Package: hms (via r-universe)

May 24, 2026

Title Pretty Time of Day

Date 2026-05-24

Version 1.1.4.9013

Description Implements an S3 class for storing and formatting time-of-day values, based on the 'difftime' class.

License MIT + file LICENSE

URL https://hms.tidyverse.org/, https://github.com/tidyverse/hms

BugReports https://github.com/tidyverse/hms/issues

Imports cli, lifecycle, methods, pkgconfig, rlang (>= 1.0.2), vctrs (>= 0.3.8)

Suggests crayon, lubridate, pillar (>= 1.1.0), testthat (>= 3.0.0)

Config/Needs/website tidyverse/tidytemplate

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0.9000

Repository https://cynkra.r-universe.dev

Date/Publication 2026-05-24 04:51:32 UTC

RemoteUrl https://github.com/tidyverse/hms

RemoteRef HEAD

RemoteSha b89649d793369b082eb79ce628e06fc97acc999c

Contents

hms *A simple class for storing time-of-day values*

Description

The values are stored as a difftime vector with a custom class, and always with "seconds" as unit for robust coercion to numeric. Supports construction from time values, coercion to and from various data types, and formatting. Can be used as a regular column in a data frame.

`hms()` is a high-level constructor that accepts second, minute, hour and day components as numeric vectors.

`new_hms()` is a low-level constructor that only checks that its input has the correct base type, numeric.

`is_hms()` checks if an object is of class `hms`.

`as_hms()` is a generic that supports conversions beyond casting. The default method forwards to `vec_cast()`.

Usage

```
hms(seconds = NULL, minutes = NULL, hours = NULL, days = NULL)

new_hms(x = numeric())

is_hms(x)

as_hms(x, ...)

## S3 method for class 'hms'
as.POSIXct(x, ...)

## S3 method for class 'hms'
as.POSIXlt(x, ...)

## S3 method for class 'hms'
as.character(x, ...)

## S3 method for class 'hms'
format(x, ...)

## S3 method for class 'hms'
print(x, ...)

## S3 method for class 'hms'
seq(from = hms(1), to = hms(1), by = NULL, ...)
```

Arguments

seconds, minutes, hours, days	Time since midnight. No bounds checking is performed.
x	An object.
...	additional arguments to be passed to or from methods.
from, to	the starting and (maximal) end values of the sequence. Of length 1 unless just from is supplied as an unnamed argument.
by	number: increment of the sequence.

Details

For hms(), all arguments must have the same length or be NULL. Odd combinations (e.g., passing only seconds and hours but not minutes) are rejected.

For arguments of type POSIXct and POSIXlt, as_hms() does not perform timezone conversion. Use lubridate::with_tz() and lubridate::force_tz() as necessary.

Examples

```
hms(56, 34, 12)
hms()

new_hms(as.numeric(1:3))
# Supports numeric only!
try(new_hms(1:3))

as_hms(1)
as_hms("12:34:56")
as_hms(Sys.time())
as.POSIXct(hms(1))
data.frame(a = hms(1))
d <- data.frame(hours = 1:3)
d$hours <- hms(hours = d$hours)
d
```

---

parse_hms *Parsing hms values*

Description

These functions convert character vectors to objects of the hms class. NA values are supported.

parse_hms() accepts values of the form "HH:MM:SS", with optional fractional seconds.

parse_hm() accepts values of the form "HH:MM".

Usage

```
parse_hms(x)

parse_hm(x)
```

Arguments

x	A character vector

Value

An object of class hms.

Examples

```
parse_hms("12:34:56")
parse_hms("12:34:56.789")
parse_hm("12:34")
```

---

round_hms *Round an hms object*

Description

Convenience functions to round to a multiple of seconds or digits.

Usage

```
round_hms(x, secs = NULL, digits = NULL)

trunc_hms(x, secs = NULL, digits = NULL)

ceiling_hms(x, secs = NULL, digits = NULL)

floor_hms(x, secs = NULL, digits = NULL)
```

Arguments

x	A vector of class hms
secs	Multiple of seconds, a positive numeric. Values less than one are supported
digits	Number of digits, a whole number. Negative numbers are supported.

Value

The input, rounded or truncated to the nearest multiple of secs (or number of digits)

Examples

```
round_hms(as_hms("12:34:56"), 5)
round_hms(as_hms("12:34:56"), 60)
round_hms(as_hms("12:34:56.78"), 0.25)
round_hms(as_hms("12:34:56.78"), digits = 1)
round_hms(as_hms("12:34:56.78"), digits = -2)
trunc_hms(as_hms("12:34:56"), 60)
ceiling_hms(as_hms("12:34:56"), 60)
floor_hms(as_hms("12:34:56"), 60)
```

---

vec_cast.hms *Casting*

Description

Double dispatch methods to support `vctrs::vec_cast()`.

Usage

```
## S3 method for class 'hms'
vec_cast(x, to, ...)
```

Arguments

x	Vectors to cast.
to	Type to cast to. If `NULL`, `x` will be returned as is.
...	For `vec_cast_common()`, vectors to cast. For `vec_cast()`, `vec_cast_default()`, and `vec_restore()`, these dots are only for future extensions and should be empty.

---

vec_ptype2.hms *Coercion*

Description

Double dispatch methods to support `vctrs::vec_ptype2()`.

Usage

```
## S3 method for class 'hms'
vec_ptype2(x, y, ..., x_arg = "", y_arg = "")
```

Arguments

x, y	Vector types.
...	These dots are for future extensions and must be empty.
x_arg, y_arg	Argument names for x and y. These are used in error messages to inform the user about the locations of incompatible types (see stop_incompatible_type()).

Index