NAME

offsetof – offset of a structure member

SYNOPSIS

#include <stddef.h>

size_t offsetof(*type*, *member*);

DESCRIPTION

The macro **offsetof**() returns the offset of the field *member* from the start of the structure *type*.

This macro is useful because the sizes of the fields that compose a structure can vary across implementations, and compilers may insert different numbers of padding bytes between fields. Consequently, an element's offset is not necessarily given by the sum of the sizes of the previous elements.

A compiler error will result if *member* is not aligned to a byte boundary (i.e., it is a bit field).

RETURN VALUE

offsetof() returns the offset of the given *member* within the given *type*, in units of bytes.

CONFORMING TO

POSIX.1-2001, POSIX.1-2008, C89, C99.

EXAMPLE

On a Linux/i386 system, when compiled using the default **gcc**(1) options, the program below produces the following output:

```
$ ./a.out
offsets: i=0; c=4; d=8 a=16
sizeof(struct s)=16
```

Program source

```
#include <stddef.h>
#include <stdio.h>
#include <stdlib.h>

int
main(void)
{
    struct s {
        int i;
        char c;
        double d;
        char a[];
    };

    /* Output is compiler dependent */

    printf("offsets: i=%zd; c=%zd; d=%zd a=%zd\n",
            offsetof(struct s, i), offsetof(struct s, c),
            offsetof(struct s, d), offsetof(struct s, a));
    printf("sizeof(struct s)=%zd\n", sizeof(struct s));

    exit(EXIT_SUCCESS);
}
```

COLOPHON

This page is part of release 5.05 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man−pages/.