

# Package: RcppBDT (via r-universe)

July 2, 2024

**Type** Package

**Title** 'Rcpp' Bindings for the Boost Date\_Time Library

**Version** 0.2.6.1

**Date** 2023-03-30

**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

**Repository** <https://eddelbuettel.r-universe.dev>

**RemoteUrl** <https://github.com/eddelbuettel/rcppbdt>

**RemoteRef** HEAD

**RemoteSha** 346becb706784f5ee669e0e8cc3fc55eb7304193

## Contents

RcppBDT-package . . . . .	2
bdtDd . . . . .	3
bdtDt . . . . .	4
bdtDu . . . . .	4
bdtPt . . . . .	5
bdtTz . . . . .	6
charToPOSIXct . . . . .	7
cToPOSIXct . . . . .	8
RcppBDT Date functions . . . . .	9
RcppBDT-constants . . . . .	10
toPOSIXct . . . . .	10
<b>Index</b>	<b>12</b>

---

RcppBDT-package      *Bindings for Boost Date\_Time*

---

### 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

Boost Date\_Time: [https://www.boost.org/doc/libs/release/doc/html/date\\_time.html](https://www.boost.org/doc/libs/release/doc/html/date_time.html)

---

bdtDd	<i>Rcpp module bdtDd for binding of Boost Date_Time date duration functionality</i>
-------	---

---

## 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\_bdtDd"): prints a (BDTdd) date duration class object

**format** signature(x = "Rcpp\_bdtDd"): formats a (BDTdd) date duration class object

## Author(s)

Dirk Eddelbuettel <edd@debian.org>

## References

Boost Date\_Time: [https://www.boost.org/doc/libs/release/doc/html/date\\_time.html](https://www.boost.org/doc/libs/release/doc/html/date_time.html)

---

**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

Boost `Date_Time`: [https://www.boost.org/doc/libs/release/doc/html/date\\_time.html](https://www.boost.org/doc/libs/release/doc/html/date_time.html)

---

**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

Boost Date\_Time: [https://www.boost.org/doc/libs/release/doc/html/date\\_time.html](https://www.boost.org/doc/libs/release/doc/html/date_time.html)

---

bdtPt

*Rcpp module bdtPt for binding of Boost Date\_Time ptime functionality*

---

## Description

The bdtDu module is created using Rcpp modules and wraps a helper class bdtPt 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\_bdtPt"): prints a (bdtPt) ptime class object

**format** signature(x = "Rcpp\_bdtPt"): formats a (bdtPt) ptime class object

**Author(s)**

Dirk Eddelbuettel <edd@debian.org>

**References**

Boost Date\_Time: [https://www.boost.org/doc/libs/release/doc/html/date\\_time.html](https://www.boost.org/doc/libs/release/doc/html/date_time.html)

---

bdtTz	<i>Rcpp module bdtTz for binding of Boost Date_Time timezone functionality</i>
-------	--

---

**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**

Boost Date\_Time: [https://www.boost.org/doc/libs/release/doc/html/date\\_time.html](https://www.boost.org/doc/libs/release/doc/html/date_time.html)

---

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

```
charToPOSIXct(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.

## Author(s)

Dirk Eddelbuettel

## Examples

```
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**

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

**Arguments**

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

**Details**

The default standard ISO format ‘YYYY-MM-DD HH:MM:SS.FFFFFFFF’ 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**

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



---

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

```
getEndOfBizWeek(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 <a href="#">Date</a> object
mon	a month, specified either as an integer or one of the constants <a href="#">Jan</a> , <a href="#">Feb</a> , ... 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 <a href="#">first</a> , <a href="#">second</a> , ... <a href="#">fifth</a> defined in this package.
dow	either an integer between 0 and 6 denoting a day of the week, or one of the constants <a href="#">Sun</a> , <a href="#">Mon</a> , ... <a href="#">Sat</a> 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**

Boost Date\_Time: [https://www.boost.org/doc/libs/release/doc/html/date\\_time.html](https://www.boost.org/doc/libs/release/doc/html/date_time.html)

---

RcppBDT-constants      *Constants for date functions with 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.

**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**

Boost Date\_Time: [https://www.boost.org/doc/libs/release/doc/html/date\\_time.html](https://www.boost.org/doc/libs/release/doc/html/date_time.html)

---

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 internal). 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**

