Package: fasttime (via r-universe)

August 24, 2024

Version 1.1-0

Title Fast Utility Function for Time Parsing and Conversion

Author Simon Urbanek <simon.urbanek@r-project.org>

Maintainer Simon Urbanek <simon.urbanek@r-project.org>

Description Fast functions for timestamp manipulation that avoid system calls and take shortcuts to facilitate operations on very large data.

URL http://www.rforge.net/fasttime

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

RemoteUrl https://github.com/s-u/fasttime

RemoteRef HEAD

RemoteSha f7a59db5da5cd60a3336880c7a1014502e712e05

Contents

`fastPOSIXct` *Fast version of* `as.POSIXct.character` *for GMT fixed format.*

Description

`fastPOSIXct` converts timestamps in textual (string) form into `POSIXct` objects. It interprets sequences of digits separated by non-digits as a timestamp in GMT. The order of interpretation is fixed: year, month, day, hour, minute, second. Note that only true (positive) POSIX dates (since 1970-01-01 00:00:00) are supported and `fastPOSIXct` accepts dates up to year 2199.

It is extremely fast (compared to `as.POSIXct` by several orders of magnitude (on some platforms 1000x faster) since it uses pure text parsing and no system calls.

`fastDate` is a faster shorthand for `as.Date(fastPOSIXct(x, "GMT", 3L, fixed))`.

Usage

```
fastPOSIXct(x, tz = NULL, required.components = 3L, fixed = NA)
fastDate(x, fixed = NA)
```

Arguments

`x`	string vector to interpret as timestamps
`tz`	timezone for the resulting `POSIXct` object - this is NOT the time of `x`, that will always be GMT!
`required.components`	minimum number of timestamp components that are required. For example 3 means only the date is required, 6 means all components (up to the seconds) are required. If the requirement is not met, the result for that entry will be `NA`.
`fixed`	`NA` for separator-based parsing. If the components of the time stamp have fixed-width format (where separators are optional or non-existent) then this can be set to the number of characters expected for the year (see details).

Details

By default the parsing is done by extracting digits separated by non-digits. Each such group of digits is then interpreted in the order year, month, day, hour, minute and seconds. The separators can be arbitrary and of any length. Years less than 100 are assumed to be 2000..2099. For example, `"2022-01-05 09:15"` has five groups (from year to minute). The same timestamp would also be parsed from `"22/1/1, time 9:15"` or the ISO standard GMT form `"2022-01-01T09:15:00Z"`.

If the format uses no separators then `fixed` can be set to the length of the year component (in digits) – months through minutes are assumed to have at most two digits each. For example, the following call will have the same timestamp result as above: `fastPOSIXct("2201010915", fixed=2)`. Note that separators are still allowed in the fixed parsing and will terminate the preceding group so `"220101 9:15"` would yield the same result.

Value

Numeric vector of the class `POSIXct` (`fastPOSIXct`) or `Date` (`fastDate`).

Author(s)

Simon Urbanek

See Also

`as.POSIXct`

Examples

```
## let us generate a bunch of random timestamps until today
ts <- as.character(.POSIXct(runif(1e4) * unclass(Sys.time()), "GMT"))

## convert them using as.POSIXct
```

```
system.time(a <- as.POSIXct(ts, "GMT"))

## same using the fast method
system.time(b <- fastPOSIXct(ts, "GMT"))

identical(a, b)

## same for dates
day <- substr(ts, 1, 10)

system.time(da <- as.Date(day))
system.time(db <- fastDate(day))
identical(da, db)

## some parsing examples
fastPOSIXct("2022-01-05 09:15", "GMT")
fastPOSIXct("22/1/1, the time is 9:15", "GMT")
fastPOSIXct("2022-01-01T09:15:00Z", "GMT")
fastPOSIXct("2201010915", "GMT", fixed=2)

## the same converted to your local time zone
fastPOSIXct("2022-01-05 09:15")
## or ET
fastPOSIXct("2022-01-05 09:15", "US/Eastern")
```

Index