Perl入門～サンプルコードによるPerl入門～ A! RSS

サンプルコード中心のPerlの入門サイトです。現代的なPerl(v5.8.1以降)の書き方に準拠しています。

- サイトマップ - サンプルコードによるPerl入門の読み方
- 最近の活動 - 筆者の最近の活動

今日のひとこと: 基礎文法最速マスターシリーズがたくさん誕生しました。火付け役になってしまったようです。

〈今年勉強したいこと | Ruby基礎文法最速マスターってい...〉

2009-12-26

Perl基礎文法最速マスター 18

Perlの文法一覧です。他の言語をある程度知っている人はこれを読めばPerlの基礎をマスターしてPerlを書くことができるようになっています。簡易リファレンスとしても利用できます。

1. 基礎

文法チェック

文法をチェックを厳しくするために最初に必ず次の2行を書くようにします。プログラムが動けばよいからという考えでこの2行を書かないというのはよくないです。この2行を加えることで事前に文法チェックが行われるので、ソースコードを記述する作業は速くなり、コードの品質も上がります。

```
use strict;
use warnings;
```

print文

print文です。

```
print "Hello world";
```

コメント

コメントです。

```
# コメント
```

変数の宣言

変数の宣言です。Perlにはスカラ変数、配列変数、ハッシュ変数があります。

```
# スカラ変数
my $num;

# 配列変数
my @students

# ハッシュ変数
my %month_num;
```

スクリプトの実行

スクリプトを実行するにはコマンドラインで次のようにします。

```
perl script.pl
```

出力結果をファイルに書き出すにはリダイレクトを使います。

```
perl script.pl > file.txt
```

コンパイルチェック

事前にコンパイルチェックを行うにはコマンドラインで次のようにします。

```
perl -c script.pl
```

デバッガの起動

デバッガを起動するにはコマンドラインで次のようにします。

```
perl -d script.pl
```

2. 数値

数値の表現

スカラ変数に数値を代入できます。整数でも小数でも代入できます。桁数が大きい場合はアンダーバーを区切り文字として利用できます。

```
my $num = 1;
my $num = 1.234;
my $num = 100_000_000;
```

四則演算

四則演算です。

```
$num = 1 + 1;
$num = 1 - 1;
$num = 1 * 2;
$num = 1 / 2;
```

余りと商の求め方です。商を求めるには普通の除算を行った後にint関数で整数部を取り出します。

```
# 商
$div = int(3/2);

# 余り
$mod = 3 % 2;
```

インクリメントとデクリメント

インクリメントとデクリメントです。

```
# インクリメント
$i++

# デクリメント
$i--
```

3. 文字列

文字列の表現

文字列はシングルクォートかダブルクォートで囲みます。ダブルクォートの中では¥t(タブ)や¥n(改行)などの特殊文字を利用することができます。またダブルクォートで囲まれた文字列の中では変数展開することができます。

```
my $str1 = 'abc';
my $str2 = "def";
my $str3 = "a¥tbc¥n";

# 変数展開(結果は abc def)
my $str4 = "$str1 def";
```

文字列操作

各種文字列操作です。

2009-12-26 perlcodesample
2009-12-26 konisimple
2009-12-26 chaichanPaPa
2009-12-26 oooooooo

最近のトラックバック

2009-12-26 Ruby基礎文法最速マスター
2009-12-26 なんとなく日記 - 汎用スクリプト言語
2009-12-26 XSLT2.0基礎文法最速(?)マスター
2009-12-26 なんとなく日記 - 基礎文法最速マスターシリーズのまとめ
2009-12-26 Lua基礎文法最速マスター

リンク

Perlブログ
Perlソースコード
Perlプログラマー
perldoc.perl.org
CPAN
PAUSE
Yokohama.pm
Kansai.pm
Fukuoka.pm
Yuki Kimoto Perl Informations
Planet Perl Iron Man

Created by 木本 裕紀

```
# 結合
my $join1 = 'aaa' . 'bbb';

my $join2 = join ',', 'aaa', 'bbb', 'ccc';

# 分割
my @record = split /,/, 'aaa,bbb,ccc';

# 長さ
my $length = length 'abcdef';

# 切り出し
my $substr = substr('abcd', 0, 2); # ab

# 検索
my $result = index('abcd', 'cd'); # 見つかった場合はその位置、蜜からなった場合は-1が返る
```

4. 配列

配列変数の宣言と代入

配列です。

```
# 配列の宣言
my @array;

# 配列への代入
@array = (1, 2, 3);
```

配列の要素の参照と代入

配列の要素を参照と代入です。

```
# 要素の参照
$array[0];
$array[1];

# 要素の代入
$array[0] = 1;
$array[1] = 2;
```

