

## <<正则指引>>

### 图书基本信息

书名：<<正则指引>>

13位ISBN编号：9787121165511

10位ISBN编号：7121165511

出版时间：2012-5

出版时间：电子工业出版社

作者：余晟

页数：336

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：<http://www.tushu007.com>

## &lt;&lt;正则指引&gt;&gt;

## 前言

前言 提到正则表达式，许多人很有点不屑一顾：这东西，不登大雅之堂，再说也不是总要用到，何必专门花时间学习？

没错，正则表达式并不“总要用到”，但到了需要的场合用不上，往往产生“一分钱难倒英雄汉”的尴尬。

经常需要处理文本的程序员自然会知道正则表达式的价值，其他的程序员如果不会正则表达式，即便开发的领域与文本处理没什么关系，也难免“躺着中枪”的命运--前几天我遇到一个问题，将一行长长的地址拆分成多行，负责这部分的程序员的日常工作只是制作PDF而已，拆分地址是很“边缘”的功能，但不会正则表达式就无法准确折行（一般需要在标点符号出现的地方折行，而不能只在空白字符处折行，但是不同语言中的标点符号各有不同），结果一筹莫展；相反，如果了解正则表达式，就可以很容易地处理各种语言中的标点字符。

以我的开发经验来看，专门花点时间掌握正则表达式，确实是非常有必要的。目前可以见到的关于正则表达式的书籍和资料有不少，但又各有不足。

在互联网上，流传着一些编程语言的正则文档和《30分钟教会你正则表达式》之类的帖子。这类资料的好处是简单直接，查到了，如果有现成的例子，而且适用于自己的语言，则可以直接拿来用；然而，其坏处也是简单直接，因为缺乏背后原理的讲解，如果找不到现成的例子，或者找不到能在自己所使用语言中行得通的例子（需知道，同样的正则表达式并不能直接套用到不同的语言中），则束手无策。

在正式的出版领域，已经有《精通正则表达式》、《正则表达式必知必会》之类的书籍出版，尤其是前者，堪称关于正则表达式的经典著作，如果想认真学习正则表达式，这类书籍是必须阅读的。但这类书籍也有一个弱点，即都是由英文版本翻译而来的，更多地侧重英文文本的处理，身为中文世界的开发人员，我们经常需要处理中文文本--对于处理英文之外的字符，正则表达式已经提供了足够丰富的功能，但如何用对、用好这些功能，资料却很匮乏。

我经常需要给人讲解正则表达式的相关知识，时常惋惜的是，开发人员为这些问题所困扰；正因为如此，本书的写作动机便是着力弥补现有资料的缺陷。

相对于正则文档和速成教学帖子，本书深入讲解了匹配背后的原理，往往会举一反三，告诉读者，这里为何这样写，如果改成其他形式，会造成什么结构；并且，集中讲解和比较了多种语言中正则表达式用法的异同，方便读者把现成的正则表达式“移植”到自己的工作环境中。

相对于《精通正则表达式》等正式的书籍，本书辟出专门的内容讲解语言和编码，告诉读者如何设定编码，如何正确处理中文等字符。

另外，本书还涵盖了.NET、Java、JavaScript、PHP、Python、Ruby六种常用语言，对每种语言给出专门章节，不但详细介绍了语言中正则表达式的用法，更点明了版本之间的细微差异，不但可以作为专门学习的教材，还可以成为有用的参考手册。

本书结构 本书可以分为三大部分。

第一部分主要讲解正则表达式的基础知识，覆盖常见正则表达式中的各种功能和结构。看完前面3章，就可以基本弄明白现在流行的各种正则表达式；尤其是如果你之前有一些经验，会觉得阅读起来并不困难。

但是我也希望读者不要忽略其他的内容，断言和匹配模式现在已经是正则表达式的“标准配备”了，而且确实可以派上大用场，所以第4章和第5章的内容，即便不是很熟悉，阅读起来可能有一些麻烦，也不应该忽略。

