Package ‘unstruwwel’

October 12, 2022

Title  Detect and Parse Historic Dates
Version 0.2.0
Maintainer Stefanie Schneider <stefanie.schneider@itg.uni-muenchen.de>
Description Automatically converts language-specific verbal information, e.g., "1st half of the 19th century," to its standardized numerical counterparts, e.g., "1801-01-01/1850-12-31." It follows the recommendations of the 'MIDAS' ('Marburger Informations- und Administrations-System'), see <doi:10.11588/artdok.00003770>.
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 7.2.1
URL https://github.com/stefanieschneider/unstruwwel
BugReports https://github.com/stefanieschneider/unstruwwel/issues
Suggests testthat, roxygen2
Imports R6, assertthat, lubridate, magrittr, stringr, tibble, tidyr, purrr, dplyr, rlang
Depends R (>= 2.10)
NeedsCompilation no
Author Stefanie Schneider [cre, aut] (<https://orcid.org/0000-0003-4915-6949>)
Repository CRAN
Date/Publication 2022-08-31 09:00:02 UTC

R topics documented:

Century ................................................................. 2
Decade .................................................................... 3
languages ................................................................ 4
midas ...................................................................... 4
Periods ...................................................................... 5
schemes .................................................................. 6
unstruwwel ............................................................. 7
Year ........................................................................ 8
Description

Set a Century and Get its Time Interval
Set a Century and Get its Time Interval

Details

An Object of R6Class with methods to set common time periods and specifications for centuries.

Super class

unstruwel::Periods -> Century

Methods

Public methods:

• Century$new()
• Century$clone()

Method new(): Helper function to specify the beginning of a century.
Helper function to specify the middle of a century.
Helper function to specify the end of a century.
Create a century.

Usage:
Century$new(value)
Arguments:
value A numerical scalar.
Returns: Object of R6Class with methods to set common time periods and specifications for centuries.

Method clone(): The objects of this class are cloneable with this method.

Usage:
Century$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Examples

if (interactive()) {
x <- Century$new(15)
x$take(2, type = "third")
}
Decade

Set a Decade and Get its Time Interval

Description

Set a Decade and Get its Time Interval
Set a Decade and Get its Time Interval

Details

An Object of R6Class with methods to set common time periods and specifications for decades.

Super class

unstruwel::Periods -> Decade

Methods

Public methods:

• Decade$new()
• Decade$clone()

Method new(): Helper function to specify the beginning of a decade.
Helper function to specify the middle of a decade.
Helper function to specify the end of a decade.
Create a decade.

Usage:
Decade$new(value, official_def = FALSE)

Arguments:
value A numerical scalar.
official_def If `TRUE`, the official definition that begins with the year 1 is used.

Returns: Object of R6Class with methods to set common time periods and specifications for decades.

Method clone(): The objects of this class are cloneable with this method.

Usage:
Decade$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Examples

if (interactive()) {
x <- Decade$new(1520)
x$take(1, type = "half")
}


**languages**

*Language Information*

**Description**

A dataset containing the names, date orders, stop words, simplifications, and replacements of 4 languages.

**Usage**

```r
data(languages)
```

**Format**

A tibble with 4 rows and 5 variables.

---

**midas**

*MIDAS Standardization Examples*

**Description**

A dataset containing eight thousand standardization examples of the MIDAS (Marburger Informations-, Dokumentations- und Administrations-System).

**Usage**

```r
data(midas)
```

**Format**

A vector of length 8115.
### Description

Set a Period and Get its Time Interval

Set a Period and Get its Time Interval

### Details

An Object of `R6Class` with methods to set common time periods and specifications for time periods.

### Public fields

- `.interval` Stores a time interval.
- `fuzzy` Either `-1` (approximate) or `1` (uncertain).
- `express` Either `-1` (before) or `1` (after).

### Active bindings

- `.interval` Stores a time interval.
- `interval` Convert and return a POSIXt time interval.
- `time_span` Convert and return a time span in years.
- `iso_format` Convert and return a date in ISO 8601.

### Methods

**Public methods:**

- `Periods$new()`
- `Periods$set_additions()`
- `Periods$take()`
- `Periods$clone()`

**Method** `new()`: Helper function to specify a time period.

Create a time period.

