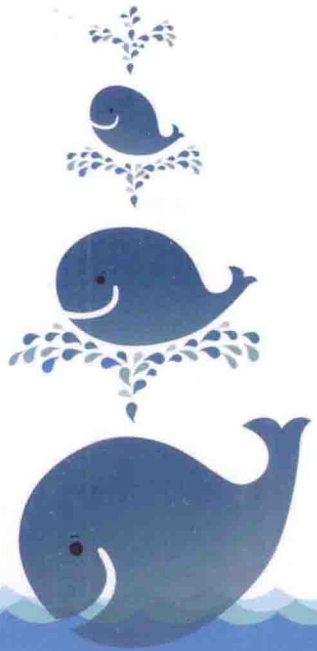
TURING 图灵程序设计丛书

两周自制脚本语言

【日】千叶滋 著 陈筱烟 译



从解释器到编译器
支持函数、数组、对象等高级功能

只需14天
从零开始设计和实现脚本语言

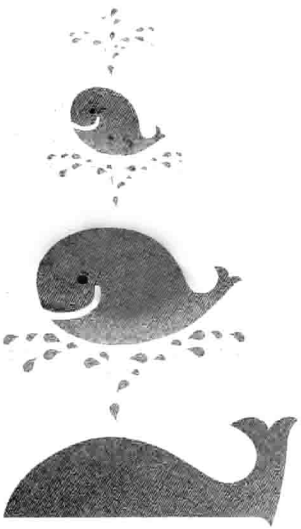
东京大学&东京工业大学教授执笔
日本编译器权威专家 **中田育男** 作序推荐

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

两周自制脚本语言

【日】千叶滋 著 陈筱烟 译



人民邮电出版社
北京

图书在版编目（CIP）数据

两周自制脚本语言 /（日）千叶滋著；陈筱烟译
--北京：人民邮电出版社，2014.6
（图灵程序设计丛书）
ISBN 978-7-115-35564-5

Ⅰ.①两… Ⅱ.①千… ②陈… Ⅲ.①JAVA语言－程序设计 Ⅳ.①TP312

中国版本图书馆CIP数据核字（2014）第093603号

内 容 提 要

本书是一本优秀的编译原理入门读物，全书穿插了大量轻松风趣的对话，读者可以随书中的人物一起从最简单的语言解释器开始，逐步添加新功能，最终完成一个支持函数、数组、对象等高级功能的语言编译器。本书与众不同的实现方式不仅大幅简化了语言处理器的复杂度，还有助于拓展读者的视野。

本书适合对编译原理及语言处理器设计有兴趣的读者以及正在学习相关课程的大中专院校学生。同时，已经学习过相关知识、有经验的开发者，也一定能从本书新颖的实现方式中受益良多。

◆ 著　[日]千叶 滋
译　陈筱烟
责任编辑　徐　骞
责任印制　焦志炜
◆ 人民邮电出版社出版发行　北京市丰台区成寿寺路11号
邮编　100164　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京天宇星印刷厂印刷
◆ 开本：800×1000　1/16
印张：18.75
字数：429千字　2014年6月第1版
印数：1-4 000册　2014年6月北京第1次印刷
著作权合同登记号　图字：01-2013-6220号

定价：59.00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

译者序

在大学时代，编译原理就是我十分感兴趣的一门课程。无论是手工进行语法分析计算，还是尝试设计一些简单的语言处理器，都给我留下了深刻的印象。为某些特殊用途的软件设计专用的程序设计语言，也是我一度着迷的课题。当时，阿尔弗雷德所著的《编译原理技术与工具》是自己包中的常客，我常带着英文原版辗转于教室、图书馆与自己的房间。

怀着对编译原理的这份兴趣与热忱，我一直都希望能做一些与之相关的工作。遇到这本《两周自制脚本语言》，算是一种缘分。

初见书名，我还有些犹豫。国内以速成为卖点的计算机书籍不少，真正值得一读的好书却不多。诱惑读者靠走捷径学到真知，常常最终使他们绕了弯路。不过在了解到作者是东京大学和东京工业大学计算机系的资深教授后，我又对这本书产生了好奇。一位仍活跃在科研与教学第一线的学者，会怎样在两周内教会读者设计一种脚本语言呢？

