PB173 Perl

06 Súborový systém

Roman Lacko <xlacko1@fi.muni.cz>

Obsah

Vstupno-výstupné operácie (dokončenie)

```
getc [FILEHANDLE]
```

Prečíta znak z `FILEHANDLE` alebo štandardného vstupu.
Na konci vráti `undef`.

```
eof FILEHANDLE
eof ()
eof
```

Zistí, či **nasledujúca** I/O operácia skončí s *End of File*.

- Bez argumentu testuje posledný použitý súbor.
- Argument `( )` (prázdny zoznam) testuje koniec `<>` a `<<>>`.

Vstupno-výstupné operácie

```
read FILEHANDLE, SCALAR, LENGTH, [OFFSET]
```

Prečíta do SCALAR zadaný počet **znakov** zo súboru.
OFFSET mení pozíciu v SCALAR, kam sa dáta zapíšu.

Vráti počet prečítaných znakov, 0 na konci súboru, `undef` pri chybe.

Čo sú **znaky** záleží od aktuálneho nastavenia kódovania.

Perl má `write`, ale robí bohužiaľ niečo úplne nesúvisiace.

Kódovanie vstupu a výstupu

Vo východzom stave Perl ukladá reťazce ako *byte strings* (ISO-8859-1).

Súbor môže mať na sebe naviazané disciplíny (alebo *layers*),
ktoré menia kódovanie reťazcov pri I/O operáciách:

```
use open IN => ':encoding(UTF-8)', OUT => ':raw';
use open ':std', ':encoding(ISO-8859-2)';
```

Lexikálna pragma, ktorá mení východzie disciplíny pre `open`.
`:std` mení disciplíny aj pre vstup a výstup.

Podobný efekt má `-C` prepínač pre `perl` (`man perlrun`), preferujte
však explicitné nastavenie kódovania v skripte.

Nepoužívajte disciplínu `:utf8`!
Používa ju interne Perl, nie je určená na vstup a výstup.

Kódovanie vstupu a výstupu

Ďalšie možnosti nastavenia disciplíny:

```
open my $fh, '<:encoding(UTF-8)', $filename;
```

Aplikuje disciplínu na otváraný súbor.

```
binmode FILEHANDLE, DISCIPLINE;
```

Aplikuje disciplínu na otvorený súbor.

Kódovanie vstupu a výstupu

Často používané disciplíny:

`:raw`
: Binárne dáta.

`:encoding(ENCODING)`
: Nastaví kódovanie vstupu alebo výstupu na `ENCODING`.
Typicky ako `:encoding(UTF-8)`.

`:crlf`
: Obvykle na Windows, konvertuje výstupné `\n` na `\r\n` a naopak pri vstupe.

Disciplíny je možné na seba skladať.

Podrobnosti viď https://metacpan.org/pod/PerlIO.

Kódovanie vstupu a výstupu

```
use utf8;
```

Pragma, ktorá deklaruje, že reťazce priamo v kóde sú kódované v UTF-8.

```
use Encode;

my $chars = Encode::decode('UTF-8', $bytes);
my $bytes = Encode::encode('UTF-8', $chars);
```

https://metacpan.org/pod/Encode

Explicitne dekóduje alebo zakóduje reťazec.

Niektoré moduly očakávajú parametre v *byte strings*, iné reťazce znakov. Čítajte manuál.

Kódovanie vstupu a výstupu (zhrnutie)

 Ako sa v tom má človek vyznať?

Kódovanie robí problém v takmer každom jazyku.

Praktické zásady:

- Používajte `utf8` pragmu.
- Argumenty dekódujte čo najskôr, ak ich potrebujete.
- Vstup dekódujte čo najskôr, výstup čo najneskôr (`open`, `binmode`).

A Million Billion Squiggly Characters
https://www.youtube.com/watch?v=TmTeXcEixEg

Prednáška od R. Signes (dokument, komédia, 2016)

Operácie na súboroch

```
seek FILEHANDLE, POSITION, WHENCE
tell [FILEHANDLE]
```

Nastaví alebo vráti pozíciu v súbore.
Funkcie pracujú v **bajtoch**, nie znakoch.

Hodnoty pre `WHENCE` sú konštanty ako pre `seek(3)`, symbolické hodnoty sa dajú importovať z `Fcntl` modulu.

```
use Fcntl ':seek';

seek $fh, 0, SEEK_SET;
```

Binárne reťazce

```
pack TEMPLATE, LIST
```

Konvertuje zoznam hodnôt na reťazec bajtov podľa zadanej šablóny.

```
pack "a*xl", 'abc', 1024;

# 0x61 0x62 0x63 0x00 0x00 0x04 0x00 0x00
# ----- a* -----   x  ------- l ---------
```

```
unpack TEMPLATE, [EXPRESSION]
```

Z binárneho reťazca extrahuje hodnoty podľa šablóny.

Súborový systém

```
stat HANDLE
stat
lstat
```

Vráti pole atribútov zo štruktúry `struct stat` systémového volania `stat(3)`.

