Package ‘synchronicity’

August 22, 2018

Version 1.3.5
Title Boost Mutex Functionality in R
Author Michael J. Kane <kaneplusplus@gmail.com>
Maintainer Michael J. Kane <bigmemoryauthors@gmail.com>
Contact Michael J. Kane <bigmemoryauthors@gmail.com>
Imports methods, bigmemory.sri, Rcpp, uuid
LinkingTo BH, Rcpp
Description Boost mutex functionality in R.
License LGPL-2 | Apache License 2.0
URL http://www.bigmemory.org
LazyLoad yes
SystemRequirements C++11
RoxygenNote 6.1.0
NeedsCompilation yes
Repository CRAN
Date/Publication 2018-08-21 23:00:02 UTC

R topics documented:

  synchronicity-package ........................................ 2
  attach.mutex .................................................. 3
  boost.mutex .................................................. 3
  boost.mutex-class ............................................ 4
  boost.mutex.descriptor-class ................................. 5
  describe ..................................................... 6
  describe,boost.mutex-method ................................. 6
  description .................................................. 7
  descriptor-class ............................................. 7
  is.timed, timeout ........................................... 8
  lock ......................................................... 9
synchronicity-package

This package provides support for synchronization via mutexes and may eventually support interprocess communication (ipc) and message passing.

Description

This package provides support for synchronization via mutexes and may eventually support interprocess communication (ipc) and message passing.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>synchronicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.3.4</td>
</tr>
<tr>
<td>License:</td>
<td>LGPL-3</td>
</tr>
<tr>
<td>OS_type:</td>
<td>unix</td>
</tr>
<tr>
<td>URL:</td>
<td><a href="http://www.bigmemory.org">http://www.bigmemory.org</a></td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
</tbody>
</table>

Author(s)

Michael J. Kane <bigmemoryauthors@gmail.com>

References


Boost Interprocess Library: http://www.boost.org/.

Examples

# No examples are provided here.
**attach.mutex**

Attach to an existing mutex.

**Description**

Attach to an existing mutex using either a file or description object.

**Usage**

```r
attach.mutex(obj, ...)
```

```r
## S4 method for signature 'character'
attach.mutex(obj, ...)
```

```r
## S4 method for signature 'boost.mutex.descriptor'
attach.mutex(obj, ...)
```

**Arguments**

- `obj` the descriptor object.
- `...` other arguments needed by attach.

**Value**

A mutex.

---

**boost.mutex**

Create a boost.mutex object

**Description**

This function creates a boost.mutex object.

**Usage**

```r
boost.mutex(sharedName = NULL, timeout = NULL, create=TRUE)
```

**Arguments**

- `sharedName` The name of the shared resource corresponding to the mutex. By default a universal unique identifier is supplied.
- `timeout` The amount of time (in seconds) that the mutex should try to attempt to get a lock. By default no timeout is supplied and the mutex will attempt to acquire the lock indefinitely.
- `create` Should the mutex be created or are we attaching to an existing one. Default is TRUE.
Value

This function returns a `boost::mutex` object.

Author(s)

Michael J. Kane <bigmemoryauthors@gmail.com>

See Also

`synchronicity`

Examples

```r
# Create a boost::mutex object with default resource name and no timeout.
x = boost::mutex()
rm(x)
gc()
```

---

**boost::mutex-class**

*Class “boost::mutex”*

Description

The `boost::mutex` class provides an R interface to the mutex functionality implemented in the Boost C++ library.

Objects from the Class

Unlike many R objects, objects should not be created by calls of the form `new(“boost::mutex”, ...)`. The function `boost::mutex()` is intended for the user.

Slots

- `isRead`: This is used internally to maintain state information and should not be touched by a user.
- `mutexInfoAddr`: Object of class "externalptr" which keeps track of information relevant to the mutex.

Extends

Class "`mutex`", directly.
Methods

- **describe**: signature(x = "boostmutex"): ...
- **is.timed**: signature(m = "boostmutex"): ...
- **lock.shared**: signature(m = "boostmutex"): ...
- **lock**: signature(m = "boostmutex"): ...
- **shared.name**: signature(m = "boostmutex"): ...
- **timeout**: signature(m = "boostmutex"): ...
- **unlock**: signature(m = "boostmutex"): ...

Author(s)

Michael J. Kane <bigmemoryauthors@gmail.com>

See Also

- **boost.mutex**

Examples

- `showClass("boost.mutex")`

---

**boost.mutex.descriptor-class**

An S4 class holding boost.mutex description information.

Description

Objects of class description allow users to “attach” to existing mutexes within or across processes.

Slots

- **description**: the list of description information.
describe, boost.mutex-method

**describe**

*Create descriptors to mutexes and attach*

**Description**

The `describe` function returns information that is needed to “connect” to a mutex from another process. This connection is performed by the `attachNmutex` function.

**Usage**

```
describe(x)
```

**Arguments**

- `x`: a `boost.mutex` object

**Value**

The `describe` function returns a `boost.mutex.descriptor` object.

**Author(s)**

Michael J. Kane <bigmemoryauthors@gmail.com>

**Examples**

```
m = boost.mutex()
mm = attachNmutex(describe(m))
# Now, both m and mm specify the same mutex.
rm(m)
rm(mm)
gc()
```

describe, boost.mutex-method

*Describe the boost.mutex object*

**Description**

The information required to “attach” to an existing mutex object.