读完本书，我颇为惊喜，原本的担心消失殆尽。这是一本有趣而实用的书，内容编排十分独特，作为一本编译原理的入门读物，本书的很多编写思路都围绕这点展开。作者没有为了增添噱头而加入大量初学者不易理解也无需急着掌握的知识与技术，而是始终以够用为本，逐步扩展语言的语法规则，帮助读者从最基础的概念到一些常用的进阶设计理念，逐步掌握语言处理器的运行原理，以及设计一门新的语言的必要步骤。书中随处可见的老师与学生、学生与学生间的轻松对话是本书的一大特色，几位性格迥异的出场人物时而为读者解惑，时而提出一些更深层次的问题，引发读者的思考。

尽管书名是自制脚本语言，但本书的内容却是自制脚本语言处理器。作者花了大量篇幅讲解语言处理器的功能增强与性能优化。与同类书相比，本书使用了一种较为新颖的实现方式，能够有效简化语言处理器的设计与维护成本。尽管它还无法完全胜任实际生活中更为复杂的系统，这种解决问题的思路却对开拓读者的眼界很有帮助。

得益于作者丰富的教学科研经验，本书涉及了不少实践中可能遇到的问题。作者没有直接给出解答，而是引导读者思考，无论是初学者还是有一定基础知识的读者，都能在阅读本书后有新的发现。在翻译本书时，我也有所收获。其中，为了深究一些细节问题，我曾专门致信向作者请教。作者立刻对我的疑问进行了解答，并附上了细致的说明，在他的帮助下，中译本的质量得到了进一步提升。在此谨对作者的支持表示衷心的感谢。此外，中译本已经参照原书的勘误及补遗表做了修改与调整，一些细节问题得到了修正。

在翻译过程中，我得到了许多人的帮助与支持。家人为自己创造了能够安心翻译的环境，并始终给予理解与关心。好友陈洁也为我提供了莫大的支持，使我可以每天以最佳状态投入工作。这里还要感谢图灵的各位编辑提出大量极具价值的建议与意见，帮助本书顺利完成并最终问世。最后，希望对编译原理有兴趣的读者都能从本书中获益。

陈筱烟

2014 年 4 月于上海

前　言

本书是一本编译原理的入门读物。过去，大家普遍认为编译器与解释器之间存在很大的差异，因此会分别编写针对编译器与解释器的图书。不过，最近编译器与解释器之间的界限越来越模糊，我们只要稍微了解一下常见的程序设计语言，就会发现两者已不再是对立的概念。

因此，与其说本书是编译原理的入门书，不如说是语言处理器的入门读物更为恰当。语言处理器是用于执行程序设计语言的软件，它同时包含了编译器与解释器。本书看似用了大量篇幅讲解解释器的原理，其实是在讲解编译器与解释器通用的理论。第 1 章将详细介绍各章节的具体内容。

本书采用了 Java 语言来实现语言处理器。在设计语言处理器时，C 语言或 C++ 语言更为常见，加之本书没有借助 yacc 等常用的工具来生成语言处理器，因此读者也许会认为本书的实用性不足。

本书在介绍语言处理器的设计方式时，尽可能采用了较新颖的手段。C 语言或 C++ 语言结合 yacc 的方式性能较差，且是上世纪 80 年代的实现方式。在那之后，程序设计语言飞速发展，已不可同日而语，其运行性能也大幅提升。入门读物也应该与时俱进，讲解与过去不同的设计方式，展现它们的实践价值。

时至今日，软件领域的发展依然日新月异，并逐渐渗透至生活的方方面面，这一势头无疑将持续下去。在此期间，各类技术必将不断发展，为了跟上技术更新的步伐，软件应当以略微领先于时代的设计思路开发。

很久以前，笔者曾使用 C++ 语言开发过适用于工作站的语言处理器，当时，时钟频率仅有 100 兆赫，内存也不过几百兆字节。那套软件幸运地在各种环境下运行了十年以上。有一天，我收到了一封邮件。我记得好像是一个德国的年轻人，他洋洋洒洒写了很多，批评那套软件的设计有不少问题。还说开发者应当合理使用模板，并灵活运用各种库，要学习使用设计模式，还要用 XML 来表示抽象语法树，等等。

