Package: Waypoint (via r-universe)

October 7, 2026

Type Package

Title Convert, Validate, Format and Print Geographic Coordinates and Waypoints

Version 2.0.2

Date 2026-10-05

Description Convert, validate, format and elegantly print geographic coordinates and waypoints (paired latitude and longitude values) in decimal degrees, degrees and minutes, and degrees, minutes and seconds using high performance C++ code to enable rapid conversion and formatting of large coordinate and waypoint datasets.

License MIT + file LICENSE

Imports methods, Rcpp (>= 1.0.13)

LinkingTo Rcpp

Encoding UTF-8

URL https://mark-eis.github.io/Waypoint/

Depends R (>= 4.5)

Config/roxygen2/version 8.0.0

Repository https://mark-eis.r-universe.dev

Date/Publication 2026-10-06 14:19:46 UTC

RemoteUrl https://github.com/mark-eis/waypoint

RemoteRef HEAD

RemoteSha 80cd01465ee2091badd66f801067dba62f42c08e

Contents

Waypoint-package 2
convert 2
coords 4
Extract 6

format 8
op-zero-length 10
review 11
validate 13
waypoints 15

Index **18**

`Waypoint-package` *Convert, Validate, Format and Print Geographic Coordinates and Waypoints*

Description

Convert, validate, format and elegantly print geographic coordinates and waypoints (paired latitude and longitude values) in decimal degrees, degrees and minutes, and degrees, minutes and seconds using high performance C++ code to enable rapid conversion and formatting of large coordinate and waypoint datasets.

Package Content

Index: This package was not yet installed at build time.

Maintainer

Mark Eisler <mark.eisler@bristol.ac.uk>

Author(s)

Mark Eisler [aut, cre, cph] (ORCID: <https://orcid.org/0000-0001-6843-3345>)

`convert` *Convert the Format of "coords" and "waypoints" Objects*

Description

Convert the format of objects of class `"coords"` or `"waypoints"` between (i) decimal degrees, (ii) degrees and minutes, and (iii) degrees, minutes and seconds.

Usage

```
convert(x, ...)

## S3 method for class 'coords'
convert(x, fmt, ...)

## S3 method for class 'waypoints'
convert(x, fmt, ...)
```

Arguments

x	object of class "coords" created by function as_coords(), or class "waypoints" created by function as_waypoints().
...	further arguments passed to or from other methods.
fmt	integer, 1L, 2L or 3L, specifying the required coordinate format.

Details

The fmt argument should be 1L to convert to *decimal degrees*, 2L, to convert to *degrees and minutes*, and 3L to convert to *degrees, minutes and seconds*. On conversion of a "coords" object, the original argument x is modified to have a decimal point after the number of whole degrees in the case of decimal degrees, after the number of whole minutes in the case of degrees and minutes, and after the number of whole seconds in the case of degrees, minutes and seconds.

Prior to conversion, the "coords" or "waypoints" object to be converted is checked to ensure its values represent valid geographic locations as described under validate().

To format or print "coords" or "waypoints" in another coordinate format without modifying the original object, use format() or print() with the required fmt argument as above.

Value

The original argument x, an object of class "coords" or "waypoints" with values converted as described under *details* and a revised "fmt" attribute reflecting the new format.

See Also

"coords", "waypoints" and validate().

Other coordsandway: coords, waypoints

Examples

```
## Continuing example from `as_coords()`...


## Named "coords" object in degrees and minutes with
## eight values each of latitude and longitude
dm

## Convert to degrees, minutes and seconds (fmt = 3)
convert(dm, 3)

## Convert to decimal degrees (fmt = 1)
convert(dm, 1)

## Show converted values as an ordinary R numeric vector
as.numeric(dm)

###
## Continuing example from `as_waypoints()`...
```

```
## "waypoints" object in degrees, minutes and seconds
wp

## Convert to degrees and minutes (fmt = 2)
convert(wp, 2)

## Convert to decimal degrees (fmt = 1)
convert(wp, 1)

## Show converted values as an ordinary R data frame
as.data.frame(wp)

rm(dm, wp)
```

---

coords *Geographic Coordinate Class*

Description

`as_coords()` creates an object of class `"coords"`, a robust representation of a series of geographic or GPS coordinate values.

