

Perl в примере

или делаем монитор системы автосборок

Максим Вуец

Арисент Украина

2008

Затравка

P
E
R
L

Practical
Extraction and
Report
Language

Practical
Extraction and
Report
Language

Практический язык
извлечения данных и
создания отчетов

Pathologically

Eclectic

Rubbish

Lister

Pathologically
Eclectic
Rubbish
Lister

Паталогически
эkleктичный язык
для распечатки
чепухи

Ларри Уолл



Формула

Unix shell + AWK + sed + C + Lisp +
регулярные выражения +
лингвистическое образование Уолла +
синтаксический сахар +
чутьочка магии

Основные структуры данных

- ▶ \$скаляр

- ▶ @массив

- ▶ %хеш

Основные структуры данных

- ▶ \$скаляр

Может содержать целое или вещественное число, строку или символ, ссылку или ничего.

- ▶ @массив

- ▶ %хеш

Основные структуры данных

- ▶ \$скаляр

Может содержать целое или вещественное число, строку или символ, ссылку или ничего.

- ▶ @массив

Упорядоченный список скаляров, доступ к которым осуществляется по порядковому номеру — индексу.

- ▶ %хеш

Основные структуры данных

- ▶ \$скаляр

Может содержать целое или вещественное число, строку или символ, ссылку или ничего.

- ▶ @массив

Упорядоченный список скаляров, доступ к которым осуществляется по порядковому номеру — индексу.

- ▶ %хеш

Неупорядоченный набор скаляров. Доступ по строковому значению — ключу.

Синтаксис

Смесь C и Unix shell-a:

```
1 # Ordinary C-like for-lopp
2 for ($n = 1; $n <= 10; $n++) {
3     print $n, "\n";
4 }
5
6 $j = 5;
7 while ($n) { # The same while-loop
8     if ($j % 2) {
9         print "\$j=$j is even\n"; # Interpolation
10    } else {
11        print '$j=', "$j is odd\n";
12    }
13    --$n;
14 }
```

Контекст

- ▶ Скалярный

```
$foo = 'bar';
```

```
$date = localtime; # Thu Oct 30 21:07:03 2008
```

Контекст

- ▶ Скалярный

```
$foo = 'bar';
```

```
$date = localtime; # Thu Oct 30 21:07:03 2008
```

- ▶ Списковый

```
@nums = (5, 10, 15);
```

```
($sec, $min, $hour, $mday, $mon, $year,  
 $wday, $yday, $isdst) = localtime;
```

Контекст

- ▶ Скалярный

```
$foo = 'bar';
```

```
$date = localtime; # Thu Oct 30 21:07:03 2008
```

- ▶ Списковый

```
@nums = (5, 10, 15);
```

```
($sec, $min, $hour, $mday, $mon, $year,  
 $wday, $yday, $isdst) = localtime;
```

- ▶ Логический

```
'', '0', 0, undef — ложь. Все остальное — истина.
```

Контекст

```
print localtime;           # 3112130910843030
```

Контекст

```
print localtime;           # 3112130910843030  
print scalar localtime;   # Thu Oct 30 21:07:03 2008
```

Контекст

```
print localtime;           # 3112130910843030
print scalar localtime;   # Thu Oct 30 21:07:03 2008
@nums = (5, 10, 15);
```

КОНТЕКСТ

```
print localtime;           # 3112130910843030
print scalar localtime;   # Thu Oct 30 21:07:03 2008
@nums = (5, 10, 15);
$foo = @nums;              # $foo = 3
```

КОНТЕКСТ

```
print localtime;           # 3112130910843030
print scalar localtime;   # Thu Oct 30 21:07:03 2008
@nums = (5, 10, 15);
$foo = @nums;             # $foo = 3
($bar) = @nums;          # $bar = 5
```

КОНТЕКСТ

```
print localtime;           # 3112130910843030
print scalar localtime;   # Thu Oct 30 21:07:03 2008
@nums = (5, 10, 15);
$foo = @nums;              # $foo = 3
($bar) = @nums;           # $bar = 5
$qux = '7xyz';
```

КОНТЕКСТ

```
print localtime;           # 3112130910843030
print scalar localtime;   # Thu Oct 30 21:07:03 2008
@nums = (5, 10, 15);
$foo = @nums;              # $foo = 3
($bar) = @nums;           # $bar = 5
$qux = '7xyz';
$garply = 0 + $qux;       # $garply = 7
```

Задача

Задача

Монитор состояния
автосборок

На входе...

