Aleksander Adamowski's useful scripts

NEW Location: https://olo.org.pl/notes/linux_stuff/useful_scripts

Also available under http://tinyurl.com/ks4bws

base64_dec.py
base64_enc.py
base64_dec
base64_enc
quoted_printable_dec
quoted_printable_enc
all_pair_combinations.py
dusage_in_curdir.sh
db_dump.pl - dump a Berkeley DB database
bekap
bekap_in_place
httrack-mirror_website.sh
check_nip.py
generate_nip.py
check_nrb.py
check_pesel.py
generate_pesel.py
check_regon.py
luhnCalculator.py
luhnCheck.py
chrome_bookmarks-extract_URLs.py
url_encode.pl
url_decode.pl
html_escape_with_entities.pl
ldap_ssha_make.py

ldap_ssha_check.py
soap_digest_check.sh
rename_to_random.sh
rename_files_for_windows_fast.pl
rename_files_perl_expression.pl
remove_newline_at_eof.pl
reread_partition_table_blockdev.sh
scsi_rescan_bus.sh
epoch2time.pl
epoch2time.sh
primes.sh
mplayer_play_DVD_iso_image_with_subs.sh
keytool_add_ca_cert.sh
list PKCS#11 KeyStore
list PKCS#12 KeyStore
Import PKCS#12 to JKS KeyStore
python_compile.sh
terroristize.lisp
jpegtran_from_jpegexif.sh
input_rate_notifier.pl
monitor_logfile.pl
exec_timeout_kill.pl
ettercap_arp_sniff.sh
amr_to_wav.sh
convert_flac_to_mp3_subtree.sh

- ffmpeg_transcode_avi_to_Android.sh
- ldapsearch_get_suffixes.sh
- wikiconv.pl
- wiki_indent
- encode_BCD.pl
- jboss_list_JNDI
- jboss_scp_deploy_and_wait.sh
- sizeof_void.c
- phl_dump_token.sh
- phl_object_deleter.sh
- explode_zips.sh
- rename_jpg_based_on_exif_timestamp.sh
- do_postmark_tests.sh
- lookup_IP_country.sh
- timed-run-err
- ctags_php
- Minimal.desktop
- node-reformat_JSON.js
- XML Starlet
- screentitle.sh
- randbetween.sh
- ps_backtrace.sh
- ps_descendants.sh

Microsoft Windows

- PowerShell

POSIX systems

Scripts in bourne shell/bash, Python, Perl etc.

Obviously, some may work on non-POSIX systems but it's not guaranteed.

Bash aliases

```
alias mkdir_curdate='mkdir $(date +%F)'
alias mkdir_cd_curdate='mkdir $(date +%F); cd $_'
alias cd_curdate='cd $(date +%F)'
alias mkdir_curdate_time='mkdir -p $(date +%F)/$(date +%H_%M_%S)'
alias mkdir_cd_curdate_time='mkdir -p $(date +%F)/$(date +%H_%M_%S); cd $_'
alias cd_curdate_time='cd $(date +%F)'
alias mkdir_curtimestamp='mkdir $(date +%F_%H_%M_%S)'
alias mkdir_cd_curtimestamp='mkdir $(date +%F_%H_%M_%S); cd $_'
```

Epoch timestamp to current date

```
date -d "@$1"
```

Exposing shells on the network

```
socat tcp-l:3333 exec:/bin/sh  <-- on the remote side
socat tcp:localhost:3333 stdout
ncat bmcaddr 3333
```

With full tty features:
https://blog.ropnop.com/upgrading-simple-shells-to-fully-interactive-ttys/#method2usingsocat

```
socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp6-l:1337

socat file:tty,raw,echo=0 tcp6-connect:<remoteaddr>:1337
```

OpenSSL

ssl_check_expiry_date.sh

```
#!/bin/sh

echo "" | openssl s_client -connect "$@" 2>/dev/null | \

 sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' | \

 openssl x509 -noout -enddate | perl -e '

use POSIX qw(strftime);

use Date::Parse;

$_ = <>;

/^notAfter=(.*)/;

print strftime("%F", Date::Parse::strptime($1));

print "\n";

';
```

tls_smtp_check_expiry_date.sh

```
#!/bin/sh

echo "" | openssl s_client -starttls smtp -connect "$1:25" 2>/dev/null | \

 sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' | \

 openssl x509 -noout -enddate | perl -e '

use POSIX qw(strftime);
```

```
use Date::Parse;

$_ = <>;

/^notAfter=(.*)/;

print strftime("%F", Date::Parse::strptime($1));

print "\n";

';
```

ssl_warn_expiry_date.sh

```
#!/bin/sh
host="$1"
echo '' | openssl s_client -connect "$1" 2>/dev/null | \
 sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' | \
 openssl x509 -noout -enddate | perl -e "
use POSIX qw(strftime);
use Date::Parse;
\$_ = <>;
/^notAfter=(.*)/;
\$expirytime=Date::Parse::str2time(\$1);
\$timediff = \$expirytime - time;
if (\$timediff < 3600*24*31) {
        print 'Only '.(\$timediff /3600/24).' days left until SSL certificate expiry on $host';
        print \"\n\";
}
";
```

Use s_client with SNI

```
openssl s_client -connect host:port -servername fqdn
```

verify_aster_eFaktura.sh

```
#!/bin/sh
openssl smime -verify -inform DER -noverify -out /dev/null -in "$1.sig" -content "$1"
```

Convert x.509 certificate to CSR

```
openssl x509 -x509toreq -in CERT.pem -signkey PRIVKEY.pem
```

reflow_PEM.pl

```
#!/usr/bin/perl -p
s/(-----BEGIN[^-]+-----)/$1\n/g;
s/(-----END[^-]+-----)/\n$1/g;
s/([^\n]{64})/$1\n/g'
```

^ That's for the cases wher PEM-encoded data got all its linebreaks removed and e.g. openssl tools refuse to parse it with

ssl-redirect.pl

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

##########################################
```

```
# Przykladowe uzycie po wrzuceniu do cgi-bin:
# SSLRequireSSL
# ErrorDocument 403 /cgi-bin/ssl-redirect.pl
##############################################

$redirect = $ENV{'REDIRECT_URL'};
$redirect =~ s/^http:/https:/;
if ($redirect !~ /^https:/) {
  $redirect = "https://$ENV{'SERVER_NAME'}" . $redirect;
}
$redirect =~ s/private/private-ssl/;
print "Content-Type: text/html; charset=UTF-8\n";
print "Cache-Control: no-cache\n";
print "Pragma: no-cache\n";
print "Location: $redirect\n";
print "Refresh: 1; url=$redirect\n";
print "\n\n";
print <<EOD;
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
```

```
<html><head>

    <title>403 Dostęp zabroniony</title>

    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">

    <META http-equiv="Cache-Control" content="no-cache">

    <META http-equiv="Pragma" content="no-cache">

    </head><body>

    <h1>Dostęp zabroniony</h1>

    <P>Ta strona wymaga połączenia szyfrowanego SSL.

