Package 'f1dataR'

March 27, 2025

Title Access Formula 1 Data

Version 2.0.1

Description
Obtain Formula 1 data via the 'Jolpica API' <https://jolpi.ca> and the unofficial API <https://www.formula1.com/en/timing/f1-live> via the 'fastf1' 'Python' library <https://docs.fastf1.dev/>.

Config/reticulate list(packages = list(list(package = ``fastf1", pip = TRUE)))

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 3.5.0), reticulate (>= 1.14),

Imports glue, magrittr, tibble, jsonlite, httr2, memoise, janitor, dplyr, tidyr, rlang, lifecycle, cli, rappdirs, cachem, withr

Suggests ggplot2, httptest2, knitr, rmarkdown, testthat (>= 3.0.0),

VignetteBuilder knitr

URL https://scasanova.github.io/f1dataR/, https://github.com/SCasanova/f1dataR

BugReports https://github.com/SCasanova/f1dataR/issues

Config/testthat/edition 3

NeedsCompilation no

Author Santiago Casanova [aut, cre, cph], Philip Bulsink [aut] (<https://orcid.org/0000-0001-9668-2429>)

Maintainer Santiago Casanova <santiago.casanova@yahoo.com>

Repository CRAN

Date/Publication 2025-03-27 04:00:07 UTC

Contents

`change_cache` *Change Caching Settings*

Description

Change caching settings for the package. By default, the cache will be set to keep the results of function calls in memory to reduce the number of requests made to online services for the same data. However, if preferred, the cache can be set to a file directory to make the results persist between sessions.

This is a particularly good idea if you're using functions like `load_driver_telemetry()`, `load_session_laps()`, `load_race_session()` or `plot_fastest()` as they take significant time and download large amounts of data each time you run the function.

If preferred for testing or waiting for data updates on race weekends, you may wish to set the cache to `'off'` instead.

Changes to cache can be made for the session (mark the argument `persist` as `FALSE`) or apply to the next session(s) by setting `persist` to `TRUE`

Usage

```
change_cache(cache = "memory", create_dir = FALSE, persist = FALSE)
```

Arguments

cache	One of 'memory', 'filesystem', 'off' or a directory. If the selection is 'filesystem' the package will automatically write the cache to the operating system's default location for permanent or temporary caches (see persist)
create_dir	Whether to create the directory if it doesn't already exist if a path cache directory is provided. By default this doesn't occur for provided cache paths, but will always happen if the cache choice is set to 'filesystem'.
persist	Whether to make this change permanent (TRUE) or a temporary cache change only (default, FALSE). Note if you set cache to 'off' and persist to TRUE the existing cache will be cleared by calling clear_cache(). If filesystem is chosen for cache and persist is set to TRUE, then a cache directory will be placed in the default location for the operating system. If instead persist is set to FALSE, then a temporary directory will be used instead, and this will be removed at the end of the session. This essentially has the same effect as having cache set to 'memory'.

Value

No return, called for side effects

Examples

```
## Not run:
change_cache("~/f1dataRcache", create_dir = TRUE)

change_cache("off", persist = FALSE)

## End(Not run)
```

---

clear_f1_cache *Clear f1dataR Cache*

Description

Clears the cache for f1dataR telemetry and Jolpica API results. Note that the cache directory can be set by setting option(f1dataR.cache = [cache dir]), but the default is a temporary directory.

You can also call the alias clear_cache() for the same result

Usage

```
clear_f1_cache()

clear_cache()
```

Value

No return value, called to erase cached data

Examples

```
## Not run:
clear_f1_cache()

## End(Not run)
```

`correct_track_ratio` *Correct Track Ratios*

Description

Correct Track Ratios helps ensure that ggplot objects are plotted with 1:1 unit ratio. Without this function, plots have different x & y ratios and the tracks come out misshapen. This is particularly evident at long tracks like Saudi Arabia or Canada.

Note that this leaves the plot object on a dark background, any plot borders will be maintained

Usage

```
correct_track_ratio(trackplot, x = "x", y = "y", background = "grey10")
```

Arguments

`trackplot`	A GGPlot object, ideally showing a track layout for ratio correction
`x, y`	Names of columns in the original data used for the plot's x and y values. Defaults to 'x' and 'y'
`background`	Background colour to use for filling out the plot edges. Defaults to `"grey10"` which is the default background colour if you use `theme_dark_f1()` to theme your plots.

