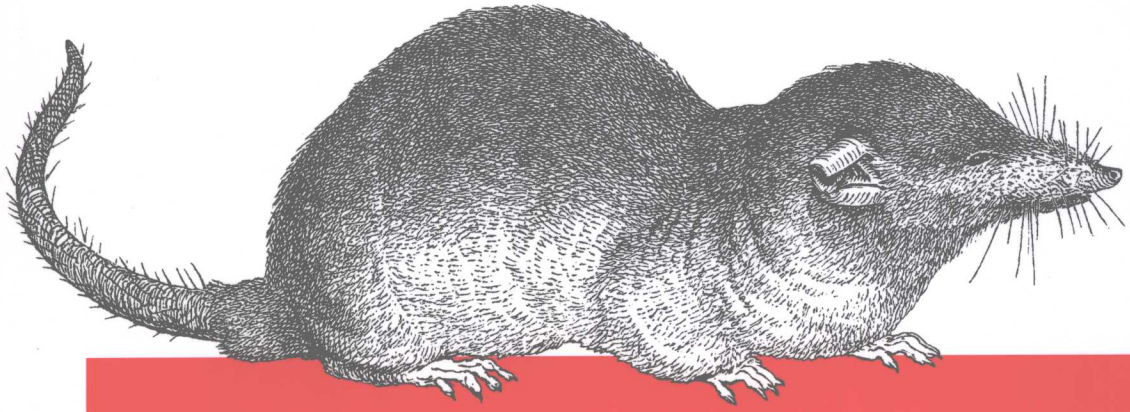
Regular Expressions Cookbook



正则表达式经典实例

使用8种编程语言的详细解决方案
包括一个正则表达式简明教程

[美] *Jan Goyvaerts*
Steven Levithan 著
郭耀 译

O'REILLY®

人民邮电出版社
POSTS & TELECOM PRESS

O'REILLY®

正则表达式经典实例

[美] Jan Goyvaerts　Steven Levithan

郭耀　译

人 民 邮 电 出 版 社

北 京

图书在版编目（CIP）数据

正则表达式经典实例 / (美) 高瓦特斯 (Goyvaerts, J.)，(美) 利维森 (Levithan, S.) 著；郭耀译. -- 北京：人民邮电出版社，2010.6
ISBN 978-7-115-22832-1

Ⅰ. ①正… Ⅱ. ①高… ②利… ③郭… Ⅲ. ①正则表达式 Ⅳ. ①TP301.2

中国版本图书馆CIP数据核字(2010)第076768号

版权声明

正则表达式经典实例

◆ 著 [美] Jan Goyvaerts Steven Levithan
译 郭 耀
责任编辑 刘映欣

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京鑫正大印刷有限公司印刷

◆ 开本：787×1000 1/16
印张：29.5
字数：617 千字 2010 年 6 月第 1 版
印数：1－4 000 册 2010 年 6 月北京第 1 次印刷

著作权合同登记号 图字：01-2009-6947 号
ISBN 978-7-115-22832-1

定价：69.00 元

读者服务热线：(010)67132705 印装质量热线：(010)67129223
反盗版热线：(010)67171154

前言

正则表达式在过去十多年间越来越普及。如今所有常用的编程语言都会包含一个强大的正则表达式函数库，或者甚至是在语言本身就内嵌了对于正则表达式的支持。许多开发人员都会利用这些正则表达式的功能，在应用程序中为用户提供使用正则表达式对其数据进行查找或者过滤的能力。正则表达式真正是无处不在。

随着正则表达式的广泛采用，出现了许多相关的著作。大多数这类书籍都很好地讲解了正则表达式的语法，并且还会提供一些例子以及参考文献。然而，我们还没有看到有任何一本书能够面向现实世界中使用的计算机，以及在各种 Internet 应用中遇到的实际问题，为读者提供基于正则表达式的解决方案。因此，本书作者 Steve 和 Jan 决定写一本书来填补这种空白。

我们特别期望能够展现给读者的是：如何使用正则表达式来解决那些对正则表达式经验较少的人们认为无法解决的问题，以及软件纯粹主义者认为不能用正则表达式来解决的问题。因为如今正则表达式无处不在，所以它们通常可以作为工具，直接被最终用户使用，而不需要程序员团队的参与。即使是对于程序员来说，常常也可以在信息检索和更新的任务中采用一些正则表达式来节省大量时间，因为这些功能如果使用过程式代码来实现，可能就会需要几个小时甚至几天的时间，也可能会由于需要采用第三方的函数库，而不得不经过事先审查和经理层的审批。