`latlon()<-` adds information to objects of class `"coords"` specifying whether individual coordinate values represent latitude or longitude.

Usage

```
as_coords(object, ...)

## Default S3 method:
as_coords(object, ..., fmt = 1L)

latlon(cd) <- value

## S3 method for class 'waypoints'
as_coords(object, which, ...)
```

Arguments

`object`	a `numeric` vector of coordinate values, optionally named, or an object of class `"waypoints"`.
`...`	further arguments passed to or from other methods.
`fmt`	`integer`, `1L`, `2L` or `3L`, specifying the required coordinate format.
`cd`	object of class `"coords"` created by function `as_coords()`.
`value`	a `logical` vector of length 1 or `length(x)`.

which	logical, indicating whether the as_coords() method for class "waypoints" extracts the latitude component of argument object (if TRUE), or the longitude (if FALSE).

Details

Individual values provided in a numeric vector argument object should have a decimal point after the number of whole degrees in the case of *decimal degrees*, after the number of whole minutes in the case of *degrees and minutes*, and after the number of whole seconds in the case of *degrees, minutes and seconds*.

The fmt argument should be 1L to represent decimal degrees, 2L for degrees and minutes, and 3L for degrees, minutes and seconds and is used to provide the format of values in the numeric vector argument object to be converted to class "coords".

The values of a newly created "coords" object are checked to ensure they are valid geographic locations as described under validate().

Individual coordinate values in a Coords object may be specified as representing latitude or longitude using latlon()<-. The value argument may either be a single value, TRUE signifying that all values are latitude, FALSE signifying that all values are longitude, or a logical vector of the same length as as the Coords object signifying whether individual values are latitude or longitude.

Value

as_cords() returns an object of class "coords", comprising a numeric vector argument with additional attributes: –

"class"	the character string "coords".
"fmt"	an integer representing the coordinate format.
"valid"	a logical vector indicating whether individual coordinate values are valid geographic locations.

The as_cords() method for class "coords" returns its numeric vector argument object modified in place, whereas the method for class 'waypoints' returns a new numeric vector.

latlon()<- returns its "coords" argument cd with a logical vector attribute "latlon" added or updated to reflect argument value.

See Also

attr(), attributes, format(<coords>), print(<coords>), validate() and [<-(<coords>).

Other coordsandway: convert(), waypoints

Examples

```
## Numeric vector representing degrees and minutes, with
## the decimal point after the number of whole minutes
dm <- c(5130.4659, 4932.7726, 4806.4339, 3853.3696, 0.0000, -3706.7044, -5306.2869, -2514.4093,
     -007.6754, 1823.9137, -12246.7203, -7702.1145, 0.0000, -1217.3178, 7331.0370, -5731.1536)

## Create an unnamed "coords" object in degrees and minutes (fmt = 2)
```

```
## (Latitude and longitude unspecified)
as_coords(dm, fmt = 2)

## Name the "coords" object
names(dm) <- rep(c("Nelson's Column", "Ostravice", "Tally Ho", "Washington Monument", "Null Island",
             "Tristan da Cunha", "Mawson Peak", "Silvio Pettirossi International Airport"), 2)
dm

## Set all values to represent longitude
## ("latlon" attribute set to FALSE, length 1)
latlon(dm) <- FALSE
dm

## Set eight values each of latitude and longitude
## ("latlon" attribute set to TRUE, n=8, and FALSE, n=8)
latlon(dm) <- rep(TRUE:FALSE, each = 8)
dm

## Show as an ordinary R numeric vector
as.numeric(dm)

rm(dm)
```

`Extract` *Extract or Replace Parts of a Coords Object*

Description

Extract or replace subsets of coords.

Usage

```
## S3 method for class 'coords'
x[i]

## S3 replacement method for class 'coords'
x[i] <- value
```

Arguments

`x`	a "`coords`" object.
`i`	indices specifying elements to extract or replace—see **base** `Extract`.
`value`	a `numeric`, a `numeric` vector of coordinate values of `length(i)`, or a "`coords`" object, possibly named.