他指出我太节省内存，只顾着提升性能，结果程序难以阅读。从当时的主流软硬件标准来看，这些批评确实合情合理，但那套系统毕竟是十年前的产物。在当时软硬件性能孱弱的情况下，如果遵循他的建议，最终完成的语言处理器恐怕会被打上缺乏使用价值的标签（顺便一提，提出批评的那位年轻人虽然说了很多，却没有写一行代码）。

然而，从这件事中我深刻体会到，软件有着惊人的生命力，即使在开发时采用了最佳设计，最终还是会随着时代的进步而被迅速淘汰。因此，前文说软件应当以略微领先于时代的设计思路

开发有其合理性。当然，我们也可以不关心他人的批评，尽可能缩短软件的生命周期，并积极抛弃过时的内容。具体采用哪种策略因人而异。

希望读者能够在阅读本书时始终记住这些理念。读过本书之后，如果大家觉得收获良多，我将深感荣幸与喜悦。

2012 年 新春

千叶滋

推荐序

本书虽然是编译原理的入门读物，但除了编译器之外，还将介绍程序设计语言的各种功能及相应实现方法的基本设计思路。不过，与现有的很多编译原理入门书不同，本书的内容十分新颖。已有的同类书大多遵循一些固定套路，以正则表达式、自动机、LL 语法、LR 语法及相关的语法分析算法等基础知识为核心，设计简化的 C 语言风格编译器。本书不仅会仔细讲解这些知识点蕴含的基本思想，还会通过现成的库来实现语言处理的词法分析与语法分析逻辑。

本书仅简单讲解词法分析与语法分析等编译器的基本知识，而将重点放在语言处理器的实现上。已有的同类书很少涉及各类具体的语言功能与它们的具体实现方式，本书将由简入繁，逐步修改语言处理器，介绍这些功能与实现。语言处理器最初只支持无变量声明的简单表达式，之后陆续添加函数与闭包、数组、面向对象类型、类型推论等功能，将它从解释器修改为编译器。

本书采用 Java 语言来实现语言处理器，不过在多次修改后，已有的程序通常需要重写，这并非我们希望看到的。本书使用了笔者开发的语言处理工具 GluonJ，因此在添加功能时无需修改已有的代码，只需另外编写必要的程序即可。因此可以轻松更改不同功能的配置。这是一种非常理想的程序开发方式。

得益于这种方式，本书能通过若干较为简短的独立程序实现语言处理器的各种功能，并将完整代码收录于书中。这正是 GluonJ 的长处，如果合理设计程序结构，这种优势能进一步得到发挥，程序的扩展将更加容易。希望读者能够通过本书体会这种编程思想。

本书并非一味教授基础知识，而会尽可能简明地讲解这些基本概念背后的原理。此外，乍一看类与函数是完全不同的概念，其实类是函数概念的一种延伸，本书也会对此进行说明。正文中插入了大量学生与教师的对话，时而质疑时而反驳，提供了很多相关信息，引发读者深入思考。