Value

a ggplot object with `ggplot2::scale_x_continuous()` and `ggplot2::scale_y_continuous()` set to the same limits to produce an image with shared x and y limits and with `ggplot2::coord_fixed()` set.

Examples

```
## Not run:
# Note that plot_fastest plots have already been ratio corrected
fast_plot <- plot_fastest(season = 2022, round = 1, session = "Q", driver = V)
correct_track_ratio(fast_plot)

## End(Not run)
```

`driver_team_lookup` *Driver & Team Look-ups*

Description

These functions provide the ability to look-up drivers or teams (and match the two) for given races or seasons.

`get_driver_abbreviation()` looks up the driver abbreviation (typically 3 letters) as used in the provided season.

`get_team_name()` looks up the officially recorded team name based on fuzzy matching to the supplied string. This is fairly inconsistent, for example, "Haas" is recorded as "Haas F1 Team", but not all sponsor names are recorded nor are all names indicating 'F1 Team' – "RB" is recorded as "RB" and not "Visa Cash App RB F1 Team". If `short = TRUE` then a short form for the team is provided ("Haas" instead of "Haas F1 Team").

`get_driver_name()` looks up a driver's full name based on fuzzy matching to the supplied string. The driver has to have participated in the session (season, round, session) for this to match properly. For full-time drivers this is easy, but for rookies who do test FP1 this is a more important note.

`get_drivers_by_team()` looks up a team's drivers for the provided race session (season, round, session). If looking for practice rookies, they typically participate in `session = FP1`.

`get_team_by_driver()` looks up the team for the specified driver (at the specified race event).

`get_session_drivers_and_teams()` returns a data frame of all drivers and their team for a provided session.

Usage

```
get_driver_abbreviation(
  driver_name,
  season = get_current_season(),
  round = 1,
  session = "R"
)

get_driver_name(
  driver_name,
  season = get_current_season(),
  round = 1,
  session = "R"
)

get_team_name(team_name, season = get_current_season(), short = FALSE)

get_drivers_by_team(
  team_name,
  season = get_current_season(),
  round = 1,
```

```
  session = "R"
)

get_team_by_driver(
  driver_name,
  season = get_current_season(),
  round = 1,
  short = FALSE
)

get_session_drivers_and_teams(season, round, session = "R")
```

Arguments

driver_name	Driver name (or unique part thereof) to look up.
season	The season for which the look-up should occur. Should be a number from 2018 to current season. Defaults to current season.
round	number from 1 to 24 (depending on season selected) and defaults to most recent. Also accepts race name.
session	the code for the session to load. Options are 'FP1', 'FP2', 'FP3', 'Q', 'S', 'SS','SQ', and 'R'. Default is 'R', which refers to Race.
team_name	The team name (as a string) to use for lookup.
short	whether to provide a shortened version of the team name. Default False.

Value

for get_session_drivers_and_teams() a data.frame, for get_drivers_by_team() a unnamed character vector with all drivers for the requested team, for all other functions a character result with the requested value.

---

get_aesthetics *Get Aesthetics*

Description

Various aesthetics can be retrieved for a driver or team for a specific session/event.

get_driver_style() gets the FastF1 style for a driver for a session - this includes team colour and line/marker style which should be reasonably (but not guaranteed) consistent across a season. Based on FastF1's get_driver_style.

get_driver_color() and its alias get_driver_colour() return a hexidecimal RGB colour code for a driver at a given season & race. Note that, in contrast to earlier versions, both drivers for a team will be provided the same color. Use get_driver_style() to develop a unique marker/linestyle for each driver in a team. Data is provided by the python FastF1 package.

get_driver_color_mapping() and its alias get_driver_colour_mapping() return a data.frame of driver short-codes and their hexidecimal colour. Like get_driver_color(), both drivers on a

team will get the same colour returned. Data is provided by the python FastF1 package. Requires provision of a specific race event (season/round/session).

`get_team_color()` and its alias `get_team_colour()` return a hexidecimal RGB colour code for a a team at a given season & race. Data is provided by the python FastF1 package.

