Cygwin API Reference

Cygwin API Reference

Table of Contents

List of Examples

Chapter 1. Compatibility

System interfaces compatible with the Single Unix Specification, Version 4:

Note that the core of the Single Unix Specification, Version 4 is also IEEE Std 1003.1-2008 (POSIX.1-2008).

```
FD_CLR
FD_ISSET
FD_SET
FD_ZERO
_Exit
_exit
_longjmp
_setjmp
_tolower
_toupper
a64l
abort
abs
accept
access
acos
acosf
acosh
acoshf
alarm
alphasort
asctime
asctime_r
asin
asinf
asinh
asinhf
atan
atan2
atan2f
atanf
atanh
atanhf
atexit
atof
atoff
atoi
atol
atoll
basename
bind
bsearch
btowc
cabs
cabsf
cacos
cacosf
cacosh
cacoshf
calloc
carg
cargf
casin
```

```
casinf
casinh
casinhf
casinhl
catan
catanf
catanh
catanhf
catclose    (available in external "catgets" library)
catgets     (available in external "catgets" library)
catopen     (available in external "catgets" library)
cbrt
cbrtf
ccos
ccosf
ccosh
ccoshf
ceil
ceilf
cexp
cexpf
cfgetispeed
cfgetospeed
cfsetispeed
cfsetospeed
chdir
chmod
chown
cimag
cimagf
clearerr
clock
clock_getcpuclockid
clock_getres
clock_gettime
clock_nanosleep  (see chapter "Implementation Notes")
clock_settime  (see chapter "Implementation Notes")
clog
clogf
close
closedir
closelog
confstr
conj
conjf
connect
copysign
copysignf
cos
cosf
cosh
coshf
cpow
cpowf
cproj
cprojf
creal
crealf
creat
crypt   (available in external "crypt" library)
csin
csinf
csinh
```

```
csinhf
csqrt
csqrtf
ctan
ctanf
ctanh
ctanhf
ctermid
ctime
ctime_r
daylight
dbm_clearerr  (available in external "libgdbm" library)
dbm_close   (available in external "libgdbm" library)
dbm_delete   (available in external "libgdbm" library)
dbm_error   (available in external "libgdbm" library)
dbm_fetch   (available in external "libgdbm" library)
dbm_firstkey  (available in external "libgdbm" library)
dbm_nextkey   (available in external "libgdbm" library)
dbm_open   (available in external "libgdbm" library)
dbm_store   (available in external "libgdbm" library)
difftime
dirfd
dirname
div
dlclose
dlerror
dlopen
dlsym
dprintf
drand48
dup
dup2
encrypt   (available in external "crypt" library)
endgrent
endhostent
endprotoent
endpwent
endservent
endutxent
environ
erand48
erf
erfc
erfcf
erff
errno
execl
execle
execlp
execv
execve
execvp
exit
exp
exp2
exp2f
expf
expm1
expm1f
fabs
fabsf
faccessat
fchdir
```

```
fchmod
fchmodat
fchown
fchownat
fclose
fcntl   (see chapter "Implementation Notes")
fdatasync
fdim
fdimf
fdopen
fdopendir
feclearexcept
fegetenv
fegetexceptflag
fegetround
feholdexcept
feof
feraiseexcept
ferror
fesetenv
fesetexceptflag
fesetround
fetestexcept
feupdateenv
fexecve
fflush
ffs
fgetc
fgetpos
fgets
fgetwc
fgetws
fileno
flockfile
floor
floorf
fma
fmaf
fmax
fmaxf
fmemopen
fmin
fminf
fmod
fmodf
fnmatch
fopen
fork
fpathconf
fpclassify   (see chapter "Implementation Notes")
fprintf
fputc
fputs
fputwc
fputws
fread
free
freeaddrinfo
freopen
frexp
frexpf
fscanf
fseek
```

```
fseeko
fsetpos
fstat
fstatat
fstatvfs
fsync
ftell
ftello
ftok
ftruncate
ftrylockfile
ftw
funlockfile
futimens
fwide
fwprintf
fwrite
fwscanf
gai_strerror
getaddrinfo
getc
getc_unlocked
getchar
getchar_unlocked
getcwd
getdelim
getdomainname
getegid
getenv
geteuid
getgid
getgrent
getgrgid
getgrgid_r
getgrnam
getgrnam_r
getgroups
gethostid
gethostname
getitimer   (see chapter "Implementation Notes")
getline
getlogin
getlogin_r
getnameinfo
getopt
getpeername
getpgid
getpgrp
getpid
getppid
getpriority
getprotobyname
getprotobynumber
getprotoent
getpwent
getpwnam
getpwnam_r
getpwuid
getpwuid_r
getrlimit
getrusage
gets
getservbyname
```

```
getservbyport
getservent
getsid
getsockname
getsockopt
getsubopt
gettimeofday
getuid
getutxent
getutxid
getutxline
getwc
getwchar
glob
globfree
gmtime
gmtime_r
grantpt
hcreate
hdestroy
hsearch
htonl
htons
hypot
hypotf
iconv   (available in external "libiconv" library)
iconv_close   (available in external "libiconv" library)
iconv_open   (available in external "libiconv" library)
if_freenameindex
if_indextoname
if_nameindex
if_nametoindex
ilogb
ilogbf
imaxabs
imaxdiv
inet_addr
inet_ntoa
inet_ntop
inet_pton
initstate
insque
ioctl
isalnum
isalpha
isascii
isatty
isblank
iscntrl
isdigit
isfinite   (see chapter "Implementation Notes")
isgraph
isgreater   (see chapter "Implementation Notes")
isgreaterequal  (see chapter "Implementation Notes")
isinf   (see chapter "Implementation Notes")
isless
islessequal   (see chapter "Implementation Notes")
islessgreater  (see chapter "Implementation Notes")
islower
isnan   (see chapter "Implementation Notes")
isnormal   (see chapter "Implementation Notes")
isprint
ispunct
```

```
isspace
isunordered   (see chapter "Implementation Notes")
isupper
iswalnum
iswalpha
iswblank
iswcntrl
iswctype
iswdigit
iswgraph
iswlower
iswprint
iswpunct
iswspace
iswupper
iswxdigit
isxdigit
j0
j1
jn
jrand48
kill
killpg
l64a
labs
lchown
lcong48
ldexp
ldexpf
ldiv
lfind
lgamma
lgammaf
link
linkat
listen
llabs
lldiv
llrint
llrintf
llrintl
llround
llroundf
localeconv
localtime
localtime_r
lockf   (see chapter "Implementation Notes")
log
log10
log10f
log1p
log1pf
log2
log2f
logb
logbf
logf
longjmp
lrand48
lrint
lrintf
lrintl
lround
```

```
lroundf
lsearch
lseek
lstat
malloc
mblen
mbrlen
mbrtowc
mbsinit
mbsnrtowcs
mbsrtowcs
mbstowcs
mbtowc
memccpy
memchr
memcmp
memcpy
memmove
memset
mkdir
mkdirat
mkdtemp
mkfifo
mkfifoat
mknod
mknodat
mkstemp
mktime
mlock
mmap
modf
modff
mprotect
mq_close
mq_getattr
mq_notify
mq_open
mq_receive
mq_send
mq_setattr
mq_timedreceive
mq_timedsend
mq_unlink
mrand48
msgctl   (see chapter "Implementation Notes")
msgget   (see chapter "Implementation Notes")
msgrcv   (see chapter "Implementation Notes")
msgsnd   (see chapter "Implementation Notes")
msync
munlock
munmap
nan
nanf
nanosleep
nearbyint
nearbyintf
nextafter
nextafterf
nftw
nice
nl_langinfo
nrand48
ntohl
```

```
ntohs
open
open_memstream
open_wmemstream
openat
opendir
openlog
optarg
opterr
optind
optopt
pathconf
pause
pclose
perror
pipe
poll
popen
posix_fadvise
