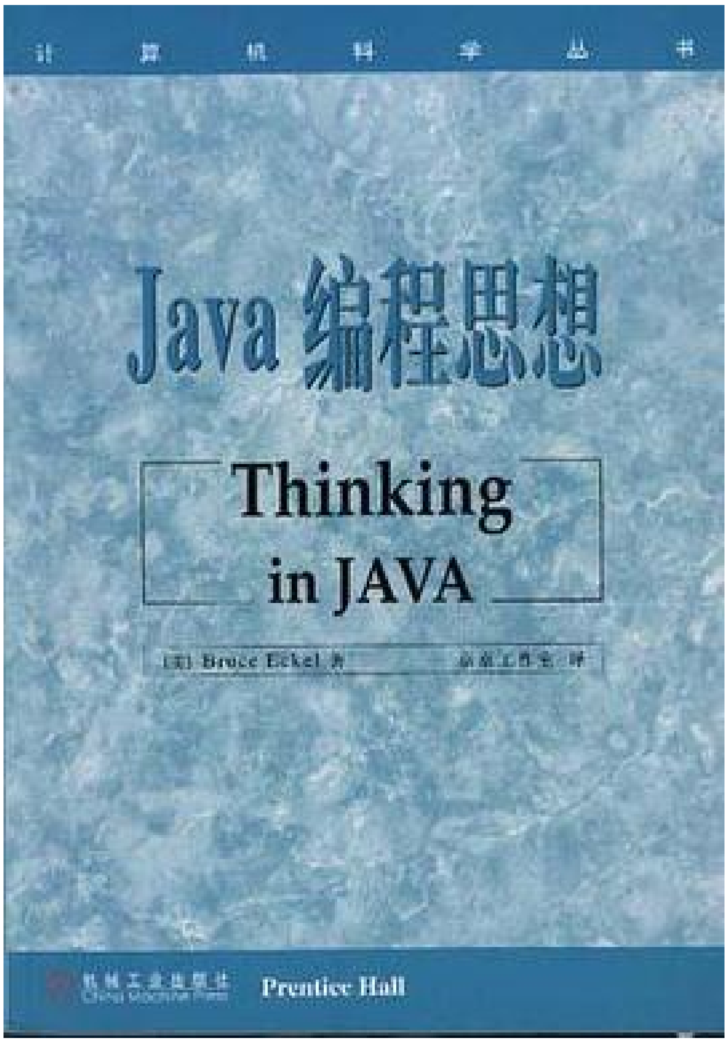
Java编程思想



Java编程思想_下载链接1_

著者:[美] Bruce Eckel

出版者:电子工业出版社

出版时间:2011-6-1

装帧:

isbn:9787121135217

《java编程思想(第4版)(评注版)》作者拥有多年教学经验，对c、c++以及java语言都有

独到、深入的见解，书中以通俗易懂且小而直接的示例阐释了一个个晦涩抽象的概念，是一本当之无愧的经典之作。本评注版讲解了java设计、语法和库的各个方面，包括java的运算符、控制逻辑、构造、回收、重用、接口、内部类、存储、异常、字符串、类型、泛型、数组、容器、i/o、注释、并发等内容。

对于国外技术图书，选择翻译版还是影印版，常常让人陷入两难的境地。本评注版力邀国内资深专家执笔，在英文原著基础上增加中文点评与注释，旨在融合二者之长，既保留经典的原创文字与味道，又以先行者的学研心得与实践感悟，对读者阅读与学习加以点拨、指明捷径。

经过评注的版本，更值得反复阅读与体会。希望这《java编程思想(第4版)(评注版)》能够帮助您跨越java的重重险阻，领略高处才有的壮美风光，做一个成功而快乐的java程序员。

作者介绍:

Bruce
Echel是MindView公司的CEO。该公司向客户提供软件咨询和培训。他是《Thinking in C++》一书的作者，并与他人合著了该书的第2卷以及其他图书。20多年来，他已发表了150多篇论文，并在全世界参与教学讲座和研讨，他是C++标准委员会成员，拥有应用物理大学学位和计算机工程硕士学位。

刘中兵：Java研究室首席技术专家
，应用数学专业，曾任职于清华同方、NEC等企业，长期深入Java／Java
EE大型企业应用的架构、设计与开发工作，目前专注于手机无线互联网与网络通信领域的架构设计与研究工作。

