**NAME**

utime – set file access and modification times

**SYNOPSIS**

```
#include <sys/types.h>
#include <utime.h>

int utime(const char *path, const struct utimbuf *times);
```

**MT-LEVEL**

Async-Signal-Safe

**DESCRIPTION**

**utime( )** sets the access and modification times of the file pointed to by *path*.

If *times* is **NULL**, the access and modification times of the file are set to the current time. A process must be the owner of the file or have write permission to use **utime( )** in this manner.

If *times* is not **NULL,** *times* is interpreted as a pointer to a **utimbuf** structure (defined in **utime.h**) and the access and modification times are set to the values contained in the designated structure. Only the owner of the file or the super-user may use **utime( )** this way. The **utimbuf** structure contains the following members:

```
time_t   actime;     /* access time */
time_t   modtime;    /* modification time */
```

The times in the members of the **utimbuf** structure are measured in seconds since 00:00:00 UTC, Jan. 1, 1970.

**utime( )** also causes the time of the last file status change (**st_ctime**) to be updated.

**RETURN VALUES**

Upon successful completion, a value of 0 is returned. Otherwise, a value of –1 is returned and **errno** is set to indicate the error.

**ERRORS**

**utime( )** will fail if one or more of the following are true:

| | |
|---|---|
| **EACCES** | Search permission is denied by a component of the *path* prefix. |
| **EACCES** | The effective user ID of the process is not super-user and not the owner of the file, write permission is denied for the file, and *times* is NULL. |
| **EFAULT** | *path* points to an illegal address. |
| **EINTR** | A signal was caught during the **utime( )** function. |
| **EIO** | An I/O error occurred while reading from or writing to the file system. |
| **ELOOP** | Too many symbolic links were encountered in translating *path*. |
| **EMULTIHOP** | Components of *path* require hopping to multiple remote machines and the file system does not allow it. |
| **ENAMETOOLONG** | The length of the *path* argument exceeds {**PATH_MAX**}, or the length of a *path* component exceeds {**NAME_MAX**} while {**_POSIX_NO_TRUNC**} is in effect. |
| **ENOENT** | The named file does not exist or is a null pathname. |

| | |
|---|---|
| **ENOLINK** | *path* points to a remote machine and the link to that machine is no longer active. |
| **ENOTDIR** | A component of the *path* prefix is not a directory. |
| **EPERM** | The effective user of the calling process is not super-user and not the owner of the file, and *times* is not **NULL**. |
| **EROFS** | The file system containing the file is mounted read-only. |

**SEE ALSO**

**stat**(2)