Details

Subsetting a "coords" object (except by an empty index) will drop all attributes except fmt, latlon, names and valid. Indices referencing values greater than length(x) will throw a subscript out of bounds error. If names are not required, use unname(), see *examples*.

Replacement values may be a single numeric, a numeric vector of coordinate values of length(i), or a "coords" object, possibly with a "latlon" attribute. However, the "latlon" attribute of the replacement value is ignored if the "coords" object x has no corresponding attribute set. If replacement values are named, the names are also ignored; to replace names, use names<-() replacement form.

The [<-(<coords>) replacement operator automatically revalidates "coords" objects after the replacement operation by invoking validate().

Value

A "coords" object.

Note

To extract and replace subsets of "waypoints" objects, simply use the **base** package [and [<- operators, taking care not to exclude the latitude and longitude columns or "Name" column (if present), which could lead to undefined results. Whereas "coords" objects are automatically revalidated after using the [<-(<coords>) replacement operator, following value replacement using the **base** [<- operator, "waypoints" objects should be revalidated using validate(), see validate *Examples*.

See Also

"coords", **base** Extract, unname(), and validate().

Examples

```
## Continuing example from `as_coords()`...


## Named "coords" object in degrees and minutes with
## eight values each of latitude and longitude
dm

## Extract the first eight values
dm[1:8]

## Exclude the first eight values
dm[-8:0]

## Index odd-numbered values
(index <- as.logical(1:16 %% 2))
dm[index]

## Extract values without names
unname(dm)[1:4]
```

```
## Create "coords" object with updated position of Tally Ho
newpos <- as_coords(c(4930.342, -12411.580), fmt = 2)
latlon(newpos) <- c(TRUE, FALSE)
newpos

## Update position using the "coords" object as replacement value
dm[c(3, 11)] <- newpos
dm[c(3, 11)]

## Or, as latlon didn't actually change, use simple numeric vector
dm[c(3, 11)] <- c(4930.342, -12411.580)
dm[c(3, 11)]

rm(dm, index, newpos)
```

`format` *Format and Print Coords or Waypoints*

Description

Elegantly format and print objects of class "coords" or "waypoints" in alternative coordinate formats.

Usage

```
## S3 method for class 'coords'
print(x, ..., max = NULL)

## S3 method for class 'waypoints'
print(x, ..., fmt = NULL, max = NULL)

## S3 method for class 'coords'
format(x, ..., usenames = TRUE, validate = TRUE, fmt = 0L)

## S3 method for class 'waypoints'
format(x, ..., usenames = TRUE, validate = TRUE, fmt = 0L)

ll_headers(width, fmt)
```

Arguments

x	object of class "coords" created by function as_coords(), or class "waypoints" created by function as_waypoints().
...	further arguments passed to or from other methods.
max	numeric or NULL, specifying the maximal number of entries to be printed. By default, when NULL, getOption("max.print") used.

fmt	integer, 1L, 2L or 3L, specifying the required coordinate format.
usenames	logical, whether or not to include names in formatted output; default TRUE.
validate	logical, whether or not to validate x before formatting; default TRUE.
width	character vector, used to match width of headers to formatted output.

Details

The format() methods for "coords" and "waypoints" objects output elegantly formatted character vector representations of their arguments, which are used by their respective print() methods.

Objects of class "coords" specified in *degrees and minutes* or in *degrees, minutes and seconds* and with a "latlon" attribute, and similarly specified "waypoints" objects are formatted with individual coordinate values followed by a capital letter representing the *cardinal direction* i.e., 'N', 'E', 'S' or 'W'. "coords" objects lacking a "latlon" attribute have formatted values followed by two possible cardinal directions in parentheses i.e., '(N/E)' for positive values and '(S/W)' for negative values. Values of "coords" or "waypoints" objects in *decimal degrees* are formatted prefixed with their sign, if negative; cardinal direction is not shown, but for "coords" objects with a "latlon" attribute, the formatted values are suffixed by either 'lat' or 'lon'.

Prior to formatting and printing, "coords" or "waypoints" objects are checked to ensure that their "valid" attribute (in the case of a "coords" object), or "validlat" and "validlon" attributes (in the case of a "waypoints" object) are present and all TRUE i.e., valid. If these attributes are found to contain any FALSE i.e. invalid values, a warning is issued and similarly, if these attributes are missing, a warning is issued and the objects are re-validated as described under validate().

