

Cygwin API Reference

Copyright © Cygwin authors

Permission is granted to make and distribute verbatim copies of this documentation provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this documentation under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this documentation into another language, under the above conditions for modified versions, except that this permission notice may be stated in a translation approved by the Free Software Foundation.

Contents

1	Compatibility	1
1.1	System interfaces compatible with the Single Unix Specification, Version 7:	1
1.2	System interfaces compatible with BSD functions:	19
1.3	System interfaces compatible with GNU or Linux extensions:	21
1.4	System interfaces compatible with Solaris or SunOS functions:	24
1.5	System interfaces not in POSIX but compatible with ISO C requirements:	25
1.6	Other UNIX system interfaces, not in POSIX.1-2008 or deprecated:	26
1.7	NOT implemented system interfaces from the Single Unix Specification, Volume 7:	27
1.8	Implementation Notes	27
2	Cygwin Functions	30
2.1	Path conversion functions	30
2.1.1	cygwin_conv_path	30
2.1.2	cygwin_conv_path_list	31
2.1.3	cygwin_create_path	31
2.1.4	cygwin_posix_path_list_p	32
2.1.5	cygwin_split_path	32
2.2	Helper functions to change user context	33
2.2.1	cygwin_logon_user	33
2.2.2	cygwin_set_impersonation_token	33
2.3	Miscellaneous functions	34
2.3.1	cygwin_attach_handle_to_fd	34
2.3.2	cygwin_internal	34
2.3.3	cygwin_stackdump	34