不同版本带来的问题

与 IT 业界流行的东西一样，正则表达式也拥有许多种不同的实现，以及不同程度的兼容性。这就出现了许多不同的正则表达式**流派**（flavor），它们在处理一个特定正则表达式的时候并不总是拥有完全一样的表现，有时候甚至会无法正常使用。

在许多书中的确也提到了目前存在的不同流派，并且指出了其中的一些区别。但是，如果某种流派缺少特定功能的时候，它们通常会选择在这里或那里略掉一些流派，而不是为之提供可替代的解决方案或者是应急方案。而当你不得不在不同的应用程序或者不同程序语言中使用不同的正则表达式流派的时候，就会感到很受挫折。

在文献中常常会看到一些不严格的表达，例如“所有人现在都在使用 Perl 风格的正则表达式”，不幸的是这种说法会把很大范围的不兼容性边缘化。即使是“Perl 风格”的函数库也可能会有显著的区别，而且与此同时 Perl 也在持续不断地演化。一旦拥有了这种过度简单化的印象，就可能会导致有些程序员浪费长达半个小时的时间来运行调

试器却得不到任何结果，而不是去认真检查他们的正则表达式的实现细节。即使当他们发现所依赖的一些功能不存在的时候，也不一定总是知道该如何找到解决方案。

本书是市场上能看到的第一本讨论功能强大的各种常见正则表达式流派的书，并且本书从头到尾都会坚持这样的原则。

目标读者

如果你经常在计算机上处理文本，不管是搜索一大堆的文档，在文本编辑器中处理文本，抑或是开发需要搜索或处理文本的软件，都应该认真读一读这本书。正则表达式对于上述这些工作来说是一个非常优秀的工具。本书会教给你需要了解的关于正则表达式的所有东西，你不需要任何先前的经验。因为我们会从关于正则表达式的最基本的概念开始讲起。

如果你已经拥有关于正则表达式的经验，那么你会看到在其他教材和网上文章中经常一带而过的大量细节。如果你曾经遭遇过正则表达式在一个应用程序中可用，而在另外一个程序中不可用的情形，那么就会因为本书中对世界上最流行的 7 种正则表达式流派给出的翔实均衡的讲解，而感到受益颇多。我们把本书组织成一本经典实例（cookbook，原意为菜谱），从而可以直接跳到你想要细细阅读的话题。如果从头到尾阅读了整本书，你就会成为一个正则表达式的世界级“大厨”。

无论你是否是程序员，本书除了会教给你关于正则表达式所需知道的所有内容之外，还会讲解更多其他内容。如果你想要在文本编辑器、查找工具，或是任意含有带“正则表达式”标签的应用程序中使用正则表达式，那么你根本不需要任何编程经验就可以阅读本书。本书中的大多数例子都拥有完全基于一个或多个正则表达式的解决方案。

如果你是程序员，那么第 3 章会讲解在源代码中实现正则表达式所需的所有信息。本章假设读者对所选择的编程语言的基本语言特性是熟悉的，但是并不假设你在源代码中曾经使用过任何的正则表达式。

涉及的技术

.NET、Java、JavaScript、PCRE、Perl、Python 以及 Ruby，这些不只是一些用在封面上的热门词汇。它们是本书要讲到的 7 种正则表达式流派。我们会把这 7 种流派等同对待，还会特别小心地指出这些正则表达式流派中所能找到的所有不一致的地方。

关于编程的一章（第 3 章）中包含使用如下语言的代码示例：C#、Java、JavaScript、PHP、Perl、Python、Ruby 以及 VB.NET。虽然这样做会让这一章看起来有些重复，但是这样读者就可以很容易跳过那些不感兴趣的语言的讨论，而不会错过对于你所选择语言应当知道的任何内容。

本书的组织结构

本书的前 3 章讲解一些有用的工具和基本信息，这些会给读者提供使用正则表达式的基础。随后的每一章则介绍各种不同的正则表达式，并对文本处理的一个领域进行深入讲解。

第 1 章“正则表达式简介”讲解正则表达式的作用，并介绍了一系列工具，它们会使你学习、创建和调试正则表达式更加容易。

第 2 章“基本正则表达式技巧”介绍正则表达式的每个元素和特性，以及有效使用正则表达式的一些重要指南。

第 3 章“使用正则表达式编程”详细介绍了编码相关的技术，并且包含了在本书中涉及的每种编程语言中使用正则表达式的代码示例。

第 4 章“合法性验证和格式化”中包含如何处理常见用户输入的实例，例如日期、电话号码以及不同国家的邮政编码。

第 5 章“单词、文本行和特殊字符”探讨常用的文本处理任务，例如检查文本行中是否包含或者不包含某个特定的单词。

第 6 章“数字”会讲解如何检测整数、浮点数以及这种输入的几种其他格式。

第 7 章“URL、路径和 Internet 地址”展示如何能够把在 Internet 上和 Windows 系统中常用的这些字符串拆分开来，并且利用它们来查找数据。

