Package ‘hms’

May 17, 2021

Title Pretty Time of Day
Date 2021-05-17
Version 1.1.0
Description Implements an S3 class for storing and formatting time-of-day values, based on the ‘difftime’ class.
Imports ellipsis (>= 0.3.2), lifecycle, methods, pkgconfig, rlang, vctrs (>= 0.3.8)
Suggests crayon, lubridate, pillar (>= 1.1.0), testthat (>= 3.0.0)
License MIT + file LICENSE
Encoding UTF-8
BugReports https://github.com/tidyverse/hms/issues
RoxygenNote 7.1.1.9001
Config/testthat/edition 3
NeedsCompilation no
Author Kirill Müller [aut, cre],
  R Consortium [fnd],
  RStudio [fnd]
Maintainer Kirill Müller <krlmlr+r@mailbox.org>
Repository CRAN
Date/Publication 2021-05-17 17:20:15 UTC

R topics documented:

hms-package .......................................................... 2
hms ................................................................. 2
parse_hms .......................................................... 4
round_hms .......................................................... 5
vec_cast.hms ......................................................... 5
vec_ptype2.hms ..................................................... 6

Index 7
# hms-package

## hms: Pretty Time of Day

## Description

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

## Details

[Stable]

### Author(s)

**Maintainer:** Kirill Müller <krlmlr+r@mailbox.org>

Other contributors:

- R Consortium [funder]
- RStudio [funder]

### See Also

Useful links:

- [https://hms.tidyverse.org/](https://hms.tidyverse.org/)
- [https://github.com/tidyverse/hms](https://github.com/tidyverse/hms)
- Report bugs at [https://github.com/tidyverse/hms/issues](https://github.com/tidyverse/hms/issues)

## 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

\texttt{hms(seconds = NULL, minutes = NULL, hours = NULL, days = NULL)}

\texttt{new_hms(x = numeric())}

\texttt{is_hms(x)}

\texttt{as_hms(x, ...)}

\texttt{## S3 method for class 'hms'}
\texttt{as.POSIXct(x, ...)}

\texttt{## S3 method for class 'hms'}
\texttt{as.POSIXlt(x, ...)}

\texttt{## S3 method for class 'hms'}
\texttt{as.character(x, ...)}

\texttt{## S3 method for class 'hms'}
\texttt{format(x, ...)}

\texttt{## S3 method for class 'hms'}
\texttt{print(x, ...)}

Arguments

- \texttt{seconds, minutes, hours, days}
  - Time since midnight. No bounds checking is performed.
- \texttt{x}
  - An object.
- \texttt{...}
  - Additional arguments to be passed to or from methods.

Details

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

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

Examples

\texttt{hms(56, 34, 12)}
\texttt{hms()}\n
\texttt{new_hms(as.numeric(1:3))}
# Supports numeric only!
\texttt{try(new_hms(1:3))}

\texttt{as_hms(1)}
parse_hms

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

```r
parse_hms(x)
parse_hm(x)
```

Arguments

- `x`  
  A character vector

Value

An object of class `hms`.

Examples

```r
parse_hms("12:34:56")
pread_hms("12:34:56.789")
pread_hm("12:34")
```
round_hms

Round or truncate to a multiple of seconds

Description
Convenience functions to round or truncate to a multiple of seconds.

Usage
round_hms(x, secs = NULL, digits = NULL)
trunc_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)

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.

### Description

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

### Usage

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

### Arguments

- **x**: Vector types.
- **y**: Vector types.
- **...**: These dots are for future extensions and must be empty.
- **x_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()`).
- **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

as.character.hms (hms), 2
as.POSIXct.hms (hms), 2
as.POSIXlt.hms (hms), 2
as_hms (hms), 2
difftime, 2
format.hms (hms), 2
hms, 2, 4, 5
hms-package, 2
is_hms (hms), 2
lubridate::force_tz(), 3
lubridate::with_tz(), 3
new_hms (hms), 2
numeric, 2
parse_hm (parse_hms), 4
parse_hms, 4
POSIXct, 3
POSIXlt, 3
print.hms (hms), 2
round_hms, 5
stop_incompatible_type(), 6
trunc_hms (round_hms), 5
vctrs::vec_cast(), 5
vctrs::vec_ptype2(), 6
vec_cast(), 2
vec_cast.hms, 5
vec_ptype2.hms, 6