# Scripting: Perl

„Practical extraction and reporting language“

# Scripting: Perl
## ein Beispiel

```
#!/usr/bin/perl -w
use strict;

if(open(PU,'/proc/uptime'))
   {
   my $data=<PU>;
   close(PU);
   chomp($data); # Zeilenende weg

   my($uptime,$idle)=split(/ /,$data);
   print (($uptime-$idle)." s keine Langeweile\n");
   }

exit(0);
```

```
$ cat /proc/uptime
10944.83 9927.30
```

# Scripting: Perl
## ein Beispiel

```
#!/usr/bin/perl -w
use strict;

if(open(PU,'/proc/uptime'))
    {
    my $data=<PU>;
    close(PU);
    chomp($data); # Zeilenende weg

    my($uptime,$idle)=split(/ /,$data);
    print (($uptime-$idle)." s keine Langeweile\n");
    }

exit(0);
```

```
$ cat /proc/uptime
10944.83 9927.30
```

# Scripting: Perl
## ein Beispiel

```
#!/usr/bin/perl -w
use strict;

if(open(PU,'/proc/uptime'))
    {
    my $data=<PU>;
    close(PU);
    chomp($data); # Zeilenende weg

    my($uptime,$idle)=split(/ /,$data);
    print (($uptime-$idle)." s keine Langeweile\n");
    }

exit(0);
```

```
$ cat /proc/uptime
10944.83 9927.30
```

# Scripting: Perl
## ein Beispiel

```
#!/usr/bin/perl -w
use strict;

if(open(PU,'/proc/uptime'))
   {
   my $data=<PU>;
   close(PU);
   chomp($data); # Zeilenende weg

   my($uptime,$idle)=split(/ /,$data);
   print (($uptime-$idle)." s keine Langeweile\n");
   }

exit(0);
```

```
$ cat /proc/uptime
10944.83 9927.30
```

# Scripting: Perl
## ein Beispiel

```
#!/usr/bin/perl -w
use strict;

if(open(PU,'/proc/uptime'))
   {
   my $data=<PU>;
   close(PU);
   chomp($data); # Zeilenende weg

   my @werte=split(/ /,$data);
   print (($werte[0]-$werte[1])." s keine Langeweile\n");
   }

exit(0);
```

```
$ cat /proc/uptime
10944.83 9927.30
```

# Scripting: Perl
## ein weiteres Beispiel

```
#!/usr/bin/perl -w
use strict;

opendir(DIR,"/tmp") || die($!);
while(my $e=readdir(DIR))
   {
   if( ($e=~m/^sess_/) && ((-A "/tmp/$e")>2) )
      {
      if(!unlink("/tmp/$e"))
         {
         warn "Loeschen fehlgeschlagen: $!\n";
         }
      }
   }

closedir(DIR);
exit(0);
```

# Scripting: Perl
## ein weiteres Beispiel

```
#!/usr/bin/perl -w
use strict;

opendir(DIR,"/tmp") || die($!);
while(my $e=readdir(DIR))
    {
    if( ($e=~m/^sess_/) && ((-A "/tmp/$e")>2) )
        {
        if(!unlink("/tmp/$e"))
            {
            warn "Loeschen fehlgeschlagen: $!\n";
            }
        }
    }

closedir(DIR);
exit(0);
```

# Scripting: Perl
## ein weiteres Beispiel

```
#!/usr/bin/perl -w
use strict;

opendir(DIR,"/tmp") || die($!);
while(my $e=readdir(DIR))
   {
   if( ($e=~m/^sess_/) && ((-A "/tmp/$e")>2) )
      {
      if(!unlink("/tmp/$e"))
         {
         warn "Loeschen fehlgeschlagen: $!\n";
         }
      }
   }

closedir(DIR);
exit(0);
```

# Scripting: Perl
## ein weiteres Beispiel

```
#!/usr/bin/perl -w
use strict;

opendir(DIR,"/tmp") || die($!);
while(my $e=readdir(DIR))
   {
   if( ($e=~m/^sess_/) && ((-A "/tmp/$e")>2) )
      {
      if(!unlink("/tmp/$e"))
         {
         warn "Loeschen fehlgeschlagen: $!\n";
         }
      }
   }

closedir(DIR);
exit(0);
```

# Scripting: Perl
## noch ein Beispiel

```
#!/usr/bin/perl -w
use strict;

while(1)
    {
    open(FH,'/proc/loadavg') || die($!);
    my $result=<FH>;
    close(FH);

    my @values=split(/ /,$result);

    if($values[0]>2)
        {
        open(FH,">>logfile") || die($!);
        print FH localtime().": $values[0]\n";
        close(FH) || die($!);
        }
    sleep(1);
    }

exit(0);
```

```
$ cat /proc/loadavg
0.10 0.25 0.18 1/98 6269
```

