NAME

elf32_newehdr, **elf64_newehdr**, **gelf_newehdr -** retrieve or allocate the object file header

LIBRARY

ELF Access Library (libelf, -lelf)

SYNOPSIS

#include <libelf.h>

*Elf32_Ehdr **
elf32_newehdr(*Elf *elf*);

*Elf64_Ehdr **
elf64_newehdr(*Elf *elf*);

#include <gelf.h>

*void **
gelf_newehdr(*Elf *elf, int elfclass*);

DESCRIPTION

These functions retrieve the ELF header from the ELF descriptor *elf*, allocating a new header if needed. File data structures are translated to their in-memory representations as described in elf(3).

Function **elf32_newehdr**() returns a pointer to a 32 bit *Elf32_Ehdr* structure. Function **elf64_newehdr**() returns a pointer to a 64 bit *Elf64_Ehdr* structure.

When argument *elfclass* has value ELFCLASS32, function **gelf_newehdr**() returns the value returned by **elf32_newehdr**(*elf*). When argument *elfclass* has value ELFCLASS64 it returns the value returned by **elf64_newehdr**(*elf*).

If a fresh header structure is allocated, the members of the structure are initialized as follows:

e_ident[EI_MAG0..EI_MAG3]
: Identification bytes at offsets EI_MAG0, EI_MAG1, EI_MAG2 and EI_MAG3 are set to the ELF signature.

e_ident[EI_CLASS]
: The identification byte at offset EI_CLASS is set to the ELF class associated with the function being called or to argument *elfclass* for function **gelf_newehdr**().

e_ident[EI_DATA]
The identification byte at offset EI_DATA is set to ELFDATANONE.

e_ident[EI_VERSION]
The identification byte at offset EI_VERSION is set to the ELF library's operating version set by a prior call to elf_version(3).

e_machine
is set to EM_NONE.

e_type
is set to ELF_K_NONE.

e_version
is set to the ELF library's operating version set by a prior call to elf_version(3).

Other members of the header are set to zero. The application is responsible for changing these values as needed before calling **elf_update**().

If successful, these three functions set the ELF_F_DIRTY flag on ELF descriptor *elf*.

RETURN VALUES

These functions return a pointer to a translated header descriptor if successful, or NULL on failure.

COMPATIBILITY

The **gelf_newehdr**() function uses a type of *void ** for its returned value. This differs from some other implementations of the elf(3) API, which use an *unsigned long* return type.

ERRORS

These functions can fail with the following errors:

[ELF_E_ARGUMENT]
The argument *elf* was null.

[ELF_E_ARGUMENT]
Argument *elf* was not a descriptor for an ELF object.

[ELF_E_ARGUMENT]
Argument *elfclass* had an unsupported value.

[ELF_E_ARGUMENT]	The class of the ELF descriptor *elf* did not match that of the requested operation.
[ELF_E_ARGUMENT]	For function **gelf_newehdr**(), the class of argument *elf* was not ELFCLASSNONE and did not match the argument *elfclass*.
[ELF_E_CLASS]	The ELF class of descriptor *elf* did not match that of the API function being called.
[ELF_E_HEADER]	A malformed ELF header was detected.
[ELF_E_RESOURCE]	An out of memory condition was detected during execution.
[ELF_E_SECTION]	The ELF descriptor in argument *elf* did not adhere to the conventions used for extended numbering.
[ELF_E_VERSION]	The ELF descriptor *elf* had an unsupported ELF version number.

SEE ALSO

elf(3), elf32_getehdr(3), elf64_getehdr(3), elf_flagdata(3), elf_getident(3), elf_update(3), elf_version(3), gelf(3), gelf_getehdr(3), elf(5)