## NAME

basename, dirname − parse pathname components

## LIBRARY

Standard C library (*libc*, *−lc*)

## SYNOPSIS

**#include <libgen.h>**

**char *dirname(char *_path_);**
**char *basename(char *_path_);**

## DESCRIPTION

Warning: there are two different functions **basename**(); see below.

The functions **dirname**() and **basename**() break a null-terminated pathname string into directory and filename components. In the usual case, **dirname**() returns the string up to, but not including, the final '/', and **basename**() returns the component following the final '/'. Trailing '/' characters are not counted as part of the pathname.

If *path* does not contain a slash, **dirname**() returns the string "." while **basename**() returns a copy of *path*. If *path* is the string "/", then both **dirname**() and **basename**() return the string "/". If *path* is a null pointer or points to an empty string, then both **dirname**() and **basename**() return the string ".".

Concatenating the string returned by **dirname**(), a "/", and the string returned by **basename**() yields a complete pathname.

Both **dirname**() and **basename**() may modify the contents of *path*, so it may be desirable to pass a copy when calling one of these functions.

These functions may return pointers to statically allocated memory which may be overwritten by subsequent calls. Alternatively, they may return a pointer to some part of *path*, so that the string referred to by *path* should not be modified or freed until the pointer returned by the function is no longer required.

The following list of examples (taken from SUSv2) shows the strings returned by **dirname**() and **basename**() for different paths:

| **path** | **dirname** | **basename** |
|---|---|---|
| /usr/lib | /usr | lib |
| /usr/ | / | usr |
| usr | . | usr |
| / | / | / |
| . | . | . |
| .. | . | .. |

## RETURN VALUE

Both **dirname**() and **basename**() return pointers to null-terminated strings. (Do not pass these pointers to **free**(3).)

## ATTRIBUTES

For an explanation of the terms used in this section, see **attributes**(7).

| **Interface** | **Attribute** | **Value** |
|---|---|---|
| **basename**(), **dirname**() | Thread safety | MT-Safe |

## STANDARDS

POSIX.1-2001, POSIX.1-2008.

## NOTES

There are two different versions of **basename**() - the POSIX version described above, and the GNU version, which one gets after

```
#define _GNU_SOURCE         /* See feature_test_macros(7) */
```

```
#include <string.h>
```

The GNU version never modifies its argument, and returns the empty string when *path* has a trailing slash, and in particular also when it is "/". There is no GNU version of **dirname**().

With glibc, one gets the POSIX version of **basename**() when <*libg en.h*> is included, and the GNU version otherwise.

## BUGS

In the glibc implementation, the POSIX versions of these functions modify the *path* argument, and segfault when called with a static string such as "/usr/".

Before glibc 2.2.1, the glibc version of **dirname**() did not correctly handle pathnames with trailing '/' characters, and generated a segfault if given a NULL argument.

## EXAMPLES

The following code snippet demonstrates the use of **basename**() and **dirname**():

```
char *dirc, *basec, *bname, *dname;
char *path = "/etc/passwd";

dirc = strdup(path);
basec = strdup(path);
dname = dirname(dirc);
bname = basename(basec);
printf("dirname=%s, basename=%s\n", dname, bname);
```

## SEE ALSO

**basename**(1), **dirname**(1)