区	375
13.2	认识 Date 与 Calendar	376
13.2.1	时间轴上瞬间的 Date	376
13.2.2	格式化时间日期的 DateFormat	377
13.2.3	处理时间日期的 Calendar	379
13.2.4	设定 TimeZone	382
13.3	新时间日期 API	383
13.3.1	机器时间观点的 API	383
13.3.2	人类时间观点的 API	385
13.3.3	对时间的运算	387
13.3.4	年历系统设计	389
13.4	重点复习	390
13.5	课后练习	391
Chapter 14	NIO 与 NIO2	393
14.1	认识 NIO	394
14.1.1	NIO 概述	394
14.1.2	Channel 架构与操作	395
14.1.3	Buffer 架构与操作	396
14.2	NIO2 文件系统	398
14.2.1	NIO2 架构	398
14.2.2	操作路径	399
14.2.3	属性读取与设定	401
14.2.4	操作文档与目录	404
14.2.5	读取、访问目录	406
14.2.6	过滤、搜索文档	410

14.3	重点复习	412
14.4	课后练习	413
Chapter 15	通用 API	414
15.1	日志	415
15.1.1	日志 API 简介	415
15.1.2	指定日志层级	417
15.1.3	使用 Handler 与 Formatter	419
15.1.4	自定义 Handler、Formatter 与 Filter	420
15.1.5	使用 logging. properties	422
15.2	国际化基础	423
15.2.1	使用 ResourceBundle	423
15.2.2	使用 Locale	424
15.3	规则表示式	426
15.3.1	规则表示式简介	426
15.3.2	Pattern 与 Matcher	433
15.4	处理数字	435
15.4.1	使用 BigInteger	435
15.4.2	使用 BigDecimal	437
15.4.3	数字的格式化	439
15.5	再谈堆栈追踪	441
15.5.1	获取 StackTraceElement	441
15.5.2	JDK9 的 Stack- Walking API	443
15.6	重点复习	447
15.7	课后练习	448
Chapter 16	整合数据库	449
16.1	JDBC 入门	450
16.1.1	JDBC 简介	450
16.1.2	连接数据库	454
16.1.3	使用 Statement、 ResultSet	459
16.1.4	使用 PreparedStatement、 CallableStatement	464

16.2	JDBC 进阶	468	18.1.2	使用 super 与?	525
16.2.1	使用 DataSource 取得 联机	468	18.2	自定义枚举	528
16.2.2	使用 ResultSet 滚动、 更新数据	471	18.2.1	了解 java.lang. Enum 类	528
16.2.3	批次更新	473	18.2.2	enum 高级运用	531
16.2.4	Blob 与 Clob	474	18.3	关于注释	536
16.2.5	交易简介	474	18.3.1	常用标准注释	536
16.2.6	metadata 简介	481	18.3.2	自定义注释类型	540
16.2.7	RowSet 简介	484	18.3.3	执行时期读取注释信息 ..	545
16.3	重点复习	486	18.4	重点复习	548
16.4	课后练习	487	18.5	课后练习	549
Chapter 17	反射与类加载器	489	Chapter 19	深入模块化	550
17.1	运用反射	490	19.1	运用模块	551
17.1.1	Class 与.class 文档	490	19.1.1	模块的种类	551
17.1.2	使用 Class. forName()	492	19.1.2	requires、exports 与 opens 细节	554
17.1.3	从 Class 获得信息	494	19.1.3	修补模块	557
17.1.4	从 Class 建立对象	496	19.1.4	放宽模块封装与依赖	558
17.1.5	操作对象方法与成员	499	19.2	模块 API	560
17.1.6	动态代理	501	19.2.1	使用 Module	560
17.1.7	当反射遇上模块	505	19.2.2	使用 ModuleDescriptor	562
17.1.8	使用 ServiceLoader	511	19.2.3	浅谈 ModuleLayer	562
17.2	了解类加载器	513	19.3	打包模块	564
17.2.1	JDK9 类加载器层级	513	19.3.1	使用 jar 打包	564
17.2.2	建立 ClassLoader 实例	516	19.3.2	使用 jmod 打包	566
17.3	重点复习	517	19.3.3	使用 jlink 建立执行时期 映像	568
17.4	课后练习	519	19.4	重点复习	569
Chapter 18	自定义泛型、枚举与 注释	520	19.5	课后练习	570
18.1	自定义泛型	521	Appendix		571
18.1.1	使用 extends 与?	521	A.1	项目环境配置	572
			A.2	打开案例	572

