

Package: iker (via r-universe)

October 1, 2026

Title Iker Is Your Data Scribe

Version 0.9.5

Description Iker defines itself as a Scribe - a person who copies out documents. The package is meant to read / write data and provide various connectors.

License GPL-3 + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Imports ktools, readr, tibble

Suggests testthat (>= 3.0.0)

Remotes thekangaroofactory/ktools

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

Config/pak/sysreqs cmake make libuv1-dev libx11-dev zlib1g-dev

Repository <https://thekangaroofactory.r-universe.dev>

Date/Publication 2026-09-23 12:02:27 UTC

RemoteUrl <https://github.com/thekangaroofactory/iker>

RemoteRef HEAD

RemoteSha 7f4bac0c040415c8b07ad16c0f39437f0f3b7c51

Contents

from_POSIXct	2
load_data	2
read_data	3
save_data	4
to_POSIXct	5
write_data	5
Index	7

from_POSIXct	<i>Convert POSIXct Columns to ISO-8601 Character</i>
--------------	--

Description

Convert POSIXct Columns to ISO-8601 Character

Usage

```
from_POSIXct(data)
```

Arguments

data a data.frame (expected to contain POSIXct columns)

Value

a data.frame, whose POSIXct columns have been converted to ISO-8601 character columns

Examples

```
from_POSIXct(data.frame(a = Sys.time()))
```

load_data	<i>Load Data From Various Connectors</i>
-----------	--

Description

Load Data From Various Connectors

Usage

```
load_data(..., verbose = FALSE, data.frame = FALSE, force_empty = FALSE)
```

Arguments

...	arguments to pass to connector (see details)
verbose	a logical (default = FALSE) to indicate if additional traces should be sent to the console
data.frame	a logical (default = FALSE) if a data.frame should be returned (instead of a tibble)
force_empty	a logical (default = FALSE) if NULL should be returned when data has no row

Details

The load_data function is a wrapper around connector functions. It offers a single point of access to the package data persistence features.

Arguments passed to ... depend on the desired connector:

- file: [read_data](#)

Value

the output of the connector function

Examples

```
## Not run:
# -- file connector
load_data(file = "my_data.csv", path = getwd())

## End(Not run)
```

read_data

Read Data

Description

Read Data

Usage

```
read_data(
  path = Sys.getenv("DATA_HOME"),
  resource = NULL,
  file,
  delim = ",",
  col_types = NULL,
  verbose = FALSE
)
```

Arguments

path	A path where to read the data (see details).
resource	An optional resource name (used as sub folder inside path).
file	The name of the file to read (with extension).
delim	Single character used to separate fields within a record.
col_types	See read_delim function from readr package.
verbose	A logical whether additional trace should be sent to the console.

Details

By default, path takes its value from the DATA_HOME environment variable. If it's not set in the environment and no value has been provided, function will be stopped (and return NULL). When NULL, it will be ignored (assuming file contains the full path to the file)

col_types will be used to detect if POSIXct columns are expected. In this case they will be read as character first (expecting an ISO-8601 compatible string) then converted to POSIXct.

Value

A tibble, output value of the read_delim() call or NULL if data can't be read.

Examples

```
## Not run:
foo <- read_data(file = "my_data.csv")

## End(Not run)
```

save_data

Save Data To Various Connectors

Description

Save Data To Various Connectors

Usage

```
save_data(..., verbose = FALSE)
```

Arguments

...	arguments to pass to connector (see details)
verbose	a logical (default = FALSE) to indicate if additional traces should be sent to the console

Details

The save_data function is a wrapper around connector functions. It offers a single point of access to the package data persistence features.

Arguments passed to ... depend on the desired connector:

- file: [write_data](#)

Value

the output of the connector function

Examples

```
## Not run:
# -- file connector
save_data(file = "my_data.csv", path = getwd())

## End(Not run)
```

to_POSIXct*Convert ISO-8601 Character Columns to POSIXct*

Description

Convert ISO-8601 Character Columns to POSIXct

Usage

```
to_POSIXct(data, names_ct)
```

Arguments

data	a data.frame
names_ct	the names of the columns to convert

Value

a data.frame with converted columns

Examples

```
to_POSIXct(data.frame(a = "2025-05-13T12:15:30+0200"), "a")
```

write_data*Write Data*

Description

Write Data

Usage

```
write_data(
  x,
  path = Sys.getenv("DATA_HOME"),
  resource = NULL,
  file,
  delim = ",",
  verbose = FALSE
)
```

Arguments

x	A data frame to write to disk.
path	A path where to write the data (see details).
resource	An optional resource name (used as sub folder inside path).
file	The name of the file to write (with extension)
delim	Single character used to separate fields within a record.
verbose	If TRUE, additional messages will be sent to the console

Details

By default, path takes its value from the DATA_HOME environment variable. If it's not set in the environment and no value has been provided, function will be stopped (and return FALSE). If the corresponding path does not exist, it won't be created - to avoid creating data inside a container if the mount has not been done for example. When NULL, no check will be done.

Resource is an optional parameter to allow the use of sub folder inside path. If not NULL, it will be concatenated to path (and created inside path if it does not exist) When path is NULL, it will be ignored.

Columns with POSIXct type will be converted to ISO-8601 character columns before writing the file

Value

returns the input x invisibly, or FALSE when it fails

Examples

```
## Not run:  
write_data(foo, file = "my_data.csv")  
  
## End(Not run)
```

Index

from_POSIXct, [2](#)

load_data, [2](#)

read_data, [3](#), [3](#)

save_data, [4](#)

to_POSIXct, [5](#)

write_data, [4](#), [5](#)