第16学时　Perl语言开发界

本学时你可以得到一个稍事休息的机会，让我们聊一聊 Perl语言的演变历史和它的文化背景。

也许你认为这一学时应该作为本书的附录或者前言，但如果作为附录或者前言，往往很容易被读者忽略。为了充分利用 Perl的潜在功能，你必须了解 Perl语言开发界的一些情况。

了解是什么因素使得 Perl语言开发界能够顺利地进行它们的开发活动，这将有于了解你可以使用什么资源，为什么存在这些资源，这些资源如何运行，为什么 Perl能够成为这样一种编程语言。许多资源都能够帮助你回答这些问题，本学时将帮助你查找这些资源。

在本学时中，你将要学习：

- 关于Perl的一些历史知识。
- 什么是CPAN，你如何使用它。
- 何处获得帮助。

16.1　Perl究竟是一种什么语言

为了获得Perl的文化背景知识，它如何运行，以及你可以使用哪些资源，你有必要知道究竟是什么因素使得 Perl能够一脉传承发展至今。

16.1.1　Perl的简单发展历史

1988年，Internet还是个非常不同的系统。首先，它的规模比较小，并且与它今天的样子大不相同。当时的 Internet大约只有 6万台计算机，而今天它的数量超过 1千万台，并且仍在迅速增加。

当时Wold Wide Web尚未问世，直到1991年在的CERN计算机网络上才提出了 World Wide Web的思路，到了 1993年，出现了第一个图形浏览器 Mosaic。

Internet上的大部分信息都是文字信息。 Usenet新闻提供了一个传信系统，使得有兴趣的用户组可以互相保持接触。当时的电子邮件与今天的情况非常相似，主要是文本邮件。文件传送和远程登录形成了 Internet上的拥挤信息。

1988年1月，Larry Wall宣布，他编写了另一个软件工具，以替代 UNIX下的awk和sed等工具，他将它称为“ Perl ”。Perl的原始手册对它作了如下的描述：

Perl是一种解释性语言，它非常适合浏览各种文本文件，从这些文本文件中提取有关的信息，并且根据这些信息打印报表。另外，它也是非常适合执行许多系统管理任务的语言。该语言注重实用性（使用方便、有效、完整），而不注重形式上的美观（小巧、精致）。从语言创建者的观点来看，它综合了 C、sed、awk和sh等语言的某些最佳特性，因此熟悉这些语言的用户使用Perl语言是不会遇到多大困难的。（语言发展的历史也留下了 csh、Pascal甚至BASIC－PLUS的某些遗迹。）Perl的表达式句法与C语言的表达式句法非常接近。如果你有一个问题，

原先使用sed、awk或sh来解决这个问题，但是sed、awk和sh感到力不从心，或者这个问题需要运行得稍快一些，而你又不想用C语言来编写解决这个问题的程序，那么可以使用Perl。另外，也有一些翻译程序，可以将你的sed和awk脚本转换成Perl脚本。

Perl的第二个版本于1988年6月推出，它与最新的Perl版本非常相似。Perl 2的大多数特性都很容易理解和使用。它曾经是并且现在仍然是一种功能丰富而完善的编程语言。正如Perl手册所说，当时Perl的特性主要是用来进行文本处理和执行系统编程任务。

对于Perl来说，1991年是不寻常的一年。1月份，Larry Wall与Randal Schwartz撰写的《Programming Perl》一书的第一版出版。这本书曾经是（并且它后来的版本仍然是）Perl语言的权威参考书。这本书的粉红色封面上印有一只骆驼，这是Perl语言的正式标记。（骆驼并不是一种好看的动物，但是它稳健可靠，值得信赖，并且用处极大。）

这本书的出版时间恰好与Perl 4的推出时间相一致。Perl 4是第一个广泛销售的Perl版本，尽管它最后修改是在1992年，但是直到今天，我们仍然能够在Internet上的遥远角落看到它的踪影。如果你在网上遇到它，你不应使用它。

1994年10月，Perl 5问世。它推出了专用变量、引用、模块和对象等特性，其中“对象”我们尚未介绍。1996年10月，《Programming Perl》一书的第二版（“蓝色骆驼”）上市，它记录了这些新特性。