配列の個数

配列の個数を取得するには配列をスカラコンテキストで評価します。Perlにしか見られない少し特殊な文法です。

```
my $array_num = @array;
```

配列の操作

配列を操作する関数です。

```
my @array = (1, 2, 3);

# shift(先頭を取り出す)
my $first = shift @array; # $firstは1

# unshift(先頭に追加)
unshift @array, 5; # @arrayは (5, 2, 3);

# pop(末尾を取り出す)
my $last = pop @array; # @arrayは (5, 2)

# push(末尾に追加)
push @array, 9; # @arrayは (5, 2, 9)
```

5. ハッシュ

ハッシュ変数の宣言と代入

```
my %hash;
%hash = (a => 1, b => 2);
```

ハッシュの要素の参照と代入

ハッシュの要素の参照と代入です。

```
# 要素の参照
$hash{a};
$hash{b};

# 要素の代入
$hash{a} = 5;
$hash{b} = 7;
```

ハッシュに関する関数

```
# キーの取得
@keys = keys %hash;

# 値の取得
@values = values %hash;

# キーの存在確認
exists $hash{a};

# ハッシュのキーの削除
delete $hash{a};
```

6. 制御文

if文

if文です。

```
if (条件) {

}
```

if ～ else文

if ～ else文です。

```
if (条件) {

}
else {

}
```

if ～ elsif 文

if ～ elsif文です。他の言語のようにelse ifではなくelsifであることに注意しましょう。

```
if (条件) {

}
elsif (条件) {

}
```

while文

while文です。

```
my $i = 0;
while ($i < 5) {

    # 処理

    $i++;
}
```

for文

for文です。

```
for (my $i = 0; $i < 5; $i++) {

}
```

foreach文

foreach文です。配列の各要素を処理できます。for文よりも好んで利用されます。(実際はperlはforeachはforのエイリアスになっていてまったく同じものです)

```
foreach my $field (@fields) {

}
```

比較演算子

比較演算子の一覧です。Perlでは数値比較と文字列比較は厳密に区別されます。

数値比較演算子の一覧です。

```
$num1 == $num2 # $num1は$num2と等しい
$num1 != $num2 # $num1は$num2と等しくない
$num1 < $num2  # $num1は$num2より小さい
$num1 > $num2  # $num1は$num2より大きい
$num1 <= $num2 # $num1は$num2以下
$num1 >= $num2 # $num1は$num2以上
```

文字列比較演算子の一覧です。

```
$str1 eq $str2 # $num1は$num2と等しい
$str1 ne $str2 # $num1は$num2は等しくない
$str1 lt $str2 # $num1は$num2より小さい
$str1 gt $str2 # $num1は$num2より大きい
$str1 le $str2 # $num1は$num2以下
$str1 ge $str2 # $num1は$num2以上
```

7. サブルーチン

Perlでは関数のことをサブルーチンともいいます。サブルーチンを作るには次のようにします。Perlでは他の言語のように明示的に引数名を指定することはありません。@_という配列に引数が入っているので自力で取り出します。戻り値を返却するにはreturnを使用します。

Perlでは戻り値としてスカラ変数だけではなく、配列変数やハッシュ変数を返すこともできます。

```
sub sum {
    my ($num1, $num2) = @_;

    my $total = $num1 + $num2;

    return $total;
}
```

8. ファイル入出力

ファイル入出力です。以下が雛形になります。ファイルをオープンすると、$fhにファイルハンドルが入ってきます。'<'は読み込みモードです。書き込む場合は'>'とします。<$fh>の部分はダイヤモンド演算子と呼ばれ、ファイルから一行読み込みます。or の後ははファイルオープンに失敗した場合に実行されます。dieはエラーメッセージを表示してプログラムを終了する関数です。$!はファイルオープンに失敗した場合のOSからのエラーメッセージです。

```
open my $fh, '<', $file
  or die "Cannot open '$file': $!";

while(my $line = <$fh>) {

}

close $fh;
```

知っておいたほうがよい文法

Perlでよく出てくる知っておいたほうがよい文法の一覧です。

Perlの真偽値

Perlで偽と判断される値は、「undef」「''」「0」「'0'」の4種類です。これ以外は真になります。

defined 値が定義されているかどうかの判定

値が定義されているかどうかを調べるにはdefined関数を使用します。

```
defined $num;
```

コマンドライン引数

コマンドライン引数を受け取るには@ARGV変数を使用します。

```
my ($file, $options) = @ARGV;
```

ひとつだけ引数受け取る場合は次のように書くことができます。暗黙的に@ARGVがshiftの引数に渡されます。

```
my $file = shift;
```

