LibBeamApi

0.3.8

Generated by Doxygen 1.5.6

Thu Dec 18 13:21:31 2008

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 Boapns Namespace Reference

Classes

- class Boapns
- class BoapEntry

Functions

- Boapns (BString name)
- BError getVersion (BString &version)
- BError getEntryList (BList< BoapEntry > &entryList)
- BError getEntry (BString name, BoapEntry &entry)
- BError addEntry (BoapEntry entry)
- BError delEntry (BString name)
- BError getNewName (BString &name)

Variables

- const BUInt32 apiVersion = 0

5.1.1 Function Documentation

5.1.1.1 BError Boapns::addEntry (BoapEntry *entry*)

5.1.1.2 Boapns::Boapns (BString *name*)

5.1.1.3 BError Boapns::delEntry (BString *name*)

5.1.1.4 BError Boapns::getEntry (BString *name*, BoapEntry & *entry*)

5.1.1.5 BError Boapns::getEntryList (BList< BoapEntry > & *entryList*)

5.1.1.6 BError Boapns::getNewName (BString & *name*)

5.1.1.7 BError Boapns::getVersion (BString & *version*)

5.1.2 Variable Documentation

5.1.2.1 const BUInt32 Boapns::apiVersion = 0

Chapter 6

Class Documentation

6.1 BArray< T > Class Template Reference

```
#include <BArray.h>
```

Public Member Functions

- BArray ()
- BArray (BSize size, T value=T())
- BArray (const BArray &array)

6.1.1 Detailed Description

template<class T> class BArray< T >

Template based Array class. This is based on the Standard C++ library vector class and has all of the functionality of that class.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 template<class T> BArray< T >::BArray () `[inline]`

6.1.2.2 template<class T> BArray< T >::BArray (BSize *size*, T *value* = `T()`) `[inline]`

6.1.2.3 template<class T> BArray< T >::BArray (const BArray< T > & *array*) `[inline]`

The documentation for this class was generated from the following file:

- BArray.h

6.2 BBuffer Class Reference

```
#include <BBuffer.h>
```

Public Member Functions

- BBuffer ()

  Create and manipulate a data buffer. On creation the buffer size defaults to 1024 bytes.

- ∼BBuffer ()
- int setSize (uint32_t size)

  Sets the bufer size.

- int setData (const void ∗data, uint32_t size)

  Sets buffer data resized to contain the data.

- int writeData (uint32_t pos, const void ∗data, uint32_t size)

  Writes data into buffer from offset pos.

- void ∗ data ()

  The data.

- uint32_t size ()

  Size of the buffer in bytes.

Private Attributes

- uint32_t osize
- uint32_t odatasize
- void ∗ odata

6.2.1 Constructor & Destructor Documentation

6.2.1.1 BBuffer::BBuffer ()

Create and manipulate a data buffer. On creation the buffer size defaults to 1024 bytes.

6.2.1.2 BBuffer::∼BBuffer ()

6.2.2 Member Function Documentation

6.2.2.1 int BBuffer::setSize (uint32_t *size*)

Sets the bufer size.

6.2.2.2 int BBuffer::setData (const void ∗ *data*, uint32_t *size*)

Sets buffer data resized to contain the data.

6.2.2.3 int BBuffer::writeData (uint32_t *pos*, const void $*$ *data*, uint32_t *size*)

Writes data into buffer from offset pos.

6.2.2.4 void $*$ BBuffer::data ()

The data.

6.2.2.5 uint32_t BBuffer::size ()

Size of the buffer in bytes.

6.2.3 Member Data Documentation

6.2.3.1 uint32_t BBuffer::osize `[private]`

6.2.3.2 uint32_t BBuffer::odatasize `[private]`

6.2.3.3 void$*$ BBuffer::odata `[private]`

The documentation for this class was generated from the following files:

- BBuffer.h
- BBuffer.cpp

6.3 BCond Class Reference

```
#include <BCond.h>
```

Public Member Functions

- BCond ()

  Thread conditional variable.

- ∼BCond ()
- int signal ()
- int wait ()
- int timedWait (int timeOutUs)

Private Attributes

- pthread_mutex_t omutex
- pthread_cond_t ocond

6.3.1 Constructor & Destructor Documentation

6.3.1.1 BCond::BCond ()

Thread conditional variable.

6.3.1.2 BCond::∼BCond ()

6.3.2 Member Function Documentation

6.3.2.1 int BCond::signal ()

6.3.2.2 int BCond::wait ()

6.3.2.3 int BCond::timedWait (int *timeOutUs*)

6.3.3 Member Data Documentation

6.3.3.1 pthread_mutex_t BCond::omutex `[private]`

6.3.3.2 pthread_cond_t BCond::ocond `[private]`

The documentation for this class was generated from the following files:

- BCond.h
- BCond.cpp

6.4 BCondBool Class Reference

Thread conditional boolean.

`#include <BCondInt.h>`

Public Member Functions

- BCondBool ()
- ∼BCondBool ()
- int set ()

    Set value. Wakes waiting.

- int clear ()

    Clear Value.

- int value ()

    Current value.

- int wait ()

    Wait until value is true.

- int timedWait (int timeOutUs)

    Wait until set, with timeout.

Private Attributes

- pthread_mutex_t omutex
- pthread_cond_t ocond
- int ovalue

6.4.1 Detailed Description

Thread conditional boolean.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 BCondBool::BCondBool ()

6.4.2.2 BCondBool::∼BCondBool ()

6.4.3 Member Function Documentation

6.4.3.1 int BCondBool::set ()

Set value. Wakes waiting.

6.4.3.2 int BCondBool::clear ()

Clear Value.

6.4.3.3 int BCondBool::value ()

Current value.

6.4.3.4 int BCondBool::wait ()

Wait until value is true.

6.4.3.5 int BCondBool::timedWait (int *timeOutUs*)

Wait until set, with timeout.

6.4.4 Member Data Documentation

6.4.4.1 pthread_mutex_t BCondBool::omutex `[private]`

6.4.4.2 pthread_cond_t BCondBool::ocond `[private]`

6.4.4.3 int BCondBool::ovalue `[private]`

The documentation for this class was generated from the following files:

- BCondInt.h
- BCondInt.cpp

6.5 BCondInt Class Reference

Thread conditional integer.

`#include <BCondInt.h>`

Public Member Functions

- BCondInt ()
- ∼BCondInt ()
- void setValue (int value)

    Set value.

- int increment ()

    Increment.

- int decrement ()

    Decrement.

- int value ()

    Current value.

- int wait ()

    Wait until value is 0.

- int waitIncrement (int timeOutUs=0)

    Wait until value is 0 then increment.

- int waitNotZero ()

    Wait until value is not 0.

- int waitNotZeroDecrement ()

    Wait until value is not 0 and then decrement.

- int tryNotZeroDecrement ()

    Test if value is not 0, if not zero then decrement.

- int timedWait (int timeOutUs)

    Wait for the condition, with timeout.

- void operator++ (int)
- void operator– (int)

Private Attributes

- pthread_mutex_t omutex
- pthread_cond_t ocond
- int ovalue

6.5.1 Detailed Description

Thread conditional integer.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 BCondInt::BCondInt ()

6.5.2.2 BCondInt::~BCondInt ()

6.5.3 Member Function Documentation

6.5.3.1 void BCondInt::setValue (int *value*)

Set value.

6.5.3.2 int BCondInt::increment ()

Increment.

6.5.3.3 int BCondInt::decrement ()

Decrement.

6.5.3.4 int BCondInt::value ()

Current value.

6.5.3.5 int BCondInt::wait ()

Wait until value is 0.

6.5.3.6 int BCondInt::waitIncrement (int *timeOutUs* = `0`)

Wait until value is 0 then increment.

6.5.3.7 int BCondInt::waitNotZero ()

Wait until value is not 0.

6.5.3.8 int BCondInt::waitNotZeroDecrement ()

Wait until value is not 0 and then decrement.

6.5.3.9 int BCondInt::tryNotZeroDecrement ()

Test if value is not 0, if not zero then decrement.

6.5.3.10 int BCondInt::timedWait (int *timeOutUs*)

Wait for the condition, with timeout.

6.5.3.11 void BCondInt::operator++ (int) `[inline]`

6.5.3.12 void BCondInt::operator– (int) `[inline]`

6.5.4 Member Data Documentation

6.5.4.1 pthread_mutex_t BCondInt::omutex `[private]`

6.5.4.2 pthread_cond_t BCondInt::ocond `[private]`

6.5.4.3 int BCondInt::ovalue `[private]`

The documentation for this class was generated from the following files:

- BCondInt.h
- BCondInt.cpp

6.6 BCondValue Class Reference

Thread conditional value.

```
#include <BCondInt.h>
```

Public Member Functions

- BCondValue ()
- ~BCondValue ()
- void setValue (int value)

    Set the value. Wakes waiting.

- int value ()

    Current value.

- int increment (int v=1)

    Increment. Wakes waiting.

- int decrement (int v=1)

    Decrement. Wakes waiting.

- int waitMoreThanOrEqual (int v, int decrement=0, int timeOutUs=0)

    Wait until value is at least the value given.

- int waitLessThanOrEqual (int v, int increment=0, int timeOutUs=0)

    Wait until value is equal to or below the value given.

- int waitLessThan (int v, int timeOutUs=0)

    Wait until value is equal to or below the value given.

- void operator+= (int v)

    Add to value. Wakes waiting.

- void operator-= (int v)

    Subtract from value. Wakes waiting.

- void operator++ (int)

    Increment value. Wakes waiting.

- void operator– (int)

    Decrement value. Wakes waiting.

Private Attributes

- pthread_mutex_t omutex
- pthread_cond_t ocond
- int ovalue

6.6.1 Detailed Description

Thread conditional value.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 BCondValue::BCondValue ()

6.6.2.2 BCondValue::∼BCondValue ()

6.6.3 Member Function Documentation

6.6.3.1 void BCondValue::setValue (int *value*)

Set the value. Wakes waiting.

6.6.3.2 int BCondValue::value ()

Current value.

6.6.3.3 int BCondValue::increment (int *v* = 1)

Increment. Wakes waiting.

6.6.3.4 int BCondValue::decrement (int *v* = 1)

Decrement. Wakes waiting.

6.6.3.5 int BCondValue::waitMoreThanOrEqual (int *v*, int *decrement* = 0, int *timeOutUs* = 0)

Wait until value is at least the value given.

6.6.3.6 int BCondValue::waitLessThanOrEqual (int *v*, int *increment* = 0, int *timeOutUs* = 0)

Wait until value is equal to or below the value given.

6.6.3.7 int BCondValue::waitLessThan (int *v*, int *timeOutUs* = 0)

Wait until value is equal to or below the value given.

6.6.3.8 void BCondValue::operator+= (int *v*) [inline]

Add to value. Wakes waiting.

6.6.3.9 void BCondValue::operator-= (int *v*) [inline]

Subtract from value. Wakes waiting.

6.6.3.10 void BCondValue::operator++ (int) `[inline]`

Increment value. Wakes waiting.

6.6.3.11 void BCondValue::operator– (int) `[inline]`

Decrement value. Wakes waiting.

6.6.4 Member Data Documentation

6.6.4.1 pthread_mutex_t BCondValue::omutex `[private]`

6.6.4.2 pthread_cond_t BCondValue::ocond `[private]`

6.6.4.3 int BCondValue::ovalue `[private]`

The documentation for this class was generated from the following files:

- BCondInt.h
- BCondInt.cpp

6.7 BCondWrap Class Reference

```
#include <BCondInt.h>
```

Public Member Functions

- BCondWrap ()
- ~BCondWrap ()
- void setValue (uint32_t value)

  Set the value. Wakes waiting.

- uint32_t value ()

  Current value.

- uint32_t increment (uint32_t v=1)

  Increment. Wakes waiting.

- uint32_t decrement (uint32_t v=1)

  Decrement. Wakes waiting.

- int waitMoreThanOrEqual (uint32_t v, uint32_t decrement=0, uint32_t timeOutUs=0)

  Wait until value is at least the value given.

- int waitLessThanOrEqual (uint32_t v, uint32_t increment=0, uint32_t timeOutUs=0)

  Wait until value is equal to or below the value given.

- int waitLessThan (uint32_t v, uint32_t timeOutUs=0)

  Wait until value is equal to or below the value given.

- void operator+= (int v)

  Add to value. Wakes waiting.

- void operator-= (int v)

  Subtract from value. Wakes waiting.

- void operator++ (int)

  Increment value. Wakes waiting.

- void operator– (int)

  Decrement value. Wakes waiting.

Private Member Functions

- int diff (uint32_t v)

Private Attributes

- pthread_mutex_t omutex
- pthread_cond_t ocond
- uint32_t ovalue

6.7.1 Constructor & Destructor Documentation

6.7.1.1 BCondWrap::BCondWrap ()

6.7.1.2 BCondWrap::∼BCondWrap ()

6.7.2 Member Function Documentation

6.7.2.1 void BCondWrap::setValue (uint32_t *value*)

Set the value. Wakes waiting.

6.7.2.2 uint32_t BCondWrap::value ()

Current value.

6.7.2.3 uint32_t BCondWrap::increment (uint32_t *v* = `1`)

Increment. Wakes waiting.

6.7.2.4 uint32_t BCondWrap::decrement (uint32_t *v* = `1`)

Decrement. Wakes waiting.

6.7.2.5 int BCondWrap::waitMoreThanOrEqual (uint32_t *v*, uint32_t *decrement* = `0`, uint32_t *timeOutUs* = `0`)

Wait until value is at least the value given.

6.7.2.6 int BCondWrap::waitLessThanOrEqual (uint32_t *v*, uint32_t *increment* = `0`, uint32_t *timeOutUs* = `0`)

Wait until value is equal to or below the value given.

6.7.2.7 int BCondWrap::waitLessThan (uint32_t *v*, uint32_t *timeOutUs* = `0`)

Wait until value is equal to or below the value given.

6.7.2.8 void BCondWrap::operator+= (int *v*) `[inline]`

Add to value. Wakes waiting.

6.7.2.9 void BCondWrap::operator-= (int *v*) `[inline]`

Subtract from value. Wakes waiting.

6.7.2.10 void BCondWrap::operator++ (int) `[inline]`

Increment value. Wakes waiting.

6.7.2.11 void BCondWrap::operator– (int) `[inline]`

Decrement value. Wakes waiting.

6.7.2.12 int BCondWrap::diff (uint32_t *v*) `[private]`

6.7.3 Member Data Documentation

6.7.3.1 pthread_mutex_t BCondWrap::omutex `[private]`

6.7.3.2 pthread_cond_t BCondWrap::ocond `[private]`

6.7.3.3 uint32_t BCondWrap::ovalue `[private]`

The documentation for this class was generated from the following files:

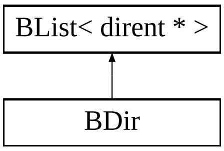
- BCondInt.h
- BCondInt.cpp

6.8 BDir Class Reference

File system directory class.

```
#include <BDir.h>
```

Inheritance diagram for BDir::



Public Member Functions

- BDir ()
- BDir (BString name)
- ∼BDir ()
- BError open (BString name)

  Reads named directory.
- BError error ()

  Current value of error.
- BError read ()

  read/re-reads directory
- void clear ()

  Clears list.
- void setWild (BString wild)

  Set wildcard filter string used on read.
- void setSort (int on)

  Set alpha sort on/off.
- BString entryName (BIter i)

  Get filename.
- struct stat entryStat (BIter i)

  Get file stats.
- struct stat64 entryStat64 (BIter i)

  Get file stats 64.

Private Attributes

- BError oerror
- BString odirname
- BString owild
- int osort

6.8.1 Detailed Description

File system directory class.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 BDir::BDir ()

6.8.2.2 BDir::BDir (BString *name*)

6.8.2.3 BDir::∼BDir ()

6.8.3 Member Function Documentation

6.8.3.1 BError BDir::open (BString *name*)

Reads named directory.

6.8.3.2 BError BDir::error ()

Current value of error.

6.8.3.3 BError BDir::read ()

read/re-reads directory

6.8.3.4 void BDir::clear () `[virtual]`

Clears list.

Reimplemented from BList< T >.

6.8.3.5 void BDir::setWild (BString *wild*)

Set wildcard filter string used on read.