16.1.2 开放源

Perl取得成功的原因之一与Perl语言的开发和销售方式有关。Perl解释程序是一个开放源软件。开放源是软件开发人员给一个老概念赋予的新术语，它称为“免费分配的软件”。这种软件可免费提供给用户，凡是希望修改软件的源代码的人，都可以查看、调整和修改该源代码。采用这种模式的其他软件包是Linux和FreeBSD操作系统，Apache Web服务器，以及Netscape的开放源浏览器Mozilla。

使用开放源模式实际上是开发软件的一种非常有效的方式。由于开放源代码是由志愿者缩写的，因此软件中通常不会包含不必要的代码。他们认为必要的特性，就会建议纳入源代码中。这种软件的质量非常好，因为对软件有兴趣的每个人都有权并且有责任认真关注它的开发过程，以找出它存在的错误。查看该代码的人越多，错误就越少。

Eric S Raymond撰写了一系列生动的文章，来介绍开发源代码的开发模式，比如为什么它运行得这么好，为什么它在经济上非常合算，以及它是如何开发而成的。第一篇文章“权威与廉价市场”（The Cathedral and the Bazaar)对开放源开发模式如何工作进行了很好的介绍。这些文章的URL在本学时的“课时小结”这一节中列出。

Larry Wall给Perl解释程序申请了版权，因此他拥有Perl的版权，可以根据自己的意愿来处理该软件的版权。但是，与大多数软件一样，用户可以购买Perl的使用许可证。软件许可证说明了软件可以如何来使用和分销，当你打开从商店购买的软件时，会发现它是个印刷得很精美的软件。Larry Wall为你提供了两个不同的软件许可证，供你选择，即GNU普通公用许可证

和Perl艺术家许可证。当你阅读这两种软件版本的许可证后，就可以根据协议条款来选择你需要的许可证，以便将Perl转售给其他用户。

两个许可证的文本都很长，现在将它们的内容概括如下：

- 你可以将Perl解释程序的源代码转售给其他用户，并将版权声明复制给他。
- 你可以修改原来的源代码，只要将你的修改明确标为你自己做的修改，并且既可以放弃这些修改，也可以清楚地指明这不是Perl的标准版本。你也必须提供Perl的标准版本。
- 将Perl转售给别的用户时，你可以收取合理的费用。也可以收取一定的支持费用，但是不得将Perl本身销售给其他用户。你可以将Perl纳入你销售的其他产品中。
- 使用Perl编写的程序不受本许可证的约束。
- 对Perl不作任何担保。

你不得将类似上面这样的对Perl许可证条款的概述用于法律目的。这些概述只是为了使你对这两种许可证的条款有一个大致的了解。

在你想要将Perl纳入另一个软件包之前，应该亲自阅读许可证的内容，并且弄清你的行为是否符合这两种许可证的规定。Perl艺术家许可证包含在销售的每个Perl中，其文件名为Artistic(艺术家)。可以通过网址http://www.gnu.org查看GNU普通公用许可证的内容。

有了许可证，Perl就可以在开放论坛中进行开发和改进。运用这种方法，凡是想要阅读和提出修改建议的用户，都可以看到Perl的全部源代码。这种方法有助于实现出色的编程，避免陷入专用的、隐蔽的和模糊不清的软件解决方案之中。

16.1.3 Perl的开发

Perl解释程序、语言以及该语言包含的各个模块的开发是在一个邮件列表上进行的，在这个邮件列表上，Perl的开发人员可以提出修改建议，查看错误报表，对Perl源代码的修改进行争论。

凡是希望参与这个开发过程的人，均会受到欢迎，这正是开放源的特色。不过，为了防止混乱，人们提出的修改意见将由一个核心开发人员小组负责筛选，这些开发人员负责批准和拒绝人们的修改意见，并维护Perl开发的核心内容。修改意见要进行评估，看它们对Perl是否有利，这些修改究竟有多大用处，是否有的人的修改取得了成功。Larry Wall负责监督这个开发过程，担当着仁慈的总监的角色，允许进行确实有利的修改，并否决他认为对Perl不利的修改。

