

《代码之髓》

图书基本信息

书名：《代码之髓》

13位ISBN编号：9787115361533

出版时间：2014-8

作者：[日] 西尾泰和

页数：236

译者：曾一鸣

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu111.com

《代码之髓》

内容概要

《代码之髓：编程语言核心概念》作者从编程语言设计的角度出发，围绕语言中共通或特有的核心概念，通过语言演变过程中的纵向比较和在多门语言中的横向比较，清晰地呈现了程序设计语言中函数、类型、作用域、类、继承等核心知识。本书旨在帮助读者更好地理解各种概念是因何而起，并在此基础上更好地判断为何使用、何时使用及怎样使用。同时，在阅读本书后，读者对今后不断出现的新概念的理解能力也将得到提升。

《代码之髓：编程语言核心概念》力求简明、通俗，注重可读性，可作为大学计算机科学和软件工程等专业程序设计语言概论教材、计算机等级考试的参考资料，也可作为软件开发人员的学习参考书。

作者简介

西尾 泰和 (Nishio Hirokazu)

24岁取得理学博士学位。2007年起在Cybozu实验室从事提高知识生产力的软件开发工作。曾担任“2011年全日本安全与程序设计实战集训”程序设计语言组组长。特别关注编程语言的多样性及发展。著作有《Python语言程序设计》《程序员应该了解的程序设计基础知识》《WEB+DB PRESS》(第60期特辑)等。

曾一鸣

2010年上海交通大学电子工程系研究生毕业，现就职于某国际独立软件开发商，从事软件售后支持工作。对面向对象程序设计、脚本语言及其在语音、图像等信号处理中的应用有着浓厚的兴趣。

书籍目录

第1章	
如何深入高效地学习语言	1
1.1 在比较中学习	2
语言不同，规则不同	2
c语言和ruby语言中的真假值	3
java语言中的真假值	3
1.2 在历史中学习	4
理解语言设计者的意图	4
应该学哪种语言，我们无从所知	4
学习适用于各种语言的知识	5
1.3 小结	6
第2章	
程序设计语言诞生史	7
2.1 程序设计语言诞生的历史	8
连接电缆	8
程序内置	9
fortran语言问世	10
2.2 程序设计语言产生的原因	11
懒惰：程序员的三大美德之一	11
语言们各有各的便捷	12
2.3 小结	13
第3章	
语法的诞生	15
3.1 什么是语法	16
运算符的优先顺序	16
语法是语言设计者制定的规则	17
3.2 栈机器和forth语言	17
计算的流程	18
如何表达计算顺序	18
现在仍然使用的栈机器	19
3.3 语法树和lisp语言	20
计算流	20
如何表达计算顺序	20
现在仍然使用的语法树	21
专栏 要确认理解是否正确，首先得表达出来	23
3.4 中缀表示法	24
语法分析器	24
规则的竞争	25
专栏 当你不知道该学习什么时	25
3.5 小结	26
第4章	
程序的流程控制	27
4.1 结构化程序设计的诞生	28
4.2 if语句诞生以前	28
为什么会有if语句	28
为什么会有if...else语句	30
4.3 while语句——让反复执行的if语句更简洁	33