目录: 《java编程思想(第4版)(评注版)》
第1部分 基本语法
operators（新增批注30条） 1
simpler print statements 1
using java operators 2
precedence 2
assignment 3
mathematical operators 4
unary minus and plus operators 6
auto increment and decrement 6
relational operators 7
testing object equivalence 7
logical operators 9
literals 10
exponential notation 11
bitwise operators 12
shift operators 13
ternary if-else operator 16
string operator + and += 17
common pitfalls when using
.operators 18
casting operators 18
truncation and rounding 19
promotion 20
java has no “sizeof ” 20

summary 20
controlling execution
（新增批注21条） 21
true and false 21
if-else 21
iteration 22
do-while 23
for 23
the comma operator 24
foreach syntax 25
return 27
break and continue 27
the infamous “goto” 29
switch 32
summary 34
第2部分 面向对象
initialization & cleanup
（新增批注55条） 35
guaranteed initialization with
the constructor 35
method overloading 37
distinguishing overloaded
methods 39
overloading with primitives 39
overloading on return values 42
default constructors 43
the this keyword 44
calling constructors from
constructors 46
the meaning of static 47
cleanup: finalization and
garbage collection 47
what is finalize() for? 48
you must perform cleanup 49
the termination condition 50
how a garbage collector works 51
member initialization 54
specifying initialization 55
constructor initialization 56
order of initialization 56
static data initialization 57
explicit static initialization 59
non-static instance initialization 61
array initialization 62
variable argument lists 65
enumerated types 70
summary 72
access control
（新增批注21条） 73
package: the library unit 74
code organization 75
creating unique package names 76
a custom tool library 79
java access specifiers 80

package access 80
public: interface access 81
private: you can’ t touch that! 82
protected: inheritance access 83
interface and implementation 85
class access 86
summary 87
reusing classes
（新增批注35条） 89
composition syntax 89
inheritance syntax 92
initializing the base class 94
delegation 96
combining composition and
inheritance 97
guaranteeing proper cleanup 99
name hiding 101
choosing composition vs.
inheritance 103
protected 104
upcasting 105
why “upcasting”? 106
composition vs. inheritance
revisited 106
the final keyword 107
final data 107
final methods 110
final classes 112
final caution 113
initialization and class
loading 113
initialization with inheritance 114
summary 115
interfaces
（新增批注16条） 117
abstract classes and methods 117
interfaces 120
“multiple inheritance” in java 123
extending an interface with
inheritance 125
name collisions when combining
interfaces 127
fields in interfaces 127
initializing fields in interfaces 128
interfaces and factories 129
summary 130
inner classes
（新增批注32条） 131
creating inner classes 131
the link to the outer class 133
using .this and .new 134
inner classes and upcasting 135
anonymous inner classes 137
factory method revisited 140

nested classes 142
classes inside interfaces 143
reaching outward from a
multiply nested class 145
why inner classes? 145
closures & callbacks 148
inheriting from inner classes 150
can inner classes be
overridden? 150
local inner classes 152
inner-class identifiers 153
summary 154
error handling with excep- tions（新增批注52条） 155
basic exceptions 155
exception arguments 156
catching an exception 157
the try block 157
exception handlers . 157
creating your own
exceptions 159
exceptions and logging 161
the exception specification 164
catching any exception 164
the stack trace 166
rethrowing an exception 167
exception chaining 169
standard java exceptions 172
special case: runtimeexception 172
performing cleanup with
finally 174
what’ s finally for? 175
using finally during return 177
pitfall: the lost exception 178
exception restrictions 180
constructors 182
exception matching 187
alternative approaches 188
passing exceptions to the console 189
summary 189
第3部分 数据存储
strings（新增批注53条） 191
immutable strings 191
overloading ‘+’ vs.
stringbuilder 192
unintended recursion 195
operations on strings 196
formatting output 199
printf() 199
system.out.format() 199
the formatter class 200
format specifiers 200
formatter conversions 202
string.format() 204
regular expressions 205

basics 206
creating regular expressions 208
quantifiers 210
pattern and matcher 211
split() 218
replace operations 218
reset() 220
regular expressions and java i/o 221
scanning input 222
scanner delimiters 224
scanning with regular
expressions 225
stringtokenizer 225
summary 226
arrays（新增批注36条） 227
why arrays are special 227
arrays are first-class objects 228
returning an array 231
multidimensional arrays 232
arrays and generics 235
creating test data 237
arrays.fill() 238
data generators 239
arrays utilities 243
copying an array 243
comparing arrays 244
array element comparisons 245
sorting an array 248
searching a sorted array 249
summary 251
holding your objects
（新增批注35条） 253
generics and type-safe
containers 254
basic concepts 256
adding groups of elements 258
printing containers 259
list 261
iterator 263
listiterator 266
linkedlist 267
stack 268
set 270
map 273
queue 276
priorityqueue 277
collection vs. iterator 279
foreach and iterators 281
the adapter method idiom 283
summary 286
containers in depth
（新增批注102条） 289
full container taxonomy 289
filling containers 290

