

Beam-lib

2.15.1

Generated by Doxygen 1.8.14



# Contents

- 1 Main Page** **1**
- 1.1 Introduction ..... 1
- 1.2 Components ..... 2
- 1.3 API Examples ..... 2
  
- 2 Namespace Index** **3**
- 2.1 Namespace List ..... 3
  
- 3 Hierarchical Index** **5**
- 3.1 Class Hierarchy ..... 5
  
- 4 Class Index** **9**
- 4.1 Class List ..... 9
  
- 5 File Index** **13**
- 5.1 File List ..... 13
  
- 6 Namespace Documentation** **17**
- 6.1 Boapns Namespace Reference ..... 17
- 6.1.1 Variable Documentation ..... 17
- 6.1.1.1 apiVersion ..... 17

<b>7 Class Documentation</b>	<b>19</b>
7.1 BArray< T > Class Template Reference	19
7.1.1 Detailed Description	20
7.1.2 Member Typedef Documentation	20
7.1.2.1 SortFunc	20
7.1.3 Constructor & Destructor Documentation	20
7.1.3.1 BArray() [1/3]	20
7.1.3.2 BArray() [2/3]	20
7.1.3.3 BArray() [3/3]	20
7.1.4 Member Function Documentation	21
7.1.4.1 append() [1/2]	21
7.1.4.2 append() [2/2]	21
7.1.4.3 del()	21
7.1.4.4 insert()	21
7.1.4.5 number()	21
7.1.4.6 rear()	22
7.1.4.7 sort()	22
7.2 BAtomic< Type > Class Template Reference	22
7.2.1 Detailed Description	22
7.2.2 Constructor & Destructor Documentation	23
7.2.2.1 BAtomic()	23
7.2.3 Member Function Documentation	23
7.2.3.1 add()	23
7.2.3.2 getValue()	23
7.2.3.3 operator Type()	23
7.2.3.4 operator++() [1/2]	23
7.2.3.5 operator++() [2/2]	24
7.2.3.6 operator--() [1/2]	24
7.2.3.7 operator--() [2/2]	24
7.2.4 Member Data Documentation	24

---

7.2.4.1	ovalue	24
7.3	BAtomicCount Class Reference	24
7.3.1	Detailed Description	25
7.3.2	Constructor & Destructor Documentation	25
7.3.2.1	BAtomicCount()	25
7.3.3	Member Function Documentation	25
7.3.3.1	add()	25
7.3.3.2	getValue()	25
7.3.3.3	operator long()	25
7.3.3.4	operator++() [1/2]	25
7.3.3.5	operator++() [2/2]	26
7.3.3.6	operator--() [1/2]	26
7.3.3.7	operator--() [2/2]	26
7.3.4	Member Data Documentation	26
7.3.4.1	ovalue	26
7.4	BBuffer Class Reference	26
7.4.1	Constructor & Destructor Documentation	27
7.4.1.1	BBuffer()	27
7.4.1.2	~BBuffer()	27
7.4.2	Member Function Documentation	27
7.4.2.1	data()	28
7.4.2.2	resize()	28
7.4.2.3	setData()	28
7.4.2.4	setSize()	28
7.4.2.5	size()	28
7.4.2.6	writeData()	28
7.4.3	Member Data Documentation	29
7.4.3.1	odata	29
7.4.3.2	odataSize	29
7.4.3.3	osize	29

---

7.5	BBufferStore Class Reference	29
7.5.1	Constructor & Destructor Documentation	30
7.5.1.1	BBufferStore()	31
7.5.1.2	~BBufferStore()	31
7.5.2	Member Function Documentation	31
7.5.2.1	getHexString()	31
7.5.2.2	getPos()	31
7.5.2.3	pop() [1/15]	31
7.5.2.4	pop() [2/15]	31
7.5.2.5	pop() [3/15]	32
7.5.2.6	pop() [4/15]	32
7.5.2.7	pop() [5/15]	32
7.5.2.8	pop() [6/15]	32
7.5.2.9	pop() [7/15]	32
7.5.2.10	pop() [8/15]	32
7.5.2.11	pop() [9/15]	32
7.5.2.12	pop() [10/15]	33
7.5.2.13	pop() [11/15]	33
7.5.2.14	pop() [12/15]	33
7.5.2.15	pop() [13/15]	33
7.5.2.16	pop() [14/15]	33
7.5.2.17	pop() [15/15]	33
7.5.2.18	push() [1/15]	33
7.5.2.19	push() [2/15]	34
7.5.2.20	push() [3/15]	34
7.5.2.21	push() [4/15]	34
7.5.2.22	push() [5/15]	34
7.5.2.23	push() [6/15]	34
7.5.2.24	push() [7/15]	34
7.5.2.25	push() [8/15]	34

---

7.5.2.26	<a href="#">push()</a> [9/15]	35
7.5.2.27	<a href="#">push()</a> [10/15]	35
7.5.2.28	<a href="#">push()</a> [11/15]	35
7.5.2.29	<a href="#">push()</a> [12/15]	35
7.5.2.30	<a href="#">push()</a> [13/15]	35
7.5.2.31	<a href="#">push()</a> [14/15]	35
7.5.2.32	<a href="#">push()</a> [15/15]	35
7.5.2.33	<a href="#">setHexString()</a>	36
7.5.2.34	<a href="#">setPos()</a>	36
7.5.3	<a href="#">Member Data Documentation</a>	36
7.5.3.1	<a href="#">opos</a>	36
7.5.3.2	<a href="#">oswapBytes</a>	36
7.6	<a href="#">BComms Class Reference</a>	36
7.6.1	<a href="#">Member Enumeration Documentation</a>	37
7.6.1.1	<a href="#">Flush</a>	37
7.6.2	<a href="#">Constructor &amp; Destructor Documentation</a>	38
7.6.2.1	<a href="#">BComms()</a>	38
7.6.2.2	<a href="#">~BComms()</a>	38
7.6.3	<a href="#">Member Function Documentation</a>	38
7.6.3.1	<a href="#">connect()</a>	38
7.6.3.2	<a href="#">disconnect()</a>	38
7.6.3.3	<a href="#">eventQueue()</a>	38
7.6.3.4	<a href="#">flush()</a>	39
7.6.3.5	<a href="#">init()</a>	39
7.6.3.6	<a href="#">isConnected()</a>	39
7.6.3.7	<a href="#">packetMode()</a>	39
7.6.3.8	<a href="#">read()</a>	39
7.6.3.9	<a href="#">readAvailable()</a>	39
7.6.3.10	<a href="#">setPacketMode()</a>	39
7.6.3.11	<a href="#">setTimeout()</a>	40

7.6.3.12	wait()	40
7.6.3.13	write()	40
7.6.3.14	writeAvailable()	40
7.6.3.15	writeChunks()	40
7.6.4	Member Data Documentation	40
7.6.4.1	oconnected	40
7.6.4.2	oevent	41
7.6.4.3	oeventNum	41
7.6.4.4	oeventQueue	41
7.6.4.5	opacketMode	41
7.6.4.6	otimeout	41
7.7	BCond Class Reference	41
7.7.1	Constructor & Destructor Documentation	42
7.7.1.1	BCond()	42
7.7.1.2	~BCond()	42
7.7.2	Member Function Documentation	42
7.7.2.1	signal()	42
7.7.2.2	timedWait()	42
7.7.2.3	wait()	42
7.7.3	Member Data Documentation	43
7.7.3.1	ocond	43
7.7.3.2	omutex	43
7.8	BCondBool Class Reference	43
7.8.1	Detailed Description	44
7.8.2	Constructor & Destructor Documentation	44
7.8.2.1	BCondBool()	44
7.8.2.2	~BCondBool()	44
7.8.3	Member Function Documentation	44
7.8.3.1	clear()	44
7.8.3.2	operator int()	44



---

7.8.3.3	set()	44
7.8.3.4	timedWait()	45
7.8.3.5	value()	45
7.8.3.6	wait()	45
7.8.4	Member Data Documentation	45
7.8.4.1	ocond	45
7.8.4.2	omutex	45
7.8.4.3	ovalue	45
7.9	BCondInt Class Reference	46
7.9.1	Detailed Description	46
7.9.2	Constructor & Destructor Documentation	46
7.9.2.1	BCondInt()	47
7.9.2.2	~BCondInt()	47
7.9.3	Member Function Documentation	47
7.9.3.1	decrement()	47
7.9.3.2	increment()	47
7.9.3.3	operator++()	47
7.9.3.4	operator+=()	47
7.9.3.5	operator--()	48
7.9.3.6	operator-=()	48
7.9.3.7	setValue()	48
7.9.3.8	value()	48
7.9.3.9	waitLessThan()	48
7.9.3.10	waitLessThanOrEqual()	49
7.9.3.11	waitMoreThanOrEqual()	49
7.9.4	Member Data Documentation	49
7.9.4.1	ocond	49
7.9.4.2	omutex	49
7.9.4.3	ovalue	49
7.10	BCondResource Class Reference	50

---

---

7.10.1	Detailed Description	50
7.10.2	Constructor & Destructor Documentation	50
7.10.2.1	BCondResource()	50
7.10.2.2	~BCondResource()	50
7.10.3	Member Function Documentation	51
7.10.3.1	end()	51
7.10.3.2	inUse()	51
7.10.3.3	lock()	51
7.10.3.4	locked()	51
7.10.3.5	start()	51
7.10.3.6	unlock()	51
7.10.4	Member Data Documentation	52
7.10.4.1	ocond	52
7.10.4.2	olock	52
7.10.4.3	omutex	52
7.10.4.4	ouse	52
7.11	BCondValue Class Reference	52
7.11.1	Detailed Description	53
7.11.2	Constructor & Destructor Documentation	53
7.11.2.1	BCondValue()	53
7.11.2.2	~BCondValue()	54
7.11.3	Member Function Documentation	54
7.11.3.1	decrement()	54
7.11.3.2	increment()	54
7.11.3.3	operator++()	54
7.11.3.4	operator+=()	54
7.11.3.5	operator--()	55
7.11.3.6	operator-=()	55
7.11.3.7	setValue()	55
7.11.3.8	value()	55

---

---

7.11.3.9	<a href="#">waitLessThan()</a>	55
7.11.3.10	<a href="#">waitLessThanOrEqual()</a>	56
7.11.3.11	<a href="#">waitMoreThanOrEqual()</a>	56
7.11.4	<a href="#">Member Data Documentation</a>	56
7.11.4.1	<a href="#">ocond</a>	56
7.11.4.2	<a href="#">omutex</a>	56
7.11.4.3	<a href="#">ovalue</a>	56
7.12	<a href="#">BCondWrap Class Reference</a>	57
7.12.1	<a href="#">Constructor &amp; Destructor Documentation</a>	57
7.12.1.1	<a href="#">BCondWrap()</a>	58
7.12.1.2	<a href="#">~BCondWrap()</a>	58
7.12.2	<a href="#">Member Function Documentation</a>	58
7.12.2.1	<a href="#">decrement()</a>	58
7.12.2.2	<a href="#">diff()</a>	58
7.12.2.3	<a href="#">increment()</a>	58
7.12.2.4	<a href="#">operator++()</a>	58
7.12.2.5	<a href="#">operator+=()</a>	59
7.12.2.6	<a href="#">operator--()</a>	59
7.12.2.7	<a href="#">operator-=()</a>	59
7.12.2.8	<a href="#">setValue()</a>	59
7.12.2.9	<a href="#">value()</a>	59
7.12.2.10	<a href="#">waitLessThan()</a>	59
7.12.2.11	<a href="#">waitLessThanOrEqual()</a>	60
7.12.2.12	<a href="#">waitMoreThanOrEqual()</a>	60
7.12.3	<a href="#">Member Data Documentation</a>	60
7.12.3.1	<a href="#">ocond</a>	60
7.12.3.2	<a href="#">omutex</a>	60
7.12.3.3	<a href="#">ovalue</a>	60
7.13	<a href="#">BConfig Class Reference</a>	61
7.13.1	<a href="#">Detailed Description</a>	61

---

---

7.13.2	Member Function Documentation	61
7.13.2.1	close()	61
7.13.2.2	fileName()	62
7.13.2.3	findValue()	62
7.13.2.4	open()	62
7.13.2.5	read()	62
7.13.2.6	write()	62
7.13.3	Member Data Documentation	62
7.13.3.1	ofile	62
7.13.3.2	ofilename	62
7.13.3.3	olock	63
7.14	BDataChunk Class Reference	63
7.14.1	Constructor & Destructor Documentation	63
7.14.1.1	BDataChunk()	63
7.14.2	Member Data Documentation	63
7.14.2.1	data	63
7.14.2.2	size	64
7.15	BDate Class Reference	64
7.15.1	Constructor & Destructor Documentation	65
7.15.1.1	BDate() [1/2]	65
7.15.1.2	BDate() [2/2]	65
7.15.1.3	~BDate()	65
7.15.2	Member Function Documentation	65
7.15.2.1	clear()	65
7.15.2.2	compare()	66
7.15.2.3	day()	66
7.15.2.4	daysInMonth()	66
7.15.2.5	getDate()	66
7.15.2.6	getString()	66
7.15.2.7	getStringFormatted()	66

---

7.15.2.8	isLeap()	67
7.15.2.9	isSet()	67
7.15.2.10	month()	67
7.15.2.11	operator BString()	67
7.15.2.12	operator"!=(	67
7.15.2.13	operator<(	67
7.15.2.14	operator<=(	67
7.15.2.15	operator==(	68
7.15.2.16	operator>(	68
7.15.2.17	operator>=(	68
7.15.2.18	set() [1/2]	68
7.15.2.19	set() [2/2]	68
7.15.2.20	setFirst()	68
7.15.2.21	setLast()	69
7.15.2.22	setNow()	69
7.15.2.23	setString()	69
7.15.2.24	setYDay()	69
7.15.2.25	yday()	69
7.15.2.26	year()	69
7.15.3	Member Data Documentation	69
7.15.3.1	oyday	70
7.15.3.2	oyear	70
7.16	BDebugBacktrace Class Reference	70
7.16.1	Constructor & Destructor Documentation	70
7.16.1.1	BDebugBacktrace()	70
7.16.1.2	~BDebugBacktrace()	70
7.16.2	Member Function Documentation	71
7.16.2.1	dumpBacktrace()	71
7.16.2.2	dumpBacktraceFile()	71
7.16.2.3	dumpBacktraceStdout()	71

---

7.16.2.4	<code>dumpBacktraceSyslog()</code>	71
7.17	<code>BDict&lt; Type &gt;</code> Class Template Reference	71
7.17.1	Member Typedef Documentation	72
7.17.1.1	<code>iterator</code>	72
7.17.2	Constructor & Destructor Documentation	73
7.17.2.1	<code>BDict()</code> [1/2]	73
7.17.2.2	<code>BDict()</code> [2/2]	73
7.17.3	Member Function Documentation	73
7.17.3.1	<code>append()</code> [1/2]	73
7.17.3.2	<code>append()</code> [2/2]	73
7.17.3.3	<code>clear()</code>	73
7.17.3.4	<code>del()</code> [1/2]	74
7.17.3.5	<code>del()</code> [2/2]	74
7.17.3.6	<code>find()</code>	74
7.17.3.7	<code>hashAdd()</code>	74
7.17.3.8	<code>hashDelete()</code>	74
7.17.3.9	<code>hashFind()</code>	75
7.17.3.10	<code>hashPrint()</code>	75
7.17.3.11	<code>hasKey()</code>	75
7.17.3.12	<code>insert()</code>	75
7.17.3.13	<code>key()</code>	75
7.17.3.14	<code>operator+()</code>	75
7.17.3.15	<code>operator=()</code>	76
7.17.3.16	<code>operator[]()</code> [1/4]	76
7.17.3.17	<code>operator[]()</code> [2/4]	76
7.17.3.18	<code>operator[]()</code> [3/4]	76
7.17.3.19	<code>operator[]()</code> [4/4]	76
7.17.4	Member Data Documentation	76
7.17.4.1	<code>ohashLists</code>	76
7.17.4.2	<code>ohashSize</code>	77

7.18	<a href="#">BDictItem&lt; Type &gt; Class Template Reference</a>	77
7.18.1	<a href="#">Detailed Description</a>	77
7.18.2	<a href="#">Constructor &amp; Destructor Documentation</a>	77
7.18.2.1	<a href="#">BDictItem()</a>	77
7.18.3	<a href="#">Member Data Documentation</a>	77
7.18.3.1	<a href="#">key</a>	78
7.18.3.2	<a href="#">value</a>	78
7.19	<a href="#">BDictMap&lt; Value &gt; Class Template Reference</a>	78
7.19.1	<a href="#">Detailed Description</a>	79
7.19.2	<a href="#">Member Typedef Documentation</a>	79
7.19.2.1	<a href="#">iterator</a>	79
7.19.3	<a href="#">Member Function Documentation</a>	79
7.19.3.1	<a href="#">clear()</a>	79
7.19.3.2	<a href="#">del() [1/2]</a>	79
7.19.3.3	<a href="#">del() [2/2]</a>	79
7.19.3.4	<a href="#">hasKey()</a>	80
7.19.3.5	<a href="#">isEnd()</a>	80
7.19.3.6	<a href="#">key()</a>	80
7.19.3.7	<a href="#">next()</a>	80
7.19.3.8	<a href="#">operator[]() [1/2]</a>	80
7.19.3.9	<a href="#">operator[]() [2/2]</a>	80
7.19.3.10	<a href="#">size()</a>	81
7.19.3.11	<a href="#">start()</a>	81
7.20	<a href="#">BDir Class Reference</a>	81
7.20.1	<a href="#">Detailed Description</a>	82
7.20.2	<a href="#">Constructor &amp; Destructor Documentation</a>	82
7.20.2.1	<a href="#">BDir() [1/2]</a>	82
7.20.2.2	<a href="#">BDir() [2/2]</a>	82
7.20.2.3	<a href="#">~BDir()</a>	82
7.20.3	<a href="#">Member Function Documentation</a>	82

---

7.20.3.1	clear()	82
7.20.3.2	entryName()	83
7.20.3.3	entryStat()	83
7.20.3.4	entryStat64()	83
7.20.3.5	error()	83
7.20.3.6	open()	83
7.20.3.7	read()	83
7.20.3.8	setSort()	84
7.20.3.9	setWild()	84
7.20.4	Member Data Documentation	84
7.20.4.1	odirname	84
7.20.4.2	oerror	84
7.20.4.3	osort	84
7.20.4.4	owild	84
7.21	BDuration Class Reference	85
7.21.1	Constructor & Destructor Documentation	85
7.21.1.1	BDuration() [1/2]	86
7.21.1.2	BDuration() [2/2]	86
7.21.1.3	~BDuration()	86
7.21.2	Member Function Documentation	86
7.21.2.1	addMicroSeconds()	86
7.21.2.2	addMilliSeconds()	86
7.21.2.3	addSeconds()	86
7.21.2.4	clear()	87
7.21.2.5	getMicroSeconds()	87
7.21.2.6	getSeconds()	87
7.21.2.7	getString()	87
7.21.2.8	hour()	87
7.21.2.9	microSecond()	87
7.21.2.10	minute()	87



---

7.21.2.11	second()	88
7.21.2.12	set()	88
7.21.2.13	setString()	88
7.21.3	Member Data Documentation	88
7.21.3.1	ohour	88
7.21.3.2	omicroSecond	88
7.21.3.3	ominute	88
7.21.3.4	osecond	89
7.21.3.5	ospare	89
7.22	BEntry Class Reference	89
7.22.1	Detailed Description	90
7.22.2	Constructor & Destructor Documentation	90
7.22.2.1	BEntry() [1/3]	90
7.22.2.2	BEntry() [2/3]	90
7.22.2.3	BEntry() [3/3]	90
7.22.3	Member Function Documentation	90
7.22.3.1	getName()	90
7.22.3.2	getValue()	91
7.22.3.3	line()	91
7.22.3.4	print()	91
7.22.3.5	setLine()	91
7.22.3.6	setName()	91
7.22.3.7	setValue()	91
7.22.4	Member Data Documentation	92
7.22.4.1	oname	92
7.22.4.2	ovalue	92
7.23	BEntryFile Class Reference	92
7.23.1	Detailed Description	93
7.23.2	Constructor & Destructor Documentation	93
7.23.2.1	BEntryFile() [1/2]	93

7.23.2.2	BEntryFile() [2/2]	93
7.23.2.3	~BEntryFile()	93
7.23.3	Member Function Documentation	93
7.23.3.1	clear()	93
7.23.3.2	filename()	94
7.23.3.3	open()	94
7.23.3.4	read()	94
7.23.3.5	write()	94
7.23.3.6	writeln()	94
7.23.4	Member Data Documentation	94
7.23.4.1	ocomments	94
7.23.4.2	ofilename	95
7.24	BEntryList Class Reference	95
7.24.1	Detailed Description	96
7.24.2	Constructor & Destructor Documentation	96
7.24.2.1	BEntryList()	96
7.24.3	Member Function Documentation	96
7.24.3.1	clear()	96
7.24.3.2	del()	96
7.24.3.3	deleteEntry()	97
7.24.3.4	find()	97
7.24.3.5	findValue()	97
7.24.3.6	getString()	97
7.24.3.7	insert()	97
7.24.3.8	isSet()	98
7.24.3.9	operator=()	98
7.24.3.10	print()	98
7.24.3.11	setValue()	98
7.24.3.12	setValueRaw()	98
7.24.4	Member Data Documentation	98

---

7.24.4.1	olastPos	99
7.25	BError Class Reference	99
7.25.1	Detailed Description	100
7.25.2	Constructor & Destructor Documentation	100
7.25.2.1	BError() [1/2]	100
7.25.2.2	BError() [2/2]	100
7.25.3	Member Function Documentation	100
7.25.3.1	clear()	100
7.25.3.2	copy()	101
7.25.3.3	getErrorNo()	101
7.25.3.4	getNumber()	101
7.25.3.5	getString()	101
7.25.3.6	num()	101
7.25.3.7	operator int()	101
7.25.3.8	set()	102
7.25.3.9	setError()	102
7.25.3.10	str()	102
7.25.4	Member Data Documentation	102
7.25.4.1	oerrNo	102
7.25.4.2	oerrStr	102
7.26	BErrorTime Class Reference	103
7.26.1	Detailed Description	103
7.26.2	Member Enumeration Documentation	103
7.26.2.1	Type	103
7.26.3	Constructor & Destructor Documentation	104
7.26.3.1	BErrorTime()	104
7.26.4	Member Function Documentation	104
7.26.4.1	clear()	104
7.26.4.2	copy()	104
7.26.4.3	getErrorNo()	104

7.26.4.4	getString()	105
7.26.4.5	getTime()	105
7.26.4.6	operator int()	105
7.26.4.7	set()	105
7.26.5	Member Data Documentation	105
7.26.5.1	oerrNo	105
7.26.5.2	oerrStr	105
7.26.5.3	oerrTime	106
7.27	BEvent Class Reference	106
7.27.1	Constructor & Destructor Documentation	106
7.27.1.1	BEvent()	106
7.27.2	Member Function Documentation	106
7.27.2.1	arg()	106
7.27.2.2	type()	107
7.27.3	Member Data Documentation	107
7.27.3.1	oarg	107
7.27.3.2	otype	107
7.28	BEvent1 Class Reference	107
7.28.1	Detailed Description	108
7.28.2	Constructor & Destructor Documentation	108
7.28.2.1	BEvent1()	108
7.28.2.2	~BEvent1()	108
7.28.3	Member Function Documentation	108
7.28.3.1	getBinary()	108
7.28.3.2	getType()	108
7.28.3.3	setBinary()	109
7.28.4	Member Data Documentation	109
7.28.4.1	otype	109
7.29	BEvent1Error Class Reference	109
7.29.1	Constructor & Destructor Documentation	109

---

7.29.1.1	BEvent1Error()	110
7.29.2	Member Function Documentation	110
7.29.2.1	getBinary()	110
7.29.2.2	setBinary()	110
7.30	BEvent1Int Class Reference	110
7.30.1	Detailed Description	111
7.30.2	Constructor & Destructor Documentation	111
7.30.2.1	BEvent1Int()	111
7.30.2.2	~BEvent1Int()	111
7.30.3	Member Function Documentation	111
7.30.3.1	clear()	111
7.30.3.2	getEvent()	111
7.30.3.3	getFd()	112
7.30.3.4	sendEvent()	112
7.30.4	Member Data Documentation	112
7.30.4.1	ofds	112
7.31	BEvent1Pipe Class Reference	112
7.31.1	Detailed Description	113
7.31.2	Constructor & Destructor Documentation	113
7.31.2.1	BEvent1Pipe()	113
7.31.2.2	~BEvent1Pipe()	113
7.31.3	Member Function Documentation	113
7.31.3.1	clear()	113
7.31.3.2	getEvent()	113
7.31.3.3	getReceiveFd()	114
7.31.3.4	sendEvent()	114
7.31.4	Member Data Documentation	114
7.31.4.1	ofds	114
7.32	BEventPipe Class Reference	114
7.32.1	Detailed Description	115

---

7.32.2	Constructor & Destructor Documentation	115
7.32.2.1	BEventPipe()	115
7.32.2.2	~BEventPipe()	115
7.32.3	Member Function Documentation	115
7.32.3.1	clear()	115
7.32.3.2	getFd()	115
7.32.3.3	read()	116
7.32.3.4	readAvailable()	116
7.32.3.5	write()	116
7.32.3.6	writeAvailable()	116
7.32.4	Member Data Documentation	116
7.32.4.1	ofds	116
7.33	BFifo< Type > Class Template Reference	117
7.33.1	Constructor & Destructor Documentation	118
7.33.1.1	BFifo()	118
7.33.1.2	~BFifo()	118
7.33.2	Member Function Documentation	118
7.33.2.1	clear()	118
7.33.2.2	operator[]()	118
7.33.2.3	read() [1/2]	119
7.33.2.4	read() [2/2]	119
7.33.2.5	readAvailable()	119
7.33.2.6	readAvailableChunk()	119
7.33.2.7	readData() [1/2]	119
7.33.2.8	readData() [2/2]	120
7.33.2.9	readDone()	120
7.33.2.10	readPos()	120
7.33.2.11	rebase()	120
7.33.2.12	resize()	120
7.33.2.13	size()	121

---

7.33.2.14 write() [1/2]	121
7.33.2.15 write() [2/2]	121
7.33.2.16 writeAvailable()	121
7.33.2.17 writeAvailableChunk()	121
7.33.2.18 writeBackup()	122
7.33.2.19 writeData() [1/2]	122
7.33.2.20 writeData() [2/2]	122
7.33.2.21 writeDone()	122
7.33.2.22 writePos()	122
7.33.3 Member Data Documentation	123
7.33.3.1 odata	123
7.33.3.2 oreadPos	123
7.33.3.3 osize	123
7.33.3.4 owritePos	123
7.34 BFifoCirc< Type > Class Template Reference	123
7.34.1 Detailed Description	125
7.34.2 Member Enumeration Documentation	125
7.34.2.1 anonymous enum	125
7.34.3 Constructor & Destructor Documentation	125
7.34.3.1 BFifoCirc()	125
7.34.3.2 ~BFifoCirc()	125
7.34.4 Member Function Documentation	125
7.34.4.1 clear()	126
7.34.4.2 mapCircularBuffer()	126
7.34.4.3 operator[]()	126
7.34.4.4 read()	126
7.34.4.5 readAvailable()	126
7.34.4.6 readData()	126
7.34.4.7 readDone()	127
7.34.4.8 readWaitAvailable()	127

7.34.4.9	size()	127
7.34.4.10	unmapCircularBuffer()	127
7.34.4.11	write()	127
7.34.4.12	writeAvailable()	128
7.34.4.13	writeData()	128
7.34.4.14	writeDone()	128
7.34.4.15	writeWaitAvailable()	128
7.34.5	Member Data Documentation	128
7.34.5.1	odata	128
7.34.5.2	olock	129
7.34.5.3	oreadPos	129
7.34.5.4	osize	129
7.34.5.5	ovmSize	129
7.34.5.6	owriteNumFifoSamples	129
7.34.5.7	owritePos	129
7.35	BFifoCircPos Class Reference	130
7.35.1	Detailed Description	130
7.35.2	Constructor & Destructor Documentation	130
7.35.2.1	BFifoCircPos()	130
7.35.3	Member Function Documentation	130
7.35.3.1	difference()	131
7.35.3.2	increment()	131
7.35.3.3	operator int()	131
7.35.3.4	operator"!=(	131
7.35.3.5	operator"+=(	131
7.35.3.6	operator"==(	131
7.35.3.7	pos()	132
7.35.3.8	set()	132
7.35.3.9	setSize()	132
7.35.4	Member Data Documentation	132



---

7.35.4.1	opos	132
7.35.4.2	osize	132
7.36	BFile Class Reference	133
7.36.1	Detailed Description	134
7.36.2	Constructor & Destructor Documentation	134
7.36.2.1	BFile() [1/2]	134
7.36.2.2	BFile() [2/2]	134
7.36.2.3	~BFile()	134
7.36.3	Member Function Documentation	135
7.36.3.1	close()	135
7.36.3.2	fgets()	135
7.36.3.3	fileName()	135
7.36.3.4	flush()	135
7.36.3.5	getFd()	135
7.36.3.6	isEnd()	136
7.36.3.7	isOpen()	136
7.36.3.8	length()	136
7.36.3.9	open() [1/3]	136
7.36.3.10	open() [2/3]	136
7.36.3.11	open() [3/3]	136
7.36.3.12	operator=()	137
7.36.3.13	position()	137
7.36.3.14	printf()	137
7.36.3.15	read()	137
7.36.3.16	readString()	137
7.36.3.17	seek()	137
7.36.3.18	setVBuf()	138
7.36.3.19	truncate()	138
7.36.3.20	write()	138
7.36.3.21	writeString()	138

7.36.4	Member Data Documentation . . . . .	138
7.36.4.1	ofile . . . . .	138
7.36.4.2	ofilename . . . . .	139
7.36.4.3	omode . . . . .	139
7.37	BFileCsv Class Reference . . . . .	139
7.37.1	Constructor & Destructor Documentation . . . . .	139
7.37.1.1	BFileCsv() . . . . .	139
7.37.2	Member Function Documentation . . . . .	140
7.37.2.1	readCsv() . . . . .	140
7.37.2.2	writeCsv() . . . . .	140
7.37.3	Member Data Documentation . . . . .	140
7.37.3.1	oseparator . . . . .	140
7.38	BFileData Class Reference . . . . .	140
7.38.1	Member Function Documentation . . . . .	141
7.38.1.1	del() . . . . .	141
7.38.1.2	find() . . . . .	141
7.38.1.3	getNextId() . . . . .	141
7.38.1.4	open() . . . . .	141
7.38.1.5	read() . . . . .	142
7.38.1.6	write() [1/2] . . . . .	142
7.38.1.7	write() [2/2] . . . . .	142
7.38.2	Member Data Documentation . . . . .	142
7.38.2.1	ofilename . . . . .	142
7.39	Blter Class Reference . . . . .	142
7.39.1	Detailed Description . . . . .	143
7.39.2	Constructor & Destructor Documentation . . . . .	143
7.39.2.1	Blter() . . . . .	143
7.39.3	Member Function Documentation . . . . .	143
7.39.3.1	operator BNode *() . . . . .	143
7.39.3.2	operator==( ) . . . . .	143

---

7.39.3.3	valid()	143
7.39.4	Member Data Documentation	143
7.39.4.1	oi	144
7.40	BList< T > Class Template Reference	144
7.40.1	Detailed Description	146
7.40.2	Member Typedef Documentation	146
7.40.2.1	SortFunc	146
7.40.3	Constructor & Destructor Documentation	146
7.40.3.1	BList() [1/2]	147
7.40.3.2	BList() [2/2]	147
7.40.3.3	~BList()	147
7.40.4	Member Function Documentation	147
7.40.4.1	append() [1/2]	147
7.40.4.2	append() [2/2]	147
7.40.4.3	begin()	148
7.40.4.4	clear()	148
7.40.4.5	del()	148
7.40.4.6	deleteFirst()	148
7.40.4.7	deleteLast()	148
7.40.4.8	end() [1/2]	149
7.40.4.9	end() [2/2]	149
7.40.4.10	front()	149
7.40.4.11	get() [1/2]	149
7.40.4.12	get() [2/2]	149
7.40.4.13	goTo()	150
7.40.4.14	has()	150
7.40.4.15	insert()	150
7.40.4.16	insertAfter()	150
7.40.4.17	isEnd()	150
7.40.4.18	next()	151

7.40.4.19	nodeCreate() [1/2]	151
7.40.4.20	nodeCreate() [2/2]	151
7.40.4.21	nodeGet() [1/2]	151
7.40.4.22	nodeGet() [2/2]	151
7.40.4.23	number()	151
7.40.4.24	operator+()	152
7.40.4.25	operator=()	152
7.40.4.26	operator[]() [1/4]	152
7.40.4.27	operator[]() [2/4]	152
7.40.4.28	operator[]() [3/4]	152
7.40.4.29	operator[]() [4/4]	152
7.40.4.30	pop()	153
7.40.4.31	position()	153
7.40.4.32	prev()	153
7.40.4.33	push()	153
7.40.4.34	queueAdd()	153
7.40.4.35	queueGet()	154
7.40.4.36	rear()	154
7.40.4.37	size()	154
7.40.4.38	sort() [1/2]	154
7.40.4.39	sort() [2/2]	154
7.40.4.40	start()	155
7.40.4.41	swap()	155
7.40.5	Member Data Documentation	155
7.40.5.1	olength	155
7.40.5.2	onodes	155
7.41	BMutex Class Reference	155
7.41.1	Detailed Description	156
7.41.2	Member Enumeration Documentation	156
7.41.2.1	Type	156

7.41.3	Constructor & Destructor Documentation	156
7.41.3.1	BMutex() [1/2]	157
7.41.3.2	BMutex() [2/2]	157
7.41.3.3	~BMutex()	157
7.41.4	Member Function Documentation	157
7.41.4.1	lock()	157
7.41.4.2	operator=()	157
7.41.4.3	timedLock()	157
7.41.4.4	tryLock()	158
7.41.4.5	unlock()	158
7.41.5	Member Data Documentation	158
7.41.5.1	omutex	158
7.42	BMutexLock Class Reference	158
7.42.1	Constructor & Destructor Documentation	158
7.42.1.1	BMutexLock()	159
7.42.1.2	~BMutexLock()	159
7.42.2	Member Function Documentation	159
7.42.2.1	lock()	159
7.42.2.2	unlock()	159
7.42.3	Member Data Documentation	159
7.42.3.1	olock	159
7.43	BMySQL Class Reference	159
7.43.1	Constructor & Destructor Documentation	160
7.43.1.1	BMySQL()	160
7.43.1.2	~BMySQL()	160
7.43.2	Member Function Documentation	160
7.43.2.1	close()	161
7.43.2.2	db()	161
7.43.2.3	del()	161
7.43.2.4	escapeString()	161

7.43.2.5	flush()	161
7.43.2.6	get()	161
7.43.2.7	insert()	162
7.43.2.8	open()	162
7.43.2.9	query()	162
7.43.2.10	setDebug()	162
7.43.2.11	update()	162
7.43.3	Member Data Documentation	162
7.43.3.1	odb	163
7.43.3.2	odebug	163
7.43.3.3	olock	163
7.43.3.4	oopened	163
7.44	BNameValue< T > Class Template Reference	163
7.44.1	Constructor & Destructor Documentation	164
7.44.1.1	BNameValue() [1/2]	164
7.44.1.2	BNameValue() [2/2]	164
7.44.2	Member Function Documentation	164
7.44.2.1	getName()	164
7.44.2.2	getValue()	164
7.44.3	Member Data Documentation	164
7.44.3.1	oname	164
7.44.3.2	ovalue	165
7.45	BNameValueList< T > Class Template Reference	165
7.45.1	Member Function Documentation	165
7.45.1.1	find()	165
7.45.1.2	findPos()	165
7.46	BNode Class Reference	166
7.46.1	Constructor & Destructor Documentation	166
7.46.1.1	BNode()	166
7.46.2	Member Data Documentation	166

---

7.46.2.1	next	166
7.46.2.2	prev	166
7.47	BoapClientObject Class Reference	167
7.47.1	Constructor & Destructor Documentation	168
7.47.1.1	BoapClientObject() [1/2]	168
7.47.1.2	~BoapClientObject()	168
7.47.1.3	BoapClientObject() [2/2]	168
7.47.2	Member Function Documentation	168
7.47.2.1	checkApiVersion()	168
7.47.2.2	connectService() [1/2]	169
7.47.2.3	connectService() [2/2]	169
7.47.2.4	disconnectService()	169
7.47.2.5	getServiceName()	169
7.47.2.6	handleReconnect()	169
7.47.2.7	performCall() [1/2]	169
7.47.2.8	performCall() [2/2]	170
7.47.2.9	performRecv() [1/2]	170
7.47.2.10	performRecv() [2/2]	170
7.47.2.11	performSend() [1/2]	170
7.47.2.12	performSend() [2/2]	170
7.47.2.13	ping()	170
7.47.2.14	pingLocked()	171
7.47.2.15	setConnectionPriority()	171
7.47.2.16	setMaxLength()	171
7.47.2.17	setTimeout()	171
7.47.3	Member Data Documentation	171
7.47.3.1	oapiVersion	171
7.47.3.2	oconnected	171
7.47.3.3	olock	172
7.47.3.4	omaxLength	172

7.47.3.5	oname	172
7.47.3.6	opriority	172
7.47.3.7	oreconnect	172
7.47.3.8	orx	172
7.47.3.9	oservice	172
7.47.3.10	otimeout	173
7.47.3.11	otx	173
7.48	Boaps::BoapEntry Class Reference	173
7.48.1	Constructor & Destructor Documentation	173
7.48.1.1	BoapEntry()	173
7.48.2	Member Data Documentation	174
7.48.2.1	addressList	174
7.48.2.2	hostName	174
7.48.2.3	name	174
7.48.2.4	port	174
7.48.2.5	service	174
7.49	BoapFuncEntry Class Reference	174
7.49.1	Constructor & Destructor Documentation	175
7.49.1.1	BoapFuncEntry() [1/2]	175
7.49.1.2	BoapFuncEntry() [2/2]	175
7.49.2	Member Data Documentation	175
7.49.2.1	ocmd [1/2]	175
7.49.2.2	ocmd [2/2]	175
7.49.2.3	ofunc	175
7.50	BoapMc1Comms Class Reference	176
7.50.1	Constructor & Destructor Documentation	177
7.50.1.1	BoapMc1Comms()	177
7.50.1.2	~BoapMc1Comms()	177
7.50.2	Member Function Documentation	177
7.50.2.1	getApiVersion()	177



---

7.50.2.2	packetRx()	178
7.50.2.3	packetRxData()	178
7.50.2.4	packetRxEnd()	178
7.50.2.5	packetTx()	178
7.50.2.6	processRequest()	178
7.50.2.7	processRequests()	178
7.50.2.8	processRx()	179
7.50.2.9	setAddress()	179
7.50.2.10	setComms() [1/2]	179
7.50.2.11	setComms() [2/2]	179
7.50.2.12	setCommsMode()	179
7.50.2.13	setTimeout()	180
7.50.3	Member Data Documentation	180
7.50.3.1	oaddressFrom	180
7.50.3.2	oaddressTo	180
7.50.3.3	oapiVersion	180
7.50.3.4	ocomms	180
7.50.3.5	oerror	180
7.50.3.6	ohalfDuplex	181
7.50.3.7	olockCall	181
7.50.3.8	olockTx	181
7.50.3.9	opacketRpcCmd	181
7.50.3.10	opacketRpcDoneSema	181
7.50.3.11	opacketRpcSema	181
7.50.3.12	opacketRx	182
7.50.3.13	opacketRxBase	182
7.50.3.14	opacketTx	182
7.50.3.15	opacketTxBase	182
7.50.3.16	oreqSize	182
7.50.3.17	othreaded	182

7.50.3.18 otimeout	183
7.51 BoapMc1Error Struct Reference	183
7.51.1 Member Data Documentation	183
7.51.1.1 number	183
7.51.1.2 string	183
7.52 BoapMc1Packet Class Reference	184
7.52.1 Member Data Documentation	184
7.52.1.1 data	184
7.52.1.2 head	184
7.53 BoapMc1PacketHead Struct Reference	184
7.53.1 Member Data Documentation	185
7.53.1.1 addressFrom	185
7.53.1.2 addressTo	185
7.53.1.3 checksum	185
7.53.1.4 cmd	185
7.53.1.5 error	185
7.53.1.6 length	185
7.53.1.7 magic	186
7.54 BoapMcClientObject Class Reference	186
7.54.1 Constructor & Destructor Documentation	186
7.54.1.1 BoapMcClientObject()	186
7.54.1.2 ~BoapMcClientObject()	187
7.54.2 Member Function Documentation	187
7.54.2.1 getApiVersion()	187
7.54.2.2 performCall()	187
7.54.2.3 performRecv()	187
7.54.2.4 performSend()	187
7.54.2.5 setAddress()	187
7.54.3 Member Data Documentation	188
7.54.3.1 oaddressFrom	188

---

7.54.3.2	<code>oaddressTo</code>	188
7.54.3.3	<code>oapiVersion</code>	188
7.54.3.4	<code>ocomms</code>	188
7.54.3.5	<code>opacket</code>	188
7.55	BoapMcComms Class Reference	188
7.55.1	Constructor & Destructor Documentation	190
7.55.1.1	<code>BoapMcComms()</code>	190
7.55.1.2	<code>~BoapMcComms()</code>	190
7.55.2	Member Function Documentation	190
7.55.2.1	<code>getApiVersion()</code>	190
7.55.2.2	<code>packetRecv()</code>	191
7.55.2.3	<code>packetSend()</code>	191
7.55.2.4	<code>performCall()</code>	191
7.55.2.5	<code>performSend()</code>	191
7.55.2.6	<code>processPacket()</code>	191
7.55.2.7	<code>processRequest()</code>	191
7.55.2.8	<code>processRequests()</code>	192
7.55.2.9	<code>processRx()</code>	192
7.55.2.10	<code>setAddress()</code>	192
7.55.2.11	<code>setComms()</code> [1/2]	192
7.55.2.12	<code>setComms()</code> [2/2]	192
7.55.2.13	<code>setCommsMode()</code>	193
7.55.2.14	<code>setTimeout()</code>	193
7.55.3	Member Data Documentation	193
7.55.3.1	<code>oaddressFrom</code>	193
7.55.3.2	<code>oaddressTo</code>	193
7.55.3.3	<code>oapiVersion</code>	193
7.55.3.4	<code>ocomms</code>	193
7.55.3.5	<code>olockCall</code>	194
7.55.3.6	<code>olockTx</code>	194

7.55.3.7	opacket	194
7.55.3.8	opacketReqQueue	194
7.55.3.9	opacketReqRx	194
7.55.3.10	opacketReqTx	194
7.55.3.11	opacketRx	195
7.55.3.12	opacketRxSema	195
7.55.3.13	opacketTx	195
7.55.3.14	opacketTxQueue	195
7.55.3.15	opacketTxQueueWriteNum	195
7.55.3.16	opacketTxSema	195
7.55.3.17	oslave	196
7.55.3.18	othreaded	196
7.55.3.19	otimeout	196
7.56	BoapMcPacket Class Reference	196
7.56.1	Member Data Documentation	196
7.56.1.1	data	196
7.56.1.2	head	197
7.57	BoapMcPacketHead Struct Reference	197
7.57.1	Member Data Documentation	197
7.57.1.1	addressFrom	197
7.57.1.2	addressTo	197
7.57.1.3	checksum	197
7.57.1.4	cmd	198
7.57.1.5	error	198
7.57.1.6	length	198
7.58	BoapMcServiceObject Class Reference	198
7.58.1	Constructor & Destructor Documentation	198
7.58.1.1	BoapMcServiceObject()	199
7.58.1.2	~BoapMcServiceObject()	199
7.58.2	Member Function Documentation	199

7.58.2.1	process()	199
7.58.2.2	processEvent()	199
7.58.2.3	sendEvent()	199
7.58.3	Member Data Documentation	199
7.58.3.1	oapiVersion	199
7.59	BoapMcSignalObject Class Reference	200
7.59.1	Constructor & Destructor Documentation	200
7.59.1.1	BoapMcSignalObject()	200
7.59.2	Member Function Documentation	200
7.59.2.1	performSend()	200
7.59.3	Member Data Documentation	200
7.59.3.1	ocomms	201
7.60	Boapns::Boapns Class Reference	201
7.60.1	Constructor & Destructor Documentation	201
7.60.1.1	Boapns()	201
7.60.2	Member Function Documentation	202
7.60.2.1	addEntry()	202
7.60.2.2	delEntry()	202
7.60.2.3	getEntry()	202
7.60.2.4	getEntryList()	202
7.60.2.5	getNewName()	202
7.60.2.6	getVersion()	202
7.61	BoapPacket Class Reference	203
7.61.1	Constructor & Destructor Documentation	204
7.61.1.1	BoapPacket() [1/2]	204
7.61.1.2	~BoapPacket() [1/2]	204
7.61.1.3	BoapPacket() [2/2]	204
7.61.1.4	~BoapPacket() [2/2]	204
7.61.2	Member Function Documentation	204
7.61.2.1	data()	205

7.61.2.2	getCmd()	205
7.61.2.3	nbytes()	205
7.61.2.4	peekHead()	205
7.61.2.5	pop() [1/10]	205
7.61.2.6	pop() [2/10]	205
7.61.2.7	pop() [3/10]	205
7.61.2.8	pop() [4/10]	206
7.61.2.9	pop() [5/10]	206
7.61.2.10	pop() [6/10]	206
7.61.2.11	pop() [7/10]	206
7.61.2.12	pop() [8/10]	206
7.61.2.13	pop() [9/10]	206
7.61.2.14	pop() [10/10]	206
7.61.2.15	popHead() [1/2]	207
7.61.2.16	popHead() [2/2]	207
7.61.2.17	push() [1/10]	207
7.61.2.18	push() [2/10]	207
7.61.2.19	push() [3/10]	207
7.61.2.20	push() [4/10]	207
7.61.2.21	push() [5/10]	207
7.61.2.22	push() [6/10]	208
7.61.2.23	push() [7/10]	208
7.61.2.24	push() [8/10]	208
7.61.2.25	push() [9/10]	208
7.61.2.26	push() [10/10]	208
7.61.2.27	pushHead() [1/2]	208
7.61.2.28	pushHead() [2/2]	208
7.61.2.29	resize()	209
7.61.2.30	setData()	209
7.61.2.31	updateHead()	209

7.61.2.32	updateLen()	209
7.61.3	Member Data Documentation	209
7.61.3.1	odata	209
7.61.3.2	onbytes	209
7.61.3.3	opos	209
7.61.3.4	osize	210
7.62	BoapPacketHead Struct Reference	210
7.62.1	Member Data Documentation	210
7.62.1.1	cmd [1/2]	210
7.62.1.2	cmd [2/2]	210
7.62.1.3	length [1/2]	211
7.62.1.4	length [2/2]	211
7.62.1.5	reserved	211
7.62.1.6	service [1/2]	211
7.62.1.7	service [2/2]	211
7.62.1.8	type [1/2]	211
7.62.1.9	type [2/2]	211
7.63	BoapServer Class Reference	212
7.63.1	Member Enumeration Documentation	213
7.63.1.1	anonymous enum	213
7.63.2	Constructor & Destructor Documentation	213
7.63.2.1	BoapServer() [1/2]	213
7.63.2.2	~BoapServer()	214
7.63.2.3	BoapServer() [2/2]	214
7.63.3	Member Function Documentation	214
7.63.3.1	addObject() [1/2]	214
7.63.3.2	addObject() [2/2]	214
7.63.3.3	clientGone()	214
7.63.3.4	closeConnections()	214
7.63.3.5	function()	214

7.63.3.6	<code>getConnectionsNumber()</code>	215
7.63.3.7	<code>getEventSocket()</code> [1/2]	215
7.63.3.8	<code>getEventSocket()</code> [2/2]	215
7.63.3.9	<code>getHostName()</code> [1/2]	215
7.63.3.10	<code>getHostName()</code> [2/2]	215
7.63.3.11	<code>getSocket()</code> [1/2]	215
7.63.3.12	<code>getSocket()</code> [2/2]	215
7.63.3.13	<code>init()</code> [1/2]	215
7.63.3.14	<code>init()</code> [2/2]	216
7.63.3.15	<code>newConnection()</code>	216
7.63.3.16	<code>process()</code> [1/2]	216
7.63.3.17	<code>process()</code> [2/2]	216
7.63.3.18	<code>processEvent()</code> [1/4]	216
7.63.3.19	<code>processEvent()</code> [2/4]	216
7.63.3.20	<code>processEvent()</code> [3/4]	217
7.63.3.21	<code>processEvent()</code> [4/4]	217
7.63.3.22	<code>run()</code> [1/2]	217
7.63.3.23	<code>run()</code> [2/2]	217
7.63.3.24	<code>sendEvent()</code> [1/2]	217
7.63.3.25	<code>sendEvent()</code> [2/2]	217
7.63.4	Member Data Documentation	217
7.63.4.1	<code>oboapNs</code>	218
7.63.4.2	<code>oboapns</code>	218
7.63.4.3	<code>oclientGoneEvent</code>	218
7.63.4.4	<code>oclients</code>	218
7.63.4.5	<code>ohostName</code>	218
7.63.4.6	<code>oisBoapns</code>	218
7.63.4.7	<code>oclock</code>	218
7.63.4.8	<code>onet</code>	218
7.63.4.9	<code>onetEvent</code>	219



7.63.4.10 onetEventAddress . . . . .	219
7.63.4.11 onumOperations . . . . .	219
7.63.4.12 opoll . . . . .	219
7.63.4.13 orx . . . . .	219
7.63.4.14 oservices . . . . .	219
7.63.4.15 othreaded . . . . .	219
7.63.4.16 otx . . . . .	220
7.64 BoapServerConnection Class Reference . . . . .	220
7.64.1 Constructor & Destructor Documentation . . . . .	221
7.64.1.1 BoapServerConnection() . . . . .	221
7.64.1.2 ~BoapServerConnection() . . . . .	221
7.64.2 Member Function Documentation . . . . .	221
7.64.2.1 function() . . . . .	221
7.64.2.2 getHead() . . . . .	221
7.64.2.3 getSocket() . . . . .	221
7.64.2.4 init() . . . . .	221
7.64.2.5 process() . . . . .	222
7.64.2.6 setMaxLength() . . . . .	222
7.64.2.7 validate() . . . . .	222
7.64.3 Member Data Documentation . . . . .	222
7.64.3.1 oboapServer . . . . .	222
7.64.3.2 omaxLength . . . . .	222
7.64.3.3 orx . . . . .	222
7.64.3.4 osocket . . . . .	222
7.64.3.5 otx . . . . .	223
7.65 BoapServiceEntry Class Reference . . . . .	223
7.65.1 Constructor & Destructor Documentation . . . . .	223
7.65.1.1 BoapServiceEntry() [1/2] . . . . .	223
7.65.1.2 BoapServiceEntry() [2/2] . . . . .	223
7.65.2 Member Data Documentation . . . . .	223

---

7.65.2.1	oobject	224
7.65.2.2	oservice	224
7.66	BoapServiceObject Class Reference	224
7.66.1	Constructor & Destructor Documentation	225
7.66.1.1	BoapServiceObject() [1/2]	225
7.66.1.2	~BoapServiceObject() [1/2]	225
7.66.1.3	BoapServiceObject() [2/2]	225
7.66.1.4	~BoapServiceObject() [2/2]	225
7.66.2	Member Function Documentation	225
7.66.2.1	doConnectionPriority()	225
7.66.2.2	doPing()	226
7.66.2.3	name() [1/2]	226
7.66.2.4	name() [2/2]	226
7.66.2.5	process() [1/2]	226
7.66.2.6	process() [2/2]	226
7.66.2.7	processEvent() [1/4]	226
7.66.2.8	processEvent() [2/4]	227
7.66.2.9	processEvent() [3/4]	227
7.66.2.10	processEvent() [4/4]	227
7.66.2.11	sendEvent() [1/4]	227
7.66.2.12	sendEvent() [2/4]	227
7.66.2.13	sendEvent() [3/4]	227
7.66.2.14	sendEvent() [4/4]	228
7.66.2.15	setName()	228
7.66.3	Member Data Documentation	228
7.66.3.1	oapiVersion	228
7.66.3.2	ofuncList	228
7.66.3.3	oname	228
7.66.3.4	oserver	228
7.67	BoapSignalObject Class Reference	229

---

7.67.1	Constructor & Destructor Documentation	229
7.67.1.1	BoapSignalObject() [1/2]	229
7.67.1.2	BoapSignalObject() [2/2]	229
7.67.2	Member Function Documentation	230
7.67.2.1	performSend() [1/2]	230
7.67.2.2	performSend() [2/2]	230
7.67.3	Member Data Documentation	230
7.67.3.1	orx	230
7.67.3.2	otx	230
7.68	BObj Class Reference	230
7.68.1	Constructor & Destructor Documentation	231
7.68.1.1	BObj()	231
7.68.1.2	~BObj()	231
7.68.2	Member Function Documentation	231
7.68.2.1	getDebugString()	231
7.68.2.2	getMember()	231
7.68.2.3	getMembers() [1/2]	232
7.68.2.4	getMembers() [2/2]	232
7.68.2.5	getType()	232
7.68.2.6	membersPrint()	232
7.68.2.7	setMember()	232
7.68.2.8	setMembers()	232
7.69	BObjMember Struct Reference	233
7.69.1	Member Data Documentation	233
7.69.1.1	dataOffset	233
7.69.1.2	name	233
7.69.1.3	size	233
7.69.1.4	type	233
7.69.1.5	typeComp	233
7.69.1.6	typeName	234

7.70 BPoll Class Reference . . . . .	234
7.70.1 Detailed Description . . . . .	235
7.70.2 Member Typedef Documentation . . . . .	235
7.70.2.1 PollFd . . . . .	235
7.70.3 Constructor & Destructor Documentation . . . . .	235
7.70.3.1 BPoll() . . . . .	235
7.70.3.2 ~BPoll() . . . . .	235
7.70.4 Member Function Documentation . . . . .	235
7.70.4.1 append() . . . . .	235
7.70.4.2 clear() . . . . .	235
7.70.4.3 delFd() . . . . .	236
7.70.4.4 doPoll() . . . . .	236
7.70.4.5 doPollEvents() . . . . .	236
7.70.4.6 getPollFds() . . . . .	236
7.70.4.7 getPollFdsNum() . . . . .	236
7.70.4.8 nextFd() . . . . .	236
7.70.5 Member Data Documentation . . . . .	237
7.70.5.1 ofds . . . . .	237
7.70.5.2 ofdsNext . . . . .	237
7.70.5.3 ofdsNum . . . . .	237
7.71 BQueue< T > Class Template Reference . . . . .	237
7.71.1 Detailed Description . . . . .	238
7.71.2 Constructor & Destructor Documentation . . . . .	238
7.71.2.1 BQueue() . . . . .	238
7.71.2.2 ~BQueue() . . . . .	238
7.71.3 Member Function Documentation . . . . .	239
7.71.3.1 clear() . . . . .	239
7.71.3.2 read() . . . . .	239
7.71.3.3 readAvailable() . . . . .	239
7.71.3.4 write() . . . . .	239

7.71.3.5	<a href="#">writeAvailable()</a>	239
7.71.4	<a href="#">Member Data Documentation</a>	240
7.71.4.1	<a href="#">oLock</a>	240
7.71.4.2	<a href="#">oNumber</a>	240
7.71.4.3	<a href="#">oSize</a>	240
7.72	<a href="#">BRefData Class Reference</a>	240
7.72.1	<a href="#">Detailed Description</a>	241
7.72.2	<a href="#">Constructor &amp; Destructor Documentation</a>	241
7.72.2.1	<a href="#">BRefData() [1/3]</a>	241
7.72.2.2	<a href="#">BRefData() [2/3]</a>	241
7.72.2.3	<a href="#">BRefData() [3/3]</a>	241
7.72.2.4	<a href="#">~BRefData()</a>	241
7.72.3	<a href="#">Member Function Documentation</a>	241
7.72.3.1	<a href="#">addRef()</a>	242
7.72.3.2	<a href="#">copy()</a>	242
7.72.3.3	<a href="#">data()</a>	242
7.72.3.4	<a href="#">deleteRef()</a>	242
7.72.3.5	<a href="#">len()</a>	242
7.72.3.6	<a href="#">operator=()</a>	242
7.72.3.7	<a href="#">setLen()</a>	243
7.72.4	<a href="#">Member Data Documentation</a>	243
7.72.4.1	<a href="#">oData</a>	243
7.72.4.2	<a href="#">oLen</a>	243
7.72.4.3	<a href="#">oRefCount</a>	243
7.73	<a href="#">BRtc Class Reference</a>	243
7.73.1	<a href="#">Detailed Description</a>	244
7.73.2	<a href="#">Constructor &amp; Destructor Documentation</a>	244
7.73.2.1	<a href="#">BRtc()</a>	244
7.73.2.2	<a href="#">~BRtc()</a>	244
7.73.3	<a href="#">Member Function Documentation</a>	244

7.73.3.1	init()	244
7.73.3.2	wait()	245
7.73.4	Member Data Documentation	245
7.73.4.1	ofd	245
7.73.4.2	orate	245
7.74	BRtcThreaded Class Reference	245
7.74.1	Detailed Description	246
7.74.2	Constructor & Destructor Documentation	246
7.74.2.1	BRtcThreaded()	246
7.74.2.2	~BRtcThreaded()	246
7.74.3	Member Function Documentation	246
7.74.3.1	function()	246
7.74.3.2	init()	247
7.74.3.3	wait()	247
7.74.4	Member Data Documentation	247
7.74.4.1	ocond	247
7.74.4.2	orate	247
7.74.4.3	ortc	247
7.75	BRWLock Class Reference	247
7.75.1	Detailed Description	248
7.75.2	Constructor & Destructor Documentation	248
7.75.2.1	BRWLock() [1/2]	248
7.75.2.2	BRWLock() [2/2]	248
7.75.2.3	~BRWLock()	248
7.75.3	Member Function Documentation	249
7.75.3.1	operator=()	249
7.75.3.2	rdLock()	249
7.75.3.3	tryRdLock()	249
7.75.3.4	tryWrLock()	249
7.75.3.5	unlock()	249

---

7.75.3.6	wrLock()	249
7.75.4	Member Data Documentation	250
7.75.4.1	olock	250
7.76	BSema Class Reference	250
7.76.1	Detailed Description	250
7.76.2	Constructor & Destructor Documentation	250
7.76.2.1	BSema() [1/2]	251
7.76.2.2	BSema() [2/2]	251
7.76.2.3	~BSema()	251
7.76.3	Member Function Documentation	251
7.76.3.1	getValue()	251
7.76.3.2	operator=()	251
7.76.3.3	post()	251
7.76.3.4	timedWait()	252
7.76.3.5	tryWait()	252
7.76.3.6	wait()	252
7.76.4	Member Data Documentation	252
7.76.4.1	osema	252
7.77	BSemaphore Class Reference	252
7.77.1	Detailed Description	253
7.77.2	Constructor & Destructor Documentation	253
7.77.2.1	BSemaphore() [1/2]	253
7.77.2.2	BSemaphore() [2/2]	253
7.77.2.3	~BSemaphore()	253
7.77.3	Member Function Documentation	253
7.77.3.1	getValue()	254
7.77.3.2	operator=()	254
7.77.3.3	set()	254
7.77.3.4	wait()	254
7.77.4	Member Data Documentation	254

---

7.77.4.1	osema	254
7.78	BSemaphoreBool Class Reference	254
7.78.1	Constructor & Destructor Documentation	255
7.78.1.1	BSemaphoreBool() [1/2]	255
7.78.1.2	BSemaphoreBool() [2/2]	255
7.78.1.3	~BSemaphoreBool()	255
7.78.2	Member Function Documentation	255
7.78.2.1	clear()	256
7.78.2.2	operator int()	256
7.78.2.3	operator=()	256
7.78.2.4	operator==()	256
7.78.2.5	set()	256
7.78.2.6	value()	256
7.78.2.7	wait()	256
7.78.3	Member Data Documentation	257
7.78.3.1	osema	257
7.78.3.2	ovalue	257
7.79	BSemaphoreCount Class Reference	257
7.79.1	Constructor & Destructor Documentation	258
7.79.1.1	BSemaphoreCount() [1/2]	258
7.79.1.2	BSemaphoreCount() [2/2]	258
7.79.1.3	~BSemaphoreCount()	258
7.79.2	Member Function Documentation	258
7.79.2.1	add()	258
7.79.2.2	operator=()	258
7.79.2.3	setValue()	258
7.79.2.4	take()	259
7.79.2.5	value()	259
7.79.2.6	wait()	259
7.79.3	Member Data Documentation	259



---

7.79.3.1	o_lock	259
7.79.3.2	o_semaphore	259
7.79.3.3	o_value	259
7.80	BSocket Class Reference	260
7.80.1	Member Enumeration Documentation	261
7.80.1.1	NType	261
7.80.1.2	Priority	261
7.80.2	Constructor & Destructor Documentation	261
7.80.2.1	BSocket() [1/4]	261
7.80.2.2	BSocket() [2/4]	261
7.80.2.3	BSocket() [3/4]	262
7.80.2.4	BSocket() [4/4]	262
7.80.2.5	~BSocket()	262
7.80.3	Member Function Documentation	262
7.80.3.1	accept() [1/2]	262
7.80.3.2	accept() [2/2]	262
7.80.3.3	bind()	262
7.80.3.4	close()	263
7.80.3.5	connect()	263
7.80.3.6	getAddress()	263
7.80.3.7	getFd()	263
7.80.3.8	getMTU()	263
7.80.3.9	getSockOpt()	263
7.80.3.10	init() [1/2]	263
7.80.3.11	init() [2/2]	264
7.80.3.12	listen()	264
7.80.3.13	recv()	264
7.80.3.14	recvAvailable()	264
7.80.3.15	recvFrom()	264
7.80.3.16	recvFromWithTimeout()	264

7.80.3.17	recvWithTimeout()	265
7.80.3.18	send()	265
7.80.3.19	sendChunks()	265
7.80.3.20	sendTo()	265
7.80.3.21	setBroadCast()	265
7.80.3.22	setFd()	266
7.80.3.23	setPriority()	266
7.80.3.24	setReuseAddress()	266
7.80.3.25	setSockOpt()	266
7.80.3.26	shutdown()	266
7.80.4	Member Data Documentation	266
7.80.4.1	osocket	266
7.81	BSocketAddress Class Reference	267
7.81.1	Detailed Description	267
7.81.2	Member Typedef Documentation	267
7.81.2.1	SockAddr	268
7.81.3	Constructor & Destructor Documentation	268
7.81.3.1	BSocketAddress() [1/3]	268
7.81.3.2	BSocketAddress() [2/3]	268
7.81.3.3	BSocketAddress() [3/3]	268
7.81.3.4	~BSocketAddress()	268
7.81.4	Member Function Documentation	268
7.81.4.1	getString()	268
7.81.4.2	len()	269
7.81.4.3	operator const SockAddr *()	269
7.81.4.4	operator"!=(	269
7.81.4.5	operator=(	269
7.81.4.6	operator==(	269
7.81.4.7	raw()	269
7.81.4.8	set()	269

---

7.81.5	Member Data Documentation	270
7.81.5.1	oaddress	270
7.81.5.2	olen	270
7.82	BSocketAddressINET Class Reference	270
7.82.1	Detailed Description	271
7.82.2	Member Typedef Documentation	271
7.82.2.1	SockAddrIP	271
7.82.3	Member Function Documentation	271
7.82.3.1	address()	271
7.82.3.2	getHostName()	271
7.82.3.3	getIpAddresses()	272
7.82.3.4	getIpAddressList()	272
7.82.3.5	getIpAddressListAll()	272
7.82.3.6	getString()	272
7.82.3.7	port()	272
7.82.3.8	set() [1/3]	272
7.82.3.9	set() [2/3]	273
7.82.3.10	set() [3/3]	273
7.82.3.11	setPort()	273
7.83	BSpi Class Reference	273
7.83.1	Detailed Description	274
7.83.2	Member Enumeration Documentation	274
7.83.2.1	Mode	274
7.83.3	Constructor & Destructor Documentation	274
7.83.3.1	BSpi()	274
7.83.4	Member Function Documentation	274
7.83.4.1	init()	274
7.83.4.2	transact()	275
7.83.5	Member Data Documentation	275
7.83.5.1	odev	275

---

7.83.5.2	odevName	275
7.84	BString Class Reference	275
7.84.1	Constructor & Destructor Documentation	278
7.84.1.1	BString() [1/9]	279
7.84.1.2	BString() [2/9]	279
7.84.1.3	BString() [3/9]	279
7.84.1.4	BString() [4/9]	279
7.84.1.5	BString() [5/9]	279
7.84.1.6	BString() [6/9]	279
7.84.1.7	BString() [7/9]	279
7.84.1.8	BString() [8/9]	280
7.84.1.9	BString() [9/9]	280
7.84.1.10	~BString()	280
7.84.2	Member Function Documentation	280
7.84.2.1	add()	280
7.84.2.2	append()	280
7.84.2.3	base64Decode()	280
7.84.2.4	base64Encode()	281
7.84.2.5	basename()	281
7.84.2.6	clear()	281
7.84.2.7	compare()	281
7.84.2.8	compareRegex()	281
7.84.2.9	compareWild()	281
7.84.2.10	compareWildExpression()	282
7.84.2.11	convert() [1/5]	282
7.84.2.12	convert() [2/5]	282
7.84.2.13	convert() [3/5]	282
7.84.2.14	convert() [4/5]	282
7.84.2.15	convert() [5/5]	283
7.84.2.16	convertHex() [1/2]	283