第 8 章“标记语言和数据交换”讲解如何处理 HTML、XML、逗号分隔的取值（CSV），以及 INI 风格的配置文件。

阅读须知

本书中在排版上采用如下约定。

`‹Regular•expression›`（正则表达式）

用来表示一个正则表达式，它可以单独出现，也可以出现在向某个应用程序的查找框中输入正则表达式的时候。如果不使用“宽松排列（free-spacing）”模式，那么正则表达式中的空格会使用一个实心圆点来表示。

`«Replacement•text»`（替代文本）

用来表示在“查找和替换”的操作中，正则表达式所匹配的文本会被替换成的文本。在替代文本（replacement text）中的空格也会用一个实心圆点来表示。

`Matched text`（匹配文本）

用来表示与一个正则表达式相匹配的目标文本（subject text）中的一部分。

...

在正则表达式中的省略号会被用来说明在使用该正则表达式之前必须“把这里的空白填好”。相应的文字解释中会告诉你在其中应该填入什么样的内容。

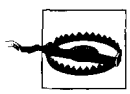
CR、LF 和 CRLF

CR、LF 和 CRLF 放在黑框中用来表示在字符串中实际出现的换行字符，而不是正则表达式中的字符转义序列（character escapes），例如\r、\n 和 \r\n。要创建这些字符，可以通过在应用程序的多行编辑面板中按回车键（Enter），或者也可以通过在源代码中使用多行字符常量，比如在 C#中的逐字字符串（verbatim strings），或是 Python 语言中的三引号字符串（triple-quoted strings）。

这个符号表示回车箭头，它与键盘上的回车（Return 或 Enter）键上的符号一样，用来说明必须打断一行才能使之符合印刷页面的宽度。当你在源代码中键入这些代码的时候，不需要按回车键，而是应该把所有内容都键入同一行之中。

提示

这个图标用来强调一个提示、建议或一般说明。



警告

这个图标用来说明一个警告或注意事项。

代码示例的使用

本书的目的就是要帮助读者完成手头的工作。一般来说，读者可以随意在程序和文档中使用本书中出现的代码。除非你打算再利用本书中大量的代码，否则并不需要联系我们以获得许可。销售或者发布 O'Reilly 图书中包含示例的 CD-ROM 则必须要获得许可。引用本书或者引用其中的示例代码来回答问题并不需要获得许可。在你的产品文档中利用本书中的大量代码示例则需要获得许可。

如果读者在引用本书时提供出处，我们会很感激，虽然我们并不要求你一定这样做。提供出处的时候通常需要包括书名、作者、出版社和书号（ISBN）。例如：“Regular Expressions Cookbook, by Jan Goyvaerts and Steven Levithan. O'Reilly 2009, 978-0-596-2068-7.”。

如果你觉得对代码示例的使用可能会超出上面所给出的许可范围，或是属于合理使用的范围之外，那么请随时通过 permissions@oreilly.com 联系我们。

联系我们

关于本书的意见和问题请按照如下地址与我们联系。

美国：

O'Reilly Media, Inc.

1005 Gravenstein Highway North

Sebastopol, CA 95472

中国：

100035 北京市西城区西直门成铭大厦 C 座 807 室

奥莱利技术咨询（北京）有限公司

我们还为本书建立了一个网页，其中包含了勘误表、示例和其他额外的信息。可以通过如下网址访问该网页：

http://www.regexcookbook.com

或者：

http://oreilly.com/catalog/9780596520687

关于本书的技术性问题或建议，请发邮件到：

bookquestions@oreilly.com

info@mail.oreilly.com.cn

关于我们的书籍、会议、资源中心和 O'Reilly Network 的更多信息，请访问我们的网站：

http://www.oreilly.com

http://www.oreilly.com.cn

致谢

我们要感谢 O'Reilly Media, Inc.的编辑 Andy Oram，他从头到尾给我们提供了莫大的帮助。我们还要感谢 Jeffrey Friedl、Zak Greant、Nikolaj Lindberg 和 Ian Morse，他们提供的详细技术审阅意见使本书得以做到全面而准确。

O'Reilly Media, Inc.介绍

为了满足读者对网络和软件技术知识的迫切需求，世界著名计算机图书出版机构 O'Reilly Media, Inc.授权人民邮电出版社，翻译出版一批该公司久负盛名的英文经典技术专著。

