Perl 文件操作 Perl 错误处理

Perl 目录操作

以下列出了一些操作目录的标准函数：

```
opendir DIRHANDLE, EXPR  # 打开目录
readdir DIRHANDLE        # 读取目录
rewinddir DIRHANDLE      # 定位指针到开头
telldir DIRHANDLE        # 返回目录的当前位置
seekdir DIRHANDLE, POS   # 定位指定到目录的 POS 位置
closedir DIRHANDLE       # 关闭目录
```

显示所有的文件

显示目录下的所有文件，以下实例使用了 glob 操作符，演示如下：

实例

```
#!/usr/bin/perl
# 显示 /tmp 目录下的所有文件
$dir = "/tmp/*";
my @files = glob( $dir );
foreach (@files ){
print $_ . "\n";
}
# 显示 /tmp 目录下所有以 .c 结尾的文件
$dir = "/tmp/*.c";
@files = glob( $dir );
foreach (@files ){
print $_ . "\n";
}
# 显示所有隐藏文件
$dir = "/tmp/.*";
@files = glob( $dir );
foreach (@files ){
print $_ . "\n";
}
# 显示 /tmp 和 /home 目录下的所有文件
$dir = "/tmp/* /home/*";
@files = glob( $dir );
foreach (@files ){
print $_ . "\n";
}
```

以下实例可以列出当前目录下的所有文件：

实例

```
#!/usr/bin/perl
opendir (DIR, '.') or die "无法打开目录, $!";
```

```
while ($file = readdir DIR) {
print "$file\n";
}
closedir DIR;
```

如果你要显示 /tmp 目录下所有以 .c 结尾的文件，可以使用以下代码：

实例

```
#!/usr/bin/perl
opendir(DIR, '.') or die "无法打开目录, $!";
foreach (sort grep(/^.*\.c$/,readdir(DIR))){
print "$_\n";
}
closedir DIR;
```

创建一个新目录

我们可以使用 **mkdir** 函数来创建一个新目录，执行前你需要有足够的权限来创建目录：

实例

```
#!/usr/bin/perl
$dir = "/tmp/perl";
# 在 /tmp 目录下创建 perl 目录
mkdir( $dir ) or die "无法创建 $dir 目录, $!";
print "目录创建成功\n";
```

删除目录

我们可以使用 **rmdir** 函数来删除目录，执行该操作需要有足够权限。另外要删除的目录必须的空目录：

实例

```
#!/usr/bin/perl
$dir = "/tmp/perl";
# 删除 /tmp 目录下的 perl 目录
rmdir( $dir ) or die "无法删除 $dir 目录, $!";
print "目录删除成功\n";
```

切换目录

我们可以使用**chdir** 函数来切换当期目录，执行该操作需要有足够权限。实例如下：

实例

```
#!/usr/bin/perl
$dir = "/home";
# 将当期目录移动到 /home 目录下
chdir( $dir ) or die "无法切换目录到 $dir , $!";
print "你现在所在的目录为 $dir\n";
```

执行以上程序，输出结果为：

```
你现在所在的目录为 /home
```

← Perl 文件操作 Perl 错误处理 →

1 篇笔记

写笔记

Perl 递归目录其实可以用更简单的 `File::Find` 函式达成。

I. 實例

```
use warnings;
use utf8;

require File::Find;

# 宣告 STDOUT, STDIN, STDERR 都使用

# UTF-8 編碼方式。

binmode(STDOUT, ":encoding(utf8)");
binmode(STDIN, ":encoding(utf8)");
binmode(STDERR, ":encoding(utf8)");

# 用來顯示資料夾資訊的回呼函式。
sub callback {
  # Perl 假定 $_ 為預設的字串，所以 unless (-d) 等價於 unless (-d $_)
  # 參閱：http://www.runoob.com/perl/perl-special-variables.html (中文)
  #  或：https://perldoc.perl.org/perlvar.html (英文，`General Variables' 部份)
  #
  # -d 代表 directory 目錄，而 unless 代表「與條件相反才執行」。
  unless (-d) {
    print "檔案所在目錄的絕對路徑：$File::Find::dir\n";
    print "檔案所在位置的絕對路徑：$File::Find::name\n";
    print "檔案名稱：$_\n";
  };
};
# 這是要遞迴目錄的陣列。
# 此處假設要遞迴 /usr 和 ~/ (家目錄) 資料夾。
my @recursiveFolder = qw(/usr ~/);
# Let's recursive it!
# 要注意傳入的 sub 子程序必須要是個參照 (reference)
# 而非只是個普通的 sub 子程序。
File::Find::find(\&callback, @recursiveFolder);
```

II. 說明

File::Find 是一個應該能相容最近 Perl 版本的函式（我自己並沒找到 File::Find 最早於哪版引進，如果資料有誤也請幫忙修正 orz）。這個函式功能十分專一，也就只有遞迴功能。

以上的範例肯定看不出些什麼有用之處，但假如我們改一下例子 -- 只遞迴 MO 檔（一種已編譯翻譯檔的格式，先別在意）：

```
sub callback {
  unless (-d || $_ !~ /.mo$/) {
    print "檔案所在目錄的絕對路徑: $File::Find::dir\n";
    print "檔案所在位置的絕對路徑: $File::Find::name\n";
    print "檔案名稱: $_\n";
  };
};
```

這麼寫就代表著「除了目錄和不以 .mo 作為結尾的檔案」。

III. 結尾

使用 File::Find 能讓遞迴檔案和資料夾的方式更加簡單。若你的 Perl 有遞迴需求，建議不要自己花時間重造輪子，直接用這個函式能事半功倍。

pan93412 1个月前 (02-08)