6.8.3.6 void BDir::setSort (int *on*)

Set alpha sort on/off.

6.8.3.7 BString BDir::entryName (BIter *i*)

Get filename.

6.8.3.8 struct stat BDir::entryStat (BIter *i*) `[read]`

Get file stats.

6.8.3.9 struct stat64 BDir::entryStat64 (BIter *i*) `[read]`

Get file stats 64.

6.8.4 Member Data Documentation

6.8.4.1 BError BDir::oerror `[private]`

6.8.4.2 BString BDir::odirname `[private]`

6.8.4.3 BString BDir::owild `[private]`

6.8.4.4 int BDir::osort `[private]`

The documentation for this class was generated from the following files:

- BDir.h
- BDir.cpp

6.9 BEntry Class Reference

Manipulate a name value pair.

```
#include <BEntry.h>
```

Public Member Functions

- BEntry ()
- BEntry (BString name, BString value)

  Set name and value.
- BEntry (BString line)

  Set name and value from white space deliminated string.
- BString getName ()

  Get the name.
- BString getValue ()

  Get the value.
- void setLine (BString line)

  Set name and value from white space deliminated string.
- void setName (BString name)

  Set the name.
- void setValue (BString value)

  Set the value.
- BString line ()

  Return name and value as padded single string.
- void print ()

  Print name and value.

Private Attributes

- BString oname
- BString ovalue

6.9.1 Detailed Description

Manipulate a name value pair.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 BEntry::BEntry ()

6.9.2.2 BEntry::BEntry (BString *name*, BString *value*)

Set name and value.

6.9.2.3 BEntry::BEntry (BString *line*)

Set name and value from white space deliminated string.

6.9.3 Member Function Documentation

6.9.3.1 BString BEntry::getName ()

Get the name.

6.9.3.2 BString BEntry::getValue ()

Get the value.

6.9.3.3 void BEntry::setLine (BString *line*)

Set name and value from white space deliminated string.

6.9.3.4 void BEntry::setName (BString *name*)

Set the name.

6.9.3.5 void BEntry::setValue (BString *value*)

Set the value.

6.9.3.6 BString BEntry::line ()

Return name and value as padded single string.

6.9.3.7 void BEntry::print ()

Print name and value.

6.9.4 Member Data Documentation

6.9.4.1 BString BEntry::oname `[private]`

6.9.4.2 BString BEntry::ovalue `[private]`

The documentation for this class was generated from the following files:

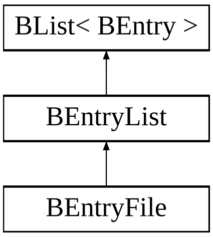
- BEntry.h
- BEntry.cpp

6.10 BEntryFile Class Reference

File of Entries.

`#include <BEntry.h>`

Inheritance diagram for BEntryFile::



Public Member Functions

- BEntryFile ()
- BEntryFile (BString filename)

    Opens entryfile.

- ∼BEntryFile ()
- int open (BString filename)

    Opens entryfile.

- int read ()

    Reads entry file and builds list.

- int write ()

    Writes list to entryfile.

- int writeList (BEntryList &l)

    Writes specified list to file.

- void clear ()

    Clears current list.

Private Attributes

- BString ofilename
- BString ocomments

6.10.1 Detailed Description

File of Entries.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 BEntryFile::BEntryFile ()

6.10.2.2 BEntryFile::BEntryFile (BString *filename*)

Opens entryfile.

6.10.2.3 BEntryFile::∼BEntryFile ()

6.10.3 Member Function Documentation

6.10.3.1 int BEntryFile::open (BString *filename*)

Opens entryfile.

6.10.3.2 int BEntryFile::read ()

Reads entry file and builds list.

6.10.3.3 int BEntryFile::write ()

Writes list to entryfile.

6.10.3.4 int BEntryFile::writeList (BEntryList & *l*)

Writes specified list to file.

6.10.3.5 void BEntryFile::clear () `[virtual]`

Clears current list.

Reimplemented from BEntryList.

6.10.4 Member Data Documentation

6.10.4.1 BString BEntryFile::ofilename `[private]`

6.10.4.2 BString BEntryFile::ocomments `[private]`

The documentation for this class was generated from the following files:

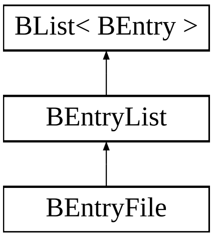
- BEntry.h
- BEntry.cpp

6.11 BEntryList Class Reference

List of Entries. Where an entry is a name value pair.

```
#include <BEntry.h>
```

Inheritance diagram for BEntryList::



Public Member Functions

- BEntryList ()
- int isSet (BString name)

  1 if name is in list and value is set
- BEntry ∗ find (BString name)

  Returns entry if name is found otherwise NULL.
- BString findValue (BString name)

  Returns value of name. Returns "" if name not found.
- int setValue (BString name, BString value)

  Set the value of name. Returns 0 if name not found.
- int setValueRaw (BString name, BString value)

  Raw setting of value without looking up existing entry.
- void deleteEntry (BString name)

  Deletes the entry.
- void print ()

  Print list.
- BString getString ()

  Return list as string. Each Entry padded and on a new line.
- void insert (BIter &i, const BEntry &item)

  Insert item before item.
- void del (BIter &i)

  Delete specified item.
- void clear ()

  Clear the list.

Private Attributes

- BIter olastPos

6.11.1 Detailed Description

List of Entries. Where an entry is a name value pair.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 BEntryList::BEntryList ()

6.11.3 Member Function Documentation

6.11.3.1 int BEntryList::isSet (BString *name*)

1 if name is in list and value is set

6.11.3.2 BEntry ∗ BEntryList::find (BString *name*)

Returns entry if name is found otherwise NULL.

6.11.3.3 BString BEntryList::findValue (BString *name*)

Returns value of name. Returns "" if name not found.

6.11.3.4 int BEntryList::setValue (BString *name*, BString *value*)

Set the value of name. Returns 0 if name not found.

6.11.3.5 int BEntryList::setValueRaw (BString *name*, BString *value*)

Raw setting of value without looking up existing entry.

6.11.3.6 void BEntryList::deleteEntry (BString *name*)

Deletes the entry.

6.11.3.7 void BEntryList::print ()

Print list.

6.11.3.8 BString BEntryList::getString ()

Return list as string. Each Entry padded and on a new line.

6.11.3.9 void BEntryList::insert (BIter & *i*, const BEntry & *item*) `[virtual]`

Insert item before item.

Reimplemented from BList< BEntry >.

6.11.3.10 void BEntryList::del (BIter & *i*) `[virtual]`

Delete specified item.

Reimplemented from BList< BEntry >.

6.11.3.11 void BEntryList::clear () `[virtual]`

Clear the list.

Reimplemented from BList< BEntry >.

Reimplemented in BEntryFile.

6.11.4 Member Data Documentation

6.11.4.1 BIter BEntryList::olastPos `[private]`

The documentation for this class was generated from the following files:

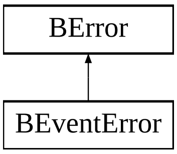
- BEntry.h
- BEntry.cpp

6.12 BError Class Reference

Error return class.

```
#include <BError.h>
```

Inheritance diagram for BError::



Public Types

- enum Type { NONE = 0, ERROR = 1 }

Public Member Functions

- BError (int errNo=NONE, BString errStr="")

  Create object.

- BError (BString errStr)

  Create with error set and error string.

- BError copy ()

  Return an independant copy.

- BError & set (int errNo, BString errStr="")

  Set error number and message.

- BError & setError (BString errStr="")

  Set error type ERROR with optional message.

- BString getString () const

  Get error message.

- int getErrorNo () const

  Get The error number.

- operator int () const

  Return error number.

Private Attributes

- int oerrNo
- BString oerrStr

6.12.1 Detailed Description

Error return class.

6.12.2 Member Enumeration Documentation

6.12.2.1 enum BError::Type

Enumerator:

NONE

ERROR

6.12.3 Constructor & Destructor Documentation

6.12.3.1 BError::BError (int *errNo* = `NONE`, BString *errStr* = "")

Create object.

6.12.3.2 BError::BError (BString *errStr*)

Create with error set and error string.

6.12.4 Member Function Documentation

6.12.4.1 BError BError::copy ()

Return an independant copy.

6.12.4.2 BError & BError::set (int *errNo*, BString *errStr* = "")

Set error number and message.

6.12.4.3 BError & BError::setError (BString *errStr* = "")

Set error type ERROR with optional message.

6.12.4.4 BString BError::getString () const

Get error message.

6.12.4.5 int BError::getErrorNo () const

Get The error number.

6.12.4.6 BError::operator int () const

Return error number.

6.12.5 Member Data Documentation

6.12.5.1 int BError::oerrNo `[private]`

6.12.5.2 BString BError::oerrStr `[private]`

The documentation for this class was generated from the following files:

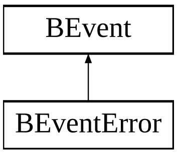
- BError.h
- BError.cpp

6.13 BEvent Class Reference

This class provides a base class for all event objects that can be sent over the events interface.

```
#include <BEvent.h>
```

Inheritance diagram for BEvent::



Public Member Functions

- BEvent (uint32_t type)
- virtual ∼BEvent ()
- uint32_t getType ()
- virtual BError getBinary (void ∗data, uint32_t &size)
- virtual BError setBinary (void ∗data, uint32_t &size)

Private Attributes

- uint32_t otype

  The event type.

6.13.1 Detailed Description

This class provides a base class for all event objects that can be sent over the events interface.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 BEvent::BEvent (uint32_t *type*)

6.13.2.2 BEvent::∼BEvent () `[virtual]`

6.13.3 Member Function Documentation

6.13.3.1 uint32_t BEvent::getType ()

6.13.3.2 BError BEvent::getBinary (void ∗ *data*, uint32_t & *size*) `[virtual]`

Reimplemented in BEventError.

6.13.3.3 BError BEvent::setBinary (void ∗ *data*, uint32_t & *size*) `[virtual]`

Reimplemented in BEventError.

6.13.4 Member Data Documentation

6.13.4.1 uint32_t BEvent::otype `[private]`

The event type.

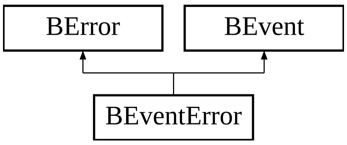
The documentation for this class was generated from the following files:

- BEvent.h
- BEvent.cpp

6.14 BEventError Class Reference

```
#include <BEvent.h>
```

Inheritance diagram for BEventError::



Public Member Functions

- BEventError (int errNo=NONE, BString errStr="")
- BError getBinary (void ∗data, uint32_t &size)
- BError setBinary (void ∗data, uint32_t &size)

6.14.1 Constructor & Destructor Documentation

6.14.1.1 BEventError::BEventError (int *errNo* = `NONE`, BString *errStr* = "")

6.14.2 Member Function Documentation

6.14.2.1 BError BEventError::getBinary (void ∗ *data*, uint32_t & *size*) `[virtual]`

Reimplemented from BEvent.

6.14.2.2 BError BEventError::setBinary (void ∗ *data*, uint32_t & *size*) `[virtual]`

Reimplemented from BEvent.

The documentation for this class was generated from the following files:

- BEvent.h
- BEvent.cpp

6.15 BEventInt Class Reference

This class provides an interface for sending simple integer events via a file descriptor. This allows threads to send events that can be picked up by the poll system call.

```
#include <BEvent.h>
```

Public Member Functions

- BEventInt ()
- ~BEventInt ()
- BError sendEvent (int event)

  Send an event.

- BError getEvent (int &event, int timeOutUs=-1)

  Receive the event.

- int getFd ()

Private Attributes

- int ofds [2]

  File descriptors for pipe.

6.15.1 Detailed Description

This class provides an interface for sending simple integer events via a file descriptor. This allows threads to send events that can be picked up by the poll system call.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 BEventInt::BEventInt ()

6.15.2.2 BEventInt::~BEventInt ()

6.15.3 Member Function Documentation

6.15.3.1 BError BEventInt::sendEvent (int *event*)

Send an event.

6.15.3.2 BError BEventInt::getEvent (int & *event*, int *timeOutUs* = `-1`)

Receive the event.

6.15.3.3 int BEventInt::getFd ()

6.15.4 Member Data Documentation

6.15.4.1 int BEventInt::ofds[2] `[private]`

File descriptors for pipe.

The documentation for this class was generated from the following files:

- BEvent.h
- BEvent.cpp

6.16 BEventPipe Class Reference

This class provides a base interface for sending events via a pipe. This allows threads to send events that can be picked up by the poll system call.

`#include <BEvent.h>`

Public Member Functions

- BEventPipe ()
- ∼BEventPipe ()
- BError sendEvent (BEvent ∗event)

  Send an event.
- BError getEvent (BEvent ∗event, int timeOutUs=-1)

  Receive the event.
- int getReceiveFd ()

  returns the receive file descriptor for the poll system call

Private Attributes

- int ofds [2]

  File descriptors for pipe.

6.16.1 Detailed Description

This class provides a base interface for sending events via a pipe. This allows threads to send events that can be picked up by the poll system call.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 BEventPipe::BEventPipe ()

6.16.2.2 BEventPipe::∼BEventPipe ()

6.16.3 Member Function Documentation

6.16.3.1 BError BEventPipe::sendEvent (BEvent ∗ *event*)

Send an event.

6.16.3.2 BError BEventPipe::getEvent (BEvent ∗ *event*, int *timeOutUs* = `-1`)

Receive the event.

6.16.3.3 int BEventPipe::getReceiveFd ()

returns the receive file descriptor for the poll system call

6.16.4 Member Data Documentation

6.16.4.1 int BEventPipe::ofds[2] `[private]`

File descriptors for pipe.

The documentation for this class was generated from the following files:

- BEvent.h
- BEvent.cpp

6.17 BFile Class Reference

File operations class.

```
#include <BFile.h>
```

Public Member Functions

- BFile ()
- BFile (BString name, BString mode)

  Create opened specifed file.

- BFile (const BFile &file)

  Create opened specified file.

- ∼BFile ()
- BError open (BString name, BString mode)

  Open file.

- BError open (FILE ∗file)

  Assign object to opened file handle.

- BError close ()

  Close file.

- BError error ()

  Returns current error state.

- FILE ∗ getFd ()

  File descriptor.

- int length ()

  File size in bytes.

- int setVBuf (char ∗buf, int mode, size_t size)

  Set stream buffering options.

- int read (void ∗buf, int nbytes)

  Read from file.

- int readString (BString &str)

  Read string. (ref fgets).

- int write (const void ∗buf, int nbytes)

  Write to file.

- int writeString (const BString &str)

  Write string to file.

- int seek (int pos, int whence)

Set seek position.

- int printf (const char ∗fmt,...)

  Formated print into the file.

- BFile & operator= (const BFile &file)

Private Attributes

- FILE ∗ ofile
- BString ofileName
- BString omode
- BError oerror

6.17.1 Detailed Description

File operations class.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 BFile::BFile ()

6.17.2.2 BFile::BFile (BString *name*, BString *mode*)

Create opened specifed file.

6.17.2.3 BFile::BFile (const BFile & *file*)

Create opened specified file.

6.17.2.4 BFile::∼BFile ()

6.17.3 Member Function Documentation

6.17.3.1 BError BFile::open (BString *name*, BString *mode*)

Open file.

6.17.3.2 BError BFile::open (FILE ∗ *file*)

Assign object to opened file handle.

6.17.3.3 BError BFile::close ()

Close file.

6.17.3.4 BError BFile::error ()

Returns current error state.

6.17.3.5 FILE ∗ BFile::getFd ()

File descriptor.

6.17.3.6 int BFile::length ()

File size in bytes.

6.17.3.7 int BFile::setVBuf (char ∗ *buf*, int *mode*, size_t *size*)

Set stream buffering options.

6.17.3.8 int BFile::read (void ∗ *buf*, int *nbytes*)

Read from file.

6.17.3.9 int BFile::readString (BString & *str*)

Read string. (ref fgets).

6.17.3.10 int BFile::write (const void ∗ *buf*, int *nbytes*)

Write to file.

6.17.3.11 int BFile::writeString (const BString & *str*)

Write string to file.