---

7.84.2.17 <code>convertHex()</code> [2/2]	283
7.84.2.18 <code>copy()</code>	283
7.84.2.19 <code>csvDecode()</code>	283
7.84.2.20 <code>csvEncode()</code>	283
7.84.2.21 <code>del()</code>	284
7.84.2.22 <code>dirname()</code>	284
7.84.2.23 <code>extension()</code>	284
7.84.2.24 <code>field()</code>	284
7.84.2.25 <code>fields()</code>	284
7.84.2.26 <code>find()</code> [1/2]	284
7.84.2.27 <code>find()</code> [2/2]	285
7.84.2.28 <code>findReverse()</code>	285
7.84.2.29 <code>firstLine()</code>	285
7.84.2.30 <code>fixedLen()</code>	285
7.84.2.31 <code>get()</code> [1/2]	285
7.84.2.32 <code>get()</code> [2/2]	285
7.84.2.33 <code>getTokenList()</code> [1/2]	286
7.84.2.34 <code>getTokenList()</code> [2/2]	286
7.84.2.35 <code>hash()</code>	286
7.84.2.36 <code>init()</code>	286
7.84.2.37 <code>insert()</code>	286
7.84.2.38 <code>inString()</code>	286
7.84.2.39 <code>isSpace()</code>	287
7.84.2.40 <code>justify()</code>	287
7.84.2.41 <code>len()</code>	287
7.84.2.42 <code>lowerFirst()</code>	287
7.84.2.43 <code>operator const char *()</code>	287
7.84.2.44 <code>operator"!=""</code> [1/2]	287
7.84.2.45 <code>operator"!=""</code> [2/2]	288
7.84.2.46 <code>operator+()</code> [1/6]	288

---

7.84.2.47 operator+() [2/6]	288
7.84.2.48 operator+() [3/6]	288
7.84.2.49 operator+() [4/6]	288
7.84.2.50 operator+() [5/6]	288
7.84.2.51 operator+() [6/6]	288
7.84.2.52 operator+=() [1/2]	289
7.84.2.53 operator+=() [2/2]	289
7.84.2.54 operator<() [1/2]	289
7.84.2.55 operator<() [2/2]	289
7.84.2.56 operator<=()	289
7.84.2.57 operator=()	289
7.84.2.58 operator==() [1/2]	289
7.84.2.59 operator==() [2/2]	290
7.84.2.60 operator>() [1/2]	290
7.84.2.61 operator>() [2/2]	290
7.84.2.62 operator>=()	290
7.84.2.63 operator[]()	290
7.84.2.64 pad()	290
7.84.2.65 printf()	291
7.84.2.66 pullLine()	291
7.84.2.67 pullSeparators()	291
7.84.2.68 pullToken()	291
7.84.2.69 pullWord()	291
7.84.2.70 removeNL()	291
7.84.2.71 removeSeparators()	292
7.84.2.72 retDouble()	292
7.84.2.73 retFloat64()	292
7.84.2.74 retInt()	292
7.84.2.75 retStr()	292
7.84.2.76 retStrDup()	292

7.84.2.77	retUInt()	293
7.84.2.78	reverse()	293
7.84.2.79	split()	293
7.84.2.80	subString()	293
7.84.2.81	toLowerCase()	293
7.84.2.82	toUpperCase()	293
7.84.2.83	translateChar()	294
7.84.2.84	truncate()	294
7.84.3	Member Data Documentation	294
7.84.3.1	ostr	294
7.85	BStringLocked Class Reference	294
7.85.1	Constructor & Destructor Documentation	295
7.85.1.1	BStringLocked() [1/3]	295
7.85.1.2	BStringLocked() [2/3]	295
7.85.1.3	BStringLocked() [3/3]	295
7.85.2	Member Function Documentation	295
7.85.2.1	len()	295
7.85.2.2	operator BString()	295
7.85.2.3	operator+()	296
7.85.2.4	operator=()	296
7.85.3	Member Data Documentation	296
7.85.3.1	olock	296
7.85.3.2	ostr	296
7.86	BStringMutex Class Reference	296
7.86.1	Constructor & Destructor Documentation	297
7.86.1.1	BStringMutex()	297
7.87	BTable Class Reference	297
7.87.1	Constructor & Destructor Documentation	297
7.87.1.1	BTable()	298
7.87.1.2	~BTable()	298

7.87.2	Member Function Documentation	298
7.87.2.1	addRow()	298
7.87.2.2	calculateWidths()	298
7.87.2.3	clear()	298
7.87.2.4	print()	298
7.87.2.5	printLine()	298
7.87.2.6	setTitle()	299
7.87.3	Member Data Documentation	299
7.87.3.1	ocolumnWidths	299
7.87.3.2	odata	299
7.87.3.3	otitle	299
7.88	BTask Class Reference	299
7.88.1	Constructor & Destructor Documentation	300
7.88.1.1	BTask()	300
7.88.1.2	~BTask()	300
7.88.2	Member Function Documentation	300
7.88.2.1	init()	300
7.88.2.2	run()	300
7.88.2.3	setPriority()	301
7.88.2.4	start()	301
7.88.2.5	stop()	301
7.88.2.6	taskFunc()	301
7.88.2.7	waitForCompletion()	301
7.88.3	Member Data Documentation	301
7.88.3.1	oname	301
7.88.3.2	opolicy	302
7.88.3.3	opriority	302
7.88.3.4	orunning	302
7.88.3.5	ostackSize	302
7.88.3.6	othread	302



---

7.89 BThread Class Reference . . . . .	302
7.89.1 Constructor & Destructor Documentation . . . . .	303
7.89.1.1 BThread() . . . . .	303
7.89.1.2 ~BThread() . . . . .	303
7.89.2 Member Function Documentation . . . . .	303
7.89.2.1 cancel() . . . . .	304
7.89.2.2 function() . . . . .	304
7.89.2.3 getThread() . . . . .	304
7.89.2.4 result() . . . . .	304
7.89.2.5 running() . . . . .	304
7.89.2.6 setInitPriority() . . . . .	304
7.89.2.7 setInitStackSize() . . . . .	304
7.89.2.8 setPriority() . . . . .	305
7.89.2.9 start() . . . . .	305
7.89.2.10 startFunc() . . . . .	305
7.89.2.11 waitForCompletion() . . . . .	305
7.89.3 Member Data Documentation . . . . .	305
7.89.3.1 opolicy . . . . .	305
7.89.3.2 opriority . . . . .	305
7.89.3.3 oresult . . . . .	305
7.89.3.4 orunning . . . . .	306
7.89.3.5 ostackSize . . . . .	306
7.89.3.6 othread . . . . .	306
7.90 BTime Class Reference . . . . .	306
7.90.1 Constructor & Destructor Documentation . . . . .	307
7.90.1.1 BTime() . . . . .	307
7.90.2 Member Function Documentation . . . . .	307
7.90.2.1 addSeconds() . . . . .	307
7.90.2.2 getDate() . . . . .	307
7.90.2.3 getSeconds() . . . . .	308

---

7.90.2.4	getString()	308
7.90.2.5	getTime()	308
7.90.2.6	isLeapYear()	308
7.90.2.7	isSet()	308
7.90.2.8	operator!=(())	308
7.90.2.9	operator+()	309
7.90.2.10	operator+=()	309
7.90.2.11	operator<()	309
7.90.2.12	operator<=()	309
7.90.2.13	operator==(())	309
7.90.2.14	operator>()	309
7.90.2.15	operator>=()	309
7.90.2.16	set() [1/2]	310
7.90.2.17	set() [2/2]	310
7.90.2.18	setString()	310
7.90.2.19	setYearDay()	310
7.90.3	Member Data Documentation	310
7.90.3.1	otime	311
7.91	BTimer Class Reference	311
7.91.1	Detailed Description	312
7.91.2	Constructor & Destructor Documentation	312
7.91.2.1	BTimer()	312
7.91.2.2	~BTimer()	312
7.91.3	Member Function Documentation	312
7.91.3.1	add()	312
7.91.3.2	average()	312
7.91.3.3	clear()	312
7.91.3.4	getElapsedTime()	313
7.91.3.5	getTime()	313
7.91.3.6	peak()	313

---

7.91.3.7	start()	313
7.91.3.8	stop()	313
7.91.4	Member Data Documentation	313
7.91.4.1	oaverage	313
7.91.4.2	oendTime	314
7.91.4.3	olock	314
7.91.4.4	onum	314
7.91.4.5	opeak	314
7.91.4.6	ostartTime	314
7.92	BTimeStamp Class Reference	314
7.92.1	Constructor & Destructor Documentation	316
7.92.1.1	BTimeStamp() [1/3]	316
7.92.1.2	BTimeStamp() [2/3]	316
7.92.1.3	BTimeStamp() [3/3]	317
7.92.1.4	~BTimeStamp()	317
7.92.2	Member Function Documentation	317
7.92.2.1	addMicroSeconds()	317
7.92.2.2	addMilliSeconds()	317
7.92.2.3	addSeconds()	317
7.92.2.4	clear()	317
7.92.2.5	compare()	318
7.92.2.6	day()	318
7.92.2.7	difference()	318
7.92.2.8	getDate()	318
7.92.2.9	getString()	318
7.92.2.10	getStringFormatted()	318
7.92.2.11	getStringNoMs()	319
7.92.2.12	getYearMicroSeconds()	319
7.92.2.13	getYearSeconds()	319
7.92.2.14	hour()	319

7.92.2.15 isLeap()	319
7.92.2.16 isSet()	319
7.92.2.17 microSecond()	319
7.92.2.18 minute()	320
7.92.2.19 month()	320
7.92.2.20 operator BString()	320
7.92.2.21 operator"!=(())	320
7.92.2.22 operator<()	320
7.92.2.23 operator<=()	320
7.92.2.24 operator=()	320
7.92.2.25 operator==(())	321
7.92.2.26 operator>()	321
7.92.2.27 operator>=()	321
7.92.2.28 second()	321
7.92.2.29 set() [1/3]	321
7.92.2.30 set() [2/3]	321
7.92.2.31 set() [3/3]	322
7.92.2.32 setFirst()	322
7.92.2.33 setLast()	322
7.92.2.34 setNow()	322
7.92.2.35 setString()	322
7.92.2.36 setTime()	322
7.92.2.37 setYDay()	323
7.92.2.38 yday()	323
7.92.2.39 year()	323
7.92.3 Member Data Documentation	323
7.92.3.1 ohour	323
7.92.3.2 omicroSecond	323
7.92.3.3 ominute	323
7.92.3.4 osecond	324

---

7.92.3.5	ospare	324
7.92.3.6	oyday	324
7.92.3.7	oyear	324
7.93	BTimeStampMs Class Reference	324
7.93.1	Constructor & Destructor Documentation	326
7.93.1.1	BTimeStampMs()	326
7.93.1.2	~BTimeStampMs()	326
7.93.2	Member Function Documentation	326
7.93.2.1	addMilliseconds()	326
7.93.2.2	addSeconds()	327
7.93.2.3	clear()	327
7.93.2.4	compare()	327
7.93.2.5	difference()	327
7.93.2.6	getDate()	327
7.93.2.7	getDurationString()	327
7.93.2.8	getDurationStringNoMs()	328
7.93.2.9	getString()	328
7.93.2.10	getStringNoMs()	328
7.93.2.11	getStringRaw()	328
7.93.2.12	getYearMilliseconds()	328
7.93.2.13	getYearSeconds()	328
7.93.2.14	isLeap()	329
7.93.2.15	operator<()	329
7.93.2.16	operator<=()	329
7.93.2.17	operator>()	329
7.93.2.18	operator>=()	329
7.93.2.19	setDurationString()	329
7.93.2.20	setNow()	330
7.93.2.21	setString()	330
7.93.2.22	subMilliseconds()	330

---

7.93.2.23	subSeconds()	330
7.93.3	Member Data Documentation	330
7.93.3.1	hour	330
7.93.3.2	milliSecond	331
7.93.3.3	minute	331
7.93.3.4	sampleNumber	331
7.93.3.5	second	331
7.93.3.6	yday	331
7.93.3.7	year	331
7.94	BTimeUs Class Reference	332
7.94.1	Constructor & Destructor Documentation	333
7.94.1.1	BTimeUs() [1/2]	333
7.94.1.2	BTimeUs() [2/2]	333
7.94.2	Member Function Documentation	333
7.94.2.1	addMicroSeconds()	333
7.94.2.2	addSeconds()	333
7.94.2.3	getDate()	334
7.94.2.4	getMicroSeconds()	334
7.94.2.5	getSeconds()	334
7.94.2.6	getString()	334
7.94.2.7	getStringUs()	334
7.94.2.8	getTime()	335
7.94.2.9	isLeapYear()	335
7.94.2.10	isSet()	335
7.94.2.11	operator BTime()	335
7.94.2.12	operator"!=(())	335
7.94.2.13	operator+()	335
7.94.2.14	operator+="()	336
7.94.2.15	operator<()	336
7.94.2.16	operator<=()	336

---

7.94.2.17	<code>operator==( )</code>	336
7.94.2.18	<code>operator&gt;( )</code>	336
7.94.2.19	<code>operator&gt;=( )</code>	336
7.94.2.20	<code>set( )</code> [1/2]	336
7.94.2.21	<code>set( )</code> [2/2]	337
7.94.2.22	<code>setString( )</code>	337
7.94.2.23	<code>setYearDay( )</code>	337
7.94.3	Member Data Documentation	337
7.94.3.1	<code>otime</code>	337
7.95	BUrl Class Reference	338
7.95.1	Detailed Description	338
7.95.2	Constructor & Destructor Documentation	338
7.95.2.1	<code>BUrl( )</code>	338
7.95.2.2	<code>~BUrl( )</code>	338
7.95.3	Member Function Documentation	339
7.95.3.1	<code>readString( )</code>	339
7.95.3.2	<code>writeData( )</code>	339
7.95.4	Member Data Documentation	339
7.95.4.1	<code>oinit</code>	339
7.95.4.2	<code>ores</code>	339
7.96	BList< T >::Node Class Reference	340
7.96.1	Constructor & Destructor Documentation	340
7.96.1.1	<code>Node( )</code>	340
7.96.2	Member Data Documentation	340
7.96.2.1	<code>item</code>	340

---

<b>8 File Documentation</b>	<b>341</b>
8.1 BArray.h File Reference	341
8.1.1 Macro Definition Documentation	341
8.1.1.1 BArrayLoop	341
8.2 BAtomic.h File Reference	341
8.2.1 Typedef Documentation	342
8.2.1.1 BAtomicInt32	342
8.2.1.2 BAtomicInt64	342
8.2.1.3 BAtomicUInt32	342
8.2.1.4 BAtomicUInt64	342
8.3 BAtomicCount.h File Reference	342
8.4 BBuffer.cpp File Reference	343
8.4.1 Variable Documentation	343
8.4.1.1 roundSize	343
8.5 BBuffer.h File Reference	343
8.5.1 Macro Definition Documentation	344
8.5.1.1 BBigEndian	344
8.6 BComms.cpp File Reference	344
8.7 BComms.h File Reference	344
8.8 BComplex.h File Reference	344
8.8.1 Typedef Documentation	344
8.8.1.1 BComplex	345
8.8.1.2 BComplex32	345
8.8.1.3 BComplex64	345
8.9 BCond.cpp File Reference	345
8.10 BCond.h File Reference	345
8.11 BCondInt.cpp File Reference	345
8.11.1 Function Documentation	346
8.11.1.1 getTimeout()	346
8.12 BCondInt.h File Reference	346



8.13 BConfig.cpp File Reference . . . . .	346
8.14 BConfig.h File Reference . . . . .	346
8.15 BCrc16.cpp File Reference . . . . .	347
8.15.1 Function Documentation . . . . .	347
8.15.1.1 bcrc16() . . . . .	347
8.15.2 Variable Documentation . . . . .	347
8.15.2.1 table_crc_hi . . . . .	347
8.15.2.2 table_crc_lo . . . . .	348
8.16 BCrc16.h File Reference . . . . .	348
8.16.1 Function Documentation . . . . .	348
8.16.1.1 bcrc16() . . . . .	348
8.17 BCrc32.cpp File Reference . . . . .	348
8.17.1 Function Documentation . . . . .	349
8.17.1.1 bcrc32() . . . . .	349
8.17.2 Variable Documentation . . . . .	349
8.17.2.1 crc32_tab . . . . .	349
8.18 BCrc32.h File Reference . . . . .	349
8.18.1 Function Documentation . . . . .	349
8.18.1.1 bcrc32() . . . . .	350
8.19 BDate-1.cpp File Reference . . . . .	350
8.19.1 Function Documentation . . . . .	350
8.19.1.1 fromBString() . . . . .	350
8.19.1.2 toBString() . . . . .	350
8.19.2 Variable Documentation . . . . .	350
8.19.2.1 mon_yday . . . . .	351
8.20 BDate.cpp File Reference . . . . .	351
8.20.1 Function Documentation . . . . .	351
8.20.1.1 fromBString() . . . . .	351
8.20.1.2 toBString() . . . . .	351
8.20.2 Variable Documentation . . . . .	352

---

8.20.2.1	mon_yday	352
8.21	BDate.h File Reference	352
8.21.1	Function Documentation	352
8.21.1.1	fromBString()	352
8.21.1.2	toBString()	353
8.22	BDebug.cpp File Reference	353
8.22.1	Function Documentation	353
8.22.1.1	bhd32()	353
8.22.1.2	bhd8()	354
8.22.1.3	bhd8a()	354
8.22.1.4	bhda32()	354
8.22.1.5	bhda8()	354
8.22.1.6	getTime()	354
8.22.1.7	setDebug()	354
8.22.2	Variable Documentation	354
8.22.2.1	bdebug	355
8.23	BDebug.h File Reference	355
8.23.1	Macro Definition Documentation	356
8.23.1.1	BDebug_STD	356
8.23.1.2	dl1printf	356
8.23.1.3	dl2printf	356
8.23.1.4	dl3printf	356
8.23.1.5	dl4printf	356
8.23.1.6	dprintf	356
8.23.1.7	eprintf	357
8.23.1.8	nprintf	357
8.23.1.9	wprintf	357
8.23.2	Function Documentation	357
8.23.2.1	bhd32()	357
8.23.2.2	bhd8()	357

8.23.2.3	bhd8a()	358
8.23.2.4	bhda8()	358
8.23.2.5	bhds32()	358
8.23.2.6	gettid()	358
8.23.2.7	getTime()	358
8.23.2.8	setDebug()	358
8.23.2.9	tprintf()	358
8.23.3	Variable Documentation	359
8.23.3.1	bdebug	359
8.24	BDict.cpp File Reference	359
8.24.1	Function Documentation	359
8.24.1.1	bdictStringToString()	359
8.24.1.2	fromBString()	359
8.24.1.3	toBString()	359
8.25	BDict.h File Reference	360
8.25.1	Typedef Documentation	360
8.25.1.1	BDictString	360
8.25.2	Function Documentation	360
8.25.2.1	bdictStringToString()	360
8.25.2.2	fromBString()	361
8.25.2.3	toBString()	361
8.26	BDictMap.h File Reference	361
8.26.1	Typedef Documentation	361
8.26.1.1	BDictMapString	361
8.27	BDir.cpp File Reference	361
8.27.1	Function Documentation	362
8.27.1.1	wild()	362
8.27.2	Variable Documentation	362
8.27.2.1	wildString	362
8.28	BDir.h File Reference	362

8.29	BDuration.cpp File Reference	362
8.30	BDuration.h File Reference	363
8.31	BEndian.cpp File Reference	363
8.31.1	Function Documentation	363
8.31.1.1	bswap_copy()	363
8.32	BEndian.h File Reference	363
8.32.1	Macro Definition Documentation	365
8.32.1.1	be16toh	365
8.32.1.2	be32toh	365
8.32.1.3	be64toh	365
8.32.1.4	htobe16	365
8.32.1.5	htobe32	365
8.32.1.6	htobe64	365
8.32.1.7	htole16	366
8.32.1.8	htole32	366
8.32.1.9	htole64	366
8.32.1.10	le16toh	366
8.32.1.11	le32toh	366
8.32.1.12	le64toh	366
8.32.2	Function Documentation	366
8.32.2.1	betoh() [1/8]	367
8.32.2.2	betoh() [2/8]	367
8.32.2.3	betoh() [3/8]	367
8.32.2.4	betoh() [4/8]	367
8.32.2.5	betoh() [5/8]	367
8.32.2.6	betoh() [6/8]	367
8.32.2.7	betoh() [7/8]	367
8.32.2.8	betoh() [8/8]	368
8.32.2.9	bswap_copy()	368
8.32.2.10	bswap_p16()	368

---

8.32.2.11 bswap_p32()	368
8.32.2.12 bswap_p64()	368
8.32.2.13 bswap_p8()	368
8.32.2.14 htobe() [1/8]	369
8.32.2.15 htobe() [2/8]	369
8.32.2.16 htobe() [3/8]	369
8.32.2.17 htobe() [4/8]	369
8.32.2.18 htobe() [5/8]	369
8.32.2.19 htobe() [6/8]	369
8.32.2.20 htobe() [7/8]	369
8.32.2.21 htobe() [8/8]	370
8.32.2.22 htole() [1/8]	370
8.32.2.23 htole() [2/8]	370
8.32.2.24 htole() [3/8]	370
8.32.2.25 htole() [4/8]	370
8.32.2.26 htole() [5/8]	370
8.32.2.27 htole() [6/8]	370
8.32.2.28 htole() [7/8]	371
8.32.2.29 htole() [8/8]	371
8.32.2.30 letoh() [1/8]	371
8.32.2.31 letoh() [2/8]	371
8.32.2.32 letoh() [3/8]	371
8.32.2.33 letoh() [4/8]	371
8.32.2.34 letoh() [5/8]	371
8.32.2.35 letoh() [6/8]	372
8.32.2.36 letoh() [7/8]	372
8.32.2.37 letoh() [8/8]	372
8.33 BEntry.cpp File Reference	372
8.34 BEntry.h File Reference	372
8.35 BError.cpp File Reference	373

---

8.36 BError.h File Reference . . . . . 373

    8.36.1 Enumeration Type Documentation . . . . . 373

        8.36.1.1 BErrorNum . . . . . 373

8.37 BErrorTime.cpp File Reference . . . . . 374

8.38 BErrorTime.h File Reference . . . . . 374

8.39 BEvent.cpp File Reference . . . . . 374

8.40 BEvent.h File Reference . . . . . 374

    8.40.1 Typedef Documentation . . . . . 375

        8.40.1.1 BEventQueue . . . . . 375

8.41 BEvent1.cpp File Reference . . . . . 375

8.42 BEvent1.h File Reference . . . . . 375

    8.42.1 Enumeration Type Documentation . . . . . 376

        8.42.1.1 BEvent1Type . . . . . 376

8.43 BFifo.h File Reference . . . . . 376

8.44 BFifo.inc File Reference . . . . . 376

8.45 BFifoCirc.cpp File Reference . . . . . 376

    8.45.1 Macro Definition Documentation . . . . . 377

        8.45.1.1 dprintf . . . . . 377

8.46 BFifoCirc.h File Reference . . . . . 377

8.47 BFifoCirc.inc File Reference . . . . . 377

8.48 BFile.cpp File Reference . . . . . 377

    8.48.1 Macro Definition Documentation . . . . . 377

        8.48.1.1 STRBUF . . . . . 378

8.49 BFile.h File Reference . . . . . 378

8.50 BFileCsv.cpp File Reference . . . . . 378

8.51 BFileCsv.h File Reference . . . . . 378

8.52 BFileData.cpp File Reference . . . . . 378

8.53 BFileData.h File Reference . . . . . 378

8.54 BList.h File Reference . . . . . 379

    8.54.1 Macro Definition Documentation . . . . . 379

---

8.54.1.1	BListLoop	379
8.55	BList_func.h File Reference	379
8.56	BMutex.cpp File Reference	379
8.56.1	Macro Definition Documentation	380
8.56.1.1	MDEBUG	380
8.57	BMutex.h File Reference	380
8.58	BMySQL.cpp File Reference	380
8.59	BMySQL.h File Reference	380
8.60	BNameValue.h File Reference	381
8.61	Boap.cpp File Reference	381
8.61.1	Macro Definition Documentation	381
8.61.1.1	APIVERSION_TEST	381
8.61.1.2	DEBUG	382
8.61.1.3	dprintf	382
8.61.1.4	IS_BIG_ENDIAN	382
8.61.2	Variable Documentation	382
8.61.2.1	boapPort	382
8.62	Boap.h File Reference	382
8.62.1	Typedef Documentation	383
8.62.1.1	BoapFunc	383
8.62.1.2	BoapService	383
8.62.2	Enumeration Type Documentation	383
8.62.2.1	BoapPriority	383
8.62.2.2	BoapType	384
8.62.3	Variable Documentation	384
8.62.3.1	BoapMagic	384
8.63	BoapMc.cpp File Reference	384
8.63.1	Macro Definition Documentation	385
8.63.1.1	DEBUG_LOCAL	385
8.63.1.2	DEBUG_LOCAL1	385

---

8.63.1.3	dl1printf	385
8.63.1.4	dlprintf	385
8.64	BoapMc.h File Reference	385
8.64.1	Enumeration Type Documentation	386
8.64.1.1	BoapMcType	386
8.64.2	Function Documentation	386
8.64.2.1	__attribute__()	386
8.64.3	Variable Documentation	386
8.64.3.1	__attribute__	387
8.64.3.2	addressFrom	387
8.64.3.3	addressTo	387
8.64.3.4	checksum	387
8.64.3.5	cmd	387
8.64.3.6	error	387
8.64.3.7	length	387
8.65	BoapMc1.cpp File Reference	388
8.65.1	Macro Definition Documentation	388
8.65.1.1	BDEBUGL1	388
8.65.1.2	BDEBUGL2	388
8.66	BoapMc1.h File Reference	388
8.66.1	Enumeration Type Documentation	389
8.66.1.1	BoapMc1Type	389
8.66.2	Function Documentation	389
8.66.2.1	__attribute__()	390
8.66.2.2	boapMc1CommsRoundupLen()	390
8.66.3	Variable Documentation	390
8.66.3.1	__attribute__	390
8.66.3.2	addressFrom	390
8.66.3.3	addressTo	390
8.66.3.4	BoapMc1Magic	390



---

8.66.3.5	checksum	391
8.66.3.6	cmd	391
8.66.3.7	data	391
8.66.3.8	error	391
8.66.3.9	head	391
8.66.3.10	length	391
8.66.3.11	magic	391
8.66.3.12	number	392
8.66.3.13	string	392
8.67	BoapnsC.cpp File Reference	392
8.68	BoapnsC.h File Reference	392
8.69	BoapnsD.cpp File Reference	393
8.70	BoapnsD.h File Reference	393
8.71	BoapSimple.cc File Reference	393
8.71.1	Macro Definition Documentation	394
8.71.1.1	DEBUG	394
8.71.1.2	dprintf	394
8.71.2	Variable Documentation	394
8.71.2.1	roundSize	394
8.72	BoapSimple.h File Reference	394
8.72.1	Typedef Documentation	395
8.72.1.1	BoapFunc	395
8.72.1.2	BoapService	395
8.72.1.3	Double	395
8.72.1.4	Int16	395
8.72.1.5	Int32	396
8.72.1.6	Int8	396
8.72.1.7	UInt16	396
8.72.1.8	UInt32	396
8.72.1.9	UInt8	396

---

8.72.2 Enumeration Type Documentation . . . . .	396
8.72.2.1 BoapType . . . . .	396
8.73 BObj.cpp File Reference . . . . .	397
8.74 BObj.h File Reference . . . . .	397
8.75 BObjStringFormat.cpp File Reference . . . . .	397
8.75.1 Function Documentation . . . . .	398
8.75.1.1 toBDictStringFromJson() . . . . .	398
8.75.1.2 toBString() [1/18] . . . . .	398
8.75.1.3 toBString() [2/18] . . . . .	398
8.75.1.4 toBString() [3/18] . . . . .	398
8.75.1.5 toBString() [4/18] . . . . .	399
8.75.1.6 toBString() [5/18] . . . . .	399
8.75.1.7 toBString() [6/18] . . . . .	399
8.75.1.8 toBString() [7/18] . . . . .	399
8.75.1.9 toBString() [8/18] . . . . .	399
8.75.1.10 toBString() [9/18] . . . . .	399
8.75.1.11 toBString() [10/18] . . . . .	400
8.75.1.12 toBString() [11/18] . . . . .	400
8.75.1.13 toBString() [12/18] . . . . .	400
8.75.1.14 toBString() [13/18] . . . . .	400
8.75.1.15 toBString() [14/18] . . . . .	400
8.75.1.16 toBString() [15/18] . . . . .	400
8.75.1.17 toBString() [16/18] . . . . .	401
8.75.1.18 toBString() [17/18] . . . . .	401
8.75.1.19 toBString() [18/18] . . . . .	401
8.75.1.20 toBStringJson() [1/18] . . . . .	401
8.75.1.21 toBStringJson() [2/18] . . . . .	401
8.75.1.22 toBStringJson() [3/18] . . . . .	401
8.75.1.23 toBStringJson() [4/18] . . . . .	402
8.75.1.24 toBStringJson() [5/18] . . . . .	402

8.75.1.25 toBStringJson() [6/18]	402
8.75.1.26 toBStringJson() [7/18]	402
8.75.1.27 toBStringJson() [8/18]	402
8.75.1.28 toBStringJson() [9/18]	402
8.75.1.29 toBStringJson() [10/18]	403
8.75.1.30 toBStringJson() [11/18]	403
8.75.1.31 toBStringJson() [12/18]	403
8.75.1.32 toBStringJson() [13/18]	403
8.75.1.33 toBStringJson() [14/18]	403
8.75.1.34 toBStringJson() [15/18]	403
8.75.1.35 toBStringJson() [16/18]	404
8.75.1.36 toBStringJson() [17/18]	404
8.75.1.37 toBStringJson() [18/18]	404
8.76 BObjStringFormat.h File Reference	404
8.76.1 Function Documentation	405
8.76.1.1 base64_decode()	405
8.76.1.2 base64_encode()	405
8.76.1.3 toBDictStringFromJson()	405
8.76.1.4 toBString() [1/18]	406
8.76.1.5 toBString() [2/18]	406
8.76.1.6 toBString() [3/18]	406
8.76.1.7 toBString() [4/18]	406
8.76.1.8 toBString() [5/18]	406
8.76.1.9 toBString() [6/18]	406
8.76.1.10 toBString() [7/18]	407
8.76.1.11 toBString() [8/18]	407
8.76.1.12 toBString() [9/18]	407
8.76.1.13 toBString() [10/18]	407
8.76.1.14 toBString() [11/18]	407
8.76.1.15 toBString() [12/18]	407

---

8.76.1.16 toBString() [13/18] . . . . .	408
8.76.1.17 toBString() [14/18] . . . . .	408
8.76.1.18 toBString() [15/18] . . . . .	408
8.76.1.19 toBString() [16/18] . . . . .	408
8.76.1.20 toBString() [17/18] . . . . .	408
8.76.1.21 toBString() [18/18] . . . . .	408
8.76.1.22 toBStringJson() [1/18] . . . . .	409
8.76.1.23 toBStringJson() [2/18] . . . . .	409
8.76.1.24 toBStringJson() [3/18] . . . . .	409
8.76.1.25 toBStringJson() [4/18] . . . . .	409
8.76.1.26 toBStringJson() [5/18] . . . . .	409
8.76.1.27 toBStringJson() [6/18] . . . . .	409
8.76.1.28 toBStringJson() [7/18] . . . . .	410
8.76.1.29 toBStringJson() [8/18] . . . . .	410
8.76.1.30 toBStringJson() [9/18] . . . . .	410
8.76.1.31 toBStringJson() [10/18] . . . . .	410
8.76.1.32 toBStringJson() [11/18] . . . . .	410
8.76.1.33 toBStringJson() [12/18] . . . . .	410
8.76.1.34 toBStringJson() [13/18] . . . . .	411
8.76.1.35 toBStringJson() [14/18] . . . . .	411
8.76.1.36 toBStringJson() [15/18] . . . . .	411
8.76.1.37 toBStringJson() [16/18] . . . . .	411
8.76.1.38 toBStringJson() [17/18] . . . . .	411
8.76.1.39 toBStringJson() [18/18] . . . . .	411
8.77 BPoll.cpp File Reference . . . . .	412
8.78 BPoll.h File Reference . . . . .	412
8.79 BQueue.h File Reference . . . . .	412
8.79.1 Typedef Documentation . . . . .	412
8.79.1.1 BQueueInt . . . . .	412
8.80 BRefData.cpp File Reference . . . . .	413

---

---

8.80.1	Macro Definition Documentation	413
8.80.1.1	CHUNK	413
8.81	BRefData.h File Reference	413
8.82	BRtc.cpp File Reference	413
8.83	BRtc.h File Reference	414
8.84	BRWLock.cpp File Reference	414
8.85	BRWLock.h File Reference	414
8.86	BSema.cpp File Reference	414
8.87	BSema.h File Reference	414
8.88	BSemaphore.cpp File Reference	415
8.89	BSemaphore.h File Reference	415
8.90	BSocket.cpp File Reference	415
8.90.1	Macro Definition Documentation	416
8.90.1.1	IP_MTU	416
8.91	BSocket.h File Reference	416
8.91.1	Macro Definition Documentation	416
8.91.1.1	MSG_NOSIGNAL	416
8.91.1.2	SO_PRIORITY	417
8.91.1.3	SOL_IP	417
8.92	BSpi.cpp File Reference	417
8.93	BSpi.h File Reference	417
8.94	BString.cpp File Reference	417
8.94.1	Macro Definition Documentation	418
8.94.1.1	MINUS	418
8.94.1.2	STRIP	418
8.94.2	Function Documentation	419
8.94.2.1	barrayToString()	419
8.94.2.2	blistToString()	419
8.94.2.3	bstringListinList()	419
8.94.2.4	bstringToArray()	419

---

8.94.2.5	<a href="#">bstringToList()</a>	419
8.94.2.6	<a href="#">bstrncpy()</a>	419
8.94.2.7	<a href="#">bstrtrim()</a>	420
8.94.2.8	<a href="#">charToArray()</a>	420
8.94.2.9	<a href="#">charToList()</a>	420
8.94.2.10	<a href="#">floatToString()</a>	420
8.94.2.11	<a href="#">fromBString()</a> [1/6]	420
8.94.2.12	<a href="#">fromBString()</a> [2/6]	420
8.94.2.13	<a href="#">fromBString()</a> [3/6]	421
8.94.2.14	<a href="#">fromBString()</a> [4/6]	421
8.94.2.15	<a href="#">fromBString()</a> [5/6]	421
8.94.2.16	<a href="#">fromBString()</a> [6/6]	421
8.94.2.17	<a href="#">gmatch()</a>	421
8.94.2.18	<a href="#">int64ToString()</a>	421
8.94.2.19	<a href="#">intToString()</a>	422
8.94.2.20	<a href="#">operator&lt;&lt;()</a>	422
8.94.2.21	<a href="#">operator&gt;&gt;()</a>	422
8.94.2.22	<a href="#">toBString()</a> [1/6]	422
8.94.2.23	<a href="#">toBString()</a> [2/6]	422
8.94.2.24	<a href="#">toBString()</a> [3/6]	422
8.94.2.25	<a href="#">toBString()</a> [4/6]	423
8.94.2.26	<a href="#">toBString()</a> [5/6]	423
8.94.2.27	<a href="#">toBString()</a> [6/6]	423
8.94.3	<a href="#">Variable Documentation</a>	423
8.94.3.1	<a href="#">base64_decode_table</a>	423
8.95	<a href="#">BString.h File Reference</a>	423
8.95.1	<a href="#">Function Documentation</a>	424
8.95.1.1	<a href="#">bstrncpy()</a>	424
8.95.1.2	<a href="#">bstrtrim()</a>	424
8.95.1.3	<a href="#">floatToString()</a>	425

8.95.1.4	<a href="#">from_hex()</a>	425
8.95.1.5	<a href="#">fromBString()</a> [1/6]	425
8.95.1.6	<a href="#">fromBString()</a> [2/6]	425
8.95.1.7	<a href="#">fromBString()</a> [3/6]	425
8.95.1.8	<a href="#">fromBString()</a> [4/6]	425
8.95.1.9	<a href="#">fromBString()</a> [5/6]	426
8.95.1.10	<a href="#">fromBString()</a> [6/6]	426
8.95.1.11	<a href="#">int64ToString()</a>	426
8.95.1.12	<a href="#">intToString()</a>	426
8.95.1.13	<a href="#">operator&lt;&lt;()</a>	426
8.95.1.14	<a href="#">operator&gt;&gt;()</a>	426
8.95.1.15	<a href="#">to_hex()</a>	427
8.95.1.16	<a href="#">toBString()</a> [1/6]	427
8.95.1.17	<a href="#">toBString()</a> [2/6]	427
8.95.1.18	<a href="#">toBString()</a> [3/6]	427
8.95.1.19	<a href="#">toBString()</a> [4/6]	427
8.95.1.20	<a href="#">toBString()</a> [5/6]	427
8.95.1.21	<a href="#">toBString()</a> [6/6]	428
8.96	<a href="#">BStringLocked.h File Reference</a>	428
8.97	<a href="#">BSystem.cpp File Reference</a>	428
8.97.1	<a href="#">Function Documentation</a>	428
8.97.1.1	<a href="#">delayMs()</a>	428
8.97.1.2	<a href="#">delayUs()</a>	429
8.98	<a href="#">BSystem.h File Reference</a>	429
8.98.1	<a href="#">Function Documentation</a>	429
8.98.1.1	<a href="#">delayMs()</a>	429
8.98.1.2	<a href="#">delayUs()</a>	429
8.99	<a href="#">BTable.cpp File Reference</a>	429
8.100	<a href="#">BTable.h File Reference</a>	430
8.101	<a href="#">BTask.cpp File Reference</a>	430

8.102BTask.h File Reference . . . . .	430
8.103BThread.cpp File Reference . . . . .	430
8.104BThread.h File Reference . . . . .	430
8.105BTime.cpp File Reference . . . . .	431
8.105.1 Function Documentation . . . . .	431
8.105.1.1 yearDays() . . . . .	431
8.105.1.2 yearIsLeap() . . . . .	431
8.105.2 Variable Documentation . . . . .	431
8.105.2.1 monDays . . . . .	431
8.106BTime.h File Reference . . . . .	432
8.107BTimer.cpp File Reference . . . . .	432
8.108BTimer.h File Reference . . . . .	432
8.109BTimeStamp.cpp File Reference . . . . .	432
8.109.1 Function Documentation . . . . .	433
8.109.1.1 fromBString() . . . . .	433
8.109.1.2 toBString() . . . . .	433
8.109.2 Variable Documentation . . . . .	433
8.109.2.1 mon_yday . . . . .	433
8.110BTimeStamp.h File Reference . . . . .	433
8.110.1 Function Documentation . . . . .	434
8.110.1.1 fromBString() . . . . .	434
8.110.1.2 toBString() . . . . .	434
8.111BTimeStampMs.cpp File Reference . . . . .	434
8.111.1 Variable Documentation . . . . .	434
8.111.1.1 mon_yday . . . . .	434
8.112BTimeStampMs.h File Reference . . . . .	435
8.113BTimeUs.cpp File Reference . . . . .	435
8.113.1 Function Documentation . . . . .	435
8.113.1.1 yearDays() . . . . .	435
8.113.1.2 yearIsLeap() . . . . .	435



8.113.2 Variable Documentation	435
8.113.2.1 monDays	436
8.114BTimeUs.h File Reference	436
8.115BTypes.h File Reference	436
8.115.1 Typedef Documentation	437
8.115.1.1 BArrayDouble	437
8.115.1.2 BArrayFloat	437
8.115.1.3 BChar	438
8.115.1.4 BDouble	438
8.115.1.5 BFloat	438
8.115.1.6 BFloat32	438
8.115.1.7 BFloat64	438
8.115.1.8 BInt	438
8.115.1.9 BInt16	438
8.115.1.10BInt32	438
8.115.1.11BInt64	439
8.115.1.12BInt8	439
8.115.1.13BBool	439
8.115.1.14BSize	439
8.115.1.15BTimeout	439
8.115.1.16BUInt	439
8.115.1.17BUInt16	439
8.115.1.18BUInt32	439
8.115.1.19BUInt64	440
8.115.1.20BUInt8	440
8.115.2 Enumeration Type Documentation	440
8.115.2.1 BEventType	440
8.115.2.2 BEventWaitSet	440
8.115.2.3 BType	440
8.115.2.4 BTypeComp	441
8.115.3 Function Documentation	441
8.115.3.1 byteSwap16()	441
8.115.3.2 byteSwap32()	442
8.115.3.3 byteSwap64()	442
8.115.3.4 byteSwap8()	442
8.115.3.5 timeoutTicks()	442
8.115.4 Variable Documentation	442
8.115.4.1 BTimeoutForever	442
8.116BUrl.cpp File Reference	442
8.117BUrl.h File Reference	443
8.118overview.dox File Reference	443
<b>Index</b>	<b>445</b>



# Chapter 1

## Main Page

### Author

Dr Terry Barnaby

### Version

2.15.1

### Date

2019-10-14

## 1.1 Introduction

The Beam Beam-lib C++ class library provides a base library for developing real-time and other applications with multi-processor and multi-host support. The Bmeam-lib system has the following features:

- Simple Object based development.
- Simple Object base library for Strings, Lists, Network access etc.
- Support for multi-threaded applications with Mutex Objects etc.
- Usable from C++ and Python.
- IDL based object creation tool allows easy creation of C++ and Python objects from IDL language.
- IDL provides the ability to create SQL database schema automatically.
- Database access that allows BDEV Objects to be stored.
- BOAP (Beam Object Access Protocol) provides a simple, low overhead protocol, that allows access to remote objects using an RPC mechanism.
- Database access via a layer that allows simultaneous access to different database systems including MYSQL and BEAM BDEV native object database.
- Concept of Object domains.

## 1.2 Components

The beam-lib system is split into the following libraries:

- LibBeam: This is the base class library containing the base 'C++' classes.
- LibBeamPy: This contains Python components to allow access to the BEAM class library from Python (limited access).
- Bidl: The BEAM BDEV IDL compiler. This takes Object definitions and Interface definitions and creates C++ and Python objects to implement them.

## 1.3 API Examples

Some simple client examples are listed below:

# Chapter 2

## Namespace Index

### 2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

[Boapns](#) . . . . . 17



# Chapter 3

## Hierarchical Index

### 3.1 Class Hierarchy

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

- BAtomic< Type > . . . . . 22
- BAtomicCount . . . . . 24
- BBuffer . . . . . 26
  - BBufferStore . . . . . 29
  - BoapPacket . . . . . 203
- BComms . . . . . 36
- BCond . . . . . 41
- BCondBool . . . . . 43
- BCondInt . . . . . 46
- BCondResource . . . . . 50
- BCondValue . . . . . 52
- BCondWrap . . . . . 57
- BDataChunk . . . . . 63
- BDate . . . . . 64
- BDebugBacktrace . . . . . 70
- BDictItem< Type > . . . . . 77
- BDuration . . . . . 85
- BEntry . . . . . 89
- BError . . . . . 99
  - BEvent1Error . . . . . 109
- BErrorTime . . . . . 103
- BEvent . . . . . 106
- BEvent1 . . . . . 107
  - BEvent1Error . . . . . 109
- BEvent1Int . . . . . 110
- BEvent1Pipe . . . . . 112
- BEventPipe . . . . . 114
- BFifo< Type > . . . . . 117
- BFifo< BoapMcPacket > . . . . . 117
- BFifoCirc< Type > . . . . . 123
- BFifoCircPos . . . . . 130
- BFile . . . . . 133
  - BFileCsv . . . . . 139
- Blter . . . . . 142

BList< T > . . . . .	144
BQueue< T > . . . . .	237
BList< BArray< BString > > . . . . .	144
BList< BDictItem< Type > > . . . . .	144
BDict< Type > . . . . .	71
BConfig . . . . .	61
BList< BEntry > . . . . .	144
BEntryList . . . . .	95
BEntryFile . . . . .	92
BList< BNameValue< T > > . . . . .	144
BNameValueList< T > . . . . .	165
BList< BoapFuncEntry > . . . . .	144
BList< BoapMcPacket > . . . . .	144
BQueue< BoapMcPacket > . . . . .	237
BList< BoapServerConnection *> . . . . .	144
BList< BoapServiceEntry > . . . . .	144
BList< BString > . . . . .	144
BList< BStringList > . . . . .	144
BFileData . . . . .	140
BList< struct dirent *> . . . . .	144
BDir . . . . .	81
BMutex . . . . .	155
BStringMutex . . . . .	296
BMutexLock . . . . .	158
BMysql . . . . .	159
BNameValue< T > . . . . .	163
BNode . . . . .	166
BList< T >::Node . . . . .	340
Boapns::BoapEntry . . . . .	173
BoapFuncEntry . . . . .	174
BoapMc1Comms . . . . .	176
BoapMc1Error . . . . .	183
BoapMc1Packet . . . . .	184
BoapMc1PacketHead . . . . .	184
BoapMcClientObject . . . . .	186
BoapMcComms . . . . .	188
BoapMcPacket . . . . .	196
BoapMcPacketHead . . . . .	197
BoapMcServiceObject . . . . .	198
BoapMcSignalObject . . . . .	200
BoapPacketHead . . . . .	210
BoapServiceEntry . . . . .	223
BoapServiceObject . . . . .	224
BObj . . . . .	230
BObjMember . . . . .	233
BPoll . . . . .	234
BRefData . . . . .	240
BRtc . . . . .	243
BRWLock . . . . .	247
BSema . . . . .	250
BSemaphore . . . . .	252
BSemaphoreBool . . . . .	254
BSemaphoreCount . . . . .	257
BSocket . . . . .	260
BoapClientObject . . . . .	167



Boapns::Boapns . . . . .	201
BoapClientObject . . . . .	167
BoapSignalObject . . . . .	229
BoapSignalObject . . . . .	229
BSocketAddress . . . . .	267
BSocketAddressINET . . . . .	270
BSpi . . . . .	273
BString . . . . .	275
BStringLocked . . . . .	294
BTable . . . . .	297
BTask . . . . .	299
BThread . . . . .	302
BoapServer . . . . .	212
BoapServerConnection . . . . .	220
BRtcThreaded . . . . .	245
BTime . . . . .	306
BTimer . . . . .	311
BTimeStamp . . . . .	314
BTimeStampMs . . . . .	324
BTimeUs . . . . .	332
BUrl . . . . .	338
map	
BDictMap< Value > . . . . .	78
vector	
BArray< T > . . . . .	19
BArray< BList< BIter > > . . . . .	19
BArray< BString > . . . . .	19
BArray< int > . . . . .	19



# Chapter 4

## Class Index

### 4.1 Class List

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

<a href="#">BArray&lt; T &gt;</a>	19
<a href="#">BAtomic&lt; Type &gt;</a>	
<a href="#">BAtomic</a> class	22
<a href="#">BAtomicCount</a>	
<a href="#">BAtomicCount</a> class	24
<a href="#">BBuffer</a>	26
<a href="#">BBufferStore</a>	29
<a href="#">BComms</a>	36
<a href="#">BCond</a>	41
<a href="#">BCondBool</a>	
Thread conditional boolean	43
<a href="#">BCondInt</a>	
Thread conditional value	46
<a href="#">BCondResource</a>	
Resource lock	50
<a href="#">BCondValue</a>	
Thread conditional value	52
<a href="#">BCondWrap</a>	57
<a href="#">BConfig</a>	
This class implements the configuration file access	61
<a href="#">BDataChunk</a>	63
<a href="#">BDate</a>	64
<a href="#">BDebugBacktrace</a>	70
<a href="#">BDict&lt; Type &gt;</a>	71
<a href="#">BDictItem&lt; Type &gt;</a>	
Template based Dictionary class	77
<a href="#">BDictMap&lt; Value &gt;</a>	78
<a href="#">BDir</a>	
File system directory class	81
<a href="#">BDuration</a>	85
<a href="#">BEntry</a>	
Manipulate a name value pair	89
<a href="#">BEntryFile</a>	
File of Entries	92
<a href="#">BEntryList</a>	
List of Entries. Where an entry is a name value pair	95

<a href="#">BError</a>	99
<a href="#">BErrorTime</a>	
Error return class	103
<a href="#">BEvent</a>	106
<a href="#">BEvent1</a>	
This class provides a base class for all event objects that can be sent over the events interface	107
<a href="#">BEvent1Error</a>	109
<a href="#">BEvent1Int</a>	
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	110
<a href="#">BEvent1Pipe</a>	
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	112
<a href="#">BEventPipe</a>	
This class provides an interface for sending simple integer events via a pipe file descriptor	114
<a href="#">BFifo&lt; Type &gt;</a>	117
<a href="#">BFifoCirc&lt; Type &gt;</a>	
This class implements a thread safe FIFO buffer	123
<a href="#">BFifoCircPos</a>	
This class implements a pointer into the Fifo's circular buffer	130
<a href="#">BFile</a>	
File operations class	133
<a href="#">BFileCsv</a>	139
<a href="#">BFileData</a>	140
<a href="#">BIter</a>	
Iterator for <a href="#">BList</a>	142
<a href="#">BList&lt; T &gt;</a>	
Template based list class	144
<a href="#">BMutex</a>	155
<a href="#">BMutexLock</a>	158
<a href="#">BMySQL</a>	159
<a href="#">BNameValue&lt; T &gt;</a>	163
<a href="#">BNameValueList&lt; T &gt;</a>	165
<a href="#">BNode</a>	166
<a href="#">BoapClientObject</a>	167
<a href="#">Boapns::BoapEntry</a>	173
<a href="#">BoapFuncEntry</a>	174
<a href="#">BoapMc1Comms</a>	176
<a href="#">BoapMc1Error</a>	183
<a href="#">BoapMc1Packet</a>	184
<a href="#">BoapMc1PacketHead</a>	184
<a href="#">BoapMcClientObject</a>	186
<a href="#">BoapMcComms</a>	188
<a href="#">BoapMcPacket</a>	196
<a href="#">BoapMcPacketHead</a>	197
<a href="#">BoapMcServiceObject</a>	198
<a href="#">BoapMcSignalObject</a>	200
<a href="#">Boapns::Boapns</a>	201
<a href="#">BoapPacket</a>	203
<a href="#">BoapPacketHead</a>	210
<a href="#">BoapServer</a>	212
<a href="#">BoapServerConnection</a>	220
<a href="#">BoapServiceEntry</a>	223
<a href="#">BoapServiceObject</a>	224
<a href="#">BoapSignalObject</a>	229
<a href="#">BObj</a>	230
<a href="#">BObjMember</a>	233

<a href="#">BPoll</a>	This class provides an interface for polling a number of file descriptors. It uses round robin polling	<a href="#">234</a>
<a href="#">BQueue&lt; T &gt;</a>	Queue class	<a href="#">237</a>
<a href="#">BRefData</a>		<a href="#">240</a>
<a href="#">BRtc</a>	Realtime clock	<a href="#">243</a>
<a href="#">BRtcThreaded</a>	Threaded real time clock	<a href="#">245</a>
<a href="#">BRWLock</a>	Thread read-write locks	<a href="#">247</a>
<a href="#">BSema</a>	Sempahore class	<a href="#">250</a>
<a href="#">BSemaphore</a>	Semaphore class	<a href="#">252</a>
<a href="#">BSemaphoreBool</a>		<a href="#">254</a>
<a href="#">BSemaphoreCount</a>		<a href="#">257</a>
<a href="#">BSocket</a>		<a href="#">260</a>
<a href="#">BSocketAddress</a>	Socket Address	<a href="#">267</a>
<a href="#">BSocketAddressINET</a>	IP aware socket address	<a href="#">270</a>
<a href="#">BSpi</a>	BSpi class	<a href="#">273</a>
<a href="#">BString</a>		<a href="#">275</a>
<a href="#">BStringLocked</a>		<a href="#">294</a>
<a href="#">BStringMutex</a>		<a href="#">296</a>
<a href="#">BTable</a>		<a href="#">297</a>
<a href="#">BTask</a>		<a href="#">299</a>
<a href="#">BThread</a>		<a href="#">302</a>
<a href="#">BTime</a>		<a href="#">306</a>
<a href="#">BTimer</a>	Stopwatch style timer	<a href="#">311</a>
<a href="#">BTimeStamp</a>		<a href="#">314</a>
<a href="#">BTimeStampMs</a>		<a href="#">324</a>
<a href="#">BTimeUs</a>		<a href="#">332</a>
<a href="#">BUrl</a>	Basic access to a Url	<a href="#">338</a>
<a href="#">BList&lt; T &gt;::Node</a>		<a href="#">340</a>



# Chapter 5

## File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

BArray.h	341
BAtomic.h	341
BAtomicCount.h	342
BBuffer.cpp	343
BBuffer.h	343
BComms.cpp	344
BComms.h	344
BComplex.h	344
BCond.cpp	345
BCond.h	345
BCondInt.cpp	345
BCondInt.h	346
BConfig.cpp	346
BConfig.h	346
BCrc16.cpp	347
BCrc16.h	348
BCrc32.cpp	348
BCrc32.h	349
BDate-1.cpp	350
BDate.cpp	351
BDate.h	352
BDebug.cpp	353
BDebug.h	355
BDict.cpp	359
BDict.h	360
BDictMap.h	361
BDir.cpp	361
BDir.h	362
BDuration.cpp	362
BDuration.h	363
BEndian.cpp	363
BEndian.h	363
BEntry.cpp	372
BEntry.h	372
BError.cpp	373

BError.h	373
BErrorTime.cpp	374
BErrorTime.h	374
BEvent.cpp	374
BEvent.h	374
BEvent1.cpp	375
BEvent1.h	375
BFifo.h	376
BFifo.inc	376
BFifoCirc.cpp	376
BFifoCirc.h	377
BFifoCirc.inc	377
BFile.cpp	377
BFile.h	378
BFileCsv.cpp	378
BFileCsv.h	378
BFileData.cpp	378
BFileData.h	378
BList.h	379
BList_func.h	379
BMutex.cpp	379
BMutex.h	380
BMysql.cpp	380
BMysql.h	380
BNameValue.h	381
Boap.cpp	381
Boap.h	382
BoapMc.cpp	384
BoapMc.h	385
BoapMc1.cpp	388
BoapMc1.h	388
BoapnsC.cpp	392
BoapnsC.h	392
BoapnsD.cpp	393
BoapnsD.h	393
BoapSimple.cc	393
BoapSimple.h	394
BObj.cpp	397
BObj.h	397
BObjStringFormat.cpp	397
BObjStringFormat.h	404
BPoll.cpp	412
BPoll.h	412
BQueue.h	412
BRefData.cpp	413
BRefData.h	413
BRtc.cpp	413
BRtc.h	414
BRWLock.cpp	414
BRWLock.h	414
BSema.cpp	414
BSema.h	414
BSemaphore.cpp	415
BSemaphore.h	415
BSocket.cpp	415
BSocket.h	416
BSpi.cpp	417
BSpi.h	417



BString.cpp	417
BString.h	423
BStringLocked.h	428
BSys.cpp	428
BSys.h	429
BTable.cpp	429
BTable.h	430
BTask.cpp	430
BTask.h	430
BThread.cpp	430
BThread.h	430
BTime.cpp	431
BTime.h	432
BTimer.cpp	432
BTimer.h	432
BTimeStamp.cpp	432
BTimeStamp.h	433
BTimeStampMs.cpp	434
BTimeStampMs.h	435
BTimeUs.cpp	435
BTimeUs.h	436
BTypes.h	436
BUrl.cpp	442
BUrl.h	443



## Chapter 6

# Namespace Documentation

### 6.1 Boapns Namespace Reference

#### Classes

- class [BoapEntry](#)
- class [Boapns](#)

#### Variables

- const [BUInt32](#) [apiVersion](#) = 0

#### 6.1.1 Variable Documentation

##### 6.1.1.1 [apiVersion](#)

```
const BUInt32 Boapns::apiVersion = 0
```



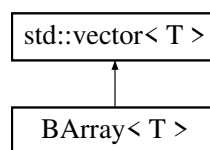
# Chapter 7

## Class Documentation

### 7.1 BArray< T > Class Template Reference

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

Inheritance diagram for BArray< T >:



#### Public Types

- typedef int(\* [SortFunc](#)) (T &a, T &b)  
*Prototype for sorting function.*

#### Public Member Functions

- [BArray](#) ()
- [BArray](#) ([BSize](#) size, T value=T())
- [BArray](#) (const [BArray](#) &array)
- [BUInt](#) [number](#) () const
- void [append](#) (const T &value)
- void [append](#) (const [BArray](#)< T > &array)
- void [insert](#) ([BUInt](#) pos, const T &value)
- void [del](#) ([BUInt](#) pos, [BUInt](#) num=1)
- T & [rear](#) ()
- void [sort](#) ()

### 7.1.1 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.

### 7.1.2 Member Typedef Documentation

#### 7.1.2.1 SortFunc

```
template<class T>
typedef int (* BArray< T >::SortFunc) (T &a, T &b)
```

Prototype for sorting function.

### 7.1.3 Constructor & Destructor Documentation

#### 7.1.3.1 BArray() [1/3]

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

#### 7.1.3.2 BArray() [2/3]

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

#### 7.1.3.3 BArray() [3/3]

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

## 7.1.4 Member Function Documentation

### 7.1.4.1 append() [1/2]

```
template<class T>
void BArray< T >::append (
    const T & value ) [inline]
```

### 7.1.4.2 append() [2/2]

```
template<class T>
void BArray< T >::append (
    const BArray< T > & array )
```

### 7.1.4.3 del()

```
template<class T>
void BArray< T >::del (
    BUInt pos,
    BUInt num = 1 ) [inline]
```

### 7.1.4.4 insert()

```
template<class T>
void BArray< T >::insert (
    BUInt pos,
    const T & value ) [inline]
```

### 7.1.4.5 number()

```
template<class T>
BUInt BArray< T >::number ( ) const [inline]
```

### 7.1.4.6 rear()

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

### 7.1.4.7 sort()

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

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

- [BArray.h](#)

## 7.2 BAtomic< Type > Class Template Reference

[BAtomic](#) class.

```
#include <BAtomic.h>
```

### Public Member Functions

- [BAtomic](#) (Type value=0)
- Type [getValue](#) () const
- Type [add](#) (long value)
- Type [operator++](#) (int)
- Type [operator++](#) ()
- Type [operator--](#) (int)
- Type [operator--](#) ()
- [operator Type](#) () const

### Private Attributes

- Type [ovalue](#)

### 7.2.1 Detailed Description

```
template<class Type>
class BAtomic< Type >
```

[BAtomic](#) class.



## 7.2.2 Constructor & Destructor Documentation

### 7.2.2.1 BAtomic()

```
template<class Type >
BAtomic< Type >::BAtomic (
    Type value = 0 ) [inline]
```

## 7.2.3 Member Function Documentation

### 7.2.3.1 add()

```
template<class Type >
Type BAtomic< Type >::add (
    long value ) [inline]
```

### 7.2.3.2 getValue()

```
template<class Type >
Type BAtomic< Type >::getValue ( ) const [inline]
```

### 7.2.3.3 operator Type()

```
template<class Type >
BAtomic< Type >::operator Type ( ) const [inline]
```

### 7.2.3.4 operator++() [1/2]

```
template<class Type >
Type BAtomic< Type >::operator++ (
    int ) [inline]
```

### 7.2.3.5 operator++() [2/2]

```
template<class Type >
Type BAtomic< Type >::operator++ ( ) [inline]
```

### 7.2.3.6 operator--() [1/2]

```
template<class Type >
Type BAtomic< Type >::operator-- (
    int ) [inline]
```

### 7.2.3.7 operator--() [2/2]

```
template<class Type >
Type BAtomic< Type >::operator-- ( ) [inline]
```

## 7.2.4 Member Data Documentation

### 7.2.4.1 ovalue

```
template<class Type >
Type BAtomic< Type >::ovalue [mutable], [private]
```

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

- [BAtomic.h](#)

## 7.3 BAtomicCount Class Reference

[BAtomicCount](#) class.

```
#include <BAtomicCount.h>
```

### Public Member Functions

- [BAtomicCount](#) (long value=0)
- long [getValue](#) () const
- long [add](#) (long value)
- long [operator++](#) (int)
- long [operator++](#) ()
- long [operator--](#) (int)
- long [operator--](#) ()
- [operator long](#) () const

## Private Attributes

- `_Atomic_word` [ovalue](#)

## 7.3.1 Detailed Description

[BAtomicCount](#) class.

## 7.3.2 Constructor & Destructor Documentation

### 7.3.2.1 BAtomicCount()

```
BAtomicCount::BAtomicCount (  
    long value = 0 ) [inline]
```

## 7.3.3 Member Function Documentation

### 7.3.3.1 add()

```
long BAtomicCount::add (  
    long value ) [inline]
```

### 7.3.3.2 getValue()

```
long BAtomicCount::getValue ( ) const [inline]
```

### 7.3.3.3 operator long()

```
BAtomicCount::operator long ( ) const [inline]
```

### 7.3.3.4 operator++() [1/2]

```
long BAtomicCount::operator++ (  
    int ) [inline]
```

### 7.3.3.5 operator++() [2/2]

```
long BAtomicCount::operator++ ( ) [inline]
```

### 7.3.3.6 operator--() [1/2]

```
long BAtomicCount::operator-- (
    int ) [inline]
```

### 7.3.3.7 operator--() [2/2]

```
long BAtomicCount::operator-- ( ) [inline]
```

## 7.3.4 Member Data Documentation

### 7.3.4.1 ovalue

```
_Atomic_word BAtomicCount::ovalue [mutable], [private]
```

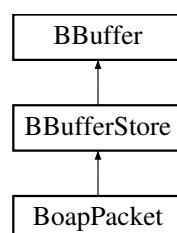
The documentation for this class was generated from the following file:

- [BAtomicCount.h](#)

## 7.4 BBuffer Class Reference

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

Inheritance diagram for BBuffer:



## Public Member Functions

- [BBuffer](#) ([BUInt size=0](#))  
*Create and manipulate a data buffer. On creation the buffer size defaults to 1024 bytes.*
- [~BBuffer](#) ()
- [int setSize](#) ([BUInt32 size](#))  
*Sets the bufer size.*
- [int setData](#) ([const void \\*data](#), [BUInt32 size](#))  
*Sets buffer data resized to contain the data.*
- [int writeData](#) ([BUInt32 pos](#), [const void \\*data](#), [BUInt32 size](#))  
*Writes data into buffer from offset pos.*
- [char \\* data](#) ()  
*The data.*
- [BUInt32 size](#) ()  
*Size of the buffer in bytes.*
- [int resize](#) ([BUInt32 size](#))  
*Alternative to [setSize\(\)](#)*

## Protected Attributes

- [BUInt32 odataSize](#)
- [char \\* odata](#)
- [BUInt32 osize](#)

### 7.4.1 Constructor & Destructor Documentation

#### 7.4.1.1 BBuffer()

```
BBuffer::BBuffer (
    BUInt size = 0 )
```

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

#### 7.4.1.2 ~BBuffer()

```
BBuffer::~BBuffer ( )
```

### 7.4.2 Member Function Documentation

#### 7.4.2.1 data()

```
char * BBuffer::data ( )
```

The data.

#### 7.4.2.2 resize()

```
int BBuffer::resize (
    BUInt32 size ) [inline]
```

Alternative to [setSize\(\)](#)

#### 7.4.2.3 setData()

```
int BBuffer::setData (
    const void * data,
    BUInt32 size )
```

Sets buffer data resized to contain the data.

#### 7.4.2.4 setSize()

```
int BBuffer::setSize (
    BUInt32 size )
```

Sets the bufer size.

#### 7.4.2.5 size()

```
BUInt32 BBuffer::size ( )
```

Size of the buffer in bytes.

#### 7.4.2.6 writeData()

```
int BBuffer::writeData (
    BUInt32 pos,
    const void * data,
    BUInt32 size )
```

Writes data into buffer from offset pos.

