

疯狂Java讲义



[疯狂Java讲义_下载链接1](#)

著者:李刚

出版者:电子工业出版社

出版时间:2014-7-1

装帧:平装

isbn:9787121236693

《疯狂Java讲义（第3版）（含CD光盘1张）》是《疯狂Java讲义》的第3版，第3版保持了

前两版系统、全面、讲解浅显、细致的特性，全面新增介绍了Java 8的新特性，《疯狂Java讲义（第3版）（含CD光盘1张）》大部分示例程序都采用Lambda表达式、流式API进行了改写，因此务必使用Java 8的JDK来编译、运行。

《疯狂Java讲义（第3版）（含CD光盘1张）》深入介绍了Java编程的相关方面，全书内容覆盖了Java的基本语法结构、Java的面向对象特征、Java集合框架体系、Java泛型、异常处理、Java GUI编程、JDBC数据库编程、Java注释、Java的IO流体系、Java多线程编程、Java网络通信编程和Java反射机制。覆盖了java.lang、java.util、java.text、java.io和java.nio、java.sql、java.awt、javax.swing包下绝大部分类和接口。本书全面介绍了Java 8的新的接口语法、Lambda表达式、方法引用、构造器引用、函数式编程、流式编程、新的日期、时间API、并行支持、改进的类型推断、重复注解、JDBC 4.2新特性等新特性。

与前两版类似，《疯狂Java讲义（第3版）（含CD光盘1张）》并不单纯从知识角度来讲解Java，而是从解决问题的角度来介绍Java语言，所以《疯狂Java讲义（第3版）（含CD光盘1张）》中涉及大量实用案例开发：五子棋游戏、梭哈游戏、仿QQ的游戏大厅、MySQL企业管理器、仿EditPlus的文本编辑器、多线程、断点下载工具、Spring框架的IoC容器……这些案例既能让读者巩固每章的知识，又可以让读者学以致用，激发编程自豪感，进而引爆内心的编程激情。《疯狂Java讲义（第3版）（含CD光盘1张）》光盘里包含书中所有示例的代码和《疯狂Java实战演义》的所有项目代码，这些项目可以作为《疯狂Java讲义（第3版）（含CD光盘1张）》课后练习的“非标准答案”，如果读者需要获取关于课后习题的解决方法、编程思路，可以登录<http://www.crazyit.org>站点与笔者及《疯狂Java讲义（第3版）（含CD光盘1张）》庞大的读者群相互交流。

《疯狂Java讲义（第3版）（含CD光盘1张）》为所有打算深入掌握Java编程的读者而编写，适合各种层次的Java学习者和工作者阅读，也适合作为大学教育、培训机构的Java教材。但如果只是想简单涉猎Java，则本书过于庞大，不适合阅读。

作者介绍:

李刚，十多年软件行业开发从业经验，疯狂软件教育中心教学总监。

疯狂Java实训营创始人，疯狂Java体系原创图书作者。

广东技术师范学院计算机科学系兼职副教授，51cto专家门诊特邀嘉宾。

培训的学生已在华为、IBM、阿里软件、网易、电信盈科等名企就职。

国内著名高端IT技术作家，已出版《疯狂Java讲义》《疯狂Android讲义》《轻量级Java EE企业应用实战》《疯狂iOS讲义》《疯狂Ajax讲义》《疯狂XML讲义》《经典Java EE企业应用实战》《疯狂HTML 5/CSS 3/JavaScript讲义》《Struts 2.x权威指南》等著作。其中疯狂Java体系图书经多年沉淀，赢得极高的市场认同，多次重印成为超级畅销书，并被多所“985”“211”院校选作教材。部分图书已被翻译成繁体中文版、授权到台湾地区。

目录: 第1章 Java语言概述与开发环境 1

1.1 Java语言的发展简史 2

1.2 Java的竞争对手及各自优势 4

1.2.1 C#简介和优势 4

1.2.2 Ruby简介和优势 5

1.2.3 Python简介和优势 5

1.3 Java程序运行机制	6
1.3.1 高级语言的运行机制	6
1.3.2 Java程序的运行机制和JVM	6
1.4 开发Java的准备	8
1.4.1 下载和安装Java 8的JDK	8
不是说JVM是运行Java程序的虚拟机吗？那JRE和JVM的关系是怎样的呢？	8
为什么不安装公共JRE呢？	9
1.4.2 设置PATH环境变量	10
为什么选择用户变量？用户变量与系统变量有什么区别？	11
1.5 第一个Java程序	12
1.5.1 编辑Java源代码	12
1.5.2 编译Java程序	12
当编译C程序时，不仅需要指定存放目标文件的位置，也需要指定目标文件的文件名，这里使用javac编译Java程序时怎么不需要指定目标文件的文件名呢？	13
1.5.3 运行Java程序	13
1.5.4 根据CLASSPATH环境变量定位类	14
1.6 Java程序的基本规则	15
1.6.1 Java程序的组织形式	15
1.6.2 Java源文件的命名规则	16
1.6.3 初学者容易犯的错误	17
1.7 垃圾回收机制	19
1.8 何时开始使用IDE工具	20
我想学习Java编程，到底是学习Eclipse好，还是学习NetBeans好呢？	21
1.9 本章小结	21
第2章 理解面向对象	22
2.1 面向对象	23
2.1.1 结构化程序设计简介	23
2.1.2 程序的三种基本结构	24
2.1.3 面向对象程序设计简介	26
2.1.4 面向对象的基本特征	27
2.2 UML（统一建模语言）介绍	28
2.2.1 用例图	30
2.2.2 类图	30
2.2.3 组件图	32
2.2.4 部署图	33
2.2.5 顺序图	33
2.2.6 活动图	34
2.2.7 状态机图	35
2.3 Java的面向对象特征	36
2.3.1 一切都是对象	36
2.3.2 类和对象	36
2.4 本章小结	37
第3章 数据类型和运算符	38
3.1 注释	39
3.1.1 单行注释和多行注释	39
3.1.2 文档注释	40
API文档是什么？	40
为什么要学习查看API文档的方法？	
3.2 标识符和关键字	46
3.2.1 分隔符	46
3.2.2 标识符规则	47
3.2.3 Java关键字	47
3.3 数据类型分类	48
什么是变量？变量有什么用？	48