Usage

```
get_driver_style(driver, season = get_current_season(), round = 1)

get_driver_color(driver, season = get_current_season(), round = 1)

get_driver_colour(driver, season = get_current_season(), round = 1)

get_team_color(team, season = get_current_season(), round = 1)

get_team_colour(team, season = get_current_season(), round = 1)

get_driver_color_map(season = get_current_season(), round = 1, session = "R")

get_driver_colour_map(season = get_current_season(), round = 1, session = "R")
```

Arguments

`driver`	Driver abbreviation or name (FastF1 performs a fuzzy-match to ambiguous strings).
`season`	A season corresponding to the race being referenced for collecting colour/style. Should be a number from 2018 to current season. Defaults to current season.
`round`	A round corresponding to the race being referenced for collecting colour/style. Should be a string name or a number from 1 to the number of rounds in the season and defaults to 1.
`team`	Team abbreviation or name (FastF1 performs a fuzzy-match to ambiguous strings).
`session`	the code for the session to load. Options are `'FP1'`, `'FP2'`, `'FP3'`, `'Q'`, `'S'`, `'SS'`,`'SQ'`, and `'R'`. Default is `'R'`, which refers to Race.

Value

for `get_driver_style()` a named list of graphic parameters for the provided driver, plus the driver identifier provided and the official abbreviation matched to that driver (names are `linestyle`, `marker`, `color`, `driver`, `abbreviation`).

for `get_driver_color()` and `get_team_color()`, a hexidecimal RGB color value.

Examples

```
if (interactive()) {
  # To get a specific season/race, specify them.
  get_driver_style(driver = "ALO", season = 2024, round = 3)

  # For drivers who haven't moved around recently, get their current season's style:
  get_driver_style(driver = "LEC")
```

```
  # Get all driver abbreviations and colors quickly:
  get_driver_color_mapping(season = 2023, round = "Montreal", session = "R")

  get_team_color(team = "Alpine", season = 2023, round = 1)
}
```

`get_current_season` *Get Current Season*

Description

Determines current season by System Date. Note returns the season prior to the current year in January and February

Usage

```
get_current_season()
```

Value

Year (four digit number) representation of current season, as numeric.

`get_fastf1_version` *Get current FastF1 version*

Description

Gets the current installed FastF1 version available (via `reticulate`) to the function. Displays a note if significantly out of date.

Usage

```
get_fastf1_version()
```

Value

version as class `package_version`

get_tire_compounds *Get Tire Compounds*

Description

Get a data.frame of all tire compound names and associated colours for a season.

Usage

```
get_tire_compounds(season = get_current_season())
```

Arguments

season	number from 2018 to current season. Defaults to current season.

Value

a data.frame with two columns: `compound` and `color`

Examples

```
if (interactive()) {
  # To get this season's tires
  get_tire_compounds()

  # Compare to 2018 tires:
  get_tire_compounds(2018)
}
```

load_circuits *Load Circuit Info*

Description

Loads circuit info for all circuits in a given season. Use `.load_circuits()` for an uncached version of this function

Usage

```
load_circuits(season = get_current_season())
```

Arguments

season	number from 1950 to current season (defaults to current season).

Value

A tibble with one row per circuit

load_circuit_details *Load Circuit Information*

Description

Loads circuit details for a specific race session. Note that different track layouts are used at some circuits depending on the year of the race.

Useful for visualizing or annotating data. Contains information on corners, marshal_lights and marshal_sectors.

Each set of these track marker types is returned as a tibble.

Also returns an angle (in degrees) to indicate the amount of rotation of the telemetry to visually align the two.

More information on the data provided (and uses) can be seen at https://docs.fastf1.dev/circuit_info.html#fastf1.mvapi.Circuit

Note that this is an exposition of FastF1 data. As such, caching is recommended (and default behavior). Cache directory can be set by setting `option(f1dataR.cache = [cache dir])`, default is the current working directory.

If you have trouble with errors mentioning 'fastf1' or 'get_fastf1_version()' read the 'Setup FastF1 Connection vignette (run `vignette('setup_fastf1', 'f1dataR')`).

Usage

```
load_circuit_details(
  season = get_current_season(),
  round = 1,
  log_level = "WARNING"
)
```

Arguments

season	number from 2018 to current season. Defaults to current season.
round	number from 1 to 23 (depending on season selected). Also accepts race name.
log_level	Detail of logging from fastf1 to be displayed. Choice of: 'DEBUG', 'INFO', 'WARNING', 'ERROR' and 'CRITICAL'. See fastf1 documentation.

Value

A list of tibbles containing corner number, marshall post number, or marshall segment, plus a numeric value for rotational offset of the data compared to telemetry data.

