Package ‘RcppBDT’

October 31, 2022

Type Package
Title 'Rcpp' Bindings for the Boost Date_Time Library
Version 0.2.6
Date 2022-10-30
Author Dirk Eddelbuettel and Romain Francois
Maintainer Dirk Eddelbuettel <edd@debian.org>
Description Access to Boost Date_Time functionality for dates, durations (both for days and date time objects), time zones, and posix time ('ptime') is provided by using 'Rcpp modules'. The posix time implementation can support high-resolution of up to nano-second precision by using 96 bits (instead of R's 64) to present a 'ptime' object (but this needs recompilation with a #define set).
License GPL (>= 2)
LazyLoad yes
Imports Rcpp, methods
LinkingTo Rcpp, BH
URL https://github.com/eddelbuettel/rcppbdt,
      https://dirk.eddelbuettel.com/code/rcpp.bdt.html
BugReports https://github.com/eddelbuettel/rcppbdt/issues
RoxygenNote 6.0.1
NeedsCompilation yes
Repository CRAN
Date/Publication 2022-10-30 23:32:49 UTC

R topics documented:

  RcppBDT-package .............................................. 2
  bdtDd .......................................................... 2
Description

This package provides R with access to Boost Date_Time functionality by using Rcpp modules. Date, Local time, duration and time zone functionality is covered.

Details

Please consult the Boost documentation for (copious) details on the Date_Time library.

Author(s)

Dirk Eddelbuettel <edd@debian.org>

References


Rcpp module bdtDd for binding of Boost Date_Time date duration functionality

Description

The bdtDd module is created using Rcpp modules and wraps a helper class bdtDd around Boost Date_time date duration functionality provided by the Boost class boost::gregorian::date_duration.

New instances can be created using an integer for days of duration.

Usage

days(...)
weeks(...)
Arguments

... suitable argument, often an integer, denoting one unit of the reference duration component

Details

Please consult the Boost documentation for (copious) details on the Date_Time library. See the Rcpp-modules vignette for details on Rcpp modules.

Method

show signature(x = "Rcpp_bdtDt"): prints a (BDTdd) date duration class object
format signature(x = "Rcpp_bdtDt"): formats a (BDTdd) date duration class object

Author(s)

Dirk Eddelbuettel <edd@debian.org>

References


bdtDt Rcpp module bdtDt for binding of Boost Date_Time Date functionality

Description

The bdtDt module is created using Rcpp modules and wraps a helper class bdtDt around Boost Date_time date functionality provided by the Boost class boost::gregorian::date.

New instances can be created using either the default constructor (without arguments) or the constructor using year, month, date arguments.

The bdt variable is a default instance of this bdtDt reference class. It facilities accessing the member functions via utility function, see for example getEndOfBizWeek or print(bdtDt) for the available methods.

Details

Please consult the Boost documentation for (copious) details on the Date_Time library. See the Rcpp-modules vignette for details on Rcpp modules.

Method

show signature(x = "Rcpp_bdtDt"): prints a (bdtDt) date class object
format signature(x = "Rcpp_bdtDt"): formats a (bdtDt) date class object
**Author(s)**

Dirk Eddelbuettel <edd@debian.org>

**References**


---

**bdtDu**  
*Rcpp module bdtDu for binding of Boost Date_Time duration functionality*

---

**Description**

The bdtDu module is created using Rcpp modules and wraps a helper class bdtDu around Boost Date_time duration functionality provided by the Boost class boost::posix_time::duration. New instances can be created using four integer values for hour, minute, seconds and fractional seconds. Fractional seconds ought to be at a nano-second granularity; there may be platforms not permitting this.

**Usage**

- `hours(...)`
- `microseconds(...)`
- `milliseconds(...)`
- `minutes(...)`
- `nanoseconds(...)`
- `seconds(...)`

**Arguments**

- `...` suitable argument, often an integer, denoting one unit of the reference duration component

**Details**

Please consult the Boost documentation for (copious) details on the Date_Time library. See the Rcpp-modules vignette for details on Rcpp modules.