### 7.4.3 Member Data Documentation

#### 7.4.3.1 odata

```
char* BBuffer::odata [protected]
```

#### 7.4.3.2 odataSize

```
BUInt32 BBuffer::odataSize [protected]
```

#### 7.4.3.3 osize

```
BUInt32 BBuffer::osize [protected]
```

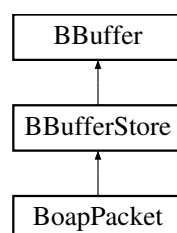
The documentation for this class was generated from the following files:

- [BBuffer.h](#)
- [BBuffer.cpp](#)

## 7.5 BBufferStore Class Reference

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

Inheritance diagram for BBufferStore:



## Public Member Functions

- [BBufferStore](#) ([BUInt](#) size=0, int swapBytes=[BBigEndian](#))
- [~BBufferStore](#) ()
- [BUInt32](#) [getPos](#) ()
- void [setPos](#) ([BUInt32](#) pos)
- [BString](#) [getHexString](#) ()
- void [setHexString](#) ([BString](#) s)
- int [push](#) ([BInt8](#) v)
- int [push](#) ([BUInt8](#) v)
- int [push](#) ([BInt16](#) v)
- int [push](#) ([BUInt16](#) v)
- int [push](#) ([BInt32](#) v)
- int [push](#) ([BUInt32](#) v)
- int [push](#) ([BInt64](#) v)
- int [push](#) ([BUInt64](#) v)
- int [push](#) ([BFloat32](#) v)
- int [push](#) ([BFloat64](#) v)
- int [push](#) (const [BString](#) &v)
- int [push](#) (const [BError](#) &v)
- int [push](#) (const [BTimeStamp](#) &v)
- int [push](#) (const [BComplex](#) &v)
- int [push](#) ([BUInt32](#) nBytes, const void \*data, const char \*swapType="1")
- int [pop](#) ([BInt8](#) &v)
- int [pop](#) ([BUInt8](#) &v)
- int [pop](#) ([BInt16](#) &v)
- int [pop](#) ([BUInt16](#) &v)
- int [pop](#) ([BInt32](#) &v)
- int [pop](#) ([BUInt32](#) &v)
- int [pop](#) ([BInt64](#) &v)
- int [pop](#) ([BUInt64](#) &v)
- int [pop](#) ([BFloat32](#) &v)
- int [pop](#) ([BFloat64](#) &v)
- int [pop](#) ([BString](#) &v)
- int [pop](#) ([BError](#) &v)
- int [pop](#) ([BTimeStamp](#) &v)
- int [pop](#) ([BComplex](#) &v)
- int [pop](#) ([BUInt32](#) nBytes, void \*data, const char \*swapType="1")

## Protected Attributes

- [BUInt32](#) [opos](#)
- int [oswapBytes](#)

### 7.5.1 Constructor & Destructor Documentation



### 7.5.1.1 BBufferStore()

```
BBufferStore::BBufferStore (
    BUInt size = 0,
    int swapBytes = BBigEndian )
```

### 7.5.1.2 ~BBufferStore()

```
BBufferStore::~BBufferStore ( )
```

## 7.5.2 Member Function Documentation

### 7.5.2.1 getHexString()

```
BString BBufferStore::getHexString ( )
```

### 7.5.2.2 getPos()

```
BUInt32 BBufferStore::getPos ( )
```

### 7.5.2.3 pop() [1/15]

```
int BBufferStore::pop (
    BInt8 & v )
```

### 7.5.2.4 pop() [2/15]

```
int BBufferStore::pop (
    BUInt8 & v )
```

**7.5.2.5 pop()** [3/15]

```
int BBufferStore::pop (
    BInt16 & v )
```

**7.5.2.6 pop()** [4/15]

```
int BBufferStore::pop (
    BUInt16 & v )
```

**7.5.2.7 pop()** [5/15]

```
int BBufferStore::pop (
    BInt32 & v )
```

**7.5.2.8 pop()** [6/15]

```
int BBufferStore::pop (
    BUInt32 & v )
```

**7.5.2.9 pop()** [7/15]

```
int BBufferStore::pop (
    BInt64 & v )
```

**7.5.2.10 pop()** [8/15]

```
int BBufferStore::pop (
    BUInt64 & v )
```

**7.5.2.11 pop()** [9/15]

```
int BBufferStore::pop (
    BFloat32 & v )
```

**7.5.2.12 pop()** [10/15]

```
int BBufferStore::pop (
    BFloat64 & v )
```

**7.5.2.13 pop()** [11/15]

```
int BBufferStore::pop (
    BString & v )
```

**7.5.2.14 pop()** [12/15]

```
int BBufferStore::pop (
    BError & v )
```

**7.5.2.15 pop()** [13/15]

```
int BBufferStore::pop (
    BTimeStamp & v )
```

**7.5.2.16 pop()** [14/15]

```
int BBufferStore::pop (
    BComplex & v )
```

**7.5.2.17 pop()** [15/15]

```
int BBufferStore::pop (
    BUInt32 nBytes,
    void * data,
    const char * swapType = "1" )
```

**7.5.2.18 push()** [1/15]

```
int BBufferStore::push (
    BInt8 v )
```

**7.5.2.19 push()** [2/15]

```
int BBufferStore::push (  
    BUInt8 v )
```

**7.5.2.20 push()** [3/15]

```
int BBufferStore::push (  
    BInt16 v )
```

**7.5.2.21 push()** [4/15]

```
int BBufferStore::push (  
    BUInt16 v )
```

**7.5.2.22 push()** [5/15]

```
int BBufferStore::push (  
    BInt32 v )
```

**7.5.2.23 push()** [6/15]

```
int BBufferStore::push (  
    BUInt32 v )
```

**7.5.2.24 push()** [7/15]

```
int BBufferStore::push (  
    BInt64 v )
```

**7.5.2.25 push()** [8/15]

```
int BBufferStore::push (  
    BUInt64 v )
```

**7.5.2.26 push()** [9/15]

```
int BBufferStore::push (
    BFloat32 v )
```

**7.5.2.27 push()** [10/15]

```
int BBufferStore::push (
    BFloat64 v )
```

**7.5.2.28 push()** [11/15]

```
int BBufferStore::push (
    const BString & v )
```

**7.5.2.29 push()** [12/15]

```
int BBufferStore::push (
    const BError & v )
```

**7.5.2.30 push()** [13/15]

```
int BBufferStore::push (
    const BTimeStamp & v )
```

**7.5.2.31 push()** [14/15]

```
int BBufferStore::push (
    const BComplex & v )
```

**7.5.2.32 push()** [15/15]

```
int BBufferStore::push (
    BUInt32 nBytes,
    const void * data,
    const char * swapType = "1" )
```

### 7.5.2.33 setHexString()

```
void BBufferStore::setHexString (
    BString s )
```

### 7.5.2.34 setPos()

```
void BBufferStore::setPos (
    BUInt32 pos )
```

## 7.5.3 Member Data Documentation

### 7.5.3.1 opos

```
BUInt32 BBufferStore::opos [protected]
```

### 7.5.3.2 oswapBytes

```
int BBufferStore::oswapBytes [protected]
```

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

- [BBuffer.h](#)
- [BBuffer.cpp](#)

## 7.6 BComms Class Reference

```
#include <BComms.h>
```

### Public Types

- enum [Flush](#) { [FlushRead](#), [FlushWrite](#), [FlushReadWrite](#) }

## Public Member Functions

- [BComms](#) ()
- virtual [~BComms](#) ()
- virtual [BError](#) [init](#) ()
- virtual [BError](#) [setPacketMode](#) ([Bool](#) packetMode)
  - Set packet mode.*
- virtual [Bool](#) [packetMode](#) ()
  - Device is in packet mode.*
- virtual [BError](#) [setTimeout](#) ([BTimeout](#) timeoutUs)
  - Set communication timeout.*
- virtual [BError](#) [connect](#) (const char \*resource)
  - Create a connection.*
- virtual [Bool](#) [isConnected](#) ()
- virtual [BError](#) [disconnect](#) ()
  - Disconnect.*
- virtual void [flush](#) ([Flush](#) flush)
- virtual [BUInt](#) [writeAvailable](#) ()
- virtual [BError](#) [write](#) (const void \*data, [BUInt32](#) nBytes, [BUInt32](#) &nTrans)=0
- virtual [BError](#) [writeChunks](#) (const [BDataChunk](#) \*chunks, [BUInt](#) nChunks, [BUInt32](#) &nTrans)
- virtual [BUInt](#) [readAvailable](#) ()
- virtual [BError](#) [read](#) (void \*data, [BUInt32](#) num, [BUInt32](#) &nTrans)=0
- virtual [BError](#) [wait](#) ([BEventWaitSet](#) events, [BTimeout](#) timeoutUs=[BTimeoutForever](#), [BUInt32](#) num=1)
- virtual void [eventQueue](#) ([BEventQueue](#) \*eventQueue, [BUInt32](#) event, [BUInt](#) num=1)

## Protected Attributes

- [Bool](#) oconnected
- [Bool](#) opacketMode
- [BTimeout](#) otimeout
- [BEventQueue](#) \* oeventQueue
- [BUInt32](#) oevent
- [BUInt](#) oeventNum

## 7.6.1 Member Enumeration Documentation

### 7.6.1.1 Flush

```
enum BComms::Flush
```

#### Enumerator

<a href="#">FlushRead</a>	
<a href="#">FlushWrite</a>	
<a href="#">FlushReadWrite</a>	

## 7.6.2 Constructor & Destructor Documentation

### 7.6.2.1 BComms()

```
BComms::BComms ( )
```

### 7.6.2.2 ~BComms()

```
BComms::~~BComms ( ) [virtual]
```

## 7.6.3 Member Function Documentation

### 7.6.3.1 connect()

```
BError BComms::connect (
    const char * resource ) [virtual]
```

Create a connection.

### 7.6.3.2 disconnect()

```
BError BComms::disconnect ( ) [virtual]
```

Disconnect.

### 7.6.3.3 eventQueue()

```
void BComms::eventQueue (
    BEventQueue * eventQueue,
    BInt32 event,
    BUInt num = 1 ) [virtual]
```



#### 7.6.3.4 flush()

```
void BComms::flush (
    Flush flush ) [virtual]
```

#### 7.6.3.5 init()

```
BError BComms::init ( ) [virtual]
```

#### 7.6.3.6 isConnected()

```
Bool BComms::isConnected ( ) [virtual]
```

#### 7.6.3.7 packetMode()

```
Bool BComms::packetMode ( ) [virtual]
```

Device is in packet mode.

#### 7.6.3.8 read()

```
virtual BError BComms::read (
    void * data,
    BUInt32 num,
    BUInt32 & nTrans ) [pure virtual]
```

#### 7.6.3.9 readAvailable()

```
BUInt BComms::readAvailable ( ) [virtual]
```

#### 7.6.3.10 setPacketMode()

```
BError BComms::setPacketMode (
    Bool packetMode ) [virtual]
```

Set packet mode.

### 7.6.3.11 setTimeout()

```
BError BComms::setTimeout (
    BTimeout timeoutUs ) [virtual]
```

Set communication timeout.

### 7.6.3.12 wait()

```
BError BComms::wait (
    BEventWaitSet events,
    BTimeout timeoutUs = BTimeoutForever,
    BUInt32 num = 1 ) [virtual]
```

### 7.6.3.13 write()

```
virtual BError BComms::write (
    const void * data,
    BUInt32 nBytes,
    BUInt32 & nTrans ) [pure virtual]
```

### 7.6.3.14 writeAvailable()

```
BUInt BComms::writeAvailable ( ) [virtual]
```

### 7.6.3.15 writeChunks()

```
BError BComms::writeChunks (
    const BDataChunk * chunks,
    BUInt nChunks,
    BUInt32 & nTrans ) [virtual]
```

## 7.6.4 Member Data Documentation

### 7.6.4.1 oconnected

```
Bool BComms::oconnected [protected]
```

#### 7.6.4.2 oevent

`BInt32` BComms::oevent [protected]

#### 7.6.4.3 oeventNum

`BUInt` BComms::oeventNum [protected]

#### 7.6.4.4 oeventQueue

`BEventQueue*` BComms::oeventQueue [protected]

#### 7.6.4.5 opacketMode

`Bool` BComms::opacketMode [protected]

#### 7.6.4.6 otimeout

`BTimeout` BComms::otimeout [protected]

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

- [BComms.h](#)
- [BComms.cpp](#)

## 7.7 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](#)

## 7.7.1 Constructor & Destructor Documentation

### 7.7.1.1 BCond()

```
BCond::BCond ( )
```

Thread conditional variable.

### 7.7.1.2 ~BCond()

```
BCond::~~BCond ( )
```

## 7.7.2 Member Function Documentation

### 7.7.2.1 signal()

```
int BCond::signal ( )
```

### 7.7.2.2 timedWait()

```
int BCond::timedWait (
    int timeOutUs )
```

### 7.7.2.3 wait()

```
int BCond::wait ( )
```

### 7.7.3 Member Data Documentation

#### 7.7.3.1 ocond

```
pthread_cond_t BCond::ocond [private]
```

#### 7.7.3.2 omutex

```
pthread_mutex_t BCond::omutex [private]
```

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

- [BCond.h](#)
- [BCond.cpp](#)

## 7.8 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.*
- [operator int](#) ()

### Private Attributes

- pthread\_mutex\_t [omutex](#)
- pthread\_cond\_t [ocond](#)
- int [ovalue](#)

### 7.8.1 Detailed Description

Thread conditional boolean.

### 7.8.2 Constructor & Destructor Documentation

#### 7.8.2.1 BCondBool()

```
BCondBool::BCondBool ( )
```

#### 7.8.2.2 ~BCondBool()

```
BCondBool::~~BCondBool ( )
```

### 7.8.3 Member Function Documentation

#### 7.8.3.1 clear()

```
int BCondBool::clear ( )
```

Clear Value.

#### 7.8.3.2 operator int()

```
BCondBool::operator int ( ) [inline]
```

#### 7.8.3.3 set()

```
int BCondBool::set ( )
```

Set value. Wakes waiting.

#### 7.8.3.4 timedWait()

```
int BCondBool::timedWait (
    int timeOutUs )
```

Wait until set, with timeout.

#### 7.8.3.5 value()

```
int BCondBool::value ( )
```

Current value.

#### 7.8.3.6 wait()

```
int BCondBool::wait ( )
```

Wait until value is true.

### 7.8.4 Member Data Documentation

#### 7.8.4.1 ocond

```
pthread_cond_t BCondBool::ocond [private]
```

#### 7.8.4.2 omutex

```
pthread_mutex_t BCondBool::omutex [private]
```

#### 7.8.4.3 ovalue

```
int BCondBool::ovalue [private]
```

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

- [BCondInt.h](#)
- [BCondInt.cpp](#)

## 7.9 BCondInt Class Reference

Thread conditional value.

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

### Public Member Functions

- [BCondInt](#) ()
- [~BCondInt](#) ()
- void [setValue](#) ([BInt](#) value)
  - Set the value. Wakes waiting.*
- [BInt](#) value () const
  - Current value.*
- [BInt](#) [increment](#) ([BInt](#) v=1)
  - Increment. Wakes waiting.*
- [BInt](#) [decrement](#) ([BInt](#) v=1)
  - Decrement. Wakes waiting.*
- [Bool](#) [waitMoreThanOrEqual](#) ([BInt](#) v, [Bool](#) decrement=0, [BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - Wait until value is at least the value given.*
- [Bool](#) [waitLessThanOrEqual](#) ([BInt](#) v, [Bool](#) increment=0, [BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - Wait until value is equal to or below the value given.*
- [Bool](#) [waitLessThan](#) ([BInt](#) v, [BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - 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](#)
- [BInt](#) [ovalue](#)

#### 7.9.1 Detailed Description

Thread conditional value.

#### 7.9.2 Constructor & Destructor Documentation



### 7.9.2.1 BCondInt()

```
BCondInt::BCondInt ( )
```

### 7.9.2.2 ~BCondInt()

```
BCondInt::~~BCondInt ( )
```

## 7.9.3 Member Function Documentation

### 7.9.3.1 decrement()

```
BInt BCondInt::decrement (
    BInt v = 1 )
```

Decrement. Wakes waiting.

### 7.9.3.2 increment()

```
BInt BCondInt::increment (
    BInt v = 1 )
```

Increment. Wakes waiting.

### 7.9.3.3 operator++()

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

Increment value. Wakes waiting.

### 7.9.3.4 operator+=()

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

Add to value. Wakes waiting.

### 7.9.3.5 operator--()

```
void BCondInt::operator-- (
    int ) [inline]
```

Decrement value. Wakes waiting.

### 7.9.3.6 operator-=()

```
void BCondInt::operator-= (
    int v ) [inline]
```

Subtract from value. Wakes waiting.

### 7.9.3.7 setValue()

```
void BCondInt::setValue (
    BInt value )
```

Set the value. Wakes waiting.

### 7.9.3.8 value()

```
BInt BCondInt::value ( ) const
```

Current value.

### 7.9.3.9 waitLessThan()

```
Bool BCondInt::waitLessThan (
    BInt v,
    BTimeout timeoutUs = BTimeoutForever )
```

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

### 7.9.3.10 waitLessThanOrEqual()

```
Bool BCondInt::waitLessThanOrEqual (
    BInt v,
    Bool increment = 0,
    BTimeout timeoutUs = BTimeoutForever )
```

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

### 7.9.3.11 waitMoreThanOrEqual()

```
Bool BCondInt::waitMoreThanOrEqual (
    BInt v,
    Bool decrement = 0,
    BTimeout timeoutUs = BTimeoutForever )
```

Wait until value is at least the value given.

## 7.9.4 Member Data Documentation

### 7.9.4.1 ocond

```
pthread_cond_t BCondInt::ocond [private]
```

### 7.9.4.2 omutex

```
pthread_mutex_t BCondInt::omutex [private]
```

### 7.9.4.3 ovalue

```
BInt BCondInt::ovalue [private]
```

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

- [BCondInt.h](#)
- [BCondInt.cpp](#)

## 7.10 BCondResource Class Reference

Resource lock.

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

### Public Member Functions

- [BCondResource](#) ()
- [~BCondResource](#) ()
- int [lock](#) (uint32\_t timeOutUs=0)  
*Lock the resource, will wait for all usage to be 0.*
- int [unlock](#) ()  
*Unlock the resource.*
- int [start](#) (uint32\_t timeOutUs=0)  
*Start using the resource.*
- int [end](#) ()  
*Finish using the resource.*
- int [locked](#) ()
- int [inUse](#) ()

### Private Attributes

- pthread\_mutex\_t [omutex](#)
- pthread\_cond\_t [ocond](#)
- int [olock](#)
- int [ouse](#)

### 7.10.1 Detailed Description

Resource lock.

### 7.10.2 Constructor & Destructor Documentation

#### 7.10.2.1 BCondResource()

```
BCondResource::BCondResource ( )
```

#### 7.10.2.2 ~BCondResource()

```
BCondResource::~BCondResource ( )
```

### 7.10.3 Member Function Documentation

#### 7.10.3.1 end()

```
int BCondResource::end ( )
```

Finish using the resource.

#### 7.10.3.2 inUse()

```
int BCondResource::inUse ( )
```

#### 7.10.3.3 lock()

```
int BCondResource::lock (
    uint32_t timeoutUs = 0 )
```

Lock the resource, will wait for all usage to be 0.

#### 7.10.3.4 locked()

```
int BCondResource::locked ( )
```

#### 7.10.3.5 start()

```
int BCondResource::start (
    uint32_t timeoutUs = 0 )
```

Start using the resource.

#### 7.10.3.6 unlock()

```
int BCondResource::unlock ( )
```

Unlock the resource.

## 7.10.4 Member Data Documentation

### 7.10.4.1 ocond

```
pthread_cond_t BCondResource::ocond [private]
```

### 7.10.4.2 olock

```
int BCondResource::olock [private]
```

### 7.10.4.3 omutex

```
pthread_mutex_t BCondResource::omutex [private]
```

### 7.10.4.4 ouse

```
int BCondResource::ouse [private]
```

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

- [BCondInt.h](#)
- [BCondInt.cpp](#)

## 7.11 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)
- void [operator--](#) (int)  
*Decrement value. Wakes waiting.*

## Private Attributes

- pthread\_mutex\_t [omutex](#)
- pthread\_cond\_t [ocond](#)
- int [ovalue](#)

### 7.11.1 Detailed Description

Thread conditional value.

### 7.11.2 Constructor & Destructor Documentation

#### 7.11.2.1 BCondValue()

```
BCondValue::BCondValue ( )
```

### 7.11.2.2 ~BCondValue()

```
BCondValue::~~BCondValue ( )
```

## 7.11.3 Member Function Documentation

### 7.11.3.1 decrement()

```
int BCondValue::decrement (
    int v = 1 )
```

Decrement. Wakes waiting.

### 7.11.3.2 increment()

```
int BCondValue::increment (
    int v = 1 )
```

Increment. Wakes waiting.

### 7.11.3.3 operator++()

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

Increment value. Wakes waiting.

### 7.11.3.4 operator+=()

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

Add to value. Wakes waiting.



### 7.11.3.5 operator--()

```
void BCondValue::operator-- (
    int ) [inline]
```

Decrement value. Wakes waiting.

### 7.11.3.6 operator-=()

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

Subtract from value. Wakes waiting.

### 7.11.3.7 setValue()

```
void BCondValue::setValue (
    int value )
```

Set the value. Wakes waiting.

### 7.11.3.8 value()

```
int BCondValue::value ( )
```

Current value.

### 7.11.3.9 waitLessThan()

```
int BCondValue::waitLessThan (
    int v,
    int timeoutUs = 0 )
```

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

#### 7.11.3.10 waitLessThanOrEqual()

```
int BCondValue::waitLessThanOrEqual (
    int v,
    int increment = 0,
    int timeoutUs = 0 )
```

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

#### 7.11.3.11 waitMoreThanOrEqual()

```
int BCondValue::waitMoreThanOrEqual (
    int v,
    int decrement = 0,
    int timeoutUs = 0 )
```

Wait until value is at least the value given.

### 7.11.4 Member Data Documentation

#### 7.11.4.1 ocond

```
pthread_cond_t BCondValue::ocond [private]
```

#### 7.11.4.2 omutex

```
pthread_mutex_t BCondValue::omutex [private]
```

#### 7.11.4.3 ovalue

```
int BCondValue::ovalue [private]
```

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

- [BCondInt.h](#)
- [BCondInt.cpp](#)

## 7.12 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](#)

#### 7.12.1 Constructor & Destructor Documentation

### 7.12.1.1 BCondWrap()

```
BCondWrap::BCondWrap ( )
```

### 7.12.1.2 ~BCondWrap()

```
BCondWrap::~~BCondWrap ( )
```

## 7.12.2 Member Function Documentation

### 7.12.2.1 decrement()

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

Decrement. Wakes waiting.

### 7.12.2.2 diff()

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

### 7.12.2.3 increment()

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

Increment. Wakes waiting.

### 7.12.2.4 operator++()

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

Increment value. Wakes waiting.

#### 7.12.2.5 operator+=( )

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

Add to value. Wakes waiting.

#### 7.12.2.6 operator--( )

```
void BCondWrap::operator-- (
    int ) [inline]
```

Decrement value. Wakes waiting.

#### 7.12.2.7 operator-=( )

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

Subtract from value. Wakes waiting.

#### 7.12.2.8 setValue( )

```
void BCondWrap::setValue (
    uint32_t value )
```

Set the value. Wakes waiting.

#### 7.12.2.9 value( )

```
uint32_t BCondWrap::value ( )
```

Current value.

#### 7.12.2.10 waitLessThan( )

```
int BCondWrap::waitLessThan (
    uint32_t v,
    uint32_t timeoutUs = 0 )
```

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

#### 7.12.2.11 waitLessThanOrEqual()

```
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.

#### 7.12.2.12 waitMoreThanOrEqual()

```
int BCondWrap::waitMoreThanOrEqual (
    uint32_t v,
    uint32_t decrement = 0,
    uint32_t timeoutUs = 0 )
```

Wait until value is at least the value given.

### 7.12.3 Member Data Documentation

#### 7.12.3.1 ocond

```
pthread_cond_t BCondWrap::ocond [private]
```

#### 7.12.3.2 omutex

```
pthread_mutex_t BCondWrap::omutex [private]
```

#### 7.12.3.3 ovalue

```
uint32_t BCondWrap::ovalue [private]
```

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

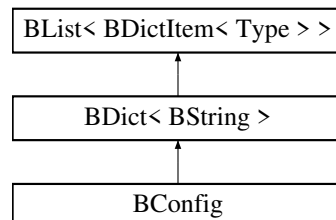
- [BCondInt.h](#)
- [BCondInt.cpp](#)

## 7.13 BConfig Class Reference

This class implements the configuration file access.

```
#include <BConfig.h>
```

Inheritance diagram for BConfig:



### Public Member Functions

- [BError open](#) (BString fileName, BString mode="r")
- [void close](#) ()
- [BError read](#) ()
- [BError write](#) ()
- [BString findValue](#) (BString name)
- [BString fileName](#) ()

### Private Attributes

- [BMutex olock](#)
- [BString ofileName](#)
- [BFile ofile](#)

### Additional Inherited Members

#### 7.13.1 Detailed Description

This class implements the configuration file access.

#### 7.13.2 Member Function Documentation

##### 7.13.2.1 close()

```
void BConfig::close ( )
```

#### 7.13.2.2 fileName()

```
BString BConfig::fileName ( )
```

#### 7.13.2.3 findValue()

```
BString BConfig::findValue (
    BString name )
```

#### 7.13.2.4 open()

```
BError BConfig::open (
    BString fileName,
    BString mode = "r" )
```

#### 7.13.2.5 read()

```
BError BConfig::read ( )
```

#### 7.13.2.6 write()

```
BError BConfig::write ( )
```

### 7.13.3 Member Data Documentation

#### 7.13.3.1 ofile

```
BFile BConfig::ofile [private]
```

#### 7.13.3.2 ofileName

```
BString BConfig::ofileName [private]
```



### 7.13.3.3 olock

```
BMutex BConfig::olock [private]
```

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

- [BConfig.h](#)
- [BConfig.cpp](#)

## 7.14 BDataChunk Class Reference

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

### Public Member Functions

- [BDataChunk](#) (void \*data=0, BUInt size=0)

### Public Attributes

- void \* [data](#)
- [BUInt](#) [size](#)

## 7.14.1 Constructor & Destructor Documentation

### 7.14.1.1 BDataChunk()

```
BDataChunk::BDataChunk (  
    void * data = 0,  
    BUInt size = 0 ) [inline]
```

## 7.14.2 Member Data Documentation

### 7.14.2.1 data

```
void* BDataChunk::data
```

### 7.14.2.2 size

`BUInt BDataChunk::size`

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

- [BTypes.h](#)

## 7.15 BDate Class Reference

```
#include <BDate.h>
```

### Public Member Functions

- [BDate](#) (int year=0, int month=1, int day=1)
- [BDate](#) (BString str)
- [~BDate](#) ()
- void [clear](#) ()  
*Clear the date/time.*
- void [setFirst](#) ()  
*Set the first date available.*
- void [setLast](#) ()  
*Set the last date available.*
- void [set](#) (time\_t time)  
*Set time using Unix time (seconds from 1970-01-01)*
- void [set](#) (int year=0, int month=1, int day=1)
- void [setYDay](#) (int year=0, int yday=0)
- void [setNow](#) ()  
*Set the timeStamp to now.*
- int [year](#) ()
- int [yday](#) ()
- int [month](#) ()
- int [day](#) ()
- void [getDate](#) (int &year, int &mon, int &day)
- [BString](#) [getString](#) ()  
*Get the time as an ISO date/time string.*
- [BString](#) [getStringFormatted](#) ([BString](#) format)  
*Gets the time in a string form as per the format. Format syntax as per strftime()*
- [BError](#) [setString](#) ([BString](#) str)  
*Set the time from an ISO date/time.*
- int [isSet](#) ()
- int [compare](#) (const [BDate](#) &date) const  
*Compare two dates.*
- [operator](#) [BString](#) ()
- int [operator==](#) (const [BDate](#) &date) const
- int [operator!=](#) (const [BDate](#) &date) const
- int [operator>](#) (const [BDate](#) &date) const
- int [operator>=](#) (const [BDate](#) &date) const
- int [operator<](#) (const [BDate](#) &date) const
- int [operator<=](#) (const [BDate](#) &date) const

## Static Public Member Functions

- static int [isLeap](#) (int [year](#))
- static int [daysInMonth](#) (int [year](#), int [month](#))

## Public Attributes

- uint16\_t [oyear](#)  
*Year (0 .. 65535)*
- uint16\_t [oyday](#)  
*Day in year (0 .. 365)*

## 7.15.1 Constructor & Destructor Documentation

### 7.15.1.1 BDate() [1/2]

```
BDate::BDate (
    int year = 0,
    int month = 1,
    int day = 1 )
```

### 7.15.1.2 BDate() [2/2]

```
BDate::BDate (
    BString str )
```

### 7.15.1.3 ~BDate()

```
BDate::~~BDate ( )
```

## 7.15.2 Member Function Documentation

### 7.15.2.1 clear()

```
void BDate::clear ( )
```

Clear the date/time.

### 7.15.2.2 compare()

```
int BDate::compare (
    const BDate & date ) const
```

Compare two dates.

### 7.15.2.3 day()

```
int BDate::day ( )
```

### 7.15.2.4 daysInMonth()

```
int BDate::daysInMonth (
    int year,
    int month ) [static]
```

### 7.15.2.5 getDate()

```
void BDate::getDate (
    int & year,
    int & mon,
    int & day )
```

### 7.15.2.6 getString()

```
BString BDate::getString ( )
```

Get the time as an ISO date/time string.

### 7.15.2.7 getStringFormatted()

```
BString BDate::getStringFormatted (
    BString format )
```

Gets the time in a string form as per the format. Format syntax as per strftime()

### 7.15.2.8 isLeap()

```
int BDate::isLeap (
    int year ) [static]
```

### 7.15.2.9 isSet()

```
int BDate::isSet ( ) [inline]
```

### 7.15.2.10 month()

```
int BDate::month ( )
```

### 7.15.2.11 operator BString()

```
BDate::operator BString ( ) [inline]
```

### 7.15.2.12 operator!=(=)

```
int BDate::operator!=(= (
    const BDate & date ) const [inline]
```

### 7.15.2.13 operator<()

```
int BDate::operator< (
    const BDate & date ) const [inline]
```

### 7.15.2.14 operator<=()

```
int BDate::operator<= (
    const BDate & date ) const [inline]
```

#### 7.15.2.15 operator==( )

```
int BDate::operator==(
    const BDate & date ) const [inline]
```

#### 7.15.2.16 operator>( )

```
int BDate::operator>(
    const BDate & date ) const [inline]
```

#### 7.15.2.17 operator>=( )

```
int BDate::operator>=(
    const BDate & date ) const [inline]
```

#### 7.15.2.18 set( ) [1/2]

```
void BDate::set (
    time_t time )
```

Set time using Unix time (seconds from 1970-01-01)

#### 7.15.2.19 set( ) [2/2]

```
void BDate::set (
    int year = 0,
    int month = 1,
    int day = 1 )
```

#### 7.15.2.20 setFirst( )

```
void BDate::setFirst ( )
```

Set the first date available.

#### 7.15.2.21 setLast()

```
void BDate::setLast ( )
```

Set the last date available.

#### 7.15.2.22 setNow()

```
void BDate::setNow ( )
```

Set the timeStamp to now.

#### 7.15.2.23 setString()

```
BError BDate::setString (
    BString str )
```

Set the time from an ISO date/time.

#### 7.15.2.24 setYDay()

```
void BDate::setYDay (
    int year = 0,
    int yday = 0 )
```

#### 7.15.2.25 yday()

```
int BDate::yday ( )
```

#### 7.15.2.26 year()

```
int BDate::year ( )
```

### 7.15.3 Member Data Documentation

### 7.15.3.1 oyday

```
uint16_t BDate::oyday
```

Day in year (0 .. 365)

### 7.15.3.2 oyear

```
uint16_t BDate::oyear
```

Year (0 .. 65535)

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

- [BDate.h](#)
- [BDate-1.cpp](#)
- [BDate.cpp](#)

## 7.16 BDebugBacktrace Class Reference

```
#include <BDebug.h>
```

### Public Member Functions

- [BDebugBacktrace](#) ()
- [~BDebugBacktrace](#) ()
- void [dumpBacktraceStdout](#) (char \*comment)
- int [dumpBacktraceFile](#) (char \*fileName, char \*comment)
- void [dumpBacktraceSyslog](#) (char \*comment)
- void [dumpBacktrace](#) (char \*strBuf, int strBufLen, char \*comment)

### 7.16.1 Constructor & Destructor Documentation

#### 7.16.1.1 BDebugBacktrace()

```
BDebugBacktrace::BDebugBacktrace ( )
```

#### 7.16.1.2 ~BDebugBacktrace()

```
BDebugBacktrace::~BDebugBacktrace ( )
```



## 7.16.2 Member Function Documentation

### 7.16.2.1 dumpBacktrace()

```
void BDebugBacktrace::dumpBacktrace (
    char * strBuf,
    int strBufLen,
    char * comment )
```

### 7.16.2.2 dumpBacktraceFile()

```
int BDebugBacktrace::dumpBacktraceFile (
    char * fileName,
    char * comment )
```

### 7.16.2.3 dumpBacktraceStdout()

```
void BDebugBacktrace::dumpBacktraceStdout (
    char * comment )
```

### 7.16.2.4 dumpBacktraceSyslog()

```
void BDebugBacktrace::dumpBacktraceSyslog (
    char * comment )
```

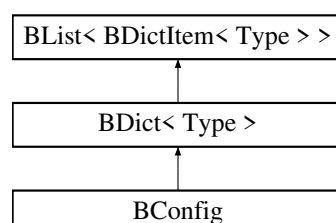
The documentation for this class was generated from the following file:

- [BDebug.h](#)

## 7.17 BDict< Type > Class Template Reference

```
#include <BDict.h>
```

Inheritance diagram for BDict< Type >:



## Public Types

- typedef [BIter](#) iterator

## Public Member Functions

- [BDict](#) (int hashSize=100)
- [BDict](#) (const [BDict](#)< Type > &dict)
- int [hasKey](#) (const [BString](#) &k) const
- [BString](#) [key](#) (const [BIter](#) &i) const
- void [clear](#) ()
  - Clear the list.*
- void [insert](#) ([BIter](#) &i, const [BDictItem](#)< Type > &item)
  - Insert item before item.*
- void [append](#) (const [BDictItem](#)< Type > &item)
- void [append](#) (const [BDict](#)< Type > &dict)
- void [del](#) (const [BString](#) &k)
- void [del](#) ([BIter](#) &i)
  - Delete specified item.*
- [BIter](#) [find](#) (const [BString](#) &k) const
- Type & [operator\[\]](#) (const [BString](#) &i)
- const Type & [operator\[\]](#) (const [BString](#) &i) const
- Type & [operator\[\]](#) (const [BIter](#) &i)
- const Type & [operator\[\]](#) (const [BIter](#) &i) const
- [BDict](#)< Type > [operator+](#) (const [BDict](#)< Type > &dict) const
- [BDict](#)< Type > & [operator=](#) (const [BDict](#)< Type > &dict)
- void [hashPrint](#) ()

## Private Member Functions

- void [hashAdd](#) (const [BString](#) &k, [BIter](#) iter)
- void [hashDelete](#) (const [BString](#) &k, [BIter](#) iter)
- int [hashFind](#) (const [BString](#) &k, [BIter](#) &iter) const

## Private Attributes

- int [oHashSize](#)
- [BArray](#)< [BList](#)< [BIter](#) > > [oHashLists](#)

## Additional Inherited Members

### 7.17.1 Member Typedef Documentation

#### 7.17.1.1 iterator

```
template<class Type >
typedef BIter BDict< Type >::iterator
```

## 7.17.2 Constructor & Destructor Documentation

### 7.17.2.1 BDict() [1/2]

```
template<class Type >
BDict< Type >::BDict (
    int hashSize = 100 )
```

### 7.17.2.2 BDict() [2/2]

```
template<class Type >
BDict< Type >::BDict (
    const BDict< Type > & dict )
```

## 7.17.3 Member Function Documentation

### 7.17.3.1 append() [1/2]

```
template<class Type >
void BDict< Type >::append (
    const BDictItem< Type > & item )
```

### 7.17.3.2 append() [2/2]

```
template<class Type >
void BDict< Type >::append (
    const BDict< Type > & dict )
```

### 7.17.3.3 clear()

```
template<class Type >
void BDict< Type >::clear ( ) [virtual]
```

Clear the list.

Reimplemented from [BList< BDictItem< Type > >](#).

#### 7.17.3.4 del() [1/2]

```
template<class Type >
void BDict< Type >::del (
    const BString & k )
```

#### 7.17.3.5 del() [2/2]

```
template<class Type >
void BDict< Type >::del (
    BIter & i ) [virtual]
```

Delete specified item.

Reimplemented from [BList< BDictItem< Type > >](#).

#### 7.17.3.6 find()

```
template<class Type >
BIter BDict< Type >::find (
    const BString & k ) const
```

#### 7.17.3.7 hashAdd()

```
template<class Type >
void BDict< Type >::hashAdd (
    const BString & k,
    BIter iter ) [private]
```

#### 7.17.3.8 hashDelete()

```
template<class Type >
void BDict< Type >::hashDelete (
    const BString & k,
    BIter iter ) [private]
```

### 7.17.3.9 hashFind()

```
template<class Type >
int BDict< Type >::hashFind (
    const BString & k,
    BIter & iter ) const [private]
```

### 7.17.3.10 hashPrint()

```
template<class Type >
void BDict< Type >::hashPrint ( )
```

### 7.17.3.11 hasKey()

```
template<class Type >
int BDict< Type >::hasKey (
    const BString & k ) const
```

### 7.17.3.12 insert()

```
template<class Type >
void BDict< Type >::insert (
    BIter & i,
    const BDictItem< Type > & item ) [virtual]
```

Insert item before item.

Reimplemented from [BList< BDictItem< Type > >](#).

### 7.17.3.13 key()

```
template<class Type >
BString BDict< Type >::key (
    const BIter & i ) const
```

### 7.17.3.14 operator+()

```
template<class Type >
BDict< Type > BDict< Type >::operator+ (
    const BDict< Type > & dict ) const
```

**7.17.3.15 operator=()**

```
template<class Type >
BDict< Type > & BDict< Type >::operator= (
    const BDict< Type > & dict )
```

**7.17.3.16 operator[]** [1/4]

```
template<class Type >
Type & BDict< Type >::operator[] (
    const BString & i )
```

**7.17.3.17 operator[]()** [2/4]

```
template<class Type >
const Type & BDict< Type >::operator[] (
    const BString & i ) const
```

**7.17.3.18 operator[]()** [3/4]

```
template<class Type >
Type & BDict< Type >::operator[] (
    const BIter & i )
```

**7.17.3.19 operator[]()** [4/4]

```
template<class Type >
const Type & BDict< Type >::operator[] (
    const BIter & i ) const
```

**7.17.4 Member Data Documentation****7.17.4.1 ohashLists**

```
template<class Type >
BArray<BList<BIter> > BDict< Type >::ohashLists [private]
```

#### 7.17.4.2 ohashSize

```
template<class Type >
int BDict< Type >::ohashSize [private]
```

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

- [BDict.h](#)

## 7.18 BDictItem< Type > Class Template Reference

Template based Dictionary class.

```
#include <BDict.h>
```

### Public Member Functions

- [BDictItem](#) (BString k="", Type v=Type())

### Public Attributes

- [BString](#) key
- [Type](#) value

### 7.18.1 Detailed Description

```
template<class Type>
class BDictItem< Type >
```

Template based Dictionary class.

### 7.18.2 Constructor & Destructor Documentation

#### 7.18.2.1 BDictItem()

```
template<class Type >
BDictItem< Type >::BDictItem (
    BString k = "",
    Type v = Type() ) [inline]
```

### 7.18.3 Member Data Documentation

### 7.18.3.1 key

```
template<class Type >
BString BDictItem< Type >::key
```

### 7.18.3.2 value

```
template<class Type >
Type BDictItem< Type >::value
```

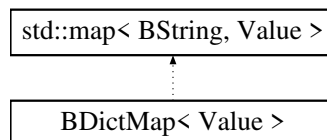
The documentation for this class was generated from the following file:

- [BDict.h](#)

## 7.19 BDictMap< Value > Class Template Reference

```
#include <BDictMap.h>
```

Inheritance diagram for BDictMap< Value >:



### Public Types

- typedef [BDictMap< Value >::iterator](#) iterator

### Public Member Functions

- void [clear](#) ()
- int [hasKey](#) (const [BString](#) &k)
- [BString](#) [key](#) (iterator &i)
- unsigned int [size](#) ()
- void [start](#) (iterator &i)
- int [isEnd](#) (iterator &i)
- void [next](#) (iterator &i)
- void [del](#) (const iterator &i)
- void [del](#) (const [BString](#) &k)
- Value & [operator\[\]](#) (iterator &i)
- Value & [operator\[\]](#) (const [BString](#) &i)



### 7.19.1 Detailed Description

```
template<typename Value>  
class BDictMap< Value >
```

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

### 7.19.2 Member Typedef Documentation

#### 7.19.2.1 iterator

```
template<typename Value >  
typedef BDictMap<Value>::iterator BDictMap< Value >::iterator
```

### 7.19.3 Member Function Documentation

#### 7.19.3.1 clear()

```
template<typename Value >  
void BDictMap< Value >::clear ( ) [inline]
```

#### 7.19.3.2 del() [1/2]

```
template<typename Value >  
void BDictMap< Value >::del (   
    const iterator & i ) [inline]
```

#### 7.19.3.3 del() [2/2]

```
template<typename Value >  
void BDictMap< Value >::del (   
    const BString & k ) [inline]
```

#### 7.19.3.4 hasKey()

```
template<typename Value >
int BDictMap< Value >::hasKey (
    const BString & k ) [inline]
```

#### 7.19.3.5 isEnd()

```
template<typename Value >
int BDictMap< Value >::isEnd (
    iterator & i ) [inline]
```

#### 7.19.3.6 key()

```
template<typename Value >
BString BDictMap< Value >::key (
    iterator & i ) [inline]
```

#### 7.19.3.7 next()

```
template<typename Value >
void BDictMap< Value >::next (
    iterator & i ) [inline]
```

#### 7.19.3.8 operator[]() [1/2]

```
template<typename Value >
Value& BDictMap< Value >::operator[] (
    iterator & i ) [inline]
```

#### 7.19.3.9 operator[]() [2/2]

```
template<typename Value >
Value& BDictMap< Value >::operator[] (
    const BString & i ) [inline]
```

## 7.19.3.10 size()

```
template<typename Value >
unsigned int BDictMap< Value >::size ( ) [inline]
```

## 7.19.3.11 start()

```
template<typename Value >
void BDictMap< Value >::start (
    iterator & i ) [inline]
```

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

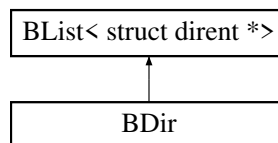
- [BDictMap.h](#)

## 7.20 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](#) (Blter i)
  - Get filename.*
- struct stat [entryStat](#) (Blter i)
  - Get file stats.*
- struct stat64 [entryStat64](#) (Blter i)
  - Get file stats 64.*

## Private Attributes

- [BError oerror](#)
- [BString odirname](#)
- [BString owild](#)
- [int osort](#)

## Additional Inherited Members

### 7.20.1 Detailed Description

File system directory class.

### 7.20.2 Constructor & Destructor Documentation

#### 7.20.2.1 BDir() [1/2]

```
BDir::BDir ( )
```

#### 7.20.2.2 BDir() [2/2]

```
BDir::BDir (
    BString name )
```

#### 7.20.2.3 ~BDir()

```
BDir::~~BDir ( )
```

### 7.20.3 Member Function Documentation

#### 7.20.3.1 clear()

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

Clears list.

Reimplemented from [BList< struct dirent \\*>](#).

### 7.20.3.2 entryName()

```
BString BDir::entryName (
    BIter i )
```

Get filename.

### 7.20.3.3 entryStat()

```
struct stat BDir::entryStat (
    BIter i )
```

Get file stats.

### 7.20.3.4 entryStat64()

```
struct stat64 BDir::entryStat64 (
    BIter i )
```

Get file stats 64.

### 7.20.3.5 error()

```
BError BDir::error ( )
```

Current value of error.

### 7.20.3.6 open()

```
BError BDir::open (
    BString name )
```

Reads named directory.

### 7.20.3.7 read()

```
BError BDir::read ( )
```

read/re-reads directory

#### 7.20.3.8 setSort()

```
void BDir::setSort (
    int on )
```

Set alpha sort on/off.

#### 7.20.3.9 setWild()

```
void BDir::setWild (
    BString wild )
```

Set wildcard filter string used on read.

### 7.20.4 Member Data Documentation

#### 7.20.4.1 odirname

```
BString BDir::odirname [private]
```

#### 7.20.4.2 oerror

```
BError BDir::oerror [private]
```

#### 7.20.4.3 osort

```
int BDir::osort [private]
```

#### 7.20.4.4 owild

```
BString BDir::owild [private]
```

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

- [BDir.h](#)
- [BDir.cpp](#)

## 7.21 BDuration Class Reference

```
#include <BDuration.h>
```

### Public Member Functions

- [BDuration](#) (int hour=0, int minute=0, int second=0, int microsecond=0)
- [BDuration](#) (BString str)
- [~BDuration](#) ()
- void [clear](#) ()
  - Clear the duration.*
- void [set](#) (int hour=0, int minute=0, int second=0, int microsecond=0)
- void [addMilliSeconds](#) (int64\_t milliSeconds)
  - Add the given number of milli seconds.*
- void [addMicroSeconds](#) (int64\_t microSeconds)
  - Add the given number of micro seconds.*
- void [addSeconds](#) (int seconds)
  - Add the given number of seconds.*
- uint32\_t [getSeconds](#) ()
  - Get number of seconds.*
- uint64\_t [getMicroSeconds](#) ()
  - Get number of micro seconds.*
- int [hour](#) ()
- int [minute](#) ()
- int [second](#) ()
- int [microSecond](#) ()
- BString [getString](#) ()
  - Get the time as an ISO date/time string.*
- BError [setString](#) (BString time)
  - Set the time from an ISO date/time.*

### Private Attributes

- uint8\_t [ohour](#)
  - Hour (0 .. 23)*
- uint8\_t [ominute](#)
  - Minute (0 .. 59)*
- uint8\_t [osecond](#)
  - Second (0 .. 59)*
- uint8\_t [ospare](#)
  - Padding.*
- uint32\_t [omicroSecond](#)
  - MicroSecond (0 .. 999999)*

#### 7.21.1 Constructor & Destructor Documentation

### 7.21.1.1 BDuration() [1/2]

```
BDuration::BDuration (
    int hour = 0,
    int minute = 0,
    int second = 0,
    int microsecond = 0 )
```

### 7.21.1.2 BDuration() [2/2]

```
BDuration::BDuration (
    BString str )
```

### 7.21.1.3 ~BDuration()

```
BDuration::~~BDuration ( )
```

## 7.21.2 Member Function Documentation

### 7.21.2.1 addMicroSeconds()

```
void BDuration::addMicroSeconds (
    int64_t microSeconds )
```

Add the given number of micro seconds.

### 7.21.2.2 addMilliSeconds()

```
void BDuration::addMilliSeconds (
    int64_t milliSeconds )
```

Add the given number of milli seconds.

### 7.21.2.3 addSeconds()

```
void BDuration::addSeconds (
    int seconds )
```

Add the given number of seconds.



#### 7.21.2.4 clear()

```
void BDuration::clear ( )
```

Clear the duration.

#### 7.21.2.5 getMicroSeconds()

```
uint64_t BDuration::getMicroSeconds ( )
```

Get number of micro seconds.

#### 7.21.2.6 getSeconds()

```
uint32_t BDuration::getSeconds ( )
```

Get number of seconds.

#### 7.21.2.7 getString()

```
BString BDuration::getString ( )
```

Get the time as an ISO date/time string.

#### 7.21.2.8 hour()

```
int BDuration::hour ( )
```

#### 7.21.2.9 microSecond()

```
int BDuration::microSecond ( )
```

#### 7.21.2.10 minute()

```
int BDuration::minute ( )
```

### 7.21.2.11 second()

```
int BDuration::second ( )
```

### 7.21.2.12 set()

```
void BDuration::set (
    int hour = 0,
    int minute = 0,
    int second = 0,
    int microsecond = 0 )
```

### 7.21.2.13 setString()

```
BError BDuration::setString (
    BString time )
```

Set the time from an ISO date/time.

## 7.21.3 Member Data Documentation

### 7.21.3.1 ohour

```
uint8_t BDuration::ohour [private]
```

Hour (0 .. 23)

### 7.21.3.2 omicroSecond

```
uint32_t BDuration::omicroSecond [private]
```

MicroSecond (0 .. 999999)

### 7.21.3.3 ominute

```
uint8_t BDuration::omminute [private]
```

Minute (0 .. 59)

#### 7.21.3.4 osecond

```
uint8_t BDuration::osecond [private]
```

Second (0 .. 59)

#### 7.21.3.5 ospare

```
uint8_t BDuration::ospare [private]
```

Padding.

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

- [BDuration.h](#)
- [BDuration.cpp](#)

## 7.22 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 delimited string.*
- [BString getName \(\)](#)  
*Get the name.*
- [BString getValue \(\)](#)  
*Get the value.*
- void [setLine \(BString line\)](#)  
*Set name and value from white space delimited 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](#)

### 7.22.1 Detailed Description

Manipulate a name value pair.

### 7.22.2 Constructor & Destructor Documentation

#### 7.22.2.1 BEntry() [1/3]

```
BEntry::BEntry ( )
```

#### 7.22.2.2 BEntry() [2/3]

```
BEntry::BEntry (
    BString name,
    BString value )
```

Set name and value.

#### 7.22.2.3 BEntry() [3/3]

```
BEntry::BEntry (
    BString line )
```

Set name and value from white space delimited string.

### 7.22.3 Member Function Documentation

#### 7.22.3.1 getName()

```
BString BEntry::getName ( )
```

Get the name.

### 7.22.3.2 getValue()

```
BString BEntry::getValue ( )
```

Get the value.

### 7.22.3.3 line()

```
BString BEntry::line ( )
```

Return name and value as padded single string.

### 7.22.3.4 print()

```
void BEntry::print ( )
```

Print name and value.

### 7.22.3.5 setLine()

```
void BEntry::setLine (
    BString line )
```

Set name and value from white space delimited string.

### 7.22.3.6 setName()

```
void BEntry::setName (
    BString name )
```

Set the name.

### 7.22.3.7 setValue()

```
void BEntry::setValue (
    BString value )
```

Set the value.

## 7.22.4 Member Data Documentation

### 7.22.4.1 oname

```
BString BEntry::oname [private]
```

### 7.22.4.2 ovalue

```
BString BEntry::ovalue [private]
```

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

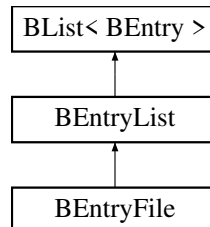
- [BEntry.h](#)
- [BEntry.cpp](#)

## 7.23 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.*
- [BString filename \(\)](#)  
*Returns the filename.*

## Private Attributes

- [BString ofilename](#)
- [BString ocomments](#)

## Additional Inherited Members

### 7.23.1 Detailed Description

File of Entries.

### 7.23.2 Constructor & Destructor Documentation

#### 7.23.2.1 BEntryFile() [1/2]

```
BEntryFile::BEntryFile ( )
```

#### 7.23.2.2 BEntryFile() [2/2]

```
BEntryFile::BEntryFile (
    BString filename )
```

Opens entryfile.

#### 7.23.2.3 ~BEntryFile()

```
BEntryFile::~BEntryFile ( )
```

### 7.23.3 Member Function Documentation

#### 7.23.3.1 clear()

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

Clears current list.

Reimplemented from [BEntryList](#).

### 7.23.3.2 filename()

```
BString BEntryFile::filename ( )
```

Returns the filename.

### 7.23.3.3 open()

```
int BEntryFile::open (
    BString filename )
```

Opens entryfile.

### 7.23.3.4 read()

```
int BEntryFile::read ( )
```

Reads entry file and builds list.

### 7.23.3.5 write()

```
int BEntryFile::write ( )
```

Writes list to entryfile.

### 7.23.3.6 writeList()

```
int BEntryFile::writeList (
    BEntryList & l )
```

Writes specified list to file.

## 7.23.4 Member Data Documentation

### 7.23.4.1 ocomments

```
BString BEntryFile::ocomments [private]
```



## 7.23.4.2 ofilename

```
BString BEntryFile::ofilename [private]
```

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

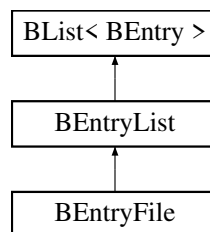
- [BEntry.h](#)
- [BEntry.cpp](#)

## 7.24 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.*
- [BEntryList & operator= \(const BEntryList &l\)](#)

## Private Attributes

- [BIter olastPos](#)

## Additional Inherited Members

### 7.24.1 Detailed Description

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

### 7.24.2 Constructor & Destructor Documentation

#### 7.24.2.1 BEntryList()

```
BEntryList::BEntryList ( )
```

### 7.24.3 Member Function Documentation

#### 7.24.3.1 clear()

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

Clear the list.

Reimplemented from [BList< BEntry >](#).

Reimplemented in [BEntryFile](#).

#### 7.24.3.2 del()

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

Delete specified item.

Reimplemented from [BList< BEntry >](#).

### 7.24.3.3 deleteEntry()

```
void BEntryList::deleteEntry (
    BString name )
```

Deletes the entry.

### 7.24.3.4 find()

```
BEntry * BEntryList::find (
    BString name )
```

Returns entry if name is found otherwise NULL.

### 7.24.3.5 findValue()

```
BString BEntryList::findValue (
    BString name )
```

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

### 7.24.3.6 getString()

```
BString BEntryList::getString ( )
```

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

### 7.24.3.7 insert()

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

Insert item before item.

Reimplemented from [BList< BEntry >](#).

#### 7.24.3.8 isSet()

```
int BEntryList::isSet (
    BString name )
```

1 if name is in list and value is set

#### 7.24.3.9 operator=()

```
BEntryList & BEntryList::operator= (
    const BEntryList & l )
```

#### 7.24.3.10 print()

```
void BEntryList::print ( )
```

Print list.

#### 7.24.3.11 setValue()

```
int BEntryList::setValue (
    BString name,
    BString value )
```

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

#### 7.24.3.12 setValueRaw()

```
int BEntryList::setValueRaw (
    BString name,
    BString value )
```

Raw setting of value without looking up existing entry.

### 7.24.4 Member Data Documentation

## 7.24.4.1 olastPos

```
BIter BEntryList::olastPos [private]
```

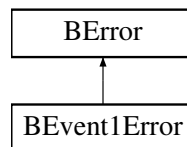
The documentation for this class was generated from the following files:

- [BEntry.h](#)
- [BEntry.cpp](#)

## 7.25 BError Class Reference

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

Inheritance diagram for BError:



## Public Member Functions

- [BError](#) (int errNo=[ErrorOk](#), [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 & clear](#) ()  
*Clear the error.*
- [BError & setError](#) ([BString](#) errStr="")  
*Set error type ERROR with optional message.*
- [BString getString](#) () const  
*Get error message.*
- int [getNumber](#) () const  
*Get The error number.*
- int [num](#) () const  
*Get The error number.*
- const char \* [str](#) () const  
*Return a char\* string.*
- int [getErrorNo](#) () const  
*Get The error number.*
- [operator int](#) () const  
*Return error number.*

## Private Attributes

- `int oerrNo`
- `BString oerrStr`

### 7.25.1 Detailed Description

Error return class. This class is used to return the error status from a function. It encapsulates an integer error number and a string. An error number of `ErrorOk` (0) indicates no error, a value of `ErrorMisc` (1) indicates some error and a value of `ErrorWarning` (2) indicates a warning. Specific error numbers are defined in `BErrorNum`. System low level errors (`errno`) are defined by negative values. Specific application errors are those above the value 64.

### 7.25.2 Constructor & Destructor Documentation

#### 7.25.2.1 `BError()` [1/2]

```
BError::BError (
    int errNo = ErrorOk,
    BString errStr = "" )
```

Create object.

#### 7.25.2.2 `BError()` [2/2]

```
BError::BError (
    BString errStr )
```

Create with error set and error string.

### 7.25.3 Member Function Documentation

#### 7.25.3.1 `clear()`

```
BError & BError::clear ( )
```

Clear the error.

### 7.25.3.2 copy()

```
BError BError::copy ( )
```

Return an independant copy.

### 7.25.3.3 getErrorNo()

```
int BError::getErrorNo ( ) const
```

Get The error number.

### 7.25.3.4 getNumber()

```
int BError::getNumber ( ) const
```

Get The error number.

### 7.25.3.5 getString()

```
BString BError::getString ( ) const
```

Get error message.

### 7.25.3.6 num()

```
int BError::num ( ) const
```

Get The error number.

### 7.25.3.7 operator int()

```
BError::operator int ( ) const [inline]
```

Return error number.

### 7.25.3.8 set()

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

Set error number and message.

### 7.25.3.9 setError()

```
BError & BError::setError (
    BString errStr = "" )
```

Set error type ERROR with optional message.

### 7.25.3.10 str()

```
const char * BError::str ( ) const
```

Return a char\* string.

## 7.25.4 Member Data Documentation

### 7.25.4.1 oerrNo

```
int BError::oerrNo [private]
```

### 7.25.4.2 oerrStr

```
BString BError::oerrStr [private]
```

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

- [BError.h](#)
- [BError.cpp](#)



## 7.26 BErrorTime Class Reference

Error return class.

```
#include <BErrorTime.h>
```

### Public Types

- enum `Type` { `None` = 0, `Error` = 1 }

### Public Member Functions

- `BErrorTime` (int `errNo`=None, `BTimeStamp` `errTime`=`BTimeStamp`(), `BString` `errStr`="")  
*Create object.*
- `BErrorTime` & `set` (int `errNo`, `BTimeStamp` `errTime`=`BTimeStamp`(), `BString` `errStr`="")  
*Set error number and message.*
- `BErrorTime` & `clear` ()  
*Clear the error.*
- int `getErrorNo` () const  
*Get The error number.*
- `BTimeStamp` `getTime` () const  
*Get time.*
- `BString` `getString` () const  
*Get error message.*
- `BErrorTime` `copy` ()  
*Return an independant copy.*
- `operator int` () const  
*Return error number.*

### Private Attributes

- int `oerrNo`
- `BTimeStamp` `oerrTime`
- `BString` `oerrStr`

#### 7.26.1 Detailed Description

Error return class.

#### 7.26.2 Member Enumeration Documentation

##### 7.26.2.1 Type

```
enum BErrorTime::Type
```

## Enumerator

None	
Error	

## 7.26.3 Constructor & Destructor Documentation

### 7.26.3.1 BErrorTime()

```
BErrorTime::BErrorTime (
    int errNo = None,
    BTimeStamp errTime = BTimeStamp(),
    BString errStr = "" )
```

Create object.

## 7.26.4 Member Function Documentation

### 7.26.4.1 clear()

```
BErrorTime & BErrorTime::clear ( )
```

Clear the error.

### 7.26.4.2 copy()

```
BErrorTime BErrorTime::copy ( )
```

Return an independant copy.

### 7.26.4.3 getErrorNo()

```
int BErrorTime::getErrorNo ( ) const
```

Get The error number.

#### 7.26.4.4 getString()

```
BString BErrorTime::getString ( ) const
```

Get error message.

#### 7.26.4.5 getTime()

```
BTimeStamp BErrorTime::getTime ( ) const
```

Get time.

#### 7.26.4.6 operator int()

```
BErrorTime::operator int ( ) const
```

Return error number.

#### 7.26.4.7 set()

```
BErrorTime & BErrorTime::set (
    int errNo,
    BTimeStamp errTime = BTimeStamp(),
    BString errStr = "" )
```

Set error number and message.

### 7.26.5 Member Data Documentation

#### 7.26.5.1 oerrNo

```
int BErrorTime::oerrNo [private]
```

#### 7.26.5.2 oerrStr

```
BString BErrorTime::oerrStr [private]
```

### 7.26.5.3 oerrTime

```
BTimeStamp BErrorTime::oerrTime [private]
```

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

- [BErrorTime.h](#)
- [BErrorTime.cpp](#)

## 7.27 BEvent Class Reference

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

### Public Member Functions

- [BEvent](#) (BUInt32 type=[BEventTypeNone](#), BUInt32 arg=0)
- [BUInt32 type](#) ()
- [BUInt32 arg](#) ()

### Private Attributes

- [BUInt32 otype](#)  
*The events type.*
- [BUInt32 oarg](#)  
*The events argument.*

## 7.27.1 Constructor & Destructor Documentation

### 7.27.1.1 BEvent()

```
BEvent::BEvent (  
    BUInt32 type = BEventTypeNone,  
    BUInt32 arg = 0 )
```

## 7.27.2 Member Function Documentation

### 7.27.2.1 arg()

```
BUInt32 BEvent::arg ( )
```

## 7.27.2.2 type()

```
BUInt32 BEvent::type ( )
```

## 7.27.3 Member Data Documentation

## 7.27.3.1 oarg

```
BUInt32 BEvent::oarg [private]
```

The events argument.

## 7.27.3.2 otype

```
BUInt32 BEvent::otype [private]
```

The events type.

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

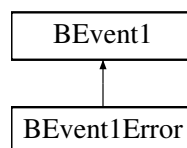
- [BEvent.h](#)
- [BEvent.cpp](#)

## 7.28 BEvent1 Class Reference

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

```
#include <BEvent1.h>
```

Inheritance diagram for BEvent1:



## Public Member Functions

- [BEvent1](#) (uint32\_t type)
- virtual [~BEvent1](#) ()
- 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.*

### 7.28.1 Detailed Description

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

### 7.28.2 Constructor & Destructor Documentation

#### 7.28.2.1 BEvent1()

```
BEvent1::BEvent1 (  
    uint32_t type )
```

#### 7.28.2.2 ~BEvent1()

```
BEvent1::~~BEvent1 ( ) [virtual]
```

### 7.28.3 Member Function Documentation

#### 7.28.3.1 getBinary()

```
BError BEvent1::getBinary (  
    void * data,  
    uint32_t & size ) [virtual]
```

Reimplemented in [BEvent1Error](#).

#### 7.28.3.2 getType()

```
uint32_t BEvent1::getType ( )
```

### 7.28.3.3 setBinary()

```
BError BEvent1::setBinary (
    void * data,
    uint32_t & size ) [virtual]
```

Reimplemented in [BEvent1Error](#).

## 7.28.4 Member Data Documentation

### 7.28.4.1 otype

```
uint32_t BEvent1::otype [private]
```

The event type.

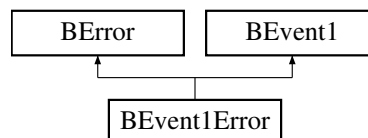
The documentation for this class was generated from the following files:

- [BEvent1.h](#)
- [BEvent1.cpp](#)

## 7.29 BEvent1Error Class Reference

```
#include <BEvent1.h>
```

Inheritance diagram for BEvent1Error:



### Public Member Functions

- [BEvent1Error](#) (int errNo=ErrorOk, BString errStr="")
- [BError](#) getBinary (void \*data, uint32\_t &size)
- [BError](#) setBinary (void \*data, uint32\_t &size)

### 7.29.1 Constructor & Destructor Documentation

### 7.29.1.1 BEvent1Error()

```
BEvent1Error::BEvent1Error (
    int errNo = ErrorOk,
    BString errStr = "" )
```

## 7.29.2 Member Function Documentation

### 7.29.2.1 getBinary()

```
BError BEvent1Error::getBinary (
    void * data,
    uint32_t & size ) [virtual]
```

Reimplemented from [BEvent1](#).

### 7.29.2.2 setBinary()

```
BError BEvent1Error::setBinary (
    void * data,
    uint32_t & size ) [virtual]
```

Reimplemented from [BEvent1](#).

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

- [BEvent1.h](#)
- [BEvent1.cpp](#)

## 7.30 BEvent1Int 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 <BEvent1.h>
```

### Public Member Functions

- [BEvent1Int](#) ()
- [~BEvent1Int](#) ()
- void [clear](#) ()  
*Clear events pending.*
- [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.*

### 7.30.1 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.

### 7.30.2 Constructor & Destructor Documentation

#### 7.30.2.1 BEvent1Int()

```
BEvent1Int::BEvent1Int ( )
```

#### 7.30.2.2 ~BEvent1Int()

```
BEvent1Int::~~BEvent1Int ( )
```

### 7.30.3 Member Function Documentation

#### 7.30.3.1 clear()

```
void BEvent1Int::clear ( )
```

Clear events pending.

#### 7.30.3.2 getEvent()

```
BError BEvent1Int::getEvent (
    int & event,
    int timeoutUs = -1 )
```

Receive the event.

### 7.30.3.3 getFd()

```
int BEvent1Int::getFd ( )
```

### 7.30.3.4 sendEvent()

```
BError BEvent1Int::sendEvent (
    int event )
```

Send an event.

## 7.30.4 Member Data Documentation

### 7.30.4.1 ofds

```
int BEvent1Int::ofds[2] [private]
```

File descriptors for pipe.