Файл заданий

версия:язык:тип

2.5.0.4000:english:business

2.5.0.4000:english:personal

2.4.1.3100:ukrainian:business

На входе

Сборочное дерево

```
builds
|-- enx/
|   |-- business/
|   |   '-- 2.5.0.4000/
|   |       |-- dist/
|   |           '-- status.log
|   '-- personal/
|       '-- ...
'-- ukr/
    '-- business/
        '-- ...
```

На входе

status.log

```
i 2008-10-27 22:28:01 Configuration: ver=2.5.0 build=...
w 2008-10-27 22:28:04 Low disk space.
i 2008-10-27 22:28:04 Build started...
i 2008-10-27 22:33:56 Fetching sources...
i 2008-10-27 22:33:56 Sources fetched.
i 2008-10-27 22:58:12 Compiling...
i 2008-10-27 22:58:12 Compiled.
i 2008-10-27 22:59:42 Making dist...
i 2008-10-27 23:03:37 Dist is made.
i 2008-10-27 23:03:42 BUILD SUCCEEDED.
```

В результате

Version	Language	Type	Started	Ended	Warnings	State
2.5.0.4000	english	business	22:28:04	23:03:42	1	Succeeded
2.5.0.4000	english	personal	23:15:22	23:24:28	2	Failed
2.4.1.3100	ukrainian	business	21:47:59	22:58:05	0	Failed

Реализация

Unix shebang и комментарии

```
1 #!/usr/bin/perl
2
3 # Sample dumb application for the "Perl by Example"
   talk
4 # (c) 2008 Maxim Vuets <http://maxim.vuets.name/>
5
```

Строгий режим и предупреждения

```
8 use strict;  
9 use warnings;
```

Константы

```
12 # Some constants
13 my $BUILDS_FILE = 'builds.dat';
14 my $STATUS_FILE = 'status.log';
15 my $REPORT_FILE = 'report.html';
16 my @STATE = ('Not started', 'In progress', 'Succeeded'
17             , 'Failed');
18 my %LANG_MAP = (
19     english => 'enx',
20     russian => 'rus',
21     ukrainian => 'ukr'
22 );
```

Константы

```
22 use constant {
23     NOT_STARTED => 0,
24     IN_PROGRESS => 1,
25     SUCCEEDED => 2,
26     FAILED => 3
27 };
```

Проверка аргумента командной строки

```
41 # Get command line argument
42 my $builds_dir = $ARGV[0];
43 unless (defined $builds_dir) {
44     print STDERR "Usage: perl $0 <builds_dir>\n";
45     exit 1;
46 }
47
48 # Check directory existence
49 die "'$builds_dir' does not exist" unless -d
    $builds_dir;
```

Чтение файла заданий

```
51 # Read and process builds data file
52 my @builds;
53 open BUILDS, $BUILDS_FILE or die "Cannot open '
    $BUILDS_FILE': $!";
54 while (<BUILDS>) {
    ...
68 }
69 close BUILDS;
```

Обработка файла заданий

```
55     chomp;
56     my ($ver, $lang, $type) = split /:/:;
57     next unless exists $LANG_MAP{$lang};
58     my %b = (
59         Version => $ver,
60         Language => $lang,
61         Type => $type,
62         Warnings => 0,
63     );
64     $b{Path} = "$builds_dir/$LANG_MAP{$lang}/$type/$ver/
65               $STATUS_FILE";
66     $b{State} = NOT_STARTED;
67     $b{StartTime} = $b{EndTime} = '';
68     push @builds, \%b;
```

