История, философия и основы работы в UNIX

POSIX

IEEE Standard for Information Technology-Portable Operating System Interface Base Specifications, Issue 7, IEEE Std 1003.1-2017

POSIX defines a standard operating system interface and environment, including a command interpreter (or “shell”), and common utility programs to support **applications portability at the source code level**.

https://pubs.opengroup.org/onlinepubs/9699919799

https://pubs.opengroup.org/onlinepubs/9699919799/download/index.html

Single UNIX Specification, Single UNIX Specification - Википедия

standards(7)

POSIX-certified:

- AIX
- HP-UX
- IRIX
- UnixWare
- Solaris
- macOS 10.5

Mostly POSIX-compliant:

- BeOS & Haiku
- Minix
- Xenix
- *BSD
- GNU/Linux (`POSIXLY_CORRECT`; Linux Standard Base, ISO/IEC 23360-1:2006)

POSIX for Windows:

- MSVCRT & Winsock, the support remains largely incomplete and not fully interoperable.
- Cygwin

- MinGW

Состав POSIX:

- Base definitions
- Shell and utilities
- System interfaces
- Rationale

Важнейшие понятия:

- пользователь
- файл
- процесс
- терминал
- хост
- узел сети
- время
- языково-культурная среда
- системный вызов

C POSIX Library

https://gnu.org/software/libc/manual/html_node/Feature-Test-Macros.html

Glibc feature test macros

```
#define _POSIX_C_SOURCE 200112L
```

Системные вызовы

```
    const char msg[] = "Hello World";
    write(1, msg, sizeof(msg));
```

```
section .data
    msg: db "Hello World", 0x0
    len: equ $-msg

section .text
    mov rax, 1     ; use the `write` syscall
    mov rdi, 1     ; write to stdout
    mov rsi, msg   ; use string "Hello World"
    mov rdx, len   ; write 11 characters
    syscall        ; make system call
```

Direct Operating System Access via Syscalls

x86 Assembly/Interfacing with Linux - Wikibooks

Searchable Linux Syscall Table for x86 and x86_64

Философия UNIX

https://ru.wikipedia.org/wiki/Философия_Unix

The Unix Philosophy: A Brief Introduction, перевод

Deconstructing the “Unix philosophy”, перевод

Design programs to do only a single thing, but to do it well, and to work together well with other programs.

Введение в UNIX

- Unix shell: абсолютно первые шаги
- Command line crash course
- Learn Enough Command Line to Be Dangerous
- Краткий справочник по «всем-всем» командам Linux

- Unix Toolbox, перевод
- Шпаргалка по работе в Vim

Интерфейс командной строки

argv silliness, CVE-2021-4034

Command Line Interface Guidelines

getopt

getopt(3)

```
#include <stdio.h>
#include <stdlib.h>
#include <getopt.h>

int main (int argc, char **argv) {
    int c;
    int digit_optind = 0;

    while (1) {
        int this_option_optind = optind ? optind : 1;
        int option_index = 0;
        static struct option long_options[] = {
            {"add", 1, 0, 0},
            {"append", 0, 0, 0},
            {"delete", 1, 0, 0},
            {"verbose", 0, 0, 0},
            {"create", 1, 0, 'c'},
            {"file", 1, 0, 0},
            {0, 0, 0, 0}
        };

        c = getopt_long (argc, argv, "abc:d:012",
                 long_options, &option_index);
        if (c == -1)
            break;

        switch (c) {
        case 0:
            printf ("параметр %s", long_options[option_index].name);
            if (optarg)
                printf (" с аргументом %s", optarg);
            printf ("\n");
            break;

        case '0':
        case '1':
        case '2':
            if (digit_optind != 0 && digit_optind != this_option_optind)
              printf ("цифры используются в двух разных элементах argv.\n");
            digit_optind = this_option_optind;
            printf ("параметр  %c\n", c);
            break;

        case 'a':
            printf ("параметр a\n");
            break;

        case 'b':
            printf ("параметр b\n");
            break;

        case 'c':
            printf ("параметр c со значением `%s'\n", optarg);
            break;

        case 'd':
            printf ("параметр d со значением `%s'\n", optarg);
            break;

        case '?':
            break;

        default:
            printf ("?? getopt возвратило код символа 0%o ??\n", c);
        }
    }

    if (optind < argc) {
        printf ("элементы ARGV, не параметры: ");
        while (optind < argc)
            printf ("%s ", argv[optind++]);
        printf ("\n");
    }

    exit (0);
}
```