- 3.4 基本数据类型 49
 - 3.4.1 整型 49
 - 3.4.2 字符型 51
 - 什么是字符集? 51
 - 3.4.3 浮点型 53
 - 3.4.4 数值中使用下划线分隔 54
 - 3.4.5 布尔型 54
- 3.5 基本类型的类型转换 55
 - 3.5.1 自动类型转换 55
 - 3.5.2 强制类型转换 56
 - 3.5.3 表达式类型的自动提升 58
- 3.6 直接量 59
 - 3.6.1 直接量的类型 59
 - 3.6.2 直接量的赋值 59
- 3.7 运算符 60
 - 3.7.1 算术运算符 60
 - 3.7.2 赋值运算符 63
 - 3.7.3 位运算符 63
 - 3.7.4 扩展后的赋值运算符 66
 - 3.7.5 比较运算符 66
 - 3.7.6 逻辑运算符 67
 - 3.7.7 三目运算符 68
 - 3.7.8 运算符的结合性和优先级 69
- 3.8 本章小结 70
- 第4章 流程控制与数组 71
 - 4.1 顺序结构 72
 - 4.2 分支结构 72
 - 4.2.1 if条件语句 72
 - 4.2.2 Java 7增强后的switch分支语句 76
 - 4.3 循环结构 78
 - 4.3.1 while循环语句 78
 - 4.3.2 do while循环语句 79
 - 4.3.3 for循环 80
 - 4.3.4 嵌套循环 83
 - 4.4 控制循环结构 84
 - 4.4.1 使用break结束循环 84
 - 4.4.2 使用continue忽略本次循环剩下语句 85
 - 4.4.3 使用return结束方法 86
 - 4.5 数组类型 86
 - 4.5.1 理解数组: 数组也是一种类型 86
 - int[]是一种类型吗? 怎么使用这种类型呢? 87
 - 4.5.2 定义数组 87
 - 4.5.3 数组的初始化 88
 - 能不能只分配内存空间, 不赋初始值呢? 88
 - 4.5.4 使用数组 89
 - 为什么要我记住这些异常信息? 89
 - 4.5.5 foreach循环 90
 - 4.6 深入数组 91
 - 4.6.1 内存中的数组 91
 - 为什么有栈内存和堆内存之分? 92
 - 4.6.2 基本类型数组的初始化 94
 - 4.6.3 引用类型数组的初始化 95
 - 4.6.4 没有多维数组 97
 - 我是否可以让图4.13中灰色覆盖的数组元素再次指向另一个数组? 这样不就可以扩展成

- 三维数组，甚至扩展成更多维的数组吗？ 98
- 4.6.5 Java 8增强的工具类：Arrays 99
- 4.6.6 数组的应用举例 102
- 4.7 本章小结 105
- 本章练习 105
- 第5章 面向对象（上） 106
- 5.1 类和对象 107
 - 5.1.1 定义类 107
 - 构造器不是没有返回值吗？为什么不能用void声明呢？ 109
 - 5.1.2 对象的产生和使用 110
 - 5.1.3 对象、引用和指针 110
 - 5.1.4 对象的this引用 111
- 5.2 方法详解 115
 - 5.2.1 方法的所属性 115
 - 5.2.2 方法的参数传递机制 116
 - 5.2.3 形参个数可变的方法 119
 - 5.2.4 递归方法 120
 - 5.2.5 方法重载 122
 - 为什么方法的返回值类型不能用于区分重载的方法？ 122
- 5.3 成员变量和局部变量 123
 - 5.3.1 成员变量和局部变量 123
 - 5.3.2 成员变量的初始化和内存中的运行机制 126
 - 5.3.3 局部变量的初始化和内存中的运行机制 128
 - 5.3.4 变量的使用规则 129
- 5.4 隐藏和封装 130
 - 5.4.1 理解封装 130
 - 5.4.2 使用访问控制符 130
 - 5.4.3 package、import和import static 133
 - 5.4.4 Java的常用包 138
- 5.5 深入构造器 138
 - 5.5.1 使用构造器执行初始化 138
 - 构造器是创建Java对象的途径，是不是说构造器完全负责创建Java对象？ 139
 - 5.5.2 构造器重载 139
 - 为什么要用this来调用另一个重载的构造器？我把另一个构造器里的代码复制、粘贴到这个构造器里不就可以了吗？ 141
- 5.6 类的继承 141
 - 5.6.1 继承的特点 141
 - 5.6.2 重写父类的方法 142
 - 5.6.3 super限定 144
 - 5.6.4 调用父类构造器 146
 - 为什么我创建Java对象时从未感觉到java.lang.Object类的构造器被调用过？ 148
- 5.7 多态 148
 - 5.7.1 多态性 148
 - 5.7.2 引用变量的强制类型转换 150
 - 5.7.3 instanceof运算符 151
- 5.8 继承与组合 152
 - 5.8.1 使用继承的注意点 152
 - 5.8.2 利用组合实现复用 153
 - 使用组合关系来实现复用时，需要创建两个Animal对象，是不是意味着使用组合关系时系统开销更大？ 156
- 5.9 初始化块 156
 - 5.9.1 使用初始化块 156
 - 5.9.2 初始化块和构造器 158