6.17.3.12 int BFile::seek (int *pos*, int *whence*)

Set seek position.

6.17.3.13 int BFile::printf (const char ∗ *fmt*, ...)

Formated print into the file.

6.17.3.14 BFile & BFile::operator= (const BFile & *file*)

6.17.4 Member Data Documentation

6.17.4.1 FILE∗ BFile::ofile `[private]`

6.17.4.2 BString BFile::ofileName `[private]`

6.17.4.3 BString BFile::omode `[private]`

6.17.4.4 BError BFile::oerror `[private]`

The documentation for this class was generated from the following files:

- BFile.h
- BFile.cpp

6.18 BIter Class Reference

Iterator for BList.

`#include <BList.h>`

Public Member Functions

- BIter (void ∗i=0)
- operator void ∗ ()
- int operator== (const BIter &i)

Private Attributes

- void ∗ oi

6.18.1 Detailed Description

Iterator for BList.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 BIter::BIter (void ∗ *i* = 0) `[inline]`

6.18.3 Member Function Documentation

6.18.3.1 BIter::operator void ∗ () `[inline]`

6.18.3.2 int BIter::operator== (const BIter & *i*) `[inline]`

6.18.4 Member Data Documentation

6.18.4.1 void∗ BIter::oi `[private]`

The documentation for this class was generated from the following file:

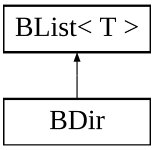
- BList.h

6.19 BList< T > Class Template Reference

Template based list class.

`#include <BList.h>`

Inheritance diagram for BList< T >::



Public Types

- typedef int(* SortFunc)(T &a, T &b)

    Prototype for sorting function.

Public Member Functions

- BList ()
- BList (const BList< T > &l)
- virtual ∼BList ()
- void start (BIter &i) const

    Iterator to start of list.

- BIter begin () const

    Iterator for start of list.

- BIter end () const

    Iterator for end of list.

- BIter end (BIter &i) const

    Iterator for end of list.

- void next (BIter &i) const

    Iterator for next item in list.

- void prev (BIter &i)

    Iterator for previous item in list.

- BIter goTo (int pos)

    Iterator for pos item in list.

- int position (BIter i)

    Postition in list item with iterator i.

- unsigned int number ()

Number of items in list.

- int isEnd (BIter i) const

  True if iterator refers to last item.

- T & front ()

  Get first item in list.

- T & rear ()

  Get last item in list.

- T & get (BIter i)

  Get item specified by iterator in list.

- const T & get (BIter i) const

  Get item specified by iterator in list.

- void append (const T &item)

  Append item to list.

- virtual void insert (BIter &i, const T &item)

  Insert item before item.

- void insertAfter (BIter &i, const T &item)

  Insert item after item.

- virtual void clear ()

  Clear the list.

- virtual void del (BIter &i)

  Delete specified item.

- void deleteLast ()

  Delete last item.

- void deleteFirst ()

  Delete fisrt item.

- void push (const T &i)

  Push item onto list.

- T pop ()

  Pop item from list deleteing item.

- void queueAdd (const T &i)

  Add item to end of list.

- T queueGet ()

  Get item from front of list deleteing item.

- void append (const BList< T > &l)

  Append list to list.

- void swap (BIter i1, BIter i2)

  Swap two items in list.

- void sort ()

  Sort list based on get(i) values.

- void sort (SortFunc func)

  Sort list based on Sort func.

- BList< T > & operator= (const BList< T > &l)
- T & operator[ ] (int i)
- const T & operator[ ] (int i) const
- T & operator[ ] (BIter i)
- const T & operator[ ] (BIter i) const
- BList< T > operator+ (const BList< T > &l) const

Protected Member Functions

- virtual Node ∗ nodeGet (BIter i)
- virtual const Node ∗ nodeGet (BIter i) const
- virtual Node ∗ nodeCreate (const T &item)

Protected Attributes

- Node ∗ onodes
- unsigned int olength

Private Member Functions

- virtual Node ∗ nodeCreate ()

Classes

- class Node

6.19.1 Detailed Description

template<class T> class BList< T >

Template based list class.

6.19.2 Member Typedef Documentation

6.19.2.1 template<class T> typedef int(∗ BList< T >::SortFunc)(T &a, T &b)

Prototype for sorting function.

6.19.3 Constructor & Destructor Documentation

6.19.3.1 template<class T> BList< T >::BList () `[inline]`

6.19.3.2 template<class T> BList< T >::BList (const BList< T > & *l*) `[inline]`

6.19.3.3 template<class T> BList< T >::~BList () `[inline, virtual]`

6.19.4 Member Function Documentation

6.19.4.1 template<class T> void BList< T >::start (BIter & *i*) const `[inline]`

Iterator to start of list.

6.19.4.2 template<class T> BIter BList< T >::begin () const `[inline]`

Iterator for start of list.

6.19.4.3 template<class T> BIter BList< T >::end () const `[inline]`

Iterator for end of list.

6.19.4.4 template<class T> BIter BList< T >::end (BIter & *i*) const `[inline]`

Iterator for end of list.

6.19.4.5 template<class T> void BList< T >::next (BIter & *i*) const `[inline]`

Iterator for next item in list.

6.19.4.6 template<class T> void BList< T >::prev (BIter & *i*) `[inline]`

Iterator for previous item in list.

6.19.4.7 template<class T> BIter BList< T >::goTo (int *pos*) `[inline]`

Iterator for pos item in list.

6.19.4.8 template<class T> int BList< T >::position (BIter *i*) `[inline]`

Postition in list item with iterator i.

6.19.4.9 template<class T> unsigned int BList< T >::number () `[inline]`

Number of items in list.

6.19.4.10 template<class T> int BList< T >::isEnd (BIter *i*) const `[inline]`

True if iterator refers to last item.

6.19.4.11 template<class T> T & BList< T >::front () `[inline]`

Get first item in list.

6.19.4.12 template<class T> T & BList< T >::rear () `[inline]`

Get last item in list.

6.19.4.13 template<class T> T & BList< T >::get (BIter *i*) `[inline]`

Get item specified by iterator in list.

6.19.4.14 template<class T> const T & BList< T >::get (BIter *i*) const `[inline]`

Get item specified by iterator in list.

6.19.4.15 template<class T> void BList< T >::append (const T & *item*) `[inline]`

Append item to list.

6.19.4.16 template<class T> void BList< T >::insert (BIter & *i*, const T & *item*) `[inline, virtual]`

Insert item before item.

Reimplemented in BEntryList.

6.19.4.17 template<class T> void BList< T >::insertAfter (BIter & *i*, const T & *item*) `[inline]`

Insert item after item.

6.19.4.18 template<class T> void BList< T >::clear () `[inline, virtual]`

Clear the list.

Reimplemented in BDir, BEntryList, and BEntryFile.

6.19.4.19 template<class T> void BList< T >::del (BIter & *i*) `[inline, virtual]`

Delete specified item.

Reimplemented in BEntryList.

6.19.4.20 template<class T> void BList< T >::deleteLast () `[inline]`

Delete last item.

6.19.4.21 template<class T> void BList< T >::deleteFirst () `[inline]`

Delete fisrt item.

6.19.4.22 template<class T> void BList< T >::push (const T & *i*) `[inline]`

Push item onto list.

6.19.4.23 template<class T> T BList< T >::pop () `[inline]`

Pop item from list deleteing item.

6.19.4.24 template<class T> void BList< T >::queueAdd (const T & *i*) `[inline]`

Add item to end of list.

6.19.4.25 template<class T> T BList< T >::queueGet () `[inline]`

Get item from front of list deleteing item.

6.19.4.26 template<class T> void BList< T >::append (const BList< T > & *l*) `[inline]`

Append list to list.

6.19.4.27 template<class T> void BList< T >::swap (BIter *i1*, BIter *i2*) `[inline]`

Swap two items in list.

6.19.4.28 template<class T> void BList< T >::sort () `[inline]`

Sort list based on get(i) values.

6.19.4.29 template<class T> void BList< T >::sort (SortFunc *func*) `[inline]`

Sort list based on Sort func.

6.19.4.30 template<class T> BList< T > & BList< T >::operator= (const BList< T > & *l*) `[inline]`

6.19.4.31]

template<class T> T & **BList**< T >::operator[ ] (int *i*) `[inline]`

6.19.4.32]

template<class T> const T & **BList**< T >::operator[ ] (int *i*) const `[inline]`

6.19.4.33]

template<class T> T & **BList**< T >::operator[ ] (**BIter** *i*) `[inline]`

6.19.4.34]

template<class T> const T & **BList**< T >::operator[ ] (**BIter** *i*) const `[inline]`

6.19.4.35 template<class T> BList< T > BList< T >::operator+ (const BList< T > & *l*) const `[inline]`

6.19.4.36 template<class T> BList< T >::Node ∗ BList< T >::nodeGet (BIter *i*) `[inline, protected, virtual]`

6.19.4.37 template<class T> const BList< T >::Node ∗ BList< T >::nodeGet (BIter *i*) const `[inline, protected, virtual]`

6.19.4.38 template<class T> BList< T >::Node ∗ BList< T >::nodeCreate (const T & *item*) `[inline, protected, virtual]`

6.19.4.39 template<class T> BList< T >::Node ∗ BList< T >::nodeCreate () `[inline, private, virtual]`

6.19.5 Member Data Documentation

6.19.5.1 template<class T> Node∗ BList< T >::onodes `[protected]`

6.19.5.2 template<class T> unsigned int BList< T >::olength `[protected]`

The documentation for this class was generated from the following files:

- BList.h
- BList_func.h

6.20 BList< T >::Node Class Reference

```
#include <BList.h>
```

Public Member Functions

- Node (const T &i)

Public Attributes

- Node ∗ next
- Node ∗ prev
- T item

template<class T> class BList< T >::Node

6.20.1 Constructor & Destructor Documentation

6.20.1.1 template<class T> BList< T >::Node::Node (const T & *i*) `[inline]`

6.20.2 Member Data Documentation

6.20.2.1 template<class T> Node∗ BList< T >::Node::next

6.20.2.2 template<class T> Node∗ BList< T >::Node::prev

6.20.2.3 template<class T> T BList< T >::Node::item

The documentation for this class was generated from the following file:

- BList.h

6.21 BMutex Class Reference

Mutex class.

```
#include <BMutex.h>
```

Public Member Functions

- BMutex ()
- BMutex (const BMutex &mutex)
- ~BMutex ()
- int lock ()

    Set lock, wait in necessary.

- int unlock ()

    Unlock the lock.

- int tryLock ()

    Test the lock.

- BMutex & operator= (const BMutex &mutex)

Private Attributes

- pthread_mutex_t omutex

6.21.1 Detailed Description

Mutex class.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 BMutex::BMutex ()

6.21.2.2 BMutex::BMutex (const BMutex & *mutex*)

6.21.2.3 BMutex::~BMutex ()

6.21.3 Member Function Documentation

6.21.3.1 int BMutex::lock ()

Set lock, wait in necessary.

6.21.3.2 int BMutex::unlock ()

Unlock the lock.

6.21.3.3 int BMutex::tryLock ()

Test the lock.

6.21.3.4 BMutex & BMutex::operator= (const BMutex & *mutex*)

6.21.4 Member Data Documentation

6.21.4.1 pthread_mutex_t BMutex::omutex `[private]`

The documentation for this class was generated from the following files:

- BMutex.h
- BMutex.cpp

6.22 BNameValue< T > Class Template Reference

```
#include <BNameValue.h>
```

Public Member Functions

- BNameValue ()
- BNameValue (BString name, const T &value)
- BString getName ()
- T & getValue ()

Private Attributes

- BString oname
- T ovalue

template<class T> class BNameValue< T >

6.22.1 Constructor & Destructor Documentation

6.22.1.1 template<class T> BNameValue< T >::BNameValue () `[inline]`

6.22.1.2 template<class T> BNameValue< T >::BNameValue (BString *name*, const T & *value*) `[inline]`

6.22.2 Member Function Documentation

6.22.2.1 template<class T> BString BNameValue< T >::getName () `[inline]`

6.22.2.2 template<class T> T& BNameValue< T >::getValue () `[inline]`

6.22.3 Member Data Documentation

6.22.3.1 template<class T> BString BNameValue< T >::oname `[private]`

6.22.3.2 template<class T> T BNameValue< T >::ovalue `[private]`

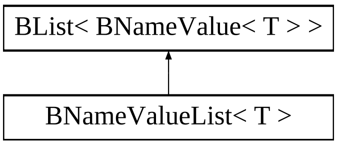
The documentation for this class was generated from the following file:

- BNameValue.h

6.23 BNameValueList< T > Class Template Reference

`#include <BNameValue.h>`

Inheritance diagram for BNameValueList< T >::



Public Member Functions

- T ∗ find (BString name)

template<class T> class BNameValueList< T >

6.23.1 Member Function Documentation

6.23.1.1 template<class T> T∗ BNameValueList< T >::find (BString *name*) `[inline]`

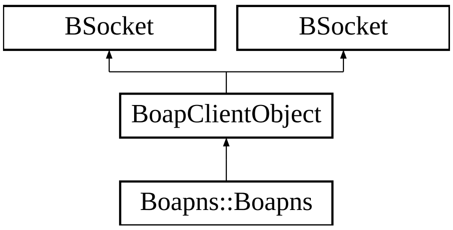
The documentation for this class was generated from the following file:

- BNameValue.h

6.24 BoapClientObject Class Reference

`#include <BoapSimple.h>`

Inheritance diagram for BoapClientObject::



Public Member Functions

- BoapClientObject (BString name="")
- BError connectService (BString name)

  Connects to the named service.

- BError disconnectService ()

  Disconnects from the named service.

- BString getServiceName ()

  Get the name of the service.

- BError ping (BUInt32 &apiVersion)

  Pings the connection and finds the remotes version number.

- BError setConnectionPriority (BoapPriority priority)

  Sets the connection priority.

- void setMaxLength (BUInt32 maxLength)

  Sets the maximum packet length.

- void setTimeout (int timeout)

  Sets the timeout in micro seconds. -1 is wait indefinitely.

- BoapClientObject (BString name)
- BError connectService (BString name)

Protected Member Functions

- BError pingLocked (BUInt32 &apiVersion)
- BError checkApiVersion ()
- BError performCall (BoapPacket &tx, BoapPacket &rx)

  Performs a RPC call to the named service.

- BError performSend (BoapPacket &tx)

Performs a send to the named service.

- BError performRecv (BoapPacket &rx)

  Performs a receive.

- BError performSend (BoapPacket &tx)
- BError performRecv (BoapPacket &rx)
- BError performCall (BoapPacket &tx, BoapPacket &rx)

Protected Attributes

- BString oname
- BUInt32 oapiVersion
- BoapPriority opriority
- BoapService oservice
- int oconnected
- BUInt32 omaxLength
- BoapPacket otx
- BoapPacket orx
- BMutex olock
- int otimeout
- int oreconnect

6.24.1 Constructor & Destructor Documentation

6.24.1.1 BoapClientObject::BoapClientObject (BString *name* = "")

6.24.1.2 BoapClientObject::BoapClientObject (BString *name*)

6.24.2 Member Function Documentation

6.24.2.1 BError BoapClientObject::connectService (BString *name*)

Connects to the named service.

6.24.2.2 BError BoapClientObject::disconnectService ()

Disconnects from the named service.

6.24.2.3 BString BoapClientObject::getServiceName ()

Get the name of the service.

6.24.2.4 BError BoapClientObject::ping (BUInt32 & *apiVersion*)

Pings the connection and finds the remotes version number.

6.24.2.5 BError BoapClientObject::setConnectionPriority (BoapPriority *priority*)

Sets the connection priority.

6.24.2.6 void BoapClientObject::setMaxLength (BUInt32 *maxLength*)

Sets the maximum packet length.

6.24.2.7 void BoapClientObject::setTimeout (int *timeout*)

Sets the timeout in micro seconds. -1 is wait indefinitely.

6.24.2.8 BError BoapClientObject::pingLocked (BUInt32 & *apiVersion*) `[protected]`

6.24.2.9 BError BoapClientObject::checkApiVersion () `[protected]`

6.24.2.10 BError BoapClientObject::performCall (BoapPacket & *tx*, BoapPacket & *rx*) `[protected]`

Performs a RPC call to the named service.

