NAME

strtol, strtoll, strtoul, strtoull, atol, atoll, atoi, lltostr, ulltostr – conversion routines

SYNOPSIS

#include <stdlib.h>

long strtol(const char **str*, **char** ***ptr*, **int** *base***);**

long long strtoll(const char **str*, **char** ***ptr*, **int** *base***);**

unsigned long strtoul(const char **str*, **char** ***ptr*, **int** *base***);**

unsigned long long strtoull(const char **str*, **char** ***ptr*, **int** *base***);**

long atol(const char **str***);**

long long atoll(const char **str***);**

int atoi(const char **str***);**

char ***lltostr(long long** *value*, **char** **ptr***);**

char ***ulltostr(unsigned long long** *value*, **char** **ptr***);**

MT-LEVEL

MT-Safe

DESCRIPTION

strtol( ) returns as a **long** integer the value represented by the character string pointed to by **str**. The string is scanned up to the first character inconsistent with *base*. Leading ''white-space'' characters (as defined by **isspace( )** in **ctype**(3C)) are ignored.

strtoll( ) is similar to **strtol( )** except that the value is returned as a **long long**.

If the value of *ptr* is not **(char **)NULL**, a pointer to the character terminating the scan is returned in the location pointed to by *ptr*. If no integer can be formed, that location is set to **str**, and zero is returned.

If *base* is positive (and not greater than 36), it is used as the base for conversion. After an optional leading sign, leading zeros are ignored, and ''0x'' or ''0X'' is ignored if *base* is 16.

If *base* is zero, the string itself determines the base as follows: After an optional leading sign a leading zero indicates octal conversion, and a leading ''0x'' or ''0X'' hexadecimal conversion. Otherwise, decimal conversion is used.

Truncation from **long** to **int** can, of course, take place upon assignment or by an explicit cast.

If the value represented by *str* would cause overflow, **LONG_MAX** or **LONG_MIN** is returned (according to the sign of the value), and **errno** is set to the value, **ERANGE**.

strtoul( ) and **strtoull( )** are similar to **strtol( )** except that **strtoul( )** returns the value represented by *str* as an **unsigned long** integer and **strtoull( )** returns this value as an **unsigned long long**. If the value represented by *str* would cause overflow, **ULONG_MAX** is returned, and **errno** is set to the value, **ERANGE**.

Except for behavior on error, **atol( )** is equivalent to: **strtol(str, (char** ****)NULL, 10)**.

Except for behavior on error, **atoll( )** is equivalent to: **strtoll(str, (char** ****)NULL, 10)**.

Except for behavior on error, **atoi( )** is equivalent to: **(int) strtol(str, (char **)NULL, 10)**.

lltostr( ) returns a pointer to the string represented by the **long long** *value*.

ulltostr( ) is similar to **lltostr( )** except that *value* is an **unsigned long long**.

RETURN VALUES

If **strtol( )** is given a *base* greater than 36, it returns 0 and sets **errno** to **EINVAL**.

SEE ALSO

ctype(3C), **scanf**(3S), **strtod**(3C)

NOTES

strtol( ) no longer accepts values greater than **LONG_MAX** as valid input. Use **strtoul( )** instead.