- 5.9.3 静态初始化块 159
- 5.10 本章小结 161
- 第6章 面向对象（下） 162
- 6.1 Java 8增强的包装类 163
 - Java为什么要对这些数据进行缓存呢? 166
- 6.2 处理对象 167
 - 6.2.1 打印对象和toString方法 167
 - 6.2.2 ==和equals方法 169
 - 上面程序中判断obj是否为Person类的实例时，为何不用obj instanceof Person来判断呢? 172
- 6.3 类成员 172
 - 6.3.1 理解类成员 172
 - 6.3.2 单例（Singleton）类 173
- 6.4 final修饰符 174
 - 6.4.1 final成员变量 175
 - 6.4.2 final局部变量 176
 - 6.4.3 final修饰基本类型变量和引用类型变量的区别 177
 - 6.4.4 可执行“宏替换”的final变量 178
 - 6.4.5 final方法 180
 - 6.4.6 final类 180
 - 6.4.7 不可变类 181
 - 6.4.8 缓存实例的不可变类 183
- 6.5 抽象类 186
 - 6.5.1 抽象方法和抽象类 186
 - 6.5.2 抽象类的作用 189
- 6.6 Java 8改进的接口 190
 - 6.6.1 接口的概念 190
 - 6.6.2 Java 8中接口的定义 190
 - 6.6.3 接口的继承 193
 - 6.6.4 使用接口 193
 - 6.6.5 接口和抽象类 195
 - 6.6.6 面向接口编程 195
- 6.7 内部类 199
 - 6.7.1 非静态内部类 199
 - 非静态内部类对象和外部类对象的关系是怎样的? 203
 - 6.7.2 静态内部类 203
 - 为什么静态内部类的实例方法也不能访问外部类的实例属性呢? 204
 - 接口里是否能定义内部接口? 205
 - 6.7.3 使用内部类 205
 - 既然内部类是外部类的成员，那么是否可以为外部类定义子类，在子类中再定义一个内部类来重写其父类中的内部类呢? 208
 - 6.7.4 局部内部类 208
 - 6.7.5 Java 8改进的匿名内部类 209
- 6.8 Java 8新增的Lambda表达式 212
 - 6.8.1 Lambda表达式入门 212
 - 6.8.2 Lambda表达式与函数式接口 214
 - 6.8.3 方法引用与构造器引用 216
 - 6.8.4 Lambda表达式与匿名内部类的联系和区别 218
 - 6.8.5 使用Lambda表达式调用Arrays的类方法 219
- 6.9 枚举类 220
 - 6.9.1 手动实现枚举类 220
 - 6.9.2 枚举类入门 221
 - 6.9.3 枚举类的成员变量、方法和构造器 222
 - 6.9.4 实现接口的枚举类 224

- 枚举类不是用final修饰了吗？怎么还能派生子类呢？ 225
- 6.9.5 包含抽象方法的枚举类 225
- 6.10 对象与垃圾回收 226
 - 6.10.1 对象在内存中的状态 227
 - 6.10.2 强制垃圾回收 227
 - 6.10.3 finalize方法 229
 - 6.10.4 对象的软、弱和虚引用 230
- 6.11 修饰符的适用范围 233
- 6.12 使用JAR文件 234
 - 6.12.1 jar命令详解 235
 - 6.12.2 创建可执行的JAR包 236
 - 6.12.3 关于JAR包的技巧 237
- 6.13 本章小结 238
- 本章练习 238
- 第7章 Java基础类库 239
 - 7.1 与用户互动 240
 - 7.1.1 运行Java程序的参数 240
 - 7.1.2 使用Scanner获取键盘输入 241
 - 7.2 系统相关 243
 - 7.2.1 System类 243
 - 7.2.2 Runtime类 245
 - 7.3 常用类 246
 - 7.3.1 Object类 246
 - 7.3.2 Java 7新增的Objects类 247
 - 7.3.3 String、StringBuffer和StringBuilder类 248
 - 7.3.4 Math类 251
 - 7.3.5 Java 7的ThreadLocalRandom与Random 253
 - 7.3.6 BigDecimal类 255
 - 7.4 Java 8的日期、时间类 257
 - 7.4.1 Date类 257
 - 7.4.2 Calendar类 258
 - 7.4.3 Java 8新增的日期、时间包 261
 - 7.5 正则表达式 263
 - 7.5.1 创建正则表达式 263
 - 7.5.2 使用正则表达式 266
 - 7.6 国际化与格式化 270
 - 7.6.1 Java国际化的思路 270
 - 7.6.2 Java支持的国家和语言 270
 - 7.6.3 完成程序国际化 271
 - 7.6.4 使用MessageFormat处理包含占位符的字符串 273
 - 7.6.5 使用类文件代替资源文件 274
 - 7.6.6 使用NumberFormat格式化数字 274
 - 7.6.7 使用DateFormat格式化日期、时间 276
 - 7.6.8 使用SimpleDateFormat格式化日期 277
 - 7.7 Java 8新增的日期、时间格式器 278
 - 7.7.1 使用DateTimeFormatter完成格式化 278
 - 7.7.2 使用DateTimeFormatter解析字符串 279
 - 7.8 本章小结 280
 - 本章练习 280
- 第8章 Java集合 281
 - 8.1 Java集合概述 282
 - 8.2 Collection和Iterator接口 283
 - 8.2.1 使用Lambda表达式遍历集合 285