posix_fallocate
posix_madvise
posix_memalign
posix_openpt
posix_spawn
posix_spawnattr_destroy
posix_spawnattr_init
posix_spawnattr_getflags
posix_spawnattr_getpgroup
posix_spawnattr_getschedparam
posix_spawnattr_getschedpolicy
posix_spawnattr_getsigdefault
posix_spawnattr_getsigmask
posix_spawnattr_setflags
posix_spawnattr_setpgroup
posix_spawnattr_setschedparam
posix_spawnattr_setschedpolicy
posix_spawnattr_setsigdefault
posix_spawnattr_setsigmask
posix_spawnp
posix_spawn_file_actions_destroy
posix_spawn_file_actions_init
posix_spawn_file_actions_addclose
posix_spawn_file_actions_adddup2
posix_spawn_file_actions_addopen
pow
powf
pread
printf
pselect
psiginfo
psignal
pthread_atfork
pthread_attr_destroy
pthread_attr_getdetachstate
pthread_attr_getguardsize
pthread_attr_getinheritsched
pthread_attr_getschedparam
pthread_attr_getschedpolicy
pthread_attr_getscope
pthread_attr_getstack
pthread_attr_getstacksize
pthread_attr_init
pthread_attr_setdetachstate
```

```
pthread_attr_setguardsize
pthread_attr_setinheritsched
pthread_attr_setschedparam
pthread_attr_setschedpolicy
pthread_attr_setscope
pthread_attr_setstack
pthread_attr_setstacksize
pthread_cancel
pthread_cond_broadcast
pthread_cond_destroy
pthread_cond_init
pthread_cond_signal
pthread_cond_timedwait
pthread_cond_wait
pthread_condattr_destroy
pthread_condattr_getclock
pthread_condattr_getpshared
pthread_condattr_init
pthread_condattr_setclock
pthread_condattr_setpshared
pthread_create
pthread_detach
pthread_equal
pthread_exit
pthread_getconcurrency
pthread_getcpuclockid
pthread_getschedparam
pthread_getspecific
pthread_join
pthread_key_create
pthread_key_delete
pthread_kill
pthread_mutex_destroy
pthread_mutex_getprioceiling
pthread_mutex_init
pthread_mutex_lock
pthread_mutex_setprioceiling
pthread_mutex_trylock
pthread_mutex_unlock
pthread_mutexattr_destroy
pthread_mutexattr_getprioceiling
pthread_mutexattr_getprotocol
pthread_mutexattr_getpshared
pthread_mutexattr_gettype
pthread_mutexattr_init
pthread_mutexattr_setprioceiling
pthread_mutexattr_setprotocol
pthread_mutexattr_setpshared
pthread_mutexattr_settype
pthread_once
pthread_rwlock_destroy
pthread_rwlock_init
pthread_rwlock_rdlock
pthread_rwlock_tryrdlock
pthread_rwlock_trywrlock
pthread_rwlock_unlock
pthread_rwlock_wrlock
pthread_rwlockattr_destroy
pthread_rwlockattr_getpshared
pthread_rwlockattr_init
pthread_rwlockattr_setpshared
pthread_self
pthread_setcancelstate
```

```
pthread_setcanceltype
pthread_setconcurrency
pthread_setschedparam
pthread_setschedprio
pthread_setspecific
pthread_sigmask
pthread_spin_destroy
pthread_spin_init
pthread_spin_lock
pthread_spin_trylock
pthread_spin_unlock
pthread_testcancel
ptsname
putc
putc_unlocked
putchar
putchar_unlocked
putenv
puts
pututxline
putwc
putwchar
pwrite
qsort
raise
rand
rand_r
random
read
readdir
readdir_r
readlink
readlinkat
readv
realloc
realpath
recv
recvfrom
recvmsg
regcomp
regerror
regexec
regfree
remainder
remainderf
remove
remque
remquo
remquof
rename
renameat
rewind
rewinddir
rint
rintf
rintl
rmdir
round
roundf
scalbln
scalblnf
scalbn
scalbnf
```

```
scandir
scanf
sched_get_priority_max
sched_get_priority_min
sched_getparam
sched_getscheduler
sched_rr_get_interval
sched_setparam
sched_setscheduler
sched_yield
seed48
seekdir
select
sem_close
sem_destroy
sem_getvalue
sem_init
sem_open
sem_post
sem_timedwait
sem_trywait
sem_unlink
sem_wait
semctl   (see chapter "Implementation Notes")
semget   (see chapter "Implementation Notes")
semop   (see chapter "Implementation Notes")
send
sendmsg
sendto
setbuf
setegid
setenv
seteuid
setgid
setgrent
sethostent
setitimer   (see chapter "Implementation Notes")
setjmp
setkey   (available in external "crypt" library)
setlocale
setlogmask
setpgid
setpgrp
setpriority
setprotoent
setpwent
setregid
setreuid
setrlimit
setservent
setsid
setsockopt
setstate
setuid
setutxent
setvbuf
shm_open
shm_unlink
shmat   (see chapter "Implementation Notes")
shmctl   (see chapter "Implementation Notes")
shmdt   (see chapter "Implementation Notes")
shmget   (see chapter "Implementation Notes")
shutdown
```

```
sigaction
sigaddset
sigdelset
sigemptyset
sigfillset
sighold
sigignore
siginterrupt
sigismember
siglongjmp
signal
signbit   (see chapter "Implementation Notes")
signgam
sigpause
sigpending
sigprocmask
sigqueue
sigrelse
sigset
sigsetjmp
sigsuspend
sigwait
sigwaitinfo
sin
sinf
sinh
sinhf
sleep
snprintf
socket
socketpair
sprintf
sqrt
sqrtf
srand
srand48
srandom
sscanf
stat
statvfs
stderr
stdin
stdout
stpcpy
stpncpy
strcasecmp
strcat
strchr
strcmp
strcoll
strcpy
strcspn
strdup
strerror
strerror_r
strfmon
strftime
strlen
strncasecmp
strncat
strncmp
strncpy
strndup
```

```
strnlen
strpbrk
strptime
strrchr
strsignal
strspn
strstr
strtod
strtof
strtoimax
strtok
strtok_r
strtol
strtoll
strtoul
strtoull
strtoumax
strxfrm
swab
swprintf
swscanf
symlink
symlinkat
sync
sysconf
syslog
system
tan
tanf
tanh
tanhf
tcdrain
tcflow
tcflush
tcgetattr
tcgetpgrp
tcsendbreak
tcsetattr
tcsetpgrp
tdelete
telldir
tempnam
tfind
tgamma
tgammaf
time
timer_create  (see chapter "Implementation Notes")
timer_delete
timer_gettime
timer_settime
times
timezone
tmpfile
tmpnam
toascii
tolower
toupper
towctrans
towlower
towupper
trunc
truncate
truncf
```

```
tsearch
ttyname
ttyname_r
twalk
tzname
tzset
umask
uname
ungetc
ungetwc
unlink
unlinkat
unlockpt
unsetenv
utime
utimensat
utimes
va_arg
va_copy
va_end
va_start
vdprintf
vfprintf
vfscanf
vfwprintf
vfwscanf
vprintf
vscanf
vsnprintf
vsprintf
vsscanf
vswprintf
vswscanf
vwprintf
vwscanf
wait
waitpid
wcpcpy
wcpncpy
wcrtomb
wcscasecmp
wcscat
wcschr
wcscmp
wcscoll
wcscpy
wcscspn
wcsdup
wcsftime
wcslen
wcsncasecmp
wcsncat
wcsncmp
wcsncpy
wcsnlen
wcsnrtombs
wcspbrk
wcsrchr
wcsrtombs
wcsspn
wcsstr
wcstod
wcstof
```

```
wcstoimax
wcstok
wcstol
wcstoll
wcstombs
wcstoul
wcstoull
wcstoumax
wcswidth
wcsxfrm
wctob
wctomb
wctrans
wctype
wcwidth
wmemchr
wmemcmp
wmemcpy
wmemmove
wmemset
wordexp
wordfree
wprintf
write
writev
wscanf
y0
y1
yn
```