The optional argument fmt may be used to specify the coordinate format desired for formatting or printing "coords" or "waypoints" objects, see the fmt argument for as_coords() and as_waypoints(); using the default, fmt = 0L, will format or print in the existing coordinate format.

ll_headers() outputs the headings "Latitude ... Longitude" formatted to the width of argument width, adjusted for format fmt and is primarily intended for use by the print() method for class "waypoints". Likewise argument validate is used by the print() methods for classes "coords" and "waypoints" to prevent unnecessary replicate validation and may otherwise be left as the default.

Value

The format() methods for both classes "coords" and "waypoints" return a character vector, respectively of length length(x) or nrow(x), and containing values formatted in decimal degrees, degrees and minutes, or degrees, minutes and seconds as appropriate.

See Also

format(), print(), "coords" and "waypoints".

Examples

```
## Continuing example from `as_coords()`...


## Print named "coords" object in degrees and minutes,
```

```
## implicitly using S3 print() method
dm

## Print explicitly using S3 print() method, specifying
## the maximal number of entries to be printed
print(dm, max = 14)

## Format as a fixed-width character vector,
## with names...
format(dm)

## ...or without them
format(dm, usenames = FALSE)

## Format as decimal degrees,
format(dm, fmt = 1)

###
## Continuing example from `as_waypoints()`...


## Print named "waypoints" object in degrees, minutes and seconds
## implicitly using S3 print() method
wp

## Print explicitly using S3 print() method, specifying
## the maximal number of entries to be printed
print(wp, max = 21)

## Print as degrees and minutes
print(wp, fmt = 2)

## Format as a fixed-width character vector,
## with names...
format(wp)

## ...or without them
format(wp, usenames = FALSE)

rm(dm, wp)
```

`op-zero-length` *Operator Providing Alternative to Zero-Length Object*

Description

Infix function implementing provision of an alternative if an object has zero length.

Usage

```
x %L% y
```

Arguments

x, y	atomic vector arguments or other objects for which `length()` is defined.

Details

The infix function `%L%` may be useful in implementing `if (length(x)) x else y` and was inspired by the null coalescing operator `%||%`.

Value

`x`, or if `length(x)` is zero, `y`.

See Also

`%||%`.

Examples

```
c4 <- letters[1:4]
c0 <- character(0)
n3 <- 1:3
n0 <- numeric(0)

c4 %L% n3
c0 %L% n3

n3 %L% c4
n0 %L% c4

rm(c4, c0, n3, n0)
```

`review` *Review Coordinates and Waypoints Validity*

Description

Review validity of elements of `"coords"` and `"waypoints"` objects.

Usage

```
review(x, ...)

## S3 method for class 'coords'
review(x, ..., show_n = 20L)

## S3 method for class 'waypoints'
review(x, ..., show_n = 20L)
```

Arguments

x	object of class "coords" created by function as_coords(), or class "waypoints" created by function as_waypoints().
...	further arguments passed to or from other methods.
show_n	integer, the maximum number of invalid elements of argument x to include in the output; default 20L.

Details

review() reveals elements of "coords" and "waypoints" objects that do not conform to the criteria checked by validate(), i.e. are not valid geographic locations.

Value

The review() method for class "coords" returns a list comprising the following elements: -

allvalid	logical, whether or not all the elements of argument x are valid.
n_invalid	integer, the number of invalid elements in argument x, if any.
invalids	numeric vector including invalid elements of argument x, if any.
which_invalid	integer vector specifying which elements of argument x are invalid, if any.

The method for class "waypoints" returns a list of two sub-lists, each sub-list with elements as described above for the method for class "coords", one each for latitude and longitude.

See Also

"coords" and "waypoints".

Other validate: validate()

Examples

```
## Continuing example from `validate()`...


## Review "coords" object in degrees and minutes, having
## an erroneous first value of more than 60 minutes
review(dm)
```

```
###
## Continuing example from `validate()`...


## Review "waypoints" object in  decimal degrees, having an erroneous
## penultimate latitude absolute value greater than 90 degrees
review(wp)

rm(dm, wp)
```

---

validate *Validate Coords or Waypoints*

Description

Validate objects of class "coords" or "waypoints" as geographic locations.