Abstract

Cygwin API Reference

Chapter 1

Compatibility

1.1 System interfaces compatible with the Single Unix Specification, Version 7:

Note that the core of the Single Unix Specification, Version 7 is also IEEE Std 1003.1-2017 (POSIX.1-2017).

```
FD_CLR
FD_ISSET
FD_SET
FD_ZERO
_Exit
_exit
_longjmp
_setjmp
_tolower
_toupper
a64l
abort
abs
accept
access
acos
acosf
acosh
acoshf
acoshl
acosl
aio_cancel
aio_error
aio_fsync
aio_read
aio_return
aio_suspend
aio_write
alarm
alphasort
asctime
asctime_r
asin
asinf
asinh
asinhf
asinhf
asinhf
asinhf
atan
```

```
atan2
atan2f
atan2l
atanf
atanh
atanhf
atanhl
atanl
atexit
atof
atoff
atoi
atol
atoll
basename      (see chapter "Implementation Notes")
bind
bsearch
btowc
cabs
cabsf
cabsl
cacos
cacosf
cacosh
cacoshf
cacoshl
cacosl
calloc
carg
cargf
cargl
casin
casinf
casinh
casinhf
casinhl
casinl
catan
catanf
catanh
catanhf
catanhl
catanl
catclose
catgets
catopen
cbrt
cbrtf
cbrtl
ccos
ccosf
ccosh
ccoshf
ccoshl
ccosl
ceil
ceilf
ceill
cexp
cexpf
cexpl
cfgetispeed
```

```
cfgetospeed
cfsetispeed
cfsetospeed
chdir
chmod
chown
cimag
cimagf
cimagl
clearerr
clock
clock_getcpuclockid
clock_getres
clock_gettime
clock_nanosleep  (see chapter "Implementation Notes")
clock_settime    (see chapter "Implementation Notes")
clog
clogf
clogl
close
closedir
closelog
confstr
conj
conjf
conjl
connect
copysign
copysignf
copysignl
cos
cosf
cosh
coshf
coshl
cosl
cpow
cpowf
cpowl
cproj
cprojf
cprojl
creal
crealf
creall
creat
crypt             (available in external "crypt" library)
csin
csinf
csinh
csinhf
csinhl
csinl
csqrt
csqrtf
csqrtl
ctan
ctanf
ctanh
ctanhf
ctanhl
ctanl
```

```
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
dlopen
dlsym
dprintf
drand48
dup
dup2
duplocale
encrypt           (available in external "crypt" library)
endgrent
endhostent
endprotoent
endpwent
endservent
endutxent
environ
erand48
erf
erfc
erfcf
erfcl
erff
erfl
errno
execl
execle
execvp
execv
execve
execvp
exit
exp
exp2
exp2f
exp2l
expf
expl
expm1
expm1f
expm1l
fabs
fabsf
fabsl
faccessat
```



```
fchdir
fchmod
fchmodat
fchown
fchownat
fclose
fcntl      (see chapter "Implementation Notes")
fdatasync
fdim
fdimf
fdiml
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
floorl
fma
fmaf
fmal
fmax
fmaxf
fmaxl
fmemopen
fmin
fminf
fminl
fmod
fmodf
fmodl
fnmatch
fopen
fork
fpathconf
fpclassify (see chapter "Implementation Notes")
fprintf
putc
puts
putwc
putws
```

```
fread
free
freeaddrinfo
freelocale
freopen
frexp
frexpf
frexpl
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      (see chapter "Implementation Notes")
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
hypotl
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
ilogbl
imaxabs
imaxdiv
inet_addr
inet_ntoa
inet_ntop
inet_pton
initstate
insque
ioctl
isalnum
isalnum_l
isalpha
isalpha_l
isascii
isatty
```

```
isblank
isblank_l
iscntrl
iscntrl_l
isdigit
isdigit_l
isfinite      (see chapter "Implementation Notes")
isgraph
isgraph_l
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
islower_l
isnan         (see chapter "Implementation Notes")
isnormal      (see chapter "Implementation Notes")
isprint
isprint_l
ispunct
ispunct_l
isspace
isspace_l
isunordered   (see chapter "Implementation Notes")
isupper
isupper_l
iswalnum
iswalnum_l
iswalpha
iswalpha_l
iswblank
iswblank_l
iswcntrl
iswcntrl_l
iswctype
iswctype_l
iswdigit
iswdigit_l
iswgraph
iswgraph_l
iswlower
iswlower_l
iswprint
iswprint_l
iswpunct
iswpunct_l
iswspace
iswspace_l
iswupper
iswupper_l
iswxdigit
iswxdigit_l
isxdigit
isxdigit_l
j0
j1
jn
jrand48
kill
killpg
```

```
l64a
labs
lchown
lcong48
ldexp
ldexpf
ldexpl
ldiv
lfind
lgamma
lgammaf
lgammal
link
linkat
lio_listio
listen
llabs
lldiv
llrint
llrintf
llrintl
llround
llroundf
llroundl
localeconv
localtime
localtime_r
lockf      (see chapter "Implementation Notes")
log
log10
log10f
log10l
log1p
log1pf
log1pl
log2
log2f
log2l
logb
logbf
logbl
logf
logl
longjmp
lrand48
lrint
lrintf
lrintl
lround
lroundf
lroundl
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
modfl
mprotect
mq_close
mq_getattr
mq_notify
mq_open
mq_receive
mq_send
mq_setattr
mq_timedreceive
mq_timedsend
mq_unlink
rand48
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
nanl
nanosleep
nearbyint
nearbyintf
nearbyintl
newlocale
nextafter
nextafterf
nextafterl
nexttoward
nexttowardf
nexttowardl
nftw
nice        (see chapter "Implementation Notes")
nl_langinfo
nl_langinfo_l
rand48
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_spawn_file_actions_addclose
posix_spawn_file_actions_adddup2
posix_spawn_file_actions_addopen
posix_spawn_file_actions_destroy
posix_spawn_file_actions_init
posix_spawnattr_destroy
posix_spawnattr_getflags
posix_spawnattr_getpgroup
posix_spawnattr_getschedparam
posix_spawnattr_getschedpolicy
posix_spawnattr_getsigdefault
posix_spawnattr_getsigmask
posix_spawnattr_init
posix_spawnattr_setflags
posix_spawnattr_setpgroup
posix_spawnattr_setschedparam
posix_spawnattr_setschedpolicy
posix_spawnattr_setsigdefault
posix_spawnattr_setsigmask
posix_spawnnp
pow
powf
powl
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_barrier_destroy
pthread_barrier_init
pthread_barrier_wait
pthread_barrierattr_destroy
pthread_barrierattr_getpshared
pthread_barrierattr_init
pthread_barrierattr_setpshared
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_timedlock
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_timedrdlock
```



```
pthread_rwlock_timedwrlock
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
regex
regfree
remainder
remainderf
remainderl
remove
remque
remquo
remquof
remquol
```

```
rename
renameat
rewind
rewinddir
rint
rintf
rintl
rmdir
round
roundf
roundl
scalbln
scalblnf
scalblnl
scalbn
scalbnf
scalbnl
scandir
scanf
sched_get_priority_max
sched_get_priority_min
sched_getparam
sched_getscheduler
sched_rr_get_interval
sched_setparam      (see chapter "Implementation Notes")
sched_setscheduler  (see chapter "Implementation Notes")
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  (see chapter "Implementation Notes")
setprotoent
```

```
setpwent
setregid
setreuid
setrlimit      (see chapter "Implementation Notes")
setserverent
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
sigaltstack
sigdelset
sigemptyset
sigfillset
sighold
sigignore
siginterrupt
sigismember
siglongjmp
signal
signbit    (see chapter "Implementation Notes")
signgam
sigpause   (see chapter "Implementation Notes")
sigpending
sigprocmask
sigqueue
sigrelse
sigset
sigsetjmp
sigsuspend
sigtimedwait
sigwait
sigwaitinfo
sin
sinf
sinh
sinhf
sinhl
sinl
sleep
snprintf
socketatmark
socket
socketpair
sprintf
sqrt
sqrtf
sqrtl
srand
srand48
srandom
sscanf
```

```
stat
statvfs
stderr
stdin
stdout
stpcpy
stpncpy
strcasemp
strcasemp_l
strcat
strchr
strcmp
strcoll
strcoll_l
strcpy
strcspn
strdup
strerror
strerror_l
strerror_r      (see chapter "Implementation Notes")
strfmon
strfmon_l
strftime
strftime_l
strlen
strncasemp
strncasemp_l
strncat
strncmp
strncpy
strndup
strnlen
strpbrk
strptime
strrchr
strsignal
strspn
strstr
strtod
strtof
strtoimax
strtok
strtok_r
strtol
strtold
strtoll
strtoul
strtoull
strtoumax
strxfrm
strxfrm_l
swab
swprintf
swscanf
symlink
symlinkat
sync
sysconf
syslog
system
tan
tanf
```

```
tanh
tanhf
tanh1
tanl
tcdrain
tcflow
tcflush
tcgetattr
tcgetpgrp
tcgetsid
tcsendbreak
tcsetattr
tcsetpgrp
tdelete
telldir
tempnam
tfind
tgamma
tgammaf
tgammal
time
timer_create      (see chapter "Implementation Notes")
timer_delete
timer_getoverrun
timer_gettime
timer_settime
times
timezone
tmpfile
tmpnam
tolower
tolower_l
toupper
toupper_l
towctrans
towctrans_l
towlower
towlower_l
toupper
toupper_l
trunc
truncate
truncf
truncl
tsearch
ttyname
ttyname_r
twalk
tzname
tzset
umask
uname
ungetc
ungetwc
unlink
unlinkat
unlockpt
unsetenv
uselocale
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
wcpncpy
wcpncpy
wctomb
wcscasecmp
wcscasecmp_l
wscat
wcschr
wcscmp
wcscoll
wcscoll_l
wcscpy
wcscspn
wcsdup
wcsftime
wcslen
wcsncasecmp
wcsncasecmp_l
wcsncat
wcsncmp
wcsncpy
wcsnlen
wcsnrtombs
wcpbrk
wcpchr
wcpstombs
wcssp
wcstod
wcstof
wcstoimax
wcstok
wcstol
wcstold
wcstoll
wcstombs
wcstoul
wcstoull
wcstoumax
wcswidth
wcxfrm
wcxfrm_l
wctob
```

```
wctomb
wctrans
wctrans_l
wctype
wctype_l
wcwidth
wmemchr
wmemcmp
wmemcpy
wmemmove
wmemset
wordexp
wordfree
wprintf
write
writev
wscanf
y0
y1
yn
```