System interfaces compatible with BSD functions:

```
__b64_ntop
__b64_pton
arc4random
arc4random_addrandom
arc4random_buf
arc4random_stir
arc4random_uniform
bindresvport
bindresvport_sa
cfmakeraw
cfsetspeed
daemon
dn_comp
dn_expand
dn_skipname
drem
eaccess
endusershell
err
errx
finite
finitef
fiprintf
flock   (see chapter "Implementation Notes")
forkpty
fpurge
freeifaddrs
fstatfs
fts_children
fts_close
fts_get_clientptr
fts_get_stream
fts_open
fts_read
fts_set
fts_set_clientptr
funopen
futimes
gamma
gamma_r
gammaf
gammaf_r
getdtablesize
getgrouplist
getifaddrs
getpagesize
getpeereid
getprogname
getusershell
herror
hstrerror
inet_aton
inet_makeaddr
inet_netof
inet_network
initgroups
```

```
iruserok
iruserok_sa
login
login_tty
logout
logwtmp
madvise
mkstemps
openpty
rcmd
rcmd_af
reallocf
res_close
res_init
res_mkquery
res_nclose
res_ninit
res_nmkquery
res_nquery
res_nquerydomain
res_nsearch
res_nsend
res_query
res_querydomain
res_search
res_send
revoke
rexec
rresvport
rresvport_af
ruserok
sbrk
setbuffer
setgroups
setlinebuf
setpassent
setprogname
settimeofday
setusershell
statfs
strcasestr
strlcat
strlcpy
strsep
updwtmp
valloc
verr
verrx
vhangup   (see chapter "Implementation Notes")
vsyslog
vwarn
vwarnx
wait3
wait4
warn
warnx
wcslcat
wcslcpy
```

