Package 'nanotime'

July 2, 2018

Type Package

Title Nanosecond-Resolution Time for R

Version 0.2.1

Date 2018-07-01

Author Dirk Eddelbuettel and Leonardo Silvestri

Maintainer Dirk Eddelbuettel <edd@debian.org>

Description Full 64-bit resolution date and time support with resolution up to nanosecond granularity is provided, with easy transition to and from the standard 'POSIXct' type.

Imports methods, bit64, RcppCCTZ (>= 0.2.3), zoo

Suggests RUnit, data.table, xts

License GPL (>= 2)

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-07-01 22:10:03 UTC

R topics documented:

nanotime-class 1

Index **7**

`nanotime-class` *Nanosecond resolution datetime functionality*

Description

Functions to operate on nanosecond time resolution using integer64 bit representation. Conversion functions for several standard R types are provided, and more will be added as needed.

`as.integer64` conversion helper returning the underlying `integer64` representation

Usage

```
nanotime(x, ...)

## S4 method for signature 'character'
nanotime(x, format = "", tz = "")

nanotime.matrix(x)

## S4 method for signature 'POSIXct'
nanotime(x)

## S4 method for signature 'POSIXlt'
nanotime(x)

## S4 method for signature 'Date'
nanotime(x)

## S4 method for signature 'nanotime'
print(x, format = "", tz = "", ...)

## S4 method for signature 'nanotime'
show(object)

## S3 method for class 'nanotime'
format(x, format = "", tz = "", ...)

## S3 method for class 'nanotime'
index2char(x, ...)

## S3 method for class 'nanotime'
as.POSIXct(x, tz = "", ...)

## S3 method for class 'nanotime'
as.POSIXlt(x, tz = "", ...)

## S3 method for class 'nanotime'
as.Date(x, ...)

## S3 method for class 'nanotime'
as.data.frame(x, ...)

## S3 method for class 'nanotime'
as.integer64(x, ...)

as.integer64(x, ...)

## S4 method for signature 'nanotime,character'
e1 - e2
```

```
## S4 method for signature 'nanotime,nanotime'
e1 - e2

## S4 method for signature 'nanotime,integer64'
e1 - e2

## S4 method for signature 'nanotime,numeric'
e1 - e2

## S4 method for signature 'ANY,nanotime'
e1 - e2

## S4 method for signature 'nanotime,ANY'
e1 - e2

## S4 method for signature 'nanotime,ANY'
e1 + e2

## S4 method for signature 'nanotime,integer64'
e1 + e2

## S4 method for signature 'nanotime,numeric'
e1 + e2

## S4 method for signature 'ANY,nanotime'
e1 + e2

## S4 method for signature 'integer64,nanotime'
e1 + e2

## S4 method for signature 'numeric,nanotime'
e1 + e2

## S4 method for signature 'nanotime,nanotime'
e1 + e2

## S4 method for signature 'nanotime,ANY'
Arith(e1, e2)

## S4 method for signature 'ANY,nanotime'
Arith(e1, e2)

## S4 method for signature 'nanotime,ANY'
Compare(e1, e2)

## S4 method for signature 'nanotime,nanotime'
Compare(e1, e2)
```

```
## S4 method for signature 'nanotime,ANY'
Logic(e1, e2)

## S4 method for signature 'ANY,nanotime'
Logic(e1, e2)

## S4 method for signature 'nanotime'
Math(x)

## S4 method for signature 'nanotime'
Math2(x, digits)

## S4 method for signature 'nanotime'
Summary(x, ..., na.rm = FALSE)

## S4 method for signature 'nanotime'
min(x, ..., na.rm = FALSE)

## S4 method for signature 'nanotime'
max(x, ..., na.rm = FALSE)

## S4 method for signature 'nanotime'
range(x, ..., na.rm = FALSE)

## S4 method for signature 'nanotime'
Complex(z)

## S4 method for signature 'nanotime'
x[i, j, ..., drop = FALSE]

## S4 replacement method for signature 'nanotime'
x[i, j, ...] <- value

## S3 method for class 'nanotime'
c(...)

## S4 replacement method for signature 'nanotime'
names(x) <- value

## S4 method for signature 'nanotime'
is.na(x)
```

Arguments

x	The object which want to convert to class nanotime
...	further arguments passed to or from methods.
format	A character string. Can also be set via options("nanotimeFormat") and uses

	'%Y-%m-%dT%H:%M:%E9S%Ez' as a default and fallback
`tz`	Required for `as.POSIXct` and `as.POSIXlt`, can be set via `options("nanotimeTz")` and uses 'UTC' as a default and fallback
`object`	argument for method `show`
`e1`	Operand of class `nanotime`
`e2`	Operand of class `nanotime`
`digits`	Required for `Math2` signature but ignored here
`na.rm`	a logical indicating whether missing values should be removed.
`z`	Required for `Complex` signature but ignored here
`i`	index specifying elements to extract or replace.
`j`	Required for `[` signature but ignored here
`drop`	Required for `[` signature but ignored here
`value`	argument for `nanotime-class`

Details

Notice that the conversion from POSIXct explicitly sets the last three digits to zero. Nanosecond time stored in a 64-bit integer has nineteen digits precision where doubles (which are used internally for POSIXct as well) only have sixteen digits. So rather than showing three more (essentially *random*) digits it is constructed such that these three additional digits are zeros.