最后的第6章，则厘清了正则表达式在使用中的若干疑惑，了解它们，你就可以相对自由地在正则表达式的世界里行走了。

第二部分主要讲解关于正则表达式的更深入的知识，这一部分用3章的内容，详细探讨了编码问题、匹配原理、解题思路。

这部分内容更抽象，需要多花一点时间来阅读和理解，但是它们确实可以帮你在正则表达式的世界里登堂入室，脱离“术”的层面，掌握万变不离其宗的“道”。

## &lt;&lt;正则指引&gt;&gt;

第三部分的作用是接地气，将之前介绍的各种知识落实到六种常用语言.NET、Java、JavaScript、PHP、Python、Ruby中来。

每一章的开头有正则功能列表，其中的功能都对应着前面部分的讲解，这些功能的具体应用实例，以及不同版本之间的差异，则在章节中详细讲解，每一章的最后还给出了常见任务的示例代码，方便日后查询。

在最后，第16章简要介绍了正则表达式在Linux下常用工具vi、grep、awk、sed中的使用，并通过一个实际的例子将这几种工具串起来，对比说明了它们适合解决的问题。

在本书的最后提供了用作参考的3个附录。

附录A是正则表达式的常用功能在不同语言中的比对，希望能给需要在多种语言中使用正则表达式或者移植正则表达式的读者提供一份有用的参考；附录B给出了若干常见的正则表达式，比如匹配邮政编码、身份证号、手机号、QQ号、电子邮件地址等，希望能成为常见问题的“速查手册”；附录C列出了常用正则表达式的工具和资源，方便大家调试自己的正则表达式，以及继续深入学习。

本书读者 本书适合以下几类读者。

经常需要进行文本处理（比如日志分析或网络运维）的技术人员。

这些读者或许已经熟悉了正则表达式的基本用法，但面对日益复杂化和海量化的数据，阅读本书可以帮助你更准确、更高效地处理文本，提升自己工作的价值。

熟悉常用开发语言的程序员。

虽然这些读者不需要专职进行文本处理，但源代码和许多数据其实也是文本，如果不会正则表达式，在偶然遇到处理源代码或文本数据的任务时，往往会产生躺着中枪的无力感。

本书第三部分可以帮你迅速找到有关的例子，并落实在自己的编程语言中。

当然前两部分也非常有必要，因为它们可以帮你夯实基础。

对正则表达式已经有一定了解的读者。

这些读者虽然能用正则表达式解决常见的任务，但未必了解正则表达式的编码问题、匹配原理、解题思路，仔细阅读本书的第二部分，可以深化完善对正则表达式的理解；而第三部分详细比较了使用正则表达式时各种语言，以及同一种语言中各种版本的差异。

所有这一切，应该可以让你对正则表达式的掌握更上一层楼。

致谢 一本书的完成，必然离不开众多人的帮忙。

首先需要感谢的是周筠老师和徐定翔、卢鹤翔两位编辑，他们在我写作的最初阶段做了大量细心、耐心的工作，完全可以说，没有他们的这些工作，我就不会有写作这本书的念头，或者坚持写完的动力。

然后要感谢的是电子工业出版社的杨福平副总编和张月萍编辑，没有他们的关照和辛劳工作，这本书的出版定然会遇到更多的困难。

感谢我的朋友霍炬和韩磊，虽然我之前阅读过《精通正则表达式》，但与翻译和写作结缘，他们给了我莫大的帮助，有了这个契机，才有了现在的《正则指引》。

尤其值得一提的是霍炬的夫人西乔，精心手绘了这本书的封面，在这里表示诚挚的谢意。

感谢我曾工作过的盛大创新院以及创新院的各位同事（李骏、郝培强、庄表伟、丁宇、许式伟、莫华枫、李道兵、赵劫、樊一鹏、张一宁等），创新院给大家宽松自由的工作环境，与各位同事的讨论加深了我对正则表达式的理解，也为我提供了许多例子。

感谢张东亮、陆亦斌、孙勇、叶劲峰等各位朋友，愿意拨冗阅读本书的草稿，并提出了大量专业的意见。

感谢何源、陈钢、贺钧、陈驰等读者，试读本书之后提出了大量的宝贵意见，在最后关头打消了我心中的许多忐忑。

在更早之前，我的父母从小就鼓励我研究和了解各种科学原理（“玩也要动脑筋”），我之所以有兴趣探究正则表达式背后的世界，而不满足于“够用/凑合”，归源都是受益于这种思维行为习惯。