1.2 System interfaces compatible with BSD functions:

```
__b64_ntop
__b64_pton
arc4random
arc4random_addrandom
arc4random_buf
arc4random_stir
arc4random_uniform
bindresvport
bindresvport_sa
cfmakeraw
cfsetspeed
clearerr_unlocked
close_range
daemon
dn_comp
dn_expand
dn_skipname
drem
eaccess
endusershell
err
errx
explicit_bzero
feof_unlocked
ferror_unlocked
fflush_unlocked
fileno_unlocked
fgetc_unlocked
finite
finitef
finitel
fiprintf
flock      (see chapter "Implementation Notes")
fls
flsl
flsl1
```

```
forkpty
fpurge
fputc_unlocked
fread_unlocked
freeifaddrs
fstatfs
fts_children
fts_close
fts_get_clientptr
fts_get_stream
fts_open
fts_read
fts_set
fts_set_clientptr
funopen
futimes
fwrite_unlocked
gamma
gamma_r
gammaf
gammaf_r
getdtablesize
getgrouplist
getifaddrs
getloadavg
getpagesize
getpeereid
getprogname
getusershell
herror
hstrerror
inet_aton
inet_makeaddr
inet_netof
inet_network
initgroups
iruserok
iruserok_sa
issetugid
login
login_tty
logout
logwtmp
madvise
mkstemp
openpty
qsort_r      (see chapter "Implementation Notes")
rcmd
rcmd_af
reallocarray
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
rpmatch
rresvport
rresvport_af
ruserok
sbrk
setbuffer
setgroups
sethostname
setlinebuf
setpassent
setprogname
settimeofday
setusershell
statfs
strcasestr
strlcat
strlcpy
strsep
timingsafe_bcmp
timingsafe_memcmp
updwtmp
valloc
verr
verrx
vhangup      (see chapter "Implementation Notes")
vsyslog
vwarn
vwarnx
wait3
wait4
warn
warnx
wcslcat
wcslcpy
```