本书是一本优秀的编译原理入门读物，它尝试以一种现代的方式设计一种现代的语言，即使读者对编译器已有一定程度的了解，也一定能从中学到很多。

中田育男

致谢

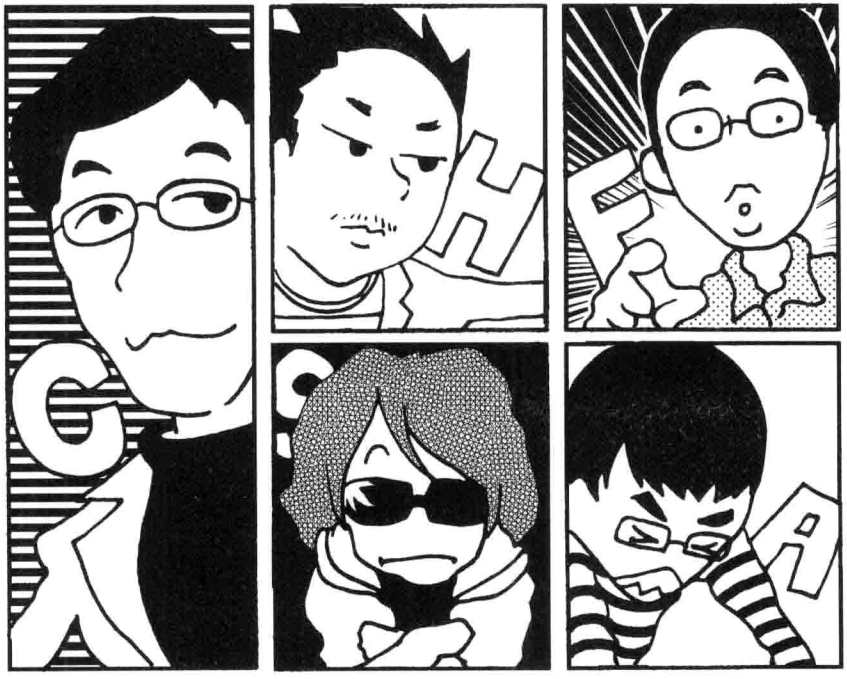
笔者在执笔期间得到了多方帮助。其中，我要特别要感谢五十岚淳、笹田耕一，以及以学生视角审读本书草稿的栗田昂裕。中田育男老师不仅审读了草稿，还特地为本书作序，在此谨深表谢意。此外，从本书策划阶段起，技术评论社的池本公平先生一直给予我诸多照顾，非常感谢。本书使用的软件与技术是笔者日常研究中积累的成果，我要感谢研究室的各位成员。最后，我要感谢爱妻典子与孩子们，正是有了他们的支持，本书才得以顺利完成。

本书的阅读方式

对话形式的补充说明在本书中随处可见。这些对话有时用于补充正文内容，有时会引入一些更深入的主题。

本书的对话中将出现以下 5 个角色。

- 出场角色



C 某大学的老师。程序设计语言研究室的负责人。

H 最年长的学生。彬彬有礼的运动型男生。

F 好为人师的学生。

S 博学的学生。平时少言寡语，一开口反而会语惊四座。

A 留过级，所谓的差生。不过他究竟是不是真的差生还是一个谜。

这些出场角色纯属虚构，与现实中存在的人物没有任何关系。希望读者能够结合对话与正文，更深入地理解本书的内容。

- **有效利用源代码**

在阅读本书时，强烈建议读者下载源代码并通过 Eclipse 等集成开发环境调试。如果不使用 Eclipse 之类的开发环境，用面向对象语言写成的程序将变得难以理解。

读者可以从以下地址下载源代码。

http://www.csg.ci.i.u-tokyo.ac.jp/~chiba/files/stone-2012feb21.zip

需要注意的是，在不同的程序设计环境中，源代码中的反斜杠 \ 可能会显示为 ¥ 等字符。本书将统一使用 \。

目录

第1部分 基础篇

第1天 来，我们一起做些什么吧 1

1.1 机器语言与汇编语言 2
1.2 解释器与编译器 3
1.3 开发语言处理器 5
1.4 语言处理器的结构与本书的框架 6

第2天 设计程序设计语言 10

2.1 麻雀虽小、五脏俱全的程序设计语言 11
2.2 句尾的分号 12
2.3 含糊不得的语言 14

第3天 分割单词 17

3.1 Token对象 18
3.2 通过正则表达式定义单词 19
3.3 借助java.util.regex设计词法分析器 22
3.4 词法分析器试运行 27

第4天 用于表示程序的对象 30

4.1 抽象语法树的定义 31
4.2 设计节点类 34
4.3 BNF 38
4.4 语法分析与抽象语法树 42

第5天 设计语法分析器 44

5.1 Stone语言的语法 45

5.2 使用解析器与组合子……46
5.3 由语法分析器生成的抽象语法树……53
5.4 测试语法分析器……59

第6天 通过解释器执行程序 62

6.1 eval方法与环境对象……63
6.2 各种类型的eval方法……65
6.3 关于GluonJ……69
6.4 执行程序……72

第7天 添加函数功能 75

7.1 扩充语法规则……76
7.2 作用域与生存周期……81
7.3 执行函数……83
7.4 计算斐波那契数……89
7.5 为闭包提供支持……90
7.6 实现闭包……92

第8天 关联Java语言 95

8.1 原生函数……96
8.2 编写使用原生函数的程序……98

第9天 设计面向对象语言 101

9.1 设计用于操作类与对象的语法……102
9.2 实现类所需的语法规则……103
9.3 实现eval方法……104
9.4 通过闭包表示对象……110
9.5 运行包含类的程序……114

