NAME

magic_open, **magic_close**, **magic_error**, **magic_errno**, **magic_descriptor**, **magic_buffer**, **magic_getflags**, **magic_setflags**, **magic_check**, **magic_compile**, **magic_list**, **magic_load**, **magic_load_buffers**, **magic_setparam**, **magic_getparam**, **magic_version** - Magic number recognition library

LIBRARY

Magic Number Recognition Library (libmagic, -lmagic)

SYNOPSIS

#include <magic.h>

magic_t
magic_open(*int flags*);

void
magic_close(*magic_t cookie*);

*const char **
magic_error(*magic_t cookie*);

int
magic_errno(*magic_t cookie*);

*const char **
magic_descriptor(*magic_t cookie*, *int fd*);

*const char **
magic_file(*magic_t cookie*, *const char *filename*);

*const char **
magic_buffer(*magic_t cookie*, *const void *buffer*, *size_t length*);

int
magic_getflags(*magic_t cookie*);

int
magic_setflags(*magic_t cookie*, *int flags*);

int
magic_check(*magic_t cookie*, *const char *filename*);

int
magic_compile(*magic_t cookie*, *const char *filename*);

int
magic_list(*magic_t cookie*, *const char *filename*);

int
magic_load(*magic_t cookie*, *const char *filename*);

int
magic_load_buffers(*magic_t cookie*, *void **buffers*, *size_t *sizes*, *size_t nbuffers*);

int
magic_getparam(*magic_t cookie*, *int param*, *void *value*);

int
magic_setparam(*magic_t cookie*, *int param*, *const void *value*);

int
magic_version(*void*);

*const char **
magic_getpath(*const char *magicfile*, *int action*);

DESCRIPTION

These functions operate on the magic database file which is described in magic(5).

The function **magic_open**() creates a magic cookie pointer and returns it. It returns NULL if there was an error allocating the magic cookie. The *flags* argument specifies how the other magic functions should behave:

MAGIC_NONE No special handling.

MAGIC_DEBUG Print debugging messages to stderr.

MAGIC_SYMLINK If the file queried is a symlink, follow it.

MAGIC_COMPRESS If the file is compressed, unpack it and look at the contents.

MAGIC_DEVICES If the file is a block or character special device, then open the device and try to

look in its contents.

MAGIC_MIME_TYPE
: Return a MIME type string, instead of a textual description.

MAGIC_MIME_ENCODING
: Return a MIME encoding, instead of a textual description.

MAGIC_MIME
: A shorthand for MAGIC_MIME_TYPE | MAGIC_MIME_ENCODING.

MAGIC_CONTINUE
: Return all matches, not just the first.

MAGIC_CHECK
: Check the magic database for consistency and print warnings to stderr.

MAGIC_PRESERVE_ATIME
: On systems that support utime(3) or utimes(2), attempt to preserve the access time of files analysed.

MAGIC_RAW
: Don’t translate unprintable characters to a \ooo octal representation.

MAGIC_ERROR
: Treat operating system errors while trying to open files and follow symlinks as real errors, instead of printing them in the magic buffer.

MAGIC_APPLE
: Return the Apple creator and type.

MAGIC_EXTENSION
: Return a slash-separated list of extensions for this file type.

MAGIC_COMPRESS_TRANSP
: Don’t report on compression, only report about the uncompressed data.

MAGIC_NO_CHECK_APPTYPE
: Don’t check for EMX application type (only on EMX).

MAGIC_NO_COMPRESS_FORK
: Don’t allow decompressors that use fork.

MAGIC_NO_CHECK_CDF
: Don’t get extra information on MS Composite Document Files.

MAGIC_NO_CHECK_COMPRESS
: Don’t look inside compressed files.

MAGIC_NO_CHECK_ELF
: Don’t print ELF details.

MAGIC_NO_CHECK_ENCODING
: Don’t check text encodings.

MAGIC_NO_CHECK_SOFT
: Don’t consult magic files.

MAGIC_NO_CHECK_TAR
: Don’t examine tar files.

MAGIC_NO_CHECK_TEXT
: Don’t check for various types of text files.

MAGIC_NO_CHECK_TOKENS
: Don’t look for known tokens inside ascii files.

MAGIC_NO_CHECK_JSON
: Don’t examine JSON files.

MAGIC_NO_CHECK_CSV
: Don’t examine CSV files.

MAGIC_NO_CHECK_SIMH
: Don’t examine SIMH tape files.

The **magic_close**() function closes the magic(5) database and deallocates any resources used.

The **magic_error**() function returns a textual explanation of the last error, or NULL if there was no error.

The **magic_errno**() function returns the last operating system error number (errno(2)) that was encountered by a system call.