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

- [BEvent1.h](#)
- [BEvent1.cpp](#)

## 7.31 BEvent1Pipe 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 <BEvent1.h>
```

### Public Member Functions

- [BEvent1Pipe \(\)](#)
- [~BEvent1Pipe \(\)](#)
- void [clear \(\)](#)  
*Clear events pending.*
- [BError sendEvent \(BEvent1 \\*event\)](#)  
*Send an event.*
- [BError getEvent \(BEvent1 \\*event, int timeOutUs=-1\)](#)  
*Receive the event.*
- int [getReceiveFd \(\)](#)  
*returns the receive file descriptor for the poll system call*

## Private Attributes

- int `ofds` [2]  
*File descriptors for pipe.*

### 7.31.1 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.

### 7.31.2 Constructor & Destructor Documentation

#### 7.31.2.1 BEvent1Pipe()

```
BEvent1Pipe::BEvent1Pipe ( )
```

#### 7.31.2.2 ~BEvent1Pipe()

```
BEvent1Pipe::~~BEvent1Pipe ( )
```

### 7.31.3 Member Function Documentation

#### 7.31.3.1 clear()

```
void BEvent1Pipe::clear ( )
```

Clear events pending.

#### 7.31.3.2 getEvent()

```
BError BEvent1Pipe::getEvent (
    BEvent1 * event,
    int timeoutUs = -1 )
```

Receive the event.

### 7.31.3.3 getReceiveFd()

```
int BEvent1Pipe::getReceiveFd ( )
```

returns the receive file descriptor for the poll system call

### 7.31.3.4 sendEvent()

```
BError BEvent1Pipe::sendEvent (
    BEvent1 * event )
```

Send an event.

## 7.31.4 Member Data Documentation

### 7.31.4.1 ofds

```
int BEvent1Pipe::ofds[2] [private]
```

File descriptors for pipe.

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

- [BEvent1.h](#)
- [BEvent1.cpp](#)

## 7.32 BEventPipe Class Reference

This class provides an interface for sending simple integer events via a pipe file descriptor.

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

### Public Member Functions

- [BEventPipe \(\)](#)
- [~BEventPipe \(\)](#)
- void [clear \(\)](#)  
*Clear events pending.*
- int [getFd \(\)](#)
- [BUInt writeAvailable \(\)](#) const
- [BError write](#) (const [BEvent](#) &event, [BTimeout](#) timeout=[BTimeoutForever](#))  
*Append an item onto the queue.*
- [BUInt readAvailable \(\)](#) const
- [BError read](#) ([BEvent](#) &event, [BTimeout](#) timeout=[BTimeoutForever](#))  
*Get an item from the queue.*

## Private Attributes

- int `ofds` [2]  
*File descriptors for pipe.*

### 7.32.1 Detailed Description

This class provides an interface for sending simple integer events via a pipe file descriptor.

### 7.32.2 Constructor & Destructor Documentation

#### 7.32.2.1 BEventPipe()

```
BEventPipe::BEventPipe ( )
```

#### 7.32.2.2 ~BEventPipe()

```
BEventPipe::~BEventPipe ( )
```

### 7.32.3 Member Function Documentation

#### 7.32.3.1 clear()

```
void BEventPipe::clear ( )
```

Clear events pending.

#### 7.32.3.2 getFd()

```
int BEventPipe::getFd ( )
```

### 7.32.3.3 read()

```
BError BEventPipe::read (
    BEvent & event,
    BTimeout timeout = BTimeoutForever )
```

Get an item from the queue.

### 7.32.3.4 readAvailable()

```
BUInt BEventPipe::readAvailable ( ) const
```

### 7.32.3.5 write()

```
BError BEventPipe::write (
    const BEvent & event,
    BTimeout timeout = BTimeoutForever )
```

Append an item onto the queue.

### 7.32.3.6 writeAvailable()

```
BUInt BEventPipe::writeAvailable ( ) const
```

## 7.32.4 Member Data Documentation

### 7.32.4.1 ofds

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

File descriptors for pipe.

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

- [BEvent.h](#)
- [BEvent.cpp](#)

## 7.33 BFifo< Type > Class Template Reference

```
#include <BFifo.h>
```

### Public Member Functions

- [BFifo \(BUInt size\)](#)
- [~BFifo \(\)](#)
- void [clear \(\)](#)
- [BUInt size \(\)](#)  
*Returns fifo size.*
- [BError resize \(BUInt size\)](#)  
*Resize FIFO, clears it as well.*
- [BError rebase \(\)](#)  
*Rebases fifo so read pointer is at zero moving memory as needed.*
- [BUInt writeAvailable \(\)](#)  
*How many items that can be written.*
- [BUInt writeAvailableChunk \(\)](#)  
*How many items that can be written in a chunk.*
- [BError write \(const Type v\)](#)  
*Write a single item.*
- [BError write \(const Type \\*data, BUInt num\)](#)  
*Write a set of items. Can only write a maximum of [writeAvailableChunk\(\)](#) to save going beyond end of FIFO buffer.*
- Type \* [writeData \(\)](#)  
*Returns a pointer to the data.*
- Type \* [writeData \(BUInt &num\)](#)  
*Returns a pointer to the data and how many can be written in a chunk.*
- void [writeDone \(BUInt num\)](#)  
*Indicates when write is complete.*
- void [writeBackup \(BUInt num\)](#)  
*Backup, remove num items at end of fifo. Careful, make sure read is not already happening.*
- [BUInt readAvailable \(\)](#)  
*How many items are available to read.*
- [BUInt readAvailableChunk \(\)](#)  
*How many items are available to read in a chunk.*
- Type [read \(\)](#)  
*Read one item.*
- [BError read \(Type \\*data, BUInt num\)](#)  
*Read a set of items.*
- Type \* [readData \(\)](#)  
*Returns a pointer to the data.*
- Type \* [readData \(BUInt &num\)](#)  
*Returns a pointer to the data and how many can be read in a chunk.*
- void [readDone \(BUInt num\)](#)
- Type [readPos \(BUInt pos\)](#)  
*Read item at given offset from current read position.*
- void [writePos \(BUInt pos, const Type &v\)](#)  
*Write item at given offset from current read position.*
- Type & [operator\[\]](#) (int pos)  
*Direct access to read samples in buffer.*

## Protected Attributes

- [BUInt osize](#)  
*The size of the FIFO.*
- `Type * odata`  
*FIFO memory buffer.*
- `volatile BUInt owritePos`  
*The write pointer.*
- `volatile BUInt oreadPos`  
*The read pointer.*

## 7.33.1 Constructor & Destructor Documentation

### 7.33.1.1 Bfifo()

```
template<class Type>
Bfifo< Type >::Bfifo (
    BUInt size )
```

### 7.33.1.2 ~Bfifo()

```
template<class Type>
Bfifo< Type >::~~Bfifo ( )
```

## 7.33.2 Member Function Documentation

### 7.33.2.1 clear()

```
template<class Type>
void Bfifo< Type >::clear ( )
```

### 7.33.2.2 operator[]()

```
template<class Type>
Type& Bfifo< Type >::operator[] (
    int pos )
```

Direct access to read samples in buffer.



### 7.33.2.3 read() [1/2]

```
template<class Type>
Type BFifo< Type >::read ( )
```

Read one item.

### 7.33.2.4 read() [2/2]

```
template<class Type>
BError BFifo< Type >::read (
    Type * data,
    BUInt num )
```

Read a set of items.

### 7.33.2.5 readAvailable()

```
template<class Type>
BUInt BFifo< Type >::readAvailable ( )
```

How many items are available to read.

### 7.33.2.6 readAvailableChunk()

```
template<class Type>
BUInt BFifo< Type >::readAvailableChunk ( )
```

How many items are available to read in a chunk.

### 7.33.2.7 readData() [1/2]

```
template<class Type>
Type* BFifo< Type >::readData ( )
```

Returns a pointer to the data.

### 7.33.2.8 readData() [2/2]

```
template<class Type>
Type* BFifo< Type >::readData (
    BUInt & num )
```

Returns a pointer to the data and how many can be read in a chunk.

### 7.33.2.9 readDone()

```
template<class Type>
void BFifo< Type >::readDone (
    BUInt num )
```

### 7.33.2.10 readPos()

```
template<class Type>
Type BFifo< Type >::readPos (
    BUInt pos )
```

Read item at given offset from current read position.

### 7.33.2.11 rebase()

```
template<class Type>
BError BFifo< Type >::rebase ( )
```

Rebases fifo so read pointer is at zero moving memory as needed.

### 7.33.2.12 resize()

```
template<class Type>
BError BFifo< Type >::resize (
    BUInt size )
```

Resize FIFO, clears it as well.

### 7.33.2.13 size()

```
template<class Type>
BUInt BFifo< Type >::size ( )
```

Returns fifo size.

### 7.33.2.14 write() [1/2]

```
template<class Type>
BError BFifo< Type >::write (
    const Type v )
```

Write a single item.

### 7.33.2.15 write() [2/2]

```
template<class Type>
BError BFifo< Type >::write (
    const Type * data,
    BUInt num )
```

Write a set of items. Can only write a maximum of [writeAvailableChunk\(\)](#) to save going beyond end of FIFO buffer.

### 7.33.2.16 writeAvailable()

```
template<class Type>
BUInt BFifo< Type >::writeAvailable ( )
```

How many items that can be written.

### 7.33.2.17 writeAvailableChunk()

```
template<class Type>
BUInt BFifo< Type >::writeAvailableChunk ( )
```

How many items that can be written in a chunk.

### 7.33.2.18 writeBackup()

```
template<class Type>
void BFifo< Type >::writeBackup (
    BUInt num )
```

Backup, remove num items at end of fifo. Careful, make sure read is not already happening.

### 7.33.2.19 writeData() [1/2]

```
template<class Type>
Type* BFifo< Type >::writeData ( )
```

Returns a pointer to the data.

### 7.33.2.20 writeData() [2/2]

```
template<class Type>
Type* BFifo< Type >::writeData (
    BUInt & num )
```

Returns a pointer to the data and how many can be written in a chunk.

### 7.33.2.21 writeDone()

```
template<class Type>
void BFifo< Type >::writeDone (
    BUInt num )
```

Indicates when write is complete.

### 7.33.2.22 writePos()

```
template<class Type>
void BFifo< Type >::writePos (
    BUInt pos,
    const Type & v )
```

Write item at given offset from current read position.

### 7.33.3 Member Data Documentation

#### 7.33.3.1 odata

```
template<class Type>
Type* BFifo< Type >::odata [protected]
```

FIFO memory buffer.

#### 7.33.3.2 oreadPos

```
template<class Type>
volatile BUInt BFifo< Type >::oreadPos [protected]
```

The read pointer.

#### 7.33.3.3 osize

```
template<class Type>
BUInt BFifo< Type >::osize [protected]
```

The size of the FIFO.

#### 7.33.3.4 owritePos

```
template<class Type>
volatile BUInt BFifo< Type >::owritePos [protected]
```

The write pointer.

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

- [BFifo.h](#)

## 7.34 BFifoCirc< Type > Class Template Reference

This class implements a thread safe FIFO buffer.

```
#include <BFifoCirc.h>
```

## Public Types

- enum { `defaultSize` = 1024 }

## Public Member Functions

- `BFifoCirc` (uint32\_t size=defaultSize)
- `~BFifoCirc` ()
- uint32\_t `size` ()  
*Return the buffers actual size.*
- void `clear` ()  
*Clear all of the data in the buffer.*
- uint32\_t `writeAvailable` ()  
*Returns the space available to write.*
- `BError writeWaitAvailable` (uint32\_t numFifoSamples)  
*Wait for the given number of samples.*
- `BError write` (const Type \*data, uint32\_t numFifoSamples)  
*Writes the data to the buffer. Blocks until complete.*
- Type \* `writeData` ()  
*Return a pointer to the current start of the buffer.*
- void `writeDone` (uint32\_t numFifoSamples)  
*Update the write pointer.*
- uint32\_t `readAvailable` ()  
*Returns the number of bytes of data available.*
- `BError readWaitAvailable` (uint32\_t numFifoSamples)  
*Wait for given number of samples.*
- `BError read` (Type \*data, uint32\_t numFifoSamples)
- Type \* `readData` ()  
*Pointer to raw data.*
- `BError readDone` (uint32\_t numFifoSamples)  
*Updates read pointer.*
- Type & `operator[]` (int pos)  
*Direct access to read samples in buffer.*

## Protected Member Functions

- `BError mapCircularBuffer` (uint32\_t size)
- void `unmapCircularBuffer` ()

## Protected Attributes

- `BMutex olock`
- uint32\_t `ovmSize`
- uint32\_t `osize`
- Type \* `odata`
- `BFifoCircPos owritePos`  
*Current write position.*
- `BCondValue owriteNumFifoSamples`  
*The number of samples in the FIFO.*
- `BFifoCircPos oreadPos`  
*Current read position.*

### 7.34.1 Detailed Description

```
template<class Type>
class BFifoCirc< Type >
```

This class implements a thread safe FIFO buffer.

### 7.34.2 Member Enumeration Documentation

#### 7.34.2.1 anonymous enum

```
template<class Type >
anonymous enum
```

##### Enumerator

defaultSize	
-------------	--

### 7.34.3 Constructor & Destructor Documentation

#### 7.34.3.1 BFifoCirc()

```
template<class Type >
BFifoCirc< Type >::BFifoCirc (
    uint32_t size = defaultSize )
```

#### 7.34.3.2 ~BFifoCirc()

```
template<class Type >
BFifoCirc< Type >::~~BFifoCirc ( )
```

### 7.34.4 Member Function Documentation

#### 7.34.4.1 clear()

```
template<class Type >
void BFifoCirc< Type >::clear ( )
```

Clear all of the data in the buffer.

#### 7.34.4.2 mapCircularBuffer()

```
template<class Type >
BError BFifoCirc< Type >::mapCircularBuffer (
    uint32_t size ) [protected]
```

#### 7.34.4.3 operator[]()

```
template<class Type >
Type& BFifoCirc< Type >::operator[] (
    int pos )
```

Direct access to read samples in buffer.

#### 7.34.4.4 read()

```
template<class Type >
BError BFifoCirc< Type >::read (
    Type * data,
    uint32_t numFifoSamples )
```

#### 7.34.4.5 readAvailable()

```
template<class Type >
uint32_t BFifoCirc< Type >::readAvailable ( )
```

Returns the number of bytes of data available.

#### 7.34.4.6 readData()

```
template<class Type >
Type* BFifoCirc< Type >::readData ( )
```

Pointer to raw data.



#### 7.34.4.7 readDone()

```
template<class Type >
BError BFifoCirc< Type >::readDone (
    uint32_t numFifoSamples )
```

Updates read pointer.

#### 7.34.4.8 readWaitAvailable()

```
template<class Type >
BError BFifoCirc< Type >::readWaitAvailable (
    uint32_t numFifoSamples )
```

Wait for given number of samples.

#### 7.34.4.9 size()

```
template<class Type >
uint32_t BFifoCirc< Type >::size ( )
```

Return the buffers actual size.

#### 7.34.4.10 unmapCircularBuffer()

```
template<class Type >
void BFifoCirc< Type >::unmapCircularBuffer ( ) [protected]
```

#### 7.34.4.11 write()

```
template<class Type >
BError BFifoCirc< Type >::write (
    const Type * data,
    uint32_t numFifoSamples )
```

Writes the data to the buffer. Blocks until complete.

#### 7.34.4.12 writeAvailable()

```
template<class Type >
uint32_t BFifoCirc< Type >::writeAvailable ( )
```

Returns the space available to write.

#### 7.34.4.13 writeData()

```
template<class Type >
Type* BFifoCirc< Type >::writeData ( )
```

Return a pointer to the current start of the buffer.

#### 7.34.4.14 writeDone()

```
template<class Type >
void BFifoCirc< Type >::writeDone (
    uint32_t numFifoSamples )
```

Update the write pointer.

#### 7.34.4.15 writeWaitAvailable()

```
template<class Type >
BError BFifoCirc< Type >::writeWaitAvailable (
    uint32_t numFifoSamples )
```

Wait for the given number of samples.

### 7.34.5 Member Data Documentation

#### 7.34.5.1 odata

```
template<class Type >
Type* BFifoCirc< Type >::odata [protected]
```

#### 7.34.5.2 olock

```
template<class Type >
BFifoCirc< Type >::olock [protected]
```

#### 7.34.5.3 oreadPos

```
template<class Type >
BFifoCircPos BFifoCirc< Type >::oreadPos [protected]
```

Current read position.

#### 7.34.5.4 osize

```
template<class Type >
uint32_t BFifoCirc< Type >::osize [protected]
```

#### 7.34.5.5 ovmSize

```
template<class Type >
uint32_t BFifoCirc< Type >::ovmSize [protected]
```

#### 7.34.5.6 owriteNumFifoSamples

```
template<class Type >
BCondValue BFifoCirc< Type >::owriteNumFifoSamples [protected]
```

The number of samples in the FIFO.

#### 7.34.5.7 owritePos

```
template<class Type >
BFifoCircPos BFifoCirc< Type >::owritePos [protected]
```

Current write position.

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

- [BFifoCirc.h](#)

## 7.35 BFifoCircPos Class Reference

This class implements a pointer into the Fifo's circular buffer.

```
#include <BFifoCirc.h>
```

### Public Member Functions

- [BFifoCircPos](#) (uint32\_t size)
- void [setSize](#) (uint32\_t size)
- void [set](#) (uint32\_t pos)  
*Sets the position.*
- uint32\_t [pos](#) ()  
*The current position.*
- void [increment](#) (uint32\_t numFifoSamples)  
*Increment the pointer by the given value.*
- uint32\_t [difference](#) (const [BFifoCircPos](#) &pos)  
*Return the difference between the two pointers.*
- [operator int](#) ()
- void [operator+=](#) (uint32\_t numFifoSamples)
- int [operator==](#) (const [BFifoCircPos](#) &pos)
- int [operator!=](#) (const [BFifoCircPos](#) &pos)

### Private Attributes

- uint32\_t [osize](#)
- uint32\_t [opos](#)

### 7.35.1 Detailed Description

This class implements a pointer into the Fifo's circular buffer.

### 7.35.2 Constructor & Destructor Documentation

#### 7.35.2.1 BFifoCircPos()

```
BFifoCircPos::BFifoCircPos (  
    uint32_t size )
```

### 7.35.3 Member Function Documentation

### 7.35.3.1 difference()

```
uint32_t BFifoCircPos::difference (
    const BFifoCircPos & pos )
```

Return the difference between the two pointers.

### 7.35.3.2 increment()

```
void BFifoCircPos::increment (
    uint32_t numFifoSamples )
```

Increment the pointer by the given value.

### 7.35.3.3 operator int()

```
BFifoCircPos::operator int ( )
```

### 7.35.3.4 operator"!=(())

```
int BFifoCircPos::operator!=(
    const BFifoCircPos & pos )
```

### 7.35.3.5 operator+="()

```
void BFifoCircPos::operator+=(
    uint32_t numFifoSamples )
```

### 7.35.3.6 operator=="()

```
int BFifoCircPos::operator==(
    const BFifoCircPos & pos )
```

#### 7.35.3.7 pos()

```
uint32_t BFifoCircPos::pos ( )
```

The current position.

#### 7.35.3.8 set()

```
void BFifoCircPos::set (
    uint32_t pos )
```

Sets the position.

#### 7.35.3.9 setSize()

```
void BFifoCircPos::setSize (
    uint32_t size )
```

### 7.35.4 Member Data Documentation

#### 7.35.4.1 opos

```
uint32_t BFifoCircPos::opos [private]
```

#### 7.35.4.2 osize

```
uint32_t BFifoCircPos::osize [private]
```

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

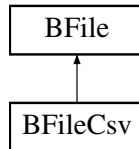
- [BFifoCirc.h](#)
- [BFifoCirc.cpp](#)

## 7.36 BFile Class Reference

File operations class.

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

Inheritance diagram for BFile:



### Public Member Functions

- [BFile](#) ()
- [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 open](#) (int fd, [BString](#) mode)
  - Assign object to opened file descriptor.*
- [BError close](#) ()
  - Close file.*
- int [isOpen](#) ()
  - Returns 1 if the file is open.*
- int [isEnd](#) ()
  - Returns 1 if at the end of the file, 0 otherwise.*
- FILE \* [getFd](#) ()
  - File descriptor.*
- [BUInt64 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)*
- char \* [fgets](#) (char \*buf, size\_t size)
  - 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](#) ([BUInt64](#) pos)
  - Set seek position.*
- [BUInt64 position](#) ()

*The files position.*

- `int printf (const char *fmt,...)`

*Formatted print into the file.*

- `BError truncate ()`

*Truncate the file.*

- `BError flush ()`

*Flush the file.*

- `BString fileName ()`

*Return file name.*

- `BFile & operator= (const BFile &file)`

## Private Attributes

- `FILE * ofile`
- `BString ofileName`
- `BString omode`

### 7.36.1 Detailed Description

File operations class.

### 7.36.2 Constructor & Destructor Documentation

#### 7.36.2.1 BFile() [1/2]

```
BFile::BFile ( )
```

#### 7.36.2.2 BFile() [2/2]

```
BFile::BFile (
    const BFile & file )
```

Create opened specified file.

#### 7.36.2.3 ~BFile()

```
BFile::~~BFile ( )
```



### 7.36.3 Member Function Documentation

#### 7.36.3.1 close()

```
BError BFile::close ( )
```

Close file.

#### 7.36.3.2 fgets()

```
char * BFile::fgets (
    char * buf,
    size_t size )
```

#### 7.36.3.3 fileName()

```
BString BFile::fileName ( )
```

Return file name.

#### 7.36.3.4 flush()

```
BError BFile::flush ( )
```

Flush the file.

#### 7.36.3.5 getFd()

```
FILE * BFile::getFd ( )
```

File descriptor.

### 7.36.3.6 isEnd()

```
int BFile::isEnd ( )
```

Returns 1 if at the end of the file, 0 otherwise.

### 7.36.3.7 isOpen()

```
int BFile::isOpen ( )
```

Returns 1 if the file is open.

### 7.36.3.8 length()

```
BUInt64 BFile::length ( )
```

File size in bytes.

### 7.36.3.9 open() [1/3]

```
BError BFile::open (
    BString name,
    BString mode )
```

Open file.

### 7.36.3.10 open() [2/3]

```
BError BFile::open (
    FILE * file )
```

Assign object to opened file handle.

### 7.36.3.11 open() [3/3]

```
BError BFile::open (
    int fd,
    BString mode )
```

Assign object to opened file descriptor.

### 7.36.3.12 operator=()

```
BFile & BFile::operator= (
    const BFile & file )
```

### 7.36.3.13 position()

```
BUInt64 BFile::position ( )
```

The files position.

### 7.36.3.14 printf()

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

Formatted print into the file.

### 7.36.3.15 read()

```
int BFile::read (
    void * buf,
    int nbytes )
```

Read from file.

### 7.36.3.16 readString()

```
int BFile::readString (
    BString & str )
```

Read string. (ref fgets)

### 7.36.3.17 seek()

```
int BFile::seek (
    BUInt64 pos )
```

Set seek position.

#### 7.36.3.18 setVBuf()

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

Set stream buffering options.

#### 7.36.3.19 truncate()

```
BError BFile::truncate ( )
```

Truncate the file.

#### 7.36.3.20 write()

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

Write to file.

#### 7.36.3.21 writeString()

```
int BFile::writeString (
    const BString & str )
```

Write string to file.

### 7.36.4 Member Data Documentation

#### 7.36.4.1 ofile

```
FILE* BFile::ofile [private]
```

#### 7.36.4.2 ofileName

```
BString BFile::ofileName [private]
```

#### 7.36.4.3 omode

```
BString BFile::omode [private]
```

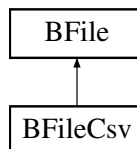
The documentation for this class was generated from the following files:

- [BFile.h](#)
- [BFile.cpp](#)

## 7.37 BFileCsv Class Reference

```
#include <BFileCsv.h>
```

Inheritance diagram for BFileCsv:



### Public Member Functions

- [BFileCsv](#) (char separator=';')
- [BError readCsv](#) (BStringList &csvList)
- [BError writeCsv](#) (BStringList &csvList)

### Private Attributes

- char [oseparator](#)

### 7.37.1 Constructor & Destructor Documentation

#### 7.37.1.1 BFileCsv()

```
BFileCsv::BFileCsv (  
    char separator = ';' )
```

## 7.37.2 Member Function Documentation

### 7.37.2.1 readCsv()

```
BError BFileCsv::readCsv (
    BStringList & csvList )
```

### 7.37.2.2 writeCsv()

```
BError BFileCsv::writeCsv (
    BStringList & csvList )
```

## 7.37.3 Member Data Documentation

### 7.37.3.1 oseparator

```
char BFileCsv::oseparator [private]
```

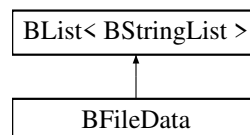
The documentation for this class was generated from the following files:

- [BFileCsv.h](#)
- [BFileCsv.cpp](#)

## 7.38 BFileData Class Reference

```
#include <BFileData.h>
```

Inheritance diagram for BFileData:



### Public Member Functions

- [BError open](#) (BString filename)
- [BError getNextId](#) (int &id)
- [BError find](#) (int id, BStringList &csvList)
- [BError write](#) (int id, BStringList &csvList)
- [BError del](#) (int id)

## Private Member Functions

- [BError read \(\)](#)
- [BError write \(\)](#)

## Private Attributes

- [BString ofilename](#)

## Additional Inherited Members

### 7.38.1 Member Function Documentation

#### 7.38.1.1 del()

```
BError BFileData::del (  
    int id )
```

#### 7.38.1.2 find()

```
BError BFileData::find (  
    int id,  
    BStringList & csvList )
```

#### 7.38.1.3 getNextId()

```
BError BFileData::getNextId (  
    int & id )
```

#### 7.38.1.4 open()

```
BError BFileData::open (  
    BString filename )
```

### 7.38.1.5 read()

```
BError BFileData::read ( ) [private]
```

### 7.38.1.6 write() [1/2]

```
BError BFileData::write (
    int id,
    BStringList & csvList )
```

### 7.38.1.7 write() [2/2]

```
BError BFileData::write ( ) [private]
```

## 7.38.2 Member Data Documentation

### 7.38.2.1 ofilename

```
BString BFileData::ofilename [private]
```

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

- [BFileData.h](#)
- [BFileData.cpp](#)

## 7.39 BIter Class Reference

Iterator for [BList](#).

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

### Public Member Functions

- [BIter](#) ([BNode](#) \*i=0)
- [operator BNode \\* \(\)](#)
- [int operator==](#) (const [BIter](#) &i)
- [int valid](#) ()



## Private Attributes

- [BNode](#) \* *oi*

### 7.39.1 Detailed Description

Iterator for [BList](#).

### 7.39.2 Constructor & Destructor Documentation

#### 7.39.2.1 BIter()

```
BIter::BIter (  
    BNode * i = 0 ) [inline]
```

### 7.39.3 Member Function Documentation

#### 7.39.3.1 operator [BNode](#) \*()

```
BIter::operator BNode * ( ) [inline]
```

#### 7.39.3.2 operator==( )

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

#### 7.39.3.3 valid()

```
int BIter::valid ( ) [inline]
```

### 7.39.4 Member Data Documentation

## 7.39.4.1 oi

```
BNode* BIter::oi [private]
```

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

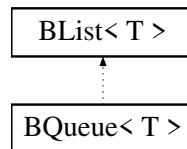
- [BList.h](#)

## 7.40 BList< T > Class Template Reference

Template based list class.

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

Inheritance diagram for BList< T >:



### Classes

- class [Node](#)

### 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) const  
*Iterator for pos item in list.*
- int `position` (BIter i)  
*Postition in list item with iterator i.*
- unsigned int `number` () const  
*Number of items in list.*
- unsigned int `size` () const  
*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.*
- int `has` (const T &i) const  
*Checks if the item is in the 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\[\]](#) (Blter i)
- const T & [operator\[\]](#) (const Blter &i) const
- [BList](#)< T > [operator+](#) (const [BList](#)< T > &l) const

### Protected Member Functions

- virtual [Node](#) \* [nodeGet](#) (Blter i)
- virtual const [Node](#) \* [nodeGet](#) (Blter i) const
- virtual [Node](#) \* [nodeCreate](#) (const T &item)

### Protected Attributes

- [Node](#) \* [onodes](#)
- unsigned int [olength](#)

### Private Member Functions

- virtual [Node](#) \* [nodeCreate](#) ()

## 7.40.1 Detailed Description

```
template<class T>  
class BList< T >
```

Template based list class.

## 7.40.2 Member Typedef Documentation

### 7.40.2.1 SortFunc

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

Prototype for sorting function.

## 7.40.3 Constructor & Destructor Documentation

### 7.40.3.1 BList() [1/2]

```
template<class T >
BList< T >::BList ( )
```

### 7.40.3.2 BList() [2/2]

```
template<class T>
BList< T >::BList (
    const BList< T > & l )
```

### 7.40.3.3 ~BList()

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

## 7.40.4 Member Function Documentation

### 7.40.4.1 append() [1/2]

```
template<class T>
void BList< T >::append (
    const T & item )
```

Append item to list.

### 7.40.4.2 append() [2/2]

```
template<class T>
void BList< T >::append (
    const BList< T > & l )
```

Append list to list.

#### 7.40.4.3 begin()

```
template<class T >
BIter BList< T >::begin ( ) const
```

Iterator for start of list.

#### 7.40.4.4 clear()

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

Clear the list.

Reimplemented in [BEntryFile](#), [BDir](#), [BEntryList](#), [BDict< Type >](#), [BQueue< T >](#), and [BQueue< BoapMcPacket >](#).

#### 7.40.4.5 del()

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

Delete specified item.

Reimplemented in [BEntryList](#), and [BDict< Type >](#).

#### 7.40.4.6 deleteFirst()

```
template<class T >
void BList< T >::deleteFirst ( )
```

Delete first item.

#### 7.40.4.7 deleteLast()

```
template<class T >
void BList< T >::deleteLast ( )
```

Delete last item.

**7.40.4.8 end()** [1/2]

```
template<class T >
BIter BList< T >::end ( ) const
```

Iterator for end of list.

**7.40.4.9 end()** [2/2]

```
template<class T >
BIter BList< T >::end (
    BIter & i ) const
```

Iterator for end of list.

**7.40.4.10 front()**

```
template<class T >
T & BList< T >::front ( )
```

Get first item in list.

**7.40.4.11 get()** [1/2]

```
template<class T >
T & BList< T >::get (
    BIter i )
```

Get item specified by iterator in list.

**7.40.4.12 get()** [2/2]

```
template<class T >
const T & BList< T >::get (
    BIter i ) const
```

Get item specified by iterator in list.

#### 7.40.4.13 goto()

```
template<class T >
BIter BList< T >::goto (
    int pos ) const
```

Iterator for pos item in list.

#### 7.40.4.14 has()

```
template<class T>
int BList< T >::has (
    const T & i ) const
```

Checks if the item is in the list.

#### 7.40.4.15 insert()

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

Insert item before item.

Reimplemented in [BEntryList](#), and [BDict< Type >](#).

#### 7.40.4.16 insertAfter()

```
template<class T>
void BList< T >::insertAfter (
    BIter & i,
    const T & item )
```

Insert item after item.

#### 7.40.4.17 isEnd()

```
template<class T >
int BList< T >::isEnd (
    BIter & i ) const
```

True if iterator refers to last item.



**7.40.4.18 next()**

```
template<class T >
void BList< T >::next (
    BIter & i ) const
```

Iterator for next item in list.

**7.40.4.19 nodeCreate()** [1/2]

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

**7.40.4.20 nodeCreate()** [2/2]

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

**7.40.4.21 nodeGet()** [1/2]

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

**7.40.4.22 nodeGet()** [2/2]

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

**7.40.4.23 number()**

```
template<class T >
unsigned int BList< T >::number ( ) const
```

Number of items in list.

#### 7.40.4.24 operator+()

```
template<class T>
BList< T > BList< T >::operator+ (
    const BList< T > & l ) const
```

#### 7.40.4.25 operator=()

```
template<class T>
BList< T > & BList< T >::operator= (
    const BList< T > & l )
```

#### 7.40.4.26 operator[]() [1/4]

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

#### 7.40.4.27 operator[]() [2/4]

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

#### 7.40.4.28 operator[]() [3/4]

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

#### 7.40.4.29 operator[]() [4/4]

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

#### 7.40.4.30 pop()

```
template<class T >
T BList< T >::pop ( )
```

Pop item from list deleting item.

#### 7.40.4.31 position()

```
template<class T >
int BList< T >::position (
    BIter i )
```

Position in list item with iterator i.

#### 7.40.4.32 prev()

```
template<class T >
void BList< T >::prev (
    BIter & i )
```

Iterator for previous item in list.

#### 7.40.4.33 push()

```
template<class T>
void BList< T >::push (
    const T & i )
```

Push item onto list.

#### 7.40.4.34 queueAdd()

```
template<class T>
void BList< T >::queueAdd (
    const T & i )
```

Add item to end of list.

#### 7.40.4.35 queueGet()

```
template<class T >
T BList< T >::queueGet ( )
```

Get item from front of list deleting item.

#### 7.40.4.36 rear()

```
template<class T >
T & BList< T >::rear ( )
```

Get last item in list.

#### 7.40.4.37 size()

```
template<class T >
unsigned int BList< T >::size ( ) const
```

Number of items in list.

#### 7.40.4.38 sort() [1/2]

```
template<class T >
void BList< T >::sort ( )
```

Sort list based on get(i) values.

#### 7.40.4.39 sort() [2/2]

```
template<class T >
void BList< T >::sort (
    SortFunc func )
```

Sort list based on Sort func.

#### 7.40.4.40 start()

```
template<class T >
void BList< T >::start (
    BIter & i ) const
```

Iterator to start of list.

#### 7.40.4.41 swap()

```
template<class T >
void BList< T >::swap (
    BIter i1,
    BIter i2 )
```

Swap two items in list.

### 7.40.5 Member Data Documentation

#### 7.40.5.1 olength

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

#### 7.40.5.2 onodes

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

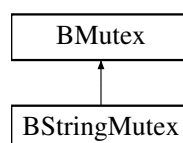
The documentation for this class was generated from the following files:

- [BList.h](#)
- [BList\\_func.h](#)

## 7.41 BMutex Class Reference

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

Inheritance diagram for BMutex:



## Public Types

- enum `Type` { `Normal`, `Recursive` }

## Public Member Functions

- `BMutex` (`Type` type=`Normal`)
- `BMutex` (const `BMutex` &mutex)
- `~BMutex` ()
- int `lock` ()
  - Set lock, wait as necessary.*
- int `timedLock` (int timeoutUs)
  - Set lock, wait as necessary but timeout after given time.*
- int `unlock` ()
  - Unlock the lock.*
- int `tryLock` ()
  - Test the lock.*
- `BMutex` & `operator=` (const `BMutex` &mutex)

## Private Attributes

- `pthread_mutex_t` `omutex`

### 7.41.1 Detailed Description

Mutex class Note these are recursive Mutexes and so you need to make sure the number of unlocks equals the number of locks.

### 7.41.2 Member Enumeration Documentation

#### 7.41.2.1 Type

```
enum BMutex::Type
```

#### Enumerator

Normal	
Recursive	

### 7.41.3 Constructor & Destructor Documentation

### 7.41.3.1 BMutex() [1/2]

```
BMutex::BMutex (
    Type type = Normal )
```

### 7.41.3.2 BMutex() [2/2]

```
BMutex::BMutex (
    const BMutex & mutex )
```

### 7.41.3.3 ~BMutex()

```
BMutex::~~BMutex ( )
```

## 7.41.4 Member Function Documentation

### 7.41.4.1 lock()

```
int BMutex::lock ( )
```

Set lock, wait as necessary.

### 7.41.4.2 operator=()

```
BMutex & BMutex::operator= (
    const BMutex & mutex )
```

### 7.41.4.3 timedLock()

```
int BMutex::timedLock (
    int timeoutUs )
```

Set lock, wait as necessary but timeout after given time.

#### 7.41.4.4 tryLock()

```
int BMutex::tryLock ( )
```

Test the lock.

#### 7.41.4.5 unlock()

```
int BMutex::unlock ( )
```

Unlock the lock.

### 7.41.5 Member Data Documentation

#### 7.41.5.1 omutex

```
pthread_mutex_t BMutex::omutex [private]
```

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

- [BMutex.h](#)
- [BMutex.cpp](#)

## 7.42 BMutexLock Class Reference

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

### Public Member Functions

- [BMutexLock](#) ([BMutex](#) &lock, int doLock=0)
- [~BMutexLock](#) ()
- int [lock](#) ()
- int [unlock](#) ()

### Private Attributes

- [BMutex](#) & olock

#### 7.42.1 Constructor & Destructor Documentation



### 7.42.1.1 BMutexLock()

```
BMutexLock::BMutexLock (
    BMutex & lock,
    int doLock = 0 ) [inline]
```

### 7.42.1.2 ~BMutexLock()

```
BMutexLock::~~BMutexLock ( ) [inline]
```

## 7.42.2 Member Function Documentation

### 7.42.2.1 lock()

```
int BMutexLock::lock ( ) [inline]
```

### 7.42.2.2 unlock()

```
int BMutexLock::unlock ( ) [inline]
```

## 7.42.3 Member Data Documentation

### 7.42.3.1 olock

```
BMutex& BMutexLock::olock [private]
```

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

- [BMutex.h](#)

## 7.43 Bmysql Class Reference

```
#include <Bmysql.h>
```

## Public Member Functions

- [BMySql](#) ()
- [~BMySql](#) ()
- [BError open](#) ([BString](#) hostName, [BString](#) dataBase, [BString](#) userName, [BString](#) password)
- [void close](#) ()
- [BError get](#) ([BString](#) table, [BString](#) where, [BDictString](#) &fields)
- [BError insert](#) ([BString](#) table, [BDictString](#) fields, [BUInt32](#) \*id=0)
- [BError update](#) ([BString](#) table, [BUInt32](#) id, [BDictString](#) fields)
- [BError del](#) ([BString](#) table, [BUInt32](#) id)
  - Delete record from table.*
- [BError flush](#) ()
  - Flush all data to disk.*
- [BString escapeString](#) ([BString](#) str)
  - Escapes special characters in the string.*
- [BError query](#) ([BString](#) cmd, [BList](#)< [BDictString](#) > &result)
- [MYSQL & db](#) ()
- [void setDebug](#) (int debug)

## Private Attributes

- [MYSQL odb](#)
- [int oopened](#)
- [int odebug](#)
- [BMutex olock](#)

## 7.43.1 Constructor & Destructor Documentation

### 7.43.1.1 BMySql()

```
BMySql::BMySql ( )
```

### 7.43.1.2 ~BMySql()

```
BMySql::~~BMySql ( )
```

## 7.43.2 Member Function Documentation

#### 7.43.2.1 close()

```
void Bmysql::close ( )
```

#### 7.43.2.2 db()

```
MYSQL & Bmysql::db ( )
```

#### 7.43.2.3 del()

```
BError Bmysql::del (
    BString table,
    BUInt32 id )
```

Delete record from table.

#### 7.43.2.4 escapeString()

```
BString Bmysql::escapeString (
    BString str )
```

Escapes special characters in the string.

#### 7.43.2.5 flush()

```
BError Bmysql::flush ( )
```

Flush all data to disk.

#### 7.43.2.6 get()

```
BError Bmysql::get (
    BString table,
    BString where,
    BDictString & fields )
```

#### 7.43.2.7 insert()

```
BError Bmysql::insert (
    BString table,
    BDictString fields,
    BUInt32 * id = 0 )
```

#### 7.43.2.8 open()

```
BError Bmysql::open (
    BString hostName,
    BString dataBase,
    BString userName,
    BString password )
```

#### 7.43.2.9 query()

```
BError Bmysql::query (
    BString cmd,
    BList< BDictString > & result )
```

#### 7.43.2.10 setDebug()

```
void Bmysql::setDebug (
    int debug )
```

#### 7.43.2.11 update()

```
BError Bmysql::update (
    BString table,
    BUInt32 id,
    BDictString fields )
```

### 7.43.3 Member Data Documentation

#### 7.43.3.1 odb

```
MYSQL BMySQL::odb [private]
```

#### 7.43.3.2 odebug

```
int BMySQL::odebug [private]
```

#### 7.43.3.3 olock

```
BMutex BMySQL::olock [private]
```

#### 7.43.3.4 oopened

```
int BMySQL::opened [private]
```

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

- [BMySQL.h](#)
- [BMySQL.cpp](#)

## 7.44 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

## 7.44.1 Constructor & Destructor Documentation

### 7.44.1.1 BNameValue() [1/2]

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

### 7.44.1.2 BNameValue() [2/2]

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

## 7.44.2 Member Function Documentation

### 7.44.2.1 getName()

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

### 7.44.2.2 getValue()

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

## 7.44.3 Member Data Documentation

### 7.44.3.1 oname

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

## 7.44.3.2 ovalue

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

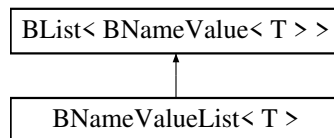
The documentation for this class was generated from the following file:

- [BNameValue.h](#)

## 7.45 BNameValueList&lt; T &gt; Class Template Reference

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

Inheritance diagram for BNameValueList< T >:



## Public Member Functions

- [T \\* find](#) (BString name)
- [Blter findPos](#) (BString name)

## Additional Inherited Members

## 7.45.1 Member Function Documentation

## 7.45.1.1 find()

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

## 7.45.1.2 findPos()

```
template<class T >
Blter BNameValueList< T >::findPos (
    BString name ) [inline]
```

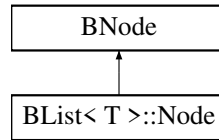
The documentation for this class was generated from the following file:

- [BNameValue.h](#)

## 7.46 BNode Class Reference

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

Inheritance diagram for BNode:



### Public Member Functions

- [BNode\(\)](#)

### Public Attributes

- [BNode\\*](#) `next`
- [BNode\\*](#) `prev`

### 7.46.1 Constructor & Destructor Documentation

#### 7.46.1.1 BNode()

```
BNode::BNode ( ) [inline]
```

### 7.46.2 Member Data Documentation

#### 7.46.2.1 next

```
BNode\* BNode::next
```

#### 7.46.2.2 prev

```
BNode\* BNode::prev
```

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

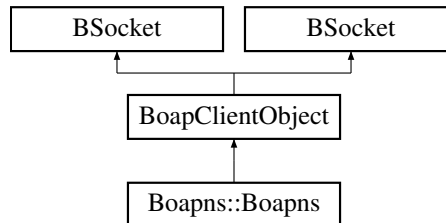
- [BList.h](#)



## 7.47 BoapClientObject Class Reference

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

Inheritance diagram for BoapClientObject:



### Public Member Functions

- [BoapClientObject](#) (BString name="")
- virtual [~BoapClientObject](#) ()
- [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.*
- virtual [BError handleReconnect](#) (BError err)  
*Handle a reconnect performing autorisaztion if required.*
- [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](#) `oclock`
- `int` `otimeout`
- `int` `oreconnect`

*Handle an automatic reconnect on timeout.*

## Additional Inherited Members

### 7.47.1 Constructor & Destructor Documentation

#### 7.47.1.1 `BoapClientObject()` [1/2]

```
BoapClientObject::BoapClientObject (
    BString name = "" )
```

#### 7.47.1.2 `~BoapClientObject()`

```
BoapClientObject::~~BoapClientObject ( ) [virtual]
```

#### 7.47.1.3 `BoapClientObject()` [2/2]

```
BoapClientObject::BoapClientObject (
    BString name )
```

### 7.47.2 Member Function Documentation

#### 7.47.2.1 `checkApiVersion()`

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

### 7.47.2.2 connectService() [1/2]

```
BError BoapClientObject::connectService (
    BString name )
```

Connects to the named service.

### 7.47.2.3 connectService() [2/2]

```
BError BoapClientObject::connectService (
    BString name )
```

### 7.47.2.4 disconnectService()

```
BError BoapClientObject::disconnectService ( )
```

Disconnects from the named service.

### 7.47.2.5 getServiceName()

```
BString BoapClientObject::getServiceName ( )
```

Get the name of the service.

### 7.47.2.6 handleReconnect()

```
BError BoapClientObject::handleReconnect (
    BError err ) [protected], [virtual]
```

Handle a reconnect performing autorisazion if required.

### 7.47.2.7 performCall() [1/2]

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

Performs a RPC call to the named service.

#### 7.47.2.8 performCall() [2/2]

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

#### 7.47.2.9 performRecv() [1/2]

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

Performs a receive.

#### 7.47.2.10 performRecv() [2/2]

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

#### 7.47.2.11 performSend() [1/2]

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

Performs a send to the named service.

#### 7.47.2.12 performSend() [2/2]

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

#### 7.47.2.13 ping()

```
BError BoapClientObject::ping (  
    BUInt32 & apiVersion )
```

Pings the connection and finds the remotes version number.

#### 7.47.2.14 pingLocked()

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

#### 7.47.2.15 setConnectionPriority()

```
BError BoapClientObject::setConnectionPriority (
    BoapPriority priority )
```

Sets the connection priority.

#### 7.47.2.16 setMaxLength()

```
void BoapClientObject::setMaxLength (
    BUInt32 maxLength )
```

Sets the maximum packet length.

#### 7.47.2.17 setTimeout()

```
void BoapClientObject::setTimeout (
    int timeout )
```

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

### 7.47.3 Member Data Documentation

#### 7.47.3.1 oapiVersion

```
BUInt32 BoapClientObject::oapiVersion [protected]
```

#### 7.47.3.2 oconnected

```
int BoapClientObject::oconnected [protected]
```

#### 7.47.3.3 olock

`BMutex` `BoapClientObject::olock` [protected]

#### 7.47.3.4 omaxLength

`BUInt32` `BoapClientObject::omaxLength` [protected]

#### 7.47.3.5 oname

`BString` `BoapClientObject::oname` [protected]

#### 7.47.3.6 opriority

`BoapPriority` `BoapClientObject::opriority` [protected]

#### 7.47.3.7 oreconnect

`int` `BoapClientObject::oreconnect` [protected]

Handle an automatic reconnect on timeout.

#### 7.47.3.8 orx

`BoapPacket` `BoapClientObject::orx` [protected]

#### 7.47.3.9 oservice

`BoapService` `BoapClientObject::oservice` [protected]

## 7.47.3.10 otimeout

```
int BoapClientObject::otimeout [protected]
```

## 7.47.3.11 otx

```
BoapPacket BoapClientObject::otx [protected]
```

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.48 Boapns::BoapEntry Class Reference

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

## Public Member Functions

- [BoapEntry](#) ([BString](#) pname=[BString](#)(), [BString](#) phostName=[BString](#)(), [BList](#)< [BString](#) > addressList=[BList](#)< [BString](#) >(), [BUInt32](#) pport=[BUInt32](#)(), [BUInt32](#) pservice=[BUInt32](#)())

## Public Attributes

- [BString](#) name
- [BString](#) hostName
- [BList](#)< [BString](#) > addressList
- [BUInt32](#) port
- [BUInt32](#) service

## 7.48.1 Constructor &amp; Destructor Documentation

## 7.48.1.1 BoapEntry()

```
Boapns::BoapEntry::BoapEntry (
    BString pname = BString(),
    BString phostName = BString(),
    BList< BString > addressList = BList<BString >(),
    BUInt32 pport = BUInt32(),
    BUInt32 pservice = BUInt32() )
```

## 7.48.2 Member Data Documentation

### 7.48.2.1 addressList

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

### 7.48.2.2 hostName

`BString Boapns::BoapEntry::hostName`

### 7.48.2.3 name

`BString Boapns::BoapEntry::name`

### 7.48.2.4 port

`BUInt32 Boapns::BoapEntry::port`

### 7.48.2.5 service

`BUInt32 Boapns::BoapEntry::service`

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

- [BoapnsD.h](#)
- [BoapnsD.cpp](#)

## 7.49 BoapFuncEntry Class Reference

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

### Public Member Functions

- [BoapFuncEntry](#) (int cmd, [BoapFunc](#) func)
- [BoapFuncEntry](#) (int cmd, [BoapFunc](#) func)



## Public Attributes

- [BUInt32 ocmd](#)
- [BoapFunc ofunc](#)
- [UInt32 ocmd](#)

## 7.49.1 Constructor & Destructor Documentation

### 7.49.1.1 BoapFuncEntry() [1/2]

```
BoapFuncEntry::BoapFuncEntry (
    int cmd,
    BoapFunc func )
```

### 7.49.1.2 BoapFuncEntry() [2/2]

```
BoapFuncEntry::BoapFuncEntry (
    int cmd,
    BoapFunc func )
```

## 7.49.2 Member Data Documentation

### 7.49.2.1 ocmd [1/2]

```
UInt32 BoapFuncEntry::ocmd
```

### 7.49.2.2 ocmd [2/2]

```
BUInt32 BoapFuncEntry::ocmd
```

### 7.49.2.3 ofunc

```
BoapFunc BoapFuncEntry::ofunc
```

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.50 BoapMc1Comms Class Reference

```
#include <BoapMc1.h>
```

### Public Member Functions

- [BoapMc1Comms](#) ([Bool](#) threaded=0, [BUInt](#) reqSize=512)
- virtual [~BoapMc1Comms](#) ()
- void [setCommsMode](#) ([Bool](#) halfDuplex)
  - Sets half duplex mode.*
- void [setComms](#) ([BComms](#) &comms)
  - Sets the communications interface to use.*
- void [setComms](#) ([BComms](#) \*comms)
  - Sets the communications interface to use.*
- void [setAddress](#) ([BUInt16](#) addressTo, [BUInt16](#) addressFrom)
  - Sets the to and from addresses.*
- [BUInt32](#) [getApiVersion](#) ()
  - Returns the API version.*
- [BUInt32](#) [setTimeout](#) ([BUInt32](#) timeoutUs)
  - Sets the call timeout returning the current value.*
- [BoapMc1Packet](#) \* [packetRx](#) ()
  - returns a reference to the current RX packet*
- virtual [BError](#) [processRx](#) ([BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - Process any RX packets queuing them as needed.*

### Protected Member Functions

- virtual [BError](#) [processRequests](#) ([BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - Check and process any requests.*
- virtual [BError](#) [processRequest](#) ([BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - Check and process any request.*
- [BError](#) [packetTx](#) ([BDataChunk](#) \*chunks, [BUInt](#) nChunks, [BUInt16](#) waitCmdReply)
- [BError](#) [packetRxData](#) (void \*data, [BUInt](#) nBytes)
- [BError](#) [packetRxEnd](#) ()

### Protected Attributes

- [Bool](#) othreaded
  - Threaded operation.*
- [BUInt32](#) oreqSize
  - The maximum request size.*
- [BMutex](#) olockCall
  - Lock for RPC calls. Only one at a time.*
- [BMutex](#) olockTx
  - Lock for TX.*
- [BComms](#) \* ocomms
- [BUInt32](#) oapiVersion
- [Bool](#) ohalfDuplex
  - Half duplex mode.*

- [BUInt32 otimeout](#)  
*The timeout in us for calls.*
- [BUInt16 oaddressTo](#)
- [BUInt16 oaddressFrom](#)
- [BoapMc1Packet opacketRxBase](#)
- [BoapMc1Packet \\* opacketRx](#)  
*The RX packet.*
- [BoapMc1Packet opacketTxBase](#)
- [BoapMc1Packet \\* opacketTx](#)  
*The TX packet.*
- [BUInt opacketRpcCmd](#)  
*Waiting for RPC reply to cmd.*
- [BSemaphore opacketRpcSema](#)  
*Wait RPC reply semaphore.*
- [BSemaphore opacketRpcDoneSema](#)  
*Wait RPC complete semaphore.*
- [BoapMc1Error oerror](#)  
*The call return error;.*

## 7.50.1 Constructor & Destructor Documentation

### 7.50.1.1 BoapMc1Comms()

```
BoapMc1Comms::BoapMc1Comms (
    Bool threaded = 0,
    BUInt reqSize = 512 )
```

### 7.50.1.2 ~BoapMc1Comms()

```
BoapMc1Comms::~~BoapMc1Comms ( ) [virtual]
```

## 7.50.2 Member Function Documentation

### 7.50.2.1 getApiVersion()

```
BUInt32 BoapMc1Comms::getApiVersion ( )
```

Returns the API version.

### 7.50.2.2 packetRx()

```
BoapMclPacket * BoapMclComms::packetRx ( )
```

returns a reference to the current RX packet

### 7.50.2.3 packetRxData()

```
BError BoapMclComms::packetRxData (
    void * data,
    BUInt nBytes ) [protected]
```

### 7.50.2.4 packetRxEnd()

```
BError BoapMclComms::packetRxEnd ( ) [protected]
```

### 7.50.2.5 packetTx()

```
BError BoapMclComms::packetTx (
    BDataChunk * chunks,
    BUInt nChunks,
    BUInt16 waitCmdReply ) [protected]
```

### 7.50.2.6 processRequest()

```
BError BoapMclComms::processRequest (
    BTimeout timeoutUs = BTimeoutForever ) [protected], [virtual]
```

Check and process any request.

### 7.50.2.7 processRequests()

```
BError BoapMclComms::processRequests (
    BTimeout timeoutUs = BTimeoutForever ) [protected], [virtual]
```

Check and process any requests.

### 7.50.2.8 processRx()

```
BError BoapMc1Comms::processRx (
    BTimeout timeoutUs = BTimeoutForever ) [virtual]
```

Process any RX packets queuing them as needed.

### 7.50.2.9 setAddress()

```
void BoapMc1Comms::setAddress (
    BUInt16 addressTo,
    BUInt16 addressFrom )
```

Sets the to and from addresses.

### 7.50.2.10 setComms() [1/2]

```
void BoapMc1Comms::setComms (
    BComms & comms )
```

Sets the communications interface to use.

### 7.50.2.11 setComms() [2/2]

```
void BoapMc1Comms::setComms (
    BComms * comms )
```

Sets the communications interface to use.

### 7.50.2.12 setCommsMode()

```
void BoapMc1Comms::setCommsMode (
    Bool halfDuplex )
```

Sets half duplex mode.

### 7.50.2.13 setTimeout()

```
BUInt32 BoapMclComms::setTimeout (
    BUInt32 timeoutUs )
```

Sets the call timeout returning the current value.

## 7.50.3 Member Data Documentation

### 7.50.3.1 oaddressFrom

```
BUInt16 BoapMclComms::oaddressFrom [protected]
```

### 7.50.3.2 oaddressTo

```
BUInt16 BoapMclComms::oaddressTo [protected]
```

### 7.50.3.3 oapiVersion

```
BUInt32 BoapMclComms::oapiVersion [protected]
```

### 7.50.3.4 ocomms

```
BComms* BoapMclComms::ocomms [protected]
```

### 7.50.3.5 oerror

```
BoapMclError BoapMclComms::oerror [protected]
```

The call return error;.

### 7.50.3.6 ohalfDuplex

`Bool` BoapMc1Comms::ohalfDuplex [protected]

Half duplex mode.

### 7.50.3.7 olockCall

`BMutex` BoapMc1Comms::olockCall [protected]

Lock for RPC calls. Only one at a time.

### 7.50.3.8 olockTx

`BMutex` BoapMc1Comms::olockTx [protected]

Lock for TX.

### 7.50.3.9 opacketRpcCmd

`BUInt` BoapMc1Comms::opacketRpcCmd [protected]

Waiting for RPC reply to cmd.

### 7.50.3.10 opacketRpcDoneSema

`BSemaphore` BoapMc1Comms::opacketRpcDoneSema [protected]

Wait RPC complete semaphore.

### 7.50.3.11 opacketRpcSema

`BSemaphore` BoapMc1Comms::opacketRpcSema [protected]

Wait RPC reply semaphore.

### 7.50.3.12 opacketRx

[BoapMclPacket](#)\* BoapMclComms::opacketRx [protected]

The RX packet.

### 7.50.3.13 opacketRxBase

[BoapMclPacket](#) BoapMclComms::opacketRxBase [protected]

### 7.50.3.14 opacketTx

[BoapMclPacket](#)\* BoapMclComms::opacketTx [protected]

The TX packet.

### 7.50.3.15 opacketTxBase

[BoapMclPacket](#) BoapMclComms::opacketTxBase [protected]

### 7.50.3.16 oreqSize

[BUInt32](#) BoapMclComms::oreqSize [protected]

The maximum request size.

### 7.50.3.17 othreaded

[Bool](#) BoapMclComms::othreaded [protected]

Threaded operation.



### 7.50.3.18 otimeout

`BUInt32 BoapMc1Comms::otimeout` [protected]

The timeout in us for calls.

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

- [BoapMc1.h](#)
- [BoapMc1.cpp](#)

## 7.51 BoapMc1Error Struct Reference

```
#include <BoapMc1.h>
```

### Public Attributes

- [BInt16 number](#)  
*The error number.*
- `char` [string](#) [32]  
*The error string.*

### 7.51.1 Member Data Documentation

#### 7.51.1.1 number

`BInt16 BoapMc1Error::number`

The error number.

#### 7.51.1.2 string

`char BoapMc1Error::string[32]`

The error string.

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

- [BoapMc1.h](#)

## 7.52 BoapMc1Packet Class Reference

```
#include <BoapMc1.h>
```

### Public Attributes

- [BoapMc1PacketHead](#) head
- char data [8]

### 7.52.1 Member Data Documentation

#### 7.52.1.1 data

```
char BoapMc1Packet::data[8]
```

#### 7.52.1.2 head

```
BoapMc1PacketHead BoapMc1Packet::head
```

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

- [BoapMc1.h](#)

## 7.53 BoapMc1PacketHead Struct Reference

```
#include <BoapMc1.h>
```

### Public Attributes

- [BUInt16](#) magic
- [BUInt16](#) length
- [BUInt16](#) addressTo
  - Do we want addresses ?*
- [BUInt16](#) addressFrom
- [BUInt16](#) cmd
- [BInt16](#) error
  - Not used. Maybe return a [BError](#) object ?*
- [BUInt32](#) checksum
  - Not used. Not sure we want this here or at the very end of the packet.*

## 7.53.1 Member Data Documentation

### 7.53.1.1 addressFrom

`BUInt16` BoapMc1PacketHead::addressFrom

### 7.53.1.2 addressTo

`BUInt16` BoapMc1PacketHead::addressTo

Do we want addresses ?

### 7.53.1.3 checksum

`BUInt32` BoapMc1PacketHead::checksum

Not used. Not sure we want this here or at the very end of the packet.

### 7.53.1.4 cmd

`BUInt16` BoapMc1PacketHead::cmd

### 7.53.1.5 error

`BInt16` BoapMc1PacketHead::error

Not used. Maybe return a `BError` object ?

### 7.53.1.6 length

`BUInt16` BoapMc1PacketHead::length

### 7.53.1.7 magic

```
BUInt16 BoapMc1PacketHead::magic
```

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

- [BoapMc1.h](#)

## 7.54 BoapMcClientObject Class Reference

```
#include <BoapMc.h>
```

### Public Member Functions

- [BoapMcClientObject](#) ([BComms](#) &comms)
- virtual [~BoapMcClientObject](#) ()
- void [setAddress](#) ([BUInt8](#) addressTo, [BUInt8](#) addressFrom)
- [BUInt32](#) [getApiVersion](#) ()  
*Returns the API version.*

### Protected Member Functions

- [BError](#) [performCall](#) ()  
*Performs a RPC call to the named service.*
- [BError](#) [performSend](#) ()  
*Performs a send to the named service.*
- [BError](#) [performRecv](#) ()  
*Performs a receive.*

### Protected Attributes

- [BUInt32](#) [oapiVersion](#)
- [BComms](#) & [ocomms](#)
- [BUInt8](#) [oaddressTo](#)
- [BUInt8](#) [oaddressFrom](#)
- [BoapMcPacket](#) [opacket](#)

## 7.54.1 Constructor & Destructor Documentation

### 7.54.1.1 BoapMcClientObject()

```
BoapMcClientObject::BoapMcClientObject (
    BComms & comms )
```

### 7.54.1.2 ~BoapMcClientObject()

```
BoapMcClientObject::~~BoapMcClientObject ( ) [virtual]
```

## 7.54.2 Member Function Documentation

### 7.54.2.1 getApiVersion()

```
BUInt32 BoapMcClientObject::getApiVersion ( )
```

Returns the API version.

### 7.54.2.2 performCall()

```
BError BoapMcClientObject::performCall ( ) [protected]
```

Performs a RPC call to the named service.

### 7.54.2.3 performRecv()

```
BError BoapMcClientObject::performRecv ( ) [protected]
```

Performs a receive.

### 7.54.2.4 performSend()

```
BError BoapMcClientObject::performSend ( ) [protected]
```

Performs a send to the named service.

### 7.54.2.5 setAddress()

```
void BoapMcClientObject::setAddress (
    BUInt8 addressTo,
    BUInt8 addressFrom )
```

### 7.54.3 Member Data Documentation

#### 7.54.3.1 oaddressFrom

[BUInt8](#) BoapMcClientObject::oaddressFrom [protected]

#### 7.54.3.2 oaddressTo

[BUInt8](#) BoapMcClientObject::oaddressTo [protected]

#### 7.54.3.3 oapiVersion

[BUInt32](#) BoapMcClientObject::oapiVersion [protected]

#### 7.54.3.4 ocomms

[BComms&](#) BoapMcClientObject::ocomms [protected]

#### 7.54.3.5 opacket

[BoapMcPacket](#) BoapMcClientObject::opacket [protected]

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

- [BoapMc.h](#)
- [BoapMc.cpp](#)

## 7.55 BoapMcComms Class Reference

```
#include <BoapMc.h>
```

## Public Member Functions

- [BoapMcComms](#) ([Bool](#) threaded=0, [BUInt](#) rxQueueSize=4)
- virtual [~BoapMcComms](#) ()
- void [setCommsMode](#) ([Bool](#) slave, [BUInt](#) txQueueSize)
  - Sets slave mode.*
- void [setComms](#) ([BComms](#) &comms)
  - Sets the communications interface to use.*
- void [setComms](#) ([BComms](#) \*comms)
  - Sets the communications interface to use.*
- void [setAddress](#) ([BUInt8](#) addressTo, [BUInt8](#) addressFrom)
  - Sets the to and from addresses.*
- [BUInt32](#) [getApiVersion](#) ()
  - Returns the API version.*
- [BUInt32](#) [setTimeout](#) ([BUInt32](#) timeoutUs)
  - Sets the call timeout returning the current value.*
- virtual [BError](#) [processRx](#) ([BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - Process any RX packets queuing them as needed.*
- virtual [BError](#) [processRequests](#) ([BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - Check and process all requests.*
- virtual [BError](#) [processRequest](#) ([BTimeout](#) timeoutUs=[BTimeoutForever](#))
  - Check and process any request.*
- virtual [BError](#) [processPacket](#) ([BoapMcPacket](#) &rx, [BoapMcPacket](#) &tx)
  - Process a recieved packet.*

## Protected Member Functions

- [BError](#) [performCall](#) ()
  - Performs a RPC call to the remote side.*
- [BError](#) [performSend](#) ()
  - Performs a RPC send to the remote side.*
- [BError](#) [packetSend](#) ([BoapMcPacket](#) &packet)
  - Receives a packet.*
- [BError](#) [packetRecv](#) ([BoapMcPacket](#) &packet)
  - Receives a packet.*

## Protected Attributes

- [Bool](#) othreaded
- [BMutex](#) olockCall
  - Lock for RPC calls. Only one at a time.*
- [BMutex](#) olockTx
  - Lock for TX.*
- [BComms](#) \* ocomms
- [BUInt32](#) oapiVersion
- [Bool](#) oslave
  - Set slave mode.*
- [BUInt32](#) otimeout
  - The timeout in us for calls.*
- [BUInt8](#) oaddressTo

- [BUInt8 oaddressFrom](#)
- [BoapMcPacket opacket](#)  
*Packet RX buffer.*
- [BoapMcPacket opacketTx](#)  
*Packet TX buffer for calls.*
- [BoapMcPacket opacketRx](#)  
*Packet RX buffer for calls.*
- [BSemaphore opacketRxSema](#)  
*Wait RX semaphore.*
- [BoapMcPacket opacketReqTx](#)  
*Packet TX buffer for requests.*
- [BoapMcPacket opacketReqRx](#)  
*Packet RX buffer for requests.*
- [BQueue< BoapMcPacket > opacketReqQueue](#)  
*Packet RX buffer queue for requests.*
- [BFifo< BoapMcPacket > opacketTxQueue](#)  
*Packet TX Queue.*
- [BSemaphoreCount opacketTxQueueWriteNum](#)  
*Packet TX Queue number.*
- [BSemaphore opacketTxSema](#)  
*Wait for TX semaphore.*

## 7.55.1 Constructor & Destructor Documentation

### 7.55.1.1 BoapMcComms()

```
BoapMcComms::BoapMcComms (
    Bool threaded = 0,
    BUInt rxQueueSize = 4 )
```

### 7.55.1.2 ~BoapMcComms()

```
BoapMcComms::~~BoapMcComms ( ) [virtual]
```

## 7.55.2 Member Function Documentation

### 7.55.2.1 getApiVersion()

```
BUInt32 BoapMcComms::getApiVersion ( )
```

Returns the API version.



### 7.55.2.2 packetRecv()

```
BError BoapMcComms::packetRecv (
    BoapMcPacket & packet ) [protected]
```

Receives a packet.

### 7.55.2.3 packetSend()

```
BError BoapMcComms::packetSend (
    BoapMcPacket & packet ) [protected]
```

Receives a packet.

### 7.55.2.4 performCall()

```
BError BoapMcComms::performCall ( ) [protected]
```

Performs a RPC call to the remote side.

### 7.55.2.5 performSend()

```
BError BoapMcComms::performSend ( ) [protected]
```

Performs a RPC send to the remote side.

### 7.55.2.6 processPacket()

```
BError BoapMcComms::processPacket (
    BoapMcPacket & rx,
    BoapMcPacket & tx ) [virtual]
```

Process a recieved packet.

### 7.55.2.7 processRequest()

```
BError BoapMcComms::processRequest (
    BTimeout timeoutUs = BTimeoutForever ) [virtual]
```

Check and process any request.

### 7.55.2.8 processRequests()

```
BError BoapMcComms::processRequests (
    BTimeout timeoutUs = BTimeoutForever ) [virtual]
```

Check and process all requests.

### 7.55.2.9 processRx()

```
BError BoapMcComms::processRx (
    BTimeout timeoutUs = BTimeoutForever ) [virtual]
```

Process any RX packets queuing them as needed.

!!! This should wait on comms for timeoutUs !!!

### 7.55.2.10 setAddress()

```
void BoapMcComms::setAddress (
    BUInt8 addressTo,
    BUInt8 addressFrom )
```

Sets the to and from addresses.

### 7.55.2.11 setComms() [1/2]

```
void BoapMcComms::setComms (
    BComms & comms )
```

Sets the communications interface to use.

### 7.55.2.12 setComms() [2/2]

```
void BoapMcComms::setComms (
    BComms * comms )
```

Sets the communications interface to use.

#### 7.55.2.13 setCommsMode()

```
void BoapMcComms::setCommsMode (
    Bool slave,
    BUInt txQueueSize )
```

Sets slave mode.

#### 7.55.2.14 setTimeout()

```
BUInt32 BoapMcComms::setTimeout (
    BUInt32 timeoutUs )
```

Sets the call timeout returning the current value.

### 7.55.3 Member Data Documentation

#### 7.55.3.1 oaddressFrom