Java SE 9 新功能索引

Java SE 9 后的特性版本时间轴变动	15
JDK 9 文档实体布局变动	18
初探模块平台系统	36
javac 新增 -release 参数	50
支持 Unicode 8.0	54
内建 jshell	56
Java API 文件支持搜索功能	112
StackTraceElement 新增方法	207
Try-with-resources 语法改进	217
定义匿名类别时的泛型语法改进	239
List、Set、Map 新增 of() 方法	258
接口支持定义 private 方法	340
Collectors 新增 filtering() 方法	356
Collectors 新增 flatMapping() 方法	359
Optional 与 Stream 增强	360
CompletableFuture 增强	369
支持 UTF-8 编码的.properties 文件	425
Stack-Walking API	443
反射与类加载器机制	490
@Deprecated 增强	537
ElementType 新增 MODULE	545
深入模块化	551

Java 平台概论

Chapter

1

学习目标

- Java 版本迁移简介
- 认识 Java SE、Java EE、Java ME
- 认识 JDK 规范与操作
- 了解 JVM、JRE 与 JDK
- 下载与安装 JDK

1.1 Java 不只是语言

从 1995 年至今, Java 已经过了 20 多个年头, 经过这些年的改进, 正如本节标题所示, Java 已不仅是个程序语言, 也代表了解决问题的平台(Platform), 更代表了原厂、各个厂商、社群、开发者与用户沟通的成果。若仅以程序语言的角度来看待 Java, 正如冰山一角, 你仅看到 Java 身为程序语言的一部分, 而没看到 Java 身为程序语言之外, 更可贵也更为庞大的资源。

1.1.1 前世今生

一个语言的诞生有其目的, 因为这个目的而成就了该语言的主要特性。探索 Java 的历史演变, 对于掌握 Java 特性与各种可用资源, 有很大帮助。

1. Java 诞生

Java 最早是 Sun 公司绿色项目(Green Project)中撰写 Star7 应用程序的程序语言, 当时的名称并不是 Java, 而是 Oak。

绿色项目始于 1990 年 12 月, 由 Patrick Naughton、Mike Sheridan 与 James Gosling(James Gosling 被尊称为 Java 之父)主持, 目的是希望构筑出下一波计算机应用趋势并加以掌握, 他们认为下一波计算机应用趋势会集中在消费性数字产品(就像现在的 PDA、手机等消费性电子产品)的使用上。1992 年 9 月 3 日, Green Team 项目小组展示了 Star7 手持设备, 这个设备具备无线网络连接、5 寸 LCD 彩色屏幕、PCMCIA 接口等功能, 而 Oak 在绿色项目中的目的, 是用来撰写 Star7 上应用程序的程序语言。

Oak 名称的由来, 是因为 James Gosling 的办公室窗外有一棵橡树(Oak), 就取了这个名称。但后来发现 Oak 已经被注册了, 工程师们边喝咖啡边讨论着新名称, 最后灵机一动而改名为 Java。

Java 本身有许多为了节省资源而作的设计, 如动态加载类别文档、字符串池(String Pool)等特性, 这是因为 Java 一开始就是为了消费性数字产品而设计, 而这类小型装置通常有着有限内存与运算资源。