**Method**

- `show` signature `x = "Rcpp_bdtDu")`: prints a (BDTdu) duration class object
- `format` signature `x = "Rcpp_bdtDu")`: formats a (BDTdu) duration class object

**Author(s)**

Dirk Eddelbuettel <edd@debian.org>
References


bdtp

Rcpp module bdtp for binding of Boost Date_Time ptime functionality

Description

The bdtp module is created using Rcpp modules and wraps a helper class bdtpPt around Boost Date_time duration functionality provided by the Boost class boost::posix_time::ptime.

New instances can be created using either a default construction (creating an unset instance) or using seven integer values for year, month, day, hour, minute, seconds and fractional seconds. Fractional seconds ought to be at a nano-second granularity; there may be platforms not permitting this.

Details

Please consult the Boost documentation for (copious) details on the Date_Time library. See the Rcpp-modules vignette for details on Rcpp modules.

Method

show signature(x = "Rcpp_bdtpPt"): prints a (bdtpPt) ptime class object

format signature(x = "Rcpp_bdtpPt"): formats a (bdtpPt) ptime class object

Author(s)

Dirk Eddelbuettel <edd@debian.org>

References


bdtTz

Rcpp module bdtTz for binding of Boost Date_Time timezone functionality

Description

The bdttz module is created using Rcpp modules and wraps a helper class bdttz around Boost Date_time timezone functionality provided mainly by the Boost classes boost::local_time::tz_database and boost::local_time::time_zone_ptr.

On startup, the database object is initialized using a local copy (in csv format) of the timezone data. Instances of the timezone object, represented by an instance of the timezone pointer class, can be created and queried.

New instances can be created using a valid timezone region string (such “Europe/London”).
Details

Please consult the Boost documentation for (copious) details on the Date_Time library. See the Rcpp-modules vignette for details on Rcpp modules.

Method

show signature(x = "Rcpp_bdtTz"): prints a (bdtTz) timezone class object
format signature(x = "Rcpp_bdtTz"): formats a (bdtTz) timezone class object

Author(s)

Dirk Eddelbuettel <edd@debian.org>

References


charToPOSIXct

Parse POSIXct objects from character variables

Description

This function uses the Boost Date_Time library to parse datetimes from strings. It returns a vector of POSIXct objects. These represent dates and time as (possibly fractional) seconds since the ‘epoch’ of January 1, 1970. A timezone can be set, if none is supplied ‘UTC’ is set.

Usage

ccharToPOSIXct(sv, tz = "UTC")

Arguments

sv A vector of type character with datetime expressions in ISO format to be parsed and converted.
tz A string with the timezone, defaults to ‘UTC’ if unset

Details

A single standard ISO format ‘YYYY-MM-DD HH:MM:SS’ (with optional trailing fractional seconds) is tried. In the case of parsing failure a NA value is returned. See the function toPOSIXct for more general input format

Fractional seconds are supported as well. As R itself only supports microseconds, the Boost compile-time option for nano-second resolution has not been enabled.

Value

A vector of ‘POSIXct’ elements.
cToPOSIXct

Author(s)
Dirk Eddelbuettel

Examples

```r
times <- c("2004-03-21 12:45:33.123456",
"2004-03-21 12:45:34")
charToPOSIXct(times)
```

cToPOSIXct

Parse POSIXct objects from character variables

Description

This function uses Rcpp to parse datetimes from strings. It returns a vector of POSIXct objects. These represent dates and time as (possibly fractional) seconds since the ‘epoch’ of January 1, 1970. A timezone can be set, if none is supplied ‘UTC’ is set.

Usage

```r
cToPOSIXct(sv, fmt = "/%Y-%m-%d %H:%M:%OS", tz = "UTC")
```

Arguments

- `sv`: A vector of type character with datetime expressions in ISO format to be parsed and converted.
- `fmt`: A format, defaults to the ISO format if unset
- `tz`: A string with the timezone, defaults to `"UTC"` if unset

Details

The default standard ISO format ‘YYYY-MM-DD HH:MM:SS.FFFFFFF’ is used by default along with the UTC time zone.

This function is for comparison only.

Value

A vector of ‘POSIXct’ elements.

Author(s)
Dirk Eddelbuettel

See Also

Rcpp
Examples

cToPOSIXct(times)

RcppBDT Date functions

_RcppBDT Date functions_  
_Date accessor and construction functions from Boost Date_Time_

Description

This constants are provided for convenience. In the C++ sources, enumeration types are used for days of the week, months of the year as well as the ordering terms.