Value

A nanotime object

Caveats

Working with dates and times is *difficult*. One needs a representation of both *time points* and *time duration*. In R, think of `Date` or `POSIXct` objects for the former, and `difftime` for the later. Here we (currently) only have time points, but they are effectively also durations relative to the epoch of January 1, 1970.

Design

There are two external libraries doing two key components.

We rely on the `bit64` package for `integer64` types to represent nanoseconds relative to the epoch. This is similar to `POSIXct` which uses fractional seconds since the epoch—so here we are essentially having the same values, but multiplied by 10 to the power 9 and stored as integers. We need to rely on the external package as we require 64-bit integers whereas R itself only has 32-bit integers. The `bit64` package is clever about how it manages to provide such an integer using only the 64-bit double type and very clever (and efficient) transformations.

The other is the CCTZ library in C++, which we access via the `RcppCCTZ` package. CCTZ extends the C++11 standard library type `chrono` type in very useful ways for time zones and localtime. We use its formating and parsing features.

Output Format

Formatting and character conversion for `nanotime` objects is done by functions from the `RcppCCTZ` package relying on code from its embedded `CCTZ` library. The default format is ISO3339 compliant: `%Y-%m-%dT%H:%M:%E9S%Ez`. It specifies a standard ISO 8601 part for date and time — as well as nine digits of precision for fractional seconds (down to nanoseconds) and on offset (typically zero as we default to UTC). It can be overriden by using `options()` with the key of `nanotimeFormat` and a suitable value. Similarly, `nanotimeTz` can be used to select a different timezone.

Author(s)

Dirk Eddelbuettel

Examples

```
x <- nanotime("1970-01-01T00:00:00.000000001+00:00")
print(x)
x <- x + 1
print(x)
format(x)
x <- x + 10
print(x)
format(x)
format(nanotime(Sys.time()) + 1:3)  # three elements each 1 ns apart
```

Index

+,ANY,nanotime-method *(nanotime-class)*, 1
+,integer64,nanotime-method *(nanotime-class)*, 1
+,nanotime,ANY-method *(nanotime-class)*, 1
+,nanotime,integer64-method *(nanotime-class)*, 1
+,nanotime,nanotime-method *(nanotime-class)*, 1
+,nanotime,numeric-method *(nanotime-class)*, 1
+,numeric,nanotime-method *(nanotime-class)*, 1
-,ANY,nanotime-method *(nanotime-class)*, 1
-,nanotime,ANY-method *(nanotime-class)*, 1
-,nanotime,character-method *(nanotime-class)*, 1
-,nanotime,integer64-method *(nanotime-class)*, 1
-,nanotime,nanotime-method *(nanotime-class)*, 1
-,nanotime,numeric-method *(nanotime-class)*, 1
[,nanotime-method *(nanotime-class)*, 1
[<-,nanotime-method *(nanotime-class)*, 1

Arith,ANY,nanotime-method *(nanotime-class)*, 1
Arith,nanotime,ANY-method *(nanotime-class)*, 1
as.data.frame.nanotime *(nanotime-class)*, 1
as.Date.nanotime *(nanotime-class)*, 1
as.integer64 *(nanotime-class)*, 1
as.POSIXct.nanotime *(nanotime-class)*, 1
as.POSIXlt.nanotime *(nanotime-class)*, 1

bit64, *5*

c.nanotime *(nanotime-class)*, 1
Compare,nanotime,ANY-method *(nanotime-class)*, 1
Compare,nanotime,nanotime-method *(nanotime-class)*, 1
Complex,nanotime-method *(nanotime-class)*, 1

format.nanotime *(nanotime-class)*, 1

index2char.nanotime *(nanotime-class)*, 1
is.na,nanotime-method *(nanotime-class)*, 1

Logic,ANY,nanotime-method *(nanotime-class)*, 1
Logic,nanotime,ANY-method *(nanotime-class)*, 1

Math,nanotime-method *(nanotime-class)*, 1
Math2,nanotime-method *(nanotime-class)*, 1
max,nanotime-method *(nanotime-class)*, 1
min,nanotime-method *(nanotime-class)*, 1

names<-,nanotime-method *(nanotime-class)*, 1
nanotime *(nanotime-class)*, 1
nanotime,character-method *(nanotime-class)*, 1
nanotime,Date-method *(nanotime-class)*, 1
nanotime,POSIXct-method *(nanotime-class)*, 1
nanotime,POSIXlt-method *(nanotime-class)*, 1
nanotime-class, 1
nanotime-package *(nanotime-class)*, 1
nanotime.matrix *(nanotime-class)*, 1

print,nanotime-method *(*nanotime-class*)*, 1

range,nanotime-method *(*nanotime-class*)*, 1

RcppCCTZ, *5*, *6*

show,nanotime-method *(*nanotime-class*)*, 1

Summary,nanotime-method *(*nanotime-class*)*, 1