此外，在中小学阶段，我的语文老师罗碧玉、郭志鸿、易玺铭培养了我对于文字的兴趣，在大学阶段

## <<正则指引>>

，东北师范大学文学学院的王确老师给了我这个理科生非常多的帮助和指引。  
对各位师长，在此一并表示感谢，能遇到你们是我的幸运。

最后还需要感谢许多为这本书做出过贡献的人，你们的名字我可能暂时无法记起，或者无法一一罗列，但我会在我心中存留对你们的谢意。

## &lt;&lt;正则指引&gt;&gt;

## 内容概要

《正则指引》针对作者在开发中遇到的实际问题，以及其他开发人员咨询的问题，总结出一套使用正则表达式解题的办法，并通过具体的例子指导读者拆解、分析问题。

全书分为三大部分：第一部分主要讲解正则表达式的基础知识，涵盖了常见正则表达式中的各种功能和结构；第二部分主要讲解关于正则表达式的更深入的知识，详细探讨了编码问题、匹配原理、解题思路；第三部分将之前介绍的各种知识落实到6种常用语言.NET、Java、JavaScript、PHP、Python、Ruby中，不但详细介绍了语言中正则表达式的用法，更点明了版本之间的细微差异，既可以作为专门学习的教材，也可以作为有用的参考手册。

本书适合经常需要进行文本处理（比如日志分析或网络运维）的技术人员、熟悉常用开发语言的程序员，以及已经对正则表达式有一定了解的读者阅读。

## <<正则指引>>

### 作者简介

余晟，毕业于计算机系，副修中文，非正统型技术爱好者。  
曾任抓虾网、银杏泰克主力程序员，盛大创新院高级研究员，现任华南某电商公司技术部总监。  
坚信计算机可以无限延伸人的能力，前提是人必须理解计算机的逻辑，所以对任何技术都不应该浅尝辄止，仅仅满足于“会用”。

已经翻译出版《精通正则表达式（第3版）》和《技术领导之路》，审阅《软件架构师应该知道的97件事》和《REST in Practice中文版》。

## &lt;&lt;正则指引&gt;&gt;

## 书籍目录

## 第一部分

## 第1章 字符组

- 1.1 普通字符组
- 1.2 关于Python的基础知识
- 1.3 普通字符组 (续)
- 1.4 元字符与转义
- 1.5 排除型字符组
- 1.6 字符组简记法
- 1.7 字符组运算
- 1.8 POSIX字符组

## 第2章 量词

- 2.1 一般形式
- 2.2 常用量词
- 2.3 数据提取
- 2.4 点号
- 2.5 滥用点号的问题
- 2.6 忽略优先量词
- 2.7 转义

## 第3章 括号

- 3.1 分组
- 3.2 多选结构
- 3.3 引用分组
  - 3.3.1 反向引用
  - 3.3.2 各种引用的记法
  - 3.3.3 命名分组
- 3.4 非捕获分组
- 3.5 补充
  - 3.5.1 转义
  - 3.5.2 URL Rewrite
  - 3.5.3 一个例子

## 第4章 断言

- 4.1 单词边界
- 4.2 行起始/结束位置
- 4.3 环视
- 4.4 补充
  - 4.4.1 环视的价值
  - 4.4.2 环视与分组编号
  - 4.4.3 环视的支持程度
  - 4.4.4 环视的组合
  - 4.4.5 断言和反向引用之间的关系

