Package 'activatr'

July 22, 2025

Type Package

Title Utilities for Parsing and Plotting Activities

Version 0.2.1

Description This contains helpful functions for parsing, managing, plotting, and visualizing activities, most often from GPX (GPS Exchange Format) files recorded by GPS devices. It allows easy parsing of the source files into standard R data formats, along with functions to compute derived data for the activity, and to plot the activity in a variety of ways.

License MIT + file LICENSE

URL https://github.com/dschafer/activatr, https://dschafer.github.io/activatr/

BugReports https://github.com/dschafer/activatr/issues

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports dplyr (>= 1.0.0), geosphere (>= 1.5), ggmap (>= 3.0.0), glue (>= 1.4.0), httr (>= 1.4.0), lubridate (>= 1.7.0), rlang (>= 0.4.0), tibble (>= 3.0.0), slider (>= 0.3.0), xml2 (>= 1.3.2)

RoxygenNote 7.2.3

Suggests covr (>= 3.5.0), ggplot2 (>= 3.4.0), knitr (>= 1.30), mockery (>= 0.4.2), rmarkdown (>= 2.6), roxygen2 (>= 7.1.0), sf (>= 1.0), testthat (>= 3.0.0)

VignetteBuilder knitr

NeedsCompilation no

Author Daniel Schafer [aut, cph, cre]

Maintainer Daniel Schafer <dan.schafer@gmail.com>

Repository CRAN

Date/Publication 2024-07-27 21:20:02 UTC

Contents

`act_tbl-class` `act_tbl` *class*

Description

The `act_tbl` S3 class is a subclass of `data.frame` and `tibble`.

Details

In nearly every respect, it can be treated like a tibble; however, this allows the package to provide an improved `summary.act_tbl()` function to get an overview of the activity.

`get_ggmap_from_df` *Get a map for a given* `act_tbl`

Description

`get_ggmap_from_df` takes an `act_tbl` object, computes the correct zoom and center for that activity, then returns a ggmap object for that zoom and center.

Usage

```
get_ggmap_from_df(df, ...)
```

Arguments

`df`	An `act_tbl` object.
`...`	Additional arguments forwarded to `ggmap::get_googlemap()`.

Details

Note that since this calls `ggmap::get_googlemap()`, you must have previously called `ggmap::register_google()` to register an API key.

Value

A ggmap object, the result of calling `ggmap::get_googlemap()`, with the correct center and size to include the entire activity represented by the `act_tbl`.

See Also

`ggmap::get_googlemap()`

Examples

```
## Not run:
example_gpx_file <- system.file(
  "extdata",
  "running_example.gpx.gz",
  package = "activatr"
)
act_tbl <- parse_gpx(example_gpx_file)
ggmap::ggmap(get_ggmap_from_df(act_tbl))

## End(Not run)
```

`localize_to_time_zone` *Localize time zone values*

Description

`localize_to_time_zone` uses Google Maps Time Zone APIs to localize the time zone in an `act_tbl`. This modifies a mutated `act_tbl` with the time column updated to contain the same absolute time, but with the appropriate time zone for where the activity took place.

Usage

```
localize_to_time_zone(df)
```

Arguments

`df`	An `act_tbl` object.

Details

Note that to avoid overuse of the APIs, this does an "approximation", in that it finds the correct time zone for the first point in the data frame, and assumes all points in that data frame use that time zone. Runs between time zones (or runs that cross daylight savings time shifts) will hence be recorded using a consistent, but not always pointwise correct, timezone.

Note that you must have previously called `ggmap::register_google()` to register an API key before calling this.

Value

That same act_tbl, but with the time column updated to be in the local time zone rather than UTC.

Examples

```
## Not run:
example_gpx_file <- system.file(
  "extdata",
  "running_example.gpx.gz",
  package = "activatr"
)
act_tbl <- parse_gpx(example_gpx_file)
act_tbl_with_tz <- localize_to_time_zone(act_tbl)

## End(Not run)
```

---

mutate_with_speed *Augments an act_tbl with a speed column*

Description

This returns a mutated act_tbl with a new column representing speed, in meters per second. See vignette("pace") for examples.