6.24.2.11 BError BoapClientObject::performSend (BoapPacket & *tx*) `[protected]`

Performs a send to the named service.

6.24.2.12 BError BoapClientObject::performRecv (BoapPacket & *rx*) `[protected]`

Performs a receive.

6.24.2.13 BError BoapClientObject::connectService (BString *name*)

6.24.2.14 BError BoapClientObject::performSend (BoapPacket & *tx*) `[protected]`

6.24.2.15 BError BoapClientObject::performRecv (BoapPacket & *rx*) `[protected]`

6.24.2.16 BError BoapClientObject::performCall (BoapPacket & *tx*, BoapPacket & *rx*) `[protected]`

6.24.3 Member Data Documentation

6.24.3.1 BString BoapClientObject::oname `[protected]`

6.24.3.2 BUInt32 BoapClientObject::oapiVersion `[protected]`

6.24.3.3 BoapPriority BoapClientObject::opriority `[protected]`

6.24.3.4 BoapService BoapClientObject::oservice `[protected]`

6.24.3.5 int BoapClientObject::oconnected `[protected]`

6.24.3.6 BUInt32 BoapClientObject::omaxLength `[protected]`

6.24.3.7 BoapPacket BoapClientObject::otx `[protected]`

6.24.3.8 BoapPacket BoapClientObject::orx `[protected]`

6.24.3.9 BMutex BoapClientObject::olock `[protected]`

6.24.3.10 int BoapClientObject::otimeout `[protected]`

6.24.3.11 int BoapClientObject::oreconnect `[protected]`

The documentation for this class was generated from the following files:

- Boap.h
- BoapSimple.h
- Boap.cpp
- BoapSimple.cc

6.25 Boapns::BoapEntry Class Reference

```
#include <BoapnsD.h>
```

Public Member Functions

- BoapEntry ()
- BoapEntry (BString pname, BString phostName, BList< BString > paddressList, UInt32 pport, UInt32 pservice)

Public Attributes

- BString name
- BString hostName
- BList< BString > addressList
- UInt32 port
- UInt32 service

6.25.1 Constructor & Destructor Documentation

6.25.1.1 Boapns::BoapEntry::BoapEntry ()

6.25.1.2 Boapns::BoapEntry::BoapEntry (BString *pname*, BString *phostName*, BList< BString > *paddressList*, UInt32 *pport*, UInt32 *pservice*)

6.25.2 Member Data Documentation

6.25.2.1 BString Boapns::BoapEntry::name

6.25.2.2 BString Boapns::BoapEntry::hostName

6.25.2.3 BList<BString> Boapns::BoapEntry::addressList

6.25.2.4 UInt32 Boapns::BoapEntry::port

6.25.2.5 UInt32 Boapns::BoapEntry::service

The documentation for this class was generated from the following files:

- BoapnsD.h
- BoapnsD.cc

6.26 BoapFuncEntry Class Reference

```
#include <BoapSimple.h>
```

Public Member Functions

- BoapFuncEntry (int cmd, BoapFunc func)
- BoapFuncEntry (int cmd, BoapFunc func)

Public Attributes

- UInt32 ocmd
- BoapFunc ofunc

6.26.1 Constructor & Destructor Documentation

6.26.1.1 BoapFuncEntry::BoapFuncEntry (int *cmd*, BoapFunc *func*)

6.26.1.2 BoapFuncEntry::BoapFuncEntry (int *cmd*, BoapFunc *func*)

6.26.2 Member Data Documentation

6.26.2.1 UInt32 BoapFuncEntry::ocmd

6.26.2.2 BoapFunc BoapFuncEntry::ofunc

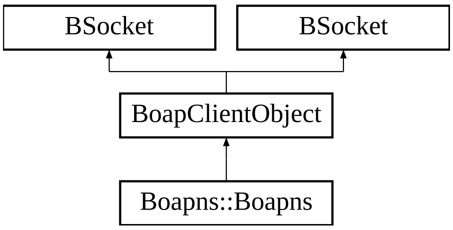
The documentation for this class was generated from the following files:

- Boap.h
- BoapSimple.h
- Boap.cpp
- BoapSimple.cc

6.27 Boapns::Boapns Class Reference

```
#include <BoapnsC.h>
```

Inheritance diagram for Boapns::Boapns::



Public Member Functions

- Boapns (BString name="")
- BError getVersion (BString &version)
- BError getEntryList (BList< BoapEntry > &entryList)
- BError getEntry (BString name, BoapEntry &entry)
- BError addEntry (BoapEntry entry)
- BError delEntry (BString name)
- BError getNewName (BString &name)

6.27.1 Constructor & Destructor Documentation

6.27.1.1 Boapns::Boapns::Boapns (BString *name* = "")

6.27.2 Member Function Documentation

6.27.2.1 BError Boapns::Boapns::getVersion (BString & *version*)

6.27.2.2 BError Boapns::Boapns::getEntryList (BList< BoapEntry > & *entryList*)

6.27.2.3 BError Boapns::Boapns::getEntry (BString *name*, BoapEntry & *entry*)

6.27.2.4 BError Boapns::Boapns::addEntry (BoapEntry *entry*)

6.27.2.5 BError Boapns::Boapns::delEntry (BString *name*)

6.27.2.6 BError Boapns::Boapns::getNewName (BString & *name*)

The documentation for this class was generated from the following file:

- BoapnsC.h

6.28 BoapPacket Class Reference

```
#include <BoapSimple.h>
```

Public Member Functions

- BoapPacket ()
- ∼BoapPacket ()
- int resize (int size)
- BError setData (void ∗data, int nbytes)
- int nbytes ()
- char ∗ data ()
- int peekHead (BoapPacketHead &head)
- UInt32 getCmd ()
- int pushHead (BoapPacketHead &head)
- int push (Int8 v)
- int push (UInt8 v)
- int push (Int16 v)
- int push (UInt16 v)
- int push (Int32 v)
- int push (UInt32 v)
- int push (Int64 v)
- int push (UInt64 v)
- int push (const BString &v)
- int push (Double v)
- int push (const BError &v)
- int push (UInt32 nBytes, const void ∗data, char ∗swapType="1")
- int popHead (BoapPacketHead &head)
- int pop (Int8 &v)
- int pop (UInt8 &v)
- int pop (Int16 &v)
- int pop (UInt16 &v)
- int pop (Int32 &v)
- int pop (UInt32 &v)
- int pop (Int64 &v)
- int pop (UInt64 &v)
- int pop (BString &v)
- int pop (Double &v)
- int pop (BError &v)
- int pop (UInt32 nBytes, void ∗data, char ∗swapType="1")
- BoapPacket ()
- ∼BoapPacket ()
- int resize (int size)
- BError setData (void ∗data, int nbytes)
- int nbytes ()
- char ∗ data ()
- int pushHead (BoapPacketHead &head)
- int push (Int8 v)
- int push (UInt8 v)

- int push (Int16 v)
- int push (UInt16 v)
- int push (Int32 v)
- int push (UInt32 v)
- int push (BString &v)
- int push (Double v)
- int push (BError &v)
- int push (UInt32 nBytes, const void ∗data)
- int popHead (BoapPacketHead &head)
- int pop (Int8 &v)
- int pop (UInt8 &v)
- int pop (Int16 &v)
- int pop (UInt16 &v)
- int pop (Int32 &v)
- int pop (UInt32 &v)
- int pop (BString &v)
- int pop (Double &v)
- int pop (BError &v)
- int pop (UInt32 nBytes, void ∗data)

Private Member Functions

- void copyWithSwap (void ∗dst, const void ∗src, UInt32 nBytes, char ∗swapType)
- void updateLen ()
- void updateLen ()

Private Attributes

- int osize
- int onbytes
- char ∗ odata
- int opos

6.28.1 Constructor & Destructor Documentation

6.28.1.1 BoapPacket::BoapPacket ()

6.28.1.2 BoapPacket::∼BoapPacket ()

6.28.1.3 BoapPacket::BoapPacket ()

6.28.1.4 BoapPacket::∼BoapPacket ()

6.28.2 Member Function Documentation

6.28.2.1 int BoapPacket::resize (int *size*)

6.28.2.2 BError BoapPacket::setData (void ∗ *data*, int *nbytes*)

6.28.2.3 int BoapPacket::nbytes ()

6.28.2.4 char ∗ BoapPacket::data ()

6.28.2.5 int BoapPacket::peekHead (BoapPacketHead & *head*)

6.28.2.6 UInt32 BoapPacket::getCmd ()

6.28.2.7 int BoapPacket::pushHead (BoapPacketHead & *head*)

6.28.2.8 int BoapPacket::push (Int8 *v*)

6.28.2.9 int BoapPacket::push (UInt8 *v*)

6.28.2.10 int BoapPacket::push (Int16 *v*)

6.28.2.11 int BoapPacket::push (UInt16 *v*)

6.28.2.12 int BoapPacket::push (Int32 *v*)

6.28.2.13 int BoapPacket::push (UInt32 *v*)

6.28.2.14 int BoapPacket::push (Int64 *v*)

6.28.2.15 int BoapPacket::push (UInt64 *v*)

6.28.2.16 int BoapPacket::push (const BString & *v*)

6.28.2.17 int BoapPacket::push (Double *v*)

6.28.2.18 int BoapPacket::push (const BError & *v*)

6.28.2.19 int BoapPacket::push (UInt32 *nBytes*, const void ∗ *data*, char ∗ *swapType* = "1")

6.28.2.20 int BoapPacket::popHead (BoapPacketHead & *head*)

6.28.2.21 int BoapPacket::pop (Int8 & *v*)

6.28.2.22 int BoapPacket::pop (UInt8 & *v*)

6.28.2.23 int BoapPacket::pop (Int16 & *v*)

6.28.2.24 int BoapPacket::pop (UInt16 & *v*)

6.28.2.25 int BoapPacket::pop (Int32 & *v*)

- Boap.h
- BoapSimple.h
- Boap.cpp
- BoapSimple.cc

6.29 BoapPacketHead Struct Reference

```
#include <BoapSimple.h>
```

Public Attributes

- UInt32 type
- UInt32 length
- UInt32 service
- UInt32 cmd
- BoapType type
- BoapService service
- UInt32 reserved [12]

6.29.1 Member Data Documentation

6.29.1.1 UInt32 BoapPacketHead::type

6.29.1.2 UInt32 BoapPacketHead::length

6.29.1.3 UInt32 BoapPacketHead::service

6.29.1.4 UInt32 BoapPacketHead::cmd

6.29.1.5 BoapType BoapPacketHead::type

6.29.1.6 BoapService BoapPacketHead::service

6.29.1.7 UInt32 BoapPacketHead::reserved[12]

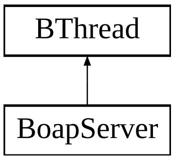
The documentation for this struct was generated from the following files:

- Boap.h
- BoapSimple.h

6.30 BoapServer Class Reference

```
#include <BoapSimple.h>
```

Inheritance diagram for BoapServer::



Public Types

- enum { NOTHREADS = 0, THREADED = 1 }

Public Member Functions

- BoapServer ()
- ∼BoapServer ()
- BError init (BString boapNsHost="", int threaded=0, int isBoapns=0)
- BError run (int inThread=0)
- BError processEvent (BoapPacket &rx)
- BError addObject (BoapServiceObject ∗object)
- BError process (BoapServerConnection ∗conn, BoapPacket &rx, BoapPacket &tx)
- BError sendEvent (BoapPacket &tx)
- BSocket & getSocket ()
- BSocket & getEventSocket ()
- BError processEvent (int fd)
- BString getHostName ()
- void clientGone (BoapServerConnection ∗client)
- int getConnectionsNumber ()
- BoapServer ()
- BError init (int boapNs=0)
- BError run ()
- BError processEvent (BoapPacket &rx)
- BError addObject (BoapServiceObject ∗object)
- BError process (int fd)
- BError sendEvent (BoapPacket &tx)
- BSocket & getSocket ()
- BSocket & getEventSocket ()
- BError processEvent (int fd)
- BString getHostName ()

Private Member Functions

- void ∗ function ()

Private Attributes

- int othreaded
- int oisBoapns
- Boapns::Boapns ∗ oboapns
- BList< BoapServerConnection ∗ > oclients
- BEventInt oclientGoneEvent
- BList< BoapServiceEntry > oservices
- BPoll opoll
- BSocket onet
- BSocket onetEvent
- BSocketAddressINET onetEventAddress
- BString ohostName
- int oboapNs
- BoapPacket orx
- BoapPacket otx

6.30.1 Member Enumeration Documentation

6.30.1.1 anonymous enum

Enumerator:

NOTHREADS

THREADED

6.30.2 Constructor & Destructor Documentation

6.30.2.1 BoapServer::BoapServer ()

6.30.2.2 BoapServer::∼BoapServer ()

6.30.2.3 BoapServer::BoapServer ()

6.30.3 Member Function Documentation

6.30.3.1 BError BoapServer::init (BString *boapNsHost* = "", int *threaded* = 0, int *isBoapns* = 0)

6.30.3.2 BError BoapServer::run (int *inThread* = 0)

6.30.3.3 BError BoapServer::processEvent (BoapPacket & *rx*)

6.30.3.4 BError BoapServer::addObject (BoapServiceObject ∗ *object*)

6.30.3.5 BError BoapServer::process (BoapServerConnection ∗ *conn*, BoapPacket & *rx*, BoapPacket & *tx*)

6.30.3.6 BError BoapServer::sendEvent (BoapPacket & *tx*)

6.30.3.7 BSocket & BoapServer::getSocket ()

6.30.3.8 BSocket & BoapServer::getEventSocket ()

6.30.3.9 BError BoapServer::processEvent (int *fd*)

6.30.3.10 BString BoapServer::getHostName ()

6.30.3.11 void BoapServer::clientGone (BoapServerConnection ∗ *client*)

6.30.3.12 int BoapServer::getConnectionsNumber ()

6.30.3.13 void ∗ BoapServer::function () `[private, virtual]`

Reimplemented from BThread.

6.30.3.14 BError BoapServer::init (int *boapNs* = 0)

6.30.3.15 BError BoapServer::run ()

6.30.3.16 BError BoapServer::processEvent (BoapPacket & *rx*)

6.30.3.17 BError BoapServer::addObject (BoapServiceObject ∗ *object*)

6.30.3.18 BError BoapServer::process (int *fd*)

6.30.3.19 BError BoapServer::sendEvent (BoapPacket & *tx*)

6.30.3.20 BSocket& BoapServer::getSocket ()

6.30.3.21 BSocket& BoapServer::getEventSocket ()

6.30.3.22 BError BoapServer::processEvent (int *fd*)

6.30.3.23 BString BoapServer::getHostName ()

6.30.4 Member Data Documentation

6.30.4.1 int BoapServer::othreaded `[private]`

6.30.4.2 int BoapServer::oisBoapns `[private]`

6.30.4.3 Boapns::Boapns∗ BoapServer::oboapns `[private]`

6.30.4.4 BList<BoapServerConnection∗> BoapServer::oclients `[private]`

6.30.4.5 BEventInt BoapServer::oclientGoneEvent `[private]`

6.30.4.6 BList< BoapServiceEntry > BoapServer::oservices `[private]`

6.30.4.7 BPoll BoapServer::opoll `[private]`

6.30.4.8 BSocket BoapServer::onet `[private]`

6.30.4.9 BSocket BoapServer::onetEvent `[private]`

6.30.4.10 BSocketAddressINET BoapServer::onetEventAddress `[private]`

6.30.4.11 BString BoapServer::ohostName `[private]`

6.30.4.12 int BoapServer::oboapNs `[private]`

6.30.4.13 BoapPacket BoapServer::orx `[private]`

6.30.4.14 BoapPacket BoapServer::otx `[private]`

The documentation for this class was generated from the following files:

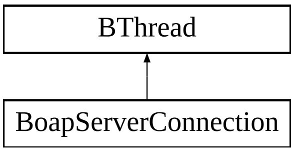
- Boap.h

- BoapSimple.h
- Boap.cpp
- BoapSimple.cc

6.31 BoapServerConnection Class Reference

`#include <Boap.h>`

Inheritance diagram for BoapServerConnection::



Public Member Functions

- BoapServerConnection (BoapServer &boapServer, int fd)
- BError process ()
- BSocket & getSocket ()
- void setMaxLength (BUInt32 maxLength)

Private Member Functions

- void ∗ function ()

Private Attributes

- BoapServer & oboapServer
- BSocket osocket
- BoapPacket orx
- BoapPacket otx
- BUInt32 omaxLength

6.31.1 Constructor & Destructor Documentation

6.31.1.1 BoapServerConnection::BoapServerConnection (BoapServer & *boapServer*, int *fd*)

6.31.2 Member Function Documentation

6.31.2.1 BError BoapServerConnection::process ()

6.31.2.2 BSocket & BoapServerConnection::getSocket ()

6.31.2.3 void BoapServerConnection::setMaxLength (BUInt32 *maxLength*)

6.31.2.4 void ∗ BoapServerConnection::function () `[private, virtual]`

Reimplemented from BThread.

