Arrange your messy dates : : CHEAT SHEET

Basics

The package functions offer the user both **flexibility** and more **precision** when dealing with **uncertain dates.** It implements the extended annotation standard for dates, the Extended Date/Time Format (EDTF), outlined in **ISO 8601-2_2019(E)** for R.

These include standardised annotation with:

- "?" for date whose source is considered dubious
- "~" for approximate date
- "X" for unspecified date
- "[ ]" for set of dates [2012-01-01, 2012-01-12] or ".." for ranges 2012-01-01..2012-01-12"

1. Coerce objects to `messydt` class

From three columns

```
make_messydate(year, month, day)
```

Year	Month	Day
2007	2	17
2000	11	19
1998	12	6

→

Date	Class
2007-02-17	messydt
2000-11-19	messydt
1998-12-06	messydt

From one column

```
as_messydate(date)
```

These functions coerce different dates classes into '**messydt' class.** This specific date class allows to apply the other package **functions.**

Date	Class
2007-XX-18	as.Date
2020-09-22	POSIXct
2017-04-11	POSIXlt
2021-04	character
2018	character
2021-01-XX	character
2012-09-12~	character
2008-01-18?	character
2012-01..2012-03	character

→

Date	Class
2007-XX-18	messydt
2020-09-22	messydt
2017-04-11	messydt
2021-04	messydt
2018	messydt
2021-01	messydt
2012-09-12~	messydt
2008-01-18?	messydt
2012-01..2012-0	messydt

2. Explore all possible dates

Contract from all possible dates to one

```
contract(messydate)
```

"2021-04-01", "2021-04-02", …, "2021-04-30" → "2021-04"

"2021-01-01", "2021-01-02", …, "2021-12-31" → "2021"

Expand to all possible dates

```
expand(messydate)
```

Once the uncertain dates are in `messydt` class, expand() transform the **single uncertain** date to all **possible dates.** If the date is one year (e.g. 2021), expand() will give all 365 possible dates.

"2019-02-01..2019-02-03" → "2019-02-01" , "2019-02-02", "2019-02-03"

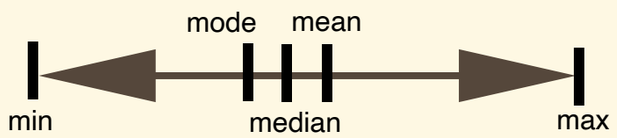
"2021-04" → "2021-04-01", "2021-04-02", …, "2021-04-29", "2021-04-30"

"2021" → "2021-01-01", "2021-01-02", "2021-01-03", …, "2021-12-30", "2021-12-31"

3. Choose a date and coerce it from `messydt` class

```
as.Date(as_messydate(date))
```

Coerce back to a **date, POSIXct** or **POSIXlt** class while choosing an exact value from all possible dates. It can be either the **minimum, maximum, mean, median,** or even the **mode** or a **random** value.



as.Date(as_messydate("2021-04"), min) → "2021-04-01"

as.Date(as_messydate("2021-04"), max) → "2021-04-30"

as.Date(as_messydate("2021-04"), mean) → "2021-04-16"

as.Date(as_messydate("2021-04"), median) → "2021-04-16"

as.Date(as_messydate("2021-04"), random) → "2021-04-11"

Other package functionalities

1) Some datasets contains arbitrary cut-off because the real date is unknown. This set of functions allows to indicate that these are approximations

Functions
on_or_before(1918-01-01)
on_or_after(2016-01-01)
add_approximation(1917-01-01)
add_uncertainty(2002-12-31)

→

Messydates
…1918-01-01
2016-01-01…
~1917-01-01
?2002-12-31

2) These functions allows to join two vectors of messydates in specific ways

```
md_intersect(messydates, messydates)
```

```
md_union(messydates, messydates)
```

```
md_multiset(messydates, messydates)
```

3) The package contains several logical tests which check whether the object is in `messydt` class or if one element is intersecting, similar to or part of a vector of messydates

```
is_messydate(messydate)
```

4) Extract only the years, months or days from the vector of `messydt` class or get the level of precision of a messydate

```
year(messydates)
```

```
precision(messydates)
```

 www.panarchic.ch • Learn more at **https://globalgov.github.io/messydates/** • package version 0.1.2 • Updated: 2021-11