popt

https://github.com/rpm-software-management/popt

```
#include <stdio.h>
#include <stdlib.h>
#include <popt.h>

void usage(poptContext optCon, int exitcode, char *error, char *addl) {
    poptPrintUsage(optCon, stderr, 0);
    if (error) fprintf(stderr, "%s: %s", error, addl);
    exit(exitcode);
}

int main(int argc, char *argv[]) {
   char    c;             /* used for argument parsing */
   int     i = 0;         /* used for tracking options */
   char    *portname;
   int     speed = 0;     /* used in argument parsing to set speed */
   int     raw = 0;       /* raw mode? */
   int     j;
   char    buf[BUFSIZ+1];
   poptContext optCon;    /* context for parsing command-line options */

   struct poptOption optionsTable[] = {
      { "bps", 'b', POPT_ARG_INT, &speed, 0,
                               "signaling rate in bits-per-second", "BPS" },
      { "crnl", 'c', 0, 0, 'c',
                               "expand cr characters to cr/lf sequences", NULL },
      { "hwflow", 'h', 0, 0, 'h',
                               "use hardware (RTS/CTS) flow control", NULL },
      { "noflow", 'n', 0, 0, 'n',
                               "use no flow control", NULL },
      { "raw", 'r', 0, &raw, 0,
                              "don't perform any character conversions", NULL },
      { "swflow", 's', 0, 0, 's',
                              "use software (XON/XOF) flow control", NULL } ,
      POPT_AUTOHELP
      { NULL, 0, 0, 0, 0,  NULL, NULL }
    };

   optCon = poptGetContext(NULL, argc, argv, optionsTable, 0);
   poptSetOtherOptionHelp(optCon, "[OPTIONS]* <port>");

   if (argc < 2) {
      poptPrintUsage(optCon, stderr, 0);
      exit(1);
   }

   /* Now do options processing, get portname */
   while ((c = poptGetNextOpt(optCon)) >= 0) {
      switch (c) {
       case 'c':
         buf[i++] = 'c';
         break;
       case 'h':
         buf[i++] = 'h';
         break;
       case 's':
         buf[i++] = 's';
         break;
       case 'n':
         buf[i++] = 'n';
         break;
      }
   }
   portname = poptGetArg(optCon);
   if((portname == NULL) || !(poptPeekArg(optCon) == NULL))
      usage(optCon, 1, "Specify a single port", ".e.g., /dev/cua0");

   if (c < -1) {
      /* an error occurred during option processing */
      fprintf(stderr, "%s: %s\n",
              poptBadOption(optCon, POPT_BADOPTION_NOALIAS),
              poptStrerror(c));
      return 1;
   }

   /* Print out options, portname chosen */
   printf("Options  chosen: ");
   for(j = 0; j < i ; j++)
      printf("-%c ", buf[j]);
   if(raw) printf("-r ");
   if(speed) printf("-b %d ", speed);
   printf("\nPortname chosen: %s\n", portname);

   poptFreeContext(optCon);
   exit(0);
}
```

```
Usage: a.out [OPTIONS]* <port>
  -b, --bps=BPS     signaling rate in bits-per-second
  -c, --crnl        expand cr characters to cr/lf sequences
  -h, --hwflow      use hardware (RTS/CTS) flow control
  -n, --noflow      use no flow control
  -r, --raw         don't perform any character conversions
  -s, --swflow      use software (XON/XOF) flow control

Help options:
  -?, --help        Show this help message
      --usage       Display brief usage message
```