```
BUInt8 BoapMcComms::oaddressFrom [protected]
```

#### 7.55.3.2 oaddressTo

```
BUInt8 BoapMcComms::oaddressTo [protected]
```

#### 7.55.3.3 oapiVersion

```
BUInt32 BoapMcComms::oapiVersion [protected]
```

#### 7.55.3.4 ocomms

```
BComms* BoapMcComms::ocomms [protected]
```

### 7.55.3.5 olockCall

[BMutex](#) BoapMcComms::olockCall [protected]

Lock for RPC calls. Only one at a time.

### 7.55.3.6 olockTx

[BMutex](#) BoapMcComms::olockTx [protected]

Lock for TX.

### 7.55.3.7 opacket

[BoapMcPacket](#) BoapMcComms::opacket [protected]

Packet RX buffer.

### 7.55.3.8 opacketReqQueue

[BQueue<BoapMcPacket>](#) BoapMcComms::opacketReqQueue [protected]

Packet RX buffer queue for requests.

### 7.55.3.9 opacketReqRx

[BoapMcPacket](#) BoapMcComms::opacketReqRx [protected]

Packet RX buffer for requests.

### 7.55.3.10 opacketReqTx

[BoapMcPacket](#) BoapMcComms::opacketReqTx [protected]

Packet TX buffer for requests.

### 7.55.3.11 opacketRx

[BoapMcPacket](#) BoapMcComms::opacketRx [protected]

Packet RX buffer for calls.

### 7.55.3.12 opacketRxSema

[BSemaphore](#) BoapMcComms::opacketRxSema [protected]

Wait RX semaphore.

### 7.55.3.13 opacketTx

[BoapMcPacket](#) BoapMcComms::opacketTx [protected]

Packet TX buffer for calls.

### 7.55.3.14 opacketTxQueue

[BFifo<BoapMcPacket>](#) BoapMcComms::opacketTxQueue [protected]

Packet TX Queue.

### 7.55.3.15 opacketTxQueueWriteNum

[BSemaphoreCount](#) BoapMcComms::opacketTxQueueWriteNum [protected]

Packet TX Queue number.

### 7.55.3.16 opacketTxSema

[BSemaphore](#) BoapMcComms::opacketTxSema [protected]

Wait for TX semaphore.

### 7.55.3.17 oslave

`Bool BoapMcComms::oslave [protected]`

Set slave mode.

### 7.55.3.18 othreaded

`Bool BoapMcComms::othreaded [protected]`

### 7.55.3.19 otimeout

`BUInt32 BoapMcComms::otimeout [protected]`

The timeout in us for calls.

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

- [BoapMc.h](#)
- [BoapMc.cpp](#)

## 7.56 BoapMcPacket Class Reference

```
#include <BoapMc.h>
```

### Public Attributes

- [BoapMcPacketHead head](#)
- char [data](#) [256 - sizeof([BoapMcPacketHead](#))]

### 7.56.1 Member Data Documentation

#### 7.56.1.1 data

```
char BoapMcPacket::data[256 - sizeof(BoapMcPacketHead)]
```

### 7.56.1.2 head

`BoapMcPacketHead` `BoapMcPacket::head`

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

- [BoapMc.h](#)

## 7.57 BoapMcPacketHead Struct Reference

```
#include <BoapMc.h>
```

### Public Attributes

- [BUInt8 length](#)
- [BUInt8 addressTo](#)
- [BUInt8 addressFrom](#)
- [BUInt8 cmd](#)
- [BUInt16 error](#)
- [BUInt16 checksum](#)

### 7.57.1 Member Data Documentation

#### 7.57.1.1 addressFrom

`BUInt8` `BoapMcPacketHead::addressFrom`

#### 7.57.1.2 addressTo

`BUInt8` `BoapMcPacketHead::addressTo`

#### 7.57.1.3 checksum

`BUInt16` `BoapMcPacketHead::checksum`

#### 7.57.1.4 cmd

[BUInt8](#) BoapMcPacketHead::cmd

#### 7.57.1.5 error

[BUInt16](#) BoapMcPacketHead::error

#### 7.57.1.6 length

[BUInt8](#) BoapMcPacketHead::length

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

- [BoapMc.h](#)

## 7.58 BoapMcServiceObject Class Reference

```
#include <BoapMc.h>
```

### Public Member Functions

- [BoapMcServiceObject](#) ()
- virtual [~BoapMcServiceObject](#) ()
- virtual [BError](#) process ([BoapMcPacket](#) &rx, [BoapMcPacket](#) &tx)
- virtual [BError](#) processEvent ([BoapMcPacket](#) &rx)

### Protected Member Functions

- [BError](#) sendEvent ([BoapMcPacket](#) &tx)

### Protected Attributes

- [BUInt32](#) oapiVersion

### 7.58.1 Constructor & Destructor Documentation



### 7.58.1.1 BoapMcServiceObject()

```
BoapMcServiceObject::BoapMcServiceObject ( )
```

### 7.58.1.2 ~BoapMcServiceObject()

```
BoapMcServiceObject::~BoapMcServiceObject ( ) [virtual]
```

## 7.58.2 Member Function Documentation

### 7.58.2.1 process()

```
BError BoapMcServiceObject::process (
    BoapMcPacket & rx,
    BoapMcPacket & tx ) [virtual]
```

### 7.58.2.2 processEvent()

```
BError BoapMcServiceObject::processEvent (
    BoapMcPacket & rx ) [virtual]
```

### 7.58.2.3 sendEvent()

```
BError BoapMcServiceObject::sendEvent (
    BoapMcPacket & tx ) [protected]
```

## 7.58.3 Member Data Documentation

### 7.58.3.1 oapiVersion

```
BUInt32 BoapMcServiceObject::oapiVersion [protected]
```

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

- [BoapMc.h](#)
- [BoapMc.cpp](#)

## 7.59 BoapMcSignalObject Class Reference

```
#include <BoapMc.h>
```

### Public Member Functions

- [BoapMcSignalObject](#) ([BComms](#) &comms)

### Protected Member Functions

- [BError performSend](#) ([BoapMcPacket](#) &tx)

### Protected Attributes

- [BComms](#) & ocomms

## 7.59.1 Constructor & Destructor Documentation

### 7.59.1.1 BoapMcSignalObject()

```
BoapMcSignalObject::BoapMcSignalObject (
    BComms & comms )
```

## 7.59.2 Member Function Documentation

### 7.59.2.1 performSend()

```
BError BoapMcSignalObject::performSend (
    BoapMcPacket & tx ) [protected]
```

## 7.59.3 Member Data Documentation

## 7.59.3.1 ocomms

`BComms& BoapMcSignalObject::ocomms` [protected]

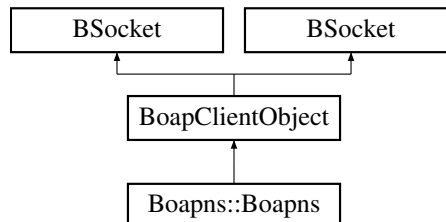
The documentation for this class was generated from the following files:

- [BoapMc.h](#)
- [BoapMc.cpp](#)

## 7.60 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`)

## Additional Inherited Members

## 7.60.1 Constructor &amp; Destructor Documentation

## 7.60.1.1 Boapns()

```
Boapns::Boapns::Boapns (
    BString name = "" )
```

## 7.60.2 Member Function Documentation

### 7.60.2.1 addEntry()

```
BError Boapns::Boapns::addEntry (
    BoapEntry entry )
```

### 7.60.2.2 delEntry()

```
BError Boapns::Boapns::delEntry (
    BString name )
```

### 7.60.2.3 getEntry()

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

### 7.60.2.4 getEntryList()

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

### 7.60.2.5 getNewName()

```
BError Boapns::Boapns::getNewName (
    BString & name )
```

### 7.60.2.6 getVersion()

```
BError Boapns::Boapns::getVersion (
    BString & version )
```

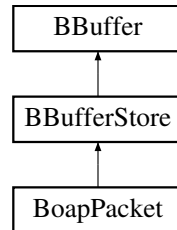
The documentation for this class was generated from the following files:

- [BoapnsC.h](#)
- [BoapnsC.cpp](#)

## 7.61 BoapPacket Class Reference

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

Inheritance diagram for BoapPacket:



### Public Member Functions

- [BoapPacket \(\)](#)
- [~BoapPacket \(\)](#)
- [BUInt32 getCmd \(\)](#)
- [int peekHead \(BoapPacketHead &head\)](#)
- [int pushHead \(BoapPacketHead &head\)](#)
- [int popHead \(BoapPacketHead &head\)](#)
- [void updateHead \(\)](#)
- [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 [updateLen](#) ()

## Private Attributes

- int [osize](#)
- int [onbytes](#)
- char \* [odata](#)
- int [opos](#)

## Additional Inherited Members

### 7.61.1 Constructor & Destructor Documentation

#### 7.61.1.1 [BoapPacket\(\)](#) [1/2]

`BoapPacket::BoapPacket ( )`

#### 7.61.1.2 [~BoapPacket\(\)](#) [1/2]

`BoapPacket::~~BoapPacket ( )`

#### 7.61.1.3 [BoapPacket\(\)](#) [2/2]

`BoapPacket::BoapPacket ( )`

#### 7.61.1.4 [~BoapPacket\(\)](#) [2/2]

`BoapPacket::~~BoapPacket ( )`

### 7.61.2 Member Function Documentation

### 7.61.2.1 data()

```
char * BoapPacket::data ( )
```

### 7.61.2.2 getCmd()

```
BUInt32 BoapPacket::getCmd ( )
```

### 7.61.2.3 nbytes()

```
int BoapPacket::nbytes ( )
```

### 7.61.2.4 peekHead()

```
int BoapPacket::peekHead (
    BoapPacketHead & head )
```

### 7.61.2.5 pop() [1/10]

```
int BoapPacket::pop (
    Int8 & v )
```

### 7.61.2.6 pop() [2/10]

```
int BoapPacket::pop (
    UInt8 & v )
```

### 7.61.2.7 pop() [3/10]

```
int BoapPacket::pop (
    Int16 & v )
```

**7.61.2.8 pop()** [4/10]

```
int BoapPacket::pop (
    UInt16 & v )
```

**7.61.2.9 pop()** [5/10]

```
int BoapPacket::pop (
    Int32 & v )
```

**7.61.2.10 pop()** [6/10]

```
int BoapPacket::pop (
    UInt32 & v )
```

**7.61.2.11 pop()** [7/10]

```
int BoapPacket::pop (
    BString & v )
```

**7.61.2.12 pop()** [8/10]

```
int BoapPacket::pop (
    Double & v )
```

**7.61.2.13 pop()** [9/10]

```
int BoapPacket::pop (
    BError & v )
```

**7.61.2.14 pop()** [10/10]

```
int BoapPacket::pop (
    UInt32 nBytes,
    void * data )
```



**7.61.2.15 popHead()** [1/2]

```
int BoapPacket::popHead (
    BoapPacketHead & head )
```

**7.61.2.16 popHead()** [2/2]

```
int BoapPacket::popHead (
    BoapPacketHead & head )
```

**7.61.2.17 push()** [1/10]

```
int BoapPacket::push (
    Int8 v )
```

**7.61.2.18 push()** [2/10]

```
int BoapPacket::push (
    UInt8 v )
```

**7.61.2.19 push()** [3/10]

```
int BoapPacket::push (
    Int16 v )
```

**7.61.2.20 push()** [4/10]

```
int BoapPacket::push (
    UInt16 v )
```

**7.61.2.21 push()** [5/10]

```
int BoapPacket::push (
    Int32 v )
```

**7.61.2.22 push()** [6/10]

```
int BoapPacket::push (
    UInt32 v )
```

**7.61.2.23 push()** [7/10]

```
int BoapPacket::push (
    BString & v )
```

**7.61.2.24 push()** [8/10]

```
int BoapPacket::push (
    Double v )
```

**7.61.2.25 push()** [9/10]

```
int BoapPacket::push (
    BError & v )
```

**7.61.2.26 push()** [10/10]

```
int BoapPacket::push (
    UInt32 nBytes,
    const void * data )
```

**7.61.2.27 pushHead()** [1/2]

```
int BoapPacket::pushHead (
    BoapPacketHead & head )
```

**7.61.2.28 pushHead()** [2/2]

```
int BoapPacket::pushHead (
    BoapPacketHead & head )
```

#### 7.61.2.29 `resize()`

```
int BoapPacket::resize (
    int size )
```

#### 7.61.2.30 `setData()`

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

#### 7.61.2.31 `updateHead()`

```
void BoapPacket::updateHead ( )
```

#### 7.61.2.32 `updateLen()`

```
void BoapPacket::updateLen ( ) [private]
```

### 7.61.3 Member Data Documentation

#### 7.61.3.1 `odata`

```
char* BoapPacket::odata [private]
```

#### 7.61.3.2 `onbytes`

```
int BoapPacket::onbytes [private]
```

#### 7.61.3.3 `opos`

```
int BoapPacket::opos [private]
```

#### 7.61.3.4 osize

```
int BoapPacket::osize [private]
```

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.62 BoapPacketHead Struct Reference

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

### Public Attributes

- [BUInt32 type](#)
- [BUInt32 length](#)
- [BUInt32 service](#)
- [BUInt32 cmd](#)
- [UInt32 length](#)
- [BoapType type](#)
- [BoapService service](#)
- [UInt32 cmd](#)
- [UInt32 reserved \[12\]](#)

### 7.62.1 Member Data Documentation

#### 7.62.1.1 cmd [1/2]

[UInt32](#) BoapPacketHead::cmd

#### 7.62.1.2 cmd [2/2]

[BUInt32](#) BoapPacketHead::cmd

**7.62.1.3 length** [1/2]

[UInt32](#) BoapPacketHead::length

**7.62.1.4 length** [2/2]

[BUInt32](#) BoapPacketHead::length

**7.62.1.5 reserved**

[UInt32](#) BoapPacketHead::reserved[12]

**7.62.1.6 service** [1/2]

[BoapService](#) BoapPacketHead::service

**7.62.1.7 service** [2/2]

[BUInt32](#) BoapPacketHead::service

**7.62.1.8 type** [1/2]

[BUInt32](#) BoapPacketHead::type

**7.62.1.9 type** [2/2]

[BoapType](#) BoapPacketHead::type

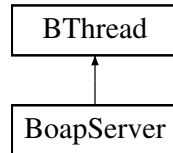
The documentation for this struct was generated from the following files:

- [Boap.h](#)
- [BoapSimple.h](#)

## 7.63 BoapServer Class Reference

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

Inheritance diagram for BoapServer:



### Public Types

- enum { `NOTHEADS =0`, `THREADED =1` }

### Public Member Functions

- `BoapServer ()`
- virtual `~BoapServer ()`
- virtual `BError init (BString boapNsHost="", int port=0, int threaded=0, int isBoapns=0)`
- virtual `BError run (int inThread=0)`
- virtual `BError process (BoapServerConnection *conn, BoapPacket &rx, BoapPacket &tx)`
- virtual `BError processEvent (BoapPacket &rx)`
- virtual `BError addObject (BoapServiceObject *object)`
- virtual `BError sendEvent (BoapPacket &tx)`
- virtual `BError processEvent (int fd)`
- virtual void `clientGone (BoapServerConnection *client)`
- `BSocket & getSocket ()`
- `BSocket & getEventSocket ()`
- `BString getHostName ()`
- int `getConnectionsNumber ()`
- void `closeConnections ()`
- virtual `BoapServerConnection * newConnection (int fd, BSocketAddressINET address)`
- `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 ()`

### Public Attributes

- `BUInt64 onumOperations`

## Private Member Functions

- void \* [function](#) ()

## Private Attributes

- [BMutex](#) olock
- int othreaded
- int oisBoapns
- [Boapns::Boapns](#) \* oboapns
- [BList](#)< [BoapServerConnection](#) \* > oclients
- [BEvent1Int](#) oclientGoneEvent
- [BList](#)< [BoapServiceEntry](#) > oservices
- [BPoll](#) opoll
- [BSocket](#) onet
- [BSocket](#) onetEvent
- [BSocketAddressINET](#) onetEventAddress
- [BString](#) ohostName
- int oboapNs
- [BoapPacket](#) orx
- [BoapPacket](#) otx

## 7.63.1 Member Enumeration Documentation

### 7.63.1.1 anonymous enum

anonymous enum

#### Enumerator

NOTHEADS	
THREADED	

## 7.63.2 Constructor & Destructor Documentation

### 7.63.2.1 BoapServer() [1/2]

[BoapServer::BoapServer](#) ( )

### 7.63.2.2 ~BoapServer()

```
BoapServer::~~BoapServer ( ) [virtual]
```

### 7.63.2.3 BoapServer() [2/2]

```
BoapServer::BoapServer ( )
```

## 7.63.3 Member Function Documentation

### 7.63.3.1 addObject() [1/2]

```
BError BoapServer::addObject (
    BoapServiceObject * object )
```

### 7.63.3.2 addObject() [2/2]

```
BError BoapServer::addObject (
    BoapServiceObject * object ) [virtual]
```

### 7.63.3.3 clientGone()

```
void BoapServer::clientGone (
    BoapServerConnection * client ) [virtual]
```

### 7.63.3.4 closeConnections()

```
void BoapServer::closeConnections ( )
```

### 7.63.3.5 function()

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

Reimplemented from [BThread](#).



### 7.63.3.6 getConnectionsNumber()

```
int BoapServer::getConnectionsNumber ( )
```

### 7.63.3.7 getEventSocket() [1/2]

```
BSocket& BoapServer::getEventSocket ( )
```

### 7.63.3.8 getEventSocket() [2/2]

```
BSocket & BoapServer::getEventSocket ( )
```

### 7.63.3.9 getHostName() [1/2]

```
BString BoapServer::getHostName ( )
```

### 7.63.3.10 getHostName() [2/2]

```
BString BoapServer::getHostName ( )
```

### 7.63.3.11 getSocket() [1/2]

```
BSocket& BoapServer::getSocket ( )
```

### 7.63.3.12 getSocket() [2/2]

```
BSocket & BoapServer::getSocket ( )
```

### 7.63.3.13 init() [1/2]

```
BError BoapServer::init (
    int boapNs = 0 )
```

#### 7.63.3.14 `init()` [2/2]

```
BError BoapServer::init (
    BString boapNsHost = "",
    int port = 0,
    int threaded = 0,
    int isBoapns = 0 ) [virtual]
```

#### 7.63.3.15 `newConnection()`

```
BoapServerConnection * BoapServer::newConnection (
    int fd,
    BSocketAddressINET address ) [virtual]
```

#### 7.63.3.16 `process()` [1/2]

```
BError BoapServer::process (
    int fd )
```

#### 7.63.3.17 `process()` [2/2]

```
BError BoapServer::process (
    BoapServerConnection * conn,
    BoapPacket & rx,
    BoapPacket & tx ) [virtual]
```

#### 7.63.3.18 `processEvent()` [1/4]

```
BError BoapServer::processEvent (
    BoapPacket & rx )
```

#### 7.63.3.19 `processEvent()` [2/4]

```
BError BoapServer::processEvent (
    int fd )
```

**7.63.3.20 processEvent()** [3/4]

```
BError BoapServer::processEvent (
    BoapPacket & rx ) [virtual]
```

**7.63.3.21 processEvent()** [4/4]

```
BError BoapServer::processEvent (
    int fd ) [virtual]
```

**7.63.3.22 run()** [1/2]

```
BError BoapServer::run ( )
```

**7.63.3.23 run()** [2/2]

```
BError BoapServer::run (
    int inThread = 0 ) [virtual]
```

**7.63.3.24 sendEvent()** [1/2]

```
BError BoapServer::sendEvent (
    BoapPacket & tx )
```

**7.63.3.25 sendEvent()** [2/2]

```
BError BoapServer::sendEvent (
    BoapPacket & tx ) [virtual]
```

**7.63.4 Member Data Documentation**

#### 7.63.4.1 oboapNs

```
int BoapServer::oboapNs [private]
```

#### 7.63.4.2 oboapns

```
Boapns::Boapns* BoapServer::oboapns [private]
```

#### 7.63.4.3 oclientGoneEvent

```
BEvent1Int BoapServer::oclientGoneEvent [private]
```

#### 7.63.4.4 oclients

```
BList<BoapServerConnection*> BoapServer::oclients [private]
```

#### 7.63.4.5 ohostName

```
BString BoapServer::ohostName [private]
```

#### 7.63.4.6 oisBoapns

```
int BoapServer::oisBoapns [private]
```

#### 7.63.4.7 olock

```
BMutex BoapServer::olock [private]
```

#### 7.63.4.8 onet

```
BSocket BoapServer::onet [private]
```

#### 7.63.4.9 onetEvent

`BSocket` BoapServer::onetEvent [private]

#### 7.63.4.10 onetEventAddress

`BSocketAddressINET` BoapServer::onetEventAddress [private]

#### 7.63.4.11 onumOperations

`BUInt64` BoapServer::onumOperations

#### 7.63.4.12 opoll

`BPoll` BoapServer::opoll [private]

#### 7.63.4.13 orx

`BoapPacket` BoapServer::orx [private]

#### 7.63.4.14 oservices

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

#### 7.63.4.15 othreaded

`int` BoapServer::othreaded [private]

## 7.63.4.16 otx

```
BoapPacket BoapServer::otx [private]
```

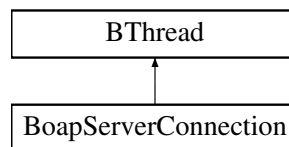
The documentation for this class was generated from the following files:

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.64 BoapServerConnection Class Reference

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

Inheritance diagram for BoapServerConnection:



### Public Member Functions

- [BoapServerConnection](#) ([BoapServer](#) &boapServer, int fd)
- virtual [~BoapServerConnection](#) ()
- virtual [BError](#) [init](#) ()  
*Initialise connection.*
- virtual [BError](#) [process](#) ()
- virtual [BSocket](#) & [getSocket](#) ()
- virtual void [setMaxLength](#) ([BUInt32](#) maxLength)
- virtual [BError](#) [getHead](#) ([BoapPacketHead](#) &head)
- virtual [BError](#) [validate](#) ()  
*Validate the connection.*

### Private Member Functions

- void \* [function](#) ()

### Private Attributes

- [BoapServer](#) & oboapServer
- [BSocket](#) osocket
- [BoapPacket](#) orx
- [BoapPacket](#) otx
- [BUInt32](#) omaxLength

## 7.64.1 Constructor & Destructor Documentation

### 7.64.1.1 BoapServerConnection()

```
BoapServerConnection::BoapServerConnection (
    BoapServer & boapServer,
    int fd )
```

### 7.64.1.2 ~BoapServerConnection()

```
BoapServerConnection::~BoapServerConnection ( ) [virtual]
```

## 7.64.2 Member Function Documentation

### 7.64.2.1 function()

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

Reimplemented from [BThread](#).

### 7.64.2.2 getHead()

```
BError BoapServerConnection::getHead (
    BoapPacketHead & head ) [virtual]
```

### 7.64.2.3 getSocket()

```
BSocket & BoapServerConnection::getSocket ( ) [virtual]
```

### 7.64.2.4 init()

```
BError BoapServerConnection::init ( ) [virtual]
```

Initialise connection.

#### 7.64.2.5 process()

`BError` BoapServerConnection::process ( ) [virtual]

#### 7.64.2.6 setMaxLength()

void BoapServerConnection::setMaxLength (   
 `BUInt32` *maxLength* ) [virtual]

#### 7.64.2.7 validate()

`BError` BoapServerConnection::validate ( ) [virtual]

Validate the connection.

### 7.64.3 Member Data Documentation

#### 7.64.3.1 oboapServer

`BoapServer`& BoapServerConnection::oboapServer [private]

#### 7.64.3.2 omaxLength

`BUInt32` BoapServerConnection::omaxLength [private]

#### 7.64.3.3 orx

`BoapPacket` BoapServerConnection::orx [private]

#### 7.64.3.4 osocket

`BSocket` BoapServerConnection::osocket [private]



## 7.64.3.5 otx

```
BoapPacket BoapServerConnection::otx [private]
```

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

- [Boap.h](#)
- [Boap.cpp](#)

## 7.65 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

### 7.65.1 Constructor & Destructor Documentation

#### 7.65.1.1 BoapServiceEntry() [1/2]

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

#### 7.65.1.2 BoapServiceEntry() [2/2]

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

### 7.65.2 Member Data Documentation

### 7.65.2.1 oobject

[BoapServiceObject](#) \* [BoapServiceEntry::oobject](#)

### 7.65.2.2 oservice

[BoapService](#) [BoapServiceEntry::oservice](#)

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

- [Boap.h](#)
- [BoapSimple.h](#)

## 7.66 BoapServiceObject Class Reference

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

### Public Member Functions

- [BoapServiceObject](#) ([BoapServer](#) &server, [BString](#) name="")
- virtual [~BoapServiceObject](#) ()
- [BError setName](#) ([BString](#) name)
- [BError sendEvent](#) ([BString](#) signalName, [BInt32](#) arg)
- virtual [BError processEvent](#) ([BString](#) objectName, [BString](#) name, [BInt32](#) 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

## 7.66.1 Constructor & Destructor Documentation

### 7.66.1.1 BoapServiceObject() [1/2]

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

### 7.66.1.2 ~BoapServiceObject() [1/2]

```
BoapServiceObject::~BoapServiceObject ( ) [virtual]
```

### 7.66.1.3 BoapServiceObject() [2/2]

```
BoapServiceObject::BoapServiceObject (
    BoapServer & server,
    BString name )
```

### 7.66.1.4 ~BoapServiceObject() [2/2]

```
virtual BoapServiceObject::~BoapServiceObject ( ) [virtual]
```

## 7.66.2 Member Function Documentation

### 7.66.2.1 doConnectionPriority()

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

### 7.66.2.2 doPing()

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

### 7.66.2.3 name() [1/2]

```
BString BoapServiceObject::name ( )
```

### 7.66.2.4 name() [2/2]

```
BString BoapServiceObject::name ( )
```

### 7.66.2.5 process() [1/2]

```
BError BoapServiceObject::process (
    BoapPacket & rx,
    BoapPacket & tx )
```

### 7.66.2.6 process() [2/2]

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

### 7.66.2.7 processEvent() [1/4]

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

**7.66.2.8 processEvent()** [2/4]

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

**7.66.2.9 processEvent()** [3/4]

```
BError BoapServiceObject::processEvent (  
    BString objectName,  
    BString name,  
    BInt32 arg ) [virtual]
```

**7.66.2.10 processEvent()** [4/4]

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

**7.66.2.11 sendEvent()** [1/4]

```
BError BoapServiceObject::sendEvent (  
    BString signalName,  
    Int32 arg )
```

**7.66.2.12 sendEvent()** [2/4]

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

**7.66.2.13 sendEvent()** [3/4]

```
BError BoapServiceObject::sendEvent (  
    BString signalName,  
    BInt32 arg )
```

#### 7.66.2.14 `sendEvent()` [4/4]

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

#### 7.66.2.15 `setName()`

```
BError BoapServiceObject::setName (  
    BString name )
```

### 7.66.3 Member Data Documentation

#### 7.66.3.1 `oapiVersion`

```
BUInt32 BoapServiceObject::oapiVersion [protected]
```

#### 7.66.3.2 `ofuncList`

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

#### 7.66.3.3 `oname`

```
BString BoapServiceObject::oname [protected]
```

#### 7.66.3.4 `oserver`

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

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.67 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](#)

### Additional Inherited Members

#### 7.67.1 Constructor & Destructor Documentation

##### 7.67.1.1 BoapSignalObject() [1/2]

```
BoapSignalObject::BoapSignalObject ( )
```

##### 7.67.1.2 BoapSignalObject() [2/2]

```
BoapSignalObject::BoapSignalObject ( )
```

## 7.67.2 Member Function Documentation

### 7.67.2.1 performSend() [1/2]

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

### 7.67.2.2 performSend() [2/2]

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

## 7.67.3 Member Data Documentation

### 7.67.3.1 orx

```
BoapPacket BoapSignalObject::orx [protected]
```

### 7.67.3.2 otx

```
BoapPacket BoapSignalObject::otx [protected]
```

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.68 BObj Class Reference

```
#include <BObj.h>
```



## Public Member Functions

- [BObj](#) ()
- virtual [~BObj](#) ()
- virtual const char \* [getType](#) () const
- virtual const [BObjMember](#) \* [getMembers](#) () const
- virtual [BError](#) [getMembers](#) ([BDictString](#) &members)
- virtual [BError](#) [getMember](#) ([BString](#) name, [BString](#) &value)
- virtual [BError](#) [setMembers](#) ([BDictString](#) &members)
- virtual [BError](#) [setMember](#) ([BString](#) name, [BString](#) value)
- virtual void [membersPrint](#) () const
  - Prints out members.*
- virtual [BString](#) [getDebugString](#) ()
  - Returns contents as a debug string.*

### 7.68.1 Constructor & Destructor Documentation

#### 7.68.1.1 BObj()

```
BObj::BObj ( )
```

#### 7.68.1.2 ~BObj()

```
BObj::~~BObj ( ) [virtual]
```

### 7.68.2 Member Function Documentation

#### 7.68.2.1 getDebugString()

```
BString BObj::getDebugString ( ) [virtual]
```

Returns contents as a debug string.

#### 7.68.2.2 getMember()

```
BError BObj::getMember (
    BString name,
    BString & value ) [virtual]
```

### 7.68.2.3 `getMembers()` [1/2]

```
const BObjMember * BObj::getMembers ( ) const [virtual]
```

### 7.68.2.4 `getMembers()` [2/2]

```
BError BObj::getMembers (
    BDictString & members ) [virtual]
```

### 7.68.2.5 `getType()`

```
const char * BObj::getType ( ) const [virtual]
```

### 7.68.2.6 `membersPrint()`

```
void BObj::membersPrint ( ) const [virtual]
```

Prints out members.

### 7.68.2.7 `setMember()`

```
BError BObj::setMember (
    BString name,
    BString value ) [virtual]
```

### 7.68.2.8 `setMembers()`

```
BError BObj::setMembers (
    BDictString & members ) [virtual]
```

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

- [BObj.h](#)
- [BObj.cpp](#)

## 7.69 BObjMember Struct Reference

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

### Public Attributes

- [BType](#) type
- [BTypeComp](#) typeComp
- [BUInt16](#) dataOffset
- [BUInt16](#) size
- const char \* [typeName](#)
- const char \* [name](#)

### 7.69.1 Member Data Documentation

#### 7.69.1.1 dataOffset

```
BUInt16 BObjMember::dataOffset
```

#### 7.69.1.2 name

```
const char* BObjMember::name
```

#### 7.69.1.3 size

```
BUInt16 BObjMember::size
```

#### 7.69.1.4 type

```
BType BObjMember::type
```

#### 7.69.1.5 typeComp

```
BTypeComp BObjMember::typeComp
```

### 7.69.1.6 typeName

```
const char* BObjMember::typeName
```

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

- [BTypes.h](#)

## 7.70 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.*
- [BError](#) [doPollEvents](#) (int &fd, int &events, int timeoutUs=-1)  
*Perform polling operation and return events.*
- 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.*

### 7.70.1 Detailed Description

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

### 7.70.2 Member Typedef Documentation

#### 7.70.2.1 PollFd

```
typedef struct pollfd BPoll::PollFd
```

### 7.70.3 Constructor & Destructor Documentation

#### 7.70.3.1 BPoll()

```
BPoll::BPoll ( )
```

#### 7.70.3.2 ~BPoll()

```
BPoll::~~BPoll ( )
```

### 7.70.4 Member Function Documentation

#### 7.70.4.1 append()

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

Append a file descriptor to polling list.

#### 7.70.4.2 clear()

```
void BPoll::clear ( )
```

#### 7.70.4.3 delFd()

```
void BPoll::delFd (
    int fd )
```

Remove a file descriptor from polling list.

#### 7.70.4.4 doPoll()

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

Perform polling operation.

#### 7.70.4.5 doPollEvents()

```
BError BPoll::doPollEvents (
    int & fd,
    int & events,
    int timeoutUs = -1 )
```

Perform polling operation and return events.

#### 7.70.4.6 getPollFds()

```
BPoll::PollFd * BPoll::getPollFds ( )
```

#### 7.70.4.7 getPollFdsNum()

```
int BPoll::getPollFdsNum ( )
```

#### 7.70.4.8 nextFd()

```
int BPoll::nextFd (
    int i ) [private]
```

## 7.70.5 Member Data Documentation

### 7.70.5.1 ofds

```
PollFd* BPoll::ofds [private]
```

The list of poll fd's.

### 7.70.5.2 ofdsNext

```
int BPoll::ofdsNext [private]
```

The next list entry for round robin polling.

### 7.70.5.3 ofdsNum

```
int BPoll::ofdsNum [private]
```

The number of FD's in list.

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

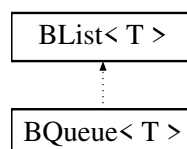
- [BPoll.h](#)
- [BPoll.cpp](#)

## 7.71 BQueue< T > Class Template Reference

Queue class.

```
#include <BQueue.h>
```

Inheritance diagram for BQueue< T >:



## Public Member Functions

- [BQueue](#) ([BUInt](#) size)
- [~BQueue](#) ()
- void [clear](#) ()  
*Clear the queue.*
- [BUInt](#) [writeAvailable](#) () const
- [BError](#) [write](#) (const T &v, [BTimeout](#) timeout=[BTimeoutForever](#))  
*Append an item onto the queue.*
- [BUInt](#) [readAvailable](#) () const
- [BError](#) [read](#) (T &v, [BTimeout](#) timeout=[BTimeoutForever](#))  
*Get an item from the queue.*

## Private Attributes

- [BMutex](#) olock
- [BUInt](#) osize
- [BCondInt](#) onumber

## Additional Inherited Members

### 7.71.1 Detailed Description

```
template<class T>
class BQueue< T >
```

Queue class.

### 7.71.2 Constructor & Destructor Documentation

#### 7.71.2.1 BQueue()

```
template<class T >
BQueue< T >::BQueue (
    BUInt size )
```

#### 7.71.2.2 ~BQueue()

```
template<class T >
BQueue< T >::~BQueue ( )
```



### 7.71.3 Member Function Documentation

#### 7.71.3.1 clear()

```
template<class T >
void BQueue< T >::clear ( ) [virtual]
```

Clear the queue.

Reimplemented from [BList< T >](#).

#### 7.71.3.2 read()

```
template<class T>
BError BQueue< T >::read (
    T & v,
    BTimeout timeout = BTimeoutForever )
```

Get an item from the queue.

#### 7.71.3.3 readAvailable()

```
template<class T >
BUInt BQueue< T >::readAvailable ( ) const
```

#### 7.71.3.4 write()

```
template<class T>
BError BQueue< T >::write (
    const T & v,
    BTimeout timeout = BTimeoutForever )
```

Append an item onto the queue.

#### 7.71.3.5 writeAvailable()

```
template<class T >
BUInt BQueue< T >::writeAvailable ( ) const
```

## 7.71.4 Member Data Documentation

### 7.71.4.1 olock

```
template<class T>
BMutex BQueue< T >::olock [private]
```

### 7.71.4.2 onumber

```
template<class T>
BCondInt BQueue< T >::onumber [private]
```

### 7.71.4.3 osize

```
template<class T>
BUInt BQueue< T >::osize [private]
```

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

- [BQueue.h](#)

## 7.72 BRefData Class Reference

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

### Public Member Functions

- [BRefData](#) ()
- [BRefData](#) (int len)
- [BRefData](#) (const [BRefData](#) &refData)
- [~BRefData](#) ()
- [BRefData](#) \* [copy](#) ()  
*Create a copy of this reference for writing, if necessary.*
- [BRefData](#) \* [addRef](#) ()  
*Increment the reference counter.*
- int [deleteRef](#) ()  
*Decrement the reference counter.*
- char \* [data](#) ()  
*Return the raw data pointer.*
- int [len](#) ()  
*Return the length in bytes.*
- [BRefData](#) & [operator=](#) (const [BRefData](#) &refData)
- void [setLen](#) (int len)  
*Set the length in bytes. Note should only be used if `orefCount = 1`.*

## Private Attributes

- [BAtomicCount](#) `orefCount`  
*The reference count, how many users.*
- `int` `olen`  
*The actual length of data in `oData`.*
- `void *` `odata`  
*Pointer to the data.*

### 7.72.1 Detailed Description

Referenced data storage. This is Thread safe to a degree. The reference counting is protected. However, [setLen\(\)](#) is not and should be protected at a higher level.

### 7.72.2 Constructor & Destructor Documentation

#### 7.72.2.1 BRefData() [1/3]

```
BRefData::BRefData ( )
```

#### 7.72.2.2 BRefData() [2/3]

```
BRefData::BRefData (
    int len )
```

#### 7.72.2.3 BRefData() [3/3]

```
BRefData::BRefData (
    const BRefData & refData )
```

#### 7.72.2.4 ~BRefData()

```
BRefData::~BRefData ( )
```

### 7.72.3 Member Function Documentation

### 7.72.3.1 addRef()

```
BRefData * BRefData::addRef ( )
```

Increment the reference counter.

### 7.72.3.2 copy()

```
BRefData * BRefData::copy ( )
```

Create a copy of this reference for writing, if necessary.

### 7.72.3.3 data()

```
char* BRefData::data ( ) [inline]
```

Return the raw data pointer.

### 7.72.3.4 deleteRef()

```
int BRefData::deleteRef ( )
```

Decrement the reference counter.

### 7.72.3.5 len()

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

Return the length in bytes.

### 7.72.3.6 operator=()

```
BRefData & BRefData::operator= (
    const BRefData & refData )
```

#### 7.72.3.7 setLen()

```
void BRefData::setLen (
    int len )
```

Set the length in bytes. Note should only be used if orefCount = 1.

### 7.72.4 Member Data Documentation

#### 7.72.4.1 odata

```
void* BRefData::odata [private]
```

Pointer to the data.

#### 7.72.4.2 olen

```
int BRefData::olen [private]
```

The actual length of data in oData.

#### 7.72.4.3 orefCount

```
BAtomicCount BRefData::orefCount [private]
```

The reference count, how many users.

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

- [BRefData.h](#)
- [BRefData.cpp](#)

## 7.73 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](#)

### 7.73.1 Detailed Description

Realtime clock.

### 7.73.2 Constructor & Destructor Documentation

#### 7.73.2.1 BRtc()

```
BRtc::BRtc ( )
```

#### 7.73.2.2 ~BRtc()

```
BRtc::~~BRtc ( )
```

### 7.73.3 Member Function Documentation

#### 7.73.3.1 init()

```
BError BRtc::init (
    int rate )
```

Setup interrupt rate.

## 7.73.3.2 wait()

```
void BRtc::wait (
    int delayUs )
```

Wait specified uS.

## 7.73.4 Member Data Documentation

## 7.73.4.1 ofd

```
int BRtc::ofd [private]
```

## 7.73.4.2 orate

```
int BRtc::orate [private]
```

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

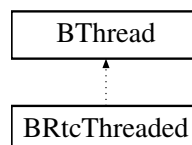
- [BRtc.h](#)
- [BRtc.cpp](#)

## 7.74 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`

### 7.74.1 Detailed Description

Threaded real time clock.

### 7.74.2 Constructor & Destructor Documentation

#### 7.74.2.1 BRtcThreaded()

```
BRtcThreaded::BRtcThreaded ( )
```

#### 7.74.2.2 ~BRtcThreaded()

```
BRtcThreaded::~~BRtcThreaded ( )
```

### 7.74.3 Member Function Documentation

#### 7.74.3.1 function()

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

Reimplemented from [BThread](#).



### 7.74.3.2 init()

```
BError BRtcThreaded::init (
    int rate )
```

Setup interrupt rate.

### 7.74.3.3 wait()

```
void BRtcThreaded::wait (
    int delayUs )
```

Wait specified uS.

## 7.74.4 Member Data Documentation

### 7.74.4.1 ocond

```
BCond BRtcThreaded::ocond [private]
```

### 7.74.4.2 orate

```
int BRtcThreaded::orate [private]
```

### 7.74.4.3 orte

```
BRTC BRtcThreaded::orte [private]
```

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

- [BRtc.h](#)
- [BRtc.cpp](#)

## 7.75 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](#)

### 7.75.1 Detailed Description

thread read-write locks

### 7.75.2 Constructor & Destructor Documentation

#### 7.75.2.1 [BRWLock\(\)](#) [1/2]

```
BRWLock::BRWLock ( )
```

#### 7.75.2.2 [BRWLock\(\)](#) [2/2]

```
BRWLock::BRWLock (
    const BRWLock & rwlock )
```

#### 7.75.2.3 [~BRWLock\(\)](#)

```
BRWLock::~~BRWLock ( )
```

### 7.75.3 Member Function Documentation

#### 7.75.3.1 operator=()

```
BRWLock & BRWLock::operator= (
    const BRWLock & rlock )
```

#### 7.75.3.2 rdLock()

```
int BRWLock::rdLock ( )
```

Set lock, wait if necessary.

#### 7.75.3.3 tryRdLock()

```
int BRWLock::tryRdLock ( )
```

Test the lock.

#### 7.75.3.4 tryWrLock()

```
int BRWLock::tryWrLock ( )
```

Test the lock.

#### 7.75.3.5 unlock()

```
int BRWLock::unlock ( )
```

Unlock the lock.

#### 7.75.3.6 wrLock()

```
int BRWLock::wrLock ( )
```

Set lock, wait if necessary.

## 7.75.4 Member Data Documentation

### 7.75.4.1 olock

```
pthread_rwlock_t BRWLock::olock [private]
```

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

- [BRWLock.h](#)
- [BRWLock.cpp](#)

## 7.76 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](#)

### 7.76.1 Detailed Description

Sempahore class.

### 7.76.2 Constructor & Destructor Documentation

### 7.76.2.1 BSemaphore() [1/2]

```
BSemaphore::BSemaphore (
    int value = 0 )
```

### 7.76.2.2 BSemaphore() [2/2]

```
BSemaphore::BSemaphore (
    const BSemaphore & sema )
```

### 7.76.2.3 ~BSemaphore()

```
BSemaphore::~BSemaphore ( )
```

## 7.76.3 Member Function Documentation

### 7.76.3.1 getValue()

```
int BSemaphore::getValue ( ) const
```

### 7.76.3.2 operator=()

```
BSemaphore & BSemaphore::operator= (
    const BSemaphore & sema )
```

### 7.76.3.3 post()

```
int BSemaphore::post ( )
```

Post condition.

#### 7.76.3.4 timedWait()

```
int BSema::timedWait (
    int timeUs )
```

Wait for condition with timeout.

#### 7.76.3.5 tryWait()

```
int BSema::tryWait ( )
```

Test for the condition.

#### 7.76.3.6 wait()

```
int BSema::wait ( )
```

Wait for contition.

### 7.76.4 Member Data Documentation

#### 7.76.4.1 osema

```
sem_t BSema::osema [private]
```

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

- [BSema.h](#)
- [BSema.cpp](#)

## 7.77 BSemaphore Class Reference

Semaphore class.

```
#include <BSemaphore.h>
```

## Public Member Functions

- [BSemaphore](#) ()
- [BSemaphore](#) (const [BSemaphore](#) &semaphore)
- [~BSemaphore](#) ()
- [Bool](#) [wait](#) ([BTimeout](#) timeoutUs=[BTimeoutForever](#))  
*Wait for the semaphore.*
- void [set](#) ()  
*Set the semaphore.*
- int [getValue](#) () const
- [BSemaphore](#) & [operator=](#) (const [BSemaphore](#) &semaphore)

## Private Attributes

- `sem_t` [osema](#)

### 7.77.1 Detailed Description

Semaphore class.

### 7.77.2 Constructor & Destructor Documentation

#### 7.77.2.1 [BSemaphore](#)() [1/2]

```
BSemaphore::BSemaphore ( )
```

#### 7.77.2.2 [BSemaphore](#)() [2/2]

```
BSemaphore::BSemaphore (
    const BSemaphore & semaphore )
```

#### 7.77.2.3 [~BSemaphore](#)()

```
BSemaphore::~~BSemaphore ( )
```

### 7.77.3 Member Function Documentation

### 7.77.3.1 getValue()

```
int BSemaphore::getValue ( ) const
```

### 7.77.3.2 operator=()

```
BSemaphore & BSemaphore::operator= (
    const BSemaphore & semaphore )
```

### 7.77.3.3 set()

```
void BSemaphore::set ( )
```

Set the semaphore.

### 7.77.3.4 wait()

```
Bool BSemaphore::wait (
    BTimeout timeoutUs = BTimeoutForever )
```

Wait for the semaphore.

## 7.77.4 Member Data Documentation

### 7.77.4.1 osema

```
sem_t BSemaphore::osema [private]
```

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

- [BSemaphore.h](#)
- [BSemaphore.cpp](#)

## 7.78 BSemaphoreBool Class Reference

```
#include <BSemaphore.h>
```



## Public Member Functions

- [BSemaphoreBool](#) ()
- [BSemaphoreBool](#) (const [BSemaphoreBool](#) &semaphore)
- [~BSemaphoreBool](#) ()
- void [set](#) ([Bool](#) on=1)
- void [clear](#) ()
- [Bool](#) [wait](#) ([Bool](#) v=1, [BTimeout](#) timeoutUs=[BTimeoutForever](#))  
*Wait for the semaphore.*
- [Bool](#) [value](#) ()
- [operator int](#) ()
- int [operator==](#) ([Bool](#) on)
- [BSemaphoreBool](#) & [operator=](#) ([Bool](#) on)

## Private Attributes

- [BSemaphore](#) osema
- volatile [Bool](#) ovalue

## 7.78.1 Constructor & Destructor Documentation

### 7.78.1.1 BSemaphoreBool() [1/2]

```
BSemaphoreBool::BSemaphoreBool ( )
```

### 7.78.1.2 BSemaphoreBool() [2/2]

```
BSemaphoreBool::BSemaphoreBool (
    const BSemaphoreBool & semaphore )
```

### 7.78.1.3 ~BSemaphoreBool()

```
BSemaphoreBool::~~BSemaphoreBool ( )
```

## 7.78.2 Member Function Documentation

### 7.78.2.1 clear()

```
void BSemaphoreBool::clear ( )
```

### 7.78.2.2 operator int()

```
BSemaphoreBool::operator int ( )
```

### 7.78.2.3 operator=()

```
BSemaphoreBool & BSemaphoreBool::operator= (
    Bool on )
```

### 7.78.2.4 operator==( )

```
int BSemaphoreBool::operator==(
    Bool on )
```

### 7.78.2.5 set()

```
void BSemaphoreBool::set (
    Bool on = 1 )
```

### 7.78.2.6 value()

```
Bool BSemaphoreBool::value ( )
```

### 7.78.2.7 wait()

```
Bool BSemaphoreBool::wait (
    Bool v = 1,
    BTimeout timeoutUs = BTimeoutForever )
```

Wait for the semaphore.

### 7.78.3 Member Data Documentation

#### 7.78.3.1 osema

```
BSemaphore BSemaphoreBool::osema [private]
```

#### 7.78.3.2 ovalue

```
volatile Bool BSemaphoreBool::ovalue [private]
```

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

- [BSemaphore.h](#)
- [BSemaphore.cpp](#)

## 7.79 BSemaphoreCount Class Reference

```
#include <BSemaphore.h>
```

### Public Member Functions

- [BSemaphoreCount](#) ()
- [BSemaphoreCount](#) (const [BSemaphoreCount](#) &semaphore)
- [~BSemaphoreCount](#) ()
- void [setValue](#) (BUInt v)
- void [add](#) (int v=1)  
*Set the semaphore.*
- [Bool](#) [wait](#) (BUInt v=1, [BTimeout](#) timeoutUs=[BTimeoutForever](#))  
*Wait for the semaphore.*
- [Bool](#) [take](#) (BUInt v=1, [BTimeout](#) timeoutUs=[BTimeoutForever](#))  
*Take for the semaphore.*
- [BUInt](#) [value](#) ()
- [BSemaphoreCount](#) & [operator=](#) (const [BSemaphoreCount](#) &semaphore)

### Private Attributes

- [BMutex](#) olock
- [BSemaphore](#) osema
- volatile [BUInt](#) ovalue

## 7.79.1 Constructor & Destructor Documentation

### 7.79.1.1 BSemaphoreCount() [1/2]

```
BSemaphoreCount::BSemaphoreCount ( )
```

### 7.79.1.2 BSemaphoreCount() [2/2]

```
BSemaphoreCount::BSemaphoreCount (
    const BSemaphoreCount & semaphore )
```

### 7.79.1.3 ~BSemaphoreCount()

```
BSemaphoreCount::~~BSemaphoreCount ( )
```

## 7.79.2 Member Function Documentation

### 7.79.2.1 add()

```
void BSemaphoreCount::add (
    int v = 1 )
```

Set the semaphore.

### 7.79.2.2 operator=()

```
BSemaphoreCount & BSemaphoreCount::operator= (
    const BSemaphoreCount & semaphore )
```

### 7.79.2.3 setValue()

```
void BSemaphoreCount::setValue (
    BUInt v )
```

#### 7.79.2.4 take()

```
Bool BSemaphoreCount::take (
    BUInt v = 1,
    BTimeout timeoutUs = BTimeoutForever )
```

Take for the semaphore.

#### 7.79.2.5 value()

```
BUInt BSemaphoreCount::value ( )
```

#### 7.79.2.6 wait()

```
Bool BSemaphoreCount::wait (
    BUInt v = 1,
    BTimeout timeoutUs = BTimeoutForever )
```

Wait for the semaphore.

### 7.79.3 Member Data Documentation

#### 7.79.3.1 olock

```
BMutex BSemaphoreCount::olock [private]
```

#### 7.79.3.2 osema

```
BSemaphore BSemaphoreCount::osema [private]
```

#### 7.79.3.3 ovalue

```
volatile BUInt BSemaphoreCount::ovalue [private]
```

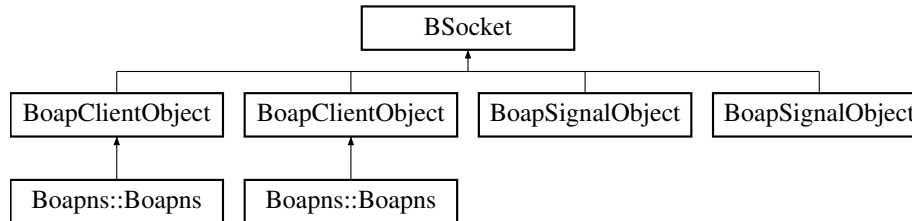
The documentation for this class was generated from the following files:

- [BSemaphore.h](#)
- [BSemaphore.cpp](#)

## 7.80 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](#) (int domain, int type, int protocol)
- [~BSocket](#) ()
- [BError](#) [init](#) (int domain, int type, int protocol)
- [BError](#) [init](#) (NType type)
- void [setFd](#) (int fd)
- 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](#) [sendChunks](#) (const [BDataChunk](#) \*chunks, [BSize](#) nChunks, [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)
- [BUInt](#) [recvAvailable](#) ()
- [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`

## 7.80.1 Member Enumeration Documentation

### 7.80.1.1 NType

```
enum BSocket::NType
```

#### Enumerator

STREAM	
DGRAM	

### 7.80.1.2 Priority

```
enum BSocket::Priority
```

#### Enumerator

PriorityLow	
PriorityNormal	
PriorityHigh	

## 7.80.2 Constructor & Destructor Documentation

### 7.80.2.1 BSocket() [1/4]

```
BSocket::BSocket ( )
```

### 7.80.2.2 BSocket() [2/4]

```
BSocket::BSocket (
    int fd )
```

### 7.80.2.3 BSocket() [3/4]

```
BSocket::BSocket (
    NType type )
```

### 7.80.2.4 BSocket() [4/4]

```
BSocket::BSocket (
    int domain,
    int type,
    int protocol )
```

### 7.80.2.5 ~BSocket()

```
BSocket::~~BSocket ( )
```

## 7.80.3 Member Function Documentation

### 7.80.3.1 accept() [1/2]

```
BError BSocket::accept (
    int & fd )
```

### 7.80.3.2 accept() [2/2]

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

### 7.80.3.3 bind()

```
BError BSocket::bind (
    const BSocketAddress & add )
```



#### 7.80.3.4 close()

```
BError BSocket::close ( )
```

#### 7.80.3.5 connect()

```
BError BSocket::connect (
    const BSocketAddress & add )
```

#### 7.80.3.6 getAddress()

```
BError BSocket::getAddress (
    BSocketAddress & address )
```

#### 7.80.3.7 getFd()

```
int BSocket::getFd ( )
```

#### 7.80.3.8 getMTU()

```
BError BSocket::getMTU (
    uint32_t & mtu )
```

#### 7.80.3.9 getSockOpt()

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

#### 7.80.3.10 init() [1/2]

```
BError BSocket::init (
    int domain,
    int type,
    int protocol )
```

### 7.80.3.11 `init()` [2/2]

```
BError BSocket::init (
    NType type )
```

### 7.80.3.12 `listen()`

```
BError BSocket::listen (
    int backlog = 5 )
```

### 7.80.3.13 `recv()`

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

### 7.80.3.14 `recvAvailable()`

```
BUInt BSocket::recvAvailable ( )
```

### 7.80.3.15 `recvFrom()`

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

### 7.80.3.16 `recvFromWithTimeout()`

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

### 7.80.3.17 recvWithTimeout()

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

### 7.80.3.18 send()

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

### 7.80.3.19 sendChunks()

```
BError BSocket::sendChunks (
    const BDataChunk * chunks,
    BSize nChunks,
    BSize & nbytesSent,
    int flags = 0 )
```

### 7.80.3.20 sendTo()

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

### 7.80.3.21 setBroadCast()

```
BError BSocket::setBroadCast (
    int on )
```

### 7.80.3.22 setFd()

```
void BSocket::setFd (
    int fd )
```

### 7.80.3.23 setPriority()

```
BError BSocket::setPriority (
    Priority priority )
```

### 7.80.3.24 setReuseAddress()

```
BError BSocket::setReuseAddress (
    int on )
```

### 7.80.3.25 setSockOpt()

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

### 7.80.3.26 shutdown()

```
BError BSocket::shutdown (
    int how )
```

## 7.80.4 Member Data Documentation

### 7.80.4.1 osocket

```
int BSocket::osocket [private]
```

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

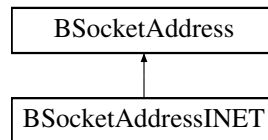
- [BSocket.h](#)
- [BSocket.cpp](#)

## 7.81 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
- [BString](#) getString () const
  - *Return string version of address <ip>:<port>*
- [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

#### 7.81.1 Detailed Description

Socket Address.

#### 7.81.2 Member Typedef Documentation

### 7.81.2.1 SockAddr

```
typedef struct sockaddr BSocketAddress::SockAddr
```

## 7.81.3 Constructor & Destructor Documentation

### 7.81.3.1 BSocketAddress() [1/3]

```
BSocketAddress::BSocketAddress ( )
```

### 7.81.3.2 BSocketAddress() [2/3]

```
BSocketAddress::BSocketAddress (
    const BSocketAddress & add )
```

### 7.81.3.3 BSocketAddress() [3/3]

```
BSocketAddress::BSocketAddress (
    SockAddr * address,
    int len )
```

### 7.81.3.4 ~BSocketAddress()

```
BSocketAddress::~BSocketAddress ( )
```

## 7.81.4 Member Function Documentation

### 7.81.4.1 getString()

```
BString BSocketAddress::getString ( ) const
```

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

#### 7.81.4.2 len()

```
int BSocketAddress::len ( ) const
```

#### 7.81.4.3 operator const SockAddr \*()

```
BSocketAddress::operator const SockAddr * ( ) const [inline]
```

#### 7.81.4.4 operator"!="()

```
int BSocketAddress::operator!= (
    const BSocketAddress & add ) const
```

#### 7.81.4.5 operator=()

```
BSocketAddress & BSocketAddress::operator= (
    const BSocketAddress & add )
```

#### 7.81.4.6 operator==(())

```
int BSocketAddress::operator==(
    const BSocketAddress & add ) const
```

#### 7.81.4.7 raw()

```
const BSocketAddress::SockAddr * BSocketAddress::raw ( ) const
```

#### 7.81.4.8 set()

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

## 7.81.5 Member Data Documentation

### 7.81.5.1 oaddress

```
SockAddr* BSocketAddress::oaddress [private]
```

### 7.81.5.2 olen

```
int BSocketAddress::olen [private]
```

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

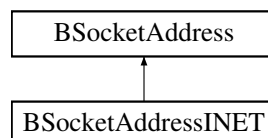
- [BSocket.h](#)
- [BSocket.cpp](#)

## 7.82 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.*

### 7.82.1 Detailed Description

IP aware socket address.

### 7.82.2 Member Typedef Documentation

#### 7.82.2.1 SockAddrIP

```
typedef struct sockaddr_in BSocketAddressINET::SockAddrIP
```

### 7.82.3 Member Function Documentation

#### 7.82.3.1 address()

```
uint32_t BSocketAddressINET::address ( )
```

Returns socket ip address.

#### 7.82.3.2 getHostName()

```
BString BSocketAddressINET::getHostName ( ) [static]
```

Get this hosts network name.

### 7.82.3.3 getIpAddresses()

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

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

### 7.82.3.4 getIpAddressList()

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

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

### 7.82.3.5 getIpAddressListAll()

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

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

### 7.82.3.6 getString()

```
BString BSocketAddressINET::getString ( )
```

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

### 7.82.3.7 port()

```
uint32_t BSocketAddressINET::port ( )
```

Returns socket port.

### 7.82.3.8 set() [1/3]

```
BError BSocketAddressINET::set (
    BString hostName,
    uint32_t port )
```

7.82.3.9 `set()` [2/3]

```
BError BSocketAddressINET::set (
    uint32_t address,
    uint32_t port )
```

7.82.3.10 `set()` [3/3]

```
BError BSocketAddressINET::set (
    BString hostName,
    BString service,
    BString type )
```

7.82.3.11 `setPort()`

```
void BSocketAddressINET::setPort (
    uint32_t port )
```

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

- [BSocket.h](#)
- [BSocket.cpp](#)

## 7.83 BSpI Class Reference

`BSpI` class.

