

TURING

图灵程序设计丛书

〔日〕西尾泰和 著
Nishio Hirokazu

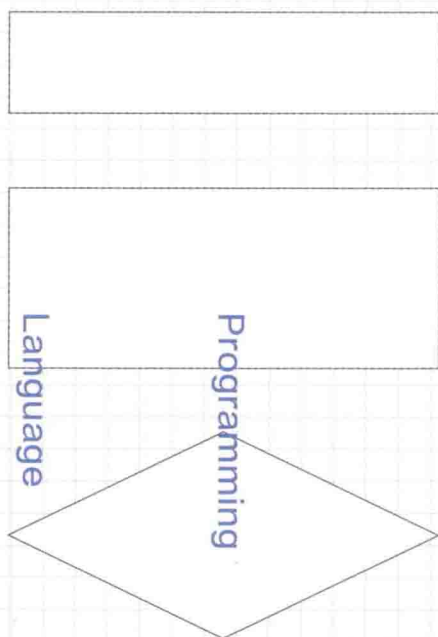
曾一鸣 译

Concepts



编程语言 核心概念

代码之髓



为什么编程语言里会有这些概念？

它们有什么用？应该何时使用？应该如何使用？

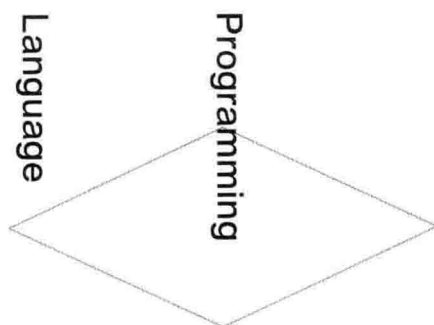
运算符 / 循环语句 / 函数 / 异常 / 作用域 / 动态类型 / 类型推断 / 数组 / 字典 / 线程 / 上锁 / 事务
内存 / 类 / 接口 / 继承 / 委托 / C3线性化 / Mix-in / Trait

人民邮电出版社
POSTS & TELECOM PRESS

代码之髓

编程语言

核心概念



TURING

图灵程序设计丛书

〔日〕西尾泰和
著

曾一鸣
译

Concepts

人民邮电出版社
北京

图书在版编目(CIP)数据

代码之髓: 编程语言核心概念 / (日) 西尾泰和著;

曾一鸣译. -- 北京: 人民邮电出版社, 2014.8

(图灵程序设计丛书)

ISBN 978-7-115-36153-0

I. ①代… II. ①西… ②曾… III. ①程序语言

IV. ①TP312

中国版本图书馆CIP数据核字(2014)第137918号

内 容 提 要

本书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。

本书力求简明、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。

◆ 著 [日] 西尾泰和

译 曾一鸣

责任编辑 徐 蹇

责任印制 焦志伟

◆ 人民邮电出版社出版发行

邮编 100164

电子邮件

网址 <http://www.ptpress.com.cn>

北京天宇星印刷厂印刷

◆ 开本: 880×1230 1/32

印张: 7.375

字数: 235千字

印数: 1-4 000册

北京市丰台区成寿寺路11号

315@ptpress.com.cn

2014年8月第1版

2014年8月北京第1次印刷

著作权合同登记号 图字: 01-2013-9190号

定价: 45.00元

读者服务热线: (010)51095186 转 600 印装质量热线: (010)81055316

反盗版热线: (010)81055315

广告经营许可证: 京崇工商广字第0021号



前言

当今程序设计语言多种多样，可供阅读的资料也非常多。但一个人的学习时间是有限的，全部都学并不现实。

另外，信息技术瞬息万变，特定语言及工具很快便已陈旧。如果不能意识到这一点而有选择性地学习一些相对稳定的知识，所学的内容将逐渐失去价值。

那么，该学习哪些知识并如何学习呢？笔者认为在学习中需要做到以下三点。

- 在比较中学习
- 在历史中学习
- 在实践中学习

第一条是指通过比较多种语言，总结出某种语言的独有特点，以及多种语言的共有特点。

第二条是指通过追溯语言的发展历史，了解语言是如何产生、变化和消失的，探寻语言发展演变的轨迹。

第三条是指亲自进行程序设计。边实践边思考如何编程，才能深入理解语言设计者的意图，同时也能发现自己原先理解不到位之处。

在阅读了各种程序设计书籍之后，相信读者们都曾产生过很多疑问。本书的目的就是解答大家的这些疑惑。本书假设读者对程序设计还不是很熟悉，侧重讲解“在比较中学习”和“在历史中学习”。如果大家在阅读本书后能掌握这些学习方法，那我将不胜欣喜。

致谢

本书在编写过程中受到多位同仁和朋友的大力支持。在此，请允许我省略敬称，对各位的帮助表示最诚挚的谢意（排名不分先后）：和田英一、井出真广、佐藤敏纪、长慎也、鹭见正人、石本敦夫、泷泽昭广、文殊堂（monjudoh）、小泉守义、笹田耕一、园田裕贵（Yugui）、庄司嘉织、尾崎智仁、水岛宏太、矢野勉、松原丰、cactusman、都元daisuke、小田切笃、竹迫良范、光成滋生、中谷秀洋、蓑轮太郎、中山心太、Pawel Marczewski、Harry Roberts。