The tibbles all have the following structure: `x` and `y` specify the position on the track map `number` is the number of the corner. Letter is optionally used to differentiate corners with the same number on some circuits, e.g. "2A". `angle` is an angle in degrees, used to visually offset the marker's placement on a track map in a logical direction (usually orthogonal to the track). `distance` is the location of the marker as a distance from the start/finish line.

load_constructors *Load Constructor Info*

Description

Loads constructor info for all participants in a given season. Use `.load_constructors()`for an uncached version of this function

Usage

```
load_constructors()
```

Value

A tibble with one row per constructor

load_drivers *Load Driver Info*

Description

Loads driver info for all participants in a given season. Use `.load_drivers()` for an uncached version of this function.

Usage

```
load_drivers(season = get_current_season())
```

Arguments

`season`	number from 1950 to current season (defaults to current season).

Value

A tibble with columns driver_id (unique and recurring), first name, last name, nationality, date of birth (yyyy-mm-dd format), driver code, and permanent number (for post-2014 drivers).

`load_driver_telemetry` *Load Telemetry Data for a Driver*

Description

Receives season, race number, driver code, and an optional fastest lap only argument to output car telemetry for the selected situation. Example usage of this code can be seen in the Introduction vignette (run `vignette('introduction', 'f1dataR')` to read). Multiple drivers' telemetry can be appended to one data frame by the user.

If you have trouble with errors mentioning 'fastf1' or 'get_fastf1_version()' read the "Setup FastF1 Connection" vignette (run `vignette('setup_fastf1', 'f1dataR')`).

Usage

```
load_driver_telemetry(
  season = get_current_season(),
  round = 1,
  session = "R",
  driver,
  laps = "fastest",
  log_level = "WARNING",
  race = lifecycle::deprecated(),
  fastest_only = lifecycle::deprecated()
)
```

Arguments

`season`	number from 2018 to current season (defaults to current season).
`round`	number from 1 to 23 (depending on season selected). Also accepts race name.
`session`	the code for the session to load Options are `'FP1'`, `'FP2'`, `'FP3'`, `'Q'`, `'S'`, `'SS'`, `'SQ'`, and `'R'`. Default is `'R'`, which refers to Race.
`driver`	three letter driver code (see `load_drivers()` for a list)
`laps`	which lap's telemetry to return. One of an integer lap number (<= total laps in the race), `fastest`, or `all`. Note that integer lap choice requires `fastf1` version 3.0 or greater.
`log_level`	Detail of logging from fastf1 to be displayed. Choice of: `'DEBUG'`, `'INFO'`, `'WARNING'`, `'ERROR'` and `'CRITICAL'`. See fastf1 documentation.
`race`	**[Deprecated]** `race` is no longer supported, use `round`.
`fastest_only`	**[Deprecated]** `fastest_only` is no longer supported, indicated preferred laps in `laps`.

Value

A tibble with telemetry data for selected driver/session.

Examples

```
if (interactive()) {
  telem <- load_driver_telemetry(
    season = 2023,
    round = "Bahrain",
    session = "Q",
    driver = "HAM",
    laps = "fastest"
  )
}
```

`load_laps` *Load Lap by Lap Time Data*

Description

Loads basic lap-by-lap time data for all drivers in a given season and round. Lap time data is available from 1996 onward. Use `.load_laps()` for a uncached version.

Usage

```
load_laps(
  season = get_current_season(),
  round = "last",
  race = lifecycle::deprecated()
)
```

Arguments

`season`	number from 1996 to current season (defaults to current season).
`round`	number from 1 to 23 (depending on season selected) and defaults to most recent. Also accepts `'last'`.
`race`	**[Deprecated]** `race` is no longer supported, use `round`.

Value

A tibble with columns driver_id (unique and recurring), position during lap, time (in clock form), lap number, time (in seconds), and season.

load_pitstops *Load Pitstop Data*

Description

Loads pit stop info (number, lap, time elapsed) for a given race in a season. Pit stop data is available from 2012 onward. Call `.load_pitstops()` for an uncached version.

Usage

```
load_pitstops(
  season = get_current_season(),
  round = "last",
  race = lifecycle::deprecated()
)
```

Arguments

`season`	number from 2011 to current season (defaults to current season).
`round`	number from 1 to 23 (depending on season selected) and defaults to most recent.Also accepts `'last'`.
`race`	**[Deprecated]** `race` is no longer supported, please use `round`.

Value

A tibble with columns driver_id, lap, stop (number), time (of day), and stop duration

load_quali *Load Qualifying Results*

Description

Loads qualifying session results for a given season and round. Use `.load_quali()` for an uncached version.