```
#include <BSpI.h>
```

### Public Types

- enum `Mode` { `Mode0` = 0, `Mode1` = 1, `Mode2` = 2, `Mode3` = 3 }

### Public Member Functions

- `BSpI` ()
- `BError` `init` (`BString` devName, `BUInt` speed=1000000, `Mode` mode=`Mode1`, `Bool` csActive=0)
- `BError` `transact` (`BUInt8` dev, void \*txBuf, int txLen, int pad, void \*rxBuf, int rxLen)

### Private Attributes

- `BString` `odevName`
- int `odev`

### 7.83.1 Detailed Description

`BSpi` class.

### 7.83.2 Member Enumeration Documentation

#### 7.83.2.1 Mode

```
enum BSpi::Mode
```

##### Enumerator

Mode0	
Mode1	
Mode2	
Mode3	

### 7.83.3 Constructor & Destructor Documentation

#### 7.83.3.1 BSpi()

```
BSpi::BSpi ( )
```

### 7.83.4 Member Function Documentation

#### 7.83.4.1 init()

```
BError BSpi::init (
    BString devName,
    BUInt speed = 1000000,
    Mode mode = Mode1,
    Bool csActive = 0 )
```

#### 7.83.4.2 transact()

```
BError BSpi::transact (
    BUInt8 dev,
    void * txBuf,
    int txLen,
    int pad,
    void * rxBuf,
    int rxLen )
```

### 7.83.5 Member Data Documentation

#### 7.83.5.1 odev

```
int BSpi::odev [private]
```

#### 7.83.5.2 odevName

```
BString BSpi::odevName [private]
```

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

- [BSpi.h](#)
- [BSpi.cpp](#)

## 7.84 BString Class Reference

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

### Public Member Functions

- [BString \(\)](#)
- [BString \(const BString &string\)](#)
- [BString \(const char \\*str\)](#)
- [BString \(const char \\*str, unsigned int len\)](#)
- [BString \(char ch\)](#)
- [BString \(BInt v\)](#)
- [BString \(BUInt v\)](#)
- [BString \(BUInt64 v\)](#)
- [BString \(double v\)](#)
- [~BString \(\)](#)
- [BString copy \(\) const](#)

*Return an independant copy.*

- 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.*
- unsigned int `retUInt` () const  
*Return string as a int.*
- double `retDouble` () const  
*Return string as a double.*
- `BFloat64 retFloat64` () const  
*Return string as a BFloat64.*
- 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 delimited patterns.*
- int `compareRegex` (const `BString` &`pattern`, int ignoreCase=0) const  
*Compare strings.*
- `BString` & `truncate` (int `len`)  
*Truncate to length len.*
- `BString` & `pad` (int `len`)  
*Pad to length len.*
- void `clear` ()  
*Clear the string.*
- `BString` & `toUpper` ()  
*Convert to uppercase.*
- `BString` & `toLower` ()  
*Convert to lowercase.*
- `BString` `lowerFirst` ()  
*Return string with lowercase first character.*
- void `removeNL` ()  
*Remove if present NL from last char.*
- `BString` `justify` (int `leftMargin`, int `width`)  
*Justify the string to the given width.*
- `BString` `fixedLen` (int `length`, int `rightJustify`=0)  
*return string formatted to fixed length*
- `BString` `firstLine` ()  
*Return first line.*
- `BString` `translateChar` (char `ch`, `BString` `replace`=" ")  
*Translate character converting them to the given string.*
- `BString` `reverse` () const  
*Reverse character order.*
- `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.*
- int `append` (const `BString` &str)
  - Append a string.*
- `BString add` (const `BString` &str) const
  - Add strings returning result.*
- `BString` & `printf` (const char \*fmt,...)
  - Formatted print into the string.*
- int `find` (char ch) const
  - Find ch in string searching forwards.*
- int `find` (`BString` str) const
  - Find string in string searching forwards.*
- int `findReverse` (char ch) const
  - Find ch in string searching backwards.*
- `BString csvEncode` () const
  - Encode a string for CSV.*
- `BString` & `csvDecode` (const `BString` str)
  - Decode a string from CSV.*
- `BString base64Encode` () const
  - Encode a string to base64.*
- `BError base64Decode` (`BString` &str) const
  - Decode a string from base64.*
- `BList< BString > getTokenList` (`BString` separators)
  - Break string into tokens.*
- `BList< BString > getTokenList` (char separator)
  - 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.*
- `BList< BString > split` (char splitChar)
  - Split string into an array based on the character separator.*
- `BString dirname` ()
- `BString basename` ()
- `BString extension` ()
- `BUInt32 hash` () const
- char & `get` (int pos)
- const char & `get` (int pos) const
- `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+` (`BInt` i) const
- `BString operator+` (`BUInt` i) const
- `BString operator+` (`BUInt64` i) const
- `operator const char *` () const
- `BString field` (int field) const
- char \*\* `fields` ()

### Static Public Member Functions

- static `BString convert` (char ch)  
*Converts char to string.*
- static `BString convert` (`BInt` value)  
*Converts int to string.*
- static `BString convert` (`BUInt` value)  
*Converts uint to string.*
- static `BString convert` (double value, int eFormat=0)  
*Converts double to string.*
- static `BString convert` (`BUInt64` value)  
*Converts long long to string.*
- static `BString convertHex` (`BInt` value)  
*Converts int to string as hex value.*
- static `BString convertHex` (`BUInt` 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

#### 7.84.1 Constructor & Destructor Documentation



**7.84.1.1 BString()** [1/9]

```
BString::BString ( )
```

**7.84.1.2 BString()** [2/9]

```
BString::BString (
    const BString & string )
```

**7.84.1.3 BString()** [3/9]

```
BString::BString (
    const char * str )
```

**7.84.1.4 BString()** [4/9]

```
BString::BString (
    const char * str,
    unsigned int len )
```

**7.84.1.5 BString()** [5/9]

```
BString::BString (
    char ch )
```

**7.84.1.6 BString()** [6/9]

```
BString::BString (
    BInt v )
```

**7.84.1.7 BString()** [7/9]

```
BString::BString (
    BUInt v )
```

#### 7.84.1.8 BString() [8/9]

```
BString::BString (
    BUInt64 v )
```

#### 7.84.1.9 BString() [9/9]

```
BString::BString (
    double v )
```

#### 7.84.1.10 ~BString()

```
BString::~~BString ( )
```

### 7.84.2 Member Function Documentation

#### 7.84.2.1 add()

```
BString BString::add (
    const BString & str ) const
```

Add strings returning result.

#### 7.84.2.2 append()

```
int BString::append (
    const BString & str )
```

Append a string.

#### 7.84.2.3 base64Decode()

```
BError BString::base64Decode (
    BString & str ) const
```

Decode a string from base64.

#### 7.84.2.4 base64Encode()

```
BString BString::base64Encode ( ) const
```

Encode a string to base64.

#### 7.84.2.5 basename()

```
BString BString::basename ( )
```

#### 7.84.2.6 clear()

```
void BString::clear ( )
```

Clear the string.

#### 7.84.2.7 compare()

```
int BString::compare (
    const BString & string ) const
```

Compare strings.

#### 7.84.2.8 compareRegex()

```
int BString::compareRegex (
    const BString & pattern,
    int ignoreCase = 0 ) const
```

Compare strings.

#### 7.84.2.9 compareWild()

```
int BString::compareWild (
    const BString & string ) const
```

Compare string to string with wildcards.

**7.84.2.10 compareWildExpression()**

```
int BString::compareWildExpression (
    const BString & string ) const
```

Compare string to space delimited patterns.

**7.84.2.11 convert()** [1/5]

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

Converts char to string.

**7.84.2.12 convert()** [2/5]

```
BString BString::convert (
    BInt value ) [static]
```

Converts int to string.

**7.84.2.13 convert()** [3/5]

```
BString BString::convert (
    BUInt value ) [static]
```

Converts uint to string.

**7.84.2.14 convert()** [4/5]

```
BString BString::convert (
    double value,
    int eFormat = 0 ) [static]
```

Converts double to string.

**7.84.2.15** `convert()` [5/5]

```
BString BString::convert (
    BUInt64 value ) [static]
```

Converts long long to string.

**7.84.2.16** `convertHex()` [1/2]

```
BString BString::convertHex (
    BInt value ) [static]
```

Converts int to string as hex value.

**7.84.2.17** `convertHex()` [2/2]

```
BString BString::convertHex (
    BUInt value ) [static]
```

Converts uint to string as hex value.

**7.84.2.18** `copy()`

```
BString BString::copy ( ) const
```

Return an independant copy.

**7.84.2.19** `csvDecode()`

```
BString & BString::csvDecode (
    const BString str )
```

Decode a string from CSV.

**7.84.2.20** `csvEncode()`

```
BString BString::csvEncode ( ) const
```

Encode a string for CSV.

#### 7.84.2.21 del()

```
int BString::del (
    int start,
    int len )
```

Delete substring.

#### 7.84.2.22 dirname()

```
BString BString::dirname ( )
```

#### 7.84.2.23 extension()

```
BString BString::extension ( )
```

#### 7.84.2.24 field()

```
BString BString::field (
    int field ) const
```

#### 7.84.2.25 fields()

```
char ** BString::fields ( )
```

#### 7.84.2.26 find() [1/2]

```
int BString::find (
    char ch ) const
```

Find ch in string searching forwards.

**7.84.2.27 find()** [2/2]

```
int BString::find (
    BString str ) const
```

Find string in string searching forwards.

**7.84.2.28 findReverse()**

```
int BString::findReverse (
    char ch ) const
```

Find ch in string searching backwards.

**7.84.2.29 firstLine()**

```
BString BString::firstLine ( )
```

Return first line.

**7.84.2.30 fixedLen()**

```
BString BString::fixedLen (
    int length,
    int rightJustify = 0 )
```

return string formatted to fixed length

**7.84.2.31 get()** [1/2]

```
char & BString::get (
    int pos )
```

**7.84.2.32 get()** [2/2]

```
const char & BString::get (
    int pos ) const
```

#### 7.84.2.33 getTokenList() [1/2]

```
BList< BString > BString::getTokenList (
    BString separators )
```

Break string into tokens.

#### 7.84.2.34 getTokenList() [2/2]

```
BList< BString > BString::getTokenList (
    char separator )
```

Break string into tokens.

#### 7.84.2.35 hash()

```
BUInt32 BString::hash ( ) const
```

#### 7.84.2.36 init()

```
void BString::init (
    const char * str ) [private]
```

#### 7.84.2.37 insert()

```
int BString::insert (
    int start,
    BString str )
```

Insert substring.

#### 7.84.2.38 inString()

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



#### 7.84.2.39 isSpace()

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

#### 7.84.2.40 justify()

```
BString BString::justify (
    int leftMargin,
    int width )
```

Justify the string to the given width.

#### 7.84.2.41 len()

```
int BString::len ( ) const
```

Length of string.

#### 7.84.2.42 lowerFirst()

```
BString BString::lowerFirst ( )
```

Return string with lowercase first character.

#### 7.84.2.43 operator const char \*()

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

#### 7.84.2.44 operator!=(=) [1/2]

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

**7.84.2.45 operator!=( )** [2/2]

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

**7.84.2.46 operator+( )** [1/6]

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

**7.84.2.47 operator+( )** [2/6]

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

**7.84.2.48 operator+( )** [3/6]

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

**7.84.2.49 operator+( )** [4/6]

```
BString BString::operator+ (  
    BInt i ) const [inline]
```

**7.84.2.50 operator+( )** [5/6]

```
BString BString::operator+ (  
    BUInt i ) const [inline]
```

**7.84.2.51 operator+( )** [6/6]

```
BString BString::operator+ (  
    BUInt64 i ) const [inline]
```

**7.84.2.52 operator+=()** [1/2]

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

**7.84.2.53 operator+=()** [2/2]

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

**7.84.2.54 operator<()** [1/2]

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

**7.84.2.55 operator<()** [2/2]

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

**7.84.2.56 operator<=()**

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

**7.84.2.57 operator=()**

```
BString & BString::operator= (
    const BString & string )
```

**7.84.2.58 operator==()** [1/2]

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

**7.84.2.59 operator==(** [2/2]

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

**7.84.2.60 operator>()** [1/2]

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

**7.84.2.61 operator>()** [2/2]

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

**7.84.2.62 operator>=()**

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

**7.84.2.63 operator[]()**

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

**7.84.2.64 pad()**

```
BString & BString::pad (
    int len )
```

Pad to length len.

#### 7.84.2.65 printf()

```
BString & BString::printf (
    const char * fmt,
    ... )
```

Formatted print into the string.

#### 7.84.2.66 pullLine()

```
BString BString::pullLine ( )
```

Pull a line out of the head of the string.

#### 7.84.2.67 pullSeparators()

```
BString BString::pullSeparators (
    BString separators )
```

Pull separators from start of string.

#### 7.84.2.68 pullToken()

```
BString BString::pullToken (
    BString terminators )
```

Pull token from start of string.

#### 7.84.2.69 pullWord()

```
BString BString::pullWord ( )
```

Pull a word out of the head of the string.

#### 7.84.2.70 removeNL()

```
void BString::removeNL ( )
```

Remove if present NL from last char.

#### 7.84.2.71 removeSeparators()

```
BString BString::removeSeparators (
    BString separators )
```

Remove any char from sepatators from string.

#### 7.84.2.72 retDouble()

```
double BString::retDouble ( ) const
```

Return string as a double.

#### 7.84.2.73 retFloat64()

```
BFloat64 BString::retFloat64 ( ) const
```

Return string as a BFloat64.

#### 7.84.2.74 retInt()

```
int BString::retInt ( ) const
```

Return string as a int.

#### 7.84.2.75 retStr()

```
const char * BString::retStr ( ) const
```

Ptr to char\* representation.

#### 7.84.2.76 retStrDup()

```
char * BString::retStrDup ( ) const
```

Ptr to newly malloc'd char\*.

#### 7.84.2.77 retUInt()

```
unsigned int BString::retUInt ( ) const
```

Return string as a int.

#### 7.84.2.78 reverse()

```
BString BString::reverse ( ) const
```

Reverse character order.

#### 7.84.2.79 split()

```
BList< BString > BString::split (
    char splitChar )
```

Split string into an array based on the character separator.

#### 7.84.2.80 subString()

```
BString BString::subString (
    int start,
    int len ) const
```

Returns substring.

#### 7.84.2.81 toLower()

```
BString & BString::toLowerCase ( )
```

Convert to lowercase.

#### 7.84.2.82 toUpper()

```
BString & BString::toUpperCase ( )
```

Convert to uppercase.

### 7.84.2.83 translateChar()

```
BString BString::translateChar (
    char ch,
    BString replace = " " )
```

Translate character converting them to the given string.

### 7.84.2.84 truncate()

```
BString & BString::truncate (
    int len )
```

Truncate to length len.

## 7.84.3 Member Data Documentation

### 7.84.3.1 ostr

```
BRefData* BString::ostr [protected]
```

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

- [BString.h](#)
- [BString.cpp](#)

## 7.85 BStringLocked Class Reference

```
#include <BStringLocked.h>
```

### Public Member Functions

- [BStringLocked](#) ()
- [BStringLocked](#) (const [BStringLocked](#) &s)
- [BStringLocked](#) (const [BString](#) &s)
- int len () const  
*Length of string.*
- operator [BString](#) () const
- [BStringLocked](#) operator+ (const [BStringLocked](#) &s) const
- [BStringLocked](#) & operator= (const [BStringLocked](#) &s)



## Private Attributes

- [BStringMutex olock](#)
- [BString ostr](#)

## 7.85.1 Constructor & Destructor Documentation

### 7.85.1.1 BStringLocked() [1/3]

```
BStringLocked::BStringLocked ( ) [inline]
```

### 7.85.1.2 BStringLocked() [2/3]

```
BStringLocked::BStringLocked (
    const BStringLocked & s ) [inline]
```

### 7.85.1.3 BStringLocked() [3/3]

```
BStringLocked::BStringLocked (
    const BString & s ) [inline]
```

## 7.85.2 Member Function Documentation

### 7.85.2.1 len()

```
int BStringLocked::len ( ) const [inline]
```

Length of string.

### 7.85.2.2 operator BString()

```
BStringLocked::operator BString ( ) const [inline]
```

### 7.85.2.3 operator+()

```
BStringLocked BStringLocked::operator+ (
    const BStringLocked & s ) const [inline]
```

### 7.85.2.4 operator=()

```
BStringLocked& BStringLocked::operator= (
    const BStringLocked & s ) [inline]
```

## 7.85.3 Member Data Documentation

### 7.85.3.1 olock

```
BStringMutex BStringLocked::olock [mutable], [private]
```

### 7.85.3.2 ostr

```
BString BStringLocked::ostr [private]
```

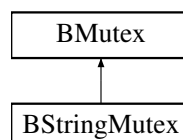
The documentation for this class was generated from the following file:

- [BStringLocked.h](#)

## 7.86 BStringMutex Class Reference

```
#include <BStringLocked.h>
```

Inheritance diagram for BStringMutex:



### Public Member Functions

- [BStringMutex \(\)](#)

## Additional Inherited Members

### 7.86.1 Constructor & Destructor Documentation

#### 7.86.1.1 BStringMutex()

```
BStringMutex::BStringMutex ( ) [inline]
```

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

- [BStringLocked.h](#)

## 7.87 BTable Class Reference

```
#include <BTable.h>
```

### Public Member Functions

- [BTable](#) ()
- [~BTable](#) ()
- void [clear](#) ()
- void [setTitle](#) ([BArray](#)< [BString](#) > title)
- void [addRow](#) ([BArray](#)< [BString](#) > data)
- void [print](#) ()

### Private Member Functions

- void [calculateWidths](#) ()
- void [printLine](#) ([BArray](#)< [BString](#) > line, int comment=0)

### Private Attributes

- [BArray](#)< [BString](#) > otitle
- [BList](#)< [BArray](#)< [BString](#) > > odata
- [BArray](#)< int > ocolumnWidths

### 7.87.1 Constructor & Destructor Documentation

### 7.87.1.1 BTable()

```
BTable::BTable ( )
```

### 7.87.1.2 ~BTable()

```
BTable::~~BTable ( )
```

## 7.87.2 Member Function Documentation

### 7.87.2.1 addRow()

```
void BTable::addRow (
    BArray< BString > data )
```

### 7.87.2.2 calculateWidths()

```
void BTable::calculateWidths ( ) [private]
```

### 7.87.2.3 clear()

```
void BTable::clear ( )
```

### 7.87.2.4 print()

```
void BTable::print ( )
```

### 7.87.2.5 printLine()

```
void BTable::printLine (
    BArray< BString > line,
    int comment = 0 ) [private]
```

### 7.87.2.6 setTitle()

```
void BTable::setTitle (
    BArray< BString > title )
```

## 7.87.3 Member Data Documentation

### 7.87.3.1 ocolumnWidths

```
BArray<int> BTable::ocolumnWidths [private]
```

### 7.87.3.2 odata

```
BList<BArray<BString> > BTable::odata [private]
```

### 7.87.3.3 otitle

```
BArray<BString> BTable::otitle [private]
```

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

- [BTable.h](#)
- [BTable.cpp](#)

## 7.88 BTask Class Reference

```
#include <BTask.h>
```

### Public Member Functions

- [BTask](#) (const char \*name="", [BUInt](#) stackSize=0, [BUInt](#) priority=1)
- [~BTask](#) ()
- void [init](#) (const char \*name, [BUInt](#) stackSize=0, [BUInt](#) priority=1)
- [BError](#) [start](#) ()  
*Starts the task running.*
- void [stop](#) ()
- void [waitForCompletion](#) ()
- int [setPriority](#) ([BUInt](#) priority)  
*Set the priority of the task: 0 upwards.*
- virtual void [run](#) ()

## Static Protected Member Functions

- static void \* [taskFunc](#) (void \*)

## Protected Attributes

- const char \* [oname](#)
- [BUInt](#) [ostackSize](#)
- [BUInt](#) [opolicy](#)
- [BUInt](#) [opriority](#)
- [pthread\\_t](#) [othread](#)
- [Bool](#) [orunning](#)

## 7.88.1 Constructor & Destructor Documentation

### 7.88.1.1 BTask()

```
BTask::BTask (
    const char * name = "",
    BUInt stackSize = 0,
    BUInt priority = 1 )
```

### 7.88.1.2 ~BTask()

```
BTask::~~BTask ( )
```

## 7.88.2 Member Function Documentation

### 7.88.2.1 init()

```
void BTask::init (
    const char * name,
    BUInt stackSize = 0,
    BUInt priority = 1 )
```

### 7.88.2.2 run()

```
void BTask::run ( ) [virtual]
```

### 7.88.2.3 setPriority()

```
int BTask::setPriority (
    BUInt priority )
```

Set the priority of the task: 0 upwards.

### 7.88.2.4 start()

```
BError BTask::start ( )
```

Starts the task running.

### 7.88.2.5 stop()

```
void BTask::stop ( )
```

### 7.88.2.6 taskFunc()

```
void * BTask::taskFunc (
    void * arg ) [static], [protected]
```

### 7.88.2.7 waitForCompletion()

```
void BTask::waitForCompletion ( )
```

## 7.88.3 Member Data Documentation

### 7.88.3.1 oname

```
const char* BTask::oname [protected]
```

### 7.88.3.2 opolicy

`BUInt BTask::opolicy [protected]`

### 7.88.3.3 opriority

`BUInt BTask::opriority [protected]`

### 7.88.3.4 orunning

`Bool BTask::orunning [protected]`

### 7.88.3.5 ostackSize

`BUInt BTask::ostackSize [protected]`

### 7.88.3.6 othread

`pthread_t BTask::othread [protected]`

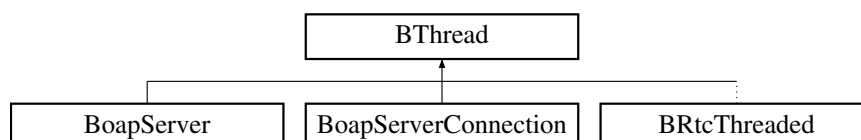
The documentation for this class was generated from the following files:

- [BTask.h](#)
- [BTask.cpp](#)

## 7.89 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 \* [oreult](#)

## 7.89.1 Constructor & Destructor Documentation

### 7.89.1.1 BThread()

```
BThread::BThread ( )
```

### 7.89.1.2 ~BThread()

```
BThread::~~BThread ( ) [virtual]
```

## 7.89.2 Member Function Documentation

#### 7.89.2.1 cancel()

```
int BThread::cancel ( )
```

#### 7.89.2.2 function()

```
void * BThread::function ( ) [virtual]
```

Reimplemented in [BoapServer](#), [BoapServerConnection](#), and [BRtcThreaded](#).

#### 7.89.2.3 getThread()

```
pthread_t BThread::getThread ( )
```

#### 7.89.2.4 result()

```
void * BThread::result ( )
```

#### 7.89.2.5 running()

```
int BThread::running ( )
```

#### 7.89.2.6 setInitPriority()

```
int BThread::setInitPriority (
    int policy,
    int priority )
```

#### 7.89.2.7 setInitStackSize()

```
int BThread::setInitStackSize (
    size_t stackSize )
```

#### 7.89.2.8 setPriority()

```
int BThread::setPriority (
    int policy,
    int priority )
```

#### 7.89.2.9 start()

```
int BThread::start ( )
```

#### 7.89.2.10 startFunc()

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

#### 7.89.2.11 waitForCompletion()

```
void * BThread::waitForCompletion ( )
```

### 7.89.3 Member Data Documentation

#### 7.89.3.1 opolicy

```
int BThread::opolicy [private]
```

#### 7.89.3.2 opriority

```
int BThread::opriority [private]
```

#### 7.89.3.3 oresult

```
void* BThread::oresult [private]
```

#### 7.89.3.4 orunning

```
int BThread::orunning [private]
```

#### 7.89.3.5 ostackSize

```
size_t BThread::ostackSize [private]
```

#### 7.89.3.6 othread

```
pthread_t BThread::othread [private]
```

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

- [BThread.h](#)
- [BThread.cpp](#)

## 7.90 BTime Class Reference

```
#include <BTime.h>
```

### Public Member Functions

- [BTime](#) (BUInt32 t=0)
- void [set](#) (BUInt32 seconds)  
*Set the date and time.*
- void [set](#) (BUInt year, BUInt month, BUInt day, BUInt hour=0, BUInt minute=0, BUInt second=0)  
*Set the date and time.*
- void [setYearDay](#) (BUInt year, BUInt yearDay, BUInt hour=0, BUInt minute=0, BUInt second=0)  
*Set the date and time.*
- void [getDate](#) (BUInt &year, BUInt &month, BUInt &day) const  
*Return the date information.*
- void [getTime](#) (BUInt &hour, BUInt &minute, BUInt &second) const  
*Return the time information.*
- BUInt32 [getSeconds](#) () const  
*Return the number of seconds.*
- int [isSet](#) () const  
*Check if set.*
- int [isLeapYear](#) ()  
*Returns if a leap year.*
- void [addSeconds](#) (int seconds)  
*Add the given number of seconds.*
- BString [getString](#) (BString format="isoT") const

*Gets the date/time in string format.*

- [BError setString](#) (const [BString](#) dateTime)

*Sets the date/time from string format.*

- int [operator==](#) (const [BTime](#) &time) const
- int [operator!=](#) (const [BTime](#) &time) const
- int [operator>](#) (const [BTime](#) &time) const
- int [operator>=](#) (const [BTime](#) &time) const
- int [operator<](#) (const [BTime](#) &time) const
- int [operator<=](#) (const [BTime](#) &time) const
- [BTime operator+](#) (int seconds) const
- [BTime & operator+=](#) (int seconds)

## Private Attributes

- [BUInt32 otime](#)

*Time in seconds since 1970. range 1970-01-02 to 2106-02-07.*

## 7.90.1 Constructor & Destructor Documentation

### 7.90.1.1 BTime()

```
BTime::BTime (
    BUInt32 t = 0 )
```

## 7.90.2 Member Function Documentation

### 7.90.2.1 addSeconds()

```
void BTime::addSeconds (
    int seconds )
```

Add the given number of seconds.

### 7.90.2.2 getDate()

```
void BTime::getDate (
    BUInt & year,
    BUInt & month,
    BUInt & day ) const
```

Return the date information.

### 7.90.2.3 getSeconds()

```
BUInt32 BTime::getSeconds ( ) const
```

Return the number of seconds.

### 7.90.2.4 getString()

```
BString BTime::getString (
    BString format = "isoT" ) const
```

Gets the date/time in string format.

### 7.90.2.5 getTime()

```
void BTime::getTime (
    BUInt & hour,
    BUInt & minute,
    BUInt & second ) const
```

Return the time information.

### 7.90.2.6 isLeapYear()

```
int BTime::isLeapYear ( )
```

Returns if a leap year.

### 7.90.2.7 isSet()

```
int BTime::isSet ( ) const [inline]
```

Check if set.

### 7.90.2.8 operator!=(())

```
int BTime::operator!=(
    const BTime & time ) const [inline]
```

### 7.90.2.9 operator+()

```
BTime BTime::operator+ (
    int seconds ) const [inline]
```

### 7.90.2.10 operator+=()

```
BTime& BTime::operator+= (
    int seconds ) [inline]
```

### 7.90.2.11 operator<()

```
int BTime::operator< (
    const BTime & time ) const [inline]
```

### 7.90.2.12 operator<=()

```
int BTime::operator<= (
    const BTime & time ) const [inline]
```

### 7.90.2.13 operator==()

```
int BTime::operator== (
    const BTime & time ) const [inline]
```

### 7.90.2.14 operator>()

```
int BTime::operator> (
    const BTime & time ) const [inline]
```

### 7.90.2.15 operator>=()

```
int BTime::operator>= (
    const BTime & time ) const [inline]
```

#### 7.90.2.16 set() [1/2]

```
void BTime::set (
    BUInt32 seconds )
```

Set the date and time.

#### 7.90.2.17 set() [2/2]

```
void BTime::set (
    BUInt year,
    BUInt month,
    BUInt day,
    BUInt hour = 0,
    BUInt minute = 0,
    BUInt second = 0 )
```

Set the date and time.

#### 7.90.2.18 setString()

```
BError BTime::setString (
    const BString dateTime )
```

Sets the date/time from string format.

#### 7.90.2.19 setYearDay()

```
void BTime::setYearDay (
    BUInt year,
    BUInt yearDay,
    BUInt hour = 0,
    BUInt minute = 0,
    BUInt second = 0 )
```

Set the date and time.

### 7.90.3 Member Data Documentation



### 7.90.3.1 otime

`BUInt32 BTime::otime [private]`

Time in seconds since 1970. range 1970-01-02 to 2106-02-07.

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

- [BTime.h](#)
- [BTime.cpp](#)

## 7.91 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](#)

### 7.91.1 Detailed Description

Stopwatch style timer.

### 7.91.2 Constructor & Destructor Documentation

#### 7.91.2.1 BTimer()

```
BTimer::BTimer ( )
```

#### 7.91.2.2 ~BTimer()

```
BTimer::~~BTimer ( )
```

### 7.91.3 Member Function Documentation

#### 7.91.3.1 add()

```
void BTimer::add (
    BTimer & timer )
```

Add two timers.

#### 7.91.3.2 average()

```
double BTimer::average ( )
```

Average time is duration between [start\(\)](#) and [stop\(\)](#) / number of stops.

#### 7.91.3.3 clear()

```
void BTimer::clear ( )
```

Clear timer.

#### 7.91.3.4 getElapsedTime()

```
double BTimer::getElapsedTime ( )
```

Returns the elapsed time from the last start.

#### 7.91.3.5 getTime()

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

#### 7.91.3.6 peak()

```
double BTimer::peak ( )
```

Peak time.

#### 7.91.3.7 start()

```
void BTimer::start ( )
```

Start timer.

#### 7.91.3.8 stop()

```
void BTimer::stop ( )
```

Stop timer.

### 7.91.4 Member Data Documentation

#### 7.91.4.1 oaverage

```
double BTimer::oaverage [private]
```

#### 7.91.4.2 oendTime

```
double BTimer::oendTime [private]
```

#### 7.91.4.3 olock

```
BMutex BTimer::olock [private]
```

#### 7.91.4.4 onum

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

#### 7.91.4.5 opeak

```
double BTimer::opeak [private]
```

#### 7.91.4.6 ostartTime

```
double BTimer::ostartTime [private]
```

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

- [BTimer.h](#)
- [BTimer.cpp](#)

## 7.92 BTimeStamp Class Reference

```
#include <BTimeStamp.h>
```

## Public Member Functions

- [BTimeStamp](#) ()
- [BTimeStamp](#) (int year, int month=1, int day=1, int hour=0, int minute=0, int second=0, int microsecond=0)
- [BTimeStamp](#) (const [BString](#) str)
- [~BTimeStamp](#) ()
- void [clear](#) ()
  - Clear the date/time.*
- void [setFirst](#) ()
  - Set the first date available.*
- void [setLast](#) ()
  - Set the last date available.*
- void [set](#) (time\_t time, int microSeconds)
  - Set time using Unix time (seconds from 1970-01-01)*
- void [set](#) (int year=0, int month=1, int day=1, int hour=0, int minute=0, int second=0, int microsecond=0)
- void [set](#) (const [BTimeStampMs](#) &timeStamp)
  - Set the timeStamp to given MS time stamp.*
- void [setYDay](#) (int year=0, int yday=0, int hour=0, int minute=0, int second=0, int microsecond=0)
- void [setTime](#) (int hour=0, int minute=0, int second=0, int microsecond=0)
- void [setNow](#) ()
  - Set the timeStamp to now.*
- int [year](#) () const
- int [yday](#) () const
- int [month](#) () const
- int [day](#) () const
- int [hour](#) () const
- int [minute](#) () const
- int [second](#) () const
- int [microSecond](#) () const
- void [getDate](#) (int &year, int &mon, int &day) const
- [BString](#) [getString](#) ([BString](#) separator="T") const
  - Get the time as an ISO date/time string.*
- [BError](#) [setString](#) (const [BString](#) dateTime)
  - Set the time from an ISO date/time.*
- [BString](#) [getStringNoMs](#) ([BString](#) separator="T") const
  - Get the time as an ISO date/time string without microseconds.*
- [BString](#) [getStringFormatted](#) ([BString](#) format) const
  - Gets the time in a string form as per the format. Format syntax as per strftime()*
- void [addMilliSeconds](#) (int milliSeconds)
  - Add the given number of milli seconds. This should be less that a year.*
- void [addMicroSeconds](#) (int64\_t microSeconds)
  - Add the given number of micro seconds. This should be less that a year.*
- void [addSeconds](#) (int seconds)
  - Add the given number of seconds. This should be less that a year.*
- uint32\_t [getYearSeconds](#) () const
  - Get number of seconds within the year.*
- uint64\_t [getYearMicroSeconds](#) () const
  - Get number of micro seconds within the year.*
- int [isSet](#) () const
- int [compare](#) (const [BTimeStamp](#) &timeStamp) const
  - Compare two dates.*
- [operator BString](#) () const

- [BTimeStamp](#) & [operator=](#) (const [BTimeStampMs](#) &timeStamp)
- int [operator==](#) (const [BTimeStamp](#) &timeStamp) const
- int [operator!=](#) (const [BTimeStamp](#) &timeStamp) const
- int [operator>](#) (const [BTimeStamp](#) &timeStamp) const
- int [operator>=](#) (const [BTimeStamp](#) &timeStamp) const
- int [operator<](#) (const [BTimeStamp](#) &timeStamp) const
- int [operator<=](#) (const [BTimeStamp](#) &timeStamp) const

### Static Public Member Functions

- static int [isLeap](#) (int year)
- static [BInt64](#) [difference](#) ([BTimeStamp](#) t2, [BTimeStamp](#) t1)

### Public Attributes

- uint16\_t [oyear](#)  
*Year (0 .. 65535)*
- uint16\_t [oyday](#)  
*Day in year (0 .. 365)*
- uint8\_t [ohour](#)  
*Hour (0 .. 23)*
- uint8\_t [ominute](#)  
*Minute (0 .. 59)*
- uint8\_t [osecond](#)  
*Second (0 .. 59)*
- uint8\_t [ospare](#)  
*Padding.*
- uint32\_t [omicroSecond](#)  
*MicroSecond (0 .. 999999)*

## 7.92.1 Constructor & Destructor Documentation

### 7.92.1.1 [BTimeStamp\(\)](#) [1/3]

```
BTimeStamp::BTimeStamp ( )
```

### 7.92.1.2 [BTimeStamp\(\)](#) [2/3]

```
BTimeStamp::BTimeStamp (
    int year,
    int month = 1,
    int day = 1,
    int hour = 0,
    int minute = 0,
    int second = 0,
    int microsecond = 0 )
```

### 7.92.1.3 BTimeStamp() [3/3]

```
BTimeStamp::BTimeStamp (
    const BString str )
```

### 7.92.1.4 ~BTimeStamp()

```
BTimeStamp::~~BTimeStamp ( )
```

## 7.92.2 Member Function Documentation

### 7.92.2.1 addMicroSeconds()

```
void BTimeStamp::addMicroSeconds (
    int64_t microSeconds )
```

Add the given number of micro seconds. This should be less than a year.

### 7.92.2.2 addMilliSeconds()

```
void BTimeStamp::addMilliSeconds (
    int milliSeconds )
```

Add the given number of milli seconds. This should be less than a year.

### 7.92.2.3 addSeconds()

```
void BTimeStamp::addSeconds (
    int seconds )
```

Add the given number of seconds. This should be less than a year.

### 7.92.2.4 clear()

```
void BTimeStamp::clear ( )
```

Clear the date/time.

#### 7.92.2.5 compare()

```
int BTimeStamp::compare (
    const BTimeStamp & timeStamp ) const
```

Compare two dates.

#### 7.92.2.6 day()

```
int BTimeStamp::day ( ) const
```

#### 7.92.2.7 difference()

```
BInt64 BTimeStamp::difference (
    BTimeStamp t2,
    BTimeStamp t1 ) [static]
```

#### 7.92.2.8 getDate()

```
void BTimeStamp::getDate (
    int & year,
    int & mon,
    int & day ) const
```

#### 7.92.2.9 getString()

```
BString BTimeStamp::getString (
    BString separator = "T" ) const
```

Get the time as an ISO date/time string.

#### 7.92.2.10 getStringFormatted()

```
BString BTimeStamp::getStringFormatted (
    BString format ) const
```

Gets the time in a string form as per the format. Format syntax as per strftime()



### 7.92.2.11 getStringNoMs()

```
BString BTimeStamp::getStringNoMs (
    BString separator = "T" ) const
```

Get the time as an ISO date/time string without microseconds.

### 7.92.2.12 getYearMicroSeconds()

```
uint64_t BTimeStamp::getYearMicroSeconds ( ) const
```

Get number of micro seconds within the year.

### 7.92.2.13 getYearSeconds()

```
uint32_t BTimeStamp::getYearSeconds ( ) const
```

Get number of seconds within the year.

### 7.92.2.14 hour()

```
int BTimeStamp::hour ( ) const
```

### 7.92.2.15 isLeap()

```
int BTimeStamp::isLeap (
    int year ) [static]
```

### 7.92.2.16 isSet()

```
int BTimeStamp::isSet ( ) const [inline]
```

### 7.92.2.17 microSecond()

```
int BTimeStamp::microSecond ( ) const
```

#### 7.92.2.18 minute()

```
int BTimeStamp::minute ( ) const
```

#### 7.92.2.19 month()

```
int BTimeStamp::month ( ) const
```

#### 7.92.2.20 operator BString()

```
BTimeStamp::operator BString ( ) const [inline]
```

#### 7.92.2.21 operator!=(=)

```
int BTimeStamp::operator!= (
    const BTimeStamp & timeStamp ) const [inline]
```

#### 7.92.2.22 operator<()

```
int BTimeStamp::operator< (
    const BTimeStamp & timeStamp ) const [inline]
```

#### 7.92.2.23 operator<=()

```
int BTimeStamp::operator<= (
    const BTimeStamp & timeStamp ) const [inline]
```

#### 7.92.2.24 operator=(=)

```
BTimeStamp& BTimeStamp::operator= (
    const BTimeStampMs & timeStamp ) [inline]
```

**7.92.2.25 operator==( )**

```
int BTimeStamp::operator==(
    const BTimeStamp & timeStamp ) const [inline]
```

**7.92.2.26 operator>( )**

```
int BTimeStamp::operator>(
    const BTimeStamp & timeStamp ) const [inline]
```

**7.92.2.27 operator>=( )**

```
int BTimeStamp::operator>=(
    const BTimeStamp & timeStamp ) const [inline]
```

**7.92.2.28 second( )**

```
int BTimeStamp::second ( ) const
```

**7.92.2.29 set( ) [1/3]**

```
void BTimeStamp::set (
    time_t time,
    int microseconds )
```

Set time using Unix time (seconds from 1970-01-01)

**7.92.2.30 set( ) [2/3]**

```
void BTimeStamp::set (
    int year = 0,
    int month = 1,
    int day = 1,
    int hour = 0,
    int minute = 0,
    int second = 0,
    int microsecond = 0 )
```

**7.92.2.31 set()** [3/3]

```
void BTimeStamp::set (
    const BTimeStampMs & timeStamp )
```

Set the timeStamp to given MS time stamp.

**7.92.2.32 setFirst()**

```
void BTimeStamp::setFirst ( )
```

Set the first date available.

**7.92.2.33 setLast()**

```
void BTimeStamp::setLast ( )
```

Set the last date available.

**7.92.2.34 setNow()**

```
void BTimeStamp::setNow ( )
```

Set the timeStamp to now.

**7.92.2.35 setString()**

```
BError BTimeStamp::setString (
    const BString dateTime )
```

Set the time from an ISO date/time.

**7.92.2.36 setTime()**

```
void BTimeStamp::setTime (
    int hour = 0,
    int minute = 0,
    int second = 0,
    int microsecond = 0 )
```

### 7.92.2.37 setYDay()

```
void BTimeStamp::setYDay (
    int year = 0,
    int yday = 0,
    int hour = 0,
    int minute = 0,
    int second = 0,
    int microsecond = 0 )
```

### 7.92.2.38 yday()

```
int BTimeStamp::yday ( ) const
```

### 7.92.2.39 year()

```
int BTimeStamp::year ( ) const
```

## 7.92.3 Member Data Documentation

### 7.92.3.1 ohour

```
uint8_t BTimeStamp::ohour
```

Hour (0 .. 23)

### 7.92.3.2 omicroSecond

```
uint32_t BTimeStamp::omicroSecond
```

MicroSecond (0 .. 999999)

### 7.92.3.3 ominute

```
uint8_t BTimeStamp::ominate
```

Minute (0 .. 59)

#### 7.92.3.4 osecond

`uint8_t BTimeStamp::osecond`

Second (0 .. 59)

#### 7.92.3.5 ospare

`uint8_t BTimeStamp::ospare`

Padding.

#### 7.92.3.6 oyday

`uint16_t BTimeStamp::oyday`

Day in year (0 .. 365)

#### 7.92.3.7 oyear

`uint16_t BTimeStamp::oyear`

Year (0 .. 65535)

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

- [BTimeStamp.h](#)
- [BTimeStamp.cpp](#)

## 7.93 BTimeStampMs Class Reference

```
#include <BTimeStampMs.h>
```

## Public Member Functions

- [BTimeStampMs](#) (BString str="")
- [~BTimeStampMs](#) ()
- void [clear](#) ()
  - Clear the date/time.*
- void [setNow](#) ()
  - Set the timeStamp to now.*
- [BTimeStampMs](#) & [addMilliseconds](#) (int milliseconds)
  - Add the given number of milli seconds. This should be less than a year.*
- [BTimeStampMs](#) & [subMilliseconds](#) (int milliseconds)
  - Add the given number of milli seconds. This should be less than a year.*
- [BTimeStampMs](#) & [addSeconds](#) (int seconds)
  - Add the given number of seconds. This should be less than a year.*
- [BTimeStampMs](#) & [subSeconds](#) (int seconds)
  - Subtract the given number of seconds. This should be less than a year.*
- uint32\_t [getYearSeconds](#) ()
  - Get number of seconds within the year.*
- uint64\_t [getYearMilliseconds](#) ()
  - Get number of seconds within the year.*
- BString [getString](#) (BString separator="T")
  - Get the time as an ISO date/time string.*
- BString [getStringNoMs](#) (BString separator="T")
  - Get the time as an ISO date/time string with no ms.*
- BError [setString](#) (BString dateTime)
  - Set the time from an ISO date/time.*
- BString [getDurationString](#) (BString separator="T")
  - Get the time as an ISO date/time string but with month's and days starting from 0.*
- BString [getDurationStringNoMs](#) (BString separator="T")
  - Get the time as an ISO date/time string but with month's and days starting from 0 with no ms.*
- BError [setDurationString](#) (BString dateTime)
  - Set the time from an ISO date/time string but with month's and days starting from 0.*
- BString [getStringRaw](#) ()
- void [getDate](#) (int &year, int &mon, int &day)
  - Get the year, month and day.*
- int [compare](#) (const [BTimeStampMs](#) &timeStamp)
  - Compare two dates.*
- int [operator>](#) (const [BTimeStampMs](#) &timeStamp)
- int [operator>=](#) (const [BTimeStampMs](#) &timeStamp)
- int [operator<](#) (const [BTimeStampMs](#) &timeStamp)
- int [operator<=](#) (const [BTimeStampMs](#) &timeStamp)

## Static Public Member Functions

- static int [isLeap](#) (int year)
- static BUInt64 [difference](#) ([BTimeStampMs](#) t2, [BTimeStampMs](#) t1)

## Public Attributes

- [uint16\\_t year](#)  
*Year (2000 .. 3000)*
- [uint16\\_t yday](#)  
*Day in year (0 .. 365)*
- [uint16\\_t hour](#)  
*Hour (0 .. 23)*
- [uint16\\_t minute](#)  
*Minute (0 .. 59)*
- [uint16\\_t second](#)  
*Second (0 .. 59)*
- [uint16\\_t milliSecond](#)  
*MilliSecond (0 .. 999)*
- [int32\\_t sampleNumber](#)  
*The sample number this time refers to.*

## 7.93.1 Constructor & Destructor Documentation

### 7.93.1.1 BTimeStampMs()

```
BTimeStampMs::BTimeStampMs (
    BString str = "" )
```

### 7.93.1.2 ~BTimeStampMs()

```
BTimeStampMs::~~BTimeStampMs ( )
```

## 7.93.2 Member Function Documentation

### 7.93.2.1 addMilliSeconds()

```
BTimeStampMs & BTimeStampMs::addMilliSeconds (
    int milliSeconds )
```

Add the given number of milli seconds. This should be less that a year.



### 7.93.2.2 addSeconds()

```
BTimeStampMs & BTimeStampMs::addSeconds (
    int seconds )
```

Add the given number of seconds. This should be less than a year.

### 7.93.2.3 clear()

```
void BTimeStampMs::clear ( )
```

Clear the date/time.

### 7.93.2.4 compare()

```
int BTimeStampMs::compare (
    const BTimeStampMs & timeStamp )
```

Compare two dates.

### 7.93.2.5 difference()

```
BUInt64 BTimeStampMs::difference (
    BTimeStampMs t2,
    BTimeStampMs t1 ) [static]
```

### 7.93.2.6 getDate()

```
void BTimeStampMs::getDate (
    int & year,
    int & mon,
    int & day )
```

Get the year, month and day.

### 7.93.2.7 getDurationString()

```
BString BTimeStampMs::getDurationString (
    BString separator = "T" )
```

Get the time as an ISO date/time string but with month's and days starting from 0.

### 7.93.2.8 getDurationStringNoMs()

```
BString BTimeStampMs::getDurationStringNoMs (
    BString separator = "T" )
```

Get the time as an ISO date/time string but with month's and days starting from 0 with no ms.

### 7.93.2.9 getString()

```
BString BTimeStampMs::getString (
    BString separator = "T" )
```

Get the time as an ISO date/time string.

### 7.93.2.10 getStringNoMs()

```
BString BTimeStampMs::getStringNoMs (
    BString separator = "T" )
```

Get the time as an ISO date/time string with no ms.

### 7.93.2.11 getStringRaw()

```
BString BTimeStampMs::getStringRaw ( )
```

### 7.93.2.12 getYearMilliSeconds()

```
uint64_t BTimeStampMs::getYearMilliSeconds ( )
```

Get number of seconds within the year.

### 7.93.2.13 getYearSeconds()

```
uint32_t BTimeStampMs::getYearSeconds ( )
```

Get number of seconds within the year.

#### 7.93.2.14 isLeap()

```
int BTimeStampMs::isLeap (
    int year ) [static]
```

#### 7.93.2.15 operator<()

```
int BTimeStampMs::operator< (
    const BTimeStampMs & timeStamp ) [inline]
```

#### 7.93.2.16 operator<=()

```
int BTimeStampMs::operator<= (
    const BTimeStampMs & timeStamp ) [inline]
```

#### 7.93.2.17 operator>()

```
int BTimeStampMs::operator> (
    const BTimeStampMs & timeStamp ) [inline]
```

#### 7.93.2.18 operator>=()

```
int BTimeStampMs::operator>= (
    const BTimeStampMs & timeStamp ) [inline]
```

#### 7.93.2.19 setDurationString()

```
BError BTimeStampMs::setDurationString (
    BString dateTime )
```

Set the time from an ISO date/time string but with month's and days starting from 0.

#### 7.93.2.20 `setNow()`

```
void BTimeStampMs::setNow ( )
```

Set the timeStamp to now.

#### 7.93.2.21 `setString()`

```
BError BTimeStampMs::setString (
    BString dateTime )
```

Set the time from an ISO date/time.

#### 7.93.2.22 `subMilliseconds()`

```
BTimeStampMs & BTimeStampMs::subMilliseconds (
    int milliseconds )
```

Add the given number of milli seconds. This should be less that a year.

#### 7.93.2.23 `subSeconds()`

```
BTimeStampMs & BTimeStampMs::subSeconds (
    int seconds )
```

Subtract the given number of seconds. This should be less that a year.

### 7.93.3 Member Data Documentation

#### 7.93.3.1 `hour`

```
uint16_t BTimeStampMs::hour
```

Hour (0 .. 23)

### 7.93.3.2 millisecond

uint16\_t BTimeStampMs::millisecond

Millisecond (0 .. 999)

### 7.93.3.3 minute

uint16\_t BTimeStampMs::minute

Minute (0 .. 59)

### 7.93.3.4 sampleNumber

int32\_t BTimeStampMs::sampleNumber

The sample number this time refers to.

### 7.93.3.5 second

uint16\_t BTimeStampMs::second

Second (0 .. 59)

### 7.93.3.6 yday

uint16\_t BTimeStampMs::yday

Day in year (0 .. 365)

### 7.93.3.7 year

uint16\_t BTimeStampMs::year

Year (2000 .. 3000)

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

- [BTimeStampMs.h](#)
- [BTimeStampMs.cpp](#)

## 7.94 BTimeUs Class Reference

```
#include <BTimeUs.h>
```

### Public Member Functions

- [BTimeUs](#) (BUInt64 t=0)
- [BTimeUs](#) (BTime t)
- void [set](#) (BUInt64 microseconds)
  - Set the time to TAI us.*
- void [set](#) (BUInt year, BUInt month, BUInt day, BUInt hour=0, BUInt minute=0, BUInt second=0, BUInt micro←  
Second=0)
  - Set the date and time from UTC.*
- void [setYearDay](#) (BUInt year, BUInt yearDay, BUInt hour=0, BUInt minute=0, BUInt second=0, BUInt micro←  
Second=0)
  - Set the date and time from UTC.*
- void [getDate](#) (BUInt &year, BUInt &month, BUInt &day) const
  - Return the date information UTC.*
- void [getTime](#) (BUInt &hour, BUInt &minute, BUInt &second) const
  - Return the time information UTC.*
- BUInt64 [getSeconds](#) () const
  - Return the number of seconds TAI.*
- BUInt64 [getMicroSeconds](#) () const
  - Return the number of micro seconds TAI.*
- int [isSet](#) () const
  - Check if set.*
- int [isLeapYear](#) ()
  - Returns if a leap year.*
- void [addSeconds](#) (BInt64 seconds)
  - Add the given number of seconds.*
- void [addMicroSeconds](#) (BInt64 microseconds)
  - Add the given number of seconds.*
- BString [getString](#) (BString format="isoT") const
  - Gets the date/time in string format.*
- BString [getStringUs](#) (BString format="isoT") const
  - Gets the date/time in string format.*
- BError [setString](#) (const BString dateTime)
  - Sets the date/time from string format.*
- [operator BTime](#) () const
- int [operator==](#) (const BTimeUs &time) const
- int [operator!=](#) (const BTimeUs &time) const
- int [operator>](#) (const BTimeUs &time) const
- int [operator>=](#) (const BTimeUs &time) const
- int [operator<](#) (const BTimeUs &time) const
- int [operator<=](#) (const BTimeUs &time) const
- BTimeUs [operator+](#) (BInt64 microseconds) const
- BTimeUs & [operator+=](#) (BInt64 microseconds)

## Private Attributes

- [BUInt64 otime](#)

*Time in microseconds since 1970-01-01 to TAI standard.*

## 7.94.1 Constructor & Destructor Documentation

### 7.94.1.1 BTimeUs() [1/2]

```
BTimeUs::BTimeUs (  
    BUInt64 t = 0 )
```

### 7.94.1.2 BTimeUs() [2/2]

```
BTimeUs::BTimeUs (  
    BTime t )
```

## 7.94.2 Member Function Documentation

### 7.94.2.1 addMicroSeconds()

```
void BTimeUs::addMicroSeconds (  
    BInt64 microSeconds )
```

Add the given number of seconds.

### 7.94.2.2 addSeconds()

```
void BTimeUs::addSeconds (  
    BInt64 seconds )
```

Add the given number of seconds.

### 7.94.2.3 getDate()

```
void BTimeUs::getDate (
    BUInt & year,
    BUInt & month,
    BUInt & day ) const
```

Return the date information UTC.

### 7.94.2.4 getMicroSeconds()

```
BUInt64 BTimeUs::getMicroSeconds ( ) const
```

Return the number of micro seconds TAI.

### 7.94.2.5 getSeconds()

```
BUInt64 BTimeUs::getSeconds ( ) const
```

Return the number of seconds TAI.

### 7.94.2.6 getString()

```
BString BTimeUs::getString (
    BString format = "isoT" ) const
```

Gets the date/time in string format.

### 7.94.2.7 getStringUs()

```
BString BTimeUs::getStringUs (
    BString format = "isoT" ) const
```

Gets the date/time in string format.



#### 7.94.2.8 getTime()

```
void BTimeUs::getTime (
    BUInt & hour,
    BUInt & minute,
    BUInt & second ) const
```

Return the time information UTC.

#### 7.94.2.9 isLeapYear()

```
int BTimeUs::isLeapYear ( )
```

Returns if a leap year.

#### 7.94.2.10 isSet()

```
int BTimeUs::isSet ( ) const [inline]
```

Check if set.

#### 7.94.2.11 operator BTime()

```
BTimeUs::operator BTime ( ) const [inline]
```

#### 7.94.2.12 operator!=(=)

```
int BTimeUs::operator!=(= (
    const BTimeUs & time ) const [inline]
```

#### 7.94.2.13 operator+()

```
BTimeUs BTimeUs::operator+ (
    BInt64 microseconds ) const [inline]
```

#### 7.94.2.14 operator+()

```
BTimeUs& BTimeUs::operator+= (
    BInt64 microSeconds ) [inline]
```

#### 7.94.2.15 operator<()

```
int BTimeUs::operator< (
    const BTimeUs & time ) const [inline]
```

#### 7.94.2.16 operator<=()

```
int BTimeUs::operator<= (
    const BTimeUs & time ) const [inline]
```

#### 7.94.2.17 operator==(())

```
int BTimeUs::operator==(
    const BTimeUs & time ) const [inline]
```

#### 7.94.2.18 operator>()

```
int BTimeUs::operator> (
    const BTimeUs & time ) const [inline]
```

#### 7.94.2.19 operator>=()

```
int BTimeUs::operator>= (
    const BTimeUs & time ) const [inline]
```

#### 7.94.2.20 set() [1/2]

```
void BTimeUs::set (
    BUInt64 microSeconds )
```

Set the time to TAI us.

#### 7.94.2.21 set() [2/2]

```
void BTimeUs::set (
    BUInt year,
    BUInt month,
    BUInt day,
    BUInt hour = 0,
    BUInt minute = 0,
    BUInt second = 0,
    BUInt microSecond = 0 )
```

Set the date and time from UTC.

#### 7.94.2.22 setString()

```
BError BTimeUs::setString (
    const BString dateTime )
```

Sets the date/time from string format.

#### 7.94.2.23 setYearDay()

```
void BTimeUs::setYearDay (
    BUInt year,
    BUInt yearDay,
    BUInt hour = 0,
    BUInt minute = 0,
    BUInt second = 0,
    BUInt microSecond = 0 )
```

Set the date and time from UTC.

### 7.94.3 Member Data Documentation

#### 7.94.3.1 otime

```
BUInt64 BTimeUs::otime [private]
```

Time in microseconds since 1970-01-01 to TAI standard.

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

- [BTimeUs.h](#)
- [BTimeUs.cpp](#)

## 7.95 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](#)

### 7.95.1 Detailed Description

Basic access to a Url.

### 7.95.2 Constructor & Destructor Documentation

#### 7.95.2.1 BUrl()

```
BUrl::BUrl ( )
```

#### 7.95.2.2 ~BUrl()

```
BUrl::~~BUrl ( )
```

## 7.95.3 Member Function Documentation

### 7.95.3.1 readString()

```
BError BUrl::readString (  
    BString url,  
    BString & str )
```

Reads URL.

### 7.95.3.2 writeData()

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

## 7.95.4 Member Data Documentation

### 7.95.4.1 oinit

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

### 7.95.4.2 ores

```
BString BUrl::ores [private]
```

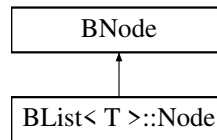
The documentation for this class was generated from the following files:

- [BUrl.h](#)
- [BUrl.cpp](#)

## 7.96 BList< T >::Node Class Reference

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

Inheritance diagram for BList< T >::Node:



### Public Member Functions

- [Node](#) (const T &i)

### Public Attributes

- [T item](#)

### 7.96.1 Constructor & Destructor Documentation

#### 7.96.1.1 Node()

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

### 7.96.2 Member Data Documentation

#### 7.96.2.1 item

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

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

- [BList.h](#)

# Chapter 8

## File Documentation

### 8.1 BArray.h File Reference

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

#### Classes

- class [BArray< T >](#)

#### Macros

- #define [BArrayLoop](#)(list, i) for([BUInt](#) i = 0; i < [list.number\(\)](#); i++)

#### 8.1.1 Macro Definition Documentation

##### 8.1.1.1 BArrayLoop

```
#define BArrayLoop(  
    list,  
    i ) for(BUInt i = 0; i < list.number\(\); i++)
```

### 8.2 BAtomic.h File Reference

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

## Classes

- class [BAtomic< Type >](#)  
*BAtomic class.*

## Typedefs

- typedef [BAtomic< BInt32 > BAtomicInt32](#)
- typedef [BAtomic< BInt64 > BAtomicInt64](#)
- typedef [BAtomic< BUInt32 > BAtomicUInt32](#)
- typedef [BAtomic< BUInt64 > BAtomicUInt64](#)

## 8.2.1 Typedef Documentation

### 8.2.1.1 BAtomicInt32

```
typedef BAtomic<BInt32> BAtomicInt32
```

### 8.2.1.2 BAtomicInt64

```
typedef BAtomic<BInt64> BAtomicInt64
```

### 8.2.1.3 BAtomicUInt32

```
typedef BAtomic<BUInt32> BAtomicUInt32
```

### 8.2.1.4 BAtomicUInt64

```
typedef BAtomic<BUInt64> BAtomicUInt64
```

## 8.3 BAtomicCount.h File Reference

```
#include <bits/atomicity.h>
```



## Classes

- class [BAtomicCount](#)  
*BAtomicCount* class.

## 8.4 BBuffer.cpp File Reference

```
#include <stdlib.h>
#include <memory.h>
#include <BBuffer.h>
#include <BEndian.h>
#include <BTimeStamp.h>
#include <BComplex.h>
```

## Variables

- const int [roundSize](#) = 256

### 8.4.1 Variable Documentation

#### 8.4.1.1 roundSize

```
const int roundSize = 256
```

## 8.5 BBuffer.h File Reference

```
#include <BTypes.h>
#include <BString.h>
#include <BError.h>
#include <BComplex.h>
#include <BEndian.h>
```

## Classes

- class [BBuffer](#)
- class [BBufferStore](#)

## Macros

- #define [BBigEndian](#) 0

## 8.5.1 Macro Definition Documentation

### 8.5.1.1 BBigEndian

```
#define BBigEndian 0
```

## 8.6 BComms.cpp File Reference

```
#include <BComms.h>
```

## 8.7 BComms.h File Reference

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

### Classes

- class [BComms](#)

## 8.8 BComplex.h File Reference

```
#include <BTypes.h>  
#include <complex>  
#include <algorithm>
```

### Typedefs

- typedef std::complex< double > [BComplex](#)
- typedef std::complex< float > [BComplex32](#)
- typedef std::complex< double > [BComplex64](#)

### 8.8.1 Typedef Documentation

### 8.8.1.1 BComplex

```
typedef std::complex<double> BComplex
```

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

### 8.8.1.2 BComplex32

```
typedef std::complex<float> BComplex32
```

### 8.8.1.3 BComplex64

```
typedef std::complex<double> BComplex64
```

## 8.9 BCond.cpp File Reference

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

## 8.10 BCond.h File Reference

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

### Classes

- class [BCond](#)

## 8.11 BCondInt.cpp File Reference

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

### Functions

- static struct timespec [getTimeout](#) (uint32\_t timeOutUs)

## 8.11.1 Function Documentation

### 8.11.1.1 getTimeout()

```
static struct timespec getTimeout (
    uint32_t timeOutUs ) [static]
```

## 8.12 BCondInt.h File Reference

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

### Classes

- class [BCondInt](#)  
*Thread conditional value.*
- class [BCondValue](#)  
*Thread conditional value.*
- class [BCondBool](#)  
*Thread conditional boolean.*
- class [BCondWrap](#)
- class [BCondResource](#)  
*Resource lock.*

## 8.13 BConfig.cpp File Reference

```
#include <BConfig.h>
#include <string.h>
```

## 8.14 BConfig.h File Reference

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

### Classes

- class [BConfig](#)  
*This class implements the configuration file access.*

## 8.15 BCrc16.cpp File Reference

```
#include <BCrc16.h>
```

### Functions

- [BUInt16 bcrc16](#) (void \*buf, [BUInt16 len](#))

### Variables

- static const [BUInt8 table\\_crc\\_hi](#) []
- static const [BUInt8 table\\_crc\\_lo](#) []

### 8.15.1 Function Documentation

#### 8.15.1.1 bcrc16()

```
BUInt16 bcrc16 (
    void * buf,
    BUInt16 len )
```

### 8.15.2 Variable Documentation

#### 8.15.2.1 table\_crc\_hi

```
const BUInt8 table_crc_hi[] [static]
```

##### Initial value:

```
= {
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40
}
```

### 8.15.2.2 table\_crc\_lo

```
const BUInt8 table_crc_lo[] [static]
```

#### Initial value:

```
= {
    0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06,
    0x07, 0xC7, 0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD,
    0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
    0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A,
    0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC, 0x14, 0xD4,
    0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
    0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3,
    0xF2, 0x32, 0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4,
    0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
    0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29,
    0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED,
    0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
    0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60,
    0x61, 0xA1, 0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67,
    0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
    0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68,
    0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E,
    0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
    0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71,
    0x70, 0xB0, 0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92,
    0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
    0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B,
    0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B,
    0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
    0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42,
    0x43, 0x83, 0x41, 0x81, 0x80, 0x40
}
```

## 8.16 BCrc16.h File Reference

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

### Functions

- [BUInt16 bcrc16](#) (void \*buf, BUInt16 len)

### 8.16.1 Function Documentation

#### 8.16.1.1 bcrc16()

```
BUInt16 brcr16 (
    void * buf,
    BUInt16 len )
```

## 8.17 BCrc32.cpp File Reference

```
#include <BCrc32.h>
```

## Functions

- [BUInt32 bcrc32](#) ([BUInt32](#) crc, const void \*buf, [BUInt32](#) len)

## Variables

- static [BUInt32 crc32\\_tab](#) []

### 8.17.1 Function Documentation

#### 8.17.1.1 bcrc32()

```
BUInt32 bcrc32 (  
    BUInt32 crc,  
    const void * buf,  
    BUInt32 len )
```

### 8.17.2 Variable Documentation

#### 8.17.2.1 crc32\_tab

```
BUInt32 crc32_tab[] [static]
```

## 8.18 BCrc32.h File Reference

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

## Functions

- [BUInt32 bcrc32](#) ([BUInt32](#) crc, const void \*buf, [BUInt32](#) len)

### 8.18.1 Function Documentation

### 8.18.1.1 bcrc32()

```
BUInt32 bcrc32 (
    BUInt32 crc,
    const void * buf,
    BUInt32 len )
```

## 8.19 BDate-1.cpp File Reference

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

### Functions

- void [toBString](#) (BDate &v, BString &s)
- void [fromBString](#) (BString &s, BDate &v)

### Variables

- static int [mon\\_yday](#) [2][13]

## 8.19.1 Function Documentation

### 8.19.1.1 fromBString()

```
void fromBString (
    BString & s,
    BDate & v )
```

### 8.19.1.2 toBString()

```
void toBString (
    BDate & v,
    BString & s )
```

## 8.19.2 Variable Documentation



### 8.19.2.1 mon\_yday

```
int mon_yday[2][13] [static]
```

#### Initial value:

```
= {  
    { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334, 365 },  
    { 0, 31, 60, 91, 121, 152, 182, 213, 244, 274, 305, 335, 366 }  
}
```

## 8.20 BDate.cpp File Reference

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

### Functions

- void [toBString](#) (BDate &v, BString &s)
- void [fromBString](#) (BString &s, BDate &v)

### Variables

- static int [mon\\_yday](#) [2][13]

## 8.20.1 Function Documentation

### 8.20.1.1 fromBString()

```
void fromBString (  
    BString & s,  
    BDate & v )
```

### 8.20.1.2 toBString()

```
void toBString (  
    BDate & v,  
    BString & s )
```

## 8.20.2 Variable Documentation

### 8.20.2.1 mon\_yday

```
int mon_yday[2][13] [static]
```

#### Initial value:

```
= {  
    { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334, 365 },  
    { 0, 31, 60, 91, 121, 152, 182, 213, 244, 274, 305, 335, 366 }  
}
```

## 8.21 BDate.h File Reference

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

### Classes

- class [BDate](#)

### Functions

- void [toBString](#) ([BDate](#) &v, [BString](#) &s)
- void [fromBString](#) ([BString](#) &s, [BDate](#) &v)

### 8.21.1 Function Documentation

#### 8.21.1.1 fromBString()

```
void fromBString (  
    BString & s,  
    BDate & v )
```

### 8.21.1.2 toBString()

```
void toBString (
    BDate & v,
    BString & s )
```

## 8.22 BDebug.cpp File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/time.h>
#include <stdarg.h>
#include <fcntl.h>
#include <ctype.h>
#include <BDebug.h>
```

### Functions

- void [bhd8](#) (const void \*[data](#), unsigned int [n](#))
- void [bhd8a](#) (const void \*[data](#), unsigned int [n](#))
- void [bhda8](#) (const void \*[data](#), unsigned int [n](#))
- void [bhd32](#) (const void \*[data](#), unsigned int [n](#))
- void [bhda32](#) (const void \*[data](#), unsigned int [n](#))
- double [getTime](#) ()
- void [setDebug](#) (int [d](#))

### Variables

- int [bdebug](#)

## 8.22.1 Function Documentation

### 8.22.1.1 bhd32()

```
void bhd32 (
    const void * data,
    unsigned int n )
```

**8.22.1.2 bhd8()**

```
void bhd8 (
    const void * data,
    unsigned int n )
```

**8.22.1.3 bhd8a()**

```
void bhd8a (
    const void * data,
    unsigned int n )
```

**8.22.1.4 bhda32()**

```
void bhda32 (
    const void * data,
    unsigned int n )
```

**8.22.1.5 bhda8()**

```
void bhda8 (
    const void * data,
    unsigned int n )
```

**8.22.1.6 getTime()**

```
double getTime ( )
```

**8.22.1.7 setDebug()**

```
void setDebug (
    int d )
```

**8.22.2 Variable Documentation**

### 8.22.2.1 bdebug

```
int bdebug
```

## 8.23 BDebug.h File Reference

```
#include <stdio.h>
#include <time.h>
#include <syslog.h>
```

### Classes

- class [BDebugBacktrace](#)

### Macros

- #define [BDebug\\_STD](#) 0x000001
- #define [dprintf](#)(level, fmt, a...)  
*General debug functions.*
- #define [nprintf](#)(fmt, a...) syslog(LOG\_NOTICE, fmt, ##a)  
*Warnings and errors logging.*
- #define [wprintf](#)(fmt, a...) syslog(LOG\_WARNING, fmt, ##a)
- #define [eprintf](#)(fmt, a...) syslog(LOG\_ERR, fmt, ##a)
- #define [dl1printf](#)(fmt, a...)
- #define [dl2printf](#)(fmt, a...)
- #define [dl3printf](#)(fmt, a...)
- #define [dl4printf](#)(fmt, a...)

### Functions

- void [bhd8](#) (const void \*data, unsigned int n)
- void [bhd8a](#) (const void \*data, unsigned int n)
- void [bhda8](#) (const void \*data, unsigned int n)
- void [bhd32](#) (const void \*data, unsigned int n)
- void [bhds32](#) (const void \*data, unsigned int n)
- double [getTime](#) ()
- void [setDebug](#) (int debug)
- void [tprintf](#) (int log, const char \*fmt,...)
- pid\_t [gettid](#) ()

### Variables

- int [bdebug](#)

## 8.23.1 Macro Definition Documentation

### 8.23.1.1 BDebug\_STD

```
#define BDebug_STD 0x000001
```

### 8.23.1.2 dl1printf

```
#define dl1printf(  
    fmt,  
    a... )
```

### 8.23.1.3 dl2printf

```
#define dl2printf(  
    fmt,  
    a... )
```

### 8.23.1.4 dl3printf

```
#define dl3printf(  
    fmt,  
    a... )
```

### 8.23.1.5 dl4printf

```
#define dl4printf(  
    fmt,  
    a... )
```

### 8.23.1.6 dprintf

```
#define dprintf(  
    level,  
    fmt,  
    a... )
```

General debug functions.

### 8.23.1.7 eprintf

```
#define eprintf(  
    fmt,  
    a... ) syslog(LOG_ERR, fmt, ##a)
```

### 8.23.1.8 nprintf

```
#define nprintf(  
    fmt,  
    a... ) syslog(LOG_NOTICE, fmt, ##a)
```

Warnings and errors logging.

