

Package ‘hms’

August 23, 2019

Title Pretty Time of Day

Date 2019-08-23

Version 0.5.1

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

Imports methods,
pkgconfig,
rlang,
vctrs (>= 0.2.0)

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

License GPL-3

Encoding UTF-8

LazyData true

URL <https://github.com/tidyverse/hms>

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

RoxygenNote 6.1.1

Roxygen list(markdown = TRUE)

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:

- The R Consortium [funder]
- RStudio [funder]

See Also

Useful links:

- <https://github.com/tidyverse/hms>
- Report bugs at <https://github.com/tidyverse/hms/issues>

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()` 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'
as.data.frame(x, row.names = NULL, optional = FALSE, ...,
  nm = paste(deparse(substitute(x)), width.cutoff = 500L), collapse =
  " ")

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

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

```

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. |
| row.names | NULL or a character vector giving the row names for the data frame. Missing values are not allowed. |
| optional | logical. If TRUE, setting row names and converting column names (to syntactic names: see make.names) is optional. Note that all of R's base package <code>as.data.frame()</code> methods use <code>optional</code> only for column names treatment, basically with the meaning of <code>data.frame(*, check.names = !optional)</code> . See also the <code>make.names</code> argument of the <code>matrix</code> method. |
| nm | Name of column in new data frame |

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 | <i>Round or truncate to a multiple of seconds</i> |
|-----------|---|

Description

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

Usage

```
round_hms(x, secs)
```

```
trunc_hms(x, secs)
```

Arguments

| | |
|------|---|
| x | A vector of class hms |
| secs | Multiple of seconds, a positive numeric. Values less than one are supported |

Value

The input, rounded or truncated to the nearest multiple of secs

Examples

```
round_hms(as.hms("12:34:56"), 5)
round_hms(as.hms("12:34:56"), 60)
trunc_hms(as.hms("12:34:56"), 60)
```

| | |
|--------------|----------------|
| vec_cast.hms | <i>Casting</i> |
|--------------|----------------|

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 <code>vec_cast_common()</code> , vectors to cast. For <code>vec_cast()</code> and <code>vec_restore()</code> , 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

| | |
|--------------------|---|
| <code>x</code> | Vector types. |
| <code>y</code> | Vector types. |
| <code>...</code> | These dots are for future extensions and must be empty. |
| <code>x_arg</code> | Argument names for <code>x</code> and <code>y</code> . These are used in error messages to inform the user about the locations of incompatible types (see stop_incompatible_type()). |
| <code>y_arg</code> | Argument names for <code>x</code> and <code>y</code> . 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.data.frame.hms` (hms), 2
`as.POSIXct.hms` (hms), 2
`as.POSIXlt.hms` (hms), 2
`as_hms` (hms), 2

`data.frame`, 3
`difftime`, 2

`format.hms` (hms), 2

`hms`, 2, 4, 5
`hms-package`, 2

`is_hms` (hms), 2

`lubridate::force_tz()`, 4
`lubridate::with_tz()`, 4

`make.names`, 3

`new_hms` (hms), 2
`numeric`, 2

`parse_hm` (parse_hms), 4
`parse_hms`, 4
`POSIXct`, 4
`POSIXlt`, 4
`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