1.3 System interfaces compatible with GNU or Linux extensions:

```
__mempcpy
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
```

```
basename      (see chapter "Implementation Notes")
canonicalize_file_name
clearenv
clog10
clog10f
clog10l
close_range   (see chapter "Implementation Notes")
crypt_r       (available in external "crypt" library)
dladdr        (see chapter "Implementation Notes")
dremf
dup3
envz_add
envz_entry
envz_get
envz_merge
envz_remove
envz_strip
error
error_at_line
euidaccess
execvpe
exp10
exp10f
exp10l
fallocate     (see chapter "Implementation Notes")
fcloseall
fcloseall_r
fedisableexcept
feenableexcept
fegetexcept
ffsl
ffsll
fgets_unlocked
fgetwc_unlocked
fgetws_unlocked
fgetxattr
flistxattr
fopencookie
fputs_unlocked
fputwc_unlocked
fputws_unlocked
fremovexattr
fsetxattr
get_avphys_pages
get_current_dir_name
get_nprocs
get_nprocs_conf
get_phys_pages
getmntent_r
getopt_long
getopt_long_only
getpt
getwc_unlocked
getwchar_unlocked
getxattr
lgetxattr
listxattr
llistxattr
lremovexattr
lsetxattr
memmem
mempcpy
```

```
memchr
mkostemp
mkostemps
pipe2
posix_spawn_file_actions_addchdir_np
posix_spawn_file_actions_addfchdir_np
pow10
pow10f
pow10l
ppoll
pthread_cond_clockwait
pthread_getaffinity_np
pthread_getattr_np
pthread_getname_np
pthread_mutex_clocklock
pthread_rwlock_clockrdlock
pthread_rwlock_clockwrlock
pthread_setaffinity_np
pthread_setname_np
pthread_sigqueue
pthread_timedjoin_np
pthread_tryjoin_np
ptsname_r
putwc_unlocked
putwchar_unlocked
renameat2      (see chapter "Implementation Notes")
qsort_r        (see chapter "Implementation Notes")
quotactl
rawmemchr
removexattr
scandirat
sched_getaffinity
sched_getcpu
sched_setaffinity
secure_getenv
sem_clockwait
setxattr
signalfd
sincos
sincosf
sincosl
strchrnul
strptime_l
strtod_l
strtof_l
strtol_l
strtold_l
strtoll_l
strtoul_l
strtoull_l
strverscmp
sysinfo
tdestroy
timerfd_create
timerfd_gettime
timerfd_settime
timegm
timelocal
toascii_l
updwtmpx
utmpxname
vasnprintf
```

```
vasprintf
vasprintf_r
versionsort
wcsftime_l
wcstod_l
wcstof_l
wcstol_l
wcstold_l
wcstoll_l
wcstoul_l
wcstoull_l
wmempcpy
```

