Package: tsibbletalk (via r-universe)

October 12, 2024

Title Interactive Graphics for Tsibble Objects

Version 0.1.0.9000

Description A shared tsibble data easily communicates between htmlwidgets on both client and server sides, powered by 'crosstalk'. A shiny module is provided to visually explore periodic/aperiodic temporal patterns.

License GPL-3

URL https://github.com/earowang/tsibbletalk

BugReports https://github.com/earowang/tsibbletalk/issues

Depends R (>= 2.10)

Imports crosstalk (>= 1.1.0.1), dendextend (>= 1.13.4), dplyr (>= 1.0.0), glue (>= 1.4.1), lubridate (>= 1.7.9), plotly (>= 4.9.2.1), R6 (>= 2.4.1), rlang (>= 0.4.6), shiny (>= 1.5.0), tsibble (>= 0.9.1), vctrs (>= 0.3.1)

Suggests fabletools (>= 0.2.0), ggplot2

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Repository https://earowang.r-universe.dev

RemoteUrl https://github.com/earowang/tsibbletalk

RemoteRef HEAD

RemoteSha ee7c629f590e8254581003431d29e0c39cf75210

Contents

`as_shared_tsibble` *Coerce to a shared tsibble from tsibble*

Description

Coerce to a shared tsibble from tsibble

Usage

```
as_shared_tsibble(x, spec)
```

Arguments

`x`	A tsibble.
`spec`	A formula to specify tsibble key structures. By default, crossing structures (i.e `key1 * key2`) are assumed for the key. The required specification for nesting is `parent / child`.

Details

The object returned from `as_shared_tsibble()` can be piped into all plotly functions to be visualised.

Value

`SharedTsibbleData` subclassing of `SharedData` from crosstalk

Examples

```
library(tsibble)
as_shared_tsibble(tourism, spec = (State / Region) * Purpose)
```

`plotly_key_tree` *Plot nesting structures in shared tsibbles using plotly*

Description

Plot nesting structures in shared tsibbles using plotly

Usage

```
plotly_key_tree(data, height = NULL, width = NULL, ...)
```

Arguments

data	A shared tsibble.
height	height
width	width
...	arguments supplied to subplot()

Examples

```
if (interactive()) {
  shared_tourism <- as_shared_tsibble(tourism_monthly,
    spec = (State / Region) * Purpose)
  plotly_key_tree(shared_tourism)
}
```

---

sunspots2019 *Yearly mean total sunspot number (1700 - 2019)*

Description

Yearly mean total sunspot number (1700 - 2019)

Usage

```
sunspots2019
```

Format

An object of class tbl_ts (inherits from tbl_df, tbl, data.frame) with 320 rows and 2 columns.

References

WDC-SILSO, Royal Observatory of Belgium, Brussels

Examples

```
data(sunspots2019)
```

`tourism_monthly` *Monthly Australian domestic overnight trips*

Description

A dataset containing the monthly overnight trips from 1998 Jan to 2019 Dec across Australia.

Usage

```
tourism_monthly
```

Format

A tsibble with 80,696 rows and 5 variables:

- **Month**: Year month (index)
- **State**: States and territories of Australia
- **Region**: The tourism regions are formed through the aggregation of Statistical Local Areas (SLAs) which are defined by the various State and Territory tourism authorities according to their research and marketing needs
- **Purpose**: Stopover purpose of visit:
  - "Holiday"
  - "Visiting friends and relatives"
  - "Business"
  - "Other reason"
- **Trips**: Overnight trips in thousands

References

Tourism Research Australia

Examples

```
data(tourism_monthly)
```

tsibble-wrap *A shiny module to easily slice and dice tsibble index for visualising periodicity*

Description

A pair of UI and server functions: `tsibbleWrapUI()` and `tsibbleWrapServer()`.

Usage

```
tsibbleWrapUI(id)

tsibbleWrapServer(id, plot, period)
```

Arguments

`id`	A unique shiny id.
`plot`	A `ggplot` or `plotly` object.
`period`	A string passed to `lubridate::period()` to specify the minimum seasonal period, for example `"1 day"`.

Examples

```
if (interactive()) {
  library(tsibble)
  library(dplyr)
  library(shiny)
  library(ggplot2)
  p <- tourism %>%
    filter(Region %in% c("Melbourne", "Sydney")) %>%
    ggplot(aes(x = Quarter, y = Trips, colour = Region)) +
    geom_line() +
    facet_wrap(~ Purpose, scales = "free_y") +
    theme(legend.position = "none")

  ui <- fluidPage(tsibbleWrapUI("dice"))
  server <- function(input, output, session) {
    tsibbleWrapServer("dice", p, period = "1 year")
  }
  shinyApp(ui, server)
}
```

Index