System interfaces compatible with GNU or Linux extensions:

```
accept4
argz_add
argz_add_sep
argz_append
argz_count
argz_create
argz_create_sep
argz_delete
argz_extract
argz_insert
argz_next
argz_replace
argz_stringify
asnprintf
asprintf
asprintf_r
canonicalize_file_name
dremf
dup3
envz_add
envz_entry
envz_get
envz_merge
envz_remove
envz_strip
error
error_at_line
euidaccess
execvpe
exp10
exp10f
fcloseall
fcloseall_r
fegetprec
fesetprec
feenableexcept
fedisableexcept
fegetexcept
ffsl
ffsll
fgetxattr
flistxattr
fopencookie
fremovexattr
fsetxattr
get_avphys_pages
get_current_dir_name
get_phys_pages
get_nprocs
get_nprocs_conf
getmntent_r
getopt_long
getopt_long_only
getpt
getxattr
lgetxattr
```

```
listxattr
llistxattr
lremovexattr
lsetxattr
memmem
mempcpy
memrchr
mkostemp
mkostemps
pipe2
pow10
pow10f
ppoll
pthread_getattr_np
pthread_sigqueue
ptsname_r
quotactl
rawmemchr
removexattr
scandirat
setxattr
strchrnul
sysinfo
tdestroy
timegm
timelocal
updwtmpx
utmpxname
vasnprintf
vasprintf
vasprintf_r
```