Similar package-level constants are provided here as well. This should be considered as experimental and may be withdrawn in a later version of the package.

Usage

getEndOfWeek(date)  
getEndOfMonth(date)  
getYear(date)  
getMonth(date)  
getDay(date)  
getDayOfWeek(date)  
getDayOfYear(date)  
getIMMDate(mon, year)  
getNthDayOfWeek(nthday, dow, mon, year)  
getLastDayOfWeekInMonth(dow, mon, year)  
getFirstDayOfWeekInMonth(dow, mon, year)  
getFirstDayOfWeekAfter(dow, date)  
getLastDayOfWeekBefore(dow, date)

Arguments

date a _Date_ object  
mon a month, specified either as an integer or one of the constants _Jan, Feb, ..._ defined in this package  
year a four-digit year, specified as an integer  
nthday either an integer between 1 and 5, or one of the constants _first, second, ... fifth_ defined in this package.  
dow either an integer between 0 and 6 denoting a day of the week, or one of the constants _Sun, Mon, ... Sat_ defined in this package.

Details

Details of the Boost functions are provided by the Boost documentation.
Value

All functions return a `Date` object.

Author(s)

Dirk Eddelbuettel <edd@debian.org>

References


Description

This constants are provided for convenience. In the C++ sources, enumeration types are used for days of the week, months of the year as well as the ordering terms.

Similar package-level constants are provided here as well. This should be considered as experimental and may be withdrawn in a later version of the package.

Details

Sun, Mon, Tue, ..., Sat can be used instead of the values 0 to 6.
Jan, Feb, ..., Dec can be used instead of the values 1 to 12.
First, second, ..., fifth can be used instead of the values 1 to 5.

We use the same values as the Boost source code. In other words, Sunday is 0, Monday is 1 and so on. Months, however, start at 1 for January.

Author(s)

Dirk Eddelbuettel <edd@debian.org>

References

toPOSIXct  Parse POSIXct objects from input data

Description

This function uses the Boost Date_Time library to parse datetimes (and dates) from strings, integers or even numeric values (which are cast to strings internally). It returns a vector of POSIXct objects. These represent dates and time as (possibly fractional) seconds since the ‘epoch’ of January 1, 1970. A timezone can be set, if none is supplied ‘UTC’ is set.

Usage

```r
toPOSIXct(x, tz = "UTC")
```

Arguments

- `x`  A vector of type character, integer or numeric with date(time) expressions to be parsed and converted.
- `tz`  A string with the timezone, defaults to ‘UTC’ if unset

Details

A number of fixed formats are tried in succession. These include the standard ISO format ‘YYYY-MM-DD HH:MM:SS’ as well as different local variants including several forms popular in the United States. Two-digits years and clearly ambiguous formats such as ‘03/04/05’ are ignored. In the case of parsing failure a NA value is returned.

Fractional seconds are supported as well. As R itself only supports microseconds, the Boost compile-time option for nano-second resolution has not been enabled.

Value

A vector of ‘POSIXct’ elements.

Author(s)

Dirk Eddelbuettel

See Also

The function in the anytime package which is a more finished variant which is based on the initial work with function, and taken into its own package.
Examples

## See the source code (hah!) for a full list of formats

times <- c("2004-03-21 12:45:33.123456",
           "2004/03/21 12:45:33.123456",
           "20040321 124533.123456",
           "21.03.2004 12:45:33.123456",
           "03/21/2004 12:45:33.123456",
           "03-21-2004 12:45:33.123456",
           "2004-03-21",
           "20040321",
           "03/21/2004",
           "03-21-2004",
           "20010101")

toPOSIXct(times)
Index

* package
  bdtDd, 2
  bdtDt, 3
  bdtDu, 4
  bdtPt, 5
  bdtTz, 5
  RcppBDT Date functions, 8
  RcppBDT-constants, 9
  RcppBDT-package, 2

