NAME

strtoul, **strtoull**, **strtoumax**, **strtouq** **-** convert a string to an *unsigned long*, *unsigned long long*, *uintmax_t*, or *u_quad_t* integer

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <stdlib.h>
#include <limits.h>

unsigned long
strtoul(*const char * restrict nptr, char ** restrict endptr, int base*);

unsigned long long
strtoull(*const char * restrict nptr, char ** restrict endptr, int base*);

#include <inttypes.h>

uintmax_t
strtoumax(*const char * restrict nptr, char ** restrict endptr, int base*);

#include <sys/types.h>
#include <stdlib.h>
#include <limits.h>

u_quad_t
strtouq(*const char *nptr, char **endptr, int base*);

DESCRIPTION

The **strtoul**() function converts the string in *nptr* to an *unsigned long* value. The **strtoull**() function converts the string in *nptr* to an *unsigned long long* value. The **strtoumax**() function converts the string in *nptr* to an *uintmax_t* value. The **strtouq**() function converts the string in *nptr* to a *u_quad_t* value. The conversion is done according to the given *base*, which must be between 2 and 36 inclusive, or be the special value 0.

The string may begin with an arbitrary amount of white space (as determined by isspace(3)) followed by a single optional '+' or '-' sign. If *base* is zero or 16, the string may then include a "0b" prefix, and the number will be read in base 2; or it may include a "0x" prefix, and the number will be read in base 16; otherwise, a zero *base* is taken as 10 (decimal) unless the next character is '0', in which case it is taken as 8 (octal).

The remainder of the string is converted to an *unsigned long* value in the obvious manner, stopping at the end of the string or at the first character that does not produce a valid digit in the given base. (In bases above 10, the letter ‘A’ in either upper or lower case represents 10, ‘B’ represents 11, and so forth, with ‘Z’ representing 35.)

If *endptr* is not NULL, **strtoul**() stores the address of the first invalid character in **endptr*. If there were no digits at all, however, **strtoul**() stores the original value of *nptr* in **endptr*. (Thus, if **nptr* is not ‘\0’ but ***endptr* is ‘\0’ on return, the entire string was valid.)

RETURN VALUES

The **strtoul**(), **strtoull**(), **strtoumax**() and **strtouq**() functions return either the result of the conversion or, if there was a leading minus sign, the negation of the result of the conversion, unless the original (non-negated) value would overflow; in the latter case, **strtoul**() returns ULONG_MAX, **strtoull**() returns ULLONG_MAX, **strtoumax**() returns UINTMAX_MAX, and **strtouq**() returns ULLONG_MAX. In all cases, *errno* is set to ERANGE. If no conversion could be performed, 0 is returned and the global variable *errno* is set to EINVAL (the last feature is not portable across all platforms).

ERRORS

[EINVAL]	The value of *base* is not supported or no conversion could be performed (the last feature is not portable across all platforms).
[ERANGE]	The given string was out of range; the value converted has been clamped.

SEE ALSO

strtol(3), strtonum(3), wcstoul(3)

STANDARDS

The **strtoul**() function conforms to ISO/IEC 9899:1990 ("ISO C90"). The **strtoull**() and **strtoumax**() functions conform to ISO/IEC 9899:1999 ("ISO C99"). The BSD **strtouq**() function is deprecated.