1.4 System interfaces compatible with Solaris or SunOS functions:

```
__fbuflen
__flbf
__fpending
__fpurge
__freadable
__freading
__fsetlocking
__fwritable
__fwriting
acl
aclcheck
aclfrommode
aclfrompbits
aclfromtext
aclsort
acltomode
acltopbits
acltotext
endmntent
facl
fegetprec
fesetprec
futimesat
getmntent
memalign
setmntent
sig2str
str2sig
xdr_array      (available in external "libtirpc" library)
xdr_bool       (available in external "libtirpc" library)
xdr_bytes      (available in external "libtirpc" library)
xdr_char       (available in external "libtirpc" library)
xdr_double     (available in external "libtirpc" library)
xdr_enum       (available in external "libtirpc" library)
xdr_float      (available in external "libtirpc" library)
xdr_free       (available in external "libtirpc" library)
xdr_hyper      (available in external "libtirpc" library)
xdr_int        (available in external "libtirpc" library)
xdr_int16_t    (available in external "libtirpc" library)
xdr_int32_t    (available in external "libtirpc" library)
xdr_int64_t    (available in external "libtirpc" library)
xdr_int8_t     (available in external "libtirpc" library)
xdr_long       (available in external "libtirpc" library)
xdr_longlong_t (available in external "libtirpc" library)
```

```

xdr_netobj      (available in external "libtirpc" library)
xdr_opaque      (available in external "libtirpc" library)
xdr_pointer     (available in external "libtirpc" library)
xdr_reference   (available in external "libtirpc" library)
xdr_short       (available in external "libtirpc" library)
xdr_sizeof      (available in external "libtirpc" library)
xdr_string      (available in external "libtirpc" library)
xdr_u_char      (available in external "libtirpc" library)
xdr_u_hyper     (available in external "libtirpc" library)
xdr_u_int       (available in external "libtirpc" library)
xdr_u_int16_t   (available in external "libtirpc" library)
xdr_u_int32_t   (available in external "libtirpc" library)
xdr_u_int64_t   (available in external "libtirpc" library)
xdr_u_int8_t    (available in external "libtirpc" library)
xdr_u_long      (available in external "libtirpc" library)
xdr_u_longlong_t (available in external "libtirpc" library)
xdr_u_short     (available in external "libtirpc" library)
xdr_uint16_t    (available in external "libtirpc" library)
xdr_uint32_t    (available in external "libtirpc" library)
xdr_uint64_t    (available in external "libtirpc" library)
xdr_uint8_t     (available in external "libtirpc" library)
xdr_union       (available in external "libtirpc" library)
xdr_vector      (available in external "libtirpc" library)
xdr_void        (available in external "libtirpc" library)
xdr_wrapstring  (available in external "libtirpc" library)
xdrmem_create   (available in external "libtirpc" library)
xdrrec_create   (available in external "libtirpc" library)
xdrrec_endofrecord (available in external "libtirpc" library)
xdrrec_eof      (available in external "libtirpc" library)
xdrrec_skiprecord (available in external "libtirpc" library)
__xdrrec_getrec  (available in external "libtirpc" library)
__xdrrec_setnonblock (available in external "libtirpc" library)
xdrstdio_create (available in external "libtirpc" library)

```

1.5 System interfaces not in POSIX but compatible with ISO C requirements:

```

aligned_alloc   (ISO C11)
at_quick_exit   (ISO C11)
c16rtomb        (ISO C11)
c32rtomb        (ISO C11)
c8rtomb         (ISO C23)
call_once       (ISO C11)
cnd_broadcast    (ISO C11)
cnd_destroy     (ISO C11)
cnd_init        (ISO C11)
cnd_signal       (ISO C11)
cnd_timedwait   (ISO C11)
cnd_wait        (ISO C11)
mbrtoc16        (ISO C11)
mbrtoc32        (ISO C11)
mbrtoc8         (ISO C23)
mtx_destroy     (ISO C11)
mtx_init        (ISO C11)
mtx_lock        (ISO C11)
mtx_timedlock   (ISO C11)
mtx_trylock     (ISO C11)
mtx_unlock      (ISO C11)
quick_exit      (ISO C11)
thrd_create     (ISO C11)