docopt

https://github.com/docopt/docopt.c

```
Naval Fate.

Usage:
  naval_fate ship create <name>...
  naval_fate ship <name> move <x> <y> [--speed=<kn>]
  naval_fate ship shoot <x> <y>
  naval_fate mine (set|remove) <x> <y> [--moored|--drifting]
  naval_fate --help
  naval_fate --version

Options:
  -h --help     Show this screen.
  --version     Show version.
  --speed=<kn>  Speed in knots [default: 10].
  --moored      Moored (anchored) mine.
  --drifting    Drifting mine.
```

```
python -m docopt_c -o docopt.c example.docopt
```

```c
#include "docopt.h"

int main(int argc, char *argv[])
{
    struct DocoptArgs args = docopt(argc, argv,
        /* help */ 1, /* version */ "2.0rc2");

    puts("Commands");
    printf("\tmine == %s\n", args.mine ? "true" : "false");

    puts("Arguments");
    printf("\tx == %s\n", args.x);

    puts("Flags");
    printf("\t--drifting == %s\n", args.drifting ? "true" : "false");

    return EXIT_SUCCESS;
}
```

gengetopt

GNU Gengetopt

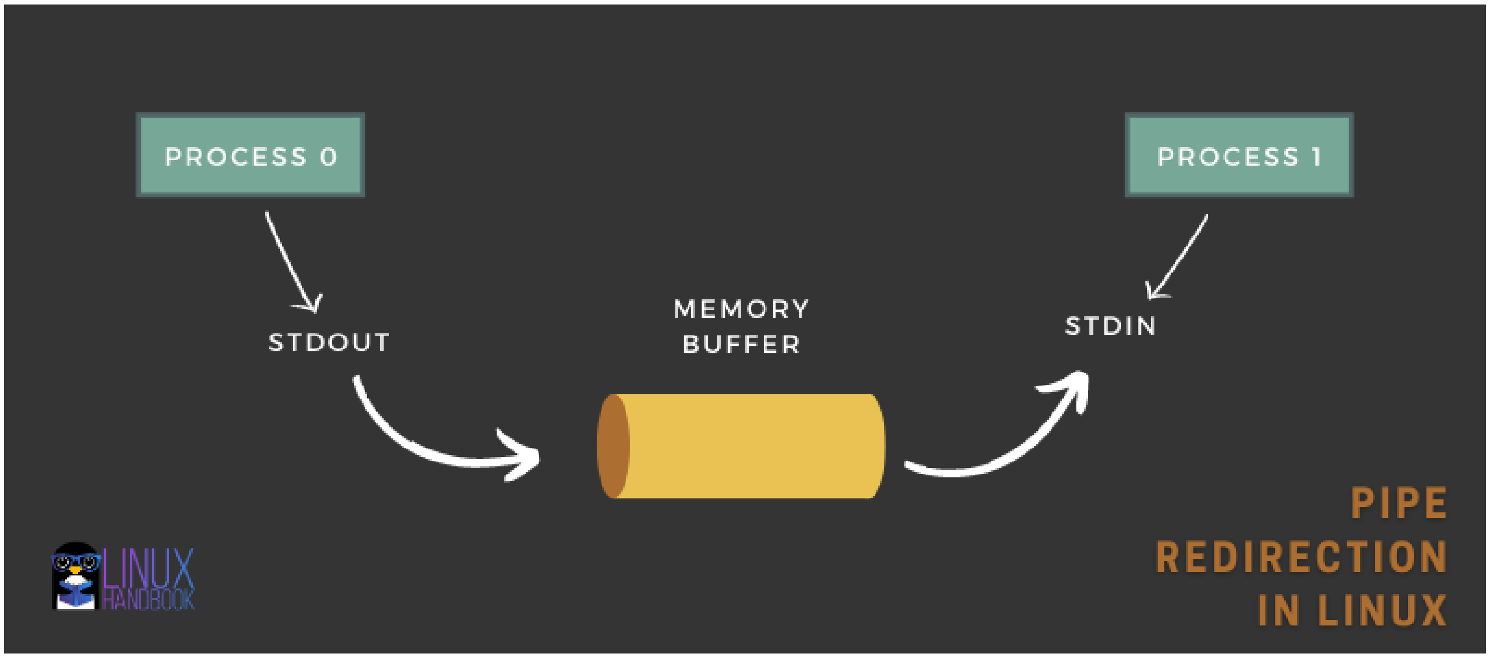
```
# cmdline.ggo
package "server"
purpose "Highload Cup entry"
version "2.0"

option "port" p
```

```
  "Port to listen on"
  int default="8080"
  typestr="PORT"
  optional

option "verbose" v
  "Be verbose"
  flag off

option "data" d
  "Location of data.zip file"
  string default="/tmp/data/data.zip"

option "threads" t
  "Number of threads to launch"
  int default="4"
  optional
```