Usage

```
mutate_with_speed(df, method = c("2D", "3D"), lead = 0, lag = 1)
```

Arguments

df	An act_tbl object
method	If "2D" (default), ignores elevation. If "3D", includes elevation. "3D" is not often necessary, but for skiing activities is likely to yield a more accurate value.
lead	How far ahead to look for the "end" point.
lag	How far behind to look for the "start" point.

Details

The speed is determined by looking at the time difference between the current point and the previous point: hence, it is always NA for the first row in the data frame.

The lead and lag values are helpful to get "smoother" values, especially if the provided activity file has GPS errors in it.

Value

That same act_tbl, but with a new speed column, in meters per second.

Examples

```
example_gpx_file <- system.file(
  "extdata",
  "running_example.gpx.gz",
  package = "activatr"
)
example_act_tbl <- parse_gpx(example_gpx_file)
example_act_tbl_with_speed <- mutate_with_speed(example_act_tbl)
example_act_tbl_with_speed
```

`pace_formatter` *Format pace durations*

Description

`pace_formatter` takes a pace duration and returns a formatted string.

Usage

```
pace_formatter(pace)
```

Arguments

`pace`	A lubridate duration, returned by `lubridate::duration` or other methods in that family.

Details

This is most useful when plotting pace as one of the axes in a graph; rather than having the "number of seconds" as the axis value, this method can convert that to a more readable format.

Most commonly, using something like `ggplot2::scale_y_reverse(label = pace_formatter)` will ensure the y-axis goes from "slowest" to "fastest", and shows paces like "8:30" rather than "510"

Value

A formatted string representing the pace.

Examples

```
pace_formatter(lubridate::dseconds(380))
pace_formatter(lubridate::dseconds(510))
pace_formatter(lubridate::dseconds(680))
```

parse_gpx *Parses a GPX file into a* act_tbl

Description

This parses a standard GPS Exchange Format XML (GPX) file into an data frame with class act_tbl. See vignette("parsing") for examples.

Usage

```
parse_gpx(filename, detail = c("basic", "latlon", "advanced"), every = NA)
```

Arguments

filename	The GPX file to parse
detail	How much detail to parse from the GPX. • If basic (the default), this will parse lat / lon / ele / time columns. • If latlon, this will only parse lat/lon. This is particularly useful for GPX files exported without time information, such as from Strava. • If advanced, it will load everything from basic, plus hr / cad. This is most useful for files that have heart rate and cadence information.
every	Optional. If provided, determines how frequently points will be sampled from the file, so if 10 is provided, every tenth point will be selected. If omitted or set to 1, every point will be selected. Must be a positive integer. This is most useful to quickly analyze a large file, since parsing is much faster when skipping 90% of the data points.

Value

A act_tbl with one row for each trackpoint in the . GPX (modified by every), and with the columns determined by detail.

lat	Latitude, a double in degrees between -90 and 90.
lon	Longitude, a double in degrees between -180 and 180.
ele	Elevation, a double in meters.
time	A date-time representing the time of the point.
hr	Heart rate, an int in beats per minute.
cad	Cadence, an int in one-foot steps per minute.

Additionally, attributes are set on the returned object containing top level data from the GPX. Each of these will be NA when not provided in the file.

filename	The filename this was parsed from, a string. This is always present, and is always the value of the filename parameter.

time	A date-time representing the time of the activity.
title	A string.
desc	A string.
type	A string.

See Also

https://en.wikipedia.org/wiki/GPS_Exchange_Format

https://www.topografix.com/gpx.asp

Examples

```
example_gpx_file <- system.file(
  "extdata",
  "running_example.gpx.gz",
  package = "activatr"
)
act_tbl <- parse_gpx(example_gpx_file)
print(act_tbl, n = 5)
attr(act_tbl, "title")

nrow(parse_gpx(example_gpx_file))
nrow(parse_gpx(example_gpx_file, every = 100))

colnames(parse_gpx(example_gpx_file))
colnames(parse_gpx(example_gpx_file, detail = "latlon"))
colnames(parse_gpx(example_gpx_file, detail = "advanced"))
```

---

parse_tcx *Parses a TCX file into a act_tbl*

Description

This parses a standard Training Center XML (TCX) file into a data frame with class act_tbl. See vignette("parsing") for examples.