## 第5章 匹配模式

- 5.1 不区分大小写模式
  - 5.1.1 模式的指定方式
- 5.2 单行模式
- 5.3 多行模式

## &lt;&lt;正则指引&gt;&gt;

5.4 注释模式

5.5 补充

5.5.1 更多的模式

5.5.2 修饰符的作用范围

5.5.3 失效修饰符

5.5.4 模式与反向引用

5.5.5 冲突策略

5.5.6 哪种方式更好

第6章 其他

6.1 转义

6.1.1 字符串转义与正则转义

6.1.2 元字符的转义

6.1.3 彻底消除元字符的特殊含义

6.1.4 字符组中的转义

6.2 正则表达式的处理形式

6.2.1 函数式处理

6.2.2 面向对象式处理

6.2.3 比较

6.2.4 线程安全性

6.3 表达式中的优先级

第二部分

第7章 Unicode

7.1 关于编码

7.2 推荐使用Unicode编码

7.3 Unicode匹配规则

7.4 单词边界

7.5 码值

7.6 Unicode属性

7.6.1 Unicode Property

7.6.2 Unicode Block

7.6.3 Unicode Script

7.7 Unicode属性列表

7.7.1 Unicode Property

7.7.2 Unicode Block

7.7.3 Unicode Script

7.8 POSIX字符组

第8章 匹配原理

8.1 有穷自动机

8.2 正则表达式的匹配过程

8.3 回溯

8.4 NFA和DFA

第9章 常见问题的解决思路

9.1 关于元素的三种逻辑

9.1.1 必须出现

9.1.2 可能出现

9.1.3 不能出现

9.2 正则表达式的常见操作



## &lt;&lt;正则指引&gt;&gt;

- 9.2.1 提取
- 9.2.2 验证
- 9.2.3 替换
- 9.2.4 切分
- 9.3 正则表达式的优化建议
  - 9.3.1 使用缓存
  - 9.3.2 尽量准确地表达意图
  - 9.3.3 避免重复匹配
  - 9.3.4 独立出文本和锚点
- 9.4 别过分依赖正则表达式
  - 9.4.1 彻底放弃字符串操作
  - 9.4.2 思维定势
  - 9.4.3 正则表达式可以匹配各种文本

## 第三部分

## 第10章 .NET

- 10.1 预备知识
- 10.2 正则功能详解
  - 10.2.1 列表
  - 10.2.2 字符组
  - 10.2.3 Unicode属性
  - 10.2.4 字符组简记法
  - 10.2.5 单词边界
  - 10.2.6 行起始/结束位置
  - 10.2.7 环视
  - 10.2.8 匹配模式
  - 10.2.9 捕获分组的引用
- 10.3 正则API简介
  - 10.3.1 Regex
  - 10.3.2 Match
- 10.4 常用操作示例
  - 10.4.1 验证
  - 10.4.2 提取
  - 10.4.3 替换
  - 10.4.4 切分

## 第11章 Java

- 11.1 预备知识
- 11.2 正则功能详解
  - 11.2.1 列表
  - 11.2.2 字符组
  - 11.2.3 Unicode属性
  - 11.2.4 字符组简记法
  - 11.2.5 单词边界
  - 11.2.6 行起始/结束位置
  - 11.2.7 环视
  - 11.2.8 匹配模式
  - 11.2.9 纯文本模式
  - 11.2.10 捕获分组的引用