System interfaces compatible with Solaris or SunOS functions:

```
__fpurge
acl
aclcheck
aclfrommode
aclfrompbits
aclfromtext
aclsort
acltomode
acltopbits
acltotext
endmntent
facl
futimesat
getmntent
memalign
setmntent
xdr_array
xdr_bool
xdr_bytes
xdr_char
xdr_double
xdr_enum
xdr_float
xdr_free
xdr_hyper
xdr_int
xdr_int16_t
xdr_int32_t
xdr_int64_t
xdr_int8_t
xdr_long
xdr_longlong_t
xdr_netobj
xdr_opaque
xdr_pointer
xdr_reference
xdr_short
xdr_sizeof
xdr_string
xdr_u_char
xdr_u_hyper
xdr_u_int
xdr_u_int16_t
xdr_u_int32_t
xdr_u_int64_t
xdr_u_int8_t
xdr_u_long
xdr_u_longlong_t
xdr_u_short
xdr_uint16_t
xdr_uint32_t
xdr_uint64_t
xdr_uint8_t
xdr_union
xdr_vector
xdr_void
```

```
xdr_wrapstring
xdrmem_create
xdrrec_create
xdrrec_endofrecord
xdrrec_eof
xdrrec_skiprecord
__xdrrec_getrec
__xdrrec_setnonblock
xdrstdio_create
```

Other UNIX system interfaces, deprecated or not in POSIX.1-2008:

```
bcmp   (POSIX.1-2001, SUSv3)
bcopy   (SUSv3)
bzero   (SUSv3)
chroot   (SUSv2) (see chapter "Implementation Notes")
clock_setres  (QNX, VxWorks) (see chapter "Implementation Notes")
cuserid   (POSIX.1-1988, SUSv2)
ecvt   (SUSv3)
endutent   (XPG2)
fcvt   (SUSv3)
ftime   (SUSv3)
gcvt   (SUSv3)
gethostbyaddr  (SUSv3)
gethostbyname  (SUSv3)
gethostbyname2  (first defined in BIND 4.9.4)
getpass   (SUSv2)
getutent   (XPG2)
getutid   (XPG2)
getutline   (XPG2)
getw   (SVID)
getwd   (SUSv3)
h_errno   (SUSv3)
index   (SUSv3)
mallinfo   (SVID)
mallopt   (SVID)
mktemp   (SUSv3)
on_exit   (SunOS)
pthread_attr_getstackaddr (SUSv3)
pthread_attr_setstackaddr (SUSv3)
pthread_continue  (XPG2)
pthread_getsequence_np (Tru64)
pthread_suspend  (XPG2)
pthread_yield  (POSIX.1c drafts)
pututline   (XPG2)
putw   (SVID)
rindex   (SUSv3)
scalb   (SUSv3)
setutent   (XPG2)
stime   (SVID)
sys_errlist   (BSD)
sys_nerr   (BSD)
sys_siglist   (BSD)
ttyslot   (SUSv2)
ualarm   (SUSv3)
usleep   (SUSv3)
utmpname   (XPG2)
vfork   (SUSv3) (see chapter "Implementation Notes")
```