EOD

      print 'Oto poprawny adres: <A HREF="'.$redirect.'">'.$redirect.'</A>. Powinno pod niego za chwilę nastąpić
przekierowanie.<hr></body></html>';
```

ssl_client.sh

```
openssl s_client -CApath /etc/ssl/cacerts/ -connect $@
```

tls_smtp_client.sh

```
if [ $# -eq 1 ]; then

  openssl s_client -starttls smtp -connect $@
```

```
else
  echo "Uzycie: $0 host:port"
  echo "n.p.:"
  echo "$0 nmail.altkom.pl:25"
fi
```

openssl_cat_PKCS12

```
#!/bin/sh
openssl pkcs12 -info -nodes -in "$1"
```

hexdump2asn1parse.sh

Read and parse a hexdump of DER-encoded ASN.1 structure:

```
#!/bin/sh
xxd -r -p < $1 | openssl asn1parse -inform DER
```

GIT

Git relative diff alias

```
 git config --global alias.d 'diff --no-prefix --relative'
```

List branches and remote upstreams they track

```
git for-each-ref refs/heads --format='%(refname:short) %(upstream:short)'
```

Sum blob sizes introduced by commit

```
#!/bin/sh
commitid=$1
(
        for blobid in $(
                git diff-tree -r -c -M -C --no-commit-id $commitid | awk '{print $4}'
        ); do
        echo $blobid | git cat-file --batch-check;
        done;
) | awk -v commitid=$commitid '
{
        s += $3
}
 END {
        print "Summary size of blobs introduced by commit " commitid ": " s " bytes"
}'
```

Sort branches by last commit timestamp

```
 git for-each-ref --sort=-committerdate refs/
```

Mercurial

hg_log_raw_messages.sh

```
hg log --template '{desc}\n--------\n' "$@"
```

hgd

```
alias hgd='hg diff --noprefix --root . --color always | less -r'
```

base64_dec.py

```
#!/usr/bin/python


import sys

for line in sys.stdin.readlines():

        print line.decode('base64')
```

base64_enc.py

```
#!/usr/bin/python


import sys
```

```
for line in sys.stdin.readlines():
        print line.encode('base64')
```

base64_dec

```
#!/usr/bin/perl
use MIME::Base64 ();
while (<>) { $buf.=$_};
print MIME::Base64::decode($buf);
```

base64_enc

```
#!/usr/bin/perl
use MIME::Base64 ();
while (read(STDIN, $buf, 60*57)) {
  print MIME::Base64::encode($buf);
};
```

quoted_printable_dec

```
#!/usr/bin/perl

use MIME::QuotedPrint;

while (<>) { $buf.=$_};

print decode_qp($buf);
```

quoted_printable_enc

```
#!/usr/bin/perl

use MIME::QuotedPrint;
while (read(STDIN, $buf, 60*57)) {

  print encode_qp($buf);

};
```

all_pair_combinations.py

```
#!/usr/bin/python

import sys

import itertools

words=sys.stdin.readlines()

combinations=itertools.combinations(words, 2)
```

```
for combination in combinations:

        print combination[0].rstrip() + ' ' + combination[1].rstrip()

        print combination[1].rstrip() + ' ' + combination[0].rstrip()
```

dusage_in_curdir.sh

```
#!/bin/sh
du -xk --max-depth=1 . | sort -n
```

db_dump.pl - dump a Berkeley DB database

```
use warnings;
use strict;
use DB_File;
use Fcntl;

my (%db, $key, $val);
tie %db, 'DB_File', $ARGV[0], O_RDONLY, 0640, $DB_BTREE or die "Cannot open ".$ARGV[0].": $!\n";
# Dump the complete database
while (($key, $val) = each %db) {
          print "$key:$val\n";
}
untie %db;
```

bekap

```
#!/bin/sh
# by OLO
# skrypt tworzy kopie zapasowa pliku w wydzielonym katalogu ~/bekap.
# kopia tworrzona jest z dopiskiem - numerem wersji pliku,
# tworzonym z daty i czasu wykonania (z dokladnoscia do 1 sekundy)

if [ $# -ge 1 ]; then
          while [ $# -ne 0 ]; do
                   if [ -e "$1" ]; then
        curdate=$(date +%Y_%m_%d_%H=%M=%S)
        basename=$(basename "$1")
                           cp -ai "$1" ~/"bekap"/"$basename.${curdate}"
        bzip2 ~/"bekap"/"$basename.${curdate}"
                   else
                           echo "plik $1 nie istnieje"
                   fi
                   shift
          done
else
          echo "Uzycie: $0 plik_do_zachowania"
fi
```

bekap_in_place

```
#!/bin/sh
```

```
# by OLO
# skrypt tworzy kopie zapasowa pliku w katalogu biezacym.
# kopia tworrzona jest z dopiskiem - numerem wersji pliku,
# tworzonym z daty i czasu wykonania (z dokladnoscia do 1 sekundy)

if [ $# -ge 1 ]; then
          while [ $# -ne 0 ]; do
                  if [ -e "$1" ]; then
        curdate=$(date +%Y_%m_%d_%H=%M=%S)
        basename=$(basename "$1")
                          cp -ai "$1" "$1.${curdate}"
        bzip2 "$1.${curdate}"
                  else
                          echo "plik $1 nie istnieje"
                  fi
                  shift
          done
else
          echo "Uzycie: $0 plik_do_zachowania"
fi
```

httrack-mirror_website.sh

```
#!/bin/sh

httrack "$@" -j -%P -w -n -b1 -%h -%k -%B -C1 -P stacja.amarczuk:8080
```

check_nip.py

```
#!/usr/bin/python

import sys


def check_nip(nip):
        sum, ct = 0, [6, 5, 7, 2, 3, 4, 5, 6, 7]
        for i in range(9):
                sum += (int(nip[i]) * ct[i])
        return ((sum%11) == int(nip[9]))


line = sys.stdin.readline()
if check_nip(line):
        print "NIP poprawny"
else:
        print "NIP NIEPOPRAWNY!"
```

generate_nip.py

```
#!/usr/bin/python
import random

nip = []
sum, ct = 0, [6, 5, 7, 2, 3, 4, 5, 6, 7]
```

```
 random.seed()
 for i in range(9):
   digit = random.randint(0, 9)
   nip.append(`digit`)
   sum += (digit * ct[i])
 checkDigit = sum % 11
 nip.append(`checkDigit`)
 print ''.join(nip)
```

check_nrb.py

```
#!/usr/bin/python

import sys


def iban_letter2num(letter):
        return str(ord(letter) - ord('A') + 10)


num = sys.stdin.readline().strip().replace(' ', '')

if len(num) == 26:
        num = 'PL'+num

if len(num) != 28:
        print "Nieprawidlowa dlugosc numeru - powinno byc 28 znakow razem z kodem kraju, jest %d" % len(num)
```

```
else:
        print num
        num = num[4:] + iban_letter2num(num[0]) + iban_letter2num(num[1]) + num[2:4]
        n = int(num)
        if n % 97 == 1:
                print "ok"
        else:
                print "zla suma kontrolna"
```

check_pesel.py

```
#!/usr/bin/python
import sys


def check_pesel(pesel):
        sum, ct = 0, [1, 3, 7, 9, 1, 3, 7, 9, 1, 3, 1]
        for i in range(11):
                sum += (int(pesel[i]) * ct[i])
```

```
        return (str(sum)[-1] == '0')


line = sys.stdin.readline()

if check_pesel(line):

        print "PESEL poprawny"

else:

        print "PESEL NIEPOPRAWNY!"
```

generate_pesel.py

```
#!/usr/bin/python
import random

def gen_pesel():
        random.seed()
        year = "%02d" % random.randint(10, 90)
        month = "%02d" % random.randint(1, 12)
        day = "%02d" % random.randint(1, 28)
        sequencenumber = "%03d" % random.randint(1, 999)
        sexdigit = "%1d" % random.randint(0, 9)
        mainpart = year + month + day + sequencenumber + sexdigit
        sum, ct = 0, [1, 3, 7, 9, 1, 3, 7, 9, 1, 3, 1]
        for i in range(10):
                sum += (int(mainpart[i]) * ct[i])
        remainder = sum % 10
        controlcode = (10 - remainder) % 10
        return mainpart + ("%1d" % controlcode) + " , sexdigit: " + sexdigit + \
```

