

第七章 格式

- ↓ 第七章 格式
 - ↓ 7.1.0 格式变量
 - ↓ 7.2.0 页脚
 - ↓ 7.2.1 访问格式的内部

Perl 有一个机制帮助你产生简单的报告和图表。为了实现这个机制，Perl 帮助你格式化你的输出，使它打印出来的时候看起来比较接近于你想要的结果。它能保持跟踪象一页里面有多少行，当前的页码，以及什么时候打印页头等等的东西。使用的关键字是从 **FORTRAN** 里面借来的：**format** 用来声明而 **write** 用来执行；参看第二十九章，函数，获取相关内容。所幸，布局时非常易读的，很象 **BASIC** 中的 **PRINT USING** 语句。也可以将它想象成 **nroff**(如果你知道 **nroff**，这也许不象是一个比较)。

格式输出，和包和子过程一样，是声明而不是执行，因此它们可以在你的程序中任何地方出现。(通常最好将所有的格式输出放在一起)。它们有他们自己的名字空间，与 Perl 中其它类型的名字空间是区分开来的。这就是说如果你有一个函数 **"Foo"**，但它不同于一个名字为 **"Foo"** 的格式输出。然而和一个文件句柄相关联的格式输出的缺省名字和该文件句柄的名字相同。因而，**STDOUT** 的缺省格式输出的名字是 **"STDOUT"**，文件句柄 **TEMP** 的缺省格式输出名字为 **"TEMP"**，它们看起来是一样的，实际上是不一样的。

输出纪录格式输出象下边一样定义：

```
format NAME =  
    FORMLIST  
    .
```

如果省略 **NAME**，将定义格式输出 **STDOUT**。 **FORMLIST** 由一些有序的行组成，每一行都是下面三种类型中的一种：

1. 注释，以第一列为 **#** 来表示。
2. 一个格式行，用来定义一个输出行的格式
3. 参数行，用来向前面的格式行中插入值

格式行除了那些需要被替换的部分外，严格按照它们的声明被输出。(注：而且，甚至那些你放进去维护列完整性的域也如此。在一个格式行中没有任何东西可以导致域的伸缩或者移位。你看到的列是按照 **WYSIWYG** 的概念分布的---假设你用的是定宽字体。就连控制字符都假设是宽度为一的字符。) 格式行中每个被替换的部分分别以 **@** 或者 **^** 开头。这些行不作任何形式的变量代换。**@** 域(不要同数组符号 **@** 相混淆)是普通的域。另一种域，**^** 域用来进行多行文本块填充。域的长度通过在格式符号 **@**，**^** 后跟随特定长度的 **<**，**>**，**|** 来定义，同时，**<**，**>**，**|** 还分别表示，左对齐，右对齐，居中对齐。如果变量超出定义的长度，那么它将被截断。

作为右对齐的另外一种方式，你可以使用 **#**(在 **@** 或 **^** 后边)来指定一个数字域。你可以在这种区域中插入一个 **.** 来制定小数点的位置。如果这些区域的值包含一个换行符，那么只输出换行符前面的文本。最后，特殊的域 **@*** 可以被用来打印多行不截断的值；这种区域通常在一个格式行中出现。

参数行指定参数的顺序必须跟相应的格式行的域顺序一致。不同参数的表达式需要使用逗号分隔。参数行被处理之前所有的参数表达式都在列表环境中求值，因此单个列表表达式会产生多个列表元素。通过使用圆括弧将表达式括起来，可以使表达式扩展到多行(因此，圆括弧必须是第一行的第一个标志)。这样就可以将值同相应的格式域对应起来方便阅读。


```
use FileHandle;
OUTF->format_name("My_Other_Format");
OUTF->format_top_name("My_Top_Format");
```

```
format Ident =  
    @<<<<<<<<<<<<  
    commify($n)  
.
```

```
format Ident =
I have an @ here.
      "@"
```

```
format Ident =
@|
                                     "Some text line"
.
```

```
$format = "format STDOUT = \n"
        . '^' . '<' x $cols . "\n"
        . '$entry' . "\n"
        . "\t^" . "<" x ($cols-8) . "~\n"
        . '$entry' . "\n"
        . ".\n";

print $format if $Debugging;
eval $format;
die $$ if $@;
```

[illegible]

<http://www.pgsql.db.org/twiki/bin/view/Perl/Formats?skin=print.pattern> 2006-01-16

7.2.0 页脚

7.2.1 访问格式的内部

<http://www.pgsql.db.org/twiki/bin/view/Perl/Formats?skin=print.pattern> 2006-01-16

如果你在使用 [FileHandle²](#) 模块, 你可以使用象下边代码一样使用 `formline`, 使一块文本在第 72 列处折行.

```
use FileHandle;  
STDOUT->formline("^" . ("<" x 72) . "~~\n", $long_text);
```

Revision: r1.1 - 22 Aug 2005 - 12:53 - [TingYu](#)

[Perl](#) > [PerlProgramming3](#) > [P3GoryDetail](#) > [Formats](#)

版权 © 1999-2006 归这里所有作者. [PostgreSQL](#) 的中文文档版权归何伟平所有.
向为这里贡献想法,文章的人致敬 [PostgreSQL](#) 中文网
[反馈意见](#)