第10天 无法割舍的数组 115

10.1 扩展语法分析器……116
10.2 仅通过修改器来实现数组……119

第2部分 性能优化篇

第11天 优化变量读写性能 123

11.1 通过简单数组来实现环境 124
11.2 用于记录全局变量的环境 127
11.3 事先确定变量值的存放位置 130
11.4 修正eval方法并最终完成性能优化 134

第12天 优化对象操作性能 137

12.1 减少内存占用 138
12.2 能否通过事先查找变量的保存位置来优化性能 141
12.3 定义lookup方法 144
12.4 整合所有修改并执行 147
12.5 内联缓存 152

第13天 设计中间代码解释器 156

13.1 中间代码与机器语言 157
13.2 Stone虚拟机 158
13.3 通过栈实现环境 167
13.4 寄存器的使用 170
13.5 引用变量的值 173
13.6 if语句与while语句 173
13.7 函数的定义与调用 175
13.8 转换为虚拟机器语言 177
13.9 通过虚拟机执行 184

第14天 为Stone语言添加静态类型支持以优化性能 187

14.1 指定变量类型 188
14.2 通过数据类型检查发现错误 193
14.3 运行程序时执行类型检查 204
14.4 对类型省略的变量进行类型推论 208
14.5 Java二进制代码转换 214
14.6 综合所有修改再次运行程序 226

第3部分 解说篇（自习时间）

第15天 手工设计词法分析器 229

15.1 修改自动机 …… 230
15.2 自动机程序 …… 233
15.3 正则表达式的极限 …… 235

第16天 语法分析方式 236

16.1 正则表达式与BNF …… 237
16.2 语法分析算法 …… 238
16.3 LL语法分析 …… 239
16.4 算符优先分析法与自底向上语法分析 …… 244

第17天 Parser库的内部结构 251

17.1 组合子分析 …… 252
17.2 解析器组合子的内部 …… 252

第18天 GluonJ的使用方法 263

18.1 设定类路径 …… 264
18.2 启动设定 …… 265
18.3 GluonJ语言 …… 267
18.4 功能总结 …… 268

第19天 抽象语法树与设计模式 271

19.1 理想的设计 …… 272
19.2 Interpreter模式 …… 273
19.3 Visitor模式 …… 276
19.4 使用反射 …… 282
19.5 面向切面语言 …… 284

第 1 部分

基础篇

第 1 天

来，我们一起做些什么吧

第 1 天 来，我们一起做些什么吧

——某大学研究室内

C 话说，我现在正在写一本新书。

H 老师，您这次写的是什么主题的书呢？

C 是一本和编译相关的书。确切地说，是关于语言处理器的书。

F 这样啊，这次是要写成一本教科书吗？

C 不，出版社要求我这次写得通俗些，所以这本书的内容会比教材来得简单。

H 那这次还会像前一本书[①]那样，通过对话形式进行解说吗？

C 这个问题现在还没有确定。有人赞成用对话的形式，但也有人反对。

F 老师，那这次的新书中会出现哪些人物呢？肯定会有 H 吧，毕竟这里他最年长。

H 哎呀，别这么说，就算没有我也没关系。

F H 肯定会出现啦。至于还会有哪些人，真是很期待呀。此外，M[②]那样称职的角色也必不可少。这次选谁才好呢？

设计程序时使用的语言称为程序设计语言。如 Java 语言、C 语言、Ruby 语言、C++ 语言、Python 语言等，都是程序设计语言。

程序员必须使用与各程序设计语言相匹配的软件来执行由该语言写成的程序。这种软件通常称为语言处理器。本章将首先说明语言处理器的基本概念。

1.1 机器语言与汇编语言

——不久后

A 该不会是要让我来扮演 M 的角色吧？真是这样倒也没问题，M 一直也很关照我。

C 不，所有出现的人物都是虚构的，不必在意。

有些程序设计语言无需借助软件执行，也就是说，它们不需要语言处理器。这些语言称为机器语言。机器语言可以由硬件直接解释执行，理论上不必使用软件。

① 千叶滋《面向方面程序设计入门》技术评论社，2005 年。

② 这里指的是在注 1 提到的书中出现的角色 M。