全球信息网(World Wide Web)兴起, Java Applet 成为网页互动技术的代表。1993 年第一个全球信息网浏览器 Mosaic 诞生, James Gosling 认为互联网与 Java 的一些特性不谋而合, 利用 Java Applet 在浏览器上展现互动性媒体, 在当时而言, 对视觉感官是一种革命性的颠覆。Green Team 仿照 Mosaic 开发出以 Java 技术为基础的浏览器 WebRunner(原名为 BladeRunner), 后来改名为 HotJava。虽然 HotJava 只是一个展示性产品, 但它使用 Java Applet 展现的多媒体效果立即吸引了许多人的注意, 图 1.1 所示即为 JDK 所附的 Java Applet 范例。

1995 年 5 月 23 日(这一天被公认为 Java 的誕生日), 正式将 Oak 改名为 Java, Java Development Kits(当时 JDK 的全名)1.0a2 版本正式对外发表。到 1996 年, Netscape Navigator 2.0 也正式支持 Java, Microsoft Internet Explorer 也开始支持 Java。从此, Java 在互联网的世界中逐渐风行起来。虽然 Star7 产品并不被当时消费性市场接受, 绿色项目面临被裁撤的命运, 然而全球信息网的兴起却给了 Java 新的生命与舞台。

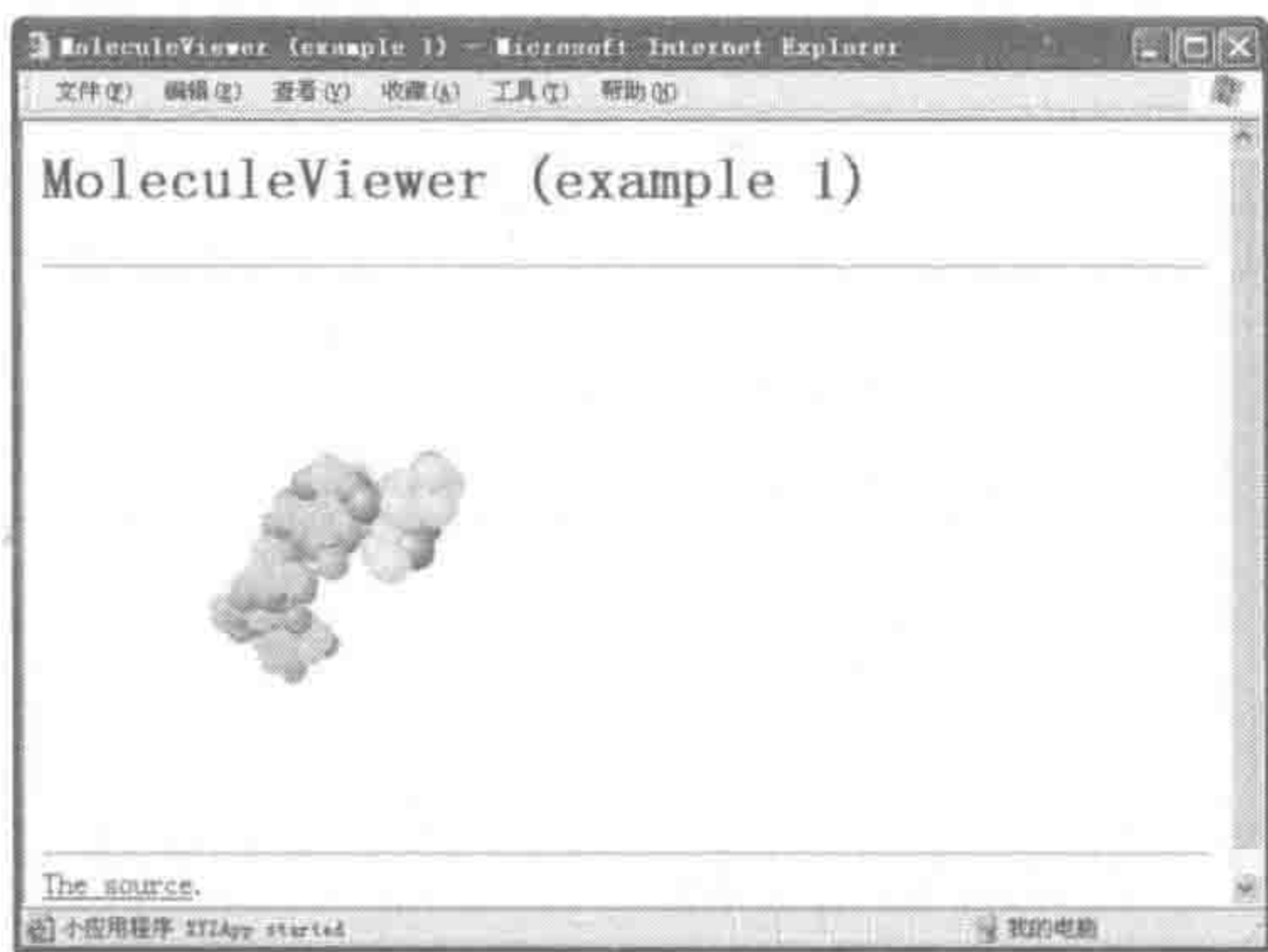


图 1.1 JDK 所附的 Java Applet 范例(JDK 文件夹\demo\applets\MoleculeViewer\example1.html)

2. 版本演进

随着 Java 越来越受到瞩目，Sun 在 1998 年 12 月 4 日发布 Java 2 Platform，简称 J2SE 1.2。Java 开发者版本一开始是以 Java Development Kit 名称发表，简称 JDK，而 J2SE 则是平台名称，包含了 JDK 与 Java 程序语言。