O'Reilly Media, Inc.是世界上在 UNIX、X、Internet 和其他开放系统图书领域具有领导地位的出版公司，同时也是联机出版的先锋。

从最畅销的 The Whole Internet User's Guide & Catalog（被纽约公共图书馆评为 20 世纪最重要的 50 本书之一）到 GNN（最早的 Internet 门户和商业网站），再到 WebSite（第一个桌面 PC 的 Web 服务器软件），O'Reilly Media, Inc.一直处于 Internet 发展的最前沿。

许多书店的反馈表明，O'Reilly Media, Inc.是最稳定的计算机图书出版商——每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly Media, Inc.具有深厚的计算机专业背景，这使得 O'Reilly Media, Inc.形成了一个非常不同于其他出版商的出版方针。O'Reilly Media, Inc.所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly Media, Inc.还有许多固定的作者群体——他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly Media,Inc.依靠他们及时地推出图书。因为 O'Reilly Media, Inc.紧密地与计算机业界联系着，所以 O'Reilly Media, Inc.知道市场上真正需要什么图书。

内 容 提 要

本书讲解了基于 8 种常用的编程语言使用正则表达式的经典实例。书中提供了上百种可以在实战中使用的实例，以帮助读者使用正则表达式来处理数据和文本。对于如何使用正则表达式来解决性能不佳、误报、漏报等常见的错误以及完成一些常见的任务，本书给出了涉及基于 C#、Java、JavaScript、Perl、PHP、Python、Ruby 和 VB.NET 等编程语言的解决方案。

本书的读者对象是对正则表达式感兴趣的软件开发人员和系统管理员。本书旨在教会读者很多新的技巧以及如何避免语言特定的陷阱，读者可以通过本书提供的实例解决方案库来解决实践中的复杂问题。

目录

第 1 章

正则表达式简介

在你打开这本书的时候，很可能已经热切地期望，要在你的代码中插入本书章节中找到的那些包含诸多括号和问号的笨拙字符串了。如果你已经准备好要“即插即用”，我们非常欢迎，实际的正则表达式会在第 4～8 章中给出并加以讲解。

但是从长远来看，阅读本书的最初几章会节省你大量的时间。例如，本章会向读者介绍许多工具——其中一些工具是本书作者之一的 Jan 所创建的，这些工具会帮助你测试和调试一个正则表达式，而不会等到把它们埋藏到代码中之后，那时候错误就非常难以查找了。而且这最初几章还会向读者展示如何使用正则表达式的不同特性和选项，帮助你轻松应对遇到的问题，并帮助你理解正则表达式，从而可以提高它们的性能，以及学习不同语言——甚至是你最喜欢的编程语言的不同版本之间——在处理正则表达式的时候出现的细微差别。

因此，我们在这些背景内容上花费了大量的精力，相信读者在开始读本书之前会阅读这些内容，或是在使用正则表达式时受到挫折而想要巩固你的理解的时候，会回头来阅读它们。

1.1 正则表达式的定义

在本书的上下文中，正则表达式（regular expression）是一种可以在许多现代应用程序和编程语言中使用的特殊形式的代码模式。可以使用它们来验证输入是否符合给定的文本模式；在一大段文本中查找匹配该模式的文本；用其他文本来替换匹配该模式的文本或者重新组织匹配文本的一部分；把一块文本划分成一系列更小的文本；或者是搬起石头砸自己的脚。本书会帮助你确切理解正在做的事情，从而避免可能会造成的灾难性后果。

术语"正则表达式"的历史

术语"正则表达式"来源于数学与计算机科学理论，它用来反映被称为"正则性"的数学表达式特点。这样一个表达式可以通过一个确定性有限自动机（DFA）用软件来实现。一个 DFA 是一个不使用回溯的有限状态机。

最早版本的 grep 工具所使用的文本模式是数学意义上的正则表达式。尽管名字看起来是一样的，然而如今流行的 Perl 风格的正则表达式已经完全不是数学意义上的正则表达式了。它们是采用非确定性的有限自动机（NFA）来实现的。你稍后就会学到和回溯有关的所有内容。关于这条说明，一个实践中的程序员需要记住的所有内容就是：象牙塔里的一些计算机科学家对于他们拥有良好定义的术语，被用于现实世界中更为有用的技术而感到非常不满。