```
                (" (kobieta)" if int(sexdigit) % 2 == 0 else " (mezczyzna)")

print gen_pesel()
```

check_regon.py

```
#!/usr/bin/python
import sys

def check_regon(regon):
        sum = 0
        w = [8, 9, 2, 3, 4, 5, 6, 7]
        if len(regon) == 7:
                w = w[2:]
        elif len(regon) == 14:
                # ciezko znalezc algorytm (wagi) do sprawdzenia ostatnich
                # pieciu cyfr, wiec olewamy
                regon = regon[:9]

        ct = int(regon[-1])

        for i in range(len(w)):
                sum += (int(regon[i]) * w[i])
        mod = sum%11
        if (mod == ct) or (mod == 10 and ct == 0):
                return True
        return False

line = sys.stdin.readline().strip()
if check_regon(line):
  print "REGON poprawny"
else:
```

```
  print "REGON NIEPOPRAWNY!"
```

luhnCalculator.py

```
#!/usr/bin/python
# From http://www.xinotes.org/notes/note/596/
import sys
import re

def doLuhn(s, evenPos):
      s = str(s)
      sum = 0
      for d in reversed(s):
          d = int(d)
          assert 0 <= d <= 9
          if evenPos:
              d *= 2
              if d > 9:
                  d = (d % 10) + 1
          sum += d
          evenPos = not evenPos
      return sum

def luhnValidation(n):
      return doLuhn(n, False) % 10 == 0

def generateCheckDigit(n):
      return (10 - (doLuhn(n, True) % 10)) % 10

while 1:
      line = sys.stdin.readline()
      if not line: break
      number = line.strip()
```

```
      if re.match("^[0-9]{9,}$", number):
          print number + ": " + str(generateCheckDigit(number))
      else:
          print number
```

luhnCheck.py

```
#!/usr/bin/python
# From http://en.wikipedia.org/wiki/Luhn_algorithm
import sys
import re

def check_number(digits):
      _sum = 0
      alt = False
      for d in reversed(digits):
          d = int(d)
          assert 0 <= d <= 9
          if alt:
              d *= 2
              if d > 9:
                  d -= 9
          _sum += d
          alt = not alt
      return (_sum % 10) == 0

while 1:
      line = sys.stdin.readline()
      if not line: break
      number = line.strip()
      if re.match("^[0-9]{9,}$", number):
          print number + ": " + str(check_number(number));
      else:
```

```
        print number
```

chrome_bookmarks-extract_URLs.py

Extracts URLs of all Chrome bookmarks in the specified bookmark folder (or many folders, if they collide by name). Works recursively.

It's also a nice example for general JSON parsing and querying (XPath-like) using Python.

Requires the jsonpath Python module (`pypi-install jsonpath`).

```
#!/usr/bin/python

import os
import json
import jsonpath

bfile=open(os.path.expanduser("~/.config/chromium/Default/Bookmarks"))
bookmarks=json.load(bfile)
term_urls = jsonpath.jsonpath(bookmarks, "$..children[?(@.name=\"SomeBookmarkFolder\")]..url")

for url in term_urls:
        print url
```

url_encode.pl

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

s/([^A-Za-z0-9])/sprintf("%%%02X", ord($1))/seg;
```

url_decode.pl

```
#!/usr/bin/perl
$str=<>;
$str =~ s/\%([A-Fa-f0-9]{2})/pack('C', hex($1))/seg;
print $str;
print "\n";
```

html_escape_with_entities.pl

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



BEGIN { use HTML::Entities; }

{

 $_ = encode_entities($_, '<>&"');

}
```

ldap_ssha_make.py

```
#!/usr/bin/python

import hashlib

import sys

import os
```

```
from base64 import urlsafe_b64encode as encode

from base64 import urlsafe_b64decode as decode


def makeSecret(password):

    salt = os.urandom(4)

    h = hashlib.sha1(password)

    h.update(salt)

    return "{SSHA}" + encode(h.digest() + salt)


print "Input password: "

password = sys.stdin.readline().rstrip()

print "Hashed:"

print makeSecret(password)
```

ldap_ssha_check.py

```
#!/usr/bin/python

import hashlib

import sys
```

```
from base64 import urlsafe_b64encode as encode
from base64 import urlsafe_b64decode as decode

def checkPassword(challenge_password, password):
    challenge_bytes = decode(challenge_password[6:])
    digest = challenge_bytes[:20]
    salt = challenge_bytes[20:]
    hr = hashlib.sha1(password)
    hr.update(salt)
    return digest == hr.digest()

if len(sys.argv) < 2:
        print "Usage: "+sys.argv[0]+" password_hash"
else:
        pass_hash = sys.argv[1]
        print "Input password: "
        password = sys.stdin.readline().rstrip()
        print pass_hash+" "+password
        print "Check result: "+str(checkPassword(pass_hash, password))
```

generate_passwords_no_ambiguous_chars.sh

```
 apg -M SNCL -E "O5Ss01Il:;|6b"
```

soap_digest_check.sh

```
#!/bin/sh


nonceb64=$1

created=$2

pass=$3

(echo -n "$nonceb64" | base64 -d ; echo -n "$created"; echo -n "$pass") | sha1sum -b | awk '{print $1}' | xxd -c 40 -p
-r | base64
```

rename_to_random.sh

```
#!/bin/sh

if [ $# -ge 1 ]; then
        while [ $# -ne 0 ]; do
                if [ -e "$1" ]; then
                        random="$(dd if=/dev/urandom bs=512 count=1 2>/dev/null| md5sum -b | awk '{print $1}')"
                        basename=$(basename "$1")
                        dirname=$(dirname "$1")
                        echo "$dirname/$basename mv to $dirname/$random"
                        mv "$dirname/$basename" "$dirname/$random"
                else
                        echo "$1 nie istnieje"
                fi
```

```
                shift
        done
else
        echo "Uzycie: $0 pliki_lub_katalogi_do_przemianowania"
fi
```

rename_files_for_windows_fast.pl

```
#!/bin/bash


(find ./ -depth -type d; find ./ -depth -type f;) | perl -e "

use File::Basename;

my \$illegal_chars = qr/[\\ \\|\\&\\?\\!\\*\\[\\]\\(\\)\\,\\=\\:\\@\\']+/;

while (<>) {

  if (/\$illegal_chars/) {

    chomp;

    my \$old = \$_;

    my \$basename = basename(\$old);

    my \$dirname = dirname(\$old);

    \$basename =~ s/\$illegal_chars/_/g;

    \$basename =~ s/_{2,}/_/g;

    my \$new = \"\$dirname/\$basename\";

    print \"\$old ->\n-> \$new\n\";
```

```
    if ( -e \$new ) {

      print STDERR \"already exists: \$new\n\";

    } else {

    rename \$old, \$new;