The **magic_file**() function returns a textual description of the contents of the *filename* argument, or NULL if an error occurred. If the *filename* is NULL, then stdin is used.

The **magic_descriptor**() function returns a textual description of the contents of the *fd* argument, or NULL if an error occurred.

The **magic_buffer**() function returns a textual description of the contents of the *buffer* argument with *length* bytes size.

The **magic_getflags**() functions returns a value representing current *flags* set.

The **magic_setflags**() function sets the *flags* described above. Note that using both MIME flags together can also return extra information on the charset.

The **magic_check**() function can be used to check the validity of entries in the colon separated database files passed in as *filename*, or NULL for the default database. It returns 0 on success and -1 on failure.

The **magic_compile**() function can be used to compile the colon separated list of database files passed in as *filename*, or NULL for the default database. It returns 0 on success and -1 on failure. The compiled files created are named from the basename(1) of each file argument with ".mgc" appended to it.

The **magic_list**() function dumps all magic entries in a human readable format, dumping first the entries that are matched against binary files and then the ones that match text files. It takes and optional *filename* argument which is a colon separated list of database files, or NULL for the default database.

The **magic_load**() function must be used to load the colon separated list of database files passed in as *filename*, or NULL for the default database file before any magic queries can performed.

The default database file is named by the MAGIC environment variable. If that variable is not set, the default database file name is /usr/share/misc/magic. **magic_load**() adds ".mgc" to the database filename as appropriate.

The **magic_load_buffers**() function takes an array of size *nbuffers* of *buffers* with a respective size for each in the array of *sizes* loaded with the contents of the magic databases from the filesystem. This function can be used in environment where the magic library does not have direct access to the filesystem, but can access the magic database via shared memory or other IPC means.

The **magic_getparam**() and **magic_setparam**() allow getting and setting various limits related to the magic library.

Parameter	**Type**	**Default**
MAGIC_PARAM_INDIR_MAX	size_t	15
MAGIC_PARAM_NAME_MAX	size_t	30

MAGIC_PARAM_ELF_NOTES_MAX	size_t	256
MAGIC_PARAM_ELF_PHNUM_MAX	size_t	128
MAGIC_PARAM_ELF_SHNUM_MAX	size_t	32768
MAGIC_PARAM_REGEX_MAX	size_t	8192
MAGIC_PARAM_BYTES_MAX	size_t	1048576

The MAGIC_PARAM_INDIR_RECURSION parameter controls how many levels of recursion will be followed for indirect magic entries.

The MAGIC_PARAM_NAME_RECURSION parameter controls how many levels of recursion will be followed for for name/use calls.

The MAGIC_PARAM_NAME_MAX parameter controls the maximum number of calls for name/use.

The MAGIC_PARAM_NOTES_MAX parameter controls how many ELF notes will be processed.

The MAGIC_PARAM_PHNUM_MAX parameter controls how many ELF program sections will be processed.

The MAGIC_PARAM_SHNUM_MAX parameter controls how many ELF sections will be processed.

The **magic_version**() command returns the version number of this library which is compiled into the shared library using the constant MAGIC_VERSION from <*magic.h*>. This can be used by client programs to verify that the version they compile against is the same as the version that they run against.

The **magic_getpath**() command returns the colon separated list of magic database locations. If the *filename* is non-NULL, then it is returned. Otherwise, if the MAGIC environment variable is defined, then it is returned. Otherwise, if *action* is 0 (meaning "file load"), then any user-specific magic database file is included. Otherwise, only the system default magic database path is included.

RETURN VALUES

The function **magic_open**() returns a magic cookie on success and NULL on failure setting errno to an appropriate value. It will set errno to EINVAL if an unsupported value for flags was given. The **magic_list**(), **magic_load**(), **magic_compile**(), and **magic_check**() functions return 0 on success and -1 on failure. The **magic_buffer**(), **magic_getpath**(), and **magic_file**(), functions return a string on success and NULL on failure. The **magic_error**() function returns a textual description of the errors of the above functions, or NULL if there was no error. The **magic_version**() always returns the version number of the library. Finally, **magic_setflags**() returns -1 on systems that don’t support utime(3), or utimes(2) when MAGIC_PRESERVE_ATIME is set.

FILES

/usr/share/misc/magic The non-compiled default magic database.
/usr/share/misc/magic.mgc The compiled default magic database.

SEE ALSO

file(1), magic(5)

BUGS

The results from **magic_buffer**() and **magic_file**() where the buffer and the file contain the same data can produce different results, because in the **magic_file**() case, the program can lseek(2) and stat(2) the file descriptor.

AUTHORS

M**ø**ns Rullg**ø**rd Initial libmagic implementation, and configuration.
Christos Zoulas API cleanup, error code and allocation handling.