

NAME

strdup, strndup, strdupa, strndupa – duplicate a string

LIBRARY

Standard C library (*libc*, *-lc*)

SYNOPSIS

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

```
char *strdup(const char *s);
```

```
char *strndup(const char s[.n], size_t n);
```

```
char *strdupa(const char *s);
```

```
char *strndupa(const char s[.n], size_t n);
```

Feature Test Macro Requirements for glibc (see **feature_test_macros(7)**):

strdup():

```
_XOPEN_SOURCE >= 500
```

```
/* Since glibc 2.12: */ _POSIX_C_SOURCE >= 200809L
```

```
/* glibc <= 2.19: */ _BSD_SOURCE || _SVID_SOURCE
```

strndup():

Since glibc 2.10:

```
_POSIX_C_SOURCE >= 200809L
```

Before glibc 2.10:

```
_GNU_SOURCE
```

strdupa(), strndupa():

```
_GNU_SOURCE
```

DESCRIPTION

The **strdup()** function returns a pointer to a new string which is a duplicate of the string *s*. Memory for the new string is obtained with **malloc(3)**, and can be freed with **free(3)**.

The **strndup()** function is similar, but copies at most *n* bytes. If *s* is longer than *n*, only *n* bytes are copied, and a terminating null byte ('\0') is added.

strdupa() and **strndupa()** are similar, but use **alloca(3)** to allocate the buffer.

RETURN VALUE

On success, the **strdup()** function returns a pointer to the duplicated string. It returns NULL if insufficient memory was available, with *errno* set to indicate the error.

ERRORS**ENOMEM**

Insufficient memory available to allocate duplicate string.

ATTRIBUTES

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

Interface	Attribute	Value
strdup() , strndup() , strdupa() , strndupa()	Thread safety	MT-Safe

STANDARDS

strdup() conforms to SVr4, 4.3BSD, POSIX.1-2001. **strndup()** conforms to POSIX.1-2008. **strdupa()** and **strndupa()** are GNU extensions.

SEE ALSO

alloca(3), **calloc(3)**, **free(3)**, **malloc(3)**, **realloc(3)**, **string(3)**, **wcsdup(3)**