- 8.2.2 使用Java 8增强的Iterator遍历集合元素 286
- 8.2.3 使用Lambda表达式遍历Iterator 287
- 8.2.4 使用foreach循环遍历集合元素 288
- 8.2.5 使用Java 8新增的Predicate操作集合 288
- 8.2.6 使用Java 8新增的Stream操作集合 289
- 8.3 Set集合 291
 - 8.3.1 HashSet类 292
 - hashCode()方法对于HashSet是不是十分重要? 293
 - 8.3.2 LinkedHashSet类 295
 - 8.3.3 TreeSet类 296
 - 8.3.4 EnumSet类 302
 - 8.3.5 各Set实现类的性能分析 303
- 8.4 List集合 304
 - 8.4.1 Java 8改进的List接口和ListIterator接口 304
 - 8.4.2 ArrayList和Vector实现类 307
 - 8.4.3 固定长度的List 308
- 8.5 Queue集合 308
 - 8.5.1 PriorityQueue实现类 309
 - 8.5.2 Deque接口与ArrayDeque实现类 309
 - 8.5.3 LinkedList实现类 311
 - 8.5.4 各种线性表的性能分析 312
- 8.6 Java 8增强的Map集合 313
 - 8.6.1 Java 8为Map新增的方法 315
 - 8.6.2 Java 8改进的HashMap和Hashtable实现类 316
 - 8.6.3 LinkedHashMap实现类 319
 - 8.6.4 使用Properties读写属性文件 319
 - 8.6.5 SortedMap接口和TreeMap实现类 320
 - 8.6.6 WeakHashMap实现类 323
 - 8.6.7 IdentityHashMap实现类 323
 - 8.6.8 EnumMap实现类 324
 - 8.6.9 各Map实现类的性能分析 325
- 8.7 HashSet和HashMap的性能选项 325
- 8.8 操作集合的工具类: Collections 326
 - 8.8.1 排序操作 326
 - 8.8.2 查找、替换操作 329
 - 8.8.3 同步控制 330
 - 8.8.4 设置不可变集合 330
- 8.9 烦琐的接口: Enumeration 331
- 8.10 本章小结 332
- 本章练习 332
- 第9章 泛型 333
 - 9.1 泛型入门 334
 - 9.1.1 编译时不检查类型的异常 334
 - 9.1.2 使用泛型 334
 - 9.1.3 Java 7泛型的“菱形”语法 335
 - 9.2 深入泛型 336
 - 9.2.1 定义泛型接口、类 336
 - 9.2.2 从泛型类派生子类 338
 - 9.2.3 并不存在泛型类 339
 - 9.3 类型通配符 339
 - 9.3.1 使用类型通配符 341
 - 9.3.2 设定类型通配符的上限 341
 - 9.3.3 设定类型形参的上限 343
 - 9.4 泛型方法 344

- 9.4.1 定义泛型方法 344
- 9.4.2 泛型方法和类型通配符的区别 346
- 9.4.3 Java 7的“菱形”语法与泛型构造器 347
- 9.4.4 设定通配符下限 348
- 9.4.5 泛型方法与方法重载 350
- 9.4.6 Java 8改进的类型推断 351
- 9.5 擦除和转换 352
- 9.6 泛型与数组 353
- 9.7 本章小结 355
- 第10章 异常处理 356
 - 10.1 异常概述 357
 - 10.2 异常处理机制 358
 - 10.2.1 使用try...catch捕获异常 358
 - 10.2.2 异常类的继承体系 360
 - 10.2.3 Java 7提供的多异常捕获 362
 - 10.2.4 访问异常信息 363
 - 10.2.5 使用finally回收资源 364
 - 10.2.6 异常处理的嵌套 366
 - 10.2.7 Java 7的自动关闭资源的try语句 366
 - 10.3 Checked异常和Runtime异常体系 368
 - 10.3.1 使用throws声明抛出异常 368
 - 10.4 使用throw抛出异常 370
 - 10.4.1 抛出异常 370
 - 10.4.2 自定义异常类 371
 - 10.4.3 catch和throw同时使用 372
 - 10.4.4 Java 7增强的throw语句 373
 - 10.4.5 异常链 374
 - 10.5 Java的异常跟踪栈 376
 - 10.6 异常处理规则 377
 - 10.6.1 不要过度使用异常 378
 - 10.6.2 不要使用过于庞大的try块 379
 - 10.6.3 避免使用Catch All语句 379
 - 10.6.4 不要忽略捕获到的异常 379
 - 10.7 本章小结 380
- 本章练习 380
- 第11章 AWT编程 381
 - 11.1 GUI (图形用户界面) 和AWT 382
 - 11.2 AWT容器 383
 - 11.3 布局管理器 386
 - 11.3.1 FlowLayout布局管理器 386
 - 11.3.2 BorderLayout布局管理器 387
 - BorderLayout最多只能放置5个组件吗? 那它也太不实用了吧? 388
 - 11.3.3 GridLayout布局管理器 389
 - 11.3.4 GridBagLayout布局管理器 390
 - 11.3.5 CardLayout布局管理器 392
 - 11.3.6 绝对定位 394
 - 11.3.7 BoxLayout布局管理器 395
 - 图11.15和图11.16显示的所有按钮都紧挨在一起, 如果希望像FlowLayout、GridLayout等布局管理器那样指定组件的间距应该怎么办? 396
 - 11.4 AWT常用组件 397
 - 11.4.1 基本组件 397
 - 11.4.2 对话框 (Dialog) 399
 - 11.5 事件处理 401
 - 11.5.1 Java事件模型的流程 401