    }

  }

}

"
```

rename_files_perl_expression.pl

```
#!/usr/bin/perl
# by OLO
# pią cze 25 14:29:24 CEST 2004
# 1) Przeszukuje rekrsywnie podkatalogi, metoda DFS
# 2) Sprawdza, czy nazwa obiektu pasuje do wzorca (arg 1)
# 2) Jesli tak, dokonuje okreslonej substytucji (arg 2)


use strict;
use Fcntl ':mode';
use Data::Dumper;

my $debug = 1;

my $pattern;
```

```
my $subst;

if ($#ARGV >= 1) {
  $pattern= qr/$ARGV[0]/;
  $subst = $ARGV[1];
} else {
  die "Przeszukuje rekursywnie biezacy podkatalog i przemianowuje zawartosc.\nUzycie:\n$0 wzorzec_PCRE substytucja_PCRE
\nPrzyklad:\n$0 '\\.doc' '.ole'\n";
}

my $dirname = '.';
my $mode = (stat($dirname))[2];

if (S_ISDIR($mode)) {
  if ($debug) { print "$dirname to katalog.\n"; }
  descend($dirname);
}

sub descend {

  my $dirname = shift;

  opendir DH, $dirname;

  my @descend_list;

  my @process_list;

  my $entry;

  my $pathname;

  while ($entry = readdir(DH)) {

    if ($entry !~ '^\.$' && $entry !~ '^\.\.$') {

      $pathname = $dirname.'/'.$entry;
```

```
    my $mode = (stat($pathname))[2];


    if (S_ISDIR($mode)) {
      push @descend_list, $pathname;
      #descend($entry);
    } elsif (-f $pathname) {
      push @process_list, $pathname;
    }


  }
}
closedir DH;
foreach $pathname (@process_list) {
  process($pathname);
}
foreach my $subdir (@descend_list) {
  descend($subdir);
  process($subdir);
}
```

```
}


sub process {
  my $pathname = shift;
  #if ($debug) { print "przetwarzanie $pathname\n"; }
  if ($pathname =~ /$pattern/) {
    #if ($debug) { print "$pathname pasuje do wzorca $pattern, dokonuje substytucji s/$pattern/$subst/g.\n"; }
    my $new_pathname = $pathname;
    $new_pathname =~ s/$pattern/$subst/g;
    print "$pathname -> $new_pathname\n";
    rename $pathname, $new_pathname;
  }


}
```

remove_newline_at_eof.pl

```
#!/usr/bin/perl -i -p
chomp if eof
```

Or directly from the shelll:

```
perl -i -pe 'chomp if eof' <filename>
```

reread_partition_table_blockdev.sh

```
#!/bin/sh

blockdev --rereadpt "$1"
```

scsi_rescan_bus.sh

```
#!/bin/sh

host_number="$1"

echo "1" > /sys/class/fc_host/host${host_number}/issue_lip

echo "- - -" > /sys/class/scsi_host/host${host_number}/scan
```

epoch2time.pl

Convert UNIX epoch to human readable time:

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

#

use strict;

my $time = $ARGV[0];
```

```
my ($sec, $min, $hour, $day,$month,$year) = (localtime($time))[0,1,2,3,4,5];

# You can use 'gmtime' for GMT/UTC dates instead of 'localtime'


print "Unix time ".$time." converts to ";

printf("%04d-%02d-%02d %02d:%02d:%02d\n", $year+1900, $month + 1, $day, $hour, $min, $sec);


my $localestrtime = localtime($time);


print "In locale format: $localestrtime\n"
```

epoch2time.sh

```
#!/bin/sh

date -d "@${1}"
```

epoch2time_bsd.sh

```
#!/bin/sh

date -j -f %s "${1}"
```

primes.sh

```
#!/bin/sh

(for int in `seq 1024`; do factor $int; done;) | grep '^[0-9]\+: [0-9]\+$'
```

mplayer_play_DVD_iso_image_with_subs.sh

```
#!/bin/sh

mplayer -sub $1 -subcp $2 -dvd-device $3 dvd://1
```

keytool_add_ca_cert.sh

```
#!/bin/sh
# default keystore pass: "changeit"
keytool -import -keystore /etc/java-1.5.0-sun/security/cacerts -trustcacerts -alias FIRMA -file PATH/TO/firma.pem
```

list PKCS#11 KeyStore

```
keytool -v -keystore NONE -storetype PKCS11 -providerClass sun.security.pkcs11.SunPKCS11   -providerArg /.../pkcs11.cfg
-list
```

list PKCS#12 KeyStore

```
keytool -v -keystore STORE.p12 -storetype PKCS12 -list
```

Import PKCS#12 to JKS KeyStore

keytool -importkeystore -srckeystore SRC.p12 -destkeystore DEST.jks -srcstoretype PKCS12 -deststoretype JKS -srcstorepass PASS1 -deststorepass PASS2

python_compile.sh

```
#!/bin/sh

(echo "import py_compile"; echo "py_compile.compile(\"$1\")") | python

(echo "import py_compile"; echo "py_compile.compile(\"$1\")") | python -O
```

terroristize.lisp

```
;This maker of food for the NSA Line Eater is copyright (C) Eli Gottlieb, December 26 2005.
;It's under the GNU General Public License version 2.0.
(defvar *dictionary* '("assasinate" "kill" "suicide bomb" "dirty bomb" "nuclear device"
              "Al-Quaeda" "insurgency" "Hamas" "Baath"
              "jihad" "Allah" "Islam"
              "Sears Tower" "Empire State Building" "White House" "Golden Gate Bridge" "New York City subway"
              "Iraq" "Afghanistan" "Palestine" "Iran" "Saudi Arabia"
              "Israel" "America" "England"
              "infidels"
              "Usama bin Laden"
              "London"))

(defun terroristize (lines)
  (if (not (equalp lines nil))
    (append
```

```
      (if (equalp (cdr lines) nil)
    (list (car lines))
    (list (car lines) (nth (random (length *dictionary*)) *dictionary*)))
      (terroristize (cdr lines)))
    nil))

(defun equal-to-any (value any)
  (cond
    ((equalp value (car any)) (car any))
    ((cdr any) (equal-to-any value (cdr any)))
    ('t nil)))

(defun all-whitespace-before-p (the-string start-index end-index)
  (do ((index start-index (- index 1)))
      ((equalp index end-index) T)
    (if (not (equalp (elt the-string index) #\ )) (return nil))))

(defun token-delimited-p (token-beginning index-of-char string-data delimiters)
  (and (equal-to-any (elt string-data index-of-char) delimiters) (not (all-whitespace-before-p string-data
index-of-char token-beginning))))

(defun tokenize (string-input delimiters &optional (inclusive nil))
  (setf tokens nil)
  (setf token-beginning 0)
  (dotimes (c (length string-input))
    (if (token-delimited-p token-beginning c string-input delimiters)
      (progn
    (setf tokens (append tokens (list (subseq string-input token-beginning (if inclusive (+ c 1) c)))))
    (setf token-beginning (+ c 1)))))
  (if (< token-beginning (length string-input))
    (append tokens (list (subseq string-input token-beginning)))
    tokens))


(defun parse-for-words (sentence)
```

```
  (tokenize sentence '(#\ )))

(defun parse-for-sentences (message)
  (tokenize message '(#\. #\? #\!) T))

(defun string-reglue (str1 str2 delimiter)
  (concatenate 'string (concatenate 'string str1 (string delimiter)) str2))

(defun list-to-string (list-input)
  (if (stringp (car list-input))
    (if (cdr list-input)
      (string-reglue (car list-input) (list-to-string (cdr list-input)) #\ )
      (car list-input))
    nil))

;This is the main function. Hand it an arbitrary string to be sprinkled with "terrorist lingo" ;-).
(defun feed-echelon (message)
  (setf sentences (mapcar 'parse-for-words (parse-for-sentences message)))
  (dotimes (sentence (length sentences))
```

```
    (setf (elt sentences sentence) (terroristize (elt sentences sentence))))
  (list-to-string (mapcar 'list-to-string sentences)))
; by OLO:
; usage:
; $ clisp -repl terroristize.lisp
; $ (feed-echelon "Hello world.")
```

jpegtran_from_jpegexif.sh

```
#!/bin/sh
for jpg in "$@"; do
    #randname="$(dd if=/dev/urandom bs=256 count=1 2>/dev/null | md5sum -b | cut -b 1-32).tmp"
    #touch -r "$jpg" -F 1 $randname
    echo $orient "$jpg";
    #jpegtran -copy all -rotate 90 "$jpg" > "$jpg.tmp" && mv -f "$jpg.tmp" "$jpg" &&  jpegexiforient -1 "$jpg";
    jhead -autorot -ft "$jpg"
    #touch -r $randname "$jpg"
    #rm $randname
done;
```