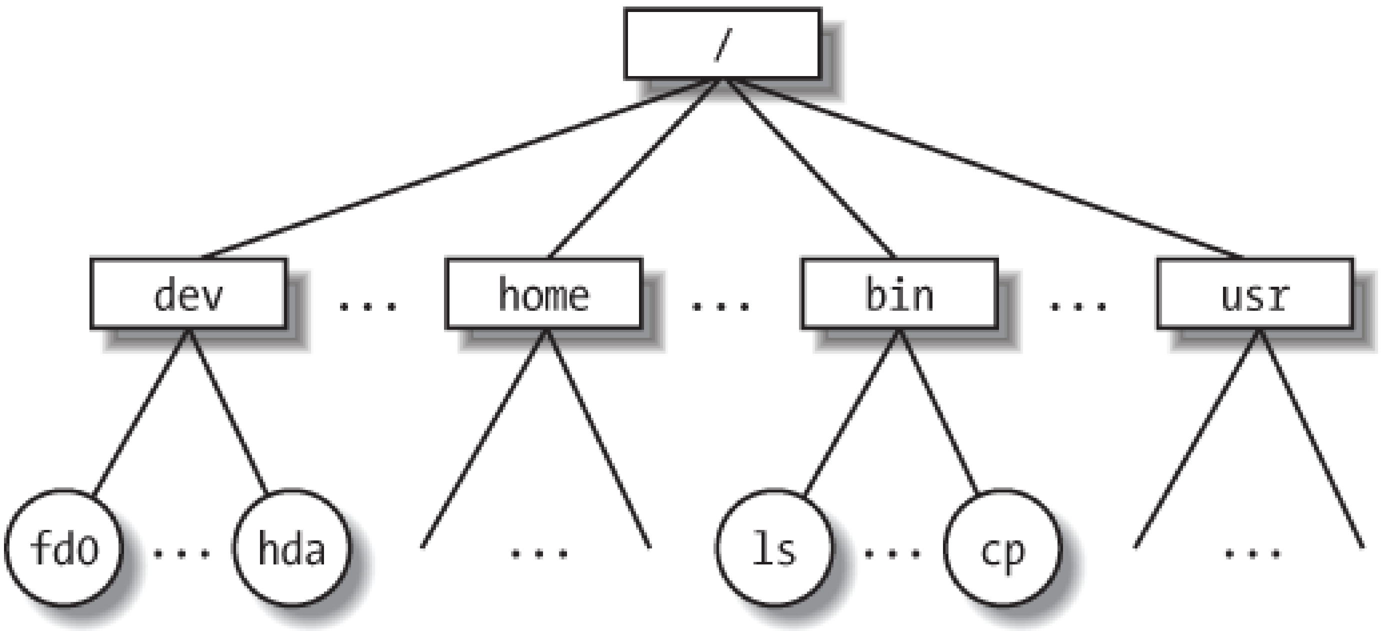
```
$ ./server --help
server 2.0

Highload Cup entry

Usage: server [OPTIONS]...

  -h, --help          Print help and exit
  -V, --version       Print version and exit
  -p, --port=PORT     Port to listen on  (default=`8080')
  -v, --verbose       Be verbose  (default=off)
  -d, --data=STRING   Location of data.zip file  (default=`/tmp/data/data.zip')
  -t, --threads=INT   Number of threads to launch  (default=`4')
```