6.31.3 Member Data Documentation

6.31.3.1 BoapServer& BoapServerConnection::oboapServer `[private]`

6.31.3.2 BSocket BoapServerConnection::osocket `[private]`

6.31.3.3 BoapPacket BoapServerConnection::orx `[private]`

6.31.3.4 BoapPacket BoapServerConnection::otx `[private]`

6.31.3.5 BUInt32 BoapServerConnection::omaxLength `[private]`

The documentation for this class was generated from the following files:

- Boap.h
- Boap.cpp

6.32 BoapServiceEntry Class Reference

```
#include <BoapSimple.h>
```

Public Member Functions

- BoapServiceEntry (BoapService service=0, BoapServiceObject ∗object=0)
- BoapServiceEntry (BoapService service=0, BoapServiceObject ∗object=0)

Public Attributes

- BoapService oservice
- BoapServiceObject ∗ oobject

6.32.1 Constructor & Destructor Documentation

6.32.1.1 BoapServiceEntry::BoapServiceEntry (BoapService *service* = 0, BoapServiceObject ∗ *object* = 0) `[inline]`

6.32.1.2 BoapServiceEntry::BoapServiceEntry (BoapService *service* = 0, BoapServiceObject ∗ *object* = 0) `[inline]`

6.32.2 Member Data Documentation

6.32.2.1 BoapService BoapServiceEntry::oservice

6.32.2.2 BoapServiceObject ∗ BoapServiceEntry::oobject

The documentation for this class was generated from the following files:

- Boap.h
- BoapSimple.h

6.33 BoapServiceObject Class Reference

```
#include <BoapSimple.h>
```

Public Member Functions

- BoapServiceObject (BoapServer &server, BString name="")
- virtual ∼BoapServiceObject ()
- BError setName (BString name)
- BError sendEvent (BString signalName, Int32 arg)
- virtual BError processEvent (BString objectName, BString name, Int32 arg)
- BString name ()
- BError doPing (BoapServerConnection ∗conn, BoapPacket &rx, BoapPacket &tx)
- BError doConnectionPriority (BoapServerConnection ∗conn, BoapPacket &rx, BoapPacket &tx)
- BError process (BoapServerConnection ∗conn, BoapPacket &rx, BoapPacket &tx)
- virtual BError processEvent (BoapPacket &rx)
- BoapServiceObject (BoapServer &server, BString name)
- virtual ∼BoapServiceObject ()
- BError sendEvent (BString signalName, Int32 arg)
- virtual BError processEvent (BString objectName, BString name, Int32 arg)
- BString name ()
- BError process (BoapPacket &rx, BoapPacket &tx)
- virtual BError processEvent (BoapPacket &rx)

Protected Member Functions

- BError sendEvent (BoapPacket &tx)
- BError sendEvent (BoapPacket &tx)

Protected Attributes

- BoapServer & oserver
- BString oname
- BUInt32 oapiVersion
- BList< BoapFuncEntry > ofuncList

6.33.1 Constructor & Destructor Documentation

6.33.1.1 BoapServiceObject::BoapServiceObject (BoapServer & *server*, BString *name* = "")

6.33.1.2 BoapServiceObject::∼BoapServiceObject () `[virtual]`

6.33.1.3 BoapServiceObject::BoapServiceObject (BoapServer & *server*, BString *name*)

6.33.1.4 virtual BoapServiceObject::∼BoapServiceObject () `[virtual]`

6.33.2 Member Function Documentation

6.33.2.1 BError BoapServiceObject::setName (BString *name*)

6.33.2.2 BError BoapServiceObject::sendEvent (BString *signalName*, Int32 *arg*)

6.33.2.3 BError BoapServiceObject::processEvent (BString *objectName*, BString *name*, Int32 *arg*) `[virtual]`

6.33.2.4 BString BoapServiceObject::name ()

6.33.2.5 BError BoapServiceObject::doPing (BoapServerConnection ∗ *conn*, BoapPacket & *rx*, BoapPacket & *tx*)

6.33.2.6 BError BoapServiceObject::doConnectionPriority (BoapServerConnection ∗ *conn*, BoapPacket & *rx*, BoapPacket & *tx*)

6.33.2.7 BError BoapServiceObject::process (BoapServerConnection ∗ *conn*, BoapPacket & *rx*, BoapPacket & *tx*)

6.33.2.8 BError BoapServiceObject::processEvent (BoapPacket & *rx*) `[virtual]`

6.33.2.9 BError BoapServiceObject::sendEvent (BoapPacket & *tx*) `[protected]`

6.33.2.10 BError BoapServiceObject::sendEvent (BString *signalName*, Int32 *arg*)

6.33.2.11 virtual BError BoapServiceObject::processEvent (BString *objectName*, BString *name*, Int32 *arg*) `[virtual]`

6.33.2.12 BString BoapServiceObject::name ()

6.33.2.13 BError BoapServiceObject::process (BoapPacket & *rx*, BoapPacket & *tx*)

6.33.2.14 virtual BError BoapServiceObject::processEvent (BoapPacket & *rx*) `[virtual]`

6.33.2.15 BError BoapServiceObject::sendEvent (BoapPacket & *tx*) `[protected]`

6.33.3 Member Data Documentation

6.33.3.1 BoapServer & BoapServiceObject::oserver `[protected]`

6.33.3.2 BString BoapServiceObject::oname `[protected]`

6.33.3.3 BUInt32 BoapServiceObject::oapiVersion `[protected]`

6.33.3.4 BList< BoapFuncEntry > BoapServiceObject::ofuncList `[protected]`

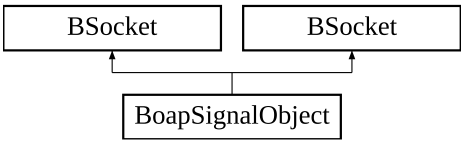
The documentation for this class was generated from the following files:

- Boap.h
- BoapSimple.h
- Boap.cpp
- BoapSimple.cc

6.34 BoapSignalObject Class Reference

```
#include <BoapSimple.h>
```

Inheritance diagram for BoapSignalObject::



Public Member Functions

- BoapSignalObject ()
- BoapSignalObject ()

Protected Member Functions

- BError performSend (BoapPacket &tx)
- BError performSend (BoapPacket &tx)

Protected Attributes

- BoapPacket otx
- BoapPacket orx

6.34.1 Constructor & Destructor Documentation

6.34.1.1 BoapSignalObject::BoapSignalObject ()

6.34.1.2 BoapSignalObject::BoapSignalObject ()

6.34.2 Member Function Documentation

6.34.2.1 BError BoapSignalObject::performSend (BoapPacket & *tx*) `[protected]`

6.34.2.2 BError BoapSignalObject::performSend (BoapPacket & *tx*) `[protected]`

6.34.3 Member Data Documentation

6.34.3.1 BoapPacket BoapSignalObject::otx `[protected]`

6.34.3.2 BoapPacket BoapSignalObject::orx `[protected]`

The documentation for this class was generated from the following files:

- Boap.h
- BoapSimple.h

- Boap.cpp
- BoapSimple.cc

6.35 BObject Class Reference

`#include <BObject.h>`

Public Member Functions

- BObject ()
- virtual ∼BObject ()
- virtual BError getBinary (BDataBuf &buf)
- virtual BError setBinary (BDataBuf &buf)
- virtual BString getString ()
- virtual BError setString (BString str)
- virtual BMemberList getMemberList ()
- virtual BError addMember (BString name, BObject ∗object)
- virtual void printIt ()
- virtual BType & getType ()

Static Public Member Functions

- static BObject ∗ createObj ()

Static Public Attributes

- static BType otype = btypesList.appendType(BType("BObject", BTypeDomainBase, BTypeObject, createObj))

6.35.1 Constructor & Destructor Documentation

6.35.1.1 BObject::BObject ()

6.35.1.2 BObject::~BObject () `[virtual]`

6.35.2 Member Function Documentation

6.35.2.1 BError BObject::getBinary (BDataBuf & *buf*) `[virtual]`

6.35.2.2 BError BObject::setBinary (BDataBuf & *buf*) `[virtual]`

6.35.2.3 BString BObject::getString () `[virtual]`

6.35.2.4 BError BObject::setString (BString *str*) `[virtual]`

6.35.2.5 BMemberList BObject::getMemberList () `[virtual]`

6.35.2.6 BError BObject::addMember (BString *name*, BObject ∗ *object*) `[virtual]`

6.35.2.7 void BObject::printIt () `[virtual]`

6.35.2.8 BType & BObject::getType () `[virtual]`

6.35.2.9 BObject ∗ BObject::createObj () `[static]`

6.35.3 Member Data Documentation

6.35.3.1 BType BObject::otype = btypesList.appendType(BType("BObject", BTypeDomainBase, BTypeObject, createObj)) `[static]`

The documentation for this class was generated from the following files:

- BObject.h
- BObject.cc

6.36 BPoll Class Reference

This class provides an interface for polling a number of file descriptors. It uses round robin polling.

```
#include <BPoll.h>
```

Public Types

- typedef struct pollfd PollFd

Public Member Functions

- BPoll ()
- ~BPoll ()
- void append (int fd, int events=POLLIN|POLLERR|POLLHUP|POLLNVAL)

  Append a file descriptor to polling list.

- void delFd (int fd)

  Remove a file descriptor from polling list.

- BError doPoll (int &fd, int timeoutUs=-1)

  Perform polling operation.

- int getPollFdsNum ()
- PollFd ∗ getPollFds ()
- void clear ()

Private Member Functions

- int nextFd (int i)

Private Attributes

- int ofdsNum

  The number of FD's in list.

- PollFd ∗ ofds

  The list of poll fd's.

- int ofdsNext

  The next list entry for round robin polling.

6.36.1 Detailed Description

This class provides an interface for polling a number of file descriptors. It uses round robin polling.

6.36.2 Member Typedef Documentation

6.36.2.1 typedef struct pollfd BPoll::PollFd [read]

6.36.3 Constructor & Destructor Documentation

6.36.3.1 BPoll::BPoll ()

6.36.3.2 BPoll::∼BPoll ()

6.36.4 Member Function Documentation

6.36.4.1 void BPoll::append (int *fd*, int *events* = POLLIN|POLLERR|POLLHUP|POLLNVAL)

Append a file descriptor to polling list.

6.36.4.2 void BPoll::delFd (int *fd*)

Remove a file descriptor from polling list.

6.36.4.3 BError BPoll::doPoll (int & *fd*, int *timeoutUs* = -1)

Perform polling operation.

6.36.4.4 int BPoll::getPollFdsNum ()

6.36.4.5 BPoll::PollFd ∗ BPoll::getPollFds ()

6.36.4.6 void BPoll::clear ()

6.36.4.7 int BPoll::nextFd (int *i*) [private]

6.36.5 Member Data Documentation

6.36.5.1 int BPoll::ofdsNum [private]

The number of FD's in list.

6.36.5.2 PollFd∗ BPoll::ofds [private]

The list of poll fd's.

6.36.5.3 int BPoll::ofdsNext [private]

The next list entry for round robin polling.

The documentation for this class was generated from the following files:

- BPoll.h

- BPoll-1.cpp
- BPoll.cpp

6.37 BRefData Class Reference

Referenced data storage.

```
#include <BRefData.h>
```

Public Member Functions

- BRefData ()
- BRefData (int len)
- BRefData (const BRefData &refData)
- ∼BRefData ()
- BRefData ∗ copy ()
- BRefData ∗ addRef ()
- int deleteRef ()
- int refCount ()
- char ∗ data ()
- int len ()
- void setLen (int len)
- BRefData & operator= (BRefData &refData)

Private Attributes

- void ∗ oData
- int oLen
- int oSize
- int oRefCount

6.37.1 Detailed Description

Referenced data storage.

6.37.2 Constructor & Destructor Documentation

6.37.2.1 BRefData::BRefData ()

6.37.2.2 BRefData::BRefData (int *len*)

6.37.2.3 BRefData::BRefData (const BRefData & *refData*)

6.37.2.4 BRefData::~BRefData ()

6.37.3 Member Function Documentation

6.37.3.1 BRefData ∗ BRefData::copy ()

6.37.3.2 BRefData ∗ BRefData::addRef ()

6.37.3.3 int BRefData::deleteRef ()

6.37.3.4 int BRefData::refCount () `[inline]`

6.37.3.5 char∗ BRefData::data () `[inline]`

6.37.3.6 int BRefData::len () `[inline]`

6.37.3.7 void BRefData::setLen (int *len*)

6.37.3.8 BRefData & BRefData::operator= (BRefData & *refData*)

6.37.4 Member Data Documentation

6.37.4.1 void∗ BRefData::oData `[private]`

6.37.4.2 int BRefData::oLen `[private]`

6.37.4.3 int BRefData::oSize `[private]`

6.37.4.4 int BRefData::oRefCount `[private]`

The documentation for this class was generated from the following files:

- BRefData.h
- BRefData.cpp

6.38 BRtc Class Reference

Realtime clock.

`#include <BRtc.h>`

Public Member Functions

- BRtc ()
- ∼BRtc ()
- BError init (int rate)

  Setup interrupt rate.
- void wait (int delayUs)

  Wait specified uS.

Private Attributes

- int ofd
- int orate

6.38.1 Detailed Description

Realtime clock.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 BRtc::BRtc ()

6.38.2.2 BRtc::∼BRtc ()

6.38.3 Member Function Documentation

6.38.3.1 BError BRtc::init (int *rate*)

Setup interrupt rate.

6.38.3.2 void BRtc::wait (int *delayUs*)

Wait specified uS.

6.38.4 Member Data Documentation

6.38.4.1 int BRtc::ofd `[private]`

6.38.4.2 int BRtc::orate `[private]`

The documentation for this class was generated from the following files:

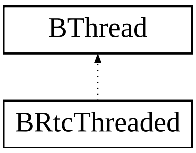
- BRtc.h
- BRtc.cpp

6.39 BRtcThreaded Class Reference

Threaded real time clock.

`#include <BRtc.h>`

Inheritance diagram for BRtcThreaded::



Public Member Functions

- BRtcThreaded ()
- ∼BRtcThreaded ()
- BError init (int rate)

  Setup interrupt rate.

- void wait (int delayUs)

  Wait specified uS.

Private Member Functions

- void ∗ function ()

Private Attributes

- BRtc ortc
- int orate
- BCond ocond

6.39.1 Detailed Description

Threaded real time clock.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 BRtcThreaded::BRtcThreaded ()

6.39.2.2 BRtcThreaded::∼BRtcThreaded ()

6.39.3 Member Function Documentation

6.39.3.1 BError BRtcThreaded::init (int *rate*)

Setup interrupt rate.

6.39.3.2 void BRtcThreaded::wait (int *delayUs*)

Wait specified uS.

6.39.3.3 void * BRtcThreaded::function () `[private, virtual]`

Reimplemented from BThread.