```

```
thrd_current    (ISO C11)
thrd_detach     (ISO C11)
thrd_equal      (ISO C11)
thrd_exit       (ISO C11)
thrd_join       (ISO C11)
thrd_sleep      (ISO C11)
thrd_yield      (ISO C11)
tss_create      (ISO C11)
tss_delete      (ISO C11)
tss_get         (ISO C11)
tss_set         (ISO C11)
```

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

```
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)
getcontext      (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)
makecontext     (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)
setcontext      (SUSv3)
setutent        (XPG2)
stime           (SVID)
swapcontext     (SUSv3)
sys_errlist     (BSD)
sys_nerr        (BSD)
sys_siglist     (BSD)
```

```

toascii      (SUSv3)
ttyslot      (SUSv2)
ualarm       (SUSv3)
usleep       (SUSv3)
utmpname     (XPG2)
vfork        (SUSv3) (see chapter "Implementation Notes")

```

1.7 NOT implemented system interfaces from the Single Unix Specification, Volume 7:

```

endnetent
fattach
fmtmsg
getdate
getdate_err
gethostent
getmsg
getnetbyaddr
getnetbyname
getnetent
getpmsg
isastream
mlockall
munlockall
posix_mem_offset
posix_trace[...]
posix_typed_ [...]
pthread_mutexattr_getrobust
pthread_mutexattr_setrobust
pthread_mutex_consistent
putmsg
setnetent
ulimit
waitid

```

1.8 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`.

`close_range` does not support the Linux-specific flag `CLOSE_RANGE_UNSHARE`.

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-revertible 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.

`qsort_r` is available in both BSD and GNU flavors, depending on whether `_BSD_SOURCE` or `_GNU_SOURCE` is defined when compiling.

The Linux-specific function `renameat2` only supports the `RENAME_NOREPLACE` flag.

`basename` is available in both POSIX and GNU flavors, depending on whether `libgen.h` is included or not.

`sigpause` is available in both BSD and SysV/XSI flavors, depending on whether `_XOPEN_SOURCE` is defined when compiling.

`strerror_r` is available in both POSIX and GNU flavors, depending on whether `_GNU_SOURCE` is defined when compiling.

`dladdr` always sets the `DI_info` members `dli_sname` and `dli_saddr` to `NULL`, indicating no symbol matching `addr` could be found.

`getrlimit` resources `RLIMIT_AS`, `RLIMIT_CPU`, `RLIMIT_FSIZE`, `RLIMIT_DATA` always return `rlim_cur` and `rlim_max` as `RLIM_INFINITY`, so `setrlimit` returns -1 and sets `EINVAL` if they are lowered, or returns 0 if unchanged. `getrlimit` resource `RLIMIT_NOFILE` always returns `rlim_cur` and `rlim_max` as `OPEN_MAX`; `setrlimit` returns 0 sets `EINVAL` if `rlim_cur > rlim_max`, does not change the value if it is `RLIM_INFINITY`, otherwise returns the result from `setdtablesize`. `getrlimit/setrlimit` resources `RLIMIT_CORE` and `RLIMIT_STACK` return the current values and set the requested values. All other resource arguments return -1 and set `EINVAL`.

`fallocate` has a few Windows quirks: The `FALLOC_FL_ZERO_RANGE` operation is NOT atomic. With flags set to 0 and `FALLOC_FL_KEEP_SIZE`, sparse blocks in the given range are re-allocated as per the POSIX requirements. This re-allocation operation isn't atomic either. Over-allocation with `FALLOC_FL_KEEP_SIZE` is only temporary on Windows until the last handle to the file is closed. Over-allocation on sparse files is entirely ignored on Windows.

