Perl 特殊变量 Perl 发送邮件

Perl 正则表达式

正则表达式(regular expression)描述了一种字符串匹配的模式，可以用来检查一个串是否含有某种子串、将匹配的子串做替换或者从某个串中取出符合某个条件的子串等。

Perl语言的正则表达式功能非常强大，基本上是常用语言中最强大的，很多语言设计正则式支持的时候都参考Perl的正则表达式。

Perl的正则表达式的三种形式，分别是匹配，替换和转化:

- 匹配：m//（还可以简写为//，略去m）
- 替换：s///
- 转化：tr///

这三种形式一般都和 **=~** 或 **!~** 搭配使用， =~ 表示相匹配，!~ 表示不匹配。

匹配操作符

匹配操作符 m// 用于匹配一个字符串语句或者一个正则表达式，例如，要匹配 标量 $bar 中的 "run",代码如下所示：

实例

```
#!/usr/bin/perl
$bar = "I am runoob site. welcome to runoob site.";
if ($bar =~ /run/){
print "第一次匹配\n";
}else{
print "第一次不匹配\n";
}
$bar = "run";
if ($bar =~ /run/){
print "第二次匹配\n";
}else{
print "第二次不匹配\n";
}
```

执行以上程序，输出结果为：

```
第一次匹配
第二次匹配
```

模式匹配修饰符

模式匹配有一些常用的修饰符，如下表所示：

修饰符	描述

i	忽略模式中的大小写
m	多行模式
o	仅赋值一次
s	单行模式，"."匹配"\n"（默认不匹配）
x	忽略模式中的空白
g	全局匹配
cg	全局匹配失败后，允许再次查找匹配串

正则表达式变量

perl处理完后会给匹配到的值存在三个特殊变量名:

- **$`:** 匹配部分的前一部分字符串
- **$&:** 匹配的字符串
- **$':** 还没有匹配的剩余字符串

如果将这三个变量放在一起,你将得到原始字符串。

实例如下：

实例

```
#!/usr/bin/perl
$string = "welcome to runoob site.";
$string =~ m/run/;
print "匹配前的字符串: $`\n";
print "匹配的字符串: $&\n";
print "匹配后的字符串: $'\n";
```

执行以上程序输出结果为：

```
匹配前的字符串: welcome to
匹配的字符串: run
匹配后的字符串: oob site.
```

替换操作符

替换操作符 s/// 是匹配操作符的扩展，使用新的字符串替换指定的字符串。基本格式如下：

```
s/PATTERN/REPLACEMENT/;
```

PATTERN 为匹配模式，REPLACEMENT 为替换的字符串。

例如我们将以下字符串的 "google" 替换为 "runoob":

实例

```
#!/usr/bin/perl
$string = "welcome to google site.";
$string =~ s/google/runoob/;
print "$string\n";
```

执行以上程序输出结果为：

```
welcome to runoob site.
```

替换操作修饰符

替换操作修饰符如下表所示：

修饰符	描述
i	如果在修饰符中加上"i"，则正则将会取消大小写敏感性，即"a"和"A" 是一样的。
m	默认的正则开始"^"和结束"$"只是对于正则字符串如果在修饰符中加上"m"，那么开始和结束将会指字符串的每一行：每一行的开头就是"^"，结尾就是"$"。
o	表达式只执行一次。
s	如果在修饰符中加入"s"，那么默认的"."代表除了换行符以外的任何字符将会变成任意字符，也就是包括换行符！
x	如果加上该修饰符，表达式中的空白字符将会被忽略，除非它已经被转义。
g	替换所有匹配的字符串。
e	替换字符串作为表达式

转化操作符

以下是转化操作符相关的修饰符：

修饰符	描述
c	转化所有未指定字符
d	删除所有指定字符
s	把多个相同的输出字符缩成一个

以下实例将变量 $string 中的所有小写字母转化为大写字母：

```
#!/usr/bin/perl

$string = 'welcome to runoob site.';
$string =~ tr/a-z/A-z/;

print "$string\n";
```

执行以上程序输出结果为：

```
WELCOME TO RUNOOB SITE.
```

以下实例使用 /s 将变量 $string 重复的字符删除：

实例

```
#!/usr/bin/perl
$string = 'runoob';
$string =~ tr/a-z/a-z/s;
print "$string\n";
```

执行以上程序输出结果为：

```
runob
```

更多实例：

```
$string =~ tr/\d/ /c;     # 把所有非数字字符替换为空格
$string =~ tr/\t //d;     # 删除tab和空格
$string =~ tr/0-9/ /cs    # 把数字间的其它字符替换为一个空格。
```

更多正则表达式规则

表达式	描述
.	匹配除换行符以外的所有字符
x?	匹配 0 次或一次 x 字符串
x*	匹配 0 次或多次 x 字符串,但匹配可能的最少次数
x+	匹配 1 次或多次 x 字符串,但匹配可能的最少次数
.*	匹配 0 次或多次的任何字符
.+	匹配 1 次或多次的任何字符

{m}	匹配刚好是 m 个 的指定字符串
{m,n}	匹配在 m个 以上 n个 以下 的指定字符串
{m,}	匹配 m个 以上 的指定字符串
[]	匹配符合 [] 内的字符
[^]	匹配不符合 [] 内的字符
[0-9]	匹配所有数字字符
[a-z]	匹配所有小写字母字符
[^0-9]	匹配所有非数字字符
[^a-z]	匹配所有非小写字母字符
^	匹配字符开头的字符
$	匹配字符结尾的字符
\d	匹配一个数字的字符,和 [0-9] 语法一样
\d+	匹配多个数字字符串,和 [0-9]+ 语法一样
\D	非数字,其他同 \d
\D+	非数字,其他同 \d+
\w	英文字母或数字的字符串,和 [a-zA-Z0-9_] 语法一样
\w+	和 [a-zA-Z0-9_]+ 语法一样
\W	非英文字母或数字的字符串,和 [^a-zA-Z0-9_] 语法一样
\W+	和 [^a-zA-Z0-9_]+ 语法一样
\s	空格,和 [\n\t\r\f] 语法一样
\s+	和 [\n\t\r\f]+ 一样
\S	非空格,和 [^\n\t\r\f] 语法一样
\S+	和 [^\n\t\r\f]+ 语法一样
\b	匹配以英文字母,数字为边界的字符串

\B	匹配不以英文字母,数值为边界的字符串
a|b|c	匹配符合a字符 或是b字符 或是c字符 的字符串
abc	匹配含有 abc 的字符串 (pattern) () 这个符号会记住所找寻到的字符串,是一个很实用的语法.第一个 () 内所找到的字符串变成 $1 这个变量或是 \1 变量,第二个 () 内所找到的字符串变成 $2 这个变量或是 \2 变量,以此类推下去.
/pattern/i	i 这个参数表示忽略英文大小写,也就是在匹配字符串的时候,不考虑英文的大小写问题. \ 如果要在 pattern 模式中找寻一个特殊字符,如 "*",则要在这个字符前加上 \ 符号,这样才会让特殊字符失效

Perl 特殊变量 Perl 发送邮件