### 8.23.1.9 wprintf

```
#define wprintf(  
    fmt,  
    a... ) syslog(LOG_WARNING, fmt, ##a)
```

## 8.23.2 Function Documentation

### 8.23.2.1 bhd32()

```
void bhd32 (  
    const void * data,  
    unsigned int n )
```

### 8.23.2.2 bhd8()

```
void bhd8 (  
    const void * data,  
    unsigned int n )
```

### 8.23.2.3 bhd8a()

```
void bhd8a (
    const void * data,
    unsigned int n )
```

### 8.23.2.4 bhda8()

```
void bhda8 (
    const void * data,
    unsigned int n )
```

### 8.23.2.5 bhds32()

```
void bhds32 (
    const void * data,
    unsigned int n )
```

### 8.23.2.6 gettid()

```
pid_t gettid ( )
```

### 8.23.2.7 getTime()

```
double getTime ( )
```

### 8.23.2.8 setDebug()

```
void setDebug (
    int debug )
```

### 8.23.2.9 tprintf()

```
void tprintf (
    int log,
    const char * fmt,
    ... )
```



### 8.23.3 Variable Documentation

#### 8.23.3.1 bdebug

```
int bdebug
```

## 8.24 BDict.cpp File Reference

```
#include <BDict.h>
```

### Functions

- void [toBString](#) (const [BDictString](#) &v, [BString](#) &s)
- void [fromBString](#) (const [BString](#) &str, [BDictString](#) &v)
- [BString bdictStringToString](#) (const [BDictString](#) &dict)

### 8.24.1 Function Documentation

#### 8.24.1.1 bdictStringToString()

```
BString bdictStringToString (  
    const BDictString & dict )
```

#### 8.24.1.2 fromBString()

```
void fromBString (  
    const BString & str,  
    BDictString & v )
```

#### 8.24.1.3 toBString()

```
void toBString (  
    const BDictString & v,  
    BString & s )
```

## 8.25 BDict.h File Reference

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

### Classes

- class [BDictItem< Type >](#)  
*Template based Dictionary class.*
- class [BDict< Type >](#)

### Typedefs

- typedef [BDict< BString >](#) [BDictString](#)

### Functions

- void [toBString](#) (const [BDictString](#) &v, [BString](#) &s)
- void [fromBString](#) (const [BString](#) &s, [BDictString](#) &v)
- [BString](#) [bdictStringToString](#) (const [BDictString](#) &dict)

### 8.25.1 Typedef Documentation

#### 8.25.1.1 BDictString

```
typedef BDict<BString> BDictString
```

### 8.25.2 Function Documentation

#### 8.25.2.1 bdictStringToString()

```
BString bdictStringToString (  
    const BDictString & dict )
```

### 8.25.2.2 fromBString()

```
void fromBString (
    const BString & s,
    BDictString & v )
```

### 8.25.2.3 toBString()

```
void toBString (
    const BDictString & v,
    BString & s )
```

## 8.26 BDictMap.h File Reference

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

### Classes

- class [BDictMap< Value >](#)

### Typedefs

- typedef [BDictMap< BString >](#) [BDictMapString](#)

### 8.26.1 Typedef Documentation

#### 8.26.1.1 BDictMapString

```
typedef BDictMap<BString> BDictMapString
```

## 8.27 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](#)

### 8.27.1 Function Documentation

#### 8.27.1.1 wild()

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

### 8.27.2 Variable Documentation

#### 8.27.2.1 wildString

```
BString wildString [static]
```

## 8.28 BDir.h File Reference

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

## Classes

- class [BDir](#)  
*File system directory class.*

## 8.29 BDuration.cpp File Reference

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

## 8.30 BDuration.h File Reference

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

### Classes

- class [BDuration](#)

## 8.31 BEndian.cpp File Reference

```
#include <BEndian.h>
#include <memory.h>
```

### Functions

- void [bswap\\_copy](#) (int swap, const void \*src, void \*dst, [BUInt32](#) nBytes, const char \*swapType)

### 8.31.1 Function Documentation

#### 8.31.1.1 bswap\_copy()

```
void bswap_copy (
    int swap,
    const void * src,
    void * dst,
    BUInt32 nBytes,
    const char * swapType )
```

## 8.32 BEndian.h File Reference

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

## Macros

- #define [htobe16\(x\)](#) `__bswap_16(x)`
- #define [htole16\(x\)](#) `(x)`
- #define [be16toh\(x\)](#) `__bswap_16(x)`
- #define [le16toh\(x\)](#) `(x)`
- #define [htobe32\(x\)](#) `__bswap_32(x)`
- #define [htole32\(x\)](#) `(x)`
- #define [be32toh\(x\)](#) `__bswap_32(x)`
- #define [le32toh\(x\)](#) `(x)`
- #define [htobe64\(x\)](#) `__bswap_64(x)`
- #define [htole64\(x\)](#) `(x)`
- #define [be64toh\(x\)](#) `__bswap_64(x)`
- #define [le64toh\(x\)](#) `(x)`

## Functions

- void [bswap\\_p8](#) (const void \*s, void \*d)
- void [bswap\\_p16](#) (const void \*s, void \*d)
- void [bswap\\_p32](#) (const void \*s, void \*d)
- void [bswap\\_p64](#) (const void \*s, void \*d)
- void [bswap\\_copy](#) (int swap, const void \*src, void \*dst, [BUInt32](#) nBytes, const char \*swapType)
- uint16\_t [htole](#) (uint16\_t v)
- int16\_t [htole](#) (int16\_t v)
- uint32\_t [htole](#) (uint32\_t v)
- int32\_t [htole](#) (int32\_t v)
- uint64\_t [htole](#) (uint64\_t v)
- int64\_t [htole](#) (int64\_t v)
- double [htole](#) (double v)
- float [htole](#) (float v)
- uint16\_t [htobe](#) (uint16\_t v)
- int16\_t [htobe](#) (int16\_t v)
- uint32\_t [htobe](#) (uint32\_t v)
- int32\_t [htobe](#) (int32\_t v)
- uint64\_t [htobe](#) (uint64\_t v)
- int64\_t [htobe](#) (int64\_t v)
- double [htobe](#) (double v)
- float [htobe](#) (float v)
- uint16\_t [letoh](#) (uint16\_t v)
- int16\_t [letoh](#) (int16\_t v)
- uint32\_t [letoh](#) (uint32\_t v)
- int32\_t [letoh](#) (int32\_t v)
- uint64\_t [letoh](#) (uint64\_t v)
- int64\_t [letoh](#) (int64\_t v)
- double [letoh](#) (double v)
- float [letoh](#) (float v)
- uint16\_t [betoh](#) (uint16\_t v)
- int16\_t [betoh](#) (int16\_t v)
- uint32\_t [betoh](#) (uint32\_t v)
- int32\_t [betoh](#) (int32\_t v)
- uint64\_t [betoh](#) (uint64\_t v)
- int64\_t [betoh](#) (int64\_t v)
- double [betoh](#) (double v)
- float [betoh](#) (float v)

## 8.32.1 Macro Definition Documentation

### 8.32.1.1 be16toh

```
#define be16toh(  
    x ) __bswap_16 (x)
```

### 8.32.1.2 be32toh

```
#define be32toh(  
    x ) __bswap_32 (x)
```

### 8.32.1.3 be64toh

```
#define be64toh(  
    x ) __bswap_64 (x)
```

### 8.32.1.4 htobe16

```
#define htobe16(  
    x ) __bswap_16 (x)
```

### 8.32.1.5 htobe32

```
#define htobe32(  
    x ) __bswap_32 (x)
```

### 8.32.1.6 htobe64

```
#define htobe64(  
    x ) __bswap_64 (x)
```

### 8.32.1.7 htole16

```
#define htole16(  
    x ) (x)
```

### 8.32.1.8 htole32

```
#define htole32(  
    x ) (x)
```

### 8.32.1.9 htole64

```
#define htole64(  
    x ) (x)
```

### 8.32.1.10 le16toh

```
#define le16toh(  
    x ) (x)
```

### 8.32.1.11 le32toh

```
#define le32toh(  
    x ) (x)
```

### 8.32.1.12 le64toh

```
#define le64toh(  
    x ) (x)
```

## 8.32.2 Function Documentation



**8.32.2.1 betoh()** [1/8]

```
uint16_t betoh (  
    uint16_t v ) [inline]
```

**8.32.2.2 betoh()** [2/8]

```
int16_t betoh (  
    int16_t v ) [inline]
```

**8.32.2.3 betoh()** [3/8]

```
uint32_t betoh (  
    uint32_t v ) [inline]
```

**8.32.2.4 betoh()** [4/8]

```
int32_t betoh (  
    int32_t v ) [inline]
```

**8.32.2.5 betoh()** [5/8]

```
uint64_t betoh (  
    uint64_t v ) [inline]
```

**8.32.2.6 betoh()** [6/8]

```
int64_t betoh (  
    int64_t v ) [inline]
```

**8.32.2.7 betoh()** [7/8]

```
double betoh (  
    double v ) [inline]
```

### 8.32.2.8 betoh() [8/8]

```
float betoh (  
    float v ) [inline]
```

### 8.32.2.9 bswap\_copy()

```
void bswap_copy (  
    int swap,  
    const void * src,  
    void * dst,  
    BUInt32 nBytes,  
    const char * swapType )
```

### 8.32.2.10 bswap\_p16()

```
void bswap_p16 (  
    const void * s,  
    void * d ) [inline]
```

### 8.32.2.11 bswap\_p32()

```
void bswap_p32 (  
    const void * s,  
    void * d ) [inline]
```

### 8.32.2.12 bswap\_p64()

```
void bswap_p64 (  
    const void * s,  
    void * d ) [inline]
```

### 8.32.2.13 bswap\_p8()

```
void bswap_p8 (  
    const void * s,  
    void * d ) [inline]
```

**8.32.2.14 htobe()** [1/8]

```
uint16_t htobe (  
    uint16_t v ) [inline]
```

**8.32.2.15 htobe()** [2/8]

```
int16_t htobe (  
    int16_t v ) [inline]
```

**8.32.2.16 htobe()** [3/8]

```
uint32_t htobe (  
    uint32_t v ) [inline]
```

**8.32.2.17 htobe()** [4/8]

```
int32_t htobe (  
    int32_t v ) [inline]
```

**8.32.2.18 htobe()** [5/8]

```
uint64_t htobe (  
    uint64_t v ) [inline]
```

**8.32.2.19 htobe()** [6/8]

```
int64_t htobe (  
    int64_t v ) [inline]
```

**8.32.2.20 htobe()** [7/8]

```
double htobe (  
    double v ) [inline]
```

**8.32.2.21 htobe()** [8/8]

```
float htobe (  
    float v ) [inline]
```

**8.32.2.22 htobe()** [1/8]

```
uint16_t htobe (  
    uint16_t v ) [inline]
```

**8.32.2.23 htobe()** [2/8]

```
int16_t htobe (  
    int16_t v ) [inline]
```

**8.32.2.24 htobe()** [3/8]

```
uint32_t htobe (  
    uint32_t v ) [inline]
```

**8.32.2.25 htobe()** [4/8]

```
int32_t htobe (  
    int32_t v ) [inline]
```

**8.32.2.26 htobe()** [5/8]

```
uint64_t htobe (  
    uint64_t v ) [inline]
```

**8.32.2.27 htobe()** [6/8]

```
int64_t htobe (  
    int64_t v ) [inline]
```

**8.32.2.28 htole()** [7/8]

```
double htole (  
    double v ) [inline]
```

**8.32.2.29 htole()** [8/8]

```
float htole (  
    float v ) [inline]
```

**8.32.2.30 letoh()** [1/8]

```
uint16_t letoh (  
    uint16_t v ) [inline]
```

**8.32.2.31 letoh()** [2/8]

```
int16_t letoh (  
    int16_t v ) [inline]
```

**8.32.2.32 letoh()** [3/8]

```
uint32_t letoh (  
    uint32_t v ) [inline]
```

**8.32.2.33 letoh()** [4/8]

```
int32_t letoh (  
    int32_t v ) [inline]
```

**8.32.2.34 letoh()** [5/8]

```
uint64_t letoh (  
    uint64_t v ) [inline]
```

### 8.32.2.35 letoh() [6/8]

```
int64_t letoh (  
    int64_t v ) [inline]
```

### 8.32.2.36 letoh() [7/8]

```
double letoh (  
    double v ) [inline]
```

### 8.32.2.37 letoh() [8/8]

```
float letoh (  
    float v ) [inline]
```

## 8.33 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>
```

## 8.34 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.*

## 8.35 BError.cpp File Reference

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

## 8.36 BError.h File Reference

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

### Classes

- class [BError](#)

### Enumerations

- enum [BErrorNum](#) {  
[ErrorOk](#) = 0, [ErrorMisc](#) = 1, [ErrorWarning](#) = 2, [ErrorParam](#) = 3,  
[ErrorTimeout](#) = 4, [ErrorNotAvailable](#) = 5, [ErrorData](#) = 6, [ErrorChecksum](#) = 7,  
[ErrorOverrun](#) = 8, [ErrorUnderrun](#) = 9, [ErrorInit](#) = 10, [ErrorConfig](#) = 11,  
[ErrorNotImplemented](#) = 12, [ErrorResourceLimit](#) = 13, [ErrorEndOfFile](#) = 14, [ErrorFile](#) = 15,  
[ErrorFormat](#) = 16, [ErrorComms](#) = 17, [ErrorAccessDenied](#) = 18, [ErrorNoData](#) = 19,  
[ErrorEndOfData](#) = 20, [ErrorDataPresent](#) = 21, [ErrorAppBase](#) = 64 }

### 8.36.1 Enumeration Type Documentation

#### 8.36.1.1 BErrorNum

```
enum BErrorNum
```

#### Enumerator

<a href="#">ErrorOk</a>	
<a href="#">ErrorMisc</a>	
<a href="#">ErrorWarning</a>	
<a href="#">ErrorParam</a>	
<a href="#">ErrorTimeout</a>	
<a href="#">ErrorNotAvailable</a>	
<a href="#">ErrorData</a>	
<a href="#">ErrorChecksum</a>	
<a href="#">ErrorOverrun</a>	
<a href="#">ErrorUnderrun</a>	
<a href="#">ErrorInit</a>	
<a href="#">ErrorConfig</a>	
<a href="#">ErrorNotImplemented</a>	

## Enumerator

ErrorResourceLimit	
ErrorEndOfFile	
ErrorFile	
ErrorFormat	
ErrorComms	
ErrorAccessDenied	
ErrorNoData	
ErrorEndOfData	
ErrorDataPresent	
ErrorAppBase	

### 8.37 BErrorTime.cpp File Reference

```
#include <BErrorTime.h>
```

### 8.38 BErrorTime.h File Reference

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

## Classes

- class [BErrorTime](#)  
*Error return class.*

### 8.39 BEvent.cpp File Reference

```
#include <BEvent.h>  
#include <BPoll.h>  
#include <stdlib.h>  
#include <unistd.h>  
#include <sys/ioctl.h>
```

### 8.40 BEvent.h File Reference

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



## Classes

- class [BEvent](#)
- class [BEventPipe](#)

*This class provides an interface for sending simple integer events via a pipe file descriptor.*

## Typedefs

- typedef [BQueue< BEvent >](#) [BEventQueue](#)

*This class provides an interface for sending simple integer events via a [BQueue](#).*

### 8.40.1 Typedef Documentation

#### 8.40.1.1 BEventQueue

```
typedef BQueue<BEvent> BEventQueue
```

This class provides an interface for sending simple integer events via a [BQueue](#).

## 8.41 BEvent1.cpp File Reference

```
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <BEvent1.h>
#include <BPoll.h>
```

## 8.42 BEvent1.h File Reference

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

## Classes

- class [BEvent1](#)

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

- class [BEvent1Error](#)
- class [BEvent1Pipe](#)

*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 [BEvent1Int](#)

*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.*

## Enumerations

- enum [BEvent1Type](#) { [BEvent1TypeNone](#), [BEvent1TypeInt](#), [BEvent1TypeError](#) }

### 8.42.1 Enumeration Type Documentation

#### 8.42.1.1 BEvent1Type

enum [BEvent1Type](#)

##### Enumerator

<a href="#">BEvent1TypeNone</a>	
<a href="#">BEvent1TypeInt</a>	
<a href="#">BEvent1TypeError</a>	

### 8.43 BFifo.h File Reference

```
#include <BTypes.h>
#include <BError.h>
#include <BFifo.inc>
```

#### Classes

- class [BFifo< Type >](#)

### 8.44 BFifo.inc File Reference

### 8.45 BFifoCirc.cpp File Reference

```
#include <BFifoCirc.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/mman.h>
```

#### Macros

- #define [dprintf](#)(fmt, a...)

## 8.45.1 Macro Definition Documentation

### 8.45.1.1 dprintf

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

## 8.46 BFifoCirc.h File Reference

```
#include <stdint.h>  
#include <BError.h>  
#include <BCondInt.h>  
#include <BMutex.h>  
#include <BFifoCirc.inc>
```

### Classes

- class [BFifoCircPos](#)  
*This class implements a pointer into the Fifo's circular buffer.*
- class [BFifoCirc< Type >](#)  
*This class implements a thread safe FIFO buffer.*

## 8.47 BFifoCirc.inc File Reference

## 8.48 BFile.cpp File Reference

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

### Macros

- #define [STRBUF](#) 10240

## 8.48.1 Macro Definition Documentation

#### 8.48.1.1 STRBUF

```
#define STRBUF 10240
```

### 8.49 BFile.h File Reference

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

#### Classes

- class [BFile](#)  
*File operations class.*

### 8.50 BFileCsv.cpp File Reference

```
#include <BFileCsv.h>  
#include <errno.h>
```

### 8.51 BFileCsv.h File Reference

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

#### Classes

- class [BFileCsv](#)

### 8.52 BFileData.cpp File Reference

```
#include <BFileCsv.h>  
#include <BFileData.h>  
#include <errno.h>
```

### 8.53 BFileData.h File Reference

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

## Classes

- class [BFileData](#)

## 8.54 BList.h File Reference

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

## Classes

- class [BNode](#)
- class [BIter](#)  
*Iterator for BList.*
- class [BList< T >](#)  
*Template based list class.*
- class [BList< T >::Node](#)

## Macros

- `#define BListLoop(list, i) for(BIter i = list.begin(); !list.isEnd(i); list.next(i))`

### 8.54.1 Macro Definition Documentation

#### 8.54.1.1 BListLoop

```
#define BListLoop(  
    list,  
    i ) for(BIter i = list.begin(); !list.isEnd(i); list.next(i))
```

## 8.55 BList\_func.h File Reference

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

## 8.56 BMutex.cpp File Reference

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

## Macros

- `#define MDEBUG 0`

## 8.56.1 Macro Definition Documentation

### 8.56.1.1 MDEBUG

```
#define MDEBUG 0
```

## 8.57 BMutex.h File Reference

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

## Classes

- class [BMutex](#)
- class [BMutexLock](#)

## 8.58 BMySQL.cpp File Reference

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

## 8.59 BMySQL.h File Reference

```
#include <BTypes.h>  
#include <BError.h>  
#include <BDict.h>  
#include <BMutex.h>  
#include <mysql/mysql.h>
```

## Classes

- class [BMySQL](#)

## 8.60 BNameValue.h File Reference

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

### Classes

- class [BNameValue< T >](#)
- class [BNameValueList< T >](#)

## 8.61 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>
```

### Macros

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

### Variables

- const int `boapPort = 12000`  
*The default BOAP connection port.*

### 8.61.1 Macro Definition Documentation

#### 8.61.1.1 APIVERSION\_TEST

```
#define APIVERSION_TEST 1
```

### 8.61.1.2 DEBUG

```
#define DEBUG 0
```

### 8.61.1.3 dprintf

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

### 8.61.1.4 IS\_BIG\_ENDIAN

```
#define IS_BIG_ENDIAN 1
```

## 8.61.2 Variable Documentation

### 8.61.2.1 boapPort

```
const int boapPort = 12000
```

The default BOAP connection port.

## 8.62 Boap.h File Reference

```
#include <stdint.h>  
#include <BTypes.h>  
#include <BPoll.h>  
#include <BSocket.h>  
#include <BThread.h>  
#include <BError.h>  
#include <BEvent1.h>  
#include <BMutex.h>  
#include <BTimeStamp.h>  
#include <BBuffer.h>
```



## Classes

- struct [BoapPacketHead](#)
- class [BoapPacket](#)
- class [BoapClientObject](#)
- class [BoapSignalObject](#)
- class [BoapServiceEntry](#)
- class [BoapServerConnection](#)
- class [BoapServer](#)
- class [BoapFuncEntry](#)
- class [BoapServiceObject](#)

## Namespaces

- [Boapns](#)

## Typedefs

- typedef [BUInt32](#) [BoapService](#)
- typedef [BError](#)([BoapServiceObject](#)::\* [BoapFunc](#)) ([BoapServerConnection](#) \*conn, [BoapPacket](#) &rx, [BoapPacket](#) &tx)

## Enumerations

- enum [BoapType](#) {  
    [BoapTypeRpc](#), [BoapTypeRpcReply](#), [BoapTypeSignal](#), [BoapTypeRpcError](#),  
    [BoapTypeRpc](#), [BoapTypeSignal](#) }
- enum [BoapPriority](#) { [BoapPriorityLow](#), [BoapPriorityNormal](#), [BoapPriorityHigh](#) }

## Variables

- const [BUInt32](#) [BoapMagic](#) = 0x424F4100

### 8.62.1 Typedef Documentation

#### 8.62.1.1 BoapFunc

```
typedef BError(BoapServiceObject::* BoapFunc) (BoapServerConnection *conn, BoapPacket &rx,  
BoapPacket &tx)
```

#### 8.62.1.2 BoapService

```
typedef BUInt32 BoapService
```

### 8.62.2 Enumeration Type Documentation

#### 8.62.2.1 BoapPriority

```
enum BoapPriority
```

**Enumerator**

BoapPriorityLow	
BoapPriorityNormal	
BoapPriorityHigh	

**8.62.2.2 BoapType**

enum [BoapType](#)

**Enumerator**

BoapTypeRpc	
BoapTypeRpcReply	
BoapTypeSignal	
BoapTypeRpcError	
BoapTypeRpc	
BoapTypeSignal	

**8.62.3 Variable Documentation****8.62.3.1 BoapMagic**

```
const BUInt32 BoapMagic = 0x424F4100
```

**8.63 BoapMc.cpp File Reference**

```
#include <BoapMc.h>
#include <BCrc16.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
```

**Macros**

- #define [DEBUG\\_LOCAL](#) 0
- #define [DEBUG\\_LOCAL1](#) 0
- #define [dlprintf](#)(fmt, a...)
- #define [dl1printf](#)(fmt, a...)

## 8.63.1 Macro Definition Documentation

### 8.63.1.1 DEBUG\_LOCAL

```
#define DEBUG_LOCAL 0
```

### 8.63.1.2 DEBUG\_LOCAL1

```
#define DEBUG_LOCAL1 0
```

### 8.63.1.3 dl1printf

```
#define dl1printf(  
    fmt,  
    a... )
```

### 8.63.1.4 dlprintf

```
#define dlprintf(  
    fmt,  
    a... )
```

## 8.64 BoapMc.h File Reference

```
#include <BTypes.h>  
#include <BMutex.h>  
#include <BSemaphore.h>  
#include <BQueue.h>  
#include <BFifo.h>  
#include <BComms.h>
```

### Classes

- struct [BoapMcPacketHead](#)
- class [BoapMcPacket](#)
- class [BoapMcClientObject](#)
- class [BoapMcSignalObject](#)
- class [BoapMcServiceObject](#)
- class [BoapMcComms](#)

## Enumerations

- enum `BoapMcType` { `BoapMcTypeRequest` = 0x00, `BoapMcTypeReply` = 0x80 }

## Functions

- struct `BoapMcPacketHead` `__attribute__` ((aligned(8), packed))

## Variables

- `BUInt8` `length`
- `BUInt8` `addressTo`
- `BUInt8` `addressFrom`
- `BUInt8` `cmd`
- `BUInt16` `error`
- `BUInt16` `checksum`
- class `BoapMcPacket` `__attribute__`

### 8.64.1 Enumeration Type Documentation

#### 8.64.1.1 BoapMcType

```
enum BoapMcType
```

##### Enumerator

<code>BoapMcTypeRequest</code>	
<code>BoapMcTypeReply</code>	

### 8.64.2 Function Documentation

#### 8.64.2.1 `__attribute__()`

```
struct BoapMcPacketHead __attribute__ (
    (aligned(8), packed) )
```

### 8.64.3 Variable Documentation

### 8.64.3.1 \_\_attribute\_\_

struct `BoapMc1Error` `__attribute__`

### 8.64.3.2 addressFrom

`BUInt8` `addressFrom`

### 8.64.3.3 addressTo

`BUInt8` `addressTo`

### 8.64.3.4 checksum

`BUInt16` `checksum`

### 8.64.3.5 cmd

`BUInt8` `cmd`

### 8.64.3.6 error

`BUInt16` `error`

### 8.64.3.7 length

`BUInt8` `length`

## 8.65 BoapMc1.cpp File Reference

```
#include <BoapMc1.h>
#include <BSys.h>
#include <BCrc32.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <BDebug.h>
```

### Macros

- `#define BDEBUGL1 0`
- `#define BDEBUGL2 0`

### 8.65.1 Macro Definition Documentation

#### 8.65.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

#### 8.65.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

## 8.66 BoapMc1.h File Reference

```
#include <BTypes.h>
#include <BMutex.h>
#include <BSemaphore.h>
#include <BQueue.h>
#include <BFifo.h>
#include <BComms.h>
```

### Classes

- struct [BoapMc1PacketHead](#)
- class [BoapMc1Packet](#)
- struct [BoapMc1Error](#)
- class [BoapMc1Comms](#)

## Enumerations

- enum `BoapMc1Type` { `BoapMc1TypeRequest` = 0x0000, `BoapMc1TypeReply` = 0x8000 }

## Functions

- struct `BoapMc1PacketHead` `__attribute__((aligned(8), packed))`
- `BUInt32 boapMc1CommsRoundupLen` (`BUInt32 len`)

## Variables

- const `BUInt16 BoapMc1Magic` = 0x5542
- `BUInt16 magic`
- `BUInt16 length`
- `BUInt16 addressTo`  
*Do we want addresses ?*
- `BUInt16 addressFrom`
- `BUInt16 cmd`
- `BInt16 error`  
*Not used. Maybe return a `BError` object ?*
- `BUInt32 checksum`  
*Not used. Not sure we want this here or at the very end of the packet.*
- `BoapMc1PacketHead head`
- char `data` [8]
- `BInt16 number`  
*The error number.*
- char `string` [32]  
*The error string.*
- class `BoapMc1Comms` `__attribute__((aligned(8), packed))`

## 8.66.1 Enumeration Type Documentation

### 8.66.1.1 BoapMc1Type

enum `BoapMc1Type`

#### Enumerator

<code>BoapMc1TypeRequest</code>	
<code>BoapMc1TypeReply</code>	

## 8.66.2 Function Documentation

### 8.66.2.1 `__attribute__()`

```
struct BoapMc1PacketHead __attribute__ (  
    (aligned(8), packed) )
```

### 8.66.2.2 `boapMc1CommsRoundupLen()`

```
BUInt32 boapMc1CommsRoundupLen (  
    BUInt32 len ) [inline]
```

## 8.66.3 Variable Documentation

### 8.66.3.1 `__attribute__`

```
class BoapMc1Comms __attribute__
```

### 8.66.3.2 `addressFrom`

```
BUInt16 addressFrom
```

### 8.66.3.3 `addressTo`

```
BUInt16 addressTo
```

Do we want addresses ?

### 8.66.3.4 `BoapMc1Magic`

```
const BUInt16 BoapMc1Magic = 0x5542
```



### 8.66.3.5 checksum

[BUInt32](#) checksum

Not used. Not sure we want this here or at the very end of the packet.

### 8.66.3.6 cmd

[BUInt16](#) cmd

### 8.66.3.7 data

char data[8]

### 8.66.3.8 error

[BInt16](#) error

Not used. Maybe return a [BError](#) object ?

### 8.66.3.9 head

[BoapMc1PacketHead](#) head

### 8.66.3.10 length

[BUInt16](#) length

### 8.66.3.11 magic

[BUInt16](#) magic

### 8.66.3.12 number

`BInt16` number

The error number.

### 8.66.3.13 string

`char string[32]`

The error string.

## 8.67 BoapnsC.cpp File Reference

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

### Namespaces

- [Boapns](#)

## 8.68 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>
```

### Classes

- class [Boapns::Boapns](#)

### Namespaces

- [Boapns](#)

### Variables

- const [BUInt32 Boapns::apiVersion](#) = 0

## 8.69 BoapnsD.cpp File Reference

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

### Namespaces

- [Boapns](#)

## 8.70 BoapnsD.h File Reference

```
#include <Boap.h>
#include <BObj.h>
#include <BDate.h>
#include <BTimeStamp.h>
#include <BComplex.h>
#include <BList.h>
#include <BArray.h>
```

### Classes

- class [Boapns::BoapEntry](#)

### Namespaces

- [Boapns](#)

## 8.71 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>
```

### Macros

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

## Variables

- const int [roundSize](#) = 256

## 8.71.1 Macro Definition Documentation

### 8.71.1.1 DEBUG

```
#define DEBUG 0
```

### 8.71.1.2 dprintf

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

## 8.71.2 Variable Documentation

### 8.71.2.1 roundSize

```
const int roundSize = 256
```

## 8.72 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](#), [BoapTypeRpcError](#),  
[BoapTypeRpc](#), [BoapTypeSignal](#) }

### 8.72.1 Typedef Documentation

#### 8.72.1.1 BoapFunc

```
typedef BError(BoapServiceObject::* BoapFunc) (BoapPacket &rx, BoapPacket &tx)
```

#### 8.72.1.2 BoapService

```
typedef uint32_t BoapService
```

#### 8.72.1.3 Double

```
typedef double Double
```

#### 8.72.1.4 Int16

```
typedef int16_t Int16
```

### 8.72.1.5 Int32

```
typedef int32_t Int32
```

### 8.72.1.6 Int8

```
typedef int8_t Int8
```

### 8.72.1.7 UInt16

```
typedef uint16_t UInt16
```

### 8.72.1.8 UInt32

```
typedef uint32_t UInt32
```

### 8.72.1.9 UInt8

```
typedef uint8_t UInt8
```

## 8.72.2 Enumeration Type Documentation

### 8.72.2.1 BoapType

```
enum BoapType
```

#### Enumerator

BoapTypeRpc	
BoapTypeRpcReply	
BoapTypeSignal	
BoapTypeRpcError	
BoapTypeRpc	
BoapTypeSignal	

## 8.73 BObj.cpp File Reference

```
#include <BObj.h>
```

## 8.74 BObj.h File Reference

```
#include <BTypes.h>
#include <BDict.h>
#include <BString.h>
#include <BError.h>
```

### Classes

- class [BObj](#)

## 8.75 BObjStringFormat.cpp File Reference

```
#include <BObjStringFormat.h>
#include <BTime.h>
#include <math.h>
```

### Functions

- [BString toBString](#) (BString n, Bool v)
- [BString toBString](#) (BString n, BInt8 v)
- [BString toBString](#) (BString n, BUInt8 v)
- [BString toBString](#) (BString n, BInt16 v)
- [BString toBString](#) (BString n, BUInt16 v)
- [BString toBString](#) (BString n, BInt32 v)
- [BString toBString](#) (BString n, BUInt32 v)
- [BString toBString](#) (BString n, BInt64 v)
- [BString toBString](#) (BString n, BUInt64 v)
- [BString toBString](#) (BString n, BFloat32 v)
- [BString toBString](#) (BString n, BFloat64 v)
- [BString toBString](#) (BString n, BChar v)
- [BString toBString](#) (BString n, const BChar \*v)
- [BString toBString](#) (BString n, BString v)
- [BString toBString](#) (BString n, BError v)
- [BString toBString](#) (BString n, BTime v)
- [BString toBString](#) (BString name, const [BObjMember](#) \*m, const void \*obj, BStringList ignoreFields)
- [BString toBString](#) (BString n, [BObj](#) &obj)
- [BString toBStringJson](#) (BString n, Bool v)
- [BString toBStringJson](#) (BString n, BInt8 v)
- [BString toBStringJson](#) (BString n, BUInt8 v)
- [BString toBStringJson](#) (BString n, BInt16 v)

- `BString toBStringJson (BString n, BUInt16 v)`
- `BString toBStringJson (BString n, BInt32 v)`
- `BString toBStringJson (BString n, BUInt32 v)`
- `BString toBStringJson (BString n, BInt64 v)`
- `BString toBStringJson (BString n, BUInt64 v)`
- `BString toBStringJson (BString n, BFloat32 v)`
- `BString toBStringJson (BString n, BFloat64 v)`
- `BString toBStringJson (BString n, BChar v)`
- `BString toBStringJson (BString n, const BChar *v)`
- `BString toBStringJson (BString n, BString v)`
- `BString toBStringJson (BString n, BError v)`
- `BString toBStringJson (BString n, BTime v)`
- `BString toBStringJson (BString n, const BObjMember *m, const void *obj, BStringList ignoreFields)`
- `BString toBStringJson (BString n, BObj &obj)`
- `BError toBDictStringFromJson (BString json, BDictString &ds)`

## 8.75.1 Function Documentation

### 8.75.1.1 toBDictStringFromJson()

```
BError toBDictStringFromJson (
    BString json,
    BDictString & ds )
```

### 8.75.1.2 toBString() [1/18]

```
BString toBString (
    BString n,
    Bool v )
```

### 8.75.1.3 toBString() [2/18]

```
BString toBString (
    BString n,
    BInt8 v )
```

### 8.75.1.4 toBString() [3/18]

```
BString toBString (
    BString n,
    BUInt8 v )
```



**8.75.1.5 toBString()** [4/18]

```
BString toBString (  
    BString n,  
    BInt16 v )
```

**8.75.1.6 toBString()** [5/18]

```
BString toBString (  
    BString n,  
    BUInt16 v )
```

**8.75.1.7 toBString()** [6/18]

```
BString toBString (  
    BString n,  
    BInt32 v )
```

**8.75.1.8 toBString()** [7/18]

```
BString toBString (  
    BString n,  
    BUInt32 v )
```

**8.75.1.9 toBString()** [8/18]

```
BString toBString (  
    BString n,  
    BInt64 v )
```

**8.75.1.10 toBString()** [9/18]

```
BString toBString (  
    BString n,  
    BUInt64 v )
```

**8.75.1.11 toBString()** [10/18]

```
BString toBString (  
    BString n,  
    BFloat32 v )
```

**8.75.1.12 toBString()** [11/18]

```
BString toBString (  
    BString n,  
    BFloat64 v )
```

**8.75.1.13 toBString()** [12/18]

```
BString toBString (  
    BString n,  
    BChar v )
```

**8.75.1.14 toBString()** [13/18]

```
BString toBString (  
    BString n,  
    const BChar * v )
```

**8.75.1.15 toBString()** [14/18]

```
BString toBString (  
    BString n,  
    BString v )
```

**8.75.1.16 toBString()** [15/18]

```
BString toBString (  
    BString n,  
    BError v )
```

**8.75.1.17 toBString()** [16/18]

```
BString toBString (
    BString n,
    BTime v )
```

**8.75.1.18 toBString()** [17/18]

```
BString toBString (
    BString name,
    const BObjMember * m,
    const void * obj,
    BStringList ignoreFields )
```

**8.75.1.19 toBString()** [18/18]

```
BString toBString (
    BString n,
    BObj & obj )
```

**8.75.1.20 toBStringJson()** [1/18]

```
BString toBStringJson (
    BString n,
    Bool v )
```

**8.75.1.21 toBStringJson()** [2/18]

```
BString toBStringJson (
    BString n,
    BInt8 v )
```

**8.75.1.22 toBStringJson()** [3/18]

```
BString toBStringJson (
    BString n,
    BUInt8 v )
```

**8.75.1.23 toBStringJson()** [4/18]

```
BString toBStringJson (
    BString n,
    BInt16 v )
```

**8.75.1.24 toBStringJson()** [5/18]

```
BString toBStringJson (
    BString n,
    BUInt16 v )
```

**8.75.1.25 toBStringJson()** [6/18]

```
BString toBStringJson (
    BString n,
    BInt32 v )
```

**8.75.1.26 toBStringJson()** [7/18]

```
BString toBStringJson (
    BString n,
    BUInt32 v )
```

**8.75.1.27 toBStringJson()** [8/18]

```
BString toBStringJson (
    BString n,
    BInt64 v )
```

**8.75.1.28 toBStringJson()** [9/18]

```
BString toBStringJson (
    BString n,
    BUInt64 v )
```

**8.75.1.29 toBStringJson()** [10/18]

```
BString toBStringJson (
    BString n,
    BFloat32 v )
```

**8.75.1.30 toBStringJson()** [11/18]

```
BString toBStringJson (
    BString n,
    BFloat64 v )
```

**8.75.1.31 toBStringJson()** [12/18]

```
BString toBStringJson (
    BString n,
    BChar v )
```

**8.75.1.32 toBStringJson()** [13/18]

```
BString toBStringJson (
    BString n,
    const BChar * v )
```

**8.75.1.33 toBStringJson()** [14/18]

```
BString toBStringJson (
    BString n,
    BString v )
```

**8.75.1.34 toBStringJson()** [15/18]

```
BString toBStringJson (
    BString n,
    BError v )
```

**8.75.1.35 toBStringJson()** [16/18]

```
BString toBStringJson (
    BString n,
    BTime v )
```

**8.75.1.36 toBStringJson()** [17/18]

```
BString toBStringJson (
    BString n,
    const BObjMember * m,
    const void * obj,
    BStringList ignoreFields )
```

**8.75.1.37 toBStringJson()** [18/18]

```
BString toBStringJson (
    BString n,
    BObj & obj )
```

**8.76 BObjStringFormat.h File Reference**

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

**Functions**

- [BString toBString](#) (BString name, Bool value)
- [BString toBString](#) (BString name, BInt8 value)
- [BString toBString](#) (BString name, BUInt8 value)
- [BString toBString](#) (BString name, BInt16 value)
- [BString toBString](#) (BString name, BUInt16 value)
- [BString toBString](#) (BString name, BInt32 value)
- [BString toBString](#) (BString name, BUInt32 value)
- [BString toBString](#) (BString name, BInt64 value)
- [BString toBString](#) (BString name, BUInt64 value)
- [BString toBString](#) (BString name, BFloat32 value)
- [BString toBString](#) (BString name, BFloat64 value)
- [BString toBString](#) (BString name, BChar value)
- [BString toBString](#) (BString name, const BChar \*value)
- [BString toBString](#) (BString name, BString value)
- [BString toBString](#) (BString name, BError value)
- [BString toBString](#) (BString name, BTime time)

- [BString toBString](#) (BString name, const [BObjMember](#) \*members, const void \*obj, BStringList ignore↔ Fields=BStringList())
- [BString toBString](#) (BString name, [BObj](#) &obj)
- [BString toBStringJson](#) (BString name, [Bool](#) value)
- [BString toBStringJson](#) (BString name, [BInt8](#) value)
- [BString toBStringJson](#) (BString name, [BUInt8](#) value)
- [BString toBStringJson](#) (BString name, [BInt16](#) value)
- [BString toBStringJson](#) (BString name, [BUInt16](#) value)
- [BString toBStringJson](#) (BString name, [BInt32](#) value)
- [BString toBStringJson](#) (BString name, [BUInt32](#) value)
- [BString toBStringJson](#) (BString name, [BInt64](#) value)
- [BString toBStringJson](#) (BString name, [BUInt64](#) value)
- [BString toBStringJson](#) (BString name, [BFloat32](#) value)
- [BString toBStringJson](#) (BString name, [BFloat64](#) value)
- [BString toBStringJson](#) (BString name, [BChar](#) value)
- [BString toBStringJson](#) (BString name, const [BChar](#) \*value)
- [BString toBStringJson](#) (BString name, [BString](#) value)
- [BString toBStringJson](#) (BString name, [BError](#) value)
- [BString toBStringJson](#) (BString name, [BTime](#) time)
- [BString toBStringJson](#) (BString name, const [BObjMember](#) \*members, const void \*obj, BStringList ignore↔ Fields=BStringList())
- [BString toBStringJson](#) (BString name, [BObj](#) &obj)
- [BError toBDictStringFromJson](#) (BString json, [BDictString](#) &ds)
- [BString base64\\_encode](#) (void \*data, [BUInt](#) len)
- [BError base64\\_decode](#) (BString strIn, [BString](#) &strOut)

## 8.76.1 Function Documentation

### 8.76.1.1 base64\_decode()

```
BError base64_decode (
    BString strIn,
    BString & strOut )
```

### 8.76.1.2 base64\_encode()

```
BString base64_encode (
    void * data,
    BUInt len )
```

### 8.76.1.3 toBDictStringFromJson()

```
BError toBDictStringFromJson (
    BString json,
    BDictString & ds )
```

**8.76.1.4 toBString()** [1/18]

```
BString toBString (  
    BString name,  
    Bool value )
```

**8.76.1.5 toBString()** [2/18]

```
BString toBString (  
    BString name,  
    BInt8 value )
```

**8.76.1.6 toBString()** [3/18]

```
BString toBString (  
    BString name,  
    BUInt8 value )
```

**8.76.1.7 toBString()** [4/18]

```
BString toBString (  
    BString name,  
    BInt16 value )
```

**8.76.1.8 toBString()** [5/18]

```
BString toBString (  
    BString name,  
    BUInt16 value )
```

**8.76.1.9 toBString()** [6/18]

```
BString toBString (  
    BString name,  
    BInt32 value )
```



**8.76.1.10 toBString()** [7/18]

```
BString toBString (  
    BString name,  
    BUInt32 value )
```

**8.76.1.11 toBString()** [8/18]

```
BString toBString (  
    BString name,  
    BInt64 value )
```

**8.76.1.12 toBString()** [9/18]

```
BString toBString (  
    BString name,  
    BUInt64 value )
```

**8.76.1.13 toBString()** [10/18]

```
BString toBString (  
    BString name,  
    BFloat32 value )
```

**8.76.1.14 toBString()** [11/18]

```
BString toBString (  
    BString name,  
    BFloat64 value )
```

**8.76.1.15 toBString()** [12/18]

```
BString toBString (  
    BString name,  
    BChar value )
```

**8.76.1.16 toBString()** [13/18]

```
BString toBString (
    BString name,
    const BChar * value )
```

**8.76.1.17 toBString()** [14/18]

```
BString toBString (
    BString name,
    BString value )
```

**8.76.1.18 toBString()** [15/18]

```
BString toBString (
    BString name,
    BError value )
```

**8.76.1.19 toBString()** [16/18]

```
BString toBString (
    BString name,
    BTime time )
```

**8.76.1.20 toBString()** [17/18]

```
BString toBString (
    BString name,
    const BObjMember * members,
    const void * obj,
    BStringList ignoreFields = BStringList() )
```

**8.76.1.21 toBString()** [18/18]

```
BString toBString (
    BString name,
    BObj & obj )
```

**8.76.1.22 toBStringJson()** [1/18]

```
BString toBStringJson (  
    BString name,  
    Bool value )
```

**8.76.1.23 toBStringJson()** [2/18]

```
BString toBStringJson (  
    BString name,  
    BInt8 value )
```

**8.76.1.24 toBStringJson()** [3/18]

```
BString toBStringJson (  
    BString name,  
    BUInt8 value )
```

**8.76.1.25 toBStringJson()** [4/18]

```
BString toBStringJson (  
    BString name,  
    BInt16 value )
```

**8.76.1.26 toBStringJson()** [5/18]

```
BString toBStringJson (  
    BString name,  
    BUInt16 value )
```

**8.76.1.27 toBStringJson()** [6/18]

```
BString toBStringJson (  
    BString name,  
    BInt32 value )
```

**8.76.1.28 toBStringJson()** [7/18]

```
BString toBStringJson (  
    BString name,  
    BUInt32 value )
```

**8.76.1.29 toBStringJson()** [8/18]

```
BString toBStringJson (  
    BString name,  
    BInt64 value )
```

**8.76.1.30 toBStringJson()** [9/18]

```
BString toBStringJson (  
    BString name,  
    BUInt64 value )
```

**8.76.1.31 toBStringJson()** [10/18]

```
BString toBStringJson (  
    BString name,  
    BFloat32 value )
```

**8.76.1.32 toBStringJson()** [11/18]

```
BString toBStringJson (  
    BString name,  
    BFloat64 value )
```

**8.76.1.33 toBStringJson()** [12/18]

```
BString toBStringJson (  
    BString name,  
    BChar value )
```

**8.76.1.34 toBStringJson()** [13/18]

```
BString toBStringJson (
    BString name,
    const BChar * value )
```

**8.76.1.35 toBStringJson()** [14/18]

```
BString toBStringJson (
    BString name,
    BString value )
```

**8.76.1.36 toBStringJson()** [15/18]

```
BString toBStringJson (
    BString name,
    BError value )
```

**8.76.1.37 toBStringJson()** [16/18]

```
BString toBStringJson (
    BString name,
    BTime time )
```

**8.76.1.38 toBStringJson()** [17/18]

```
BString toBStringJson (
    BString name,
    const BObjMember * members,
    const void * obj,
    BStringList ignoreFields = BStringList() )
```

**8.76.1.39 toBStringJson()** [18/18]

```
BString toBStringJson (
    BString name,
    BObj & obj )
```

## 8.77 BPoll.cpp File Reference

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

## 8.78 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.*

## 8.79 BQueue.h File Reference

```
#include <BTypes.h>
#include <BError.h>
#include <BList.h>
#include <BMutex.h>
#include <BCondInt.h>
```

### Classes

- class [BQueue< T >](#)

*Queue class.*

### Typedefs

- typedef [BQueue< BInt32 >](#) [BQueueInt](#)

### 8.79.1 Typedef Documentation

#### 8.79.1.1 BQueueInt

```
typedef BQueue<BInt32> BQueueInt
```

## 8.80 BRefData.cpp File Reference

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

### Macros

- `#define` [CHUNK](#) 16

### 8.80.1 Macro Definition Documentation

#### 8.80.1.1 [CHUNK](#)

```
#define CHUNK 16
```

## 8.81 BRefData.h File Reference

```
#include <BAAtomicCount.h>
```

### Classes

- class [BRefData](#)

## 8.82 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>
```

## 8.83 BRtc.h File Reference

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

### Classes

- class [BRtc](#)  
*Realtime clock.*
- class [BRtcThreaded](#)  
*Threaded real time clock.*

## 8.84 BRWLock.cpp File Reference

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

## 8.85 BRWLock.h File Reference

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

### Classes

- class [BRWLock](#)  
*thread read-write locks*

## 8.86 BSema.cpp File Reference

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

## 8.87 BSema.h File Reference

```
#include <sys/types.h>
#include <semaphore.h>
```



## Classes

- class [BSema](#)  
*Sempahore class.*

## 8.88 BSemaphore.cpp File Reference

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

## 8.89 BSemaphore.h File Reference

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

## Classes

- class [BSemaphore](#)  
*Semaphore class.*
- class [BSemaphoreBool](#)
- class [BSemaphoreCount](#)

## 8.90 BSocket.cpp File Reference

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

## Macros

- #define [IP\\_MTU](#) 14

## 8.90.1 Macro Definition Documentation

### 8.90.1.1 IP\_MTU

```
#define IP_MTU 14
```

## 8.91 BSocket.h File Reference

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

### Classes

- class [BSocketAddress](#)  
*Socket Address.*
- class [BSocketAddressINET](#)  
*IP aware socket address.*
- class [BSocket](#)

### Macros

- #define [SOL\\_IP](#) 0
- #define [SO\\_PRIORITY](#) 12
- #define [MSG\\_NOSIGNAL](#) 0

## 8.91.1 Macro Definition Documentation

### 8.91.1.1 MSG\_NOSIGNAL

```
#define MSG_NOSIGNAL 0
```

### 8.91.1.2 SO\_PRIORITY

```
#define SO_PRIORITY 12
```

### 8.91.1.3 SOL\_IP

```
#define SOL_IP 0
```

## 8.92 BSpI.cpp File Reference

```
#include <BSpI.h>  
#include <fcntl.h>  
#include <errno.h>  
#include <sys/ioctl.h>  
#include <linux/spi/spidev.h>
```

## 8.93 BSpI.h File Reference

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

### Classes

- class [BSpI](#)  
*BSpI class.*

## 8.94 BString.cpp File Reference

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

### Macros

- #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)
- int [bstringListinList](#) (BStringList &list, [BString](#) s)
- [BString blistToString](#) (const BStringList &list)
- BStringList [bstringToList](#) ([BString](#) str, int stripSpaces)
- BStringList [charToList](#) (const char \*\*str)
- [BString barrayToString](#) (const BStringArray &list)
- BStringArray [bstringToArray](#) ([BString](#) str, int stripSpaces)
- BStringArray [charToArray](#) (const char \*\*str)
- void [toBString](#) ([BString](#) &v, [BString](#) &s)
- void [toBString](#) (BStringList &v, [BString](#) &s)
- void [toBString](#) (BInt32 &v, [BString](#) &s)
- void [toBString](#) (BUInt32 &v, [BString](#) &s)
- void [toBString](#) (BUInt64 &v, [BString](#) &s)
- void [toBString](#) (BFloat64 &v, [BString](#) &s)
- void [fromBString](#) ([BString](#) &s, [BString](#) &v)
- void [fromBString](#) ([BString](#) &s, BStringList &v)
- void [fromBString](#) ([BString](#) &s, BInt32 &v)
- void [fromBString](#) ([BString](#) &s, BUInt32 &v)
- void [fromBString](#) ([BString](#) &s, BUInt64 &v)
- void [fromBString](#) ([BString](#) &s, BFloat64 &v)
- const char \* [intToString](#) (char \*str, BUInt strLen, int value, int base)
- const char \* [int64ToString](#) (char \*str, BUInt strLen, BInt64 value, int base)
- const char \* [floatToString](#) (char \*str, BUInt strLen, BFloat32 f, BUInt precision)
- char \* [bstrncpy](#) (char \*dest, const char \*src, size\_t n)
- char \* [bstrtrim](#) (char \*str)

## Variables

- static const [BUInt8 base64\\_decode\\_table](#) []

### 8.94.1 Macro Definition Documentation

#### 8.94.1.1 MINUS

```
#define MINUS '-'
```

#### 8.94.1.2 STRIP

```
#define STRIP 0x7f
```

## 8.94.2 Function Documentation

### 8.94.2.1 barrayToString()

```
BString barrayToString (
    const BStringArray & list )
```

### 8.94.2.2 blistToString()

```
BString blistToString (
    const BStringList & list )
```

### 8.94.2.3 bstringListinList()

```
int bstringListinList (
    BStringList & list,
    BString s )
```

### 8.94.2.4 bstringToArray()

```
BStringArray bstringToArray (
    BString str,
    int stripSpaces )
```

### 8.94.2.5 bstringToList()

```
BStringList bstringToList (
    BString str,
    int stripSpaces )
```

### 8.94.2.6 bstrncpy()

```
char* bstrncpy (
    char * dest,
    const char * src,
    size_t n )
```

#### 8.94.2.7 bstrtrim()

```
char* bstrtrim (
    char * str )
```

#### 8.94.2.8 charToArray()

```
BStringArray charToArray (
    const char ** str )
```

#### 8.94.2.9 charToList()

```
BStringList charToList (
    const char ** str )
```

#### 8.94.2.10 floatToString()

```
const char* floatToString (
    char * str,
    BUInt strLen,
    BFloat32 f,
    BUInt precision )
```

#### 8.94.2.11 fromBString() [1/6]

```
void fromBString (
    BString & s,
    BString & v )
```

#### 8.94.2.12 fromBString() [2/6]

```
void fromBString (
    BString & s,
    BStringList & v )
```

**8.94.2.13 fromBString()** [3/6]

```
void fromBString (
    BString & s,
    BInt32 & v )
```

**8.94.2.14 fromBString()** [4/6]

```
void fromBString (
    BString & s,
    BUInt32 & v )
```

**8.94.2.15 fromBString()** [5/6]

```
void fromBString (
    BString & s,
    BUInt64 & v )
```

**8.94.2.16 fromBString()** [6/6]

```
void fromBString (
    BString & s,
    BFloat64 & v )
```

**8.94.2.17 gmatch()**

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

**8.94.2.18 int64ToString()**

```
const char* int64ToString (
    char * str,
    BUInt strLen,
    BInt64 value,
    int base )
```

#### 8.94.2.19 intToString()

```
const char* intToString (
    char * str,
    BUInt strLen,
    int value,
    int base )
```

#### 8.94.2.20 operator<<()

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

#### 8.94.2.21 operator>>()

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

#### 8.94.2.22 toBString() [1/6]

```
void toBString (
    BString & v,
    BString & s )
```

#### 8.94.2.23 toBString() [2/6]

```
void toBString (
    BStringList & v,
    BString & s )
```

#### 8.94.2.24 toBString() [3/6]

```
void toBString (
    BInt32 & v,
    BString & s )
```





## Classes

- class [BString](#)

## Functions

- `std::ostream & operator<<` (`std::ostream &o`, [BString](#) &s)
- `std::istream & operator>>` (`std::istream &i`, [BString](#) &s)
- `void toBString` ([BString](#) &v, [BString](#) &s)
- `void toBString` ([BStringList](#) &v, [BString](#) &s)
- `void toBString` ([BInt32](#) &v, [BString](#) &s)
- `void toBString` ([BUInt32](#) &v, [BString](#) &s)
- `void toBString` ([BUInt64](#) &v, [BString](#) &s)
- `void toBString` ([BFloat64](#) &v, [BString](#) &s)
- `void fromBString` ([BString](#) &s, [BString](#) &v)
- `void fromBString` ([BString](#) &s, [BStringList](#) &v)
- `void fromBString` ([BString](#) &s, [BInt32](#) &v)
- `void fromBString` ([BString](#) &s, [BUInt32](#) &v)
- `void fromBString` ([BString](#) &s, [BUInt64](#) &v)
- `void fromBString` ([BString](#) &s, [BFloat64](#) &v)
- `char from_hex` (`char ch`)
- `char to_hex` (`char code`)
- `char * bstrncpy` (`char *dest`, `const char *src`, `size_t n`)
- `char * bstrtrim` (`char *str`)
- `const char * intToString` (`char *str`, [BUInt](#) strLen, `int value`, `int base=10`)
- `const char * int64ToString` (`char *str`, [BUInt](#) strLen, [BInt64](#) value, `int base=10`)
- `const char * floatToString` (`char *str`, [BUInt](#) strLen, [BFloat32](#) f, [BUInt](#) precision)

## 8.95.1 Function Documentation

### 8.95.1.1 `bstrncpy()`

```
char* bstrncpy (
    char * dest,
    const char * src,
    size_t n )
```

### 8.95.1.2 `bstrtrim()`

```
char* bstrtrim (
    char * str )
```

### 8.95.1.3 floatToString()

```
const char* floatToString (
    char * str,
    BUInt strlen,
    BFloat32 f,
    BUInt precision )
```

### 8.95.1.4 from\_hex()

```
char from_hex (
    char ch ) [inline]
```

### 8.95.1.5 fromBString() [1/6]

```
void fromBString (
    BString & s,
    BString & v )
```

### 8.95.1.6 fromBString() [2/6]

```
void fromBString (
    BString & s,
    BStringList & v )
```

### 8.95.1.7 fromBString() [3/6]

```
void fromBString (
    BString & s,
    BInt32 & v )
```

### 8.95.1.8 fromBString() [4/6]

```
void fromBString (
    BString & s,
    BUInt32 & v )
```

**8.95.1.9 fromBString()** [5/6]

```
void fromBString (
    BString & s,
    BUInt64 & v )
```

**8.95.1.10 fromBString()** [6/6]

```
void fromBString (
    BString & s,
    BFloat64 & v )
```

**8.95.1.11 int64ToString()**

```
const char* int64ToString (
    char * str,
    BUInt strLen,
    BInt64 value,
    int base = 10 )
```

**8.95.1.12 intToString()**

```
const char* intToString (
    char * str,
    BUInt strLen,
    int value,
    int base = 10 )
```

**8.95.1.13 operator<<()**

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

**8.95.1.14 operator>>()**

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

**8.95.1.15 to\_hex()**

```
char to_hex (
    char code ) [inline]
```

**8.95.1.16 toBString()** [1/6]

```
void toBString (
    BString & v,
    BString & s )
```

**8.95.1.17 toBString()** [2/6]

```
void toBString (
    BStringList & v,
    BString & s )
```

**8.95.1.18 toBString()** [3/6]

```
void toBString (
    BInt32 & v,
    BString & s )
```

**8.95.1.19 toBString()** [4/6]

```
void toBString (
    BUInt32 & v,
    BString & s )
```

**8.95.1.20 toBString()** [5/6]

```
void toBString (
    BUInt64 & v,
    BString & s )
```

### 8.95.1.21 toBString() [6/6]

```
void toBString (
    BFloat64 & v,
    BString & s )
```

## 8.96 BStringLocked.h File Reference

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

### Classes

- class [BStringMutex](#)
- class [BStringLocked](#)

## 8.97 BSys.cpp File Reference

```
#include <BSys.h>
#include <time.h>
```

### Functions

- void [delayUs](#) (BUInt us)  
*Will delay for given time in us, if tasks running task will sleep.*
- void [delayMs](#) (BUInt ms)  
*Will delay for given time in ms, if tasks running task will sleep.*

### 8.97.1 Function Documentation

#### 8.97.1.1 delayMs()

```
void delayMs (
    BUInt ms )
```

Will delay for given time in ms, if tasks running task will sleep.

### 8.97.1.2 delayUs()

```
void delayUs (  
    BUInt us )
```

Will delay for given time in us, if tasks running task will sleep.

## 8.98 BSys.h File Reference

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

### Functions

- void [delayUs](#) (BUInt us)  
*Will delay for given time in us, if tasks running task will sleep.*
- void [delayMs](#) (BUInt ms)  
*Will delay for given time in ms, if tasks running task will sleep.*

### 8.98.1 Function Documentation

#### 8.98.1.1 delayMs()

```
void delayMs (  
    BUInt ms )
```

Will delay for given time in ms, if tasks running task will sleep.

#### 8.98.1.2 delayUs()

```
void delayUs (  
    BUInt us )
```

Will delay for given time in us, if tasks running task will sleep.

## 8.99 BTable.cpp File Reference

```
#include <BTable.h>
```

## 8.100 BTable.h File Reference

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

### Classes

- class [BTable](#)

## 8.101 BTask.cpp File Reference

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

## 8.102 BTask.h File Reference

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

### Classes

- class [BTask](#)

## 8.103 BThread.cpp File Reference

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

## 8.104 BThread.h File Reference

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

### Classes

- class [BThread](#)



## 8.105 BTime.cpp File Reference

```
#include <BTime.h>
```

### Functions

- static bool [yearIsLeap](#) ([BUInt16](#) year)
- static [BUInt16](#) [yearDays](#) ([BUInt16](#) year)

### Variables

- static [BUInt16](#) [monDays](#) [2][13]

### 8.105.1 Function Documentation

#### 8.105.1.1 yearDays()

```
static BUInt16 yearDays (  
    BUInt16 year ) [inline], [static]
```

#### 8.105.1.2 yearIsLeap()

```
static bool yearIsLeap (  
    BUInt16 year ) [inline], [static]
```

### 8.105.2 Variable Documentation

#### 8.105.2.1 monDays

```
BUInt16 monDays[2][13] [static]
```

#### Initial value:

```
= {  
    { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334, 365 },  
    { 0, 31, 60, 91, 121, 152, 182, 213, 244, 274, 305, 335, 366 }  
}
```

## 8.106 BTime.h File Reference

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

### Classes

- class [BTime](#)

## 8.107 BTimer.cpp File Reference

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

## 8.108 BTimer.h File Reference

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

### Classes

- class [BTimer](#)  
*Stopwatch style timer.*

## 8.109 BTimeStamp.cpp File Reference

```
#include <BTimeStamp.h>
#include <BTimeStampMs.h>
#include <sys/time.h>
```

### Functions

- void [toBString](#) ([BTimeStamp](#) &v, [BString](#) &s)
- void [fromBString](#) ([BString](#) &s, [BTimeStamp](#) &v)

### Variables

- static int [mon\\_yday](#) [2][13]

## 8.109.1 Function Documentation

### 8.109.1.1 fromBString()

```
void fromBString (
    BString & s,
    BTimeStamp & v )
```

### 8.109.1.2 toBString()

```
void toBString (
    BTimeStamp & v,
    BString & s )
```

## 8.109.2 Variable Documentation

### 8.109.2.1 mon\_yday

```
int mon_yday[2][13] [static]
```

#### Initial value:

```
= {
    { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334, 365 },
    { 0, 31, 60, 91, 121, 152, 182, 213, 244, 274, 305, 335, 366 }
}
```

## 8.110 BTimeStamp.h File Reference

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

### Classes

- class [BTimeStamp](#)

## Functions

- void `toBString` (`BTimeStamp &v`, `BString &s`)
- void `fromBString` (`BString &s`, `BTimeStamp &v`)

### 8.110.1 Function Documentation

#### 8.110.1.1 `fromBString()`

```
void fromBString (  
    BString & s,  
    BTimeStamp & v )
```

#### 8.110.1.2 `toBString()`

```
void toBString (  
    BTimeStamp & v,  
    BString & s )
```

## 8.111 BTimeStampMs.cpp File Reference

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

## Variables

- static int `mon_yday` [2][13]

### 8.111.1 Variable Documentation

#### 8.111.1.1 `mon_yday`

```
int mon_yday[2][13] [static]
```

#### Initial value:

```
= {  
    { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334, 365 },  
    { 0, 31, 60, 91, 121, 152, 182, 213, 244, 274, 305, 335, 366 }  
}
```

## 8.112 BTimeStampMs.h File Reference

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

### Classes

- class [BTimeStampMs](#)

## 8.113 BTimeUs.cpp File Reference

```
#include <BTimeUs.h>
#include <stdio.h>
```

### Functions

- static bool [yearIsLeap](#) ([BUInt16](#) year)
- static [BUInt16](#) [yearDays](#) ([BUInt16](#) year)

### Variables

- static [BUInt16](#) [monDays](#) [2][13]

### 8.113.1 Function Documentation

#### 8.113.1.1 yearDays()

```
static BUInt16 yearDays (
    BUInt16 year ) [inline], [static]
```

#### 8.113.1.2 yearIsLeap()

```
static bool yearIsLeap (
    BUInt16 year ) [inline], [static]
```

### 8.113.2 Variable Documentation

### 8.113.2.1 monDays

```
BUInt16 monDays[2][13] [static]
```

#### Initial value:

```
= {
    { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334, 365 },
    { 0, 31, 60, 91, 121, 152, 182, 213, 244, 274, 305, 335, 366 }
}
```

## 8.114 BTimeUs.h File Reference

```
#include <BTypes.h>
#include <BError.h>
#include <BString.h>
#include <BTime.h>
```

### Classes

- class [BTimeUs](#)

## 8.115 BTypes.h File Reference

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

### Classes

- class [BDataChunk](#)
- struct [BObjMember](#)

### Typedefs

- typedef bool [Bool](#)
- 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 [BFloat32](#)
- typedef double [BFloat64](#)
- typedef char [BChar](#)
- typedef [BInt32](#) [BInt](#)
- typedef [BUInt32](#) [BUInt](#)
- typedef [BFloat32](#) [BFloat](#)
- typedef [BFloat64](#) [BDouble](#)
- typedef size\_t [BSize](#)
- typedef std::vector< [BFloat32](#) > [BArrayFloat](#)
- typedef std::vector< [BFloat64](#) > [BArrayDouble](#)
- typedef [BUInt32](#) [BTimeout](#)

## Enumerations

- enum `BEventType` {  
    `BEventTypeNone`, `BEventTypeError`, `BEventTypeRead`, `BEventTypeWrite`,  
    `BEventTypeDisconnect` }
- enum `BEventWaitSet` {  
    `BEventWaitNone` = 0x00, `BEventWaitError` = 0x01, `BEventWaitRead` = 0x02, `BEventWaitWrite` = 0x04,  
    `BEventWaitDisconnect` = 0x08, `BEventWaitAny` = 0xFFFFFFFF }
- enum `BType` {  
    `BTypeNone`, `BTypeBool`, `BTypeInt8`, `BTypeUInt8`,  
    `BTypeInt16`, `BTypeUInt16`, `BTypeInt32`, `BTypeUInt32`,  
    `BTypeInt64`, `BTypeUInt64`, `BTypeFloat32`, `BTypeFloat64`,  
    `BTypeChar`, `BTypeString`, `BTypeError`, `BTypeTime`,  
    `BTypeTimeUs`, `BTypeObj` = 100 }
- enum `BTypeComp` {  
    `BTypeCompSingle`, `BTypeCompArray`, `BTypeCompArrayFixed`, `BTypeCompList`,  
    `BTypeCompDict` }

## Functions

- `BTimeout timeoutTicks` (`BTimeout timeoutUs`)
- void `byteSwap8` (void \*d, void \*s)
- void `byteSwap16` (void \*d, void \*s)
- void `byteSwap32` (void \*d, void \*s)
- void `byteSwap64` (void \*d, void \*s)

## Variables

- const `BTimeout BTimeoutForever` = 0xFFFFFFFF

### 8.115.1 Typedef Documentation

#### 8.115.1.1 BArrayDouble

```
typedef std::vector<BFloat64> BArrayDouble
```

#### 8.115.1.2 BArrayFloat

```
typedef std::vector<BFloat32> BArrayFloat
```

**8.115.1.3 BChar**

```
typedef char BChar
```

**8.115.1.4 BDouble**

```
typedef BFloat64 BDouble
```

**8.115.1.5 BFloat**

```
typedef BFloat32 BFloat
```

**8.115.1.6 BFloat32**

```
typedef float BFloat32
```

**8.115.1.7 BFloat64**

```
typedef double BFloat64
```

**8.115.1.8 BInt**

```
typedef BInt32 BInt
```

**8.115.1.9 BInt16**

```
typedef int16_t BInt16
```

**8.115.1.10 BInt32**

```
typedef int32_t BInt32
```



**8.115.1.11 BInt64**

```
typedef int64_t BInt64
```

**8.115.1.12 BInt8**

```
typedef int8_t BInt8
```

**8.115.1.13 Bool**

```
typedef bool Bool
```

**8.115.1.14 BSize**

```
typedef size_t BSize
```

**8.115.1.15 BTimeout**

```
typedef BUInt32 BTimeout
```

**8.115.1.16 BUInt**

```
typedef BUInt32 BUInt
```

**8.115.1.17 BUInt16**

```
typedef uint16_t BUInt16
```

**8.115.1.18 BUInt32**

```
typedef uint32_t BUInt32
```

**8.115.1.19 BUInt64**

```
typedef uint64_t BUInt64
```

**8.115.1.20 BUInt8**

```
typedef uint8_t BUInt8
```

**8.115.2 Enumeration Type Documentation****8.115.2.1 BEventType**

```
enum BEventType
```

**Enumerator**

BEventTypeNone	
BEventTypeError	
BEventTypeRead	
BEventTypeWrite	
BEventTypeDisconnect	

**8.115.2.2 BEventWaitSet**

```
enum BEventWaitSet
```

**Enumerator**

BEventWaitNone	
BEventWaitError	
BEventWaitRead	
BEventWaitWrite	
BEventWaitDisconnect	
BEventWaitAny	

**8.115.2.3 BType**

```
enum BType
```

## Enumerator

BTypeNone	
BTypeBool	
BTypeInt8	
BTypeUInt8	
BTypeInt16	
BTypeUInt16	
BTypeInt32	
BTypeUInt32	
BTypeInt64	
BTypeUInt64	
BTypeFloat32	
BTypeFloat64	
BTypeChar	
BTypeString	
BTypeError	
BTypeTime	
BTypeTimeUs	
BTypeObj	

## 8.115.2.4 BTypeComp

```
enum BTypeComp
```

## Enumerator

BTypeCompSingle	
BTypeCompArray	
BTypeCompArrayFixed	
BTypeCompList	
BTypeCompDict	

## 8.115.3 Function Documentation

## 8.115.3.1 byteSwap16()

```
void byteSwap16 (  
    void * d,  
    void * s ) [inline]
```

### 8.115.3.2 byteSwap32()

```
void byteSwap32 (
    void * d,
    void * s ) [inline]
```

### 8.115.3.3 byteSwap64()

```
void byteSwap64 (
    void * d,
    void * s ) [inline]
```

### 8.115.3.4 byteSwap8()

```
void byteSwap8 (
    void * d,
    void * s ) [inline]
```

### 8.115.3.5 timeoutTicks()

```
BTimeout timeoutTicks (
    BTimeout timeoutUs ) [inline]
```

## 8.115.4 Variable Documentation

### 8.115.4.1 BTimeoutForever