Usage

```
parse_tcx(filename, detail = c("basic", "latlon", "advanced"), every = NA)
```

Arguments

filename	The TCX file to parse
detail	How much detail to parse from the TCX • If basic (the default), this will parse lat / lon / ele / time columns. • If latlon, this will only parse lat/lon. This is particularly useful for TCX files exported without time information, such as from Strava.

- If advanced, it will load everything from basic, plus hr / cad. This is most useful for files that have heart rate and cadence information.

every — Optional. If provided, determines how frequently points will be sampled from the file, so if 10 is provided, every tenth point will be selected. If omitted or set to 1, every point will be selected. Must be a positive integer.

This is most useful to quickly analyze a large file, since parsing is much faster when skipping 90% of the data points.

Value

A act_tbl with one row for each trackpoint in the TCX (modified by every), and with the columns determined by detail.

lat	Latitude, a double in degrees between -90 and 90.
lon	Longitude, a double in degrees between -180 and 180.
ele	Elevation, a double in meters.
time	A date-time representing the time of the point.
hr	Heart rate, an int in beats per minute.
cad	Cadence, an int in one-foot steps per minute.

Additionally, attributes are set on the tibble containing top level data from the TCX. Each of these will be NA when not provided in the file.

filename	The filename this was parsed from, a string. This is always present, and is always the value of the filename parameter.
time	A date-time representing the time of the activity.
type	A string.

See Also

https://en.wikipedia.org/wiki/Training_Center_XML

Examples

```
example_tcx_file <- system.file(
  "extdata",
  "running_example.tcx.gz",
  package = "activatr"
)
act_tbl <- parse_tcx(example_tcx_file)
print(act_tbl, n = 5)
attr(act_tbl, "title")

nrow(parse_tcx(example_tcx_file))
nrow(parse_tcx(example_tcx_file, every = 100))

colnames(parse_tcx(example_tcx_file))
colnames(parse_tcx(example_tcx_file, detail = "latlon"))
colnames(parse_tcx(example_tcx_file, detail = "advanced"))
```

speed_to_mile_pace *Convert speed to mile pace*

Description

speed_to_mile_pace converts a speed (in meters per second) to a mile pace. This method is vectorized, so it works on a column in a data frame. This is most useful after calling mutate_with_speed(), to convert that speed to the more-commonly-used pace. See vignette("pace") for examples.

Usage

```
speed_to_mile_pace(speed)
```

Arguments

speed	A vector of doubles representing speed in meters per second, as from mutate_with_speed().

Value

A corresponding vector of lubridate::duration values, representing the mile pace.

Examples

```
speed_to_mile_pace(3)
speed_to_mile_pace(1)
```

summary.act_tbl *Summarizes* act_tbl *objects.*

Description

summary.act_tbl returns a tibble with canonical information about the activity.

Usage

```
## S3 method for class 'act_tbl'
summary(object, full = FALSE, units = c("imperial", "metric"), ...)
```

Arguments

object	an object for which a summary is desired
full	Whether every column should be included, and filled with NA if missing. Most useful to ensure the tibble has the same shape for every file, allowing eventual use of dplyr::bind_rows() or purrr::map_dfr() to create a full summary data set.
units	Which units should be used?

- "imperial" returns distance in miles, pace in minutes per mile, and elevation in feet.
- "metric" returns distance in kilometers, pace in minutes per kilometer, and elevation in meters.

... Additional arguments.

Details

This is designed to allow for easy creation of activity summary data sets by mapping summary over each act_tbl then using dplyr::bind_rows(), purrr::map_dfr(), or equivalent to create a complete data set.

Value

Returns a tibble with a single row, containing a summary of the given act_tbl.

Examples

```
example_gpx_file <- system.file(
  "extdata",
  "running_example.gpx.gz",
  package = "activatr"
)
act_tbl <- parse_gpx(example_gpx_file)
summary(act_tbl)

## Not run:
files <- list.files("path/to/many/files", pattern = "*.gpx")
gpxs <- files |> purrr::map(\(f) parse_gpx(f))
summaries <- gpxs |> purrr::map_dfr(\(g) summary(g, full = TRUE))

## End(Not run)
```

Index