Перенаправление ввода-вывода

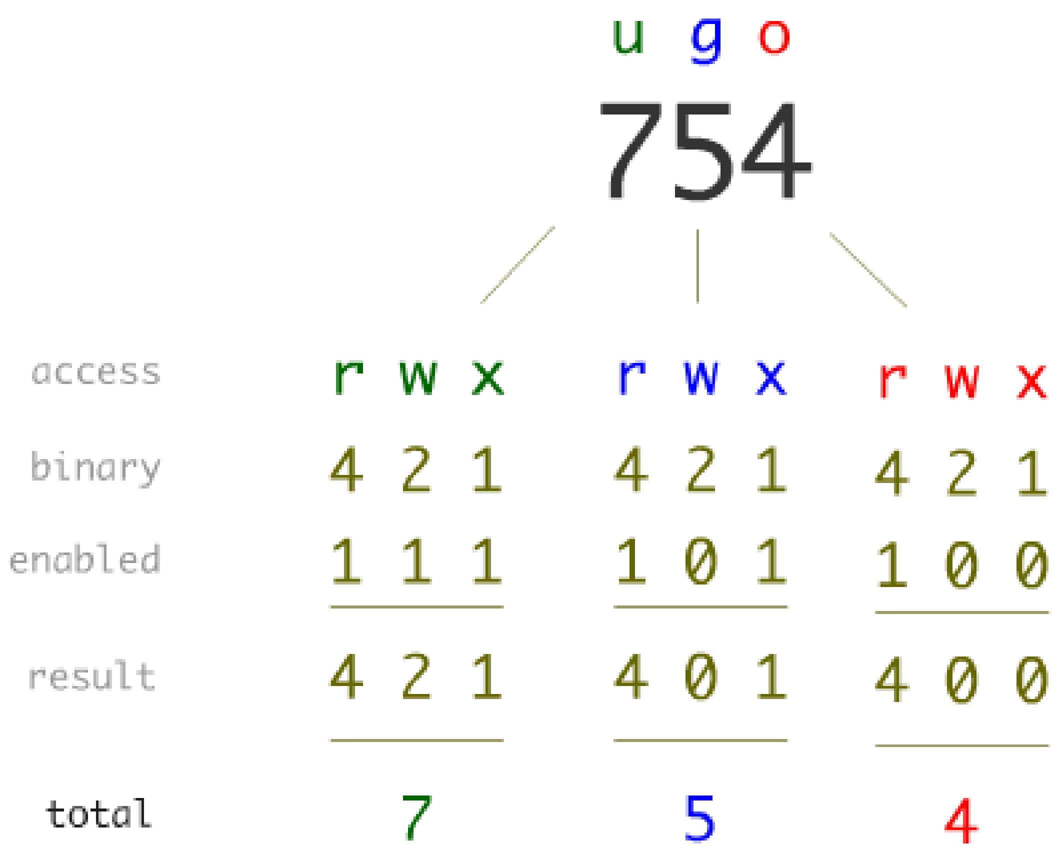
```
ls / | grep tmp
```



Файловая система



chroot(2)



chmod(2)

stat(2)

Инструменты сборки

Make

Владимир Игнатов, Эффективное использование GNU Make

```
all: hello

hello: main.c libmy.a
    $(CC) $(CFLAGS) -Wall -Wextra -pedantic -std=c11 `pkg-config --cflags --libs libcurl` $^ -o $@

libmy.a: my.o
    $(AR) rcs $@ $^

my.o: my.c
    $(CC) -c $(CFLAGS) -Wall -Wextra -Wpedantic -std=c11 $^ -o $@

clean:
    $(RM) hello libmy.a core *.o

.PHONY: all clean
```

Automatic Variables

Built-in Rules

BSD make vs GNU make: GNUMakefile

```
GNUMAKE?=gmake

all:
    ${GNUMAKE} $@

.DEFAULT:
    ${GNUMAKE} $@

.PHONY: all
```

A Tutorial on Portable Makefiles

Makefile best practices

- Makefiles should remove all content that it will generate.
- Dependency builds - A second invocation of make within a sandbox should always be a NOP.
- Hardcoded values - avoid them like the plague.
- Always include dependencies when creating a target.