Usage

```
load_quali(season = get_current_season(), round = "last")
```

Arguments

`season`	number from 2003 to current season (defaults to current season).
`round`	number from 1 to 23 (depending on season), and defaults to most recent. Also accepts `'last'`.

Value

A tibble with one row per driver

`load_race_session` *Load Session Data*

Description

Loads telemetry and general data from the official F1 data stream via the fastf1 python library. Data is available from 2018 onward.

The data loaded can optionally be assigned to a R variable, and then interrogated for session data streams. See the fastf1 documentation for more details on the data returned by the python API.

If you have trouble with errors mentioning 'fastf1' or 'get_fastf1_version()' read the 'Setup FastF1 Connection vignette (run `vignette('setup_fastf1', 'f1dataR')`).

Cache directory can be set by setting `option(f1dataR.cache = [cache dir])`, default is the current working directory.

Usage

```
load_race_session(
  obj_name = "session",
  season = get_current_season(),
  round = 1,
  session = "R",
  log_level = "WARNING",
  race = lifecycle::deprecated()
)
```

Arguments

`obj_name`	name assigned to the loaded session to be referenced later. Leave as `'session'` unless otherwise required.
`season`	number from 2018 to current season. Defaults to current season.
`round`	number from 1 to 24 (depending on season selected) and defaults to most recent. Also accepts race name.
`session`	the code for the session to load. Options are `'FP1'`, `'FP2'`, `'FP3'`, `'Q'`, `'S'`, `'SS'`,`'SQ'`, and `'R'`. Default is `'R'`, which refers to Race.
`log_level`	Detail of logging from fastf1 to be displayed. Choice of: `'DEBUG'`, `'INFO'`, `'WARNING'`, `'ERROR'` and `'CRITICAL.'` See fastf1 documentation.
`race`	**[Deprecated]** `race` is no longer supported, use `round`

Value

A session object to be used in other functions invisibly.

See Also

load_session_laps() plot_fastest()

Examples

```
# Load the quali session from 2019 first round
if (interactive()) {
  session <- load_race_session(season = 2019, round = 1, session = "Q")
}
```

---

load_results *Load Results*

Description

Loads final race results for a given year and round. Use .load_results() for an uncached version

Usage

```
load_results(season = get_current_season(), round = "last")
```

Arguments

season	number from 1950 to current season (or the word 'current') (defaults to current season).
round	number from 1 to 23 (depending on season), and defaults to most recent. Also accepts 'last'.

Value

A tibble with one row per driver, with columns for driver & constructor ID, the points won by each driver in the race, their finishing position, their starting (grid) position, number of completed laps, status code, gap to leader (or time of race), fastest lap ranking, drivers' fastest lap time, top speed achieved, and fastest lap time in seconds.

---

load_schedule *Load Schedule*

Description

Loads schedule information for a given F1 season. Use .load_schedule() for an uncached version.

Usage

```
load_schedule(season = get_current_season())
```

Arguments

season	number from 1950 to current season (defaults to current season). 'current' also accepted.

Value

A tibble with one row per round in season. Indicates in sprint_date if a specific round has a sprint race

---

load_session_laps *Load Lapwise Data*

Description

Loads lapwise data for a race session.

Includes each driver's each lap's laptime, pit in/out time, tyre information, track status, and (optionally) weather information. The resulting data frame contains a column for the session type. Note that quali sessions are labelled Q1, Q2 & Q3.

Cache directory can be set by setting `option(f1dataR.cache = [cache dir])`, default is the current working directory.

If you have trouble with errors mentioning 'fastf1' or 'get_fastf1_version()' read the 'Setup FastF1 Connection vignette (run `vignette('setup_fastf1', 'f1dataR')`).

Usage

```
load_session_laps(
  season = get_current_season(),
  round = 1,
  session = "R",
  log_level = "WARNING",
  add_weather = FALSE,
  race = lifecycle::deprecated()
)
```

Arguments

season	number from 2018 to current season. Defaults to current season.
round	number from 1 to 24 (depending on season selected) and defaults to most recent. Also accepts race name.
session	the code for the session to load. Options are 'FP1', 'FP2', 'FP3', 'Q', 'S', 'SS','SQ', and 'R'. Default is 'R', which refers to Race.
log_level	Detail of logging from fastf1 to be displayed. Choice of: 'DEBUG', 'INFO', 'WARNING', 'ERROR' and 'CRITICAL.' See fastf1 documentation.
add_weather	Whether to add weather information to the laps. See fastf1 documentation for info on weather.
race	**[Deprecated]** race is no longer supported, use round

Value

A tibble. Note time information is in seconds, see fastf1 documentation for more information on timing.