Usage

```
validate(x, ...)

## S3 method for class 'coords'
validate(x, ..., force = TRUE)

## S3 method for class 'waypoints'
validate(x, ..., force = TRUE)
```

Arguments

x	object of class "coords" created by function as_coords(), or class "waypoints" created by function as_waypoints().
...	further arguments passed to or from other methods.
force	logical signifying whether, if TRUE, to perform full *de novo* revalidation or, if FALSE, simply check existing "valid" attribute in the case of a "coords" object, or "validlat" and "validlon" attributes in the case of a "waypoints" object and only revalidate if any of these are missing; default TRUE.

Details

Individual coordinate values within "coords" or "waypoints" objects are checked to ensure they represent valid geographic locations.

To be valid, the absolute values of coordinates in degrees must not exceed 180°, or 90° if degrees of latitude and, similarly, the absolute values of the minutes and seconds components, where given, must not exceed 60. Otherwise, a warning will be issued and the "valid" attribute in the case of a "coords" object, or "validlat" and "validlon" attributes in the case of a "waypoints" object will be set to FALSE for any non-compliant coordinate values.

Argument force is primarily intended for use by the print() methods for classes "coords" and "waypoints" and should otherwise left as the default value TRUE.

Value

validate() returns its argument with logical vector attribute "valid", or attributes "validlat" and "validlon" updated as appropriate for "coords" and' "waypoints" objects respectively.

See Also

"coords", "waypoints" and [<-(<coords>).

Other validate: review()

Examples

```
## Continuing example from `as_coords()`...


## Validate "coords" object in degrees and minutes
validate(dm)

## Deliberately change the first coordinate to a value greater
## than 60 minutes using the `[<-(<coords>)` replacement operator,
## which itself invokes `validate(dm)` -- see help(`[<-.coords`)
dm[1] <- 5160.4659

dm

## Examine "valid" attribute of dm
attr(dm, "valid")

###
## Continuing second example from `as_waypoints()`...


## Validate "waypoints" object in decimal degrees

validate(wp)

## Deliberately change the penultimate latitude to an absolute
## value greater than 90 degrees, using the usual base package
## replacement `[<-` operator, which does not invoke `validate(dm)`
## -- we must do so ourselves
wp$lat[7] <- -93.104781

validate(wp)

## Examine "validlat" attribute of wp
attr(wp, "validlat")

rm(dm, wp)
```

waypoints *Geographic Waypoint Class*

Description

`as_waypoints()` creates an object of class `"waypoints"`, a robust representation of a series of geographic or GPS waypoints of paired latitude and longitude values.

Usage

```
as_waypoints(object, ...)

## Default S3 method:
as_waypoints(object, ..., fmt = 1L)
```

Arguments

`object`	a data frame with each row representing a waypoint, comprising at least two `numeric` columns containing values of latitude and longitude, and optionally a `character` column of waypoint names (see *Details*).
`...`	further arguments passed to or from other methods.
`fmt`	`integer`, `1L`, `2L` or `3L`, specifying the required coordinate format.

Details

Data frame argument `object` should have `numeric` vector latitude and longitude columns with individual values having a decimal point after the number of whole degrees in the case of *decimal degrees*, after the number of whole minutes in the case of *degrees and minutes*, and after the number of whole seconds in the case of *degrees, minutes and seconds*. These should be the first two columns of the data frame, or the second and third columns if the first column contains waypoints names (see below). Alternative columns may be specified for the latitude and longitude by setting `"llcols"` as a length 2 `integer` vector attribute of `object` indicating their positions in the data frame.

The `fmt` argument should be `1L` to represent decimal degrees, `2L` for degrees and minutes, and `3L` for degrees, minutes and seconds and is used to provide the format of values in data frame argument `object` to be converted to class `"waypoints"`.

If the waypoints have names, these should be included in a "Name" column of data frame argument `object`, by default immediately before (on the left-hand side of) the latitude and longitude columns. An alternative column for waypoint names may be specified by setting an `integer` `attribute` named "namescol" indicating its position in `object`. If neither a "Name" column nor a `"namescol"` attribute is present in `object`, `row.names` are used for waypoint names.