Makefiles - Best practices and suggestions

Makefiles, Best Practices

Your Makefiles are wrong

Reproducible Builds

pkg-config

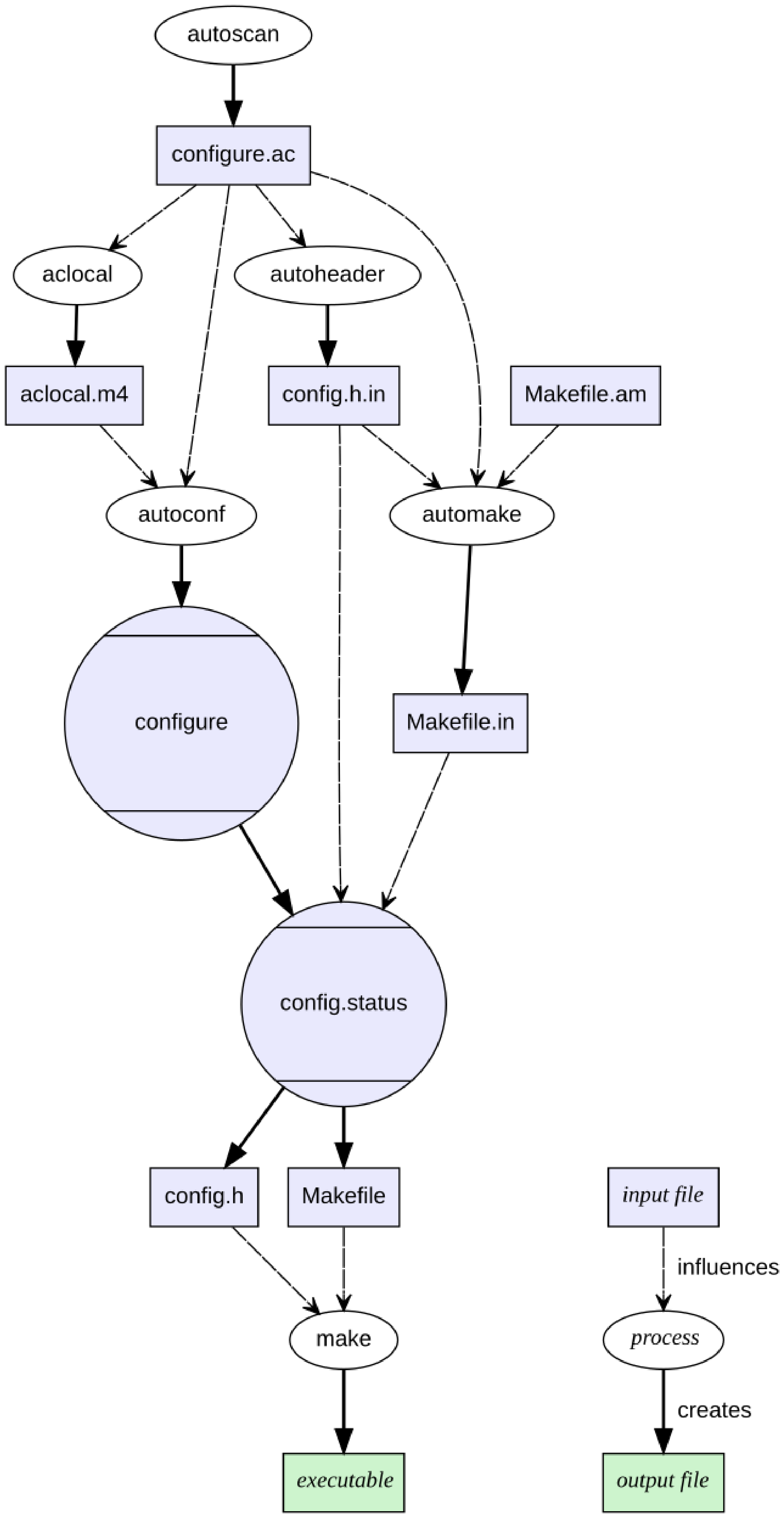
pkg-config was originally designed for Linux, but it is now also available for BSD, Microsoft Windows, macOS, and Solaris.

Guide to pkg-config

```
# /usr/lib/x86_64-linux-gnu/pkgconfig/libpng.pc

prefix=/usr/local
exec_prefix=${prefix}
libdir=${exec_prefix}/lib
includedir=${exec_prefix}/include

Name: libpng
Description: Loads and saves PNG files
Version: 1.2.8
Libs: -L${libdir} -lpng12 -lz
Cflags: -I${includedir}/libpng12
```