使用while语句的表达方式	33
不使用while语句的表达方式	34
4.4 for语句——让数值渐增的while语句更简洁	35
使用for语句的表达方式	35
不使用for语句的表达方式	35
foreach——根据处理的对象来控制循环操作	36
4.5 小结	37
第5章	
函数	39
5.1 函数的作用	40
便于理解——如同一个组织	40
便于再利用——如同零部件	41
程序中再利用的特征	41
5.2 返回命令	42
函数的诞生	43
记录跳转目的地的专用内存	44
专栏 函数命名	45
栈	45
5.3 递归调用	47
嵌套结构体的高效处理	48
嵌套结构体的处理方法	48
5.4 小结	52
第6章	
错误处理	53
6.1 程序也会出错	54
6.2 如何传达错误	55
通过返回值传达出错信息	55
出错则跳转	58
6.3 将可能出错的代码括起来的语句结构	61
john goodenough 的观点	61
引入clu语言	62
引入c++语言	62
引入windows nt 3.1	63
6.4 出口只要一个	64
为什么引入finally	64
成对操作的无遗漏执行	64
6.5 何时抛出异常	68
函数调用时参数不足的情况	68
数组越界的情况	69
出错后就要立刻抛出异常	70
6.6 异常传递	71
异常传递的问题	71
java语言的检查型异常	71
检查型异常没有得到普及的原因	73
专栏 具体的知识和抽象的知识	73
专栏 学习讲求细嚼慢咽	74
6.7 小结	74
专栏 从需要的地方开始阅读	75
第7章	

名字和作用域	77
7.1 为什么要取名	78
怎样取名	79
名字冲突	80
如何避免冲突	80
7.2 作用域的演变	81
动态作用域	82
静态作用域	84
7.3 静态作用域是完美的吗	88
专栏 其他语言中的作用域	88
嵌套函数的问题	89
外部作用域的再绑定问题	91
7.4 小结	93
第8章	
类型	95
8.1 什么是类型	96
8.2 数值的on和off的表达方式	97
数位的发明	97
七段数码管显示器	98
算盘	99
8.3 一个数位上需要几盏灯泡	100
从十进制到二进制	100
八进制与十六进制	102
8.4 如何表达实数	103
定点数——小数点位置确定	103
浮点数——数值本身包含小数部分何处开始的信息	104
8.5 为什么会出现类型	107
没有类型带来的麻烦	107
早期的fortran语言中的类型	108
告诉处理器变量的类型	108
隐性类型转换	109
8.6 类型的各种展开	111
用户定义型和面向对象	112
作为功能的类型	112
总称型、泛型和模板	113
动态类型	116
类型推断	118
8.7 小结	122
专栏 先掌握概要再阅读细节	122
第9章	
容器和字符串	125
9.1 容器种类多样	126
9.2 为什么存在不同种类的容器	127
数组与链表	127
链表的长处与短处	130
专栏 大o表示法——简洁地表达计算时间和数据量之间的关系	131
语言的差异	132
9.3 字典、散列、关联数组	132
散列表	133

树	134
元素的读取时间	136
没有万能的容器	138
9.4 什么是字符	139
字符集和字符的编码方式	139
计算机诞生以前的编码	140
edsac的字符编码	142
ascii时代和ebcdic时代	142
日语的编码	144
shift_jis编码对程序的破坏	145
魔术注释符	147
unicode带来了统一	148
9.5 什么是字符串	150
带有长度信息的pascal语言字符串和不带这一信息的c语言字符串	150
1个字符为16比特的java语言字符串	153
python 3中引入的设计变更	153
ruby 1.9的挑战	154
9.6 小结	155
第10章	
并行处理	157
10.1 什么是并行处理	158
10.2 细分后再执行	158
10.3 交替的两种方法	159
协作式多任务模式——在合适的节点交替	159
抢占式多任务模式——一定时间后进行交替	160
10.4 如何避免竞态条件	160
竞态条件成立的三个条件	161
没有共享——进程和actor模型	162
不修改——const、val、immutable	164
不介入	164
10.5 锁的问题及对策	166
锁的问题	166
借助事务内存来解决	167
事务内存的历史	168
事务内存成功吗	169
10.6 小结	170
第11章	
对象与类	171
11.1 什么是面向对象	172
内涵因语言而异的面向对象	172
对象是现实世界的模型	174
什么是类	175
11.2 归集变量与函数建立模型的方法	175
11.3 方法1：模块、包	176
什么是模块、包	176
用perl语言的包设计对象	177
光有模块不够用	178
分开保存数据	179
向参数传递不同的散列	179