Exif_timestamps_to_file_modtime_in_curdir.py

```
#!/usr/bin/env python

import exifread
import os
import time



def get_exif_timestamp(filename):
    with open(filename) as f:
        tags = exifread.process_file(f, stop_tag='DateTimeOriginal')
        #print tags.keys()
        return tags.get('EXIF DateTimeOriginal')

def exif_ts_to_unix_ts(exif_ts):
    return time.mktime(time.strptime(exif_ts.values, "%Y:%m:%d %H:%M:%S"))

def set_file_ts_from_exif(filename, exif_ts):
    if exif_ts:
        unix_ts = exif_ts_to_unix_ts(exif_ts)
        os.utime(filename, (unix_ts, unix_ts))
        print filename + ' timestamp set to: ' + str(unix_ts) + ' (' + exif_ts.values + ')'
```

```
def main():
    for filename in os.listdir('.'):
        if not os.path.isfile(filename):
            continue
        try:
            exif_ts = get_exif_timestamp(filename)
            set_file_ts_from_exif(filename, exif_ts)
        except:
            print 'ERROR when processing file ' + filename

if __name__ == '__main__':
    main()
```

input_rate_notifier.pl

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

# by Aleksander Adamowski

# Wed Nov 23 23:26:12 CET 2005

# Skrypt wykrywa przekroczenie pewnej ilosci wierszy na ilosc sekund na STDIN

# i uruchamia dane polecenie powiadamiajace o przekroczeniu.
```

```
use strict;

if (scalar(@ARGV) < 3) {
  die "Uzycie: $0 ilosc_wierszy ilosc_sekund polecenie_powiadamiajace\n";
}

my $rowcount = $ARGV[0];
my $seconds = $ARGV[1];
my $notificator = $ARGV[2];
my @queue = ();

# Kiedy ostatnio bylo powiadomienie (moze byc nie czesciej, niz ilosc_sekund aby uniknac spamowania):
my $lastnotification = 0;

my $time;
my $tailtime;
my $queuetime;

while (<STDIN>) {
```

```
  $time = time;

  unshift @queue, $time;

  # Jesli kolejka jest pelna, zaczynamy zdejmowac wpisy z jej konca:

  if (scalar(@queue) > $rowcount) {

    pop @queue;

    # Czas na koncu kolejki:

    $tailtime = $queue[$#queue];

    # Jesli koniec kolejki jest malo oddalony w czasie (ponizej $seconds) od poczatku, to przeplywnosc jest
przekroczona:

    $queuetime = (time - $tailtime);

    if ($queuetime < $seconds && (time - $lastnotification) > $seconds) {

      # Powiadamiamy:

      #print "Rate EXCEEDED! $rowcount rows / $queuetime seconds\n";

      system $notificator or die $?;

      $lastnotification = time;

    }

  }

}
```

monitor_logfile.pl

```
#!/usr/bin/perl -w
# by OLO
# Tue Aug 16 11:41:26 CEST 2005
# Monitoruje przyrastajacy plik logow oczekujac na pojawienie sie wzorca.
# Po pojawieniu sie wzorca wysyla powiadomienie pod zadany adres e-mail.
use strict;
use File::Tail;
use Mail::Sendmail;
use Getopt::Std;

$| = 1;

our($opt_n);
getopts('n');

if (scalar(@ARGV) < 3) {
  print STDERR <<EOD
Uzycie:
 $0 [-n] plik_logu 'wzorzec regexp' adres_email_do_powiadomienia
EOD
;
        die("Nieprawidlowa skladnia.\n");
}

my $regexp_source = $ARGV[1];
my $regexp = qr/$regexp_source/;

my $line;
my $file=File::Tail->new(name => $ARGV[0], interval => 1, maxinterval => 2, ignore_nonexistant => 1);

while (defined($line=$file->read)) {

        if ($line =~ /$regexp/) {

                my $message = "W pliku ${ARGV[0]} pojawil sie wiersz odpowiadajacy wzorcowi $regexp.\n\nOto ten
```

```
wiersz:\n\n$line\n";

                my %mail = ( To      => $ARGV[2],

                                From    => 'postmaster@altkom.pl',

                                Subject => 'Pojawil sie wpis w logu odpowiadajacy wzorcowi',

                                Message => $message

                                );

                print $message;

                sendmail(%mail) or die $Mail::Sendmail::error;

    if (! $opt_n) {

      exit 0;

    }

        }

}
```

exec_timeout_kill.pl

```
#!/usr/bin/perl -w
# by OLO
# czw lis  3 11:25:23 CET 2005
# Skrypt uruchamia komende, czeka az komenda sie zakonczy lub minie timeout.
# Jesli komenda sie zakonczy w zadanym czasie, zwraca taki kod, jak ta komenda;
# jesli komenda dalej dziala - KILLuje ja najpierw SIGTERMem, potem SIGKILLem
# jesli to nie pomoze i zwraca kod 1 (jesli wystarczyl SIGTERM) lub kod 2 (SIGKILL byl konieczny).
```

```
use strict;
use Getopt::Std;
use POSIX;
#use POSIX ":sys_wait_h";

our($opt_t);
getopts('t:');

if (defined($opt_t) && defined($ARGV[0])) {
  #print "$opt_t\n";
  #print $ARGV[0]."\n";
  my $pid = fork();
  die "fork() failed: $!" unless defined $pid;
  if ($pid) {
    # Jestem tatusiem!
    my $countdown = int($opt_t);
    if (!$countdown > 0 ) {
      die("Nieprawidlowa wartosc timeout!\n");
    }
    for ($countdown = int($opt_t); $countdown > 0; $countdown--) {
      #print "$countdown\n";
      sleep 1;
      my $kid = waitpid($pid, WNOHANG);
      my $status = $?;
      if ($kid == 0) {
        #print "Dziecko dziala\n";
      } else {
        my $retval = ($status >> 8);
        #print "Retval: $retval\n";
        #$?>> 8
        exit $retval;
      }
    }
    # Minal timeout, a dziecko caly czas dziala
    # Killujemy zatem:
    kill SIGTERM, $pid;
```

```
  sleep 2;
  my $kid = waitpid($pid, WNOHANG);

  my $status = $?;

  if ($kid == 0) {

    print STDERR "Dziecko CIAGLE dziala, ubijam na twardo...\n";

    kill SIGKILL, $pid;

    exit 2;

  } else {

      my $retval = ($status >> 8);

      #print "Retval: $retval\n";

      #exit $retval;

      exit 1;

  }

}

else {

  # Da-da!

  exec(@ARGV) or die("Couldn't exec: ".join(' ', @ARGV));

  exit(0);

}
```

```
} else {

  print STDERR "Usage:\n$0 -t timeout command [args] \n";

}
```

ettercap_arp_sniff.sh

```
#!/bin/sh

ettercap -Tq -M arp //
```

amr_to_wav.sh

```
#!/bin/sh
mplayer "$1" -ao pcm:file="$1.wav" -vc null -vo null
```

convert_flac_to_mp3_subtree.sh

```
#!/bin/sh
if [ $# -ge 1 ]; then
find "$1" -type f -iname '*.flac' | xargs -n 1 dirname | uniq | perl -pe 's{^}{mp3/};' | xargs mkdir -p
```