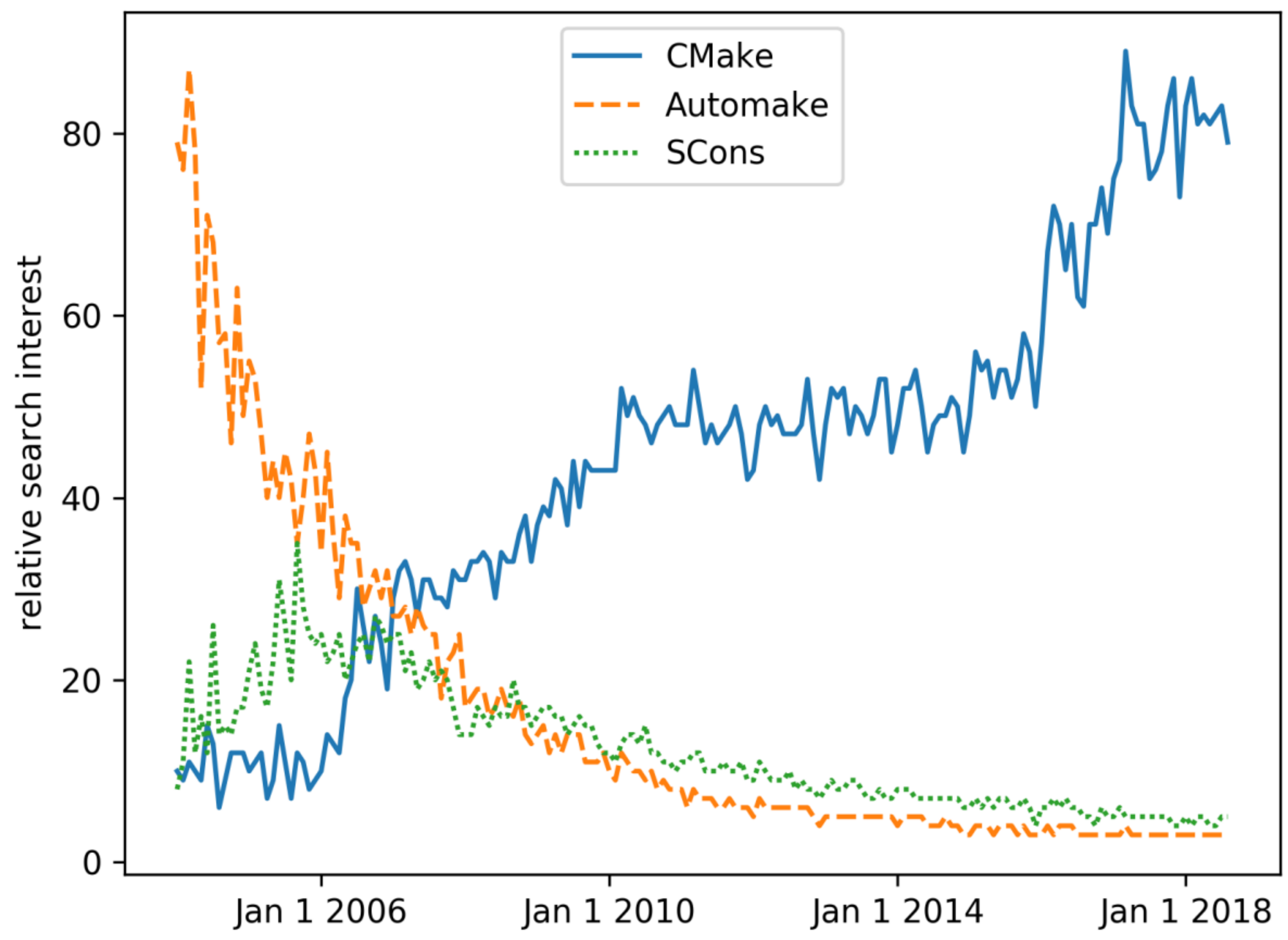
```
$ pkg-config --list-all | grep libpng
libpng                         libpng - Loads and saves PNG files

$ pkg-config --cflags libpng
-I/usr/include/libpng12

$ pkg-config --libs libpng
-lpng12

$ pkg-config --cflags --libs libpng
-I/usr/include/libpng12 -lpng12
```

autotools


autoscan
configure.ac
aclocal
autoheader
aclocal.m4
config.h.in
Makefile.am
autoconf
automake
configure
Makefile.in
config.status
config.h
Makefile
make
executable
input file
influences
process
creates
output file

Autotools Mythbuster

Основы сборки проектов при помощи Autotools


CMake
Automake
SCons
relative search interest
Jan 1 2006
Jan 1 2010
Jan 1 2014
Jan 1 2018