把初始化处理也放入包中	180
把散列和包绑定在一起	181
11.4 方法2：把函数也放入散列中	183
first class	183
把函数放入散列中	184
创建多个计数器	185
把共享的属性放入原型中	186
这就是面向对象吗	189
11.5 方法3：闭包	190
什么是闭包	190
为什么叫做闭包	191
11.6 方法4：类	191
霍尔设想的类	192
c++语言中的类	192
功能说明的作用	193
类的三大作用	193
11.7 小结	194
第12章	
继承与代码再利用	195
12.1 什么是继承	196
继承的不同实现策略	197
继承是把双刃剑	199
里氏置换原则	199
12.2 多重继承	201
一种事物在多个分类中	201
多重继承对于实现方式再利用非常便利	202
12.3 多重继承的问题——还是有冲突	203
解决方法1：禁止多重继承	205
解决方法2：按顺序进行搜索	207
解决方法3：混入式处理	211
解决方法4：trait	213
12.4 小结	216
专栏 从头开始逐章手抄	217

精彩短评

- 1、定位看不懂系列。然而书内容其实还行
 - 2、内容很基础，刚入门编程时可以看看。书名起得有点大了，书中有些地方挖得也不够深，而且并没有什么独到的见解。另外，书中有几处明显的翻译错误。
 - 3、应该都是挺基础的东西，不过有的内容还是得有多语言的体验才能看得明白。后面关于类的历史和观点挺有启发的。
 - 4、作为休闲读物来说还算不错。上下班地铁上很快就读完了
 - 5、难得不仅是会某个东西，而是把会的东西嚼烂了让牙口没那么好的人也能品尝到美食。可以看出作者是一个善于思考的人，写出了这么一本形象生动，对于非科班程序员和编程初学者很有帮助的书。
 - 6、感觉等我懂得多一点的时候可以再读一次
 - 7、对于各种程序设计语言的特性进行了较为精炼的描述，对于语言的特性、各种语言的设计都进行了大致的介绍，确实很能开阔视野~另一方面，有些理解确实加深了，比如委托的概念，之前在C#中学习感觉实在有些朦胧，本书用Java演示了一下，就清晰多了~
 - 8、到底还是纸版阅读体验更好
 - 再读读
 - 9、值得一看。
 - 10、这书名字虽然恶心，但是还真是不错
 - 11、巩固基础的一本书。
 - 12、作为只会C的辣鸡到后面的面向对象看得云里雾里，从此书开始立下flag，学编程的路坚持坚持再坚持
 - 13、编程语言的核心概念每天都在应用，却不知道其实也是经过进化而来的。了解这些概念的来历，不错。
 - 14、写的有点太浅了
 - 15、从历史发展取舍和多语音横向对比来讲述编程语言中的基本概念，可以在某个闲暇的下午快速浏览一遍。
 - 16、光看书名，还以为是讲如何写出牛逼代码的。不过内容主要讲了一门编程语言的各个考量点，对比了很多过去、现在的语言，讲解了为什么这样设计、不同语言它们是怎样设计的。科普版的编译原理，挺有意思的。200多页，当闲时读物挺不错的，值得一读。
 - 17、通过对多种语言之间的差别来阐述编程中的基本概念
 - 18、介绍一些基础的东西，之前确实没想过，很好
 - 19、适合初学者 完全没多少有价值的内容 有点太标题党了
 - 20、专注于编程语言的通识读本，适合对若干编程语言有基本了解的人阅读，编辑错误太多
 - 21、I love the book!
 - 22、这本书的定位有点奇怪，非常基础的编程科普，却又要求读者了解数门编程语。但是既然已经了解数门编程语言了为什么还要回去读这种基础科普呢？当然书还是不错，这是因为这个奇怪的位置似乎没有其他书可以作为衡量标准。
 - 23、语言基本概念的深度探索
 - 24、读书研究的好方法
 - 25、本书首先讲了如何深入高效地学习程序设计语言，探讨了程序设计语言是如何产生的。接着介绍了和程序设计语言相关的各种概念，讲概念时，作者不会以某特定语言为叙述前提，他会将几种语言作比较。“培养对不同语言都适用的理解能力，是非常重要的。”
- 我想的是，如果我上学的时候或者最初工作的时候读到这本书，会不会对编程产生浓烈兴趣？跟学校老师的“实用性”讲解风格不同，作者会讲到语言设计者为解决何种问题而创造了这种语言，或者某种功能，这对语言学习非常有帮助。感觉理解更透彻。
- 26、比起“语言”像是生动的“语言学”一样的科普读物，没吃过肉，先尝骨髓。
 - 27、简单介绍了很多编程语言的基本概念，里面对于同一概念多种语言的比较很有意思。

