第二章　The Gory Details

概念型

Unicode，区分 byte 和 character，character 可能是多 bytes 的。

内置数据类型：标量$、数组@、散列%

名字，名字空间

typegolb：Sometimes you just want a name for everything named foo, regardless of its funny character. So symbol table entries can be named with an initial *, where the asterisk stands for all the other funny characters. These are called typeglobs, and they have several uses. They can also function as lvalues. Assignment to typeglobs is how perl implements importing of symbols from one symbols table to another.

Perl 保留字，Perl 的保留字为小写，所以在定义文件句柄或者标签时使用大写以避免与保留字冲突，至于变量名由于有特殊字符开头，所以不会冲突。同样用户定义的模块名也把第一个字母大写，以避免于内置模块（全部小写）冲突。

变量名可以是字母数字和下划线，可以是纯字母或纯数字，其他字符的话只能是两字符的变量名，以特殊字符开头外加一个其他字符，一般都有 perl 的预先定义好的意义。

使用 '::' 符号来指明变量名属于那个符号表

包变量和词法范围变量，前者名义上是包私有的，实际上是全局的，因为包就是全局的，后者用 my 来声明。

Here 文档：用 '<<' 来声明结束字符串，它后面的字符串用来结束 HERE 文档。字符串用的单引号还是双引号还是反引号区别是否内插和执行。

V 字符串文本，一个以字符 v 开头，后面跟着多个用句点分隔的整数的文本，会被当作一个字串文本。该字串的字符的自然数对应 v 文本里的数值。V 字串在表示版本和 IP 地址的时候很有用。

上下文环境：

1. 标量上下文和列表上下文
2. 布尔上下文
3. 空上下文
4. 内插上下文

typeglob，可以传递和存储文件句柄，可以给给符号表取别名，如：

```
$fh =  *STDOUT;
$fh = \*STDOUT;     #reference
*foo = *bar;
```

所有的 bar 为名字的类型都可以用 foo 为名字的类型代替。

```
*foo = \$bar;
```

$foo 是$bar 的别名，而@foo 不是@bar 的别名，所有这些都只影响全局变量，词法不能通过符号表记录访问。

重要型

变量内容存储需要空间，变量自身（名字和相关定义）存储也需要空间，这些地方叫做名字空间，通常被称为符号表和词法范围，也叫做 packages 和 pads。通常符号表是全局散列，包含所有全局变量（包括其他符号表的散列），相对来说，词法范围那些局部未命名的不存在于任何符号表中的中间结果，附着在你的代码后面，它们包含进能被该块所见的变量。

关于变量定位：

1. 首先 Perl 在最先结束的块里面查找，看看该变量是否用 my 或者 our 声明，如果存在 my 定义，则该变量为词法范围变量，仅属于该代码块，不存在于任何包。如果用 our 声明，则该变量是包变量，用 our 仅仅是给了它一个词法范围的别名。
2. 如果上述过程失败，则 Perl 尝试在包含该代码段的块里查找，如果 Perl 没有发现定义，那么它将重复第二步知道用光所有上层闭合块。
3. 当 perl 用光所有上层闭合块以后，它检查整个编辑单元，把它当作代码块寻找声明（一个编辑单元就是整个文件，或者被 eval STRING 操作符编译过得当前字串）。
4. 如果还没找到，Perl 放弃词法范围，假设它是一个包变量。如果你使用了 strict program，那么会报错，因为 strict 不允许使用非全称的全局名字。

几个输入输出操作符：

1. 反引号，将所引内容在 shell 中执行，并返回输出的字串，通过更改$/更改行分隔符，返回的数字状态值在$?中保存。
2. 行输入操作符<>，也叫尖脚操作符或者 readline 函数。

琐碎型

Unicode whitespace characters are allowed in a Unicode Perl Program, but you need to be careful. If you use the special Unicode paragraph and line separators, be aware that Perl may count line numbers differently than your editor does, so error messages may be more difficult to interpret. It's best to stick with good old-fashioned newlines.

perl 将忽略以＝开头的行直到=cut 结束。被忽略的部分被认为是 pod。

Construct	*Meaning*
$days	Simple scalar value
$days[28]	29th element of array @days
$days{'Feb'}	“Feb” value from hash days
${days}	Same as $days but unambiguous before alphanumerics
$Dog::days	Different $days variable, in the Dog package
$#days	Last index of array @days
$days->[28]	29th element of array pointed to by referenced $days
$days[0][2]	Multidimensional array
$days{2000}{'Feb'}	Multidimensional hash
$days{2000, 'Feb'}	Multidimensional hash emulation

@days	Array containing ($days[0], $days[1], ... , $days[n])
@days[3, 4, 5]	Array slice containing ($days[3], $days[4], $days[5])
@days[3..5]	Array slice containing ($days[3], $days[4], $days[5])
@days{'Jan'. 'Feb'}	Hash slice containing ($days{'Jan'}, $days{'Feb'})
%days	(Jan => 31, Feb => $leap?29: 28, ...)

Perl 用下划线分隔长数字

引用是强类型的。

关于引号，单引号，双引号

Customary	*Generic*	*Meaning*	*Interpolates*
"	q//	Literal string	No
“ ”	qq//	Literal string	Yes
``	qx//	Command execution	Yes
()	qw//	Word list	No
//	m//	Pattern match	Yes
s///	s///	Pattern substitution	Yes
y///	tr///	Character translation	No
“ ”	Qr///	Regular expression	Yes

应该把任何以双下划线开头和结束的标识符看作由 Perl 保留做特殊语法处理的记号。其中有两个这类特殊文本是 LINE 和__FILE__，分别意味着在你的程序某点的当前行号和文件名，__PACKAGE__是当前编译进入的包的名字，END 可以用于在真正的文件结束符之前表示脚本的逻辑结束。任何后面的文本都被忽略，不过可以通过 DATA 句柄读取。DATA 记号的作用类似于 END 记号，不过它是在当前包的名字空间打开 DATA 句柄，因此你所 require 的所有文件可以同时打开，每个文件都拥有自己的 DATA 文件句柄。

心得型