Java 平台标准版约以两年为周期推出重大版本更新，1998 年 12 月 4 日发表 J2SE 1.2，2000 年 5 月 8 日发表 J2SE 1.3，2002 年 2 月 13 日发表 J2SE 1.4，Java 2 这个名称也从 J2SE 1.2 开始沿用至各个版本。

2004 年 9 月 29 日发表的 Java 平台标准版的版号不是 1.5，而直接跳到 5.0，称为 J2SE 5.0，这是为了彰显这个版本与之前版本有极大不同，如语法上的简化、增加泛型(Generics)、枚举(Enum)、注释(Annotation)等重大功能。

2006 年 12 月 11 日发表的 Java 平台标准版，除了版本号之外，名称也有了变化，称为 Java Platform, Standard Edition 6，简称 Java SE 6。JDK6 全名则称为 Java SE Development Kit 6，也就是不再像以前 Java 2 带有 2 这个号码，版本号 6 或 1.6.0 都使用，6 是产品版本(Product Version)，而 1.6.0 是开发者版本(Developer Version)。

大部分的 Java 标准版平台都会取个代码名称(Code Name)，例如 J2SE 5.0 的代码名称为 Tiger(老虎)，为了引人注目，在发表会上还真的抱了一只小白老虎出来作为噱头，而许多书的封面也相应地放上老虎的图片。有关 JDK 代码名称与发布日期，如表 1.1 所示。

表 1.1 Java 版本、代码名称与发布日期

版 本	代 码 名 称	发 布 日 期
JDK 1.1.4	Sparkler(烟火)	1997/9/12
JDK 1.1.5	Pumpkin(南瓜)	1997/12/3
JDK 1.1.6	Abigail(圣经故事人物名称)	1998/4/24
JDK 1.1.7	Brutus(罗马政治家名称)	1998/9/28
JDK 1.1.8	Chelsea(足球俱乐部名称)	1999/4/8
J2SE 1.2	Playground(游乐场)	1998/12/4
J2SE 1.2.1	无	1999/3/30
J2SE 1.2.2	Cricket(蟋蟀)	1999/7/8
J2SE 1.3	Kestrel(红隼)	2000/5/8