a generator solution 291
map generators 292
collection functionality 294
optional operations 297
unsupported operations 298
list functionality 300
sets and storage order 302
sortedset 306
queues 307
priority queues 308
deques 309
understanding maps 310
performance 311
sortedmap 314
linkedhashmap 315
hashing and hash codes 316
understanding hashcode() 319
hashing for speed 321
overriding hashcode() 324
choosing an implementation 329
a performance test framework 330
choosing between lists 333
microbenchmarking dangers 338
choosing between sets 339
choosing between maps 341
utilities 344
sorting and searching lists 347
making a collection or map
unmodifiable 349
synchronizing a collection or
map 350
holding references 351
the weakhashmap 353
java 1.0/1.1 containers 355
vector & enumeration 355
hashtable 356
stack 356
bitset 357
summary 359
第4部分 核心功能
i/o（新增批注89条） 361
the file class 361
a directory lister 361
directory utilities 364
checking for and creating
directories 369
input and output 370
types of inputstream 371
types of outputstream 372
adding attributes and useful
interfaces 373
reading from an inputstream
with filterinputstream 374
writing to an outputstream

with filteroutputstream 375
readers & writers 376
sources and sinks of data 377
modifying stream behavior 377
unchanged classes 378
off by itself:
randomaccessfile 379
typical uses of i/o streams 379
buffered input file 379
input from memory 380
formatted memory input 381
basic file output 382
storing and recovering data 383
reading and writing
random-access files 385
piped streams 386
file reading &
writing utilities 386
reading binary files 389
standard i/o 389
reading from standard input 389
changing system.out to a
printwriter 390
redirecting standard i/o 391
process control 391
new i/o 393
converting data 396
fetching primitives 398
view buffers 399
data manipulation with buffers 403
buffer details 404
memory-mapped files 406
file locking 409
compression 411
simple compression with gzip 412
multifile storage with zip 413
java archives (jars) 415
object serialization 416
finding the class 419
controlling serialization 420
using persistence 427
xml 432
summary 434
concurrency
（新增批注117条） 435
basic threading 435
defining tasks 435
the thread class 436
using executors 438
producing return values from
tasks 440
sleeping 442
priority 443
yielding 444

daemon threads 445
coding variations 446
joining a thread 450
catching exceptions 451
sharing resources 454
resolving shared resource
contention 454
atomicity and volatility 457
atomic classes 460
critical sections 462
synchronizing on other objects 462
thread local storage 463
terminating tasks 464
the ornamental garden 465
terminating when blocked 467
interruption 469
cooperation between tasks 475
wait() and notifyall() 475
notify() vs. notifyall() 479
producers and consumers 482
producer-consumers and queues 486
using pipes for i/o between tasks 491
deadlock 492
new library components 497
countdownlatch 497
cyclicbarrier 499
delayqueue 501
priorityblockingqueue 503
the greenhouse controller with
scheduledexecutor 505
semaphore 508
exchanger 511
simulation 513
bank teller simulation 513
the restaurant simulation 517
distributing work 521
performance tuning 526
comparing mutex technologies 526
lock-free containers 532
readwritelocks 533
active objects 535
summary 537
第5部分 高级特性
type information
（新增批注59条） 539
the need for rtti 539
the class object 541
class literals 545
generic class references 547
new cast syntax 549
checking before a cast 550
using class literals 555
a dynamic instanceof 557
counting recursively 558

registered factories 559
instanceof vs. class
equivalence 562
reflection: runtime class
information 563
a class method extractor 564
dynamic proxies 566
null objects 570
mock objects & stubs 575
interfaces and type
information 576
summary 580
generics
（新增批注126条） 583
comparison with c++ 584
simple generics 584
a tuple library 586
a stack class 588
randomlist 589
generic interfaces 590
generic methods 593
leveraging type argument
inference 594
varargs and generic methods 596
a generic method to use with
generators 596
a general-purpose generator 597
simplifying tuple use 598
a set utility 600
anonymous inner classes 603
building complex models 604
the mystery of erasure 606
the c++ approach 607
migration compatibility 609
the problem with erasure 611
the action at the boundaries 612
compensating for erasure 615
creating instances of types 616
arrays of generics 618
bounds 622
wildcards 625
how smart is the compiler? 628
contravariance 629
unbounded wildcards 632
capture conversion 636
issues 637
no primitives as type parameters 637
implementing parameterized
interfaces 639
casting and warnings 640
overloading 641
base class hijacks an interface 642
self-bounded types 642
curiously-recurring generics 643