如果学会了使用正则表达式的技巧，它们就会简化许多编程和文本处理的任务，并且使得许多没有正则表达式根本无法实现的任务成为可能。从一个文档中提取所有的 E-mail 地址，至少需要几十行，甚至是几百行过程式代码——这些代码编写起来费事，而且也很难维护。但是如果采用了合适的正则表达式，例如在实例 4.1 中所给的一样，那么就只需要很少的几行代码，或者甚至只要一行代码就可以了。

但是如果你试图想要用一个正则表达式来做太多的事情，或者是在事实上根本不适合的情形中非要使用正则表达式，那么就会明白为什么会存在如下的说法[1]：

有些人每当遇到一个问题的时候，就会想"我知道怎么做，用正则表达式就可以了。"那么现在他们就有两个问题需要解决了。

这些人所遇到的第二个问题指的就是他们并不会去阅读用户手册，也就是你现在手里拿到的这本书。请继续读下去。正则表达式是一个强大的工具。如果你的工作会涉及到在计算机上操作或是抽取文本，那么牢固地掌握正则表达式就会为你省去很多个不眠之夜。

多种正则表达式流派

说实话，上一小节的标题是在说谎。我们并没有定义正则表达式到底是什么。我们也不可能给出定义。对于哪些文本模式是正则表达式，而哪些不是正则表达式，并不存在正式的标准来给出恰如其分的定义。你可以想象得到，每种编程语言的设计人员，以及每个文本处理程序的开发人员，对于正则表达式应该是什么样子都会有自己不同的想法。因此，我们就不得不面对这样一大堆不同的正则表达式流派。

幸运的是，绝大多数设计人员与开发人员都比较懒惰。如果你可以复制别人已经做好的工作，为什么非要自己创建一些全新的东西呢？正因为此，所有现代的正则表达

[1] Jeffrey Friedl 在他的博客 http://regex.info/blog/2006-09-15/247 中探讨了这句话的来源和历史。

式流派，其中当然包含了本书要讨论的这些流派，都可以把它们的历史追溯到 Perl 这种编程语言。我们把这些流派都称作 Perl 风格的正则表达式。它们的正则表达式语法是非常相似的，而且在大多数情况下是兼容的，但是并不是在所有情况下都完全兼容。

即使是作者有时候也会偷懒。在本书中，我们通常会使用 regex 或者 regexp 来指代一个单个的正则表达式，而使用 regexes 用来指代其复数形式[1]。

正则流派并不是和编程语言一一对应的。脚本语言倾向于拥有它们自己的、内置的正则表达式流派。其他语言则会依赖于函数库来提供正则表达式支持。有些函数库是对多种语言可用的，而某些特定的语言则会选用一些不同的函数库。

本章要讲解的内容只与正则表达式的流派有关，因此会彻底忽略任何与编程有关的考虑事项。从第 3 章开始，我们会给出一些代码列表，因此你可以先跳到第 3 章的“编程语言与正则表达式流派”一节，以了解你将会使用哪些流派来工作。但是现在请先忽略所有与编程相关的内容。下面一小节中列出的工具将会通过“动手学习”的方式，让你可以更方便地探索正则表达式的语法。

本书涉及的正则流派

在本书中，我们选择了如今在使用中的最为流行的正则流派。这些都是 Perl 风格的正则流派。有些流派会比其他流派多一些特性。但是如果两种流派拥有同一个特性的话，那么它们通常都会使用相同的语法。当然也会有例外，当我们遇到这些烦人的不一致情况时，在书中会加以提示。

所有这些正则流派都属于目前正在活跃开发中的编程语言和函数库的一部分。下面的流派列表会告诉你本书所用到的是哪些版本。在本书后面，如果所讲解的正则表达式在所有流派中效果都一样，那么我们在提到该流派时就会不区分其版本。而在几乎所有情况下都会是这种情况。除了会影响到一些边界情况的 bug 修复之外，正则表达式流派一般都不会改变，唯一例外是添加新的特性，从而原来被认为是错误的语法现在会被指定新的含义。

Perl

> Perl 对于正则表达式的内置支持是正则表达式今天得以流行的主要原因。本书会涉及 Perl 5.6、Perl 5.8 以及 Perl 5.10。
>
> 许多应用程序和正则库都号称它们使用的是 Perl 或者与 Perl 兼容的正则表达式，而事实上它们仅仅是使用了 Perl 风格的正则表达式。它们使用一种与 Perl 相似的正则语法，但是并不支持相同的正则特性集。最有可能的情形是，它们使用的是这一特性集中的正则表达式流派之一，而这些流派都是属于 Perl 风格的。

[1] 译者注：本书中将“Regular Expression”翻译为“正则表达式”，“regex”、“regexp”或者“regexes”则简称为“正则式”或“正则”。