已经推出的Perl版本按两种方法编号。1999年8月以前，它们按照主次修补次数(major.minor_patchlevel)的格式进行编号。因此，4．036__18是指Perl第4版，第36次发布，第18次修补。有时Perl版本号不包含修补次数。截止到1999年秋撰写本书时，Perl的当前版本是5.005。

Perl的下一个版本是5.6。这个版本的编写方案比较传统，它采用主、次版本号格式。因此，Perl 5.6的下一个版本将是5.7等等。

16.2 Perl综合存档文件网（CPAN）

Perl提供了另一些模块，以便进一步扩大你的开发环境。这些模块均包含在CPAN中。

16.2.1 什么是CPAN

Perl综合存档文件网即CPAN，它是Perl文档和软件的一个大型集合体。该软件是由为 Perl语言家族编写模块、程序和文档的志愿软件人员共同开发的。

CPAN中可以使用的模块非常广泛。在撰写本书时， CPAN的建立大约已有4年历史，可供安装的模块超过3500个。这些模块涵盖的编程问题的范围极其广泛。表 16-1是个简短的列表，可以使你对CPAN中的模块有个大致的了解。

表16-1 CPAN中的部分模块一览表

TK	用于Perl程序的图形接口。可以使用特定的工具箱模块来访问特定的图形库，比如Win32 API、Gtk、Gnome、Qt或XII工具箱
Net∷*	网络模块。它们是用于Mail、Telnet、IRC、LDAP和40多个其他程序的接口
Math∷*	包含30多个模块，用于复数、快速傅立叶转换和矩阵操作的各种结构
Date∷*Time∷	* 用于将日期／时间转换成各种不同格式和将各种格式转换成日期／时间并对它们进行操作的模块
Date∷*Tree∷*、	用于对链接列表和B-树状结构之类的数据结构进行操作的模块
DBI∷*	数据库的通用结构
DBD∷*	商用和免费数据库的接口，这些数据库包括 Oracle、informix、Ingres、ODBC、MsqI、Mysql、Sybase和许多其他数据库
Term∷*	对文本模式的屏幕（如DOS的Command窗口或UNIX终端仿真程序）进行精确控制的模块
String∷*，Text∷*	包含几十个模块，用于对文本进行分析和格式化
CGI∷*	用于Web页的创建、服务、提取和分析的各个模块
URI∷*	
HTML∷*	
LWP∷*	
GD，Graphics∷*，Image∷*	用于对图形和图像进行操作的各个模块
Win32∷*，Win32API∷*	用于对Microsoft Windows进行操作的模块

需要记住的最重要一点是，对于大多数问题来说，已有的模块至少可以部分地解决某个问题。CPAN中的这些解决方案已经具有相应的代码，经过了测试，并且有许多程序员对代码进行了审查，以保证它们的正确性和完整性。

CPAN中的所有模块的版权均由各自的开发人员所拥有，因此你应该阅读每个模块所附的README文件，以了解使用模块时应该遵守的条款。大多数情况下，这些模块的销售条款与Perl本身的条款相同，也就是要按照 Perl艺术家许可证或GNU普通公用许可证的条款进销售。

CPAN也是一个标准模块的名字，它用于帮助用户将辅助模块安装到你的 Perl中。CPAN模块在本书附录“安装模块”中作了说明。

16.2.2 为什么人们愿意提供自己的开发成果

在过去半个世纪的计算机编程发展历程中，程序员一次又一次解决着一些相同的问题。从50年代以来，搜索、排序、通信、读取、写入，这些编程问题实际上没有多大的变化。关于计算机编程理论和管理的一些著作在二三十年以后仍然可用。

一次又一次地解决相同的编程问题并不总是一件有意思的事情，并且常常会产生一些质量不高的解决方案，这叫做“仿制车轮”。最终，程序员对解决一些有意思的编程问题产生了极大的兴趣。

令程序开发人员感到尴尬的一种情况是：花费了很长的时间，投入了大量精力，以便解决一个复杂的问题，结果却发现可以使用一个简单而巧妙的方案就可以解决这个问题。这种尴尬促使程序开发人员去寻求一些方法以便与其他人共享代码。共享代码带来的一个非常有趣的副产品是可以产生更好的代码，因为其他程序员将会发现你的代码中存在的问题，而你自己却没有注意到。