```
for file in $(find "$1" -type f -iname '*.flac' ); do
          ARTIST=`metaflac "$file" --show-tag=ARTIST | sed s/.*=//g`
          TITLE=`metaflac "$file" --show-tag=TITLE | sed s/.*=//g`
          ALBUM=`metaflac "$file" --show-tag=ALBUM | sed s/.*=//g`
          GENRE=`metaflac "$file" --show-tag=GENRE | sed s/.*=//g`
          TRACKNUMBER=`metaflac "$file" --show-tag=TRACKNUMBER | sed s/.*=//g`
          DATE=`metaflac "$file" --show-tag=DATE | sed s/.*=//g`

          outfile="mp3/$(echo "$file" | perl -pe 's/.flac$/.mp3/i')";
          flac -d -c  "$file" | lame --preset standard - "$outfile";
          id3 -t "$TITLE" -T "${TRACKNUMBER:-0}" -a "$ARTIST" -A "$ALBUM" -y "$DATE" -g "${GENRE:-12}" "$outfile"
done;
else
          echo "Usage: $0 directory_with_flac_files"
fi
```

ffmpeg_transcode_avi_to_Android.sh

```
#!/bin/sh
# Using a xargs queue for 4 parallel processes to fully utilise a 4-core CPU:
(for krtekavi in ../*Krtek*.avi; do
        krtekandroid="$(basename "$krtekavi").mp4"
        echo " -threads 4 -strict experimental -i $krtekavi -s 480x320 -vcodec mpeg4 -acodec aac -ac 1 -ar 16000 -r 13
-ab 32000 -aspect 3:2 $krtekandroid"
done) | xargs -t -P 4 -L 1 ffmpeg
```

ldapsearch_get_suffixes.sh

```
#!/bin/sh
ldapsearch -x -H ldap://127.0.0.1 -b "" -s base "(objectclass=*)" namingContexts
```

AD_search_groups.sh

```
#!/bin/sh

if [ $# -lt 2 ]; then

      cat <<EOF >&2
Usage:
AD_search_groups.sh bind_DN member_DN
e.g.:
AD_search_groups.sh 'CN=Adamowski Aleksander,OU=Users,OU=Organization,DC=example,DC=com' 'CN=Adamowski
Aleksander,OU=Users,OU=Organization,DC=example,DC=com'
EOF
else
      ldapsearch -LLL -H ldap://your-dc-hostname-from-logonserver-windows-env-var -D "$1" -b DC=example,DC=com -W -x
"member=$2" name displayname mail
fi
```

iconv_utf8_to_ascii_translit

```
 iconv -f UTF-8 -t US-ASCII//TRANSLIT
```

wikiconv.pl

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

use strict;
use CGI;
```

```
my @standardFilteredTags = ('div', 'span', 'font');
my $q = CGI->new();
my $title = 'Konwerter HTML 2 MediaWiki';
print $q->header(-type=>'text/html', -charset=>'utf-8'),
        $q->start_html($title),
        $q->h1($title);


print $q->start_form(-method=>'POST'),
        $q->p('Filtrowane znaczniki HTML:'),
        $q->checkbox_group(-name=>'standard_tags',
            -values=> \@standardFilteredTags,
            -rows=>2, -columns=>2, -checked=>1),
        $q->p('Dodatkowe filtrowane znaczniki HTML (rozdziel spacjami):'),
          $q->textfield(-name=>'additional_tags',
              -size=>50),
          $q->p('Tekst HTML:'),
          $q->textarea(-name=>'htmltext',
              -default=>'<html><h1>Nagłówek 1</h1><h2>Nagłowek 2</h2><p><span class="klaska">Jakiś</span> <font
color="red">tekst</font></p></html>',
              -rows=>20,
              -columns=>120),
          $q->p($q->submit(-name=>'submit',
              -value=>'Konwertuj')),
          $q->endform;

if ($q->param('htmltext')) {
  use HTML::WikiConverter;
  my $wc = new HTML::WikiConverter( dialect => 'MediaWiki' );

  my $mediaWikiText = $wc->html2wiki($q->param('htmltext'));

  my @standards = $q->param('standard_tags');
  my @standardsFiltered = map {+s/[^a-zA-Z0-9]//g; $_} @standards;
  my @additionals = split(' ', $q->param('additional_tags'));
  my @additionalsFiltered = map {+s/[^a-zA-Z0-9]//g; $_} @additionals;
```

```
  my @tags = (@standardsFiltered, @additionalsFiltered);

  foreach my $tag (@tags){
      $mediaWikiText =~ s/\<\/?$tag[^>]*>//g;
  }

  print $q->h1('Wynik w formacie MediaWiki');
  print '<pre style="border: thin solid black; background-color: lightgrey;">';
  print "\n";
  print $q->escapeHTML($mediaWikiText);
#print $mediaWikiText;
  print "</pre>\n";
}
print $q->end_html;
```

wiki_indent

```
alias wiki_indent="perl -e '@all=<>; @out= map {\$_ = \" \".\$_;} @all; print @out; print \"\n\";'"
```

encode_BCD.pl

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

sub encode_BCD {
return pack "H*", join "", @_
};

my $decString = $ARGV[0];
my $bin = unpack("B*",encode_BCD($decString));
```

```
my $groupedBin = $bin;

$groupedBin =~ s/(.{4})/$1 /g;
print "Packed BCD representation of decimal string $decString:\n";
print "binary:\n";
print "$bin\n";
print "grouped binary:\n";
print "$groupedBin\n";
print "hexadecimal (the same as decimal):\n";
print "$decString\n";
```

jboss_list_JNDI

```
sh /opt/jboss/bin/twiddle.sh invoke jboss:service=JNDIView list true
```

Listuje zawartość JNDI.

jboss_scp_deploy_and_wait.sh

```
#!/bin/sh
# deploys EARs from a project to all server nodes of a JBoss cluster,
# waiting for successful deployment of each EAR on given server.
# The target servers _must_ have the Expect package that contains the
# "unbuffer" utility installed!

BUILD_DIR=$HOME/workspace/projectname
JBOSS_DIR=/opt/jboss/server/projectname

for host in server01 server02; do
      login=root@$host
      for ear in projectname-integration projectname-business; do
```

```
          fullpath=$(find $BUILD_DIR/ -wholename "*ear/target/$ear-?.?.?.ear" | head -n 1)
          echo ========================================================
          echo $fullpath

          # Make sure that the file is deemed newer than any previously rsync-ed:
          touch $fullpath
          rsync -vP -T $JBOSS_DIR/tmp $fullpath $login:$JBOSS_DIR/deploy/
          # Monitor remote log file, waiting until the app starts up:
          ssh $login "grep -m 1 'Started J2EE application.*$ear-[-0-9.]\+.ear' <(unbuffer -p tail -n 0 -f
$JBOSS_DIR/log/server.log)"
          echo "Deployment of $ear on $host ended."
          sleep 1
      done;
done;
```

sizeof_void.c

```
#include <stdio.h>
int main(int argc, char *argv[]) {
        printf("=== Pointers: ===\n");
        printf("sizeof(void *) is %d\n", (int)sizeof(void *));
        printf("sizeof(int *) is %d\n", (int)sizeof(int *));
        printf("sizeof(char *) is %d\n", (int)sizeof(char *));
        printf("=== Integral types: ===\n");
        printf("sizeof(unsigned char) is %d\n", (int)sizeof(unsigned char));
        printf("sizeof(short) is %d\n", (int)sizeof(short));
        printf("sizeof(char) is %d\n", (int)sizeof(char));
```