Анализ состояния сборок

```
71 # Analyze build logs
72 for (my $n = 0; $n < @builds; ++$n) {
73     my $b = $builds[$n];
74     if ($b->{State} == NOT_STARTED) {
75         $b->{State} = IN_PROGRESS if -e $b->{Path};
76     }
77     if ($b->{State} == IN_PROGRESS) {
78         open LOG, '<', $b->{Path} or die "Cannot open '$b
           ->{Path}': $!";
79         while (<LOG>) {
           ...
           }
93         close LOG;
94     }
95 }
96 }
```

Анализ журнала сборки

```
80     chomp;
81     if (m/^w/) {
82         $b->{Warnings}++;
83         next;
84     }
85     if (/^i.+?(\\d{2}:\\d{2}:\\d{2}) Build started
        \\.\\.\\.$/ ) {
86         $b->{StartTime} = $1;
87     }
88     elsif (/^i.+?(\\d{2}:\\d{2}:\\d{2}) BUILD (
        SUCCEEDED|FAILED)\\.$/ ) {
89         $b->{EndTime} = $1;
90         $b->{State} = $2 eq 'FAILED' ? FAILED :
            SUCCEEDED;
91         last;
92     }
```

Начало HTML отчета

```
98 # Create HTML report
99 open REPORT, ">$REPORT_FILE" or die "Cannot open '
    $REPORT_FILE': $!";
100 print REPORT <<'HTML';
101 <html>
102 <head>
103   <title>Builds Report</title>
    ...
117   <th>Ended</th>
118   <th>Warnings</th>
119   <th>State</th>
120 </thead>
121 HTML
```

Вывод состояния каждой сборки

```
122 foreach my $b (sort {$a->{Language} cmp $b->{Language  
    }} @builds) {  
    ...  
144 }
```

Выбор цвета на основе состояния

```
123 my $color;
124 my $state = $b->{State};
125 if ($state == IN_PROGRESS)
126     { $color = 'ccf' }
127 elsif ($state == FAILED)
128     { $color = 'f99' }
129 elsif ($state == SUCCEEDED)
130     { $color = '6f6' }
131 $color = defined $color ? qq[ style="background-
    color: #${color}"] : '';
```

Экранирование специальных HTML СИМВОЛОВ

```
30 # Prepare a string to be used in HTML
31 sub htmlize {
32     my $s = $_[0];
33     $$s =~ s/ & / & amp; /g;
34     $$s =~ s/ < / & lt; /g;
35     $$s =~ s/ > / & gt; /g;
36     $$s =~ s/ ' / & apos; /g;
37     $$s =~ s/ " / & quot; /g;
38 }

...

132 htmlize(\$b->{$_}) foreach (keys %$b);
```

Формирование и вывод HTML по сборке

```
133     print REPORT <<HTML;
134     <tr$color>
135         <td>$b->{Version}</td>
136         <td>$b->{Language}</td>
137         <td>$b->{Type}</td>
138         <td>$b->{StartTime}</td>
139         <td>$b->{EndTime}</td>
140         <td>$b->{Warnings}</td>
141         <td>$STATE[$b->{State}]</td>
142     </tr>
143     HTML
```

Завершение HTML отчета

```
145 print REPORT <<'HTML';  
146     </table>  
147 </body>  
148 </html>  
149 HTML  
150 close REPORT;
```

Демонстрация

Получение помощи

- ▶ `perldoc perlsyn`
Синтаксис
- ▶ `perldoc perlop`
Операторы и конструкции
- ▶ `perldoc perldata`
Типы данных
- ▶ `perldoc perlvar`
Специальные переменные
- ▶ `perldoc perlfunc`
Встроенные функции
- ▶ `perldoc -f <имя_функции>`
Описание конкретной функции

Спасибо!

© 2008 Максим Вуец <<http://maxim.vuets.name/>>

На условиях <http://creativecommons.org/licenses/by-sa/3.0/>

Вопросы?

© 2008 Максим Вуец <<http://maxim.vuets.name/>>

На условиях <http://creativecommons.org/licenses/by-sa/3.0/>