CPAN是Perl语言开发界为了避免进行不必要的开发工作所作努力的结果。它所包含的模块能使你不会经历“仿制车轮”的尴尬。

大多数模块的质量是很好的，模块与 Perl一样，都是在开放源生产方式下开发的。当你在系统中安装一个模块时，你将自动拥有该模块的源代码。你可以自己查看源代码，并且根据许可证条款的规定，将源代码的各个部分用于你自己的程序，并可修改源代码，甚至可以与源代码的开发者联系，提出修改建议。

从表面上看，CPAN的开发是大量程序员共同努力的结果，但开发人员为 CPAN提供自己的开发成果的实际原因是千差万别的。有时是为了帮助其他人解决类似的编程问题，有时是为了达到一个很好的目标，有时是为了得到同行们的尊重和崇敬，这是一种很强的动力。不管原因是什么，最终结果是大量的成果可以用于你自己的程序。

16.3 下一步你要做的工作

当你阅读了本书前面的三分之二的内容之后，应该对 Perl的基本概念有所了解了。不过你并没有学到该语言的全部知识。在我的书架上，至少放着 5、6本关于Perl语言的著作，除去重复的内容，它们总共有2300页左右，尽管如此，仍然有许多问题没有包括。

你无法从一种资源中学到Perl的全部知识，不过下列各节能够告诉你下面应该做的一些工作。

这里介绍的一些资源都是按照你查找资源时应该遵循的次序列出的。虽然有些例外情况，但是总的来说，按照这个顺序来学习，可以用最快的方式来解决你的问题。

16.3.1 要做的第一步工作

当你遇到一个Perl的问题时，要确定必须采取的第一步操作是很困难的。你会感到不知所措，如果你为解决这个问题花了一点儿时间之后，你可能变得心烦意乱。不过，不要慌，要相信自己能够解决所有问题的。不管怎样，这是重要的第一步。大多数人在解决一个问题时如果进展不大，就会灰心丧气，这会影响你清晰的思路，结果会把事情搞得更糟。

这时，应该暂时放下你的工作，让自己平静下来，精神放松。最终你一定能够解决你的难题。

16.3.2 最有用的工具

你的Perl工具箱中的最有用的工具就是 Perl本身。首先，你必须搞清要解决的问题究竟是个什么性质的问题。通常问题可以分为两类，一类是语句错误，另一类是逻辑错误。

如果你的问题与语句相关，通常可以将它分为两个较小的问题。一个问题是某些 Perl元素的使用不正确，另一个问题是键入错误。请运行你的程序，查看出错消息。出错消息通常能够指明Perl对出错代码行的最佳判断。你可以查看这个代码行，看看是否存在下列情况：

- Perl的出错消息是否专门指明你应该查看代码行上的哪个位置。然后请查看这个位置。Perl解释程序可以成为你查找错误的最佳助手。
- 左括号、左方括号和左花括号是否有匹配的右括号？
- 是否仔细检查了输入文字的拼写？请再检查一次。你会惊奇地发现有多少程序错误是拼写不正确造成的。
- 是否漏掉了什么东西？比如逗号或句号？
- 指定的某一行的前面一行是否正确？
- 如果你回到本书中介绍某种类型的语句这一节，能找到类似你编写的代码的示例代码吗？
- 如果你从另一个源代码中拷贝了代码，是否检查了其他位置，以便找出类似的一段代码？它可能存在错误。

如果你的Perl程序能够运行，但是它不能产生正确的结果，那么你的逻辑可能有问题。在你分析原因之前，请执行下列操作步骤：

1) 确保程序中以＃！开头的代码行包含一个 -w。

2) 确保程序顶部附近的某个位置上有 use strict。

许多明显的逻辑错误都是-w和use strict能够捕获的简单错误，请使用这些工具。如果仍有问题，请继续阅读下面的内容。

16.3.3 查找程序中的错误

如果你能够肯定你的程序的语句是正确的，但是它无法执行正确的操作，那么你就必须进行基本的程序调试。

调试程序时首先和最常用的一个方法是使用普通的 print语句。如果你在程序中小心地使用这个语句，它能够对正在运行的程序进行某种运行期诊断。请看下面这个例子中 print是如何运行的：

```
sub foo {
     my($a1, $a2)=@_;
     # Diagnostic added to see if everything's OK.
     print STDERR "DEBUG: Made it to foo with $a1 $a2\n"
}
```