スカラコンテキストとリストコンテキスト。

Perlでは文脈によって戻り値が異なる関数があります。たとえばlocaltime関数などです。

```
# スカラコンテキスト(日付・時刻を文字列で取得)
my $time_str = localtime();

# リストコンテキスト(日付・時刻の各要素を配列として取得)
my @datetime = localtime();
```

unless文

unlessはif文の否定を表現します。

```
unless (条件) {

}
```

後置のif, 後置のunless

Perlではifやunlessを文の後ろに置くことができます。

```
# 後置のif
print $num if $num > 3;

# 後置のunless
die "error" unless $num;
```

配列スライスとハッシュスライス

配列スライスとハッシュスライスを使うと指定した要素のみを配列として取得できます。

```
# 配列スライス
@select = @array[1, 4, 5];

# ハッシュスライス
@select = @hash{'a', 'b', 'd'};
```

map

mapを使うと配列の各要素を変換することができます。$_には@arrayの要素が順に代入されます。

```
@mapped = map { $_ * 2 } @array;
```

grep

grepを使うと条件に一致した要素のみを配列として取得できます。$_ には@arrayの要素が順に代入されます。

```
@select = grep { $_ =~ 'cat' } @array;
```

リスト代入

リスト代入と呼ばれる代入方法があります。

```
my ($num1, $num2) = ($num3, $num4);
```

範囲演算子

整数の範囲を表現する範囲演算子と呼ばれるものがあります。

```
my @numes = (0 .. 5);
```

文字列リスト演算子

文字列のリストを簡単に書く構文があります。

```
my @strs = qw/aaa bbb ccc/;
```

単独のreturn

サブルーチンの中でreturnと書くと、スカラコンテキストの場合はundefが返却され、リストコンテキストの場合は空のリスト()が返却されます。戻り値を使ってサブルーチンでエラーが発生したことを伝えたい場合はreturn undef とは書かないで return と書くようにします。

```
sub name {
    my @args = @_;

    return;
}
```

例外処理

例外を投げるにはdieを使用します。

```
die "Error message";
```

例外を捕獲するにはevalを使用して$@を判定します。

```
eval { dieが発生する可能性のある関数など };

if ($@) {

}
```

ファイルからすべて読み込む

ファイルからすべて読み込む関数はないので行をすべて配列に読み込んでjoinでつなげます。

```
my @lines = <$fh>;
my $content = join '', @lines;
```

これは一行で書けます。

```
my $content = join '', <$fh>;
```

3項演算子

3項演算子です。下のサンプルの場合は$flgが真値の場合は1が、偽値の場合は2が$numに代入されます。

```
my $num = $flg ? 1 : 2;
```

||=

左辺値が偽値の場合に右辺値を代入します。下のサンプルの場合は$numが偽値であった場合に2が$numに代入されます。

```
$num ||= 2;
```

モジュールの読み込み

モジュールを読み込むにはuseを使います。

```
use モジュール名;
```

よく使用する関数・モジュール一覧

よく使用する関数・モジュール一覧です

現代的なPerlの記述方法

現代的なPerlの記述方法の解説です

Permalink | コメント(10) | トラックバック(45) | 23:42 1272 users

コメントを書く

tokuhirom 2010/01/24 00:49
Perl5 において forとforeachは同義なので、その点についてふれた方が親切かとおもいます。

for my $field (@fields) { }
も
foreach (my $i = 0; $i < 5; $i++) { }
もうごきます。

名無しさん 2010/01/24 01:20
"配列スライスとハッシュスライス"の項が色々とおかしいです。
ハッシュの例と配列の例が反対＆この状況なら、a, b, cはクオートするべき。

名無しさん 2010/01/24 01:40
shift()は呼び出し元に応じてもう一つ意味があって、shift(@_)

laclefdor 2010/01/24 04:47
Learning Perlに準じた解説記事ですね。大変分かりやすいです。

perlcodesample 2010/01/24 07:30
>tokuhiromさん
foreach と for がPerlでは実は同じという点を追記しました。

