NAME

HTTP::Date − HTTP::Date − date conversion routines

VERSION

version 6.05

SYNOPSIS

```
use HTTP::Date;

$string = time2str($time);    # Format as GMT ASCII time
$time = str2time($string);    # convert ASCII date to machine time
```

DESCRIPTION

This module provides functions that deal the date formats used by the HTTP protocol (and then some more). Only the first two functions, **time2str()** and **str2time()**, are exported by default.

time2str([$time])

The **time2str()** function converts a machine time (seconds since epoch) to a string. If the function is called without an argument or with an undefined argument, it will use the current time.

The string returned is in the format preferred for the HTTP protocol. This is a fixed length subset of the format defined by RFC 1123, represented in Universal Time (GMT). An example of a time stamp in this format is:

```
Sun, 06 Nov 1994 08:49:37 GMT
```

str2time(`$str` [, `$zone`])

The **str2time()** function converts a string to machine time. It returns `undef` if the format of `$str` is unrecognized, otherwise whatever the `Time::Local` functions can make out of the parsed time. Dates before the system's epoch may not work on all operating systems. The time formats recognized are the same as for **parse_date()**.

The function also takes an optional second argument that specifies the default time zone to use when converting the date. This parameter is ignored if the zone is found in the date string itself. If this parameter is missing, and the date string format does not contain any zone specification, then the local time zone is assumed.

If the zone is not "`GMT`" or numerical (like "`−0800`" or "`+0100`"), then the `Time::Zone` module must be installed in order to get the date recognized.

parse_date(`$str`)

This function will try to parse a date string, and then return it as a list of numerical values followed by a (possible undefined) time zone specifier; ($year, `$month`, `$day`, `$hour`, `$min`, `$sec`, `$tz`). The `$year` will be the full 4−digit year, and `$month` numbers start with 1 (for January).

In scalar context the numbers are interpolated in a string of the "YYYY-MM-DD hh:mm:ss TZ"−format and returned.

If the date is unrecognized, then the empty list is returned (`undef` in scalar context).

The function is able to parse the following formats:

```
"Wed, 09 Feb 1994 22:23:32 GMT"       -- HTTP format
"Thu Feb  3 17:03:55 GMT 1994"        -- ctime(3) format
"Thu Feb  3 00:00:00 1994",           -- ANSI C asctime() format
"Tuesday, 08-Feb-94 14:15:29 GMT"     -- old rfc850 HTTP format
"Tuesday, 08-Feb-1994 14:15:29 GMT"   -- broken rfc850 HTTP format

"03/Feb/1994:17:03:55 -0700"   -- common logfile format
"09 Feb 1994 22:23:32 GMT"     -- HTTP format (no weekday)
"08-Feb-94 14:15:29 GMT"       -- rfc850 format (no weekday)
"08-Feb-1994 14:15:29 GMT"     -- broken rfc850 format (no weekday)
```

```
"1994-02-03 14:15:29 -0100"    -- ISO 8601 format
"1994-02-03 14:15:29"          -- zone is optional
"1994-02-03"                   -- only date
"1994-02-03T14:15:29"          -- Use T as separator
"19940203T141529Z"             -- ISO 8601 compact format
"19940203"                     -- only date

"08-Feb-94"         -- old rfc850 HTTP format    (no weekday, no time)
"08-Feb-1994"       -- broken rfc850 HTTP format (no weekday, no time)
"09 Feb 1994"       -- proposed new HTTP format  (no weekday, no time)
"03/Feb/1994"       -- common logfile format     (no time, no offset)

"Feb  3  1994"      -- Unix 'ls -l' format
"Feb  3 17:03"      -- Unix 'ls -l' format

"11-15-96  03:52PM" -- Windows 'dir' format
```

The parser ignores leading and trailing whitespace. It also allow the seconds to be missing and the month to be numerical in most formats.

If the year is missing, then we assume that the date is the first matching date *before* current month. If the year is given with only 2 digits, then **parse_date()** will select the century that makes the year closest to the current date.

time2iso([$time])

Same as **time2str()**, but returns a ‘‘YYYY-MM-DD hh:mm:ss’’-formatted string representing time in the local time zone.

time2isoz([$time])

Same as **time2str()**, but returns a ‘‘YYYY-MM-DD hh:mm:ssZ’’-formatted string representing Universal Time.

SEE ALSO

‘‘time’’ in perlfunc, Time::Zone

AUTHOR

Gisle Aas <gisle@activestate.com>

COPYRIGHT AND LICENSE