6.39.4 Member Data Documentation

6.39.4.1 BRtc BRtcThreaded::ortc `[private]`

6.39.4.2 int BRtcThreaded::orate `[private]`

6.39.4.3 BCond BRtcThreaded::ocond `[private]`

The documentation for this class was generated from the following files:

- BRtc.h
- BRtc.cpp

6.40 BRWLock Class Reference

thread read-write locks

```
#include <BRWLock.h>
```

Public Member Functions

- BRWLock ()
- BRWLock (const BRWLock &rwlock)
- ~BRWLock ()
- int rdLock ()

  Set lock, wait if necessary.
- int tryRdLock ()

  Test the lock.
- int wrLock ()

  Set lock, wait if necessary.
- int tryWrLock ()

  Test the lock.
- int unlock ()

  Unlock the lock.
- BRWLock & operator= (const BRWLock &rwlock)

Private Attributes

- pthread_rwlock_t olock

6.40.1 Detailed Description

thread read-write locks

6.40.2 Constructor & Destructor Documentation

6.40.2.1 BRWLock::BRWLock ()

6.40.2.2 BRWLock::BRWLock (const BRWLock & *rwlock*)

6.40.2.3 BRWLock::~BRWLock ()

6.40.3 Member Function Documentation

6.40.3.1 int BRWLock::rdLock ()

Set lock, wait if necessary.

6.40.3.2 int BRWLock::tryRdLock ()

Test the lock.

6.40.3.3 int BRWLock::wrLock ()

Set lock, wait if necessary.

6.40.3.4 int BRWLock::tryWrLock ()

Test the lock.

6.40.3.5 int BRWLock::unlock ()

Unlock the lock.

6.40.3.6 BRWLock & BRWLock::operator= (const BRWLock & *rwlock*)

6.40.4 Member Data Documentation

6.40.4.1 pthread_rwlock_t BRWLock::olock `[private]`

The documentation for this class was generated from the following files:

- BRWLock.h
- BRWLock.cpp

6.41 BSema Class Reference

Sempahore class.

`#include <BSema.h>`

Public Member Functions

- BSema (int value=0)
- BSema (const BSema &sema)
- ∼BSema ()
- int post ()

  Post condition.

- int wait ()

  Wait for contition.

- int timedWait (int timeUs)

  Wait for condition with timeout.

- int tryWait ()

  Test for the condition.

- int getValue () const
- BSema & operator= (const BSema &sema)

Private Attributes

- sem_t osema

6.41.1 Detailed Description

Sempahore class.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 BSema::BSema (int *value* = `0`)

6.41.2.2 BSema::BSema (const BSema & *sema*)

6.41.2.3 BSema::∼BSema ()

6.41.3 Member Function Documentation

6.41.3.1 int BSema::post ()

Post condition.

6.41.3.2 int BSema::wait ()

Wait for contition.

6.41.3.3 int BSema::timedWait (int *timeUs*)

Wait for condition with timeout.

6.41.3.4 int BSema::tryWait ()

Test for the condition.

6.41.3.5 int BSema::getValue () const

6.41.3.6 BSema & BSema::operator= (const BSema & *sema*)

6.41.4 Member Data Documentation

6.41.4.1 sem_t BSema::osema `[private]`

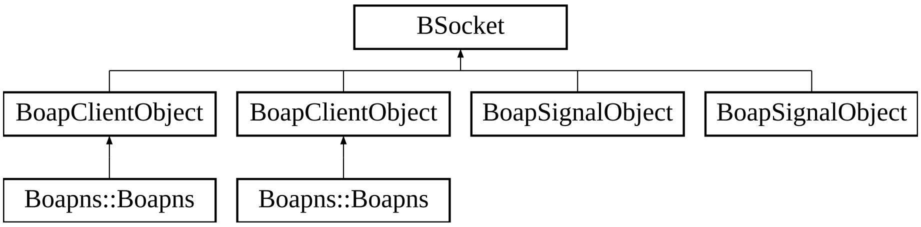
The documentation for this class was generated from the following files:

- BSema.h
- BSema.cpp

6.42 BSocket Class Reference

```
#include <BSocket.h>
```

Inheritance diagram for BSocket::



Public Types

- enum NType { STREAM, DGRAM }
- enum Priority { PriorityLow, PriorityNormal, PriorityHigh }

Public Member Functions

- BSocket ()
- BSocket (int fd)
- BSocket (NType type)
- ∼BSocket ()
- BError init (NType type)
- int getFd ()
- BError bind (const BSocketAddress &add)
- BError connect (const BSocketAddress &add)
- BError shutdown (int how)
- BError close ()
- BError listen (int backlog=5)
- BError accept (int &fd)
- BError accept (int &fd, BSocketAddress &address)
- BError send (const void ∗buf, BSize nbytes, BSize &nbytesSent, int flags=0)
- BError sendTo (const BSocketAddress &address, const void ∗buf, BSize nbytes, BSize &nbytesSent, int flags=0)
- BError recv (void ∗buf, BSize maxbytes, BSize &nbytesRecv, int flags=0)
- BError recvFrom (BSocketAddress &address, void ∗buf, BSize maxbytes, BSize &nbytesRecv, int flags=0)
- BError recvWithTimeout (void ∗buf, BSize maxbytes, BSize &nbytesRecv, int timeout, int flags=0)
- BError recvFromWithTimeout (BSocketAddress &address, void ∗buf, BSize maxbytes, BSize &nbytesRecv, int timeout, int flags=0)
- BError setSockOpt (int level, int optname, void ∗optval, unsigned int optlen)
- BError getSockOpt (int level, int optname, void ∗optval, unsigned int ∗optlen)
- BError setReuseAddress (int on)
- BError setBroadCast (int on)
- BError setPriority (Priority priority)
- BError getMTU (uint32_t &mtu)
- BError getAddress (BSocketAddress &address)

Private Attributes

- int osocket

6.42.1 Member Enumeration Documentation

6.42.1.1 enum BSocket::NType

Enumerator:

STREAM

DGRAM

6.42.1.2 enum BSocket::Priority

Enumerator:

PriorityLow

PriorityNormal

PriorityHigh

6.42.2 Constructor & Destructor Documentation

6.42.2.1 BSocket::BSocket ()

6.42.2.2 BSocket::BSocket (int *fd*)

6.42.2.3 BSocket::BSocket (NType *type*)

6.42.2.4 BSocket::∼BSocket ()

6.42.3 Member Function Documentation

6.42.3.1 BError BSocket::init (NType *type*)

6.42.3.2 int BSocket::getFd ()

6.42.3.3 BError BSocket::bind (const BSocketAddress & *add*)

6.42.3.4 BError BSocket::connect (const BSocketAddress & *add*)

6.42.3.5 BError BSocket::shutdown (int *how*)

6.42.3.6 BError BSocket::close ()

6.42.3.7 BError BSocket::listen (int *backlog* = 5)

6.42.3.8 BError BSocket::accept (int & *fd*)

6.42.3.9 BError BSocket::accept (int & *fd*, BSocketAddress & *address*)

6.42.3.10 BError BSocket::send (const void ∗ *buf*, BSize *nbytes*, BSize & *nbytesSent*, int *flags* = 0)

6.42.3.11 BError BSocket::sendTo (const BSocketAddress & *address*, const void ∗ *buf*, BSize *nbytes*, BSize & *nbytesSent*, int *flags* = 0)

6.42.3.12 BError BSocket::recv (void ∗ *buf*, BSize *maxbytes*, BSize & *nbytesRecv*, int *flags* = 0)

6.42.3.13 BError BSocket::recvFrom (BSocketAddress & *address*, void ∗ *buf*, BSize *maxbytes*, BSize & *nbytesRecv*, int *flags* = 0)

6.42.3.14 BError BSocket::recvWithTimeout (void ∗ *buf*, BSize *maxbytes*, BSize & *nbytesRecv*, int *timeout*, int *flags* = 0)

6.42.3.15 BError BSocket::recvFromWithTimeout (BSocketAddress & *address*, void ∗ *buf*, BSize *maxbytes*, BSize & *nbytesRecv*, int *timeout*, int *flags* = 0)

6.42.3.16 BError BSocket::setSockOpt (int *level*, int *optname*, void ∗ *optval*, unsigned int *optlen*)

6.42.3.17 BError BSocket::getSockOpt (int *level*, int *optname*, void ∗ *optval*, unsigned int ∗ *optlen*)

6.42.3.18 BError BSocket::setReuseAddress (int *on*)

6.42.3.19 BError BSocket::setBroadCast (int *on*)

6.42.3.20 BError BSocket::setPriority (Priority *priority*)

6.42.3.21 BError BSocket::getMTU (uint32_t & *mtu*)

6.42.3.22 BError BSocket::getAddress (BSocketAddress & *address*)

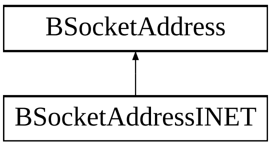
- BSocket.h
- BSocket.cpp

6.43 BSocketAddress Class Reference

Socket Address.

```
#include <BSocket.h>
```

Inheritance diagram for BSocketAddress::



Public Types

- typedef struct sockaddr SockAddr

Public Member Functions

- BSocketAddress ()
- BSocketAddress (const BSocketAddress &add)
- BSocketAddress (SockAddr *address, int len)
- ~BSocketAddress ()
- BError set (SockAddr *address, int len)
- const SockAddr * raw () const
- int len () const
- BSocketAddress & operator= (const BSocketAddress &add)
- operator const SockAddr * () const
- int operator== (const BSocketAddress &add) const
- int operator!= (const BSocketAddress &add) const

Private Attributes

- int olen
- SockAddr * oaddress

6.43.1 Detailed Description

Socket Address.

6.43.2 Member Typedef Documentation

6.43.2.1 typedef struct sockaddr BSocketAddress::SockAddr `[read]`

6.43.3 Constructor & Destructor Documentation

6.43.3.1 BSocketAddress::BSocketAddress ()

6.43.3.2 BSocketAddress::BSocketAddress (const BSocketAddress & *add*)

6.43.3.3 BSocketAddress::BSocketAddress (SockAddr ∗ *address*, int *len*)

6.43.3.4 BSocketAddress::∼BSocketAddress ()

6.43.4 Member Function Documentation

6.43.4.1 BError BSocketAddress::set (SockAddr ∗ *address*, int *len*)

6.43.4.2 const BSocketAddress::SockAddr ∗ BSocketAddress::raw () const

6.43.4.3 int BSocketAddress::len () const

6.43.4.4 BSocketAddress & BSocketAddress::operator= (const BSocketAddress & *add*)

6.43.4.5 BSocketAddress::operator const SockAddr ∗ () const `[inline]`

6.43.4.6 int BSocketAddress::operator== (const BSocketAddress & *add*) const

6.43.4.7 int BSocketAddress::operator!= (const BSocketAddress & *add*) const

6.43.5 Member Data Documentation

6.43.5.1 int BSocketAddress::olen `[private]`

6.43.5.2 SockAddr∗ BSocketAddress::oaddress `[private]`

The documentation for this class was generated from the following files:

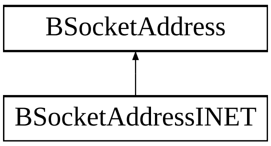
- BSocket.h
- BSocket.cpp

6.44 BSocketAddressINET Class Reference

IP aware socket address.

`#include <BSocket.h>`

Inheritance diagram for BSocketAddressINET::



Public Types

- typedef struct sockaddr_in SockAddrIP

Public Member Functions

- BError set (BString hostName, uint32_t port)
- BError set (uint32_t address, uint32_t port)
- BError set (BString hostName, BString service, BString type)
- void setPort (uint32_t port)
- uint32_t address ()

  Returns socket ip address.

- uint32_t port ()

  Returns socket port.

- BString getString ()

  Return string version of address <ip>:<port>.

Static Public Member Functions

- static BString getHostName ()

  Get this hosts network name.

- static BList< uint32_t > getIpAddresses ()

  Get a list of all the IP addresses of this host.

- static BList< BString > getIpAddressList ()

  Get a list of all the IP addresses of this host under hostname.

- static BList< BString > getIpAddressListAll ()

  Get a list of all the IP addresses of this host looking at physical interfaces.

6.44.1 Detailed Description

IP aware socket address.

6.44.2 Member Typedef Documentation

6.44.2.1 typedef struct sockaddr_in BSocketAddressINET::SockAddrIP [read]

6.44.3 Member Function Documentation

6.44.3.1 BError BSocketAddressINET::set (BString *hostName*, uint32_t *port*)

6.44.3.2 BError BSocketAddressINET::set (uint32_t *address*, uint32_t *port*)

6.44.3.3 BError BSocketAddressINET::set (BString *hostName*, BString *service*, BString *type*)

6.44.3.4 void BSocketAddressINET::setPort (uint32_t *port*)

6.44.3.5 uint32_t BSocketAddressINET::address ()

Returns socket ip address.

6.44.3.6 uint32_t BSocketAddressINET::port ()

Returns socket port.

6.44.3.7 BString BSocketAddressINET::getString ()

Return string version of address <ip>:<port>.

6.44.3.8 BString BSocketAddressINET::getHostName () [static]

Get this hosts network name.

6.44.3.9 BList< uint32_t > BSocketAddressINET::getIpAddresses () [static]

Get a list of all the IP addresses of this host.

6.44.3.10 BList< BString > BSocketAddressINET::getIpAddressList () [static]

Get a list of all the IP addresses of this host under hostname.

6.44.3.11 BList< BString > BSocketAddressINET::getIpAddressListAll () [static]

Get a list of all the IP addresses of this host looking at physical interfaces.

The documentation for this class was generated from the following files:

- BSocket.h

- BSocket.cpp

6.45 BString Class Reference

```
#include <BString.h>
```

Public Member Functions

- BString ()
- BString (const BString &string)
- BString (const char ∗str)
- BString (char ch)
- BString (int v)
- BString (unsigned int v)
- BString (long v)
- BString (unsigned long long)
- BString (double v)
- virtual ∼BString ()
- BString copy ()

  Return an independant copy.

- virtual void strChanged ()
- int len () const

  Length of string.

- const char ∗ retStr () const

  Ptr to char∗ representation.

- char ∗ retStrDup () const

  Ptr to newly malloc'd char∗.

- int retInt () const

  Return string as a int.

- double retDouble () const

  Return string as a double.

- int compare (const BString &string) const

  Compare strings.

- int compareWild (const BString &string) const

  Compare string to string with wildcards.

- int compareWildExpression (const BString &string) const

  Compare string to space deliminated patterns.

- BString add (const BString &str) const

  Add two strings.

- BString & truncate (int len)

  Truncate to length len.

- BString & pad (int len)

  Pad to length len.

- BString & toUpper ()

  Convert to uppercase.

- BString & toLower ()

  Convert to lowercase.

- void removeNL ()

  Remove if present NL from last char.

- BString subString (int start, int len) const

  Returns substring.

- int del (int start, int len)

  Delete substring.

- int insert (int start, BString str)

  Insert substring.

- void printf (const char ∗fmt,...)

  Formated print into the string.

- int find (char ch) const

  Find ch in string searching forwards.

- int findReverse (char ch) const

  Find ch in string searching backwards.

- BList< BString > getTokenList (BString separators)

  Break string into tokens.

- BString removeSeparators (BString separators)

  Remove any char from sepatators from string.

- BString pullToken (BString terminators)

  Pull token from start of string.

- BString pullSeparators (BString separators)

  Pull separators from start of string.

- BString pullWord ()

  Pull a word out of the head of the string.

- BString pullLine ()

  Pull a line out of the head of the string.

- BString field (int field) const

- char ∗∗ fields ()
- BString & operator= (const BString &string)
- char & operator[ ] (int pos)
- int operator== (const BString &s) const
- int operator== (const char ∗s) const
- int operator> (const BString &s) const
- int operator> (const char ∗s) const
- int operator< (const BString &s) const
- int operator< (const char ∗s) const
- int operator>= (const BString &s) const
- int operator<= (const BString &s) const
- int operator!= (const BString &s) const
- int operator!= (const char ∗s) const
- BString operator+ (const BString &s) const
- BString operator+ (const char ∗s) const
- BString operator+= (const BString &s)
- BString operator+= (const char ∗s)
- BString operator+ (char ch) const
- BString operator+ (int i) const
- BString operator+ (unsigned int i) const
- BString operator+ (unsigned long long i) const
- operator const char ∗ () const

Static Public Member Functions

- static BString convert (char ch)

  Converts char to string.

- static BString convert (int value)

  Converts int to string.

- static BString convert (unsigned int value)

  Converts uint to string.

- static BString convert (long value)

  Converts long to string.

- static BString convert (double value)

  Converts double to string.

- static BString convert (unsigned long long value)

  Converts u long long to string.

- static BString convertHex (int value)

  Converts int to string as hex value.