此外，技术评论社的池田大树先生对本书的出版倾尽心血，在此一并表示感谢。

最后，我要感谢我的妻子，在她的协助下，我才有充分的时间投入到本书的编写当中。

2013年3月 西尾泰和

本书构成

本书共分为 12 章。

第 1 章围绕如何深入高效地学习语言，举例说明“在比较中学习”和“在历史中学习”两种学习方法。

第 2 章探讨程序设计语言是如何产生的。

从第 3 章开始将介绍和程序设计语言相关的各种概念。本书不以某特定语言为叙述前提，如果讲解的一些知识大家尚未接触到，理解起来可能有些困难。比如，只有 C 语言相关经验的读者可能对第 6 章错误处理的内容理解起来稍显吃力。如果没有使用过线程，就不太好理解第 10 章的并行处理。这时，大家可以先阅读其他章节。

第 3 章重点讨论程序设计语言中为什么有那么多的语法规则。着重介绍运算符的优先顺序，并比较规则相对较少的 FORTH 语言和 LISP 语言。

第 4 章通过比较不具有控制语句的汇编语言和具有控制语句的 C 语言，探讨 if、while、for 等控制语句产生的原因。

第 5 章讨论函数是如何产生的，并学习递归调用的使用场合。

第 6 章围绕现在许多语言中称为异常的错误处理机制，介绍这一机制的必要性以及它的发展过程。

第 7 章阐述变量与函数的名字产生的原因，并介绍作用域的必要性及其进化的过程。

第 8 章探讨类型存在的必要性。首先从数的表达方法讲起，接着介绍类型及其应用。

第 9 章会学习一种能存入多个元素的物体，即容器。容器种类多样，本章将解释为什么会有这么多种类、各种类型之间的差异，以及各自的优缺点。本章后半部分会学习字符串，在历史中学习字符编码，并在比较中学习不同语言中字符串的差异。

第 10 章讨论的是同时执行多个处理的并行处理中存在的问题以及它的规避策略，我们将在不同语言的比较中展开这部分内容。

第 11 章将学习面向对象。首先，通过比较 SmallTalk 语言和 C++ 语言，介绍面向对象这一术语指示的内容在不同语言中有哪些差异。接下来将阐述面向对象发明的原因。另外，本章还会介绍类和其他不同的对象创建机制。

第 12 章围绕继承，探讨不同语言中的继承机制及它们各自的优缺点。

示例代码下载

本书中的示例代码，以及在编写本书过程中供验证使用的代码，均可从笔者创建的支持网站下载。但因篇幅所限，部分代码没有包含在本书中。

<http://nhiro.org/langbook/>

本书是在弊社杂志《WEB+DB PRESS》Vol.66 特辑之《程序员应该知道的程序设计基础知识——熟知语言核心设计如有神助》（プログラマが知るべき言語設計の基礎知識——言語の核を知れば、自ずと作法が見えてくる）的基础上，大量添加内容并修正之后编纂而成。

本书展示的代码已在 Mac OS 10.7.5 的以下程序语言环境中运行并确认过。不同的执行环境可能带来操作顺序、显示画面及执行结果的差异。

- Scala: version 2.9.2 (Java HotSpot 64-Bit Server VM, Java 1.7.0_05)
- Haskell: GHCi, version 6.10.4
- Python: 2.7.3
- Ruby: 1.9.3p194 (2012-04-20 revision 35410) [x86_64-darwin11]
- Perl: v5.12.4 built for darwin-thread-multi-2level
- JavaScript: Node.js.v0.6.18
- Java: build 1.7.0_05-b05
- C/C++: gcc version 4.2.1 (Based on Apple Inc. build 5658) (LLVM build 2336.9.00)

对于使用本书信息所造成的后果，技术评论社、原书作者、人民邮电出版社以及译者，概不负责。

本书中出现的公司名称及商品名称，通常是各相关公司的商标或注册商标。本书在使用时已省略™、©、®等标记。

版 权 声 明

CODING WO SASAERU GIJUTSU by Hirokazu Nishio

Copyright © 2013 Hirokazu Nishio

All rights reserved.

Original Japanese edition published by Gijyutsu-Hyoron Co., Ltd., Tokyo

This Simplified Chinese language edition published by arrangement with Gijyutsu-Hyoron Co., Ltd., Tokyo in care of Tuttle-Mori Agency, Inc., Tokyo

本书中文简体字版由 Gijyutsu-Hyoron Co., Ltd. 授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

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
	FORTTRAN 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 语句的表达式	33
	不使用 while 语句的表达式	34
4.4	for 语句——让数值渐增的 while 语句更简洁	35
	使用 for 语句的表达式	35
	不使用 for 语句的表达式	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