- 11.5.2 事件和事件监听器 403
- 11.5.3 事件适配器 407
- 11.5.4 使用内部类实现监听器 408
- 11.5.5 使用外部类实现监听器 408
- 11.5.6 类本身作为事件监听器类 409
- 11.5.7 匿名内部类实现监听器 410
- 11.6 AWT菜单 410
 - 11.6.1 菜单条、菜单和菜单项 410
 - 11.6.2 右键菜单 412
- 为什么即使我没有给多行文本域编写右键菜单，但当我在多行文本域上单击右键时也一样会弹出右键菜单？ 414
- 11.7 在AWT中绘图 414
 - 11.7.1 画图的实现原理 414
 - 11.7.2 使用Graphics类 415
- 11.8 处理位图 419
 - 11.8.1 Image抽象类和BufferedImage实现类 419
 - 11.8.2 使用ImageIO输入/输出位图 421
- 11.9 剪贴板 425
 - 11.9.1 数据传递的类和接口 426
 - 11.9.2 传递文本 426
 - 11.9.3 使用系统剪贴板传递图像 428
 - 11.9.4 使用本地剪贴板传递对象引用 430
 - 11.9.5 通过系统剪贴板传递Java对象 433
- 11.10 拖放功能 435
 - 11.10.1 拖放目标 436
 - 11.10.2 拖放源 439
- 11.11 本章小结 440
- 本章练习 440
- 第12章 Swing编程 441
 - 12.1 Swing概述 442
 - 12.2 Swing基本组件的用法 443
 - 12.2.1 Java 7的Swing组件层次 443
 - 12.2.2 AWT组件的Swing实现 444
 - 为什么单击Swing多行文本域时不是弹出像AWT多行文本域中的右键菜单？ 450
 - 12.2.3 为组件设置边框 450
 - 12.2.4 Swing组件的双缓冲和键盘驱动 452
 - 12.2.5 使用JToolBar创建工具条 453
 - 12.2.6 使用JFileChooser和Java 7增强的JColorChooser 455
 - 12.2.7 使用JOptionPane 462
 - 12.3 Swing中的特殊容器 467
 - 12.3.1 使用JSplitPane 467
 - 12.3.2 使用JTabbedPane 469
 - 12.3.3 使用JLayeredPane、JdesktopPane和JInternalFrame 473
 - 12.4 Swing简化的拖放功能 480
 - 12.5 Java 7新增的Swing功能 481
 - 12.5.1 使用JLayer装饰组件 481
 - 12.5.2 创建透明、不规则形状窗口 487
 - 12.6 使用JProgressBar、ProgressMonitor和BoundedRangeModel创建进度条 489
 - 12.6.1 创建进度条 489
 - 12.6.2 创建进度对话框 492
 - 12.7 使用JSlider和BoundedRangeModel创建滑动条 494
 - 12.8 使用JSpinner和SpinnerModel创建微调控制器 497
 - 12.9 使用JList、JComboBox创建列表框 500
 - 12.9.1 简单列表框 500

- 12.9.2 不强制存储列表项的ListModel和ComboBoxModel 503
- 12.9.3 强制存储列表项的DefaultListModel和DefaultComboBoxModel 506
- 为什么JComboBox提供了添加、删除列表项的方法？而JList没有提供添加、删除列表项的方法呢？ 508
- 12.9.4 使用ListCellRenderer改变列表项外观 508
- 12.10 使用JTree和TreeModel创建树 510
 - 12.10.1 创建树 511
 - 12.10.2 拖动、编辑树节点 513
 - 12.10.3 监听节点事件 517
 - 12.10.4 使用DefaultTreeCellRenderer改变节点外观 519
 - 12.10.5 扩展DefaultTreeCellRenderer改变节点外观 520
 - 12.10.6 实现TreeCellRenderer改变节点外观 523
- 12.11 使用JTable和TableModel创建表格 524
 - 12.11.1 创建表格 525
 - 我们指定的表格数据、表格列标题都是Object类型的数组，JTable如何显示这些Object对象？ 525
 - 12.11.2 TableModel和监听器 530
 - 12.11.3 TableColumnModel和监听器 534
 - 12.11.4 实现排序 537
 - 12.11.5 绘制单元格内容 540
 - 12.11.6 编辑单元格内容 543
- 12.12 使用JFormattedTextField和JtextPane创建格式文本 546
 - 12.12.1 监听Document的变化 547
 - 12.12.2 使用JPasswordField 549
 - 12.12.3 使用JFormattedTextField 549
 - 12.12.4 使用JEditorPane 557
 - 12.12.5 使用JTextPane 557
- 12.13 本章小结 564
- 本章练习 564
- 第13章 MySQL数据库与JDBC编程 565
 - 13.1 JDBC基础 566
 - 13.1.1 JDBC简介 566
 - 13.1.2 JDBC驱动程序 567
 - 13.2 SQL语法 568
 - 13.2.1 安装数据库 568
 - 13.2.2 关系数据库基本概念和MySQL基本命令 570
 - 13.2.3 SQL语句基础 572
 - 13.2.4 DDL语句 573
 - 13.2.5 数据库约束 577
 - 13.2.6 索引 584
 - 13.2.7 视图 585
 - 13.2.8 DML语句语法 585
 - 13.2.9 单表查询 588
 - 13.2.10 数据库函数 592
 - 13.2.11 分组和组函数 594
 - 13.2.12 多表连接查询 596
 - 13.2.13 子查询 599
 - 13.2.14 集合运算 601
 - 13.3 JDBC的典型用法 602
 - 13.3.1 JDBC 4.2常用接口和类简介 602
 - 13.3.2 JDBC编程步骤 604
 - 前面给出的仅仅是MySQL和Oracle两种数据库的驱动，我看不出驱动类字符串有什么规律啊。如果我希望使用其他数据库，那怎么找到其他数据库的驱动类呢？ 604
 - 13.4 执行SQL语句的方式 607