Modul `File::stat` pre pohodlnejšiu prácu

```
use File::stat;

my $stat = stat $fh;
say $stat->mode;
```

`-X` operátory

Typ súboru sa obvykle určuje z `mode` zo `stat`.
Perl má navyše pohodlnejšie operátory tvaru `-X`:

```
-X HANDLE
-X EXPR
-X
```

Predikát, ktorý otestuje nejakú vlastnosť súboru.
Môže byť zadaný menom, alebo ako otvorený súbor.

Bez argumentu testuje `$_`.

```
                # use Fcntl ':stat';
-f FILE         # Same as (stat FILE)[2] & S_IFREG
-x FILE         # Same as (stat FILE)[2] & S_IXUSR
-s FILE         # Same as (stat FILE)[7]
```

-X operátory

Každé volanie operátora nad reťazcom stojí systémové volanie.
Špeciálny otvorený súbor _ zopakuje test na výsledku `stat` alebo iného testu:

```
stat FILE;

say "regular" if -f _;
say "directory" if -d _;
```

Perl má navyše syntaktický cukor pre podmienky, ktoré majú platiť súčasne:

```
-f -x FILE              # Same as -f FILE && -x FILE
```

Súborové operácie

```
truncate WHAT, LENGTH
```

Zmení veľkosť otvoreného alebo pomenovaného súboru na `LENGTH`.

```
chmod MODE, LIST
chown UID, GID, LIST
```

Zmenia práva, vlastníka a skupinu zadaných súborov.
`MODE` zadávajte ideálne ako číslo, alebo použite `oct`.

```
chmod 0644, $fh;         # Okay.
chmod oct "0644", $fh;   # Also okay.
chmod "0644", $fh;       # Whoopsie, sets mode to ‹--w----r-T›!
```

Operácie s adresármi

```
glob EXPR
<*.c>
```

V zoznamovom kontexte vráti mená súborov v aktuálnom pracovnom adresári, ktoré vyhovujú zadanému vzoru (podobne ako shell).
V skalárnom kontexte vracia záznamy postupne, po poslednom `undef`.

```
glob "*.pl"             # Files ending with '.pl'
glob "x??"              # Files starting with 'x' and two more characters
glob "x[ab]"            # Either 'xa' or 'xb'
glob "ex{01,02}*"       # Files starting with 'ex01' or 'ex02'
glob "a* b*"            # Files starting with 'a' or 'b'
```

Ak vzor obsahuje **len** neprázdne `{...}` zátvorky, vráti expandované reťazce bez ohľadu na existenciu súborov (rovnako ako shell).

Operácie s adresármi

Pozor na nejednoznačnosť <X> operátora!

Podľa `X` totiž môže znamenať `readline` alebo `glob`:

`readline`:

- `X` je *bareword* alebo *typeglob* označujúci súbor (napr. `STDIN`).
- `X` je skalárna premenná (napr. `$fh`).

`glob`:

- `X` neoznačuje otvorený súbor
- `X` je výraz iný, než skalárna premenná (napr. `$hash{value}`).

Ak nechcete riskovať, pokojne píšte `readline` a `glob` explicitne.

Operácie s adresármi

```
opendir DIRHANDLE, EXPR
closedir DIRHANDLE
```

Obdoba `open` a `close` pre adresáre.

Ako pri súboroch, aj pri adresároch sa o zatvorenie postará Perl automaticky, keď zanikne posledná referencia na `DIRHANDLE`.

```
opendir my $dirh, "/etc"
    or die "/etc: $!\n";
```

Operácie s adresármi

```
readdir DIRHANDLE
```

V zoznamovom kontexte vráti (zostávajúce) položky v adresári.
V skalárnom vracia postupne položky adresára, po poslednej vráti `undef`.

```
while (readdir $dirh) {
    say "file $_" if -f;
    say "dir  $_" if -d;
}
```

Položky obsahujú len názov objektu, nie celú cestu k nemu.

https://metacpan.org/pod/File::Spec, metódy `catfile()` a `catdir()`.

Operácie s adresármi

```
rewinddir DIRHANDLE
```

Vráti kurzor čítaného adresára na začiatok, takže ďalšie volanie `readdir` začne znova od prvého záznamu.

```
telldir DIRHANDLE
seekdir DIRHANDLE, POS
```

Vráti resp. nastaví pozíciu kurzora v čítanom adresári.

Špeciálne literály

V zdrojovom kóde Perlu sa môže nachádzať šesť špeciálnych literálov:

`__FILE__`
: Názov súboru

`__LINE__`
: Aktuálne číslo riadka

`__PACKAGE__`
: Názov modulu

`__SUB__`
: Referencia na aktuálnu funkciu.

Špeciálne literály

`__END__`

Označuje koniec skriptu, zvyšok súboru sa ignoruje.

`__DATA__`

Označuje koniec skriptu, zvyšok súboru je dostupný ako súbor `DATA`.
Takto je možné do skriptu pribaliť nejaké vstupy.

```
print while <DATA>;

__DATA__
1
2
3
```