# Scripting: Perl
## noch ein Beispiel

```
#!/usr/bin/perl -w
use strict;

while(1)
    {
    open(FH,'/proc/loadavg') || die($!);
    my $result=<FH>;
    close(FH);

    my @values=split(/ /,$result);

    if($values[0]>2)
        {
        open(FH,">>logfile") || die($!);
        print FH localtime().": $values[0]\n";
        close(FH) || die($!);
        }
    sleep(1);
    }

exit(0);
```

```
$ cat /proc/loadavg
0.10 0.25 0.18 1/98 6269
```

# Scripting: Perl
## noch ein Beispiel

```
#!/usr/bin/perl -w
use strict;

while(1)
    {
    open(FH,'/proc/loadavg') || die($!);
    my $result=<FH>;
    close(FH);

    my @values=split(/ /,$result);

    if($values[0]>2)
        {
        open(FH,">>logfile") || die($!);
        print FH localtime().": $values[0]\n";
        close(FH) || die($!);
        }
    sleep(1);
    }

exit(0);
```

```
$ cat /proc/loadavg
0.10 0.25 0.18 1/98 6269
```

# Scripting: Perl
## noch ein Beispiel

```
#!/usr/bin/perl -w
use strict;

while(1)
    {
    open(FH,'/proc/loadavg') || die($!);
    my $result=<FH>;
    close(FH);

    my @values=split(/ /,$result);

    if($values[0]>2)
        {
        open(FH,">>logfile") || die($!);
        print FH localtime().": $values[0]\n";
        close(FH) || die($!);
        }
    sleep(1);
    }

exit(0);
```

```
$ cat /proc/loadavg
0.10 0.25 0.18 1/98 6269
```

# Scripting: Perl
## freie Syntax

Bedingungen voran- oder hintergestellt:

```
if($< == 0)
   {
   print "Sie sind root!";
   }

print "Sie sind root!" if($< == 0);
```

Funktionen mit oder ohne Klammern aufrufen:

```
print "Das ist eine Ausgabe";
print("... und das erst recht!");
```

# Scripting: Perl
## freie Syntax

### temporäre Variable $_

```
foreach(1..10)
    {
    print;
    }
```

### Funktionen ohne benannte Parameter

```
sub addiere_alle
{
my $tmp=0;
foreach(@_)
    {
    $tmp+=$_;
    }
return $tmp;
}
```

# Scripting: Perl
## freie Syntax

### kontextabhängige Rückgabewerte

```
my @files=readir(DIR); # alle Dateien
my $file=readdir(DIR); # eine Datei
```

### kontextabhängige Funktionen

```
#!/usr/bin/perl -w
#

my $arg1=shift; # shift(@ARGV)
show($arg1);
exit(0);

sub show
{
my $wert=shift; # shift(@_)
print $wert;
}
```

shift ARRAY
liefert das erste Element der übergebenen Liste zurück

# Scripting: Perl
## Ähnlichkeiten zur Shell

Redirektoren/Pipes bei open(), system() u.a. wie in der Shell:

```
open(MAILER,'| /usr/lib/sendmail -t') || die($!);
print MAILER "To: rob\n";
print MAILER "Subject: Halllllooooo\n\n";
print MAILER "Das ist der Text der Mail";
close(MAILER);
```

```
if(system('ping -c 1 127.0.0.1 >/dev/null')==0)
    {
    print 'OK';
    }
```

# Scripting: Perl
## Ähnlichkeiten zur Shell

### grep

```
@zahlen=(3,10,21,4,17,1,29,6,13,42);
@gerade_zahlen=grep {($_%2)==0} @zahlen;
```

### sort

```
@sortierte_zahlen=sort {$a <=> $b} @zahlen;
```

### Dateitests

```
if(-e "/tmp") {print "/tmp existiert"}
if(-d "/tmp") {print "/tmp ist ein Verzeichnis"}

print "Kernel: ".(-s "/boot/vmlinux")." bytes";
```

# Scripting: Perl
## Und was sonst noch?

- Hashes (assoziative Listen)
- Referenzen auf Variablen und Funktionen
- Generierung und Ausführung von Perlcode zur Laufzeit
- reguläre Ausdrücke (nächster Vortrag)
- Objektorientierung

# Scripting: Perl
## weitere Anwendungen

- Webanwendungen
- Systemmanagement
- Server/Client
- Vieles, vieles mehr durch tausende Module

# Scripting: Perl
## Hilfe

$ man perltoc
$ man perlfaq[1-9]

$ perldoc -f eingebaute_Funktion
$ man perlfunc

$ perldoc Weltfrieden

google://perl "<dein problem hier>"

http://www.cpan.org/
http://www.perl.com/

# Scripting: Perl
## Fragen?

```
foreach(@question)
    {
    print (answer($_) || "keine Ahnung");
    }

print "DANKE!";

exit(0);
```