- 13.4.1 使用Java 8新增的executeLargeUpdate方法执行DDL和DML语句 607
- 13.4.2 使用execute方法执行SQL语句 608
- 13.4.3 使用PreparedStatement执行SQL语句 610
- 13.4.4 使用CallableStatement调用存储过程 614
- 13.5 管理结果集 615
 - 13.5.1 可滚动、可更新的结果集 615
 - 13.5.2 处理Blob类型数据 617
 - 13.5.3 使用ResultSetMetaData分析结果集 622
- 13.6 Java 7的RowSet 1.1 624
 - 13.6.1 Java 7新增的RowSetFactory与RowSet 625
 - 13.6.2 离线RowSet 627
 - 13.6.3 离线RowSet的查询分页 629
- 13.7 事务处理 630
 - 13.7.1 事务的概念和MySQL事务支持 630
 - 13.7.2 JDBC的事务支持 632
 - 13.7.3 Java 8增强的批量更新 634
- 13.8 分析数据库信息 635
 - 13.8.1 使用DatabaseMetaData分析数据库信息 635
 - 13.8.2 使用系统表分析数据库信息 636
 - 13.8.3 选择合适的分析方式 637
- 13.9 使用连接池管理连接 638
 - 13.9.1 DBCP数据源 638
 - 13.9.2 C3P0数据源 639
- 13.10 本章小结 640
- 本章练习 640
- 第14章 Annotation (注释) 641
 - 14.1 基本Annotation 642
 - 14.1.1 限定重写父类方法: @Override 642
 - 14.1.2 标示已过时: @Deprecated 643
 - 14.1.3 抑制编译器警告: @SuppressWarnings 644
 - 14.1.4 Java 7的“堆污染”警告与@SafeVarargs 644
 - 14.1.5 Java 8的函数式接口与@FunctionalInterface 645
 - 14.2 JDK的元Annotation 646
 - 14.2.1 使用@Retention 646
 - 14.2.2 使用@Target 647
 - 14.2.3 使用@Documented 647
 - 14.2.4 使用@Inherited 648
 - 14.3 自定义Annotation 649
 - 14.3.1 定义Annotation 649
 - 14.3.2 提取Annotation信息 650
 - 14.3.3 使用Annotation的示例 652
 - 14.3.4 Java 8新增的重复注解 656
 - 14.3.5 Java 8新增的Type Annotation 658
 - 14.4 编译时处理Annotation 659
 - 14.5 本章小结 663
- 第15章 输入/输出 664
 - 15.1 File类 665
 - 15.1.1 访问文件和目录 665
 - 15.1.2 文件过滤器 667
 - 15.2 理解Java的IO流 668
 - 15.2.1 流的分类 668
 - 15.2.2 流的概念模型 669
 - 15.3 字节流和字符流 670
 - 15.3.1 InputStream和Reader 670

- 15.3.2 OutputStream和Writer 672
- 15.4 输入/输出流体系 673
 - 15.4.1 处理流的用法 674
 - 15.4.2 输入/输出流体系 674
 - 15.4.3 转换流 677
 - 怎么没有把字符流转换成字节流的转换流呢? 677
 - 15.4.4 推回输入流 678
- 15.5 重定向标准输入/输出 679
- 15.6 Java虚拟机读写其他进程的数据 680
- 15.7 RandomAccessFile 682
- 15.8 对象序列化 686
 - 15.8.1 序列化的含义和意义 686
 - 15.8.2 使用对象流实现序列化 686
 - 15.8.3 对象引用的序列化 688
 - 15.8.4 自定义序列化 692
 - 15.8.5 另一种自定义序列化机制 696
 - 15.8.6 版本 698
- 15.9 NIO 699
 - 15.9.1 Java新IO概述 699
 - 15.9.2 使用Buffer 699
 - 15.9.3 使用Channel 702
 - 15.9.4 字符集和Charset 705
 - 二进制序列与字符之间如何对应呢? 706
 - 15.9.5 文件锁 707
- 15.10 Java 7的NIO.2 709
 - 15.10.1 Path、Paths和Files核心API 709
 - 15.10.2 使用FileVisitor遍历文件和目录 710
 - 15.10.3 使用WatchService监控文件变化 711
 - 15.10.4 访问文件属性 712
- 15.11 本章小结 714
- 本章练习 714
- 第16章 多线程 715
 - 16.1 线程概述 716
 - 16.1.1 线程和进程 716
 - 16.1.2 多线程的优势 717
 - 16.2 线程的创建和启动 718
 - 16.2.1 继承Thread类创建线程类 718
 - 16.2.2 实现Runnable接口创建线程类 719
 - 16.2.3 使用Callable和Future创建线程 720
 - 16.2.4 创建线程的三种方式对比 722
 - 16.3 线程的生命周期 722
 - 16.3.1 新建和就绪状态 722
 - 16.3.2 运行和阻塞状态 724
 - 16.3.3 线程死亡 725
 - 16.4 控制线程 726
 - 16.4.1 join线程 726
 - 16.4.2 后台线程 727
 - 16.4.3 线程睡眠: sleep 728
 - 16.4.4 线程让步: yield 729
 - 16.4.5 改变线程优先级 730
 - 16.5 线程同步 731
 - 16.5.1 线程安全问题 731
 - 16.5.2 同步代码块 733
 - 16.5.3 同步方法 735

