Package ‘hms’

October 13, 2022

Title Pretty Time of Day

Date 2022-08-19

Version 1.1.2

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 7.2.1

Config/testthat/edition 3

Config/autostyle/scope line_breaks

Config/autostyle/strict false

Config/Needs/website tidyverse/tidytemplate

NeedsCompilation no

Author Kirill Müller [aut, cre] (<https://orcid.org/0000-0002-1416-3412>), R Consortium [fnd], RStudio [fnd]

Maintainer Kirill Müller <kirill@cynkra.com>

Repository CRAN

Date/Publication 2022-08-19 09:30:02 UTC
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 <kirill@cynkra.com> (ORCID)

Other contributors:

• R Consortium [funder]
• RStudio [funder]

See Also

Useful links:

• https://hms.tidyverse.org/
• 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()` is a generic that supports conversions beyond casting. The default method forwards to `vec_cast()`.

Usage

```r
hms(seconds = NULL, minutes = NULL, hours = NULL, days = NULL)
new_hms(x = numeric())
is_hms(x)
as_hms(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.

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

```r
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
```

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")
parse_hms("12:34:56.789")
parse_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**

```r
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**

```r
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

```r
## 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

```r
## 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

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