**Usage**

```
## S4 method for signature 'boost.mutex'
describe(x)
```
description

Arguments

x the boost mutex object to describe.

Description

Retrieve the list of description information from a descriptor object.

Usage

description(x)

## S4 method for signature 'descriptor'
description(x)

Arguments

x the descriptor object.

Value

a list of description information.

descriptor-class

An S4 class holding mutex description information.

Description

Objects of class description allow users to “attach” to existing mutexes within or across processes.

Slots

description the list of description information.
is.timed, timeout

Timeout operations for boost.mutex objects

Description

The is.timed function tells if a boost.mutex object has a timeout. The timeout function tells how long a mutex will wait for a timeout.

Usage

\[
\text{is.timed}(m) \\
\text{timeout}(m)
\]

Arguments

\[ m \]

a boost.mutex object to get timeout information for

Value

is.timed returns TRUE if the object has a timeout and FALSE otherwise. If a timeout has been set timeout returns the number of seconds a boost.mutex object will attempt to acquire a lock and NULL otherwise.

Author(s)

Michael J. Kane <bigmemoryauthors@gmail.com>

See Also

synchronicity

Examples

\[
x = \text{boost.mutex(timeout=5)} \\
y = \text{boost.mutex()} \\
\text{print(is.timed(x))} \\
\text{print(is.timed(y))} \\
\text{print(timeout(x))} \\
\text{print(timeout(y))}
\]
Lock and Unlock a Mutex

Description

The lock and unlock functions allow a user to specify exclusive or shared access to a resource.

Usage

lock(m, ...)
lock.shared(m, ...)
unlock(m, ...)
unlock.shared(m, ...)

Arguments

m a mutex.
... options associated with the mutex being used including block which forces the mutex to return immediately after trying to acquire a lock

Details

A call to lock gives exclusive access to a resource; no other mutex may acquire a lock. A call to lock.shared allows other mutexes to acquire a shared lock on the resource. When shared lock is called while a exclusive lock has been acquired, the shared lock will block until the exclusive lock is release. Likewise, if an exclusive lock is called while a shared lock has been acquired, the exclusive lock will block until the shared lock is released.

Value

The function returns TRUE if the lock is successfully called and FALSE otherwise

Examples

m = boost.mutex()
lock(m)
# Some code that needs to be synchronized...
unlock(m)
shared

Is it a shared mutex?

Description

Tells the user if a mutex is a shared mutex. If it is not then it must be a write (exclusive) mutex.

Usage

shared(m)

## S4 method for signature 'boost.mutex'
shared(m)

Arguments

m the mutex

Value

TRUE if the mutex is shared, FALSE otherwise.

shared.name

The name of a mutex's shared resource

Description

This function returns the shared resource associated with a boost.mutex object.

Usage

shared.name(m)

Arguments

m a boost.mutex object

Value

A string specifying the shared resource associated with the given boost.mutex object.

Author(s)

Michael J. Kane <bigmemoryauthors@gmail.com>
**uuid**

**See Also**

*synchronicity*

**Examples**

```python
x = boost.mutex()
print(shared.name(x))
```

---

**uuid**

Create a universal unique identifier.

**Description**

This function creates an identifier that will be (with high probability) unique on a single machine or group of machines.

**Usage**

```python
uuid()
```

**Details**

The functions uses the boost uuid functionality.

**Value**

A unique string.

**Author(s)**

Michael J. Kane <bigmemoryauthors@gmail.com>

**References**

[http://www.boost.org/doc/libs/1_42_0/libs/uuid/uuid.html](http://www.boost.org/doc/libs/1_42_0/libs/uuid/uuid.html)

**Examples**

```python
print(uuid())
print(uuid())
```
Index

*Topic classes
  * Topic misc
    boost.mutex, 3
    describe, 6
    is.timed, timeout, 8
    shared.name, 10
  * Topic programming
    boost.mutex, 3
    describe, 6
    is.timed, timeout, 8
    shared.name, 10
  attach.mutex, 3
  attach.mutex, boost.mutex.descriptor-method
    (attach.mutex), 3
  attach.mutex, character-method
    (attach.mutex), 3
  boost.mutex, 3, 5
  boost.mutex-class, 4
  boost.mutex.descriptor-class, 5
  describe, 6
  describe, boost.mutex-method, 6
  description, 7
  description, descriptor-method
    (description), 7
  descriptor-class, 7
  is.timed, is.timed, timeout, 8
  is.timed, timeout, 8
  is.timed, boost.mutex-method
    (boost.mutex-class), 4
  lock, 9
  lock, boost.mutex-method
    (boost.mutex-class), 4
  lock.shared, boost.mutex-method
    (boost.mutex-class), 4
  mutex, 4
  mutex-class (boost.mutex-class), 4
  shared, 10
  shared, boost.mutex-method (shared), 10
  shared.name, 10
  shared.name, boost.mutex-method
    (boost.mutex-class), 4
  synchronicity, 4, 8, 11
  synchronicity (synchronicity-package), 2
  synchronicity-package, 2
  timeout (is.timed, timeout), 8
  timeout, boost.mutex-method
    (boost.mutex-class), 4
  unlock (lock), 9
  unlock, boost.mutex-method
    (boost.mutex-class), 4
  uuid, 11