```
        printf("sizeof(int) is %d\n", (int)sizeof(int));
        printf("sizeof(long) is %d\n", (int)sizeof(long));
        printf("sizeof(long long) is %d\n", (int)sizeof(long long));
        return 0;
}
```

```
gcc -Wall -pedantic -o sizeof_void sizeof_void.c
```

Na 32-bitowej zobaczysz:

```
=== Pointers: ===
sizeof(void *) is 4
sizeof(int *) is 4
sizeof(char *) is 4
=== Integral types: ===
sizeof(unsigned char) is 1
sizeof(short) is 2
sizeof(char) is 1
sizeof(int) is 4
sizeof(long) is 4
sizeof(long long) is 8
```

Na 64-bitowej natomiast:

```
=== Pointers: ===
sizeof(void *) is 8
sizeof(int *) is 8
sizeof(char *) is 8
=== Integral types: ===
sizeof(unsigned char) is 1
sizeof(short) is 2
sizeof(char) is 1
sizeof(int) is 4
sizeof(long) is 8
sizeof(long long) is 8
```

Jak widać jedyna różnica jest w rozmiarze long-ów (model LP64) i wskaźników.

phl_dump_token.sh

```
#!/bin/sh
(echo 6; echo N; echo; echo 99;) |/opt/PSV_linux_client/csv/utils/phl
```

phl_object_deleter.sh

```
#!/bin/sh
(echo 11; echo; echo; echo 35; echo; echo 38; echo; echo; echo 38; echo; echo;exit; ) |
/opt/PSV_linux_client/csv/utils/phl
```

explode_zips.sh

```
#!/bin/sh
# Iteratively explode all ZIP/JAR/WAR/EAR archives in the current dir and all the archives they contain.

foundzips=1
while [ $foundzips -gt 0 ]; do
  IFS=$'\n'
  foundzips=0
```

```
  # Not very efficient to re-find for each sublevel, but it's simple:
  for zipfile in $(find ./ -type f -iname '*.zip' -or -iname '*.[jwe]ar'); do
      foundzips=1
      echo "Archive: [$zipfile]"
      expdir="$zipfile.exploded"
      if [ ! -e "$expdir" ]; then
        dirpushed=0
        zipfilebasename="$(basename "$zipfile")"
        echo "making $expdir" && \
        mkdir "$expdir" && \
        echo "entering $expdir" && \
        pushd "$expdir" && \
        dirpushed=1 && \
        echo "unzipping $PWD/../$zipfilebasename" && \
        unzip "../$zipfilebasename" && \
        echo "removing $PWD/../$zipfilebasename" && \
        rm "../$zipfilebasename" ||
        mv "../$zipfilebasename" "../$zipfilebasename.bad"
        if [ $dirpushed -gt 0 ]; then
          popd
        fi;
      fi;
      finish=0
  done;
done;
```

rename_jpg_based_on_exif_timestamp.sh

```
#!/bin/sh
for jpg in *.jpg; do
          tag="$(exif -m -t 0x9003 "$jpg" | perl -pe 'tr/: /-_/;')"
```

```
          mv -i "Renaming [$jpg] to [$tag.jpg]"
          mv -i "$jpg" "$tag.jpg"
done
```

do_postmark_tests.sh

```
#!/bin/sh
# (c) 2010 Aleksander Adamowski
# This script benchmarks various GPT partition offsets
# in order to find an optimal one for performance.
#
# Written for optimal partitioning of my new 4 kB sector
# Western Digital Advanced Format HDD (WD15EARS).
#

warning="WARNING!!!
This script completely erases significant portions of the configured block
device (the \"device\" variable specified in its body).

Shoots first, asks questions later.

In order to certify that you know what you're doing and accept that you WON'T
HOLD THE AUTHOR OF THIS SCRIPT LIABLE FOR ANY DAMAGES THAT IT MIGH CAUSE,
you have to modify the script by removing  or commenting out the \"exit 5\"
command that prevents it from running.
EOD"

echo "$warning"

exit 5

# Here begins the actual script logic
device=sdb
export LANG=C
```

```
# part_offset in sectors (512B unless the device reports other size to parted):
part_offset=41

for part_offset in $(seq 41 64); do
      umount /mnt/${device}1
      parted /dev/$device unit s rm 1
      sleep 1
      parted_out="$(parted -s /dev/$device unit s "mkpart primary ext2 $part_offset -1")"
      if [ $? -ne 0 ]; then
          echo "Parted error:"
          echo "$parted_out"
          last_sector="$(echo "$parted_out" | fgrep 'closest location we can manage' | perl -e '$_ = <>; if ($_ =~ /to
([0-9]+)s./) { print $1; }' )"
          echo "Last possible partition sector:"
          echo "[$last_sector]"
          parted_out="$(parted -s /dev/$device unit s "mkpart primary ext2 $part_offset $last_sector")"
          if [ $? -ne 0 ]; then
              echo "FATAL ERROR: could not determine proper sector range for new partition:"
              echo "$parted_out"
              exit 1
          fi
      fi
      sleep 1
      parted /dev/$device unit s print | tee -a "parted_dev_${device}_partition_at_${part_offset}s.txt"
      sync
      mkfs.ext4 -T largefile4 /dev/${device}1
      mount /dev/${device}1 /mnt/${device}1
      ls -la /mnt/${device}1
      cat /proc/partitions  | egrep "($device|major)" > "proc_partitions_${device}_start_at_${part_offset}s.txt"
      sync
      sleep 1
      echo "Executing postmark-quick with partition offset $part_offset:"
      postmark postmark-quick.conf | tee -a "postmark-quick_${device}_${part_offset}.txt"

      echo "Sleeping 5 seconds..."
      sleep 5
```

```
done
```

lookup_IP_country.sh

```
#!/bin/sh
# Example usage:
# List countries with which you currently have established TCP connections:
# netstat -anp | fgrep ESTABLISHED | awk '{print $5}' | sort | uniq | fgrep -v 127.0.0.1 \
# | perl -pe 's/:[^:]+$/\n/;' | xargs -n 1 lookup_IP_country.sh

ip_addr=$1
rev_ip_addr="$(echo "$ip_addr" | perl -pe 's/^([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})$/$4.$3.$2.$1/;')"
# 227.73.63.60.cc.iploc.org descriptive text "CN"
host -t TXT "${rev_ip_addr}.cc.iploc.org" | perl -pe 's/^[0-9.]+\.cc\.iploc\.org descriptive text
\"([A-Za-z0-9]+)\"$/$1/;'
```

ss_list_listening_sockets.sh

List listening TCP sockets for selected port numbers:

```
#!/bin/bash
# Example usage:
# ss_list_listening_sockets.sh 22 80 443 8089

filter=''

while (($#)); do
    if [ $notfirst ]; then
        filter="$filter or"
```

```
    fi
    filter="$filter sport = $1"
    notfirst=1
    shift
done

echo "FILTER: $filter"

ss -tlnup "$filter"
```

timed-run-err

Expect's timed-run example script forks a command and if it does not complete in a given time, kills it.
This variant of the timed-run example script exists with an error code when timeout occurs:

```
#!/bin/sh
# \
exec expect -f "$0" ${1+"$@"}
# run a program for a given amount of time
# i.e. time 20 long_running_program

set timeout [lindex $argv 0]
eval spawn [lrange $argv 1 end]
expect {
      timeout  {
          exit 1
      }
}
```

ctags_php

```
ctags -R --langmap=php:+.inc *
```

Minimal.desktop

```
[Desktop Entry]
Type=Application
Name=ArgoUML
Exec=/var/soft/argouml/argouml.sh
Icon=/var/soft/argouml/icon/ArgoIcon512x512.png
```

node-reformat_JSON.js