- 16.5.4 释放同步监视器的锁定 737
- 16.5.5 同步锁 (Lock) 737
- 16.5.6 死锁 739
- 16.6 线程通信 741
 - 16.6.1 传统的线程通信 741
 - 16.6.2 使用Condition控制线程通信 744
 - 16.6.3 使用阻塞队列 (BlockingQueue) 控制线程通信 746
- 16.7 线程组和未处理的异常 749
- 16.8 线程池 752
 - 16.8.1 Java 8改进的线程池 752
 - 16.8.2 Java 8增强的ForkJoinPool 754
- 16.9 线程相关类 757
 - 16.9.1 ThreadLocal类 757
 - 16.9.2 包装线程不安全的集合 759
 - 16.9.3 线程安全的集合类 759
- 16.10 本章小结 760
- 第17章 网络编程 761
 - 17.1 网络编程的基础知识 762
 - 17.1.1 网络基础知识 762
 - 17.1.2 IP地址和端口号 763
 - 17.2 Java的基本网络支持 764
 - 17.2.1 使用InetAddress 764
 - 17.2.2 使用URLDecoder和URLEncoder 765
 - 17.2.3 URL、URLConnection和URLPermission 766
 - 17.3 基于TCP协议的网络编程 772
 - 17.3.1 TCP协议基础 772
 - 17.3.2 使用ServerSocket创建TCP服务器端 773
 - 17.3.3 使用Socket进行通信 773
 - 17.3.4 加入多线程 776
 - 17.3.5 记录用户信息 778
 - 17.3.6 半关闭的Socket 785
 - 17.3.7 使用NIO实现非阻塞Socket通信 786
 - 17.3.8 使用Java 7的AIO实现非阻塞通信 792
 - 上面程序中好像没用到④⑤号代码的get()方法的返回值，这两个地方不调用get()方法行吗？ 795
 - 17.4 基于UDP协议的网络编程 798
 - 17.4.1 UDP协议基础 799
 - 17.4.2 使用DatagramSocket发送、接收数据 799
 - 17.4.3 使用MulticastSocket实现多点广播 803
 - 17.5 使用代理服务器 813
 - 17.5.1 直接使用Proxy创建连接 813
 - 17.5.2 使用ProxySelector自动选择代理服务器 814
 - 17.6 本章小结 817
- 本章练习 817
- 第18章 类加载机制与反射 818
 - 18.1 类的加载、连接和初始化 819
 - 18.1.1 JVM和类 819
 - 18.1.2 类的加载 820
 - 18.1.3 类的连接 821
 - 18.1.4 类的初始化 821
 - 18.1.5 类初始化的时机 822
 - 18.2 类加载器 823
 - 18.2.1 类加载器简介 823
 - 18.2.2 类加载机制 824

- 18.2.3 创建并使用自定义的类加载器 826
- 18.2.4 URLClassLoader类 829
- 18.3 通过反射查看类信息 830
 - 18.3.1 获得Class对象 830
 - 18.3.2 从Class中获取信息 831
 - 18.3.3 Java 8新增的方法参数反射 835
- 18.4 使用反射生成并操作对象 836
 - 18.4.1 创建对象 836
 - 18.4.2 调用方法 838
 - 18.4.3 访问成员变量值 840
 - 18.4.4 操作数组 841
- 18.5 使用反射生成JDK动态代理 842
 - 18.5.1 使用Proxy和InvocationHandler创建动态代理 843
 - 18.5.2 动态代理和AOP 844
- 18.6 反射和泛型 848
 - 18.6.1 泛型和Class类 848
 - 18.6.2 使用反射来获取泛型信息 850
- 18.7 本章小结 851
- 本章练习 851
- • • • • [\(收起\)](#)

[疯狂Java讲义_下载链接1](#)

标签

Java

编程

入门

计算机

java

编程语言

李刚

基础

评论

此书是知识点的堆砌，比较枯燥，行文啰嗦，正文中的描述和实例代码有不同之处。不是入门好书。

没有看完，这真的是教材不是字典？【2016年3月】

挺好的一本入门书，说得很透彻，既覆盖基本知识点，又有进阶层面的讲解

知识脉络清晰，适合新手入门。学习一门技术就是来回倒腾，倒腾地多了就上手了。书有点厚，八百多页，把书拆掉把，按照章节来服用。界面设计那两张可以跳过了，现在做PC单机软件的意义不大了。时间用在后面的JavaEE更有价值。