请记住，当你的程序完成时，必须取出所有的 print调试语句。建议你将某个字符串插入这些语句（“ DEBUG ”），这样就可以在以后将它们全部找出来。通过输入到 STDERR文件句柄，可以将你的正常输出与诊断信息分开。如果你将原义符号 _LINE_和_FILE_纳入你的诊断信息，Perl就能输出当前代码行和文件的名字。

可以试用的另一种方法是使用 Perl调试程序，几乎可以将调试程序用于任何 Perl程序。观察程序按步骤运行的情况，会给你以很大的启发。第 12学时对Perl调试程序的使用方法作了详细的说明。

16.3.4 首先要靠自己来解决问题

如果你的程序语句完全正确，代码逻辑也没有问题，但是仍然无法得到你想要的结果，那么你应该寻求外界的帮助。首先可以通过 Perl文档来寻找解决问题的办法。

正如我们在第1学时中介绍的那样，用户安装的每个 Perl产品都配有一套完整的文档资料。如果是Perl 5.005版本，它的文档资料超过 1700页。每个模块，每个函数，以及 Perl语言的大多数特性，都在文档资料中作了介绍，并且在常见问题列表中也有相应的说明。

若要得到可用文档资料的清单，请在命令提示符处键入 perldoc perl。它列出了手册的每一节内容和对Perl的总的描述。

常见问题列表包含了初学者和专家们对 Perl编程语言提出的最常见的问题。你至少应该对它进行一次浏览，以便对里面列出的各种问题有个基本的了解，即使你并不完全理解它的答案，也值得浏览一下。

如果因为某个原因，你的系统上没有安装 Perl文档资料，或者 perldoc并没有显示该文档，那么首先应该告诉你的系统管理员，来查找该文档。正确安装该文档是非常重要的，因为在线文档与你运行的Perl版本是完全匹配的。其他任何文档很可能与此不同。

如果你无法访问在线文档，也可以通过网址 http：//www.perl.com找到该文档。

16.3.5 从别人的程序错误中吸取教训

Usenet是个分布式传信系统， 80年代初开发成功，并立即应用到方兴未艾的 Internet上。Usenet分为成千上万个讨论组，涉及的问题从医疗、园艺、信息处理、科幻小说到曲棍球比赛和电动剃须刀，无所不包。并且还有许多地区性讨论组，世界上每个地区都有。下面是 Perl的一些特定新闻组：

comp.lang.perl.announce	关于Perl的新版本、新模块和信息的新闻
comp.lang.perl.moderated	小信息量讨论组，对Perl进行适度讨论
comp.lang perl.misc	大信息量讨论组，讨论与 Perl相关的任何问题

若要读取Usenet的新闻，需要一个新闻阅读器。找到新闻阅读器并不难，可以访问任何一个软件下载站点，抓取一个新闻阅读器。可以访问若干个 Web站点，比如 deja.com或Supernews.com，它们镜像了Web格式的Usenet新闻组，只要求你拥有一个 Web浏览器就可以阅读新闻。

在这些新闻组中，人们可以提出他们在使用 Perl时遇到的问题，其他人则可以回答这些问题，并且全部是在自愿的基础上进行。另外，这里也可以讨论与 Perl有关的人们普遍感兴趣的问题。

在我的整个编程生涯中，我认为在信息处理领域中不存在什么原始的问题。你所遇到的问题，其他人以前就遇到过。关键是要找到提出问题的人和他能提供什么答案。很可能至少有一个人提出的问题与你在某个新闻组中提出的问题非常相似。

deja.com维护了关于Usenet的许多情况的在线历史记录。使用它的搜索引擎，利用几个选择准确的关键字，就能够找到你的问题的答案。

例如，你想了解如何编写用于抓取 Web页的Perl程序，可以访问 deja.com站点的Power Search屏幕，然后将下列信息填入屏幕：

```
Keywords:  fetch web page
Forum:     comp.lang.perl.misc
```