```
#!/usr/bin/node
var fs = require('fs');
var input = fs.readFileSync('/dev/stdin').toString();
console.log(JSON.stringify(JSON.parse(input), null, 2));
```

XML Starlet

Extract module identifier (groupId:artifactId) from a pom.xml file using XMLStarlet (note that the maven namespace must be assigned to a prefix):

```
xml sel -N 'm=http://maven.apache.org/POM/4.0.0' -t -c '/m:project/m:groupId/text()' -c 'string(":")' \\
  -c '/m:project/m:artifactId/text()' pom.xml
```

screentitle.sh

GNU Screen - set titles for all current and future windows of the current screen session:

```
#!/bin/sh
newtitle=$1
screen -X eval "at \\\# title $newtitle" "shelltitle $newtitle" "hardstatus string $newtitle"
```

screenfitall.sh

GNU Screen - fit all windows of the current session to current terminal size (e.g. after resizes):

```
#!/bin/sh
screen -X eval "at \\# fit"
```

Note that the same can be achieved interactively within a screen session by hitting:
CTRL-a, :
then typing the screen command:
`at \# fit`

randbetween.sh

Generate a random integer number between two values (not less than 0 and not greater than 32767)

```
#!/bin/bash
echo $(( ( RANDOM % ($2 - $1 + 1 ) ) + $1 ));
```

ps_backtrace.sh

get a ps process listing for the current process and all its ancestors up to init

```
#!/bin/bash

if [ $# -lt 1 ]; then
  echo "Usage: $0 PID" > /dev/stderr
  exit 1
fi

declare -i pid=$1;
ppid=0;
header_modifier="";
while : ; do
  if [ $ppid -ne 0 ]; then
    header_modifier=h;
  fi;
  ppid=$(ps -o ppid= $pid);
  ps uww $header_modifier -p $pid;
  if [ $pid -eq 1 ]; then
    break;
  fi;
```

```
  pid=$ppid;
done;
```

Usage:

```
$ ps_backtrace.sh PROCESS_PID
```

e.g.:

```
$ ps_backtrace.sh $$
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
olo      3372425  0.0  0.0 108180  3632 pts/23   Ss   11:29   0:00 /bin/bash
olo       212112  0.0  0.2 326644 208496 ?       Ss   Jan27   0:15 SCREEN -S www
olo       212111  0.0  0.0 118996   980 pts/13   S+   Jan27   0:04 screen -S www
olo        14054  0.0  0.0 107868  3200 pts/13   Ss   Jan27   0:00 -bash
...
root           1  0.0  0.0  19280   944 ?        Ss    2014   6:44 /sbin/init
```

ps_descendants.sh

```
#!/bin/sh
pstree -p $1 | perl -ne 'while (/\((\d+)\)/g) { print "$sep$1"; $sep="," }'
```

ps_blocked_processes.sh

```
#!/bin/sh
ps axr
```

pid_listening_on_tcp_port.sh

```
#!/bin/sh
port=$1
lsof -i TCP:$port
```

regexp_optimizer.pl

```
#!/usr/bin/perl
use Regexp::Optimizer;
$big_re = "";
$first = 1;
while ($_ = <>) {
  chomp $_;
  if (!$first) {
    $big_re .= "|"
  }
  $first = 0;
  $big_re .= $_;
}
```

```
print Regexp::Optimizer->new->optimize(qr/$big_re/);
print "\n";
```

Microsoft Windows

PowerShell, DOS batch etc.

PowerShell

List certificates and their fingerprints

```
gci cert:\CurrentUser\My
```

Additionally, display their expiration dates (the **NotAfter** property):

```
gci -recurse cert:\CurrentUser\My | ft -wrap -Property Thumbprint,Issuer,Subject,NotAfter
```

Show private key info for certificate given by fingerprint

```
(gci cert:\CurrentUser\My | ? {$_.thumbprint -like "FINGERPRINT_HEX"}).PrivateKey
```

CSP private key's container info:

```
(gci cert:\CurrentUser\My | ? {$_.thumbprint -like "FINGERPRINT_HEX"}) | ForEach-Object
```

```
{$_.PrivateKey.CspKeyContainerInfo}
```

For RSA keys, the property UniqueKeyContainerName points to file name in `[System.Environment]::GetFolderPath([System.Environment+SpecialFolder]::ApplicationData) + "\Microsoft\Crypto\RSA\"` on Windows 7 this typically is `c:\Users\username\AppData\Roaming\Microsoft\Crypto\RSA\`

Locate the file pathname that holds the given certificate's RSA Private Key belonging to the current user:

```
(gci cert:\CurrentUser\My | ? {$_.thumbprint -like "426027FBB6742904212CB813C416F5EF63D8EEAB"}) | `
  ForEach-Object { `
    [System.Environment]::GetFolderPath([System.Environment+SpecialFolder]::ApplicationData) + `
    "\Microsoft\Crypto\RSA\" + `
    [System.Security.Principal.WindowsIdentity]::GetCurrent().User.Value + "\" + `
    $_.PrivateKey.CspKeyContainerInfo.UniqueKeyContainerName `
  }
```

Determine LDAP DN for a given Active Directory user

```
$login_ad = 'LOGIN_AD';
$searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]''); $searcher.Filter =
"(&(objectClass=user)(samaccountname=$login_ad))"; $adfind = $searcher.FindAll(); $adfind[0].Properties.adspath
```

Add/update thumbnail photo in AD for a given user

```
$login_ad = "aadamowski";
$photo_path = "C:\temp\foto.jpg";
```

```
$searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]'');
$searcher.Filter = "(&(objectClass=user)(samaccountname=$login_ad))";
$adfind = $searcher.FindAll();
$user=[adsi]$adfind[0].Path;
$photo = [byte[]](Get-Content $photo_path -Encoding byte)
$user.Properties["thumbnailphoto"].clear()
$user.Properties["thumbnailphoto"].add($photo)
$user.CommitChanges()
```

Dump thumbnail photos from AD to files

```
$searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]'');
$searcher.Filter = "(ANY LDAP FILTER)"
$adfind = $searcher.FindAll();
foreach ($user in $adfind) { if ($user.Properties['thumbnailphoto']) { $thumbnailphoto =
$user.Properties['thumbnailphoto'].Item(0); $uid = $user.Properties['samaccountname'];
[System.IO.File]::WriteAllBytes("C:\temp\photos\$uid.jpg", $thumbnailphoto); } }
```

Convert a file / Active Directory timestamp to Date/Time:

```
$timestamp=129699324000000000
[System.DateTime]::FromFileTime($timestamp)
```

Override locale for given command:

```
[threading.thread]::currentThread.currentCulture = 'en-US'; [System.DateTime]::FromFileTime($timestamp)
```

Doing it within a single line is crucial!

List an AD group's members

```
$groupName = 'GROUP_NAME'
$searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]'');
$searcher.Filter = "(&(objectclass=group)(cn=$groupName))"
$adfind = $searcher.FindAll();
foreach ($group in $adfind) {"============"; $group.Properties['cn']; "------------"; $group.Properties['member']}
```

Find an AD user by his mobile number

WARNING: slow and taxing for the AD server (due to complex, wildcarded search filter's pattern)!

```
$mobileNumberSearchPattern=*666*777*888*
$searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]'');
$searcher.Filter = "(mobile=$mobileNumberSearchPattern)";
$adfind = $searcher.FindAll();
$adfind[0].Properties;
```

JavaScript

Speed up all videos on page

```
[...document.querySelectorAll('video')].forEach(v => v.playbackRate = 1.5)
```

For more of my stuff, visit http://olo.org.pl