NOT implemented system interfaces from the Single Unix Specification, Volume 4:

```
acoshl
acosl
aio_cancel
aio_error
aio_fsync
aio_read
aio_return
aio_suspend
aio_write
asinhl
asinl
atan2l
atanhl
atanl
cabsl
cacoshl
cacosl
cargl
casinl
catanhl
catanl
cbrtl
ccoshl
ccosl
ceill
cexpl
cimagl
clogl
conjl
copysignl
coshl
cosl
cpowl
cprojl
creall
csinhl
csinl
csqrtl
ctanhl
ctanl
duplocale
endnetent
erfcl
erfl
exp2l
expl
expm1l
fabsl
fattach
fdiml
floorl
fmal
fmaxl
fminl
fmodl
fmtmsg
```

```
freelocale
frexpl
getdate
getdate_err
gethostent
getmsg
getnetbyaddr
getnetbyname
getnetent
getpmsg
hypotl
ilogbl
isalnum_l
isalpha_l
isastream
isblank_l
iscntrl_l
isdigit_l
isgraph_l
islower_l
isprint_l
ispunct_l
isspace_l
isupper_l
iswalnum_l
iswalpha_l
iswblank_l
iswcntrl_l
iswdigit_l
iswgraph_l
iswlower_l
iswprint_l
iswpunct_l
iswspace_l
iswupper_l
iswxdigit_l
isxdigit_l
ldexpl
lgammal
lio_listio
llroundl
log10l
log1pl
log2l
logbl
logl
lroundl
mlockall
modfl
munlockall
nanl
nearbyintl
newlocale
nextafterl
nexttoward
nexttowardf
nexttowardl
posix_mem_offset
posix_trace[...]
posix_typed_[...]
powl
pthread_barrier[...]
pthread_mutexattr_getrobust
```

```
pthread_mutexattr_setrobust
pthread_mutex_consistent
pthread_mutex_timedlock
pthread_rwlock_timedrdlock
pthread_rwlock_timedwrlock
putmsg
reminderl
remquol
roundl
scalblnl
scalbnl
setnetent
sigaltstack
sigtimedwait
sinhl
sinl
sockatmark
sqrtl
strcasecmp_l
strcoll_l
strfmon_l
strncasecmp_l
strtold
strxfrm_l
tanhl
tanl
tcgetsid
tgammal
timer_getoverrun
tolower_l
toupper_l
towctrans_l
truncl
ulimit
uselocale
waitid
wcscasecmp_l
wcsncasecmp_l
wcstold
wcsxfrm_l
wctrans_l
wctype_l
```

Implementation Notes

`chroot` only emulates a chroot function call by keeping track of the current root and accomodating this in the file related function calls. A real chroot functionality is not supported by Windows however.

`clock_nanosleep` currently supports only CLOCK_REALTIME and CLOCK_MONOTONIC. `clock_setres`, `clock_settime`, and `timer_create` currently support only CLOCK_REALTIME.

POSIX file locks via `fcntl` or `lockf`, as well as BSD `flock` locks are advisory locks. They don't interact with Windows mandatory locks, nor do POSIX fcntl locks interfere with BSD flock locks or vice versa.

BSD file locks created via `flock` are only propagated to the direct parent process, not to grand parents or sibling processes. The locks are only valid in the creating process, its parent process, and subsequently started child processes sharing the same file descriptor.

In very rare circumstances an application would want to use Windows mandatory locks to interact with non-Cygwin Windows processes accessing the same file (databases, etc). For these purposes, the entire locking mechanism (fcntl/flock/lockf) can be switched to Windows mandatory locks on a per-descriptor/per-process basis. For this purpose, use the call

```
fcntl (fd, F_LCK_MANDATORY, 1);
```

After that, all file locks on this descriptor will follow Windows mandatory record locking semantics: Locks are per-descriptor/per-process; locks are not propagated to child processes, not even via `execve`; no atomic replacement of read locks with write locks and vice versa on the same descriptor; locks have to be unlocked exactly as they have been locked.