>名無しさん
配列スライスとハッシュスライスのサンプルを修正しました。

ありがとうございます。

fuge- 2010/01/24 08:26
蜜からなった

ooooooo 2010/01/24 10:36
grep のサンプルが
@select = map { $_ =~ 'cat' } @array;
と map になってるよ

chaichanPaPa 2010/01/24 11:46
「配列の操作」のとこで、%hash宣言の残骸がありますよ

konisimple 2010/01/24 14:48
わかりやすかったです！
日本語の解説読むよりサンプルコード見た方が早いですもんね！
ありがとうございます！

perlcodesample 2010/01/24 14:49
>oooooooさん
mapの間違い修正しました。

>chaichanPaPaさん
残骸を削除しました。

ありがとうございます。

なまえ メール(非公開) URL

画像認証

投稿

トラックバック – http://d.hatena.ne.jp/perlcodesample/20091226/1264257759

Perl入門〜サンプルコードによるPerl入門〜 – サンプルコードによる...
燈明日記 – Perlの基本の基礎がよくわかる
Movable Type 4 で使用できるプラグイン
[TODAY's TOPIC] January 24, 2010
omoisanのmemo – perl入門参考
atqの日記 – Perl基礎文法
Perl基礎文法
src's note – 気になる技術メモ
Perl基礎文法最速マスター – Perl入門?サンプルコードによるPerl入...
My Bookmark – 2010/1/25(月)の出来事
fumixtokyo Twitter Log – 2010-01-25のツイート
Perl入門〜サンプルコードによるPerl入門〜 – Ruby基礎文法最速マス...
本日のニュース – 2010年1月26日
D++のはまり日誌 – Python 基礎文法最速マスター
燈明日記 – 基礎文法最速マスター
tosh-tの日記
何かしらの言語による記述を解析する日記 – VBA基礎文法最速マスター
おれさま新聞 – ●各言語の最速マスター特集
何かしらの言語による記述を解析する日記 – Java基礎文法最速マスタ...
医者を志す妻を応援する夫の日記 – Brainf*ck基礎文法最速マスター
think and error – Haskell基礎文法最速マスター
CX's VBScript Diary – その他の基礎文法マスター
CX's VBScript Diary – VBScript 基礎文法最速マスター
何かしらの言語による記述を解析する日記 – Bash基礎文法最速マスタ...
Perl入門〜サンプルコードによるPerl入門〜 – Perl基礎文法最速マス...
web探検隊 – JavaScript基礎文法最速マスター
なんとなく日記 – JavaScript基礎文法最速マスター
CX's UWSC Diary – UWSC 基礎文法最速マスター
燈明日記 – 基礎文法最速マスターぞくぞくキターーー!
surume000の日記 – プログラミング言語基礎文法最速マスターまとめ
はてなかよっ! – D言語基礎文法最速マスター
きまぐれメモ – 各種言語による基礎文法最速マスターまとめ
shikaku's memo blog – ○○基礎文法最速マスター
永遠に未完成 – Vimスクリプト基礎文法最速マスター
きまぐれメモ – 各種言語による基礎文法最速マスターまとめ
[Flash] ActionScript 3.0 基礎文法最速マスター
(rubikitch loves (Emacs Ruby CUI)) – Emacs Lisp基礎文法最速マス...
どうでもいい情報置き場 – Whitespace基礎文法最速マスター
[Diksam]Diksam基礎文法最速マスター
use GFx::WebLog; – XS基礎文法最速マスター
Life like a clown – はてな的プログラミング言語人気ランキング
[Perl] Perl基礎文法最速マスター
Lua基礎文法最速マスター
なんとなく日記 – 基礎文法最速マスターシリーズのまとめ
Ruby基礎文法最速マスター

idトラックバック

et3 – メチャクチャ詳しいPerl学習サイト

リンク元

3669 http://b.hatena.ne.jp/
1616 http://b.hatena.ne.jp/hotentry
837 http://reader.livedoor.com/reader/
698 http://pipes.yahoo.com/pipes/pipe.info?_id=faa858a20082ef6d25ad27557e37e011
619 http://b.hatena.ne.jp/hotentry/it
461 http://www.sleipnirstart.com/
401 http://www.1x1.jp/blog/2010/01/php-basic-syntax.html
394 http://ig.gmodules.com/gadgets/ifr?view=home&url=http://choichoi.sakura.ne.jp/hatena_bookmark.xml&nocache=0&up_num_feed=10&lang=ja&country=jp&.lang=ja&.country=jp&synd=ig&mid=42&ifpctok=

-3426623299131673352&exp_split_js=1&exp_track_js=1&exp_new_js_flags=1

377 http://d.hatena.ne.jp/

360 http://route477.net/d/?date=20100125

〈今年勉強したいこと | Ruby基礎文法最速マスターってい...〉