NAME

elf_flagdata, elf_flagehdr, elf_flagelf, elf_flagphdr, elf_flagscn, elf_flagshdr – manipulate flags

SYNOPSIS

cc [*flag* …] *file* … –**lelf** [*library* …]

#include <libelf.h>

unsigned elf_flagdata(Elf_Data **data*, **Elf_Cmd** *cmd*, **unsigned** *flags***);**

unsigned elf_flagehdr(Elf **elf*, **Elf_Cmd** *cmd*, **unsigned** *flags***);**

unsigned elf_flagelf(Elf **elf*, **Elf_Cmd** *cmd*, **unsigned** *flags***);**

unsigned elf_flagphdr(Elf **elf*, **Elf_Cmd** *cmd*, **unsigned** *flags***);**

unsigned elf_flagscn(Elf_Scn **scn*, **Elf_Cmd** *cmd*, **unsigned** *flags***);**

unsigned elf_flagshdr(Elf_Scn **scn*, **Elf_Cmd** *cmd*, **unsigned** *flags***);**

MT-LEVEL

Unsafe

DESCRIPTION

These functions manipulate the flags associated with various structures of an ELF file. Given an ELF descriptor (*elf*), a data descriptor (*data*), or a section descriptor (*scn*), the functions may set or clear the associated status bits, returning the updated bits. A null descriptor is allowed, to simplify error handling; all functions return zero for this degenerate case.

cmd may have the following values:

ELF_C_CLR The functions clear the bits that are asserted in *flags*. Only the non-zero bits in *flags* are cleared; zero bits do not change the status of the descriptor.

ELF_C_SET The functions set the bits that are asserted in *flags*. Only the non-zero bits in *flags* are set; zero bits do not change the status of the descriptor.

Descriptions of the defined *flags* bits appear below:

ELF_F_DIRTY When the program intends to write an ELF file, this flag asserts the associated information needs to be written to the file. Thus, for example, a program that wished to update the ELF header of an existing file would call **elf_flagehdr( )** with this bit set in *flags* and *cmd* equal to **ELF_C_SET**. A later call to **elf_update( )** would write the marked header to the file.

ELF_F_LAYOUT Normally, the library decides how to arrange an output file. That is, it automatically decides where to place sections, how to align them in the file, etc. If this bit is set for an ELF descriptor, the program assumes responsibility for determining all file positions. This bit is meaningful only for **elf_flagelf( )** and applies to the entire file associated with the descriptor.

When a flag bit is set for an item, it affects all the subitems as well. Thus, for example, if the program sets the **ELF_F_DIRTY** bit with **elf_flagelf( )**, the entire logical file is ''dirty.''

EXAMPLES

The following fragment shows how one might mark the ELF header to be written to the output file:

```
/* dirty ehdr ... */
ehdr = elf32_getehdr(elf);
elf_flagehdr(elf, ELF_C_SET, ELF_F_DIRTY);
```

SEE ALSO

elf(3E), **elf32_getehdr**(3E), **elf_getdata**(3E), **elf_update**(3E)