`fpclassify`, `isfinite`, `isgreater`, `isgreaterequal`, `isinf`, `isless`, `islessequal`, `islessgreater`, `isnan`, `isnormal`, `isunordered`, and `signbit` only support float and double arguments, not long double arguments.

`getitimer` and `setitimer` only support ITIMER_REAL for now.

`link` will fail on FAT, FAT32, and other filesystems not supporting hardlinks, just as on Linux.

`lseek` only works properly on files opened in binary mode. On files opened in textmode (via mount mode or explicit open flag) its positioning is potentially unreliable.

`setuid` is only safe against reverting the user switch after a call to one of the exec(2) functions took place. Windows doesn't support a non-revertable user switch within the context of Win32 processes.

`vfork` just calls `fork`.

`vhangup` and `revoke` always return -1 and set errno to ENOSYS. `grantpt` and `unlockpt` always just return 0.

The XSI IPC functions `semctl`, `semget`, `semop`, `shmat`, `shmctl`, `shmdt`, `shmget`, `msgctl`, `msgget`, `msgrcv` and `msgsnd` are only available when cygserver is running.

The Linux-specific function `quotactl` only implements what works on Windows: Windows only supports user block quotas on NTFS, no group quotas, no inode quotas, no time constraints.

Chapter 2. Cygwin Functions

Path conversion functions

These functions are specific to Cygwin itself, and probably won't be found anywhere else.

cygwin_conv_path

```
extern "C" ssize_t cygwin_conv_path(what, from, to, size);

cygwin_conv_path_t what;
const void * from;
void * to;
size_t size;
```

Use this function to convert POSIX paths in *from* to Win32 paths in *to* or, vice versa, Win32 paths in *from* to POSIX paths in *to*. *what* defines the direction of this conversion and can be any of the below values.

```
  CCP_POSIX_TO_WIN_A      /* from is char *posix, to is char *win32       */
  CCP_POSIX_TO_WIN_W,     /* from is char *posix, to is wchar_t *win32    */
  CCP_WIN_A_TO_POSIX,     /* from is char *win32, to is char *posix       */
  CCP_WIN_W_TO_POSIX,     /* from is wchar_t *win32, to is char *posix    */
```

You can additionally or the following values to *what*, to define whether you want the resulting path in *to* to be absolute or if you want to keep relative paths in relative notation. Creating absolute paths is the default.

```
  CCP_ABSOLUTE = 0,       /* Request absolute path (default).             */
  CCP_RELATIVE = 0x100    /* Request to keep path relative.               */
```

size is the size of the buffer pointed to by *to* in bytes. If *size* is 0, cygwin_conv_path just returns the required buffer size in bytes. Otherwise, it returns 0 on success, or -1 on error and errno is set to one of the below values.

```
    EINVAL        what has an invalid value or from is NULL.
    EFAULT        from or to point into nirvana.
    ENAMETOOLONG  the resulting path is longer than 32K, or, in case
                  of what == CCP_POSIX_TO_WIN_A, longer than MAX_PATH.
    ENOSPC        size is less than required for the conversion.
```

Example 2.1. Example use of cygwin_conv_path

```
#include <sys/cygwin.h>

/* Conversion from incoming Win32 path given as wchar_t *win32 to POSIX path.
   If incoming path is a relative path, stick to it.  First ask how big
   the output buffer has to be and allocate space dynamically. */
ssize_t size;
char *posix;
size = cygwin_conv_path (CCP_WIN_W_TO_POSIX | CCP_RELATIVE, win32, NULL, 0);
if (size < 0)
  perror ("cygwin_conv_path");
else
  {
    posix = (char *) malloc (size);
```

```
  if (cygwin_conv_path (CCP_WIN_W_TO_POSIX | CCP_RELATIVE, win32,
                        posix, size))
    perror ("cygwin_conv_path");
}
```

cygwin_conv_path_list

```
extern "C" ssize_t cygwin_conv_path_list(what, from, to, size);

cygwin_conv_path_t what;
const void * from;
void * to;
size_t size;
```

This is the same as `cygwin_conv_path`, but the input is treated as a path list in $PATH or %PATH% notation.