self-bounding 644
argument covariance 646
dynamic type safety 649
exceptions 650
mixins 651
mixins in c++ 651
mixing with interfaces 653
using the decorator pattern 654
mixins with dynamic proxies 655
latent typing 657
compensating for the lack of
latent typing 660
reflection 661
applying a method to a sequence 662
when you don’ t happen
to have the right interface 664
simulating latent typing
with adapters 665
using function objects as
strategies 668
summary: is casting really
so bad? 672
enumerated types
（新增批注55条） 675
basic enum features 675
using static imports with enums 676
adding methods to an enum 677
overriding enum methods 678
enums in switch statements 678
the mystery of values() 679
implements, not inherits 681
random selection 682
using interfaces for
organization 683
using enumset instead of
flags 686
using enummap 688
constant-specific methods 689
chain of responsibility with
enums 692
state machines with enums 695
multiple dispatching 700
dispatching with enums 702
using constant-specific methods 704
dispatching with enummaps 705
using a 2-d array 706
summary 707
annotations
（新增批注51条） 709
basic syntax 710
defining annotations 710
meta-annotations 712
writing annotation processors 712
annotation elements 713

default value constraints 713
generating external files 714
annotations don’ t
support inheritance 717
implementing the processor 717
using apt to process
annotations 719
using the visitor pattern with
apt 723
annotation-based unit testing 726
using @unit with generics 733
no “suites” necessary 735
implementing @unit 735
removing test code 741
sum
· · · · · · (收起)

Java编程思想_下载链接1_

标签

java

Java

程序设计

计算机

经典

Programming

科技

英文

评论

这本书编排不太好 阉割就是傻逼 时隔4年 终于把这本书囫囵吞枣看完了
现在已经转行的我想说 可乐喝多了 这本书都很赛博朋克！

原书5星。这本实在没法看。
随便举个例子，看到第37页关于override，overload的评注我就吓尿了。。刘同学您java基础差也就算了，说什么overload必须无参我就原谅你了，可是旁边原书赫然给出的代码示例print那么多行的XXX
overloaded您愣是假装没看到吗？还说这叫overwrite。。再说overwrite是个神马东西？ 还有说overload取决于参数表我也笑了，您知道什么叫signature么，throws exception算在签名里不知道么，谁说光是“参数的个数和类型”就决定了是否能overload？最基本的概念弄不清就出书，真是有这么浮躁么？

实例坑爹...唯一的优点就是纸张比较大...评注不如不加,有很多删节未告知...最坑爹的是居然把polymorphic那一章完全删除了....你知道那个Glyph的构造器的例子有多经典吗

适合想要提高Java基础（内力）的程序员们，需具备一定的英文阅读能力

Java编程思想_下载链接1_

书评

文：@yuguo //08年9月26日更新
终于决定为这本心目中五星的书籍写一篇评论，因为目前的普遍舆论对这本书有一些误读，这些误读可能会误导初学者。我只想尽量客观的提出自己的观点，也许这不太可能，因为我认为这本书是五星的，那么多少我会作者持有偏好。那么请花一些时间阅读...

如题。。 先分享三句话。
“《Java编程思想》这本书很好，但还不适合你们现在去读，在合适的时候做合适的事，OK？”一心想速成的我并没有把教学视频中的这句小插曲放在心上，只是依稀记住了这本书。。然后快乐地学着速成JAVA。

“你学这些框架，做这种项目有什么用？你知道大...

总的来说，Java编程思想是一本好书；但是因为译者可能不懂计算机，很多地方都有严重错误。
之前和朋友抱怨过，朋友提议抱怨无用不如干点实事。遂决定边看边将自己找到的翻译错误贴出来，希望能给别人一些帮助。如果有错误之处，欢迎指正。 第15章 泛型
1.P352第二段：原文“但...

20190118 更新一下：
此书评写于将近8年前，回头再看这篇书评，深感羞愧。也正如很多评论所指出的，了解底层机制在程序开发尤其是大型系统的开发上至关重要。同时，多动手、快速动手也是成为一名高效率软件工程师的必经之路。
如果我能回到8年前，我一定会对那时的自己说：先不...