- static BString convertHex (unsigned int value)

  Converts uint to string as hex value.

Protected Attributes

- BRefData ∗ ostr

Private Member Functions

- void Init (const char ∗str)
- int inString (int pos) const
- int isSpace (char ch) const

6.45.1 Constructor & Destructor Documentation

6.45.1.1 BString::BString ()

6.45.1.2 BString::BString (const BString & *string*)

6.45.1.3 BString::BString (const char ∗ *str*)

6.45.1.4 BString::BString (char *ch*)

6.45.1.5 BString::BString (int *v*)

6.45.1.6 BString::BString (unsigned int *v*)

6.45.1.7 BString::BString (long *v*)

6.45.1.8 BString::BString (unsigned long long *value*)

6.45.1.9 BString::BString (double *v*)

6.45.1.10 BString::∼BString () `[virtual]`

6.45.2 Member Function Documentation

6.45.2.1 BString BString::convert (char *ch*) `[static]`

Converts char to string.

6.45.2.2 BString BString::convert (int *value*) `[static]`

Converts int to string.

6.45.2.3 BString BString::convert (unsigned int *value*) `[static]`

Converts uint to string.

6.45.2.4 BString BString::convert (long *value*) `[static]`

Converts long to string.

6.45.2.5 BString BString::convert (double *value*) `[static]`

Converts double to string.

6.45.2.6 BString BString::convert (unsigned long long *value*) `[static]`

Converts u long long to string.

6.45.2.7 BString BString::convertHex (int *value*) `[static]`

Converts int to string as hex value.

6.45.2.8 BString BString::convertHex (unsigned int *value*) `[static]`

Converts uint to string as hex value.

6.45.2.9 BString BString::copy ()

Return an independant copy.

6.45.2.10 void BString::strChanged () `[virtual]`

6.45.2.11 int BString::len () const

Length of string.

6.45.2.12 const char * BString::retStr () const

Ptr to char* representation.

6.45.2.13 char * BString::retStrDup () const

Ptr to newly malloc'd char*.

6.45.2.14 int BString::retInt () const

Return string as a int.

6.45.2.15 double BString::retDouble () const

Return string as a double.

6.45.2.16 int BString::compare (const BString & *string*) const

Compare strings.

6.45.2.17 int BString::compareWild (const BString & *string*) const

Compare string to string with wildcards.

6.45.2.18 int BString::compareWildExpression (const BString & *string*) const

Compare string to space deliminated patterns.

6.45.2.19 BString BString::add (const BString & *str*) const

Add two strings.

6.45.2.20 BString & BString::truncate (int *len*)

Truncate to length len.

6.45.2.21 BString & BString::pad (int *len*)

Pad to length len.

6.45.2.22 BString & BString::toUpper ()

Convert to uppercase.

6.45.2.23 BString & BString::toLower ()

Convert to lowercase.

6.45.2.24 void BString::removeNL ()

Remove if present NL from last char.

6.45.2.25 BString BString::subString (int *start*, int *len*) const

Returns substring.

6.45.2.26 int BString::del (int *start*, int *len*)

Delete substring.

6.45.2.27 int BString::insert (int *start*, BString *str*)

Insert substring.

6.45.2.28 void BString::printf (const char $*$ *fmt*, ...)

Formated print into the string.

6.45.2.29 int BString::find (char *ch*) const

Find ch in string searching forwards.

6.45.2.30 int BString::findReverse (char *ch*) const

Find ch in string searching backwards.

6.45.2.31 BList< BString > BString::getTokenList (BString *separators*)

Break string into tokens.

6.45.2.32 BString BString::removeSeparators (BString *separators*)

Remove any char from sepatators from string.

6.45.2.33 BString BString::pullToken (BString *terminators*)

Pull token from start of string.

6.45.2.34 BString BString::pullSeparators (BString *separators*)

Pull separators from start of string.

6.45.2.35 BString BString::pullWord ()

Pull a word out of the head of the string.

6.45.2.36 BString BString::pullLine ()

Pull a line out of the head of the string.

6.45.2.37 BString BString::field (int *field*) const

6.45.2.38 char $**$ BString::fields ()

6.45.2.39 BString & BString::operator= (const BString & *string*)

6.45.2.40]

char & BString::operator[ ] (int *pos*)

6.45.2.41 int BString::operator== (const BString & *s*) const `[inline]`

6.45.2.42 int BString::operator== (const char ∗ *s*) const `[inline]`

6.45.2.43 int BString::operator> (const BString & *s*) const `[inline]`

6.45.2.44 int BString::operator> (const char ∗ *s*) const `[inline]`

6.45.2.45 int BString::operator< (const BString & *s*) const `[inline]`

6.45.2.46 int BString::operator< (const char ∗ *s*) const `[inline]`

6.45.2.47 int BString::operator>= (const BString & *s*) const `[inline]`

6.45.2.48 int BString::operator<= (const BString & *s*) const `[inline]`

6.45.2.49 int BString::operator!= (const BString & *s*) const `[inline]`

6.45.2.50 int BString::operator!= (const char ∗ *s*) const `[inline]`

6.45.2.51 BString BString::operator+ (const BString & *s*) const `[inline]`

6.45.2.52 BString BString::operator+ (const char ∗ *s*) const `[inline]`

6.45.2.53 BString BString::operator+= (const BString & *s*) `[inline]`

6.45.2.54 BString BString::operator+= (const char ∗ *s*) `[inline]`

6.45.2.55 BString BString::operator+ (char *ch*) const `[inline]`

6.45.2.56 BString BString::operator+ (int *i*) const `[inline]`

6.45.2.57 BString BString::operator+ (unsigned int *i*) const `[inline]`

6.45.2.58 BString BString::operator+ (unsigned long long *i*) const `[inline]`

6.45.2.59 BString::operator const char ∗ () const `[inline]`

6.45.2.60 void BString::Init (const char ∗ *str*) `[private]`

6.45.2.61 int BString::inString (int *pos*) const `[private]`

6.45.2.62 int BString::isSpace (char *ch*) const `[private]`

6.45.3 Member Data Documentation

6.45.3.1 BRefData∗ BString::ostr `[protected]`

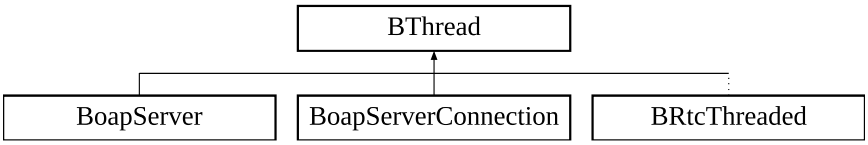
The documentation for this class was generated from the following files:

- BString.h
- BString.cpp

6.46 BThread Class Reference

`#include <BThread.h>`

Inheritance diagram for BThread::



Public Member Functions

- BThread ()
- virtual ∼BThread ()
- int setInitPriority (int policy, int priority)
- int setInitStackSize (size_t stackSize)
- int start ()
- void ∗ result ()
- int running ()
- int setPriority (int policy, int priority)
- int cancel ()
- void ∗ waitForCompletion ()
- pthread_t getThread ()
- virtual void ∗ function ()

Static Private Member Functions

- static void ∗ startFunc (void ∗)

Private Attributes

- pthread_t othread
- size_t ostackSize
- int opolicy
- int opriority
- int orunning
- void ∗ oresult

6.46.1 Constructor & Destructor Documentation

6.46.1.1 BThread::BThread ()

6.46.1.2 BThread::∼BThread () [virtual]

6.46.2 Member Function Documentation

6.46.2.1 int BThread::setInitPriority (int *policy*, int *priority*)

6.46.2.2 int BThread::setInitStackSize (size_t *stackSize*)

6.46.2.3 int BThread::start ()

6.46.2.4 void ∗ BThread::result ()

6.46.2.5 int BThread::running ()

6.46.2.6 int BThread::setPriority (int *policy*, int *priority*)

6.46.2.7 int BThread::cancel ()

6.46.2.8 void ∗ BThread::waitForCompletion ()

6.46.2.9 pthread_t BThread::getThread ()

6.46.2.10 void ∗ BThread::function () [virtual]

Reimplemented in BoapServerConnection, BoapServer, and BRtcThreaded.

6.46.2.11 void ∗ BThread::startFunc (void ∗ *arg*) [static, private]

6.46.3 Member Data Documentation

6.46.3.1 pthread_t BThread::othread [private]

6.46.3.2 size_t BThread::ostackSize [private]

6.46.3.3 int BThread::opolicy [private]

6.46.3.4 int BThread::opriority [private]

6.46.3.5 int BThread::orunning [private]

6.46.3.6 void∗ BThread::oresult [private]

The documentation for this class was generated from the following files:

- BThread.h
- BThread.cpp

6.47 BTimer Class Reference

Stopwatch style timer.

```
#include <BTimer.h>
```

Public Member Functions

- BTimer ()
- ~BTimer ()
- void start ()

  Start timer.
- void stop ()

  Stop timer.
- void clear ()

  Clear timer.
- double getElapsedTime ()

  Returns the elapsed time from the last start.
- void add (BTimer &timer)

  Add two timers.
- double average ()

  Average time is duration between start() and stop() / number of stops.
- double peak ()

  Peak time.

Static Private Member Functions

- static double getTime ()

Private Attributes

- BMutex olock
- unsigned int onum
- double ostartTime
- double oendTime
- double oaverage
- double opeak

6.47.1 Detailed Description

Stopwatch style timer.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 BTimer::BTimer ()

6.47.2.2 BTimer::∼BTimer ()

6.47.3 Member Function Documentation

6.47.3.1 void BTimer::start ()

Start timer.

6.47.3.2 void BTimer::stop ()

Stop timer.

6.47.3.3 void BTimer::clear ()

Clear timer.

6.47.3.4 double BTimer::getElapsedTime ()

Returns the elapsed time from the last start.

6.47.3.5 void BTimer::add (BTimer & *timer*)

Add two timers.

6.47.3.6 double BTimer::average ()

Average time is duration between start() and stop() / number of stops.

6.47.3.7 double BTimer::peak ()

Peak time.

6.47.3.8 double BTimer::getTime () `[static, private]`

6.47.4 Member Data Documentation

6.47.4.1 BMutex BTimer::olock `[private]`

6.47.4.2 unsigned int BTimer::onum `[private]`

6.47.4.3 double BTimer::ostartTime `[private]`

6.47.4.4 double BTimer::oendTime `[private]`

6.47.4.5 double BTimer::oaverage `[private]`

6.47.4.6 double BTimer::opeak `[private]`

The documentation for this class was generated from the following files:

- BTimer.h
- BTimer.cpp

6.48 BUrl Class Reference

Basic access to a Url.

`#include <BUrl.h>`

Public Member Functions

- BUrl ()
- ∼BUrl ()
- BError readString (BString url, BString &str)

  Reads URL.

Static Private Member Functions

- static size_t writeData (void ∗data, size_t size, size_t elSize, void ∗stream)

Private Attributes

- BString ores

Static Private Attributes

- static int oinit

6.48.1 Detailed Description

Basic access to a Url.

6.48.2 Constructor & Destructor Documentation

6.48.2.1 BUrl::BUrl ()

6.48.2.2 BUrl::∼BUrl ()

6.48.3 Member Function Documentation

6.48.3.1 BError BUrl::readString (BString *url*, BString & *str*)

Reads URL.

6.48.3.2 size_t BUrl::writeData (void ∗ *data*, size_t *size*, size_t *elSize*, void ∗ *stream*) `[static, private]`

6.48.4 Member Data Documentation

6.48.4.1 int BUrl::oinit `[static, private]`

6.48.4.2 BString BUrl::ores `[private]`

The documentation for this class was generated from the following files:

- BUrl.h
- BUrl.cpp

Chapter 7

File Documentation

7.1 BArray.h File Reference

```
#include <BTypes.h>
#include <vector>
```

Classes

- class BArray< T >

Defines

- #define BArray_H 1

7.1.1 Define Documentation

7.1.1.1 #define BArray_H 1

7.2 BBuffer.cpp File Reference

```
#include <stdlib.h>
#include <memory.h>
#include <BBuffer.h>
```

Defines

- #define SIZE 1024

7.2.1 Define Documentation

7.2.1.1 #define SIZE 1024

7.3 BBuffer.h File Reference

```
#include <stdint.h>
```

Classes

- class BBuffer

Defines

- #define BBUFFER_H 1

7.3.1 Define Documentation

7.3.1.1 #define BBUFFER_H 1

7.4 BCond.cpp File Reference

```
#include <BCond.h>
#include <sys/time.h>
#include <stdio.h>
```

7.5 BCond.h File Reference

```
#include <pthread.h>
```

Classes

- class BCond

Defines

- #define BCOND_H 1

7.5.1 Define Documentation

7.5.1.1 #define BCOND_H 1

7.6 BCondInt.cpp File Reference

```
#include <BCondInt.h>
#include <sys/time.h>
#include <stdio.h>
#include <errno.h>
```

7.7 BCondInt.h File Reference

```
#include <BTypes.h>
#include <pthread.h>
```

Classes

- class BCondValue

  Thread conditional value.

- class BCondInt

  Thread conditional integer.

- class BCondBool

  Thread conditional boolean.

- class BCondWrap

Defines

- #define BCONDINT_H 1

7.7.1 Define Documentation

7.7.1.1 #define BCONDINT_H 1

7.8 BDir.cpp File Reference

```
#include <BDir.h>
#include <dirent.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
```

Functions

- static int wild (const dirent *e)

Variables

- static BString wildString

7.8.1 Function Documentation

7.8.1.1 static int wild (const dirent * *e*) `[static]`

7.8.2 Variable Documentation

7.8.2.1 BString wildString `[static]`

7.9 BDir.h File Reference

```
#include <BList.h>
#include <BString.h>
#include <BError.h>
#include <sys/stat.h>
```

Classes

- class BDir

  File system directory class.

Defines

- #define BDIR_H 1

7.9.1 Define Documentation

7.9.1.1 #define BDIR_H 1

7.10 BEntry.cpp File Reference

```
#include <ctype.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#include <BEntry.h>
```

7.11 BEntry.h File Reference

```
#include <BList.h>
#include <BString.h>
```

Classes

- class BEntry

  Manipulate a name value pair.

- class BEntryList

  List of Entries. Where an entry is a name value pair.

- class BEntryFile

  File of Entries.

7.12 BError.cpp File Reference

```
#include <BError.h>
```

7.13 BError.h File Reference

```
#include <BString.h>
```

Classes

- class BError

  Error return class.

Defines

- #define BERROR_H 1

7.13.1 Define Documentation

7.13.1.1 #define BERROR_H 1

7.14 BEvent.cpp File Reference

```
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <BEvent.h>
#include <BPoll.h>
```

7.15 BEvent.h File Reference

```
#include <stdint.h>
#include <BError.h>
```

Classes

- class BEvent

  This class provides a base class for all event objects that can be sent over the events interface.

- class BEventError
- class BEventPipe

  This class provides a base interface for sending events via a pipe. This allows threads to send events that can be picked up by the poll system call.

- class BEventInt

  This class provides an interface for sending simple integer events via a file descriptor. This allows threads to send events that can be picked up by the poll system call.

Defines

- #define BEvent_H 1

Enumerations

- enum BEventType { BEventTypeNone, BEventTypeInt, BEventTypeError }

7.15.1 Define Documentation

7.15.1.1 #define BEvent_H 1

7.15.2 Enumeration Type Documentation

7.15.2.1 enum BEventType

Enumerator:

BEventTypeNone

BEventTypeInt

BEventTypeError

7.16 BFile.cpp File Reference

```
#include <BFile.h>
#include <sys/stat.h>
#include <string.h>
#include <stdarg.h>
#include <errno.h>
```

Defines

- #define STRBUF 10240

7.16.1 Define Documentation

7.16.1.1 #define STRBUF 10240

7.17 BFile.h File Reference

```
#include <stdio.h>
#include <BString.h>
#include <BError.h>
```

Classes

- class BFile

  File operations class.

Defines

- #define BFILE_H 1

7.17.1 Define Documentation

7.17.1.1 #define BFILE_H 1

7.18 BList.h File Reference

```
#include <BList_func.h>
```

Classes

- class BIter

  Iterator for BList.

- class BList< T >

  Template based list class.