在这个例子中，你可以将所有其他域置空。当搜索结果返回时（几乎会有 100个匹配的搜索结果），大多数匹配的搜索结果均与你提的问题有关。记住，你在 Usenet中阅读文章时应该

注意下列要点：

- 并非所有答案都正确。任何人都可以提出问题，任何人也可以回答问题。你可以阅读几个答案，自己确定哪些答案有道理。你从这些答案中得到的收获是各不相同的。
- 如果你无法确定某个答案是否正确，可以根据这个答案，自己来核实有关的信息。可以访问关于该问题的在线手册页，现在你知道何处可以查找在线手册了。
- deja.com对5年中的新闻进行了存档。它提供的答案在 5年前是正确的，但现在的正确性也许要打折扣了。

16.3.6 请求他人的帮助

如果你已经查看了在线文档，参考书和 Usenet的历史信息，但是仍然没有找到问题的答案，那么就应该求助于其他人了。

请求他人的帮助应该是你最后采取的措施，当然不是你首先采取的措施。专家是回答问题的最佳人选。他们能够接受你提出的措词糟糕的问题，并且在某个时候为你的问题提供出色的解决方案。不过与我提到的所有其他资源不同的是，人不具备解决问题的无限能力。他们会感到疲倦，他们会有心情不好的日子，他们尤其会厌倦一次又一次回答同一个问题。

虽然你所问的这个人很可能知道问题的答案，但是应该记住，你要他回答你的问题必然要占用他的时间，并且是利用他的经验。在麻烦他人帮你解决问题之前，你有责任先从其他地方寻求解决问题的方法。

若要在Usenet上提出问题，必须使用新闻阅读器或者前面讲到的一种 Web新闻接口。在你提出问题时，请遵循下列原则：

1) 在你做其他事情之前，先要看一看新闻组是否拥有常见问题列表。 Perl新闻组有一个这样的列表，它是随着 Perl解释程序一道提供给你的。对于其他新闻组，请先搜索 deja.com，找出该组的常见问题的列表，然后再发布你的消息。

2) 应该将问题提供给正确的新闻组。一般的 Perl语言问题应该提供给 comp.lamg. perl.misc新闻组。与CGI相关的编程问题应该在 comp.infosystems.www.authoring.cgi上发布。通过该新闻组的常见问题列表，你就会知道是否在正确的地方发布了你的问题。

3) 为你发布的问题选择一个比较好的主题行。它应该很好地描述你要提的问题，避免毫无用处的文字（“帮帮忙”、“新问题”之类的文字都是多余的），既要有表义性，也要简明扼要。

4) 确保问题的主体包含下列元素：

a. 说明你究竟想干什么（甚至应该说明为什么要这样做）。

b. 说明到现在为止你已经做了哪些试验。

c. 说明你遇到过哪些错误。

如果你发布了你的代码的出错消息或代码的引用，那么也应该发布足够的代码，使回答问题的人能够知道你的代码的运行情况。如果你打算处理数据，则应包含一些代码行作为例子。

问题的主体不应包括下列元素：

a. 大的代码段。

b. EXE等二进制文件或uuencoded编码的文件。

c. MIME附件。相反，你可以将例子和代码纳入文本的主体中。

5) 务必发布一个有效的电子邮件地址，以备有人想要回答你的问题但又不想公开回答时使用。

6) 最重要的一点是：你应该非常有礼貌。你是寻求素不相识的人为你提供帮助。任何人都没有帮助你的义务。你应该多说“请”和“谢谢你“，并且不要使用不礼貌的评语。不要使用欺骗性手段来谋求他人的帮助，比如说“帮助一个可怜的小女孩编写她的 CGI程序”,或者说”我将为你提供一个免费的 Web页，如果你……”这些话语显得非常失礼，而且有些低声下气。

当你将文章发布在新闻组上之后，应该等待其他人提供解决方案。 Usenet新闻要花费数天时间才能传遍全世界，人们不可能跟踪和阅读每一篇文章。你应该有耐心，在等待解决方案的同时，可以再次提出这个问题。无论你做什么，不要过早在 Usenet上再次提出这个问题。至少要等待两个星期之后，再提出这个问题。你应该改变提出问题的措词，使主题行更加清楚，然后再试一次。