## &lt;&lt;正则指引&gt;&gt;

## 11.3 正则API简介

## 11.3.1 Pattern

## 11.3.2 Matcher

## 11.3.3 String

## 11.4 常用操作示例

## 11.4.1 验证

## 11.4.2 提取

## 11.4.3 替换

## 11.4.4 切分

## 第12章 JavaScript

## 12.1 预备知识

## 12.2 正则功能详解

## 12.2.1 列表

## 12.2.2 字符组

## 12.2.3 字符组简记法

## 12.2.4 单词边界

## 12.2.5 行起始/结束位置

## 12.2.6 环视

## 12.2.7 匹配模式

## 12.2.8 捕获分组的引用

## 12.3 正则API简介

## 12.3.1 RegExp

## 12.3.2 String

## 12.4 常用操作示例

## 12.4.1 验证

## 12.4.2 提取

## 12.4.3 替换

## 12.4.4 切分

## 12.5 关于ActionScript

## 12.5.1 RegExp

## 12.5.2 匹配规则

## 12.5.3 匹配模式

## 12.5.4 正则API

## 第13章 PHP

## 13.1 预备知识

## 13.2 正则功能详解

## 13.2.1 列表

## 13.2.2 字符组

## 13.2.3 Unicode属性

## 13.2.4 字符组简记法

## 13.2.5 单词边界

## 13.2.6 行起始/结束位置

## 13.2.7 环视

## 13.2.8 匹配模式

## 13.2.9 纯文本模式

## 13.2.10 捕获分组的引用

## 13.3 正则API简介

## &lt;&lt;正则指引&gt;&gt;

- 13.3.1 PREG 常量说明
- 13.3.2 preg\_quote
- 13.3.3 preg\_grep
- 13.3.4 preg\_match
- 13.3.5 preg\_match\_all
- 13.3.6 preg\_last\_error
- 13.3.7 preg\_replace
- 13.3.8 preg\_replace\_callback
- 13.3.9 preg\_filter
- 13.3.10 preg\_split
- 13.4 常见的正则操作举例
  - 13.4.1 验证
  - 13.4.2 提取
  - 13.4.3 替换
  - 13.4.4 切分
- 第14章 Python
  - 14.1 预备知识
  - 14.2 正则功能详解
    - 14.2.1 列表
    - 14.2.2 字符组
    - 14.2.3 Unicode属性
    - 14.2.4 字符组简记法
    - 14.2.5 单词边界
    - 14.2.6 行起始/结束位置
    - 14.2.7 环视
    - 14.2.8 匹配模式
    - 14.2.9 捕获分组的引用
  - 14.3 正则API简介
    - 14.3.1 RegexpObject
    - 14.3.2 re.compile ( regex[, flags] )
    - 14.3.3 re.search ( pattern, string[, flags] )
    - 14.3.4 MatchObject
    - 14.3.5 re.match ( pattern, string[, flags] )
    - 14.3.6 re.findall ( pattern, string[, flags] )
    - 14.3.7 re.finditer ( pattern, string[, flags] )
    - 14.3.8 re.split ( pattern, string[, maxsplit=0, flags=0] )
    - 14.3.9 re.sub ( pattern, repl, string[, count, flags] )
  - 14.4 常用操作示例
    - 14.4.1 验证
    - 14.4.2 提取
    - 14.4.3 替换
    - 14.4.4 切分
- 第15章 Ruby
  - 15.1 预备知识
  - 15.2 正则功能详解
    - 15.2.1 列表
    - 15.2.2 字符组

## &lt;&lt;正则指引&gt;&gt;

- 15.2.3 Unicode属性
- 15.2.4 字符组简记法
- 15.2.5 单词边界
- 15.2.6 行起始/结束位置
- 15.2.7 环视
- 15.2.8 匹配模式
- 15.2.9 捕获分组的引用
- 15.3 正则API简介
  - 15.3.1 Regexp
  - 15.3.2 Regexp.match ( text )
  - 15.3.3 Regexp.quote ( text ) 和 Regexp.escape ( text )
  - 15.3.4 String.index ( Regexp )
  - 15.3.5 String.scan ( Regexp )
  - 15.3.6 String.slice ( Regexp )
  - 15.3.7 String.split ( Regexp )
  - 15.3.8 String.sub ( Regexp, Str )
  - 15.3.9 String.gsub ( Regexp, String )
- 15.4 常用操作示例
  - 15.4.1 验证
  - 15.4.2 提取
  - 15.4.3 替换
  - 15.4.4 切分
- 15.5 Ruby 1.9的新变化
- 第16章 Linux/UNIX
  - 16.1 POSIX
    - 16.1.1 POSIX规范
    - 16.1.2 POSIX字符组
  - 16.2 vi
    - 16.2.1 字符组及简记法
    - 16.2.2 量词
    - 16.2.3 多选结构和捕获分组
    - 16.2.4 环视
    - 16.2.5 锚点和单词边界
    - 16.2.6 替换操作的特殊字符
    - 16.2.7 replacement中的特殊变量
    - 16.2.8 补充
  - 16.3 grep
    - 16.3.1 基本用法
    - 16.3.2 字符组
    - 16.3.3 锚点和单词边界
    - 16.3.4 量词
    - 16.3.5 多选结构和捕获分组
    - 16.3.6 options
    - 16.3.7 egrep和fgrep
    - 16.3.8 补充
  - 16.4 awk
    - 16.4.1 基本用法

## <<正则指引>>

16.4.2 字符组及简记法

16.4.3 锚点和单词边界

16.4.4 量词

16.4.5 多选结构

16.4.6 补充

16.5 sed

16.5.1 基本用法

16.5.2 字符组及简记法

16.5.3 锚点和单词边界

16.5.4 量词

16.5.5 多选结构和捕获分组

16.5.6 options

16.5.7 补充

16.6 总结

附录A 常用语言中正则特性一览

附录B 常用的正则表达式

附录C 常用的正则表达式工具及资源

## &lt;&lt;正则指引&gt;&gt;

## 章节摘录

版权页：插图：第1章 字符组 1.1 普通字符组 字符组 (Character Class) 是正则表达式最基本的结构之一，要理解正则表达式的“灵活”，认识它是第一步。

顾名思义，字符组就是一组字符，在正则表达式中，它表示“在同一个位置可能出现的各种字符”，其写法是在一对方括号【和】之间列出所有可能出现的字符，简单的字符组比如[ab]、[314]、【#. ?