`load_sprint` *Load Sprint Results*

Description

Loads final race results for a given year and round. Note not all rounds have sprint results. Use `.load_sprint()` for an uncached version of this function.

Usage

```
load_sprint(season = get_current_season(), round = "last")
```

Arguments

`season`	number from 2021 to current season (defaults to current season).
`round`	number from 1 to 23 (depending on season), and defaults to most recent. Also accepts `'last'`.

Value

A dataframetibble with columns driver_id, constructor_id, points awarded, finishing position, grid position, laps completed, race status (finished or otherwise), gap to first place, fastest lap, fastest lap time, fastest lap in seconds, or NULL if no sprint exists for this season/round combo

---

`load_standings` *Load Standings*

Description

Loads standings at the end of a given season and round for drivers' or constructors' championships. Use `.load_standings()` for an uncached version of this function.

Usage

```
load_standings(season = get_current_season(), round = "last", type = "driver")
```

Arguments

`season`	number from 2003 to current season (defaults to current season).
`round`	number from 1 to 23 (depending on season), and defaults to most recent. Also accepts `'last'`.
`type`	select `'driver'` or `'constructor'` championship data. Defaults to `'driver'`

Value

A tibble with columns driver_id (or constructor_id), position, points, wins (and constructors_id in the case of drivers championship).

`plot_fastest` *Plot Fastest Lap*

Description

Creates a ggplot graphic that details the fastest lap for a driver in a race. Complete with a gearshift or speed analysis.

Usage

```
plot_fastest(
  season = get_current_season(),
  round = 1,
  session = "R",
  driver,
  color = "gear",
  race = lifecycle::deprecated()
)
```

Arguments

`season`	number from 2018 to current season (defaults to current season).
`round`	number from 1 to 23 (depending on season selected) and defaults to most recent.
`session`	the code for the session to load Options are `'FP1'`, `'FP2'`, `'FP3'`, `'Q'`, `'S'`, `'SS'`, `'SQ'`, and `'R'`. Default is `'R'`, which refers to Race.
`driver`	three letter driver code (see load_drivers() for a list) or name to be fuzzy matched to a driver from the session if FastF1 >= 3.4.0 is available.
`color`	argument that indicates which variable to plot along the circuit. Choice of `'gear'` or `'speed'`, default `'gear'`.
`race`	number from 1 to 23 (depending on season selected) and defaults to most recent.

Value

A ggplot object that indicates grand prix, driver, time and selected color variable.

Examples

```
# Plot Verstappen's fastest lap (speed) from Bahrain 2023:
if (interactive()) {
  plot_fastest(2023, 1, "R", "VER", "speed")
}
```

`setup_fastf1` *Setup fastf1 connection*

Description

Installs or optionally updates `fastf1` Python package in the current active Python environment/virtualenv/conda env.

More information on how to manage complex environment needs can be read in the reticulate docs, and tools for managing virtual environments are documented in virtualenv-tools and conda-tools

Usage

```
setup_fastf1(
  ...,
  envname = "f1dataR_env",
  new_env = identical(envname, "f1dataR_env")
)
```

Arguments

`...`	Additional parameters to pass to py_install
`envname`	Optionally pass an environment name used. Defaults to package default of `f1dataR_env`.
`new_env`	Whether or not to completely remove and recreate the environment provided in `envname`. This will fix any issues experienced by `fastf1` related to package dependencies.

Value

No return value, called to install or update `fastf1` Python package.

Examples

```
## Not run:
# Install fastf1 into the currently active Python environment
setup_fastf1()

# Reinstall fastf1 and recreate the environment.
setup_fastf1(envname = "f1dataR_env", new_env = TRUE)

## End(Not run)
```

theme_dark_f1 *Dark F1-style Theme for ggplot*

Description

Theme for all f1dataR plot functions. Mimics Formula 1 style.

Usage

```
theme_dark_f1(axis_marks = FALSE)
```

Arguments

axis_marks	True or false, whether axis line, ticks and title should be shown or not. Defaults to false

Value

A ggplot object that indicates grand prix, driver, time and selected color variable.

time_to_sec *Convert Clock time to seconds*

Description

This function converts clock format time (0:00.000) to seconds (0.000s)

Usage

```
time_to_sec(time)
```

Arguments

time	character string with clock format (0:00.000)

Value

A numeric variable that represents that time in seconds

Index