对你的文章的反应可能立即出现（在几分钟内），也可能在一个月或更长时间之后出现。正如在前面讲过的那样，对你提出的问题的解决方案的质量差别很大。有些很有参考价值，有些可能是错的。有些回答很有礼貌，有些则非常粗鲁。按照网络礼仪，你应该感谢为你提供解决方案的人。如果有人过分热情，你不必在意。

16.4 其他资源

如果想了解Perl、Perl编程和Perl开发界的情况，可以查看下列辅助资源：

- Larry Wall、Tom Christiansen和Randal Schwartz撰写的《Programming Perl》一书。这本书被人们视为 Perl程序员的圣经。当你学习了 Perl的基本知识后，可以将这本书作为最佳参考书来使用。
- Tom Christiansen和Nothn Torkington撰写的《The Perl Cookbook》一书。这是涉及Perl的各种问题、代码举例、解决方案以及对数百个问题的评论的总汇，它采用食谱的书写格式。它提出了每个问题，每个问题的解决办法，然后是每种解决办法的代码举例和说明。
- Perl季刊。这份季刊自称是“ Perl语言开发界之声 ”。这是一份真正的技术性刊物，里面的文章都是Perl语言开发界（每天使用 Perl的程序开发人员）的成员撰写的，而不是由权威学者或专业撰稿人撰写的。它的创刊号就声称：“我们的目标是……使之成为一份知识性出版物，以探讨 Perl的技巧，编程的技巧和其他的一些技巧…… ”

若要进一步深入了解这些问题，请查看下列信息：

Internet的发展历史：Hobbe的Internet Timeline，网址为：

```
http://www.isoc.org/zakon/Internet/History/HIT.html
```

Perl的发展历史：CPAST，网址为：

```
http://history.perl.com
```

Perl季刊，网址为：

```
http://www.tpj.com
```

CPAN，网址为：

```
http://www.perl.com/CPAN
```

在线文档：

在你的系统上，也可以查看 http：//www. Perl.com。

Eric S. Raymond的开放源文章：

http://www.netaxs.com/~esr/writings

16.5 课时小结

本学时你学习了关于 Perl的发展历史以及开放源开发模式如何用于 Perl，还学习了关于CPAN的一些知识，它为什么会存在，以及谁负责对它进行维护。最后，学习了在开发 Perl程序时遇到问题时可以使用何种类型的资源来加以解决。

16.6 课外作业

16.6.1 专家答疑

问题：如果说Web是在Perl之后问世的，那么为什么 Perl是个CGI语言？

解答：Perl之所以属于CGI语言，其原因与计算机可以用来玩游戏是相同的。原因并不在于它们为了什么目的而发明，而在于 Perl非常适合作为一种 CGI语言。下一个学时我们将要详细介绍为什么Perl是个非常好的CGI语言。

问题：我在Usenet上提出了一个编程问题，但是却得到了一个粗鲁和令人恼火的答复。我应该怎么办？

解答：首先，这个使你恼火的答复是否包含某些好的建议。如果有，那么你应该采纳这些建议，不要在乎粗鲁无礼的一面。否则你可以对这个答复不予理睬。人生苦短，不要把生命浪费在无谓的纠纷上。

问题：有没有一种简便的方法可以用来搜索 CPAN？

解答：有的。位于 http://search.cpan.com上的Web页包含一个通用搜索函数，你可以用来浏览CPAN最近的修改情况，并可按分类查看各个模块。

16.6.2 思考题

1) 关于用Perl进行CGI编程的问题首先应该传送给哪个 Usenet组？

a.comp.infosystems.www.authoring.cgi

b.comp.lang.perl.misc

2) 如果你的系统没有这个文档，应该怎么办？

a.要求系统管理员安装该文档。

b.将消息发送到comp.lang.perl.misc。

c、设法从辅助来源那里获取该文档 ,比如http://www. Perl.com。

16.6.3 解答

1) 答案是a。comp.lang.Perl.misc可以作为你提出CGI问题的第二个地方。

2) 答案是a和c。可能是先a后c这个顺序。文档可能已经安装，系统管理员可以帮你找到它。如果不行，www.Perl.com有一组文档的最新拷贝。