《代码之髓》

- 28、去年读这本书时，我只会c语言，看的一知半解。现在我会c python javascript , lisp也稍微了解了一点点，再来看这本书，感觉懂了很多。这本书很适合用来回顾、归纳知识用。
- 29、不错的入门书。真正的深入浅出，乍读觉得自己什么都会。认真读一读后，还是会有很多收获。作者最后的话很有道理，多关注What和Why而不是How。这个思想也贯穿本书，先提出程序语言设计中遇到的问题，再介绍各个语言的解决方案，如果能紧跟作者节奏去思考，一定会有不少收获。本书的目的绝不是简单介绍各个语言的语法特性，而正因如此，反而更让人容易理解各种语法特性。我不得不收回我说的第一句话了，作者刻意从简单的概念讲起，很容易让读者觉得没有干货。如果你真觉得它是入门书，要么是你的水平太高，要么就是还没有真正读懂。
- 30、通俗易懂、开阔眼界的好书。从中知道了 green thread、Mix-in 等概念。加深了我对 OOP 的理解。今后也应该关注日本科技作者的书籍了。
- 31、编程科普书籍，从语言的设计原理角度阐述
- 32、语言概念的科普文，挺不错的~
- 33、CS
- 34、最后两章对象与类、集成与代码再利用很多看不懂。不过总的来说学到不少。
- 35、写了这么久代码，还多时候都是知道怎么实现功能，对语言的设计的缘由并不了解。本书在深度上并没有太多展开，但是引出了许多有趣的话题。
- 36、programming
- 37、这本书还是有些过于简单了，很多问题讨论的并不深入。不过这本书的讨论形式还是让我有些收获。作者关于多种编程语言的比较，直指一些编程语言概念的根本目的，让人能抛弃一些理所当然的狭隘观点，例如关于面向对象，类，还有动态类型语言的讨论。另外这本书是在亚马逊上购买的，能在北美亚马逊上直接买到中文书还是蛮方便。
- 38、专注于讨论why。
- 39、额，太入门了...基本就是常识，应该给新手看。四星是给娓娓道来不装逼的
- 40、舒筋活血，意犹未尽
- 41、太浅了
- 42、科普的书
- 43、犹如读小说般品着，还是不错的。计算机程序设计原理的介绍
- 44、不如称为通用语言编程思想吧！很适合新手们作为入门书籍。
- 45、从宏观上了解编程的原理和设计的动机
- 46、比较简单，通识书~
- 47、定位是个问题，这些内容不适合初学者，而懂一点的又没啥学习价值
- 48、通识书，非常适合初学编程者学习语言时配合阅读。涉及具体要点不多，但多是微言大义
- 49、刷了一下午知乎，不过还是紧赶慢感把这本书看完了。这本书是讲编程语言设计的思路和原则的，可以和语言具体的实现(语法、程序风格等)互相印证。还是很不错的，值得一看。
- 50、虽然我在编程方面还是个辣鸡，还是觉得这本书很浅。。。