】在解决一些常见问题时，使用字符组可以大大简化操作，下面举“匹配数字字符”的例子来说明。字符可以分为很多类，比如数字、字母、标点等。

有时候要求“只出现一个数字字符”，换句话说，这个位置上的字符只能是0、1、2、...、8、9这10个字符之一。

要进行这种判断，通常的思路是：用10个条件分别判断字符是否等于这10个字符，对10个结果取“或”，只要其中一个条件成立，就返回True，表示这是一个数字字符，其伪代码如例1—1所示。

例1—1判断数字字符的伪代码 `charStr=="0"||charStr=="1"...||charStr=="9"` 注：因为正则表达式处理的都是“字符串”(String)而不是“字符”，所以这里假设变量charStr（虽然它只包含一个字符）也是字符串类型，使用了双引号，在有些语言中字符串也用单引号表示。

这种解法的问题在于太烦琐——如果要判断是否是一个小写英文字母，就要用||连接26个判断；如果还要兼容大写字母，则要连接52个判断，代码长到几乎无法阅读。

相反，用字符组解决起来却异常简单，具体思路是：列出可能出现的所有字符（在这个例子里就是10个数字字符），只要出现了其中任何一个，就返回True。

例1—2给出了使用字符组判断的例子，程序语言使用Python。

例1—2用正则表达式判断数字字符 `re.search("[8123456789]", charStr)!`

`=None` `re.search()` 是Python提供的正则表达式操作函数，表示“进行正则表达式匹配”；charStr仍然是需要判断的字符串，而`[0123456789]`则是以字符串形式给出的正则表达式，它是一个字符组，表示“这里可以是0、1、2、...、8、9中的任意一个字符。

只要charStr与其中任何一个字符相同（或者说“charStr可以由【0123456789】匹配”），就会得到一个Matchobject对象（这个对象暂时不必关心，在第21页会详细讲解）；否则，返回None。

所以判断结果是否为None，就可以判断charStr是否是数字字符。

当今流行的编程语言大多支持正则表达式，上面的例子在各种语言中的写法大抵相同，唯一的区别在于如何调用正则表达式的功能，所以用法其实大同小异。

例1—3列出了常见语言中的表示，如果你现在就希望知道语言的细节，可以参考本书第三部分的具体章节。

## <<正则指引>>

### 编辑推荐

《正则指引》适合经常需要进行文本处理（比如日志分析或网络运维）的技术人员、熟悉常用开发语言的程序员，以及已经对正则表达式有一定了解的读者阅读。

## &lt;&lt;正则指引&gt;&gt;

## 名人推荐

掌握正则表达式应该是IT工程师的一项标准技能，遗憾的是，过去，不少人多多少少忽视了这一点，所以在工作中总要应对正则表达式带来的【麻烦】。

我相信只有掌握并熟练运用它才有可能成为一个高效率的工程师。

期待每个人手边都有一本正则表达式的参考书，当然，最好就是你现在看到的这本。

就在写这句话的几分钟前，我又从这本书中学到了一个有用的技巧。

——冯大辉 配合恰当的案例，大量的反问，使读者自问、思考，扣人心弦，比较有代入感。

加上配图，很容易让读者全面认识正则表达式。

在原理讲解的章节，对比两种理论模型的区别，顺其自然地引入NFA引擎的关键要素:回溯。

使读者从匹配原理上了解回溯，写出高效、严谨的正则表达式。

——陈驰 本书由浅入深地讲述了正则表达式，在正则的应用和调优方面有非常详细的介绍，特别在正则表达式处理中文方面有独到的阐述。

对于需要经常处理中文的国内技术人员来说，无疑是非常值得拥有的一本手册。

——贺钧 正则表达式是程序员的必备知识。

如果您还没有使用过这个强大的工具，或者学习正则表达式总不得要领。

确实可以读读《正则指引》。

——何源 这是一本通俗版的“精通正则表达式”。

高手很难挑出毛病，一般程序员会受益匪浅，普通用户一步步读下去也能登堂入室。

——张东亮 余晟在之前翻译业内名著的基础上，结合中文环境和自己的丰富经验，再接再厉推出自己的原创著作，实在是我等码农的一大幸事。

——陈钢



## 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:<http://www.tushu007.com>