`sched_setpolicy` only emulates API behavior because Windows does not offer alternative scheduling policies. If `SCHED_OTHER` or `SCHED_BATCH` is selected, the Windows priority is set according to the nice value. If `SCHED_IDLE` is selected, the Windows priority is set to `IDLE_PRIORITY_CLASS`. If `SCHED_FIFO` or `SCHED_RR` is selected, the nice value is preserved and the Windows priority is set according to the `sched_priority` value. If the `SCHED_RESET_ON_FORK` flag is set, realtime policies and negative nice values are dropped on fork.

`nice`, `setpriority`, `sched_setparam` and `sched_setpolicy` map the nice value (`SCHED_OTHER`, `SCHED_BATCH`) or the `sched_priority` (`SCHED_FIFO`, `SCHED_RR`) to Windows priority classes as follows:

<code>SCHED_OTHER</code>	<code>SCHED_BATCH</code>	<code>SCHED_FIFO/RR</code>	
nice value	nice value	<code>sched_priority</code>	Windows priority class
12...19	4...19	1...6	<code>IDLE_PRIORITY_CLASS</code>
4...11	-4...3	7...12	<code>BELOW_NORMAL_PRIORITY_CLASS</code>
-4...3	-12...-5	13...18	<code>NORMAL_PRIORITY_CLASS</code>

-12...-5	-13..-19	19...24	ABOVE_NORMAL_PRIORITY_CLASS
-13..-19	-20	25...30	HIGH_PRIORITY_CLASS
-20	-	31...32	REALTIME_PRIORITY_CLASS

The use of values which are mapped to the `REALTIME_PRIORITY_CLASS` require administrative privileges.

Chapter 2

Cygwin Functions

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

2.1 Path conversion functions

2.1.1 cygwin_conv_path

cygwin_conv_path

Synopsis

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

```
ssize_t cygwin_conv_path(cygwin_conv_path_t what, const void * from, void * to, size_t size);
```

Description

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.                 */
CCP_PROC_CYGDRIVE = 0x200 /* Request to return /proc/cygdrive path
                        (only with CCP_*_TO_POSIX).          */
```

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

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");
}
```

2.1.2 `cygwin_conv_path_list`

`cygwin_conv_path_list`

Synopsis

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

```
ssize_t cygwin_conv_path_list(cygwin_conv_path_t what, const void * from, void * to, size_t size);
```

Description

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

See also [cygwin_conv_path](#)

2.1.3 `cygwin_create_path`

`cygwin_create_path`

Synopsis

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

```
void * cygwin_create_path(cygwin_conv_path_t what, const void * from);
```

Description

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

See also [cygwin_conv_path](#)

2.1.4 cygwin_posix_path_list_p

`cygwin_posix_path_list_p`

Synopsis

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

```
int cygwin_posix_path_list_p(const char *path);
```

Description

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.

2.1.5 cygwin_split_path

`cygwin_split_path`

Synopsis

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

```
void cygwin_split_path (const char * path, char * dir, char * file);
```

Description

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

Example

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);
```

2.2 Helper functions to change user context

2.2.1 `cygwin_logon_user`

`cygwin_logon_user`

Synopsis

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

HANDLE `cygwin_logon_user`(const struct passwd *passwd_entry, const char *password);

Description

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

See also the chapter [Switching the user context](#) in the Cygwin User's guide.

See also [cygwin_set_impersonation_token](#)

2.2.2 `cygwin_set_impersonation_token`

`cygwin_set_impersonation_token`

Synopsis

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

void `cygwin_set_impersonation_token`(const **HANDLE** token);

Description

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

See also the chapter [Switching the user context](#) in the Cygwin User's guide.

See also [cygwin_logon_user](#)

2.3 Miscellaneous functions

2.3.1 `cygwin_attach_handle_to_fd`

`cygwin_attach_handle_to_fd`

Synopsis

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

```
int cygwin_attach_handle_to_fd(char *name, int fd, HANDLE handle, int bin, int access);
```

Description

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.

2.3.2 `cygwin_internal`

`cygwin_internal`

Synopsis

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

```
uintptr_t cygwin_internal(cygwin_getinfo_types t, ...);
```

Description

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.

2.3.3 `cygwin_stackdump`

`cygwin_stackdump`

Synopsis

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

```
void cygwin_stackdump(void);
```

Description

Outputs a stackdump to stderr from the called location.

Note: This function only has an effect the first time it is called by a process.

Note: This function is deprecated.