大二时看的，其实蛮后悔把这本书作为入门书的。好厚啊！差点就没坚持下去。不过还是很感谢刚大，现在已经是一名后端准入职员工啦！未来，等我！

实体书除了中间两章关于图形界面部分没有看以外（本想看完了再发），其余都看完了（仅仅是看完了）。知识点全面也很细，文字中可以感觉到作者鲜明的个性，感觉不像是在看一本书。但是如果需要一种更为细腻理性的思维，我浏览过的《Java 8编程官方教程》感觉还要好一点。如果时间思维能力足够，不管是Java还是Android开发，对于基础的积累这本书都推荐。而且对于Android开发，中间那两章能够理解的话可以对图形界面开发的思维有很大的启发作用。网上有很多人推荐《Thinking in Java》，我觉得不如前两本书好，可能更适合搞了多年开发的人看。

过于厚重的API手册，必须按章节劈开，背着小份去上课。

实用性很强，囫圇吞枣读了一遍，大致有点印象，具体用到了还是要来翻书

读没用，要实战练习才行。

一本字典，个人觉得不应该读这种书

作为一本“砖书”，自然是当字典用。如果零基础用来自学，还是找个教程来按一定顺序详略跳读更好。没读过同类其他书，推荐度不好说。不过现在看当时在目录上标记的必看内容还是很靠谱的，虽然很多到现在也没看过~

<http://www.cnblogs.com/liushengchieh/p/7892538.html>

作为Java入门书籍，还是有阅读的价值。Java知识点被作者解释得比较形象而细致，但缺点就是语言啰嗦。

我的java入门书，讲的很详细，只是有点啰嗦。

较为通俗易懂，适合入门

中国人写的java书就是本字典，拿一堆API嚼碎了给你吃，我还不如查文档。

入门书，简要看过。大量的代码堆积，有点敷衍。建议直接看200页左右的书入门，然后java核心思想详细看。

虽然内容很多像是字典，但是通篇看完还是有收获的，每个点都介绍的比较详细也通俗

易懂，讲的东西虽然基础但也需要多多复习

乱七八糟

读书笔记: <https://www.jianshu.com/p/95c241bb2184>

[疯狂Java讲义_下载链接1](#)

书评

如果说和国内其他一些高校老师们写的教材比，这本书好太多了。思路还是比较清楚的，讲的也比较全面，程序写的也不少。我从中受益不少，这100块钱还是花的值的。当然，这本书真的只是入门。

读完以后总觉得少了点什么。我敢说里面的内容肯定有抄的，或是参考的，如果想突破国内...

之前看了一半的Java编程思想,一边实现代码一边理解"编程思想",总有一种遥遥无期的感觉.尝试着用速度的方式来看这本书总体上还是感觉很轻松的.现在深刻的体会到看一本书(无论是技术的还是研究其他的知识的)比理解书中的意思更重要的事情是完完整整的看完它!就像之前看"Java编程...

工作一年多，做了许多东西，也解决过很多难题，但是一直发现自己的理论知识不扎实，当读到该书前两章的时候，我为我的java应用知识找到了归宿，并且学习到了更多的要注意的细节问题，使我对自己的提升充满了信心！！

大家都说这是一本好书，但本来已经有买其他书所以就不想买。不过后来上过作者开过的基础班的课，虽然不长，而且之前已经基本自学完基础知识，不过仍然受益菲浅，不得不说老师讲得相当好，被纠正了无数次，又被点醒了无数次。。。所以买了这本书。

这本书到底好在哪里，我还不知...

只要你真正想学习Java，你翻开书看上十多分钟，你绝对会被这本书吸引，介绍操作的部分，非常具体。介绍原理时候，则讲解得十分深入，尤其是5章、6章的知识，我相信很少有人真正对Java面向对象特性掌握得如此细致，如此地深入，即时我用Java已经有4年多了，读起来依然让我发现...

这本书最大的问题就是习题，对于初学者来说，没有习题的跟进是很难掌握每一个细小的知识点的，作者归纳的知识点非常多，也很细，但本人读下来觉得较为空泛，加上没有习题的跟进总是觉得云里雾里的，目前读到第六章还算有所收获，但学习java之路只是刚刚入门，未来还有很多问题...

非常好的一本书！有些人说这本书不深奥，没看点，但是其实真正要学好一门编程语言，你需要的就是扎实的基础知识，一切优秀的算法，优秀的程序都是建立在程序员对基础的理解上的，这本书通俗易懂，JAVA入门力荐！我觉得即使已经学习过java，这本书也值得一看，因为并不是每一...

看一遍知识点还挺多的，不适合编程新手看，我自己有c++基础，看得还行，但发现作者整本书里对类初始化块、类变量定义默认值的初始化执行顺序理解都是错的，不知道最新版的改了没，总结一句话，知识点多，需要谨慎阅读。

现在正在读这本书，自我感觉疯狂系列的书还不错，我觉得最大的优点是知识点全，java中的知识点几乎都涉及到了，但是有个缺点时作者没有更深入的引导你怎么去开发自己的代码，推荐给java初学者学习可以当字典使用，遇到不懂的知识点可以查查。

所谓大道至简。有必要把基础知识这么长篇大论的讲出来么？书的意义在于引导和启发