1、编程语言核心概念,这就是本书的原标题,我想代码之髓一定是中文编辑后来画蛇添足加上去的.编程语言本身已经走过了很长的发展里程,经过了摸索化,实践化,理论化,理论实践化等很多个阶段.在现在的时代,已经呈现出过度复杂化,过度概念化的倾向.如果我们不能够追根溯源,从历史里面把握住语言中的核心概念是如何出现的,各自解决了什么最根本的问题,那么我们会迷失在当下看起来一派繁荣的语言森林里面.本书就是做这个去粗取精,寻根究源的工作,让我们重新认识语言中这些如此重要的概念的缘起,有了这个根本性的认识,我们就会知道,哪些是根,哪些是枝子,哪些是花.而且特别难能可贵的一点是,作者一点也不故弄玄虚,从理论中来,但是并不用理论唬人,只用特别简单的,朴素的话来讲这些概念,特别好.以上.

2、从“编码”(コーディング)到“编程”(プログラミング),这个名词的小小改变,其实凝结了人类的许多智慧和心血。而这其中最重要的大发明,就是程序语言。人类擅长的是用语言交流,所以发明了各种各样的语言,这么一来人类就可以用自己熟悉的语言来编写程序,而不必直接去编写只有机器才能识别和执行的代码了。编程语言发展到今天,已经经历了好几代,从种类上看更是有数千之众。这些语言当然不可能都为着同样的目标而被设计出来,所以它们之间的差异还是非常大的。这些差异性体现在诸多方面,语法的差异非常显著,而相似的语法背后的语义差异则更是无形而且微妙。如果考虑到语言的运行环境、应用领域,还有语言之上建立的方法论,更是让人感觉:难道在程序语言这个完全由人类创造的领域,也要出现像自然语言一样的,把程序员分成无数宗派甚至族群的“巴别塔”了吗?其实不然,程序语言毕竟是工程学产物,它的发展其实是有着统一的计算模型作为背景的,因而可以视为一种“约束下的发展成果”。计算模型落实为实际计算机的体系结构,种类并没有很多。因此,建构在其上的程序语言,由于最终还是要落实为在这些体系结构之上执行的代码,结构和实现上不会完全失控。但话说回来,要写作一本关于程序语言的通识读本,能够在一本书里讲清楚程序语言的共性,并能够指导具体语言的学习,这件事还是很困难的。很容易就会写成建立在编译原理基础之上的大部头,这么一来对读者的要求就非常高了。还容易写成各种语言混在一起的大杂烩,把读者搞得晕头转向。更何况,如果真拿具体的语言来讲,现在的语言版本更新迭代速度很快,写成的书很快就过时了。但是本书就很不一样,它抓的点非常好,是一些通用但是不会过时的点。这些点从程序结构来说,包括了名字和作用域(哪种语言会没有名字和作用域呢?）、函数、类型、异常处理;在库的方面,讨论了字符串和容器;从运行环境来说,讨论了并行处理;从方法论的角度,讨论了至今仍然是主流的OOP和类继承。每个主题都是既深入浅出,不停留在概念层面,又点到即止,避免纠缠技术细节,显示了日本工业界务实而又认真的做事态度,以及不俗的技术实力和表达能力。我尤其喜欢第一章提出的两种程序语言的学习思路:在比较中学习,在历史中学习。从每种语言的独特语法结构中,我们能够体会到这种语言针对的特定问题领域的难点和特色,这也引导着我们提出新的问题,或者在不同领域的问题之间建立关联。而从那些从属于不同时代语言的代际比较,我们更可以同时体会到早期语言的强大和不足之处,以及新生语言做了哪些取舍(一定有舍,不可能只有改进,因为结果是一样的)来向哪些方面做了妥协。其实在学习自然语言的时候,这两个办法也是非常管用的,只是学习多种程序语言的人比学习多种自然语言的人要多得多而已。一开篇就开宗明义地提出语言学习方法,这也决定了本书的高度。总之,非常好的一部程序语言通识读本,向所有同行推荐。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:www.tushu111.com