*Usage:*

`Periods$new(...)`

*Arguments:*

... Intervals, numerical scalars, or objects of class `Period`.

- `x` A numerical scalar. The range of valid values depends on `type`. If `type` is "early", "mid", or "late", `x` is ignored.

- `type` A character scalar. The following values are supported: "early", "mid", "late", "quarter", "third", and "half". If `type` is ‘NULL’, `x` defines a year or decade.
Method `set_additions()`: Set additions for a time period.

Usage:
`Periods$set_additions(x)`

Arguments:
- `x` A character vector.

Method `take()`: Specify a period.

Usage:
`Periods$take(x = NA, type = NA, ignore_errors = FALSE)`

Arguments:
- `x` A numerical scalar. The range of valid values depends on `type`. If `type` is "early", "mid", or "late", `x` is ignored.
- `type` A character scalar. The following values are supported: "early", "mid", "late", "quarter", "third", and "half". If `type` is 'NULL', `x` defines a year or decade.
- `ignore_errors` If 'TRUE', error messages are ignored.

Returns: Object of `R6Class` with methods to set common time periods and specifications for time periods.

Method `clone()`: The objects of this class are cloneable with this method.

Usage:
`Periods$clone(deep = FALSE)`

Arguments:
- `deep` Whether to make a deep clone.

---

**schemes**

<table>
<thead>
<tr>
<th>Language-Specific Scheme Variants</th>
</tr>
</thead>
</table>

**Description**

A dataset containing the values, schemes, and languages for over three thousand language-specific scheme variants.

**Usage**

`data(schemes)`

**Format**

A tibble with 3583 rows and 3 variables.
unstruwwel  Detect and Parse Historic Dates

Description


Usage

unstruwwel(
  x,
  language = NULL,
  verbose = TRUE,
  scheme = "time-span",
  fuzzify = c(0, 0)
)

Arguments

x Input vector. Either a character vector, or something coercible to one.
language Language code of the input vector as defined in ISO 639-1. If NULL, language is detected automatically.
verbose If TRUE, additional diagnostics are printed.
scheme Scheme code of the output list. Either time-span, iso-format, or object.
fuzzify A numerical vector of length 2 to extend the interval of approximate or uncertain time periods. This is only applied if scheme == "time-span".

Value

A named list of vectors or objects of R6Class.

Note

Although multiple languages can be detected, only dominant ones are ultimately set.

Examples

if (interactive()) {
  unstruwwel("1. Hälfte 19. Jahrhundert", language = "de")
  unstruwwel("circa between 1901 and 1905", language = "en")
}
Description
Set a Year and Get its Time Interval
Set a Year and Get its Time Interval

Details
An Object of `R6Class` with methods to set common time periods and specifications for years.

Super class
`unstruwel::Periods` -> Year

Methods

Public methods:
- `Year$new()`
- `Year$take()`
- `Year$clone()`

Method `new()`: Helper function to specify a time period.
Helper function to specify a season.
Helper function to specify a month.
Create a year.
Usage:
`Year$new(value)`
Arguments:
value  A numerical scalar.
Returns:  Object of `R6Class` with methods to set common time periods and specifications for years.

Method `take()`: Specify a year.
Usage:
`Year$take(x = NA, type = NA, ignore_errors = FALSE)`
Arguments:
x  A numerical scalar. The range of valid values depends on type. If type is "spring", "summer", "autumn", or "winter", x is ignored.
type  A character scalar. The following values are supported: "spring", "summer", "autumn", "winter", and all English-language months.
ignore_errors  If ‘TRUE’, error messages are ignored.
Returns: Object of \texttt{R6Class} with methods to set common time periods and specifications for years.

Method clone(): The objects of this class are cloneable with this method.

Usage:
\texttt{Year}$\texttt{clone(deep = FALSE)}$

Arguments:
\texttt{deep} Whether to make a deep clone.

Examples

if (interactive()) {
  x <- Year$new(1520)
  x$take(15, type = "june")
}
Index

* datasets
  languages, 4
  midas, 4
  schemes, 6

Century, 2
Decade, 3
languages, 4
midas, 4
Periods, 5
R6Class, 2, 3, 5–9
schemes, 6
unstruwwel, 7
unstruwwel::Periods, 2, 3, 8
Year, 8