If *`what`* is CCP_POSIX_TO_WIN_A or CCP_POSIX_TO_WIN_W, given a POSIX $PATH-style string (i.e. /foo:/bar) convert it to the equivalent Win32 %PATH%-style string (i.e. d:\;e:\bar).

If *`what`* is CCP_WIN_A_TO_POSIX or CCP_WIN_W_TO_POSIX, given a Win32 %PATH%-style string (i.e. d:\;e:\bar) convert it to the equivalent POSIX $PATH-style string (i.e. /foo:/bar).

`size` is the size of the buffer pointed to by *`to`* in bytes.

See also cygwin_conv_path

cygwin_create_path

```
extern "C" void * cygwin_create_path(what, from);

cygwin_conv_path_t what;
const void * from;
```

This is equivalent to the `cygwin_conv_path`, except that `cygwin_create_path` does not take a buffer pointer for the result of the conversion as input. Rather it allocates the buffer itself using `malloc`(3) and returns a pointer to this buffer. In case of error it returns NULL and sets errno to one of the values defined for `cygwin_conv_path`. Additionally errno can be set to the below value.

```
ENOMEM        Insufficient memory was available.
```

When you don't need the returned buffer anymore, use `free`(3) to deallocate it.

See also cygwin_conv_path

cygwin_posix_path_list_p

```
extern "C" int cygwin_posix_path_list_p(path);

const char *path;
```

This function tells you if the supplied *`path`* is a POSIX-style path (i.e. posix names, forward slashes, colon delimiters) or a Win32-style path (drive letters, reverse slashes, semicolon delimiters. The return value is true if the path is a POSIX path. Note that "_p" means "predicate", a lisp term meaning that the function tells you something about the parameter.

cygwin_split_path

```
extern "C" void cygwin_split_path (path, dir, file);

const char * path;
char * dir;
char * file;
```

Split a path into the directory and the file portions. Both *dir* and *file* are expected to point to buffers of sufficient size.

Example 2.2. Example use of cygwin_split_path

```
char dir[200], file[100];
cygwin_split_path("c:/foo/bar.c", dir, file);
printf("dir=%s, file=%s\n", dir, file);
```

Helper functions to change user context

cygwin_logon_user

```
extern "C" HANDLE cygwin_logon_user(passwd_entry, password);

const struct passwd *passwd_entry;
const char *password;
```

Given a pointer to a passwd entry of a user and a cleartext password, returns a HANDLE to an impersonation token for this user which can be used in a subsequent call to `cygwin_set_impersonation_token` to impersonate that user. This function can only be called from a process which has the required NT user rights to perform a logon.

See also the chapter Switching the user context [../cygwin-ug-net/ntsec.html#ntsec-setuid-overview] in the Cygwin User's guide.

See also cygwin_set_impersonation_token

cygwin_set_impersonation_token

```
extern "C" void cygwin_set_impersonation_token(token);

const HANDLE token;
```

Use this function to enable the token given as parameter as impersonation token for the next call to `setuid` or `seteuid`. Use `cygwin_set_impersonation_token` together with `cygwin_logon_user` to impersonate users using password authentication.

See also the chapter Switching the user context [../cygwin-ug-net/ntsec.html#ntsec-setuid-overview] in the Cygwin User's guide.

See also cygwin_logon_user

Miscellaneous functions

cygwin_attach_handle_to_fd

```
extern "C" int cygwin_attach_handle_to_fd(name, fd, handle, bin,
access);

char *name;
int fd;
HANDLE handle;
int bin;
int access;
```

This function can be used to turn a Win32 "handle" into a posix-style file handle. *fd* may be -1 to make cygwin allocate a handle; the actual handle is returned in all cases.

Even after using function, Cygwin doesn't know anything about the underlying file or device. It just tries to supply the typical file functions on a "best-effort" basis. Use with care. Don't expect too much.

cygwin_internal

```
extern "C" DWORD cygwin_internal(t, ...);

cygwin_getinfo_types t;
...;
```

This function gives you access to various internal data and functions. It takes two arguments. The first argument is a type from the 'cygwin_getinfo_types' enum. The second is an optional pointer.

Stay away unless you know what you're doing.

cygwin_stackdump

```
extern "C" void cygwin_stackdump();
```

Outputs a stackdump to stderr from the called location.