JPERLについて

2014年2月12日
13:29

技術開発部にあるJPERLでファイル操作を行う事が多いここでは、JPERLでよく使う事がらについてまとめます。注意点として、配列のリファレンスとかローカル変数に使用するmyが使えません。

〇JPERL設定
JPERL.EXEがeb/ebFramework/bin/JPERL.EXEにあります。
dosプロンプトで使えるようにPATHを指定してください。

〇コマンドライン引数
@ARGVでコマンドライン引数を取得できます。
コマンド個数は

```
	$command_num = @ARGV;
	print $command_num."\n";
```

コマンド内容は

```
	foreach $command (@ARGV){
		print $command."\n";
	}
```

〇配列（リスト）

```
@tmp1 = (1, 2, 3);
@tmp2 = ("apple", "orange", "grape");

print $tmp1[1]; # 2
print $tmp2[0]; #apple
```

〇連想配列（ハッシュ）

```
%tmp = ("apple",1,"orange",2,"grape"3);
print $tmp{"apple"}; # 1

#キーを取得する
@tmp_keys = keys(%tmp);
foreach(@tmp_keys){
	print $_;
}
#値を取得する
@tmp_values = values(%tmp);
foreach(@tmp_values){
	print $_;
}
#キーと値を取得する
while(($key,$value)= each %tmp){
	print "$key,$value";
}
```

※連想配列はperl内部で順序が決まるためkeys,valuesで取得した値は必ずしも初期化時の順番になっていない事に注意が必要です。

〇パターンマッチ（/パターン/）

```
#文字の中にパターンが含まれている場合
$word = "apple";
print (($word =~ /a/) ? "match!" : "not match"); #match!

#$_の中にパターンが含まれている場合
```

```
$_ = "apple";
print ((/a/) ? "match!" : "not match"); #match!
```

〇置換（s/パターン１/パターン２/）
#文字の中にパターンが含まれている場合

```
$word = "This is a pen.";
$word =~ s/pen/book/;
print $word; #This is a pen.

#$_の中にパターンが含まれている場合
$_ = "This is a pen.";
s/pen/book/;
print $_;#This is a book.
```

〇open関数でパイプ処理
open関数のファイル名の部分に"command |"もしくは"| command"としてcommand部分にコマンドラインへの出力文字を入れる事でパイプ処理を行う事が出来ます。

```
#"command|"の場合は、コマンドの実行結果をファイルハンドルに返します。
open(IN, "ls -1");
while(<IN>){
	print _$;#「ls -1」で得られる結果が文字列として表示される。
}
close(IN);

#"| command"の場合は、ファイルハンドルからコマンドラインへ出力します。
open FH, '|/usr/sbin/sendmail -t user@foo.com' or die "Can't Open\n";
print FH 'From: user@foo.com', "\n";
print FH 'To: user@bar.com', "\n";
print FH "Subject: test mail.\n\n";
print FH "Good!.\n";
close FH;
```

貼り付け元　<http://rfs.jp/sb/perl/02/08.html>

〇再帰的処理
ディレクトリ内のファイル名を再帰的に取得する。
参考サイト<http://stepbystep.wiki2.jp/?p=155>

```
#ディレクトリ内のファイル名を再帰的に取得する
sub recursive{
	local($base_dir) = @_;
	local(@result);
	local(@files);

	opendir(DIRHANDLE, $base_dir);
	foreach(readdir(DIRHANDLE)){
		next if /^\.{1,2}$/;#.および..を無効にする
		push(@files, $base_dir."/".$_);
	}
	closedir(DIRHANDLE);

	foreach $filename(@files){
		if(-d $filename){
			#ディレクトリの場合は再帰的に処理する
			push(@result, &recursive($filename));
		} elsif(-f $filename){
			push(@result, $filename);
		}
```

```
	}
	return @result;
}
@ret = &recursive(".");
#結果出力
foreach(@ret){
	print $_."\n";
}
```

※perlにリファレンスというものがあり、C言語でいうところの参照渡しだと思います。ただ、現状のJPERL.EXEではリファレンスが出来ないようです。そのため、上記のプログラムのように戻り値に結果を渡しています。

〇文字列操作
よく使う文字列操作関数をまとめてみました。

・length
	文字列の長さを取得します。
	例）length("This is a pen.");#14
・substr
	文字列の部分文字列を取得または置換します。
	文字列取得　例）
		$word = "This is a pen.";
		substr($word, 10, 3);#pen
	文字列置換　例）
		$word = "This is a pen.";
		substr("This is a pen.", 10, 3)="book";
		print $word;#This is a book.
・split
	文字列を指定のパターンで分割します。
	例）以下は空白文字で分割します。
		$word = "This is a pen.";
		@subword = split(/ /, $word);#"This","is","a","pen."
・index
	文字列の中に部分文字列が含まれるか検索し、最初の位置を返す。
	例）index("This is a pen.", "pen"); #10

〇関数
　よく使う関数についてまとめてみました。

・system
	外部コマンドを実行します。
	例）system("copy /b a.bmp b.bmp");

・unlink
	ファイルを削除します。
	例）unlink("data.txt");
・chop
	文字列の最後の１文字を削除します。
	例）chop($word);

・参考サイト
http://rfs.jp/sb/perl/02/08.html
http://www.kent-web.com/perl/chap7.html
http://www.lr.pi.titech.ac.jp/~abekawa/perl/perl_lecture.html
http://stepbystep.wiki2.jp/?p=155