The latitude and longitude values of a newly created `"waypoints"` object are checked to ensure they are valid geographic locations as described under `validate()`.

Value

An object of classes "waypoints" and "data.frame", comprising the original data frame argument object, with additional attributes: –

"class"	the character string "waypoints".
"fmt"	an integer indicating the coordinate format.
"namescol"	an integer indicating the position of a waypoint names column, if present.
"llcols"	a length 2 integer vector indicating the positions of latitude and longitude columns.
"validlat" and "validlon"	logical vectors indicating whether individual latitude and longitude values are valid geographic locations.

See Also

attr(), data.frame() format(<waypoints>), print(<waypoints>), and validate().

Other coordsandway: convert(), coords

Examples

```
## Dataframe representing waypoint names, and latitude and longitude values
## in degrees, minutes and seconds
wp1 <- data.frame(
   name = c("Nelson's Column", "Ostravice", "Tally Ho", "Washington Monument", "Null Island",
            "Tristan da Cunha", "Mawson Peak", "Silvio Pettirossi International Airport"),
   lat = c(513027.95, 493246.36, 480626.04, 385322.18, 0, -370642.26, -530617.21, -251424.56),
   lon = c(-00740.53, 182354.82, -1224643.22, -770206.87, 0, -121719.07, 733102.22, -573109.21)
)

## Create "waypoints" object in degrees, minutes and seconds (fmt = 3)
as_waypoints(wp1, fmt = 3)

## Show as an ordinary R data frame
as.data.frame(wp1)

###

## Dataframe representing unnamed latitude and longitude
## values in decimal degrees
wp2 <- data.frame(
   lat = c(51.507765, 49.54621, 48.107232, 38.889494, 0, -37.11174, -53.104781, -25.240156),
   lon = c(-0.127924, 18.398562, -122.778671, -77.035242, 0, -12.28863, 73.517283, -57.519227)
)

## Create unnamed "waypoints" object in decimal degrees (default fmt = 1)
as_waypoints(wp2)

## Add waypoint names as row.names
row.names(wp2) <-
    c("Nelson's Column", "Ostravice", "Tally Ho", "Washington Monument", "Null Island",
```

```
      "Tristan da Cunha", "Mawson Peak", "Silvio Pettirossi International Airport")

wp2

## Show as an ordinary R data frame
as.data.frame(wp2)

rm(wp1, wp2)
```

Index

∗ **character**
  format, 8
∗ **coordsandway**
  convert, 2
  coords, 4
  waypoints, 15
∗ **extract**
  Extract, 6
∗ **logic**
  op-zero-length, 10
∗ **package**
  Waypoint-package, 2
∗ **print**
  format, 8
∗ **programming**
  convert, 2
  op-zero-length, 10
∗ **utils**
  op-zero-length, 10
∗ **validate**
  review, 11
  validate, 13
[, *7*
[.coords *(*Extract*)*, 6
[<-.coords *(*Extract*)*, 6
%L% *(*op-zero-length*)*, 10
%, 11

as_coords, *3, 4*, *8, 9*, *12, 13*
as_coords *(*coords*)*, 4
as_waypoints, *3*, *8, 9*, *12, 13*
as_waypoints *(*waypoints*)*, 15
attr, *5*, *16*
attribute, *15*
attributes, *5*

convert, 2
convert(), *5*, *16*
coords, *3*, 4, *6–9*, *12–14*, *16*

data.frame, *16*

Extract, 6, *6, 7*

format, *3*, *5*, 8, *9*, *16*

getOption, *8*

latlon<- *(*coords*)*, 4
list, *12*
ll_headers *(*format*)*, 8

op-zero-length, 10

print, *3*, *5*, *9*, *16*
print.coords *(*format*)*, 8
print.waypoints *(*format*)*, 8

review, 11
review(), *14*
row.names, *15*

unname, *7*

validate, *3*, *5*, *7*, *9*, *12*, 13, *15, 16*
validate(), *12*

Waypoint *(*Waypoint-package*)*, 2
Waypoint-package, 2
waypoints, *3*, *5*, *7–9*, *12–14*, 15