```
const BTimeout BTimeoutForever = 0xFFFFFFFF
```

## 8.116 BUrl.cpp File Reference

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

## 8.117 BUrl.h File Reference

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

### Classes

- class [BUrl](#)  
*Basic access to a Url.*

## 8.118 overview.dox File Reference



# Index

- \_\_attribute\_\_
  - BoapMc.h, [386](#)
  - BoapMc1.h, [389](#), [390](#)
- ~BBuffer
  - BBuffer, [27](#)
- ~BBufferStore
  - BBufferStore, [31](#)
- ~BComms
  - BComms, [38](#)
- ~BCond
  - BCond, [42](#)
- ~BCondBool
  - BCondBool, [44](#)
- ~BCondInt
  - BCondInt, [47](#)
- ~BCondResource
  - BCondResource, [50](#)
- ~BCondValue
  - BCondValue, [53](#)
- ~BCondWrap
  - BCondWrap, [58](#)
- ~BDate
  - BDate, [65](#)
- ~BDebugBacktrace
  - BDebugBacktrace, [70](#)
- ~BDir
  - BDir, [82](#)
- ~BDuration
  - BDuration, [86](#)
- ~BEntryFile
  - BEntryFile, [93](#)
- ~BEvent1
  - BEvent1, [108](#)
- ~BEvent1Int
  - BEvent1Int, [111](#)
- ~BEvent1Pipe
  - BEvent1Pipe, [113](#)
- ~BEventPipe
  - BEventPipe, [115](#)
- ~BFifo
  - BFifo, [118](#)
- ~BFifoCirc
  - BFifoCirc, [125](#)
- ~BFile
  - BFile, [134](#)
- ~BList
  - BList, [147](#)
- ~BMutex
  - BMutex, [157](#)
- ~BMutexLock
  - BMutexLock, [159](#)
- ~BMysql
  - BMysql, [160](#)
- ~BObj
  - BObj, [231](#)
- ~BPoll
  - BPoll, [235](#)
- ~BQueue
  - BQueue, [238](#)
- ~BRWLock
  - BRWLock, [248](#)
- ~BRefData
  - BRefData, [241](#)
- ~BRtc
  - BRtc, [244](#)
- ~BRtcThreaded
  - BRtcThreaded, [246](#)
- ~BSema
  - BSema, [251](#)
- ~BSemaphore
  - BSemaphore, [253](#)
- ~BSemaphoreBool
  - BSemaphoreBool, [255](#)
- ~BSemaphoreCount
  - BSemaphoreCount, [258](#)
- ~BSocket
  - BSocket, [262](#)
- ~BSocketAddress
  - BSocketAddress, [268](#)
- ~BString
  - BString, [280](#)
- ~BTable
  - BTable, [298](#)
- ~BTask
  - BTask, [300](#)
- ~BThread
  - BThread, [303](#)
- ~BTimeStamp
  - BTimeStamp, [317](#)
- ~BTimeStampMs
  - BTimeStampMs, [326](#)
- ~BTimer
  - BTimer, [312](#)
- ~BUrl
  - BUrl, [338](#)
- ~BoapClientObject
  - BoapClientObject, [168](#)
- ~BoapMc1Comms

- BoapMc1Comms, 177
- ~BoapMcClientObject
  - BoapMcClientObject, 186
- ~BoapMcComms
  - BoapMcComms, 190
- ~BoapMcServiceObject
  - BoapMcServiceObject, 199
- ~BoapPacket
  - BoapPacket, 204
- ~BoapServer
  - BoapServer, 213
- ~BoapServerConnection
  - BoapServerConnection, 221
- ~BoapServiceObject
  - BoapServiceObject, 225
- APIVERSION\_TEST
  - Boap.cpp, 381
- accept
  - BSocket, 262
- add
  - BAtomic, 23
  - BAtomicCount, 25
  - BSemaphoreCount, 258
  - BString, 280
  - BTimer, 312
- addEntry
  - Boapns::Boapns, 202
- addMicroSeconds
  - BDuration, 86
  - BTimeStamp, 317
  - BTimeUs, 333
- addMilliSeconds
  - BDuration, 86
  - BTimeStamp, 317
  - BTimeStampMs, 326
- addObject
  - BoapServer, 214
- addRef
  - BRefData, 241
- addRow
  - BTable, 298
- addSeconds
  - BDuration, 86
  - BTime, 307
  - BTimeStamp, 317
  - BTimeStampMs, 326
  - BTimeUs, 333
- address
  - BSocketAddressINET, 271
- addressFrom
  - BoapMc.h, 387
  - BoapMc1.h, 390
  - BoapMc1PacketHead, 185
  - BoapMcPacketHead, 197
- addressList
  - Boapns::BoapEntry, 174
- addressTo
  - BoapMc.h, 387
- BoapMc1.h, 390
- BoapMc1PacketHead, 185
- BoapMcPacketHead, 197
- apiVersion
  - Boapns, 17
- append
  - BArray, 21
  - BDict, 73
  - BList, 147
  - BPoll, 235
  - BString, 280
- arg
  - BEvent, 106
- average
  - BTimer, 312
- BArray
  - append, 21
  - BArray, 20
  - del, 21
  - insert, 21
  - number, 21
  - rear, 21
  - sort, 22
  - SortFunc, 20
- BArray< T >, 19
- BArray.h, 341
  - BArrayLoop, 341
- BArrayDouble
  - BTypes.h, 437
- BArrayFloat
  - BTypes.h, 437
- BArrayLoop
  - BArray.h, 341
- BAtomic
  - add, 23
  - BAtomic, 23
  - getValue, 23
  - operator Type, 23
  - operator++, 23
  - operator--, 24
  - ovalue, 24
- BAtomic< Type >, 22
- BAtomic.h, 341
  - BAtomicInt32, 342
  - BAtomicInt64, 342
  - BAtomicUInt32, 342
  - BAtomicUInt64, 342
- BAtomicCount, 24
  - add, 25
  - BAtomicCount, 25
  - getValue, 25
  - operator long, 25
  - operator++, 25
  - operator--, 26
  - ovalue, 26
- BAtomicCount.h, 342
- BAtomicInt32
  - BAtomic.h, 342



- BAtomicInt64
  - BAtomic.h, 342
- BAtomicUInt32
  - BAtomic.h, 342
- BAtomicUInt64
  - BAtomic.h, 342
- BBigEndian
  - BBuffer.h, 344
- BBuffer, 26
  - ~BBuffer, 27
  - BBuffer, 27
  - data, 27
  - odata, 29
  - odataSize, 29
  - osize, 29
  - resize, 28
  - setData, 28
  - setSize, 28
  - size, 28
  - writeData, 28
- BBuffer.cpp, 343
  - roundSize, 343
- BBuffer.h, 343
  - BBigEndian, 344
- BBufferStore, 29
  - ~BBufferStore, 27
  - BBufferStore, 30
  - getHexString, 31
  - getPos, 31
  - opos, 36
  - oswapBytes, 36
  - pop, 31–33
  - push, 33–35
  - setHexString, 35
  - setPos, 36
- BChar
  - BTypes.h, 437
- BComms, 36
  - ~BComms, 38
  - BComms, 38
  - connect, 38
  - disconnect, 38
  - eventQueue, 38
  - Flush, 37
  - flush, 38
  - init, 39
  - isConnected, 39
  - oconnected, 40
  - oevent, 40
  - oeventNum, 41
  - oeventQueue, 41
  - opacketMode, 41
  - otimeout, 41
  - packetMode, 39
  - read, 39
  - readAvailable, 39
  - setPacketMode, 39
  - setTimeout, 39
  - wait, 40
  - write, 40
  - writeAvailable, 40
  - writeChunks, 40
- BComms.cpp, 344
- BComms.h, 344
- BComplex
  - BComplex.h, 344
- BComplex.h, 344
  - BComplex, 344
  - BComplex32, 345
  - BComplex64, 345
- BComplex32
  - BComplex.h, 345
- BComplex64
  - BComplex.h, 345
- BCond, 41
  - ~BCond, 42
  - BCond, 42
  - ocond, 43
  - omutex, 43
  - signal, 42
  - timedWait, 42
  - wait, 42
- BCond.cpp, 345
- BCond.h, 345
- BCondBool, 43
  - ~BCondBool, 44
  - BCondBool, 44
  - clear, 44
  - ocond, 45
  - omutex, 45
  - operator int, 44
  - ovalue, 45
  - set, 44
  - timedWait, 44
  - value, 45
  - wait, 45
- BCondInt, 46
  - ~BCondInt, 47
  - BCondInt, 46
  - decrement, 47
  - increment, 47
  - ocond, 49
  - omutex, 49
  - operator++, 47
  - operator+=", 47
  - operator--, 47
  - operator-=, 48
  - ovalue, 49
  - setValue, 48
  - value, 48
  - waitLessThan, 48
  - waitLessThanOrEqual, 48
  - waitMoreThanOrEqual, 49
- BCondInt.cpp, 345
  - getTimeout, 346
- BCondInt.h, 346

- BCondResource, 50
  - ~BCondResource, 50
  - BCondResource, 50
  - end, 51
  - inUse, 51
  - lock, 51
  - locked, 51
  - ocond, 52
  - olock, 52
  - omutex, 52
  - ouse, 52
  - start, 51
  - unlock, 51
- BCondValue, 52
  - ~BCondValue, 53
  - BCondValue, 53
  - decrement, 54
  - increment, 54
  - ocond, 56
  - omutex, 56
  - operator++, 54
  - operator+=", 54
  - operator--, 54
  - operator-=", 55
  - ovalue, 56
  - setValue, 55
  - value, 55
  - waitLessThan, 55
  - waitLessThanOrEqual, 55
  - waitMoreThanOrEqual, 56
- BCondWrap, 57
  - ~BCondWrap, 58
  - BCondWrap, 57
  - decrement, 58
  - diff, 58
  - increment, 58
  - ocond, 60
  - omutex, 60
  - operator++, 58
  - operator+=", 58
  - operator--, 59
  - operator-=", 59
  - ovalue, 60
  - setValue, 59
  - value, 59
  - waitLessThan, 59
  - waitLessThanOrEqual, 59
  - waitMoreThanOrEqual, 60
- BConfig, 61
  - close, 61
  - fileName, 61
  - findValue, 62
  - ofile, 62
  - ofileName, 62
  - olock, 62
  - open, 62
  - read, 62
  - write, 62
- BConfig.cpp, 346
- BConfig.h, 346
- BCrc16.cpp, 347
  - bcrc16, 347
  - table\_crc\_hi, 347
  - table\_crc\_lo, 347
- BCrc16.h, 348
  - bcrc16, 348
- BCrc32.cpp, 348
  - bcrc32, 349
  - crc32\_tab, 349
- BCrc32.h, 349
  - bcrc32, 349
- BDEBUGL1
  - BoapMc1.cpp, 388
- BDEBUGL2
  - BoapMc1.cpp, 388
- BDataChunk, 63
  - BDataChunk, 63
  - data, 63
  - size, 63
- BDate, 64
  - ~BDate, 65
  - BDate, 65
  - clear, 65
  - compare, 65
  - day, 66
  - daysInMonth, 66
  - getDate, 66
  - getString, 66
  - getStringFormatted, 66
  - isLeap, 66
  - isSet, 67
  - month, 67
  - operator BString, 67
  - operator!=", 67
  - operator<, 67
  - operator<=", 67
  - operator>, 68
  - operator>=", 68
  - operator==, 67
  - oyday, 69
  - oyear, 70
  - set, 68
  - setFirst, 68
  - setLast, 68
  - setNow, 69
  - setString, 69
  - setYDay, 69
  - yday, 69
  - year, 69
- BDate-1.cpp, 350
  - fromBString, 350
  - mon\_yday, 350
  - toBString, 350
- BDate.cpp, 351
  - fromBString, 351
  - mon\_yday, 352

- toBString, 351
- BDate.h, 352
  - fromBString, 352
  - toBString, 352
- BDebug.cpp, 353
  - bdebug, 354
  - bhd32, 353
  - bhd8, 353
  - bhd8a, 354
  - bhda32, 354
  - bhda8, 354
  - getTime, 354
  - setDebug, 354
- BDebug.h, 355
  - BDebug\_STD, 356
  - bdebug, 359
  - bhd32, 357
  - bhd8, 357
  - bhd8a, 357
  - bhda8, 358
  - bhds32, 358
  - dl1printf, 356
  - dl2printf, 356
  - dl3printf, 356
  - dl4printf, 356
  - dprintf, 356
  - eprintf, 356
  - getTime, 358
  - gettid, 358
  - nprintf, 357
  - setDebug, 358
  - tprintf, 358
  - wprintf, 357
- BDebug\_STD
  - BDebug.h, 356
- BDebugBacktrace, 70
  - ~BDebugBacktrace, 70
  - BDebugBacktrace, 70
  - dumpBacktrace, 71
  - dumpBacktraceFile, 71
  - dumpBacktraceStdout, 71
  - dumpBacktraceSyslog, 71
- BDict
  - append, 73
  - BDict, 73
  - clear, 73
  - del, 73, 74
  - find, 74
  - hasKey, 75
  - hashAdd, 74
  - hashDelete, 74
  - hashFind, 74
  - hashPrint, 75
  - insert, 75
  - iterator, 72
  - key, 75
  - ohashLists, 76
  - ohashSize, 76
  - operator+, 75
  - operator=, 75
  - operator[], 76
- BDict< Type >, 71
- BDict.cpp, 359
  - bdictStringToString, 359
  - fromBString, 359
  - toBString, 359
- BDict.h, 360
  - BDictString, 360
  - bdictStringToString, 360
  - fromBString, 360
  - toBString, 361
- BDictItem
  - BDictItem, 77
  - key, 77
  - value, 78
- BDictItem< Type >, 77
- BDictMap
  - clear, 79
  - del, 79
  - hasKey, 79
  - isEnd, 80
  - iterator, 79
  - key, 80
  - next, 80
  - operator[], 80
  - size, 80
  - start, 81
- BDictMap< Value >, 78
- BDictMap.h, 361
  - BDictMapString, 361
- BDictMapString
  - BDictMap.h, 361
- BDictString
  - BDict.h, 360
- BDir, 81
  - ~BDir, 82
  - BDir, 82
  - clear, 82
  - entryName, 82
  - entryStat, 83
  - entryStat64, 83
  - error, 83
  - odirname, 84
  - oerror, 84
  - open, 83
  - osort, 84
  - owild, 84
  - read, 83
  - setSort, 83
  - setWild, 84
- BDir.cpp, 361
  - wild, 362
  - wildString, 362
- BDir.h, 362
- BDouble
  - BTypes.h, 438

- BDuration, 85
  - ~BDuration, 86
  - addMicroSeconds, 86
  - addMilliSeconds, 86
  - addSeconds, 86
  - BDuration, 85, 86
  - clear, 86
  - getMicroSeconds, 87
  - getSeconds, 87
  - getString, 87
  - hour, 87
  - microSecond, 87
  - minute, 87
  - ohour, 88
  - omicroSecond, 88
  - omminute, 88
  - osecond, 88
  - ospare, 89
  - second, 87
  - set, 88
  - setString, 88
- BDuration.cpp, 362
- BDuration.h, 363
- BEndian.cpp, 363
  - bswap\_copy, 363
- BEndian.h, 363
  - be16toh, 365
  - be32toh, 365
  - be64toh, 365
  - betoh, 366, 367
  - bswap\_copy, 368
  - bswap\_p16, 368
  - bswap\_p32, 368
  - bswap\_p64, 368
  - bswap\_p8, 368
  - htobe, 368, 369
  - htobe16, 365
  - htobe32, 365
  - htobe64, 365
  - htole, 370, 371
  - htole16, 365
  - htole32, 366
  - htole64, 366
  - le16toh, 366
  - le32toh, 366
  - le64toh, 366
  - letoh, 371, 372
- BEntry, 89
  - BEntry, 90
  - getName, 90
  - getValue, 90
  - line, 91
  - oname, 92
  - ovalue, 92
  - print, 91
  - setLine, 91
  - setName, 91
  - setValue, 91
- BEntry.cpp, 372
- BEntry.h, 372
- BEntryFile, 92
  - ~BEntryFile, 93
  - BEntryFile, 93
  - clear, 93
  - filename, 93
  - ocomments, 94
  - ofilename, 94
  - open, 94
  - read, 94
  - write, 94
  - writeList, 94
- BEntryList, 95
  - BEntryList, 96
  - clear, 96
  - del, 96
  - deleteEntry, 96
  - find, 97
  - findValue, 97
  - getString, 97
  - insert, 97
  - isSet, 97
  - olastPos, 98
  - operator=, 98
  - print, 98
  - setValue, 98
  - setValueRaw, 98
- BError, 99
  - BError, 100
  - clear, 100
  - copy, 100
  - getErrorNo, 101
  - getNumber, 101
  - getString, 101
  - num, 101
  - oerrNo, 102
  - oerrStr, 102
  - operator int, 101
  - set, 101
  - setError, 102
  - str, 102
- BError.cpp, 373
- BError.h, 373
  - BErrorNum, 373
- BErrorNum
  - BError.h, 373
- BErrorTime, 103
  - BErrorTime, 104
  - clear, 104
  - copy, 104
  - getErrorNo, 104
  - getString, 104
  - getTime, 105
  - oerrNo, 105
  - oerrStr, 105
  - oerrTime, 105
  - operator int, 105

- set, [105](#)
  - Type, [103](#)
- [BErrorTime.cpp](#), [374](#)
- [BErrorTime.h](#), [374](#)
- [BEvent](#), [106](#)
  - arg, [106](#)
  - BEvent, [106](#)
  - oarg, [107](#)
  - otype, [107](#)
  - type, [106](#)
- [BEvent.cpp](#), [374](#)
- [BEvent.h](#), [374](#)
  - [BEventQueue](#), [375](#)
- [BEvent1](#), [107](#)
  - [~BEvent1](#), [108](#)
  - [BEvent1](#), [108](#)
  - [getBinary](#), [108](#)
  - [getType](#), [108](#)
  - otype, [109](#)
  - [setBinary](#), [108](#)
- [BEvent1.cpp](#), [375](#)
- [BEvent1.h](#), [375](#)
  - [BEvent1Type](#), [376](#)
- [BEvent1Error](#), [109](#)
  - [BEvent1Error](#), [109](#)
  - [getBinary](#), [110](#)
  - [setBinary](#), [110](#)
- [BEvent1Int](#), [110](#)
  - [~BEvent1Int](#), [111](#)
  - [BEvent1Int](#), [111](#)
  - clear, [111](#)
  - [getEvent](#), [111](#)
  - [getFd](#), [111](#)
  - ofds, [112](#)
  - [sendEvent](#), [112](#)
- [BEvent1Pipe](#), [112](#)
  - [~BEvent1Pipe](#), [113](#)
  - [BEvent1Pipe](#), [113](#)
  - clear, [113](#)
  - [getEvent](#), [113](#)
  - [getReceiveFd](#), [113](#)
  - ofds, [114](#)
  - [sendEvent](#), [114](#)
- [BEvent1Type](#)
  - [BEvent1.h](#), [376](#)
- [BEventPipe](#), [114](#)
  - [~BEventPipe](#), [115](#)
  - [BEventPipe](#), [115](#)
  - clear, [115](#)
  - [getFd](#), [115](#)
  - ofds, [116](#)
  - read, [115](#)
  - [readAvailable](#), [116](#)
  - write, [116](#)
  - [writeAvailable](#), [116](#)
- [BEventQueue](#)
  - [BEvent.h](#), [375](#)
- [BEventType](#)
  - [BTypes.h](#), [440](#)
- [BEventWaitSet](#)
  - [BTypes.h](#), [440](#)
- [BFifo](#)
  - [~BFifo](#), [118](#)
  - [BFifo](#), [118](#)
  - clear, [118](#)
  - odata, [123](#)
  - operator[], [118](#)
  - [oreadPos](#), [123](#)
  - osize, [123](#)
  - [owritePos](#), [123](#)
  - read, [118](#), [119](#)
  - [readAvailable](#), [119](#)
  - [readAvailableChunk](#), [119](#)
  - [readData](#), [119](#)
  - [readDone](#), [120](#)
  - [readPos](#), [120](#)
  - rebase, [120](#)
  - resize, [120](#)
  - size, [120](#)
  - write, [121](#)
  - [writeAvailable](#), [121](#)
  - [writeAvailableChunk](#), [121](#)
  - [writeBackup](#), [121](#)
  - [writeData](#), [122](#)
  - [writeDone](#), [122](#)
  - [writePos](#), [122](#)
- [BFifo< Type >](#), [117](#)
- [BFifo.h](#), [376](#)
- [BFifo.inc](#), [376](#)
- [BFifoCirc](#)
  - [~BFifoCirc](#), [125](#)
  - [BFifoCirc](#), [125](#)
  - clear, [125](#)
  - [mapCircularBuffer](#), [126](#)
  - odata, [128](#)
  - olock, [128](#)
  - operator[], [126](#)
  - [oreadPos](#), [129](#)
  - osize, [129](#)
  - ovmSize, [129](#)
  - [owriteNumFifoSamples](#), [129](#)
  - [owritePos](#), [129](#)
  - read, [126](#)
  - [readAvailable](#), [126](#)
  - [readData](#), [126](#)
  - [readDone](#), [126](#)
  - [readWaitAvailable](#), [127](#)
  - size, [127](#)
  - [unmapCircularBuffer](#), [127](#)
  - write, [127](#)
  - [writeAvailable](#), [127](#)
  - [writeData](#), [128](#)
  - [writeDone](#), [128](#)
  - [writeWaitAvailable](#), [128](#)
- [BFifoCirc< Type >](#), [123](#)
- [BFifoCirc.cpp](#), [376](#)

- dprintf, 377
- BFifoCirc.h, 377
- BFifoCirc.inc, 377
- BFifoCircPos, 130
  - BFifoCircPos, 130
  - difference, 130
  - increment, 131
  - operator int, 131
  - operator!=, 131
  - operator+=", 131
  - operator==, 131
  - opos, 132
  - osize, 132
  - pos, 131
  - set, 132
  - setSize, 132
- BFile, 133
  - ~BFile, 134
  - BFile, 134
  - close, 135
  - fgets, 135
  - fileName, 135
  - flush, 135
  - getFd, 135
  - isEnd, 135
  - isOpen, 136
  - length, 136
  - ofile, 138
  - ofileName, 138
  - omode, 139
  - open, 136
  - operator=, 136
  - position, 137
  - printf, 137
  - read, 137
  - readString, 137
  - seek, 137
  - setVBuf, 137
  - truncate, 138
  - write, 138
  - writeString, 138
- BFile.cpp, 377
  - STRBUF, 377
- BFile.h, 378
- BFileCsv, 139
  - BFileCsv, 139
  - oseparator, 140
  - readCsv, 140
  - writeCsv, 140
- BFileCsv.cpp, 378
- BFileCsv.h, 378
- BFileData, 140
  - del, 141
  - find, 141
  - getNextId, 141
  - ofilename, 142
  - open, 141
  - read, 141
  - write, 142
- BFileData.cpp, 378
- BFileData.h, 378
- BFloat
  - BTypes.h, 438
- BFloat32
  - BTypes.h, 438
- BFloat64
  - BTypes.h, 438
- BInt
  - BTypes.h, 438
- BInt16
  - BTypes.h, 438
- BInt32
  - BTypes.h, 438
- BInt64
  - BTypes.h, 438
- BInt8
  - BTypes.h, 439
- BIter, 142
  - BIter, 143
  - oi, 143
  - operator BNode \*, 143
  - operator==, 143
  - valid, 143
- BList
  - ~BList, 147
  - append, 147
  - BList, 146, 147
  - begin, 147
  - clear, 148
  - del, 148
  - deleteFirst, 148
  - deleteLast, 148
  - end, 148, 149
  - front, 149
  - get, 149
  - goTo, 149
  - has, 150
  - insert, 150
  - insertAfter, 150
  - isEnd, 150
  - next, 150
  - nodeCreate, 151
  - nodeGet, 151
  - number, 151
  - olength, 155
  - onodes, 155
  - operator+, 151
  - operator=, 152
  - operator[], 152
  - pop, 152
  - position, 153
  - prev, 153
  - push, 153
  - queueAdd, 153
  - queueGet, 153
  - rear, 154

- size, 154
- sort, 154
- SortFunc, 146
- start, 154
- swap, 155
- BList< T >, 144
- BList< T >::Node, 340
- BList.h, 379
  - BListLoop, 379
- BList::Node
  - item, 340
  - Node, 340
- BList\_func.h, 379
- BListLoop
  - BList.h, 379
- BMutex, 155
  - ~BMutex, 157
  - BMutex, 156, 157
  - lock, 157
  - omutex, 158
  - operator=, 157
  - timedLock, 157
  - tryLock, 157
  - Type, 156
  - unlock, 158
- BMutex.cpp, 379
  - MDEBUG, 380
- BMutex.h, 380
- BMutexLock, 158
  - ~BMutexLock, 159
  - BMutexLock, 158
  - lock, 159
  - olock, 159
  - unlock, 159
- BMysql, 159
  - ~BMysql, 160
  - BMysql, 160
  - close, 160
  - db, 161
  - del, 161
  - escapeString, 161
  - flush, 161
  - get, 161
  - insert, 161
  - odb, 162
  - odebug, 163
  - olock, 163
  - opened, 163
  - open, 162
  - query, 162
  - setDebug, 162
  - update, 162
- BMysql.cpp, 380
- BMysql.h, 380
- BNameValue
  - BNameValue, 164
  - getName, 164
  - getValue, 164
  - oname, 164
  - ovalue, 164
- BNameValue< T >, 163
- BNameValue.h, 381
- BNameValueList
  - find, 165
  - findPos, 165
- BNameValueList< T >, 165
- BNode, 166
  - BNode, 166
  - next, 166
  - prev, 166
- BObj, 230
  - ~BObj, 231
  - BObj, 231
  - getDebugString, 231
  - getMember, 231
  - getMembers, 231, 232
  - getType, 232
  - membersPrint, 232
  - setMember, 232
  - setMembers, 232
- BObj.cpp, 397
- BObj.h, 397
- BObjMember, 233
  - dataOffset, 233
  - name, 233
  - size, 233
  - type, 233
  - typeComp, 233
  - typeName, 233
- BObjStringFormat.cpp, 397
  - toBDictStringFromJson, 398
  - toBString, 398–401
  - toBStringJson, 401–404
- BObjStringFormat.h, 404
  - base64\_decode, 405
  - base64\_encode, 405
  - toBDictStringFromJson, 405
  - toBString, 405–408
  - toBStringJson, 408–411
- BPoll, 234
  - ~BPoll, 235
  - append, 235
  - BPoll, 235
  - clear, 235
  - delFd, 235
  - doPoll, 236
  - doPollEvents, 236
  - getPollFds, 236
  - getPollFdsNum, 236
  - nextFd, 236
  - ofds, 237
  - ofdsNext, 237
  - ofdsNum, 237
  - PollFd, 235
- BPoll.cpp, 412
- BPoll.h, 412

- BQueue
  - ~BQueue, 238
  - BQueue, 238
  - clear, 239
  - lock, 240
  - onumber, 240
  - osize, 240
  - read, 239
  - readAvailable, 239
  - write, 239
  - writeAvailable, 239
- BQueue< T >, 237
- BQueue.h, 412
  - BQueueInt, 412
- BQueueInt
  - BQueue.h, 412
- BRWLock, 247
  - ~BRWLock, 248
  - BRWLock, 248
  - lock, 250
  - operator=, 249
  - rdLock, 249
  - tryRdLock, 249
  - tryWrLock, 249
  - unlock, 249
  - wrLock, 249
- BRWLock.cpp, 414
- BRWLock.h, 414
- BRefData, 240
  - ~BRefData, 241
  - addRef, 241
  - BRefData, 241
  - copy, 242
  - data, 242
  - deleteRef, 242
  - len, 242
  - odata, 243
  - olen, 243
  - operator=, 242
  - orefCount, 243
  - setLen, 242
- BRefData.cpp, 413
  - CHUNK, 413
- BRefData.h, 413
- BRtc, 243
  - ~BRtc, 244
  - BRtc, 244
  - init, 244
  - ofd, 245
  - orate, 245
  - wait, 244
- BRtc.cpp, 413
- BRtc.h, 414
- BRtcThreaded, 245
  - ~BRtcThreaded, 246
  - BRtcThreaded, 246
  - function, 246
  - init, 246
  - ocond, 247
  - orate, 247
  - ortc, 247
  - wait, 247
- BSema, 250
  - ~BSema, 251
  - BSema, 250, 251
  - getValue, 251
  - operator=, 251
  - osema, 252
  - post, 251
  - timedWait, 251
  - tryWait, 252
  - wait, 252
- BSema.cpp, 414
- BSema.h, 414
- BSemaphore, 252
  - ~BSemaphore, 253
  - BSemaphore, 253
  - getValue, 253
  - operator=, 254
  - osema, 254
  - set, 254
  - wait, 254
- BSemaphore.cpp, 415
- BSemaphore.h, 415
- BSemaphoreBool, 254
  - ~BSemaphoreBool, 255
  - BSemaphoreBool, 255
  - clear, 255
  - operator int, 256
  - operator=, 256
  - operator==, 256
  - osema, 257
  - ovalue, 257
  - set, 256
  - value, 256
  - wait, 256
- BSemaphoreCount, 257
  - ~BSemaphoreCount, 258
  - add, 258
  - BSemaphoreCount, 258
  - lock, 259
  - operator=, 258
  - osema, 259
  - ovalue, 259
  - setValue, 258
  - take, 258
  - value, 259
  - wait, 259
- BSize
  - BTypes.h, 439
- BSocket, 260
  - ~BSocket, 262
  - accept, 262
  - BSocket, 261, 262
  - bind, 262
  - close, 262



- connect, 263
- getAddress, 263
- getFd, 263
- getMTU, 263
- getSockOpt, 263
- init, 263
- listen, 264
- NType, 261
- osocket, 266
- Priority, 261
- recv, 264
- recvAvailable, 264
- recvFrom, 264
- recvFromWithTimeout, 264
- recvWithTimeout, 264
- send, 265
- sendChunks, 265
- sendTo, 265
- setBroadCast, 265
- setFd, 265
- setPriority, 266
- setReuseAddress, 266
- setSockOpt, 266
- shutdown, 266
- BSocket.cpp, 415
  - IP\_MTU, 416
- BSocket.h, 416
  - MSG\_NOSIGNAL, 416
  - SO\_PRIORITY, 416
  - SOL\_IP, 417
- BSocketAddress, 267
  - ~BSocketAddress, 268
  - BSocketAddress, 268
  - getString, 268
  - len, 268
  - oaddress, 270
  - olen, 270
  - operator const SockAddr \*, 269
  - operator!=, 269
  - operator=, 269
  - operator==, 269
  - raw, 269
  - set, 269
  - SockAddr, 267
- BSocketAddressINET, 270
  - address, 271
  - getHostName, 271
  - getIpAddressList, 272
  - getIpAddressListAll, 272
  - getIpAddresses, 271
  - getString, 272
  - port, 272
  - set, 272, 273
  - setPort, 273
  - SockAddrIP, 271
- BSpi, 273
  - BSpi, 274
  - init, 274
  - Mode, 274
  - odev, 275
  - odevName, 275
  - transact, 274
- BSpi.cpp, 417
- BSpi.h, 417
- BString, 275
  - ~BString, 280
  - add, 280
  - append, 280
  - BString, 278–280
  - base64Decode, 280
  - base64Encode, 280
  - basename, 281
  - clear, 281
  - compare, 281
  - compareRegex, 281
  - compareWild, 281
  - compareWildExpression, 281
  - convert, 282
  - convertHex, 283
  - copy, 283
  - csvDecode, 283
  - csvEncode, 283
  - del, 283
  - dirname, 284
  - extension, 284
  - field, 284
  - fields, 284
  - find, 284
  - findReverse, 285
  - firstLine, 285
  - fixedLen, 285
  - get, 285
  - getTokenList, 285, 286
  - hash, 286
  - inString, 286
  - init, 286
  - insert, 286
  - isSpace, 286
  - justify, 287
  - len, 287
  - lowerFirst, 287
  - operator const char \*, 287
  - operator!=, 287
  - operator<, 289
  - operator<=, 289
  - operator>, 290
  - operator>=, 290
  - operator+, 288
  - operator+=, 288, 289
  - operator=, 289
  - operator==, 289
  - operator[], 290
  - ostr, 294
  - pad, 290
  - printf, 290
  - pullLine, 291

- pullSeparators, 291
- pullToken, 291
- pullWord, 291
- removeNL, 291
- removeSeparators, 291
- retDouble, 292
- retFloat64, 292
- retInt, 292
- retStr, 292
- retStrDup, 292
- retUInt, 292
- reverse, 293
- split, 293
- subString, 293
- toLower, 293
- toUpper, 293
- translateChar, 293
- truncate, 294
- BString.cpp, 417
  - barrayToString, 419
  - base64\_decode\_table, 423
  - blstToString, 419
  - bstringListinList, 419
  - bstringToArray, 419
  - bstringToList, 419
  - bstrncpy, 419
  - bstrtrim, 419
  - charToArray, 420
  - charToList, 420
  - floatToString, 420
  - fromBString, 420, 421
  - gmatch, 421
  - int64ToString, 421
  - intToString, 421
  - MINUS, 418
  - operator<<, 422
  - operator>>, 422
  - STRIP, 418
  - toBString, 422, 423
- BString.h, 423
  - bstrncpy, 424
  - bstrtrim, 424
  - floatToString, 424
  - from\_hex, 425
  - fromBString, 425, 426
  - int64ToString, 426
  - intToString, 426
  - operator<<, 426
  - operator>>, 426
  - to\_hex, 426
  - toBString, 427
- BStringLocked, 294
  - BStringLocked, 295
  - len, 295
  - oLock, 296
  - operator BString, 295
  - operator+, 295
  - operator=, 296
  - ostr, 296
- BStringLocked.h, 428
- BStringMutex, 296
  - BStringMutex, 297
- BSys.cpp, 428
  - delayMs, 428
  - delayUs, 428
- BSys.h, 429
  - delayMs, 429
  - delayUs, 429
- BTable, 297
  - ~BTable, 298
  - addRow, 298
  - BTable, 297
  - calculateWidths, 298
  - clear, 298
  - ocolumnWidths, 299
  - odata, 299
  - otitle, 299
  - print, 298
  - printLine, 298
  - setTitle, 298
- BTable.cpp, 429
- BTable.h, 430
- BTask, 299
  - ~BTask, 300
  - BTask, 300
  - init, 300
  - oname, 301
  - opolicy, 301
  - opriority, 302
  - orunning, 302
  - ostackSize, 302
  - othread, 302
  - run, 300
  - setPriority, 300
  - start, 301
  - stop, 301
  - taskFunc, 301
  - waitForCompletion, 301
- BTask.cpp, 430
- BTask.h, 430
- BThread, 302
  - ~BThread, 303
  - BThread, 303
  - cancel, 303
  - function, 304
  - getThread, 304
  - opolicy, 305
  - opriority, 305
  - oreult, 305
  - orunning, 305
  - ostackSize, 306
  - othread, 306
  - result, 304
  - running, 304
  - setInitPriority, 304
  - setInitStackSize, 304

- setPriority, 304
- start, 305
- startFunc, 305
- waitForCompletion, 305
- BThread.cpp, 430
- BThread.h, 430
- BTime, 306
  - addSeconds, 307
  - BTime, 307
  - getDate, 307
  - getSeconds, 307
  - getString, 308
  - getTime, 308
  - isLeapYear, 308
  - isSet, 308
  - operator!=, 308
  - operator<, 309
  - operator<=, 309
  - operator>, 309
  - operator>=, 309
  - operator+, 308
  - operator+=, 309
  - operator==, 309
  - otime, 310
  - set, 309, 310
  - setString, 310
  - setYearDay, 310
- BTime.cpp, 431
  - monDays, 431
  - yearDays, 431
  - yearIsLeap, 431
- BTime.h, 432
- BTimeStamp, 314
  - ~BTimeStamp, 317
  - addMicroSeconds, 317
  - addMilliSeconds, 317
  - addSeconds, 317
  - BTimeStamp, 316
  - clear, 317
  - compare, 317
  - day, 318
  - difference, 318
  - getDate, 318
  - getString, 318
  - getStringFormatted, 318
  - getStringNoMs, 318
  - getYearMicroSeconds, 319
  - getYearSeconds, 319
  - hour, 319
  - isLeap, 319
  - isSet, 319
  - microSecond, 319
  - minute, 319
  - month, 320
  - ohour, 323
  - omicroSecond, 323
  - omminute, 323
  - operator BString, 320
  - operator!=, 320
  - operator<, 320
  - operator<=, 320
  - operator>, 321
  - operator>=, 321
  - operator=, 320
  - operator==, 320
  - osecond, 323
  - ospare, 324
  - oyday, 324
  - oyear, 324
  - second, 321
  - set, 321
  - setFirst, 322
  - setLast, 322
  - setNow, 322
  - setString, 322
  - setTime, 322
  - setYDay, 322
  - yday, 323
  - year, 323
- BTimeStamp.cpp, 432
  - fromBString, 433
  - mon\_yday, 433
  - toBString, 433
- BTimeStamp.h, 433
  - fromBString, 434
  - toBString, 434
- BTimeStampMs, 324
  - ~BTimeStampMs, 326
  - addMilliSeconds, 326
  - addSeconds, 326
  - BTimeStampMs, 326
  - clear, 327
  - compare, 327
  - difference, 327
  - getDate, 327
  - getDurationString, 327
  - getDurationStringNoMs, 327
  - getString, 328
  - getStringNoMs, 328
  - getStringRaw, 328
  - getYearMilliSeconds, 328
  - getYearSeconds, 328
  - hour, 330
  - isLeap, 328
  - milliSecond, 330
  - minute, 331
  - operator<, 329
  - operator<=, 329
  - operator>, 329
  - operator>=, 329
  - sampleNumber, 331
  - second, 331
  - setDurationString, 329
  - setNow, 329
  - setString, 330
  - subMilliSeconds, 330

- subSeconds, 330
- yday, 331
- year, 331
- BTimeStampMs.cpp, 434
  - mon\_yday, 434
- BTimeStampMs.h, 435
- BTimeUs, 332
  - addMicroSeconds, 333
  - addSeconds, 333
  - BTimeUs, 333
  - getDate, 333
  - getMicroSeconds, 334
  - getSeconds, 334
  - getString, 334
  - getStringUs, 334
  - getTime, 334
  - isLeapYear, 335
  - isSet, 335
  - operator BTime, 335
  - operator!=, 335
  - operator<, 336
  - operator<=, 336
  - operator>, 336
  - operator>=, 336
  - operator+, 335
  - operator+=, 335
  - operator==, 336
  - otime, 337
  - set, 336
  - setString, 337
  - setYearDay, 337
- BTimeUs.cpp, 435
  - monDays, 435
  - yearDays, 435
  - yearIsLeap, 435
- BTimeUs.h, 436
- BTimeout
  - BTypes.h, 439
- BTimeoutForever
  - BTypes.h, 442
- BTimer, 311
  - ~BTimer, 312
  - add, 312
  - average, 312
  - BTimer, 312
  - clear, 312
  - getElapsedTime, 312
  - getTime, 313
  - oaverage, 313
  - oendTime, 313
  - oclock, 314
  - onum, 314
  - opeak, 314
  - ostartTime, 314
  - peak, 313
  - start, 313
  - stop, 313
- BTimer.cpp, 432
- BTimer.h, 432
- BType
  - BTypes.h, 440
- BTypeComp
  - BTypes.h, 441
- BTypes.h, 436
  - BArrayDouble, 437
  - BArrayFloat, 437
  - BChar, 437
  - BDouble, 438
  - BEventType, 440
  - BEventWaitSet, 440
  - BFloat, 438
  - BFloat32, 438
  - BFloat64, 438
  - BInt, 438
  - BInt16, 438
  - BInt32, 438
  - BInt64, 438
  - BInt8, 439
  - BSize, 439
  - BTimeout, 439
  - BTimeoutForever, 442
  - BType, 440
  - BTypeComp, 441
  - BUInt, 439
  - BUInt16, 439
  - BUInt32, 439
  - BUInt64, 439
  - BUInt8, 440
  - Bool, 439
  - byteSwap16, 441
  - byteSwap32, 441
  - byteSwap64, 442
  - byteSwap8, 442
  - timeoutTicks, 442
- BUInt
  - BTypes.h, 439
- BUInt16
  - BTypes.h, 439
- BUInt32
  - BTypes.h, 439
- BUInt64
  - BTypes.h, 439
- BUInt8
  - BTypes.h, 440
- BUrl, 338
  - ~BUrl, 338
  - BUrl, 338
  - oinit, 339
  - ores, 339
  - readString, 339
  - writeData, 339
- BUrl.cpp, 442
- BUrl.h, 443
- barrayToString
  - BString.cpp, 419
- base64\_decode

- BObjStringFormat.h, 405
- base64\_decode\_table
  - BString.cpp, 423
- base64\_encode
  - BObjStringFormat.h, 405
- base64Decode
  - BString, 280
- base64Encode
  - BString, 280
- basename
  - BString, 281
- bcrc16
  - BCrc16.cpp, 347
  - BCrc16.h, 348
- bcrc32
  - BCrc32.cpp, 349
  - BCrc32.h, 349
- bdebug
  - BDebug.cpp, 354
  - BDebug.h, 359
- bdictStringToString
  - BDict.cpp, 359
  - BDict.h, 360
- be16toh
  - BEndian.h, 365
- be32toh
  - BEndian.h, 365
- be64toh
  - BEndian.h, 365
- begin
  - BList, 147
- betoh
  - BEndian.h, 366, 367
- bhd32
  - BDebug.cpp, 353
  - BDebug.h, 357
- bhd8
  - BDebug.cpp, 353
  - BDebug.h, 357
- bhd8a
  - BDebug.cpp, 354
  - BDebug.h, 357
- bhda32
  - BDebug.cpp, 354
- bhda8
  - BDebug.cpp, 354
  - BDebug.h, 358
- bhds32
  - BDebug.h, 358
- bind
  - BSocket, 262
- blistToString
  - BString.cpp, 419
- Boap.cpp, 381
  - APIVERSION\_TEST, 381
  - boapPort, 382
  - DEBUG, 381
  - dprintf, 382
  - IS\_BIG\_ENDIAN, 382
- Boap.h, 382
  - BoapFunc, 383
  - BoapMagic, 384
  - BoapPriority, 383
  - BoapService, 383
  - BoapType, 384
- BoapClientObject, 167
  - ~BoapClientObject, 168
  - BoapClientObject, 168
  - checkApiVersion, 168
  - connectService, 168, 169
  - disconnectService, 169
  - getServiceName, 169
  - handleReconnect, 169
  - oapiVersion, 171
  - oconnected, 171
  - oLock, 171
  - omaxLength, 172
  - oname, 172
  - opriority, 172
  - oreconnect, 172
  - orx, 172
  - oservice, 172
  - otimeout, 172
  - otx, 173
  - performCall, 169
  - performRecv, 170
  - performSend, 170
  - ping, 170
  - pingLocked, 170
  - setConnectionPriority, 171
  - setMaxLength, 171
  - setTimeout, 171
- BoapEntry
  - Boapns::BoapEntry, 173
- BoapFunc
  - Boap.h, 383
  - BoapSimple.h, 395
- BoapFuncEntry, 174
  - BoapFuncEntry, 175
  - ocmd, 175
  - ofunc, 175
- BoapMagic
  - Boap.h, 384
- BoapMc.cpp, 384
  - DEBUG\_LOCAL1, 385
  - DEBUG\_LOCAL, 385
  - dl1printf, 385
  - dlprintf, 385
- BoapMc.h, 385
  - \_\_attribute\_\_, 386
  - addressFrom, 387
  - addressTo, 387
  - BoapMcType, 386
  - checksum, 387
  - cmd, 387
  - error, 387

- length, 387
- BoapMc1.cpp, 388
  - BDEBUGL1, 388
  - BDEBUGL2, 388
- BoapMc1.h, 388
  - \_\_attribute\_\_, 389, 390
  - addressFrom, 390
  - addressTo, 390
  - boapMc1CommsRoundupLen, 390
  - BoapMc1Magic, 390
  - BoapMc1Type, 389
  - checksum, 390
  - cmd, 391
  - data, 391
  - error, 391
  - head, 391
  - length, 391
  - magic, 391
  - number, 391
  - string, 392
- BoapMc1Comms, 176
  - ~BoapMc1Comms, 177
  - BoapMc1Comms, 177
  - getApiVersion, 177
  - oaddressFrom, 180
  - oaddressTo, 180
  - oapiVersion, 180
  - ocomms, 180
  - oerror, 180
  - ohalfDuplex, 180
  - olockCall, 181
  - olockTx, 181
  - opacketRpcCmd, 181
  - opacketRpcDoneSema, 181
  - opacketRpcSema, 181
  - opacketRx, 181
  - opacketRxBASE, 182
  - opacketTx, 182
  - opacketTxBase, 182
  - oreqSize, 182
  - othreaded, 182
  - otimeout, 182
  - packetRx, 177
  - packetRxData, 178
  - packetRxEnd, 178
  - packetTx, 178
  - processRequest, 178
  - processRequests, 178
  - processRx, 178
  - setAddress, 179
  - setComms, 179
  - setCommsMode, 179
  - setTimeout, 179
- boapMc1CommsRoundupLen
  - BoapMc1.h, 390
- BoapMc1Error, 183
  - number, 183
  - string, 183
- BoapMc1Magic
  - BoapMc1.h, 390
- BoapMc1Packet, 184
  - data, 184
  - head, 184
- BoapMc1PacketHead, 184
  - addressFrom, 185
  - addressTo, 185
  - checksum, 185
  - cmd, 185
  - error, 185
  - length, 185
  - magic, 185
- BoapMc1Type
  - BoapMc1.h, 389
- BoapMcClientObject, 186
  - ~BoapMcClientObject, 186
  - BoapMcClientObject, 186
  - getApiVersion, 187
  - oaddressFrom, 188
  - oaddressTo, 188
  - oapiVersion, 188
  - ocomms, 188
  - opacket, 188
  - performCall, 187
  - performRecv, 187
  - performSend, 187
  - setAddress, 187
- BoapMcComms, 188
  - ~BoapMcComms, 190
  - BoapMcComms, 190
  - getApiVersion, 190
  - oaddressFrom, 193
  - oaddressTo, 193
  - oapiVersion, 193
  - ocomms, 193
  - olockCall, 193
  - olockTx, 194
  - opacket, 194
  - opacketReqQueue, 194
  - opacketReqRx, 194
  - opacketReqTx, 194
  - opacketRx, 194
  - opacketRxSema, 195
  - opacketTx, 195
  - opacketTxQueue, 195
  - opacketTxQueueWriteNum, 195
  - opacketTxSema, 195
  - oslave, 195
  - othreaded, 196
  - otimeout, 196
  - packetRecv, 190
  - packetSend, 191
  - performCall, 191
  - performSend, 191
  - processPacket, 191
  - processRequest, 191
  - processRequests, 191

- processRx, 192
- setAddress, 192
- setComms, 192
- setCommsMode, 192
- setTimeout, 193
- BoapMcPacket, 196
  - data, 196
  - head, 196
- BoapMcPacketHead, 197
  - addressFrom, 197
  - addressTo, 197
  - checksum, 197
  - cmd, 197
  - error, 198
  - length, 198
- BoapMcServiceObject, 198
  - ~BoapMcServiceObject, 199
  - BoapMcServiceObject, 198
  - oapiVersion, 199
  - process, 199
  - processEvent, 199
  - sendEvent, 199
- BoapMcSignalObject, 200
  - BoapMcSignalObject, 200
  - ocomms, 200
  - performSend, 200
- BoapMcType
  - BoapMc.h, 386
- BoapPacket, 203
  - ~BoapPacket, 204
  - BoapPacket, 204
  - data, 204
  - getCmd, 205
  - nbytes, 205
  - odata, 209
  - onbytes, 209
  - opos, 209
  - osize, 209
  - peekHead, 205
  - pop, 205, 206
  - popHead, 206, 207
  - push, 207, 208
  - pushHead, 208
  - resize, 208
  - setData, 209
  - updateHead, 209
  - updateLen, 209
- BoapPacketHead, 210
  - cmd, 210
  - length, 210, 211
  - reserved, 211
  - service, 211
  - type, 211
- boapPort
  - Boap.cpp, 382
- BoapPriority
  - Boap.h, 383
- BoapServer, 212
  - ~BoapServer, 213
  - addObject, 214
  - BoapServer, 213, 214
  - clientGone, 214
  - closeConnections, 214
  - function, 214
  - getConnectionsNumber, 214
  - getEventSocket, 215
  - getHostName, 215
  - getSocket, 215
  - init, 215
  - newConnection, 216
  - oboapNs, 217
  - oboapns, 218
  - oclientGoneEvent, 218
  - oclients, 218
  - ohostName, 218
  - oisBoapns, 218
  - oclock, 218
  - onet, 218
  - onetEvent, 218
  - onetEventAddress, 219
  - onumOperations, 219
  - opoll, 219
  - orx, 219
  - oservices, 219
  - othreaded, 219
  - otx, 219
  - process, 216
  - processEvent, 216, 217
  - run, 217
  - sendEvent, 217
- BoapServerConnection, 220
  - ~BoapServerConnection, 221
  - BoapServerConnection, 221
  - function, 221
  - getHead, 221
  - getSocket, 221
  - init, 221
  - oboapServer, 222
  - omaxLength, 222
  - orx, 222
  - osocket, 222
  - otx, 222
  - process, 221
  - setMaxLength, 222
  - validate, 222
- BoapService
  - Boap.h, 383
  - BoapSimple.h, 395
- BoapServiceEntry, 223
  - BoapServiceEntry, 223
  - object, 223
  - oservice, 224
- BoapServiceObject, 224
  - ~BoapServiceObject, 225
  - BoapServiceObject, 225
  - doConnectionPriority, 225

- doPing, 225
- name, 226
- oapiVersion, 228
- ofuncList, 228
- oname, 228
- oserver, 228
- process, 226
- processEvent, 226, 227
- sendEvent, 227
- setName, 228
- BoapSignalObject, 229
  - BoapSignalObject, 229
  - orx, 230
  - otx, 230
  - performSend, 230
- BoapSimple.cc, 393
  - DEBUG, 394
  - dprintf, 394
  - roundSize, 394
- BoapSimple.h, 394
  - BoapFunc, 395
  - BoapService, 395
  - BoapType, 396
  - Double, 395
  - Int16, 395
  - Int32, 395
  - Int8, 396
  - UInt16, 396
  - UInt32, 396
  - UInt8, 396
- BoapType
  - Boap.h, 384
  - BoapSimple.h, 396
- Boapns, 17
  - apiVersion, 17
  - Boapns::Boapns, 201
- Boapns::BoapEntry, 173
  - addressList, 174
  - BoapEntry, 173
  - hostName, 174
  - name, 174
  - port, 174
  - service, 174
- Boapns::Boapns, 201
  - addEntry, 202
  - Boapns, 201
  - delEntry, 202
  - getEntry, 202
  - getEntryList, 202
  - getNewName, 202
  - getVersion, 202
- BoapnsC.cpp, 392
- BoapnsC.h, 392
- BoapnsD.cpp, 393
- BoapnsD.h, 393
- Bool
  - BTypes.h, 439
- bstringListinList
  - BString.cpp, 419
- bstringToArray
  - BString.cpp, 419
- bstringToList
  - BString.cpp, 419
- bstrncpy
  - BString.cpp, 419
  - BString.h, 424
- bstrtrim
  - BString.cpp, 419
  - BString.h, 424
- bswap\_copy
  - BEndian.cpp, 363
  - BEndian.h, 368
- bswap\_p16
  - BEndian.h, 368
- bswap\_p32
  - BEndian.h, 368
- bswap\_p64
  - BEndian.h, 368
- bswap\_p8
  - BEndian.h, 368
- byteSwap16
  - BTypes.h, 441
- byteSwap32
  - BTypes.h, 441
- byteSwap64
  - BTypes.h, 442
- byteSwap8
  - BTypes.h, 442
- CHUNK
  - BRefData.cpp, 413
- calculateWidths
  - BTable, 298
- cancel
  - BThread, 303
- charToArray
  - BString.cpp, 420
- charToList
  - BString.cpp, 420
- checkApiVersion
  - BoapClientObject, 168
- checksum
  - BoapMc.h, 387
  - BoapMc1.h, 390
  - BoapMc1PacketHead, 185
  - BoapMcPacketHead, 197
- clear
  - BCondBool, 44
  - BDate, 65
  - BDict, 73
  - BDictMap, 79
  - BDir, 82
  - BDuration, 86
  - BEntryFile, 93
  - BEntryList, 96
  - BError, 100
  - BErrorTime, 104



- BEvent1Int, 111
- BEvent1Pipe, 113
- BEventPipe, 115
- BFifo, 118
- BFifoCirc, 125
- BList, 148
- BPoll, 235
- BQueue, 239
- BSemaphoreBool, 255
- BString, 281
- BTable, 298
- BTimeStamp, 317
- BTimeStampMs, 327
- BTimer, 312
- clientGone
  - BoapServer, 214
- close
  - BConfig, 61
  - BFile, 135
  - BMySQL, 160
  - BSocket, 262
- closeConnections
  - BoapServer, 214
- cmd
  - BoapMc.h, 387
  - BoapMc1.h, 391
  - BoapMc1PacketHead, 185
  - BoapMcPacketHead, 197
  - BoapPacketHead, 210
- compare
  - BDate, 65
  - BString, 281
  - BTimeStamp, 317
  - BTimeStampMs, 327
- compareRegex
  - BString, 281
- compareWild
  - BString, 281
- compareWildExpression
  - BString, 281
- connect
  - BComms, 38
  - BSocket, 263
- connectService
  - BoapClientObject, 168, 169
- convert
  - BString, 282
- convertHex
  - BString, 283
- copy
  - BError, 100
  - BErrorTime, 104
  - BRefData, 242
  - BString, 283
- crc32\_tab
  - BCrc32.cpp, 349
- csvDecode
  - BString, 283
- csvEncode
  - BString, 283
- DEBUG\_LOCAL1
  - BoapMc.cpp, 385
- DEBUG\_LOCAL
  - BoapMc.cpp, 385
- DEBUG
  - Boap.cpp, 381
  - BoapSimple.cc, 394
- data
  - BBuffer, 27
  - BDataChunk, 63
  - BRefData, 242
  - BoapMc1.h, 391
  - BoapMc1Packet, 184
  - BoapMcPacket, 196
  - BoapPacket, 204
- dataOffset
  - BObjMember, 233
- day
  - BDate, 66
  - BTimeStamp, 318
- daysInMonth
  - BDate, 66
- db
  - BMySQL, 161
- decrement
  - BCondInt, 47
  - BCondValue, 54
  - BCondWrap, 58
- del
  - BArray, 21
  - BDict, 73, 74
  - BDictMap, 79
  - BEntryList, 96
  - BFileData, 141
  - BList, 148
  - BMySQL, 161
  - BString, 283
- delEntry
  - Boapns::Boapns, 202
- delFd
  - BPoll, 235
- delayMs
  - BSys.cpp, 428
  - BSys.h, 429
- delayUs
  - BSys.cpp, 428
  - BSys.h, 429
- deleteEntry
  - BEntryList, 96
- deleteFirst
  - BList, 148
- deleteLast
  - BList, 148
- deleteRef
  - BRefData, 242
- diff

- BCondWrap, 58
- difference
  - BFifoCircPos, 130
  - BTimeStamp, 318
  - BTimeStampMs, 327
- dirname
  - BString, 284
- disconnect
  - BComms, 38
- disconnectService
  - BoapClientObject, 169
- dl1printf
  - BDebug.h, 356
  - BoapMc.cpp, 385
- dl2printf
  - BDebug.h, 356
- dl3printf
  - BDebug.h, 356
- dl4printf
  - BDebug.h, 356
- dlprintf
  - BoapMc.cpp, 385
- doConnectionPriority
  - BoapServiceObject, 225
- doPing
  - BoapServiceObject, 225
- doPoll
  - BPoll, 236
- doPollEvents
  - BPoll, 236
- Double
  - BoapSimple.h, 395
- dprintf
  - BDebug.h, 356
  - BFifoCirc.cpp, 377
  - Boap.cpp, 382
  - BoapSimple.cc, 394
- dumpBacktrace
  - BDebugBacktrace, 71
- dumpBacktraceFile
  - BDebugBacktrace, 71
- dumpBacktraceStdout
  - BDebugBacktrace, 71
- dumpBacktraceSyslog
  - BDebugBacktrace, 71
- end
  - BCondResource, 51
  - BList, 148, 149
- entryName
  - BDir, 82
- entryStat
  - BDir, 83
- entryStat64
  - BDir, 83
- eprintf
  - BDebug.h, 356
- error
  - BDir, 83
- BoapMc.h, 387
- BoapMc1.h, 391
- BoapMc1PacketHead, 185
- BoapMcPacketHead, 198
- escapeString
  - BMySQL, 161
- eventQueue
  - BComms, 38
- extension
  - BString, 284
- fgets
  - BFile, 135
- field
  - BString, 284
- fields
  - BString, 284
- fileName
  - BConfig, 61
  - BFile, 135
- filename
  - BEntryFile, 93
- find
  - BDict, 74
  - BEntryList, 97
  - BFileData, 141
  - BNameValueList, 165
  - BString, 284
- findPos
  - BNameValueList, 165
- findReverse
  - BString, 285
- findValue
  - BConfig, 62
  - BEntryList, 97
- firstLine
  - BString, 285
- fixedLen
  - BString, 285
- floatToString
  - BString.cpp, 420
  - BString.h, 424
- Flush
  - BComms, 37
- flush
  - BComms, 38
  - BFile, 135
  - BMySQL, 161
- from\_hex
  - BString.h, 425
- fromBString
  - BDate-1.cpp, 350
  - BDate.cpp, 351
  - BDate.h, 352
  - BDict.cpp, 359
  - BDict.h, 360
  - BString.cpp, 420, 421
  - BString.h, 425, 426
  - BTimeStamp.cpp, 433

- BTimeStamp.h, [434](#)
- front
  - BList, [149](#)
- function
  - BRtcThreaded, [246](#)
  - BThread, [304](#)
  - BoapServer, [214](#)
  - BoapServerConnection, [221](#)
- get
  - BList, [149](#)
  - BMySQL, [161](#)
  - BString, [285](#)
- getAddress
  - BSocket, [263](#)
- getApiVersion
  - BoapMc1Comms, [177](#)
  - BoapMcClientObject, [187](#)
  - BoapMcComms, [190](#)
- getBinary
  - BEvent1, [108](#)
  - BEvent1Error, [110](#)
- getCmd
  - BoapPacket, [205](#)
- getConnectionsNumber
  - BoapServer, [214](#)
- getDate
  - BDate, [66](#)
  - BTime, [307](#)
  - BTimeStamp, [318](#)
  - BTimeStampMs, [327](#)
  - BTimeUs, [333](#)
- getDebugString
  - BObj, [231](#)
- getDurationString
  - BTimeStampMs, [327](#)
- getDurationStringNoMs
  - BTimeStampMs, [327](#)
- getElapsedTime
  - BTimer, [312](#)
- getEntry
  - Boapns::Boapns, [202](#)
- getEntryList
  - Boapns::Boapns, [202](#)
- getErrorNo
  - BError, [101](#)
  - BErrorTime, [104](#)
- getEvent
  - BEvent1Int, [111](#)
  - BEvent1Pipe, [113](#)
- getEventSocket
  - BoapServer, [215](#)
- getFd
  - BEvent1Int, [111](#)
  - BEventPipe, [115](#)
  - BFile, [135](#)
  - BSocket, [263](#)
- getHead
  - BoapServerConnection, [221](#)
- getHexString
  - BBufferStore, [31](#)
- getHostName
  - BSocketAddressINET, [271](#)
  - BoapServer, [215](#)
- getIpAddressList
  - BSocketAddressINET, [272](#)
- getIpAddressListAll
  - BSocketAddressINET, [272](#)
- getIpAddresses
  - BSocketAddressINET, [271](#)
- getMTU
  - BSocket, [263](#)
- getMember
  - BObj, [231](#)
- getMembers
  - BObj, [231](#), [232](#)
- getMicroSeconds
  - BDuration, [87](#)
  - BTimeUs, [334](#)
- getName
  - BEntry, [90](#)
  - BNameValue, [164](#)
- getNewName
  - Boapns::Boapns, [202](#)
- getNextId
  - BFileData, [141](#)
- getNumber
  - BError, [101](#)
- getPollFds
  - BPoll, [236](#)
- getPollFdsNum
  - BPoll, [236](#)
- getPos
  - BBufferStore, [31](#)
- getReceiveFd
  - BEvent1Pipe, [113](#)
- getSeconds
  - BDuration, [87](#)
  - BTime, [307](#)
  - BTimeUs, [334](#)
- getServiceName
  - BoapClientObject, [169](#)
- getSockOpt
  - BSocket, [263](#)
- getSocket
  - BoapServer, [215](#)
  - BoapServerConnection, [221](#)
- getString
  - BDate, [66](#)
  - BDuration, [87](#)
  - BEntryList, [97](#)
  - BError, [101](#)
  - BErrorTime, [104](#)
  - BSocketAddress, [268](#)
  - BSocketAddressINET, [272](#)
  - BTime, [308](#)
  - BTimeStamp, [318](#)

- BTimeStampMs, 328
- BTimeUs, 334
- getStringFormatted
  - BDate, 66
  - BTimeStamp, 318
- getStringNoMs
  - BTimeStamp, 318
  - BTimeStampMs, 328
- getStringRaw
  - BTimeStampMs, 328
- getStringUs
  - BTimeUs, 334
- getThread
  - BThread, 304
- getTime
  - BDebug.cpp, 354
  - BDebug.h, 358
  - BErrorTime, 105
  - BTime, 308
  - BTimeUs, 334
  - BTimer, 313
- getTimeout
  - BCondInt.cpp, 346
- getTokenList
  - BString, 285, 286
- getType
  - BEvent1, 108
  - BObj, 232
- getValue
  - BAtomic, 23
  - BAtomicCount, 25
  - BEntry, 90
  - BNameValue, 164
  - BSema, 251
  - BSemaphore, 253
- getVersion
  - Boapns::Boapns, 202
- getYearMicroSeconds
  - BTimeStamp, 319
- getYearMilliSeconds
  - BTimeStampMs, 328
- getYearSeconds
  - BTimeStamp, 319
  - BTimeStampMs, 328
- gettid
  - BDebug.h, 358
- gmatch
  - BString.cpp, 421
- goTo
  - BList, 149
- handleReconnect
  - BoapClientObject, 169
- has
  - BList, 150
- hasKey
  - BDict, 75
  - BDictMap, 79
- hash
  - BString, 286
- hashAdd
  - BDict, 74
- hashDelete
  - BDict, 74
- hashFind
  - BDict, 74
- hashPrint
  - BDict, 75
- head
  - BoapMc1.h, 391
  - BoapMc1Packet, 184
  - BoapMcPacket, 196
- hostName
  - Boapns::BoapEntry, 174
- hour
  - BDuration, 87
  - BTimeStamp, 319
  - BTimeStampMs, 330
- htobe
  - BEndian.h, 368, 369
- htobe16
  - BEndian.h, 365
- htobe32
  - BEndian.h, 365
- htobe64
  - BEndian.h, 365
- htole
  - BEndian.h, 370, 371
- htole16
  - BEndian.h, 365
- htole32
  - BEndian.h, 366
- htole64
  - BEndian.h, 366
- IP\_MTU
  - BSocket.cpp, 416
- IS\_BIG\_ENDIAN
  - Boap.cpp, 382
- inString
  - BString, 286
- inUse
  - BCondResource, 51
- increment
  - BCondInt, 47
  - BCondValue, 54
  - BCondWrap, 58
  - BFifoCircPos, 131
- init
  - BComms, 39
  - BRtc, 244
  - BRtcThreaded, 246
  - BSocket, 263
  - BSpi, 274
  - BString, 286
  - BTask, 300
  - BoapServer, 215
  - BoapServerConnection, 221

- insert
  - BArray, [21](#)
  - BDict, [75](#)
  - BEntryList, [97](#)
  - BList, [150](#)
  - BMysql, [161](#)
  - BString, [286](#)
- insertAfter
  - BList, [150](#)
- Int16
  - BoapSimple.h, [395](#)
- Int32
  - BoapSimple.h, [395](#)
- int64ToString
  - BString.cpp, [421](#)
  - BString.h, [426](#)
- Int8
  - BoapSimple.h, [396](#)
- intToString
  - BString.cpp, [421](#)
  - BString.h, [426](#)
- isConnected
  - BComms, [39](#)
- isEnd
  - BDictMap, [80](#)
  - BFile, [135](#)
  - BList, [150](#)
- isLeap
  - BDate, [66](#)
  - BTimeStamp, [319](#)
  - BTimeStampMs, [328](#)
- isLeapYear
  - BTime, [308](#)
  - BTimeUs, [335](#)
- isOpen
  - BFile, [136](#)
- isSet
  - BDate, [67](#)
  - BEntryList, [97](#)
  - BTime, [308](#)
  - BTimeStamp, [319](#)
  - BTimeUs, [335](#)
- isSpace
  - BString, [286](#)
- item
  - BList::Node, [340](#)
- iterator
  - BDict, [72](#)
  - BDictMap, [79](#)
- justify
  - BString, [287](#)
- key
  - BDict, [75](#)
  - BDictItem, [77](#)
  - BDictMap, [80](#)
- le16toh
  - BEndian.h, [366](#)
- le32toh
  - BEndian.h, [366](#)
- le64toh
  - BEndian.h, [366](#)
- len
  - BRefData, [242](#)
  - BSocketAddress, [268](#)
  - BString, [287](#)
  - BStringLocked, [295](#)
- length
  - BFile, [136](#)
  - BoapMc.h, [387](#)
  - BoapMc1.h, [391](#)
  - BoapMc1PacketHead, [185](#)
  - BoapMcPacketHead, [198](#)
  - BoapPacketHead, [210, 211](#)
- letoh
  - BEndian.h, [371, 372](#)
- line
  - BEntry, [91](#)
- listen
  - BSocket, [264](#)
- lock
  - BCondResource, [51](#)
  - BMutex, [157](#)
  - BMutexLock, [159](#)
- locked
  - BCondResource, [51](#)
- lowerFirst
  - BString, [287](#)
- MDEBUG
  - BMutex.cpp, [380](#)
- MINUS
  - BString.cpp, [418](#)
- MSG\_NOSIGNAL
  - BSocket.h, [416](#)
- magic
  - BoapMc1.h, [391](#)
  - BoapMc1PacketHead, [185](#)
- mapCircularBuffer
  - BFifoCirc, [126](#)
- membersPrint
  - BObj, [232](#)
- microSecond
  - BDuration, [87](#)
  - BTimeStamp, [319](#)
- milliSecond
  - BTimeStampMs, [330](#)
- minute
  - BDuration, [87](#)
  - BTimeStamp, [319](#)
  - BTimeStampMs, [331](#)
- Mode
  - BSpi, [274](#)
- mon\_yday
  - BDate-1.cpp, [350](#)
  - BDate.cpp, [352](#)

- BTimeStamp.cpp, [433](#)
- BTimeStampMs.cpp, [434](#)
- monDays
  - BTime.cpp, [431](#)
  - BTimeUs.cpp, [435](#)
- month
  - BDate, [67](#)
  - BTimeStamp, [320](#)
- NType
  - BSocket, [261](#)
- name
  - BObjMember, [233](#)
  - BoapServiceObject, [226](#)
  - Boapns::BoapEntry, [174](#)
- nbytes
  - BoapPacket, [205](#)
- newConnection
  - BoapServer, [216](#)