我it行业工作10年，这本书我读了8遍，对于那些一遍都读不下来的，一张口要8K-15K的程序员来说，你真的认为你值那么多，坑爹呢吧 Update：
以上是过去的一些评论，那时的认知还存在偏差，而且有些激进。
其实，读书是为了超越自己，而非凌驾于某人之上，更没有资格对任何人、...

很多人学习Java是从《Thinking in Java》这本书入手的，但是我认为这本书是不适合初学者的。我认为正确的使用这本书的方法应该是作为辅助的读物。《Thinking in Java》并不是在完整的介绍Java的整个体系，而是一种跳跃式的写作方法，是一种类似tips的方法来对Java很多知识点进...

Hello，大家好，如果是作为一个学习Java语言的新手，我强烈推荐Java官方手册：[https://docs.oracle.com/javase/tutorial/tutorialLearningPaths.html。]
我发现当我粗略看完Java编程思想的时候，我发现一个尤其大的特点就是作者所举的例子非常长，让我没有耐心继续看代码，我...

有幸得到一本第四版的，虽然以前看过第三版的，总觉得翻译不佳，而且看起来比较难懂，尤其是输入输出系统的时候完全找不到感觉，几个流就把我弄得云里雾里。25号得到第四版，从开始看到现在看到初始化与清理，感觉条理相当的清晰，内容理解的也比以前深入了 ，但是还是有些地方...

大三的时候，我刚学java，被这本书深深的吸引，不得不说作者在教学上的造诣之高，远大于这本书所传授的知识本身，而在于一种写作的风格上面，确实是很多技术书籍可以借鉴的。
大四的时候，我重温了一下，发现其实我自己还有好多东西遗漏或者遗忘，但是发现这本书相对于《java ...

搞JAVA不多，但是因为很多好书都以JAVA为实例代码，所以也自学的JAVA，这本书我利用十一的假期，完整的看了一遍，虽然都很基础，但是对内力有绝对的提升！
还记得那时我在使用Delphi，去面试C#的职位，之前从来没看过.NET/C#，就因为事先看过这本书，居然笔试答的很好，面试也...

《Thinking in Java》不是“那么好”的一本书，至少与C语言的《the c programming language》相比，与《advanced programing in unix
environment》相比，它在有条理的全面和漂亮的简洁这两头都没有做到优秀。不知道是不是介绍Java的书里没有一本能达到《the c programming lan...

第四版原书869页，讲并发的：“As another example, suppose you have a number of threads running tasks that use the file system.”
而翻译版的书中658页翻译为：“作为另一个示例，假设你有大量的线程，那它们运行的任务将使用文件系统。”

目前还在读,但就前两章就不适合初学者
-- 综述:
有几万行代码的程序员读起来一定会很爽,有种时时被点醒的感觉
初学者读起来目测要跪(当然天姿比较高的同学不在其列~) --------------------------...

翻译的大哥，继承或者子类这种用语请不要翻译成“导出”。我看了“导出”这个词我就火大，这也太低级了。

可以说我入门Java的时候，就选了这么一本书，虽然被大家称作传世之作，可是对于没

有一点Java基础的人们来说，想要理解书中的一些语句还是很困难的，当然这本书将每个章节都罗列出来，进行深入剖析，然后这却成了新手继续Java之路的一道屏障。越看越迷糊，越迷糊就越没兴趣了，...

做为一本优秀java的入门教材，的确值得每一位java初学者一读。接口，内部类等基本概念讲的很清楚。观云最近在看第二遍，明白了不少东西。hoho～
我买的那本应该和斑竹示出的那本一样的，机械工业出版社出的小砖头。呵呵～
价钱方面的确是￥66.0，不过这本书国内好像有不同的...

It is a really good material for starting java programming. Especially if you are coming from C/C++. The text are easy to read, with adequate examples to illustrate author's idea and his programming practices. However, it maybe a little bit out-dated, si...

在Java的初学阶段，这本书的作用是让你对于Java的语法有个一定的认识，能够着手去开发一个比较拙劣的项目。因此，在一开始，我仅仅看了部分章节的基本内容。
有了一段时间的编程经验后，再次去读这本书，特别是学完《设计模式》，使得对于OO的理解，对于Java中...

关于nest class: 看完Callback.java，蛋痛的nest class华丽转身为hottest cat
java也能变成拥有闭包的小辣椒。 关于异常:
JAVA的异常机制很烂，至少现在我是这么认为的。
它不如Python的异常来的简单实用。
为了声明一个异常，程序员需要做很多throws声明，这直接影响到...

Java编程思想_下载链接1_