Apr (RcppBDT-constants), 9
arith_bdtDd_bdtDd (bdtDd), 2
arith_bdtDd_bdtDt (bdtDt), 2
arith_bdtDd_int (bdtDd), 2
arith_bdtDt_bdtDd (bdtDt), 3
arith_bdtDt_int (bdtDt), 3
arith_bdtDu_bdtDu (bdtDu), 4
arith_bdtDu_bdtPt (bdtDu), 4
arith_bdtDu_int (bdtDu), 4
arith_bdtPt_bdtDu (bdtPt), 5
arith_bdtPt_double (bdtPt), 5
arith_double_bdtPt (bdtPt), 5
arith_int_bdtDd (bdtDd), 2
arith_int_bdtDt (bdtDt), 3
arith_int_bdtDu (bdtDu), 4
Aug (RcppBDT-constants), 9

bdt (bdtDt), 3
bdtDd, 2
bdtDt, 3
bdtDu, 4
bdtPt, 5
bdtTz, 5

charToPOIXct, 6
compare_bdtDd_bdtDd (bdtDd), 2
compare_bdtDt_bdtDt (bdtDt), 3
compare_bdtDu_bdtDu (bdtDu), 4
compare_bdtPt_bdtPt (bdtPt), 5
cToPOIXct, 7

date, 8, 9
days (bdtDd), 2
Dec (RcppBDT-constants), 9

Feb, 8
Feb (RcppBDT-constants), 9
fifth, 8
fifth (RcppBDT-constants), 9
first, 8
first (RcppBDT-constants), 9
format, Rcpp_bdtDd-method (bdtDd), 2
format, Rcpp_bdtDt-method (bdtDt), 3
format, Rcpp_bdtDu-method (bdtDu), 4
format, Rcpp_bdtPt-method (bdtPt), 5
format, Rcpp_bdtTz-method (bdtTz), 5
fourth (RcppBDT-constants), 9
Fri (RcppBDT-constants), 9

getDay (RcppBDT Date functions), 8
gDayOfWeek (RcppBDT Date functions), 8
gDayOfYear (RcppBDT Date functions), 8
gEndOfBizWeek, 3
gEndOfBizWeek (RcppBDT Date functions), 8
gEndOfMonth (RcppBDT Date functions), 8
gFirstDayOfWeekAfter (RcppBDT Date functions), 8
gFirstDayOfWeekBefore (RcppBDT Date functions), 8
gIMDDate (RcppBDT Date functions), 8
gLastDayOfBizWeekBefore (RcppBDT Date functions), 8
gLastDayOfWeekInMonth (RcppBDT Date functions), 8
gMonth (RcppBDT Date functions), 8
gNthDayOfWeek (RcppBDT Date functions), 8
gYear (RcppBDT Date functions), 8
hours (bdtDu), 4
Jan, 8
Jan (RcppBDT-constants), 9
Jul (RcppBDT-constants), 9
Jun (RcppBDT-constants), 9
Mar (RcppBDT-constants), 9
May (RcppBDT-constants), 9
microseconds (bdtDu), 4
milliseconds (bdtDu), 4
minutes (bdtDu), 4
Mon, 8
Mon (RcppBDT-constants), 9
nanoseconds (bdtDu), 4
Nov (RcppBDT-constants), 9
Oct (RcppBDT-constants), 9
Rcpp, 7
Rcpp_bdtDd-class (bdtDd), 2
Rcpp_bdtDt-class (bdtDt), 3
Rcpp_bdtDu-class (bdtDu), 4
Rcpp_bdtPt-class (bdtPt), 5
Rcpp_bdtTz-class (bdtTz), 5
RcppBDT (RcppBDT-package), 2
RcppBDT Date functions, 8
RcppBDT-constants, 9
RcppBDT-package, 2
Sat, 8
Sat (RcppBDT-constants), 9
second, 8
second (RcppBDT-constants), 9
seconds (bdtDu), 4
Sep (RcppBDT-constants), 9
show, Rcpp_bdtDd-method (bdtDd), 2
show, Rcpp_bdtDt-method (bdtDt), 3
show, Rcpp_bdtDu-method (bdtDu), 4
show, Rcpp_bdtPt-method (bdtPt), 5
show, Rcpp_bdtTz-method (bdtTz), 5
Sun, 8
Sun (RcppBDT-constants), 9
third (RcppBDT-constants), 9
Thu (RcppBDT-constants), 9
toPOSIXct, 6, 10
Tue (RcppBDT-constants), 9
Wed (RcppBDT-constants), 9
weeks (bdtDd), 2