```
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, 3
  - bdtDt, 4
  - bdtDu, 4
  - bdtPt, 5
  - bdtTz, 6
  - RcppBDT Date functions, 9
  - RcppBDT-constants, 10
  - RcppBDT-package, 2
- Apr (RcppBDT-constants), 10
- arith\_bdtDd\_bdtDd (bdtDd), 3
  - arith\_bdtDd\_bdtDt (bdtDd), 3
  - arith\_bdtDd\_int (bdtDd), 3
  - arith\_bdtDt\_bdtDd (bdtDt), 4
  - arith\_bdtDt\_int (bdtDt), 4
  - 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), 3
  - arith\_int\_bdtDt (bdtDt), 4
  - arith\_int\_bdtDu (bdtDu), 4
- Aug (RcppBDT-constants), 10
- bdt (bdtDt), 4
- bdtDd, 3
  - bdtDt, 4
  - bdtDu, 4
  - bdtPt, 5
  - bdtTz, 6
- charToPOSIXct, 7
- compare\_bdtDd\_bdtDd (bdtDd), 3
  - compare\_bdtDt\_bdtDt (bdtDt), 4
  - compare\_bdtDu\_bdtDu (bdtDu), 4
  - compare\_bdtPt\_bdtPt (bdtPt), 5
- cToPOSIXct, 8
- Date, 9
- days (bdtDd), 3
  - Dec (RcppBDT-constants), 10
- Feb, 9
- Feb (RcppBDT-constants), 10
  - fifth, 9
  - fifth (RcppBDT-constants), 10
  - first, 9
  - first (RcppBDT-constants), 10
  - format, Rcpp\_bdtDd-method (bdtDd), 3
  - format, Rcpp\_bdtDt-method (bdtDt), 4
  - format, Rcpp\_bdtDu-method (bdtDu), 4
  - format, Rcpp\_bdtPt-method (bdtPt), 5
  - format, Rcpp\_bdtTz-method (bdtTz), 6
  - fourth (RcppBDT-constants), 10
  - Fri (RcppBDT-constants), 10
- getDay (RcppBDT Date functions), 9
- getDayOfWeek (RcppBDT Date functions), 9
- getDayOfYear (RcppBDT Date functions), 9
- getEndOfBizWeek, 4
- getEndOfBizWeek (RcppBDT Date functions), 9
- getEndOfMonth (RcppBDT Date functions), 9
- getFirstDayOfWeekAfter (RcppBDT Date functions), 9
- getFirstDayOfWeekInMonth (RcppBDT Date functions), 9
- getIMMDate (RcppBDT Date functions), 9
- getLastDayOfWeekBefore (RcppBDT Date functions), 9
- getLastDayOfWeekInMonth (RcppBDT Date functions), 9
- getMonth (RcppBDT Date functions), 9
- getNthDayOfWeek (RcppBDT Date functions), 9
- getYear (RcppBDT Date functions), 9

hours (bdtDu), 4

Jan, 9  
Jan (RcppBDT-constants), 10  
Jul (RcppBDT-constants), 10  
Jun (RcppBDT-constants), 10

Mar (RcppBDT-constants), 10  
May (RcppBDT-constants), 10  
microseconds (bdtDu), 4  
milliseconds (bdtDu), 4  
minutes (bdtDu), 4  
Mon, 9  
Mon (RcppBDT-constants), 10

nanoseconds (bdtDu), 4  
Nov (RcppBDT-constants), 10

Oct (RcppBDT-constants), 10

Rcpp, 8  
Rcpp\_bdtDd-class (bdtDd), 3  
Rcpp\_bdtDt-class (bdtDt), 4  
Rcpp\_bdtDu-class (bdtDu), 4  
Rcpp\_bdtPt-class (bdtPt), 5  
Rcpp\_bdtTz-class (bdtTz), 6  
RcppBDT (RcppBDT-package), 2  
RcppBDT Date functions, 9  
RcppBDT-constants, 10  
RcppBDT-package, 2

Sat, 9  
Sat (RcppBDT-constants), 10  
second, 9  
second (RcppBDT-constants), 10  
seconds (bdtDu), 4  
Sep (RcppBDT-constants), 10  
show, Rcpp\_bdtDd-method (bdtDd), 3  
show, Rcpp\_bdtDt-method (bdtDt), 4  
show, Rcpp\_bdtDu-method (bdtDu), 4  
show, Rcpp\_bdtPt-method (bdtPt), 5  
show, Rcpp\_bdtTz-method (bdtTz), 6  
Sun, 9  
Sun (RcppBDT-constants), 10

third (RcppBDT-constants), 10  
Thu (RcppBDT-constants), 10  
toPOSIXct, 7, 10  
Tue (RcppBDT-constants), 10

Wed (RcppBDT-constants), 10  
weeks (bdtDd), 3