- class BList< T >::Node

Defines

- #define BLIST_H 1

7.18.1 Define Documentation

7.18.1.1 #define BLIST_H 1

7.19 BList_func.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <memory.h>
```

7.20 BMutex.cpp File Reference

```
#include <BMutex.h>
```

Defines

- #define MDEBUG 0

7.20.1 Define Documentation

7.20.1.1 #define MDEBUG 0

7.21 BMutex.h File Reference

```
#include <pthread.h>
```

Classes

- class BMutex

  Mutex class.

Defines

- #define BMUTEX_H 1

7.21.1 Define Documentation

7.21.1.1 #define BMUTEX_H 1

7.22 BNameValue.h File Reference

```
#include <BList.h>
#include <BString.h>
```

Classes

- class BNameValue< T >
- class BNameValueList< T >

Defines

- #define BNAMEVALUE_H 1
- #define TEMPLATE_NEW 1

7.22.1 Define Documentation

7.22.1.1 #define BNAMEVALUE_H 1

7.22.1.2 #define TEMPLATE_NEW 1

7.23 Boap.cpp File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <unistd.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/tcp.h>
#include <Boap.h>
#include <byteswap.h>
#include <BoapnsD.h>
#include <BoapnsC.h>
```

Defines

- #define DEBUG 0
- #define APIVERSION_TEST 1
- #define dprintf(fmt, a...)
- #define IS_BIG_ENDIAN 1

Functions

- static void swap8 (char ∗d, char ∗s)
- static void swap16 (char ∗d, char ∗s)
- static void swap32 (char ∗d, char ∗s)
- static void swap64 (char ∗d, char ∗s)

Variables

- const int boapPort = 12000

  The default BOAP connection port.

- const int roundSize = 256

7.23.1 Define Documentation

7.23.1.1 #define APIVERSION_TEST 1

7.23.1.2 #define DEBUG 0

7.23.1.3 #define dprintf(fmt, a...)

7.23.1.4 #define IS_BIG_ENDIAN 1

7.23.2 Function Documentation

7.23.2.1 static void swap16 (char ∗ *d*, char ∗ *s*) `[inline, static]`

7.23.2.2 static void swap32 (char ∗ *d*, char ∗ *s*) `[inline, static]`

7.23.2.3 static void swap64 (char ∗ *d*, char ∗ *s*) `[inline, static]`

7.23.2.4 static void swap8 (char ∗ *d*, char ∗ *s*) `[inline, static]`

7.23.3 Variable Documentation

7.23.3.1 const int boapPort = 12000

The default BOAP connection port.

7.23.3.2 const int roundSize = 256

7.24 Boap.h File Reference

```
#include <stdint.h>
#include <BPoll.h>
#include <BSocket.h>
#include <BThread.h>
#include <BError.h>
#include <BEvent.h>
#include <BMutex.h>
#include <BTypes.h>
```

Namespaces

- namespace Boapns

Classes

- struct BoapPacketHead
- class BoapPacket
- class BoapClientObject
- class BoapSignalObject
- class BoapServiceEntry
- class BoapServerConnection
- class BoapServer
- class BoapFuncEntry
- class BoapServiceObject

Typedefs

- typedef UInt32 BoapService
- typedef BError(BoapServiceObject::∗ BoapFunc)(BoapServerConnection ∗conn, BoapPacket &rx, BoapPacket &tx)

Enumerations

- enum BoapType {
  BoapTypeRpc, BoapTypeRpcReply, BoapTypeSignal, BoapTypeRpc,
  BoapTypeSignal }
- enum BoapPriority { BoapPriorityLow, BoapPriorityNormal, BoapPriorityHigh }

Variables

- const UInt32 BoapMagic = 0x424F4100

7.24.1 Typedef Documentation

7.24.1.1 typedef BError(BoapServiceObject::∗ BoapFunc)(BoapServerConnection ∗conn, BoapPacket &rx, BoapPacket &tx)

7.24.1.2 typedef UInt32 BoapService

7.24.2 Enumeration Type Documentation

7.24.2.1 enum BoapPriority

Enumerator:

BoapPriorityLow

BoapPriorityNormal

BoapPriorityHigh

7.24.2.2 enum BoapType

Enumerator:

BoapTypeRpc

BoapTypeRpcReply

BoapTypeSignal

BoapTypeRpc

BoapTypeSignal

7.24.3 Variable Documentation

7.24.3.1 const UInt32 BoapMagic = 0x424F4100

7.25 BoapnsC.cc File Reference

```
#include <BoapnsC.h>
```

Namespaces

- namespace Boapns

Functions

- Boapns::Boapns (BString name)
- BError Boapns::getVersion (BString &version)
- BError Boapns::getEntryList (BList< BoapEntry > &entryList)
- BError Boapns::getEntry (BString name, BoapEntry &entry)
- BError Boapns::addEntry (BoapEntry entry)
- BError Boapns::delEntry (BString name)
- BError Boapns::getNewName (BString &name)

7.26 BoapnsC.h File Reference

```
#include <stdlib.h>
#include <stdint.h>
#include <Boap.h>
#include <BString.h>
#include <BList.h>
#include <BArray.h>
#include <BoapnsD.h>
```

Namespaces

- namespace Boapns

Classes

- class Boapns::Boapns

Defines

- #define BOAPNSC_H 1

Variables

- const BUInt32 Boapns::apiVersion = 0

7.26.1 Define Documentation

7.26.1.1 #define BOAPNSC_H 1

7.27 BoapnsD.cc File Reference

```
#include <BoapnsD.h>
```

Namespaces

- namespace Boapns

7.28 BoapnsD.h File Reference

```
#include <Boap.h>
#include <BList.h>
#include <BArray.h>
```

Namespaces

- namespace Boapns

Classes

- class Boapns::BoapEntry

Defines

- #define BOAPNSD_H 1

7.28.1 Define Documentation

7.28.1.1 #define BOAPNSD_H 1

7.29 BoapSimple.cc File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <Boap.h>
#include <BoapnsD.h>
#include <BoapnsC.h>
```

Defines

- #define DEBUG 0
- #define dprintf(fmt, a...)

Variables

- const int roundSize = 256

7.29.1 Define Documentation

7.29.1.1 #define DEBUG 0

7.29.1.2 #define dprintf(fmt, a...)

7.29.2 Variable Documentation

7.29.2.1 const int roundSize = 256

7.30 BoapSimple.h File Reference

```
#include <stdint.h>
#include <BPoll.h>
#include <BSocket.h>
#include <BError.h>
```

Classes

- struct BoapPacketHead
- class BoapPacket
- class BoapClientObject
- class BoapSignalObject
- class BoapServiceEntry
- class BoapServer
- class BoapFuncEntry
- class BoapServiceObject

Typedefs

- typedef int8_t Int8
- typedef uint8_t UInt8
- typedef int16_t Int16
- typedef uint16_t UInt16
- typedef int32_t Int32
- typedef uint32_t UInt32
- typedef double Double
- typedef uint32_t BoapService
- typedef BError(BoapServiceObject::∗ BoapFunc)(BoapPacket &rx, BoapPacket &tx)

Enumerations

- enum BoapType {

  BoapTypeRpc, BoapTypeRpcReply, BoapTypeSignal, BoapTypeRpc,

  BoapTypeSignal }

7.30.1 Typedef Documentation

7.30.1.1 typedef BError(BoapServiceObject::∗ BoapFunc)(BoapPacket &rx, BoapPacket &tx)

7.30.1.2 typedef uint32_t BoapService

7.30.1.3 typedef double Double

7.30.1.4 typedef int16_t Int16

7.30.1.5 typedef int32_t Int32

7.30.1.6 typedef int8_t Int8

7.30.1.7 typedef uint16_t UInt16

7.30.1.8 typedef uint32_t UInt32

7.30.1.9 typedef uint8_t UInt8

7.30.2 Enumeration Type Documentation

7.30.2.1 enum BoapType

Enumerator:

BoapTypeRpc

BoapTypeRpcReply

BoapTypeSignal

BoapTypeRpc

BoapTypeSignal

7.31 BObject.cc File Reference

```
#include <stdio.h>
#include <ctype.h>
#include <memory.h>
#include <string.h>
#include <BObject.h>
#include <iostream>
```

Defines

- #define DEBUG 0

7.31.1 Define Documentation

7.31.1.1 #define DEBUG 0

7.32 BObject.h File Reference

```
#include <BType.h>
#include <BDataBuf.h>
#include <BString.h>
#include <BNameValue.h>
#include <BList.h>
#include <BError.h>
```

Classes

- class BObject

Defines

- #define BOBJECT_H 1

Typedefs

- typedef BNameValue< BObject * > BMember
- typedef BNameValueList< BObject * > BMemberList

7.32.1 Define Documentation

7.32.1.1 #define BOBJECT_H 1

7.32.2 Typedef Documentation

7.32.2.1 typedef BNameValue<BObject*> BMember

7.32.2.2 typedef BNameValueList<BObject*> BMemberList

7.33 BPoll-1.cpp File Reference

```
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <BPoll.h>
```

7.34 BPoll.cpp File Reference

```
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <BPoll.h>
```

7.35 BPoll.h File Reference

```
#include <BList.h>
#include <BError.h>
#include <sys/poll.h>
```

Classes

- class BPoll

  This class provides an interface for polling a number of file descriptors. It uses round robin polling.

Defines

- #define BPOLL_H 1

7.35.1 Define Documentation

7.35.1.1 #define BPOLL_H 1

7.36 BRefData.cpp File Reference

```
#include <stdlib.h>
#include <string.h>
#include <BRefData.h>
```

Defines

- #define DEBUG 0
- #define CHUNK 16

7.36.1 Define Documentation

7.36.1.1 #define CHUNK 16

7.36.1.2 #define DEBUG 0

7.37 BRefData.h File Reference

Classes

- class BRefData

  Referenced data storage.

Defines

- #define BREFDATA_H 1

7.37.1 Define Documentation

7.37.1.1 #define BREFDATA_H 1

7.38 BRtc.cpp File Reference

```
#include <BRtc.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/ioctl.h>
#include <linux/rtc.h>
```

7.39 BRtc.h File Reference

```
#include <BError.h>
#include <BThread.h>
#include <BCond.h>
```

Classes

- class BRtc

  Realtime clock.

- class BRtcThreaded

  Threaded real time clock.

7.40 BRWLock.cpp File Reference

```
#include <BRWLock.h>
```

7.41 BRWLock.h File Reference

```
#include <pthread.h>
```

Classes

- class BRWLock

  thread read-write locks

Defines

- #define BRWLOCK_H 1

7.41.1 Define Documentation

7.41.1.1 #define BRWLOCK_H 1

7.42 BSema.cpp File Reference

```
#include <BSema.h>
#include <errno.h>
#include <sys/time.h>
```

7.43 BSema.h File Reference

```
#include <semaphore.h>
```

Classes

- class BSema

  Sempahore class.

Defines

- #define BSEMA_H 1

7.43.1 Define Documentation

7.43.1.1 #define BSEMA_H 1

7.44 BSocket.cpp File Reference

```
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <net/if.h>
#include "BSocket.h"
```

Defines

- #define IP_MTU 14

7.44.1 Define Documentation

7.44.1.1 #define IP_MTU 14

7.45 BSocket.h File Reference

```
#include <BString.h>
#include <BError.h>
#include <BTypes.h>
#include <stdint.h>
#include <sys/types.h>
#include <sys/prctl.h>
```

Classes

- class BSocketAddress

  Socket Address.

- class BSocketAddressINET

  IP aware socket address.

- class BSocket

Defines

- #define BSOCKET_H 1

7.45.1 Define Documentation

7.45.1.1 #define BSOCKET_H 1

7.46 BString.cpp File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdarg.h>
#include <ctype.h>
#include "BString.h"
```

Defines

- #define DEBUG 0
- #define STRIP 0x7f
- #define MINUS '-'

Functions

- static int gmatch (const char ∗s, const char ∗p)
- std::ostream & operator<< (std::ostream &o, BString &s)
- std::istream & operator>> (std::istream &i, BString &s)

7.46.1 Define Documentation

7.46.1.1 #define DEBUG 0

7.46.1.2 #define MINUS '-'

7.46.1.3 #define STRIP 0x7f

7.46.2 Function Documentation

7.46.2.1 static int gmatch (const char ∗ *s*, const char ∗ *p*) `[static]`

7.46.2.2 std::ostream& operator<< (std::ostream & *o*, BString & *s*)

7.46.2.3 std::istream& operator>> (std::istream & *i*, BString & *s*)

7.47 BString.h File Reference

```
#include <BRefData.h>
#include <BList.h>
#include <iostream>
```

Classes

- class BString

Defines

- #define BSTRING_H 1

Functions

- std::ostream & operator<< (std::ostream &o, BString &s)
- std::istream & operator>> (std::istream &i, BString &s)

7.47.1 Define Documentation

7.47.1.1 #define BSTRING_H 1

7.47.2 Function Documentation

7.47.2.1 std::ostream& operator<< (std::ostream & *o*, BString & *s*)

7.47.2.2 std::istream& operator>> (std::istream & *i*, BString & *s*)

7.48 BThread.cpp File Reference

```
#include <BThread.h>
#include <unistd.h>
#include <errno.h>
#include <sys/types.h>
```

7.49 BThread.h File Reference

```
#include <pthread.h>
```

Classes

- class BThread

Defines

- #define BTHREAD_H 1

7.49.1 Define Documentation

7.49.1.1 #define BTHREAD_H 1

7.50 BTimer.cpp File Reference

```
#include <BTimer.h>
#include <sys/time.h>
```

7.51 BTimer.h File Reference

```
#include <BMutex.h>
```

Classes

- class BTimer

  Stopwatch style timer.

7.52 BTypes.h File Reference

```
#include <stdint.h>
#include <sys/types.h>
#include <vector>
```

Defines

- #define BTYPES_H 1

Typedefs

- typedef int8_t BInt8
- typedef uint8_t BUInt8
- typedef int16_t BInt16
- typedef uint16_t BUInt16
- typedef int32_t BInt32
- typedef uint32_t BUInt32
- typedef int64_t BInt64
- typedef uint64_t BUInt64
- typedef float BFloat
- typedef double BDouble
- typedef size_t BSize
- typedef uint32_t BUInt
- typedef std::vector< float > BArrayFloat
- typedef std::vector< double > BArrayDouble
- typedef int8_t Int8
- typedef uint8_t UInt8
- typedef int16_t Int16
- typedef uint16_t UInt16
- typedef int32_t Int32
- typedef uint32_t UInt32
- typedef int64_t Int64
- typedef uint64_t UInt64
- typedef float Float
- typedef double Double

7.52.1 Define Documentation

7.52.1.1 #define BTYPES_H 1

7.52.2 Typedef Documentation

7.52.2.1 typedef std::vector<double> BArrayDouble

7.52.2.2 typedef std::vector<float> BArrayFloat

7.52.2.3 typedef double BDouble

7.52.2.4 typedef float BFloat

7.52.2.5 typedef int16_t BInt16

7.52.2.6 typedef int32_t BInt32

7.52.2.7 typedef int64_t BInt64

7.52.2.8 typedef int8_t BInt8

7.52.2.9 typedef size_t BSize

7.52.2.10 typedef uint32_t BUInt

7.52.2.11 typedef uint16_t BUInt16

7.52.2.12 typedef uint32_t BUInt32

7.52.2.13 typedef uint64_t BUInt64

7.52.2.14 typedef uint8_t BUInt8

7.52.2.15 typedef double Double

7.52.2.16 typedef float Float

7.52.2.17 typedef int16_t Int16

7.52.2.18 typedef int32_t Int32

7.52.2.19 typedef int64_t Int64

7.52.2.20 typedef int8_t Int8

7.52.2.21 typedef uint16_t UInt16

7.52.2.22 typedef uint32_t UInt32

7.52.2.23 typedef uint64_t UInt64

7.52.2.24 typedef uint8_t UInt8

7.53 BUrl.cpp File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <memory.h>
#include <BUrl.h>
#include <curl/curl.h>
```

7.54 BUrl.h File Reference

```
#include <stdio.h>
#include <BString.h>
#include <BError.h>
```

Classes

- class BUrl

  Basic access to a Url.

Defines

- #define BURL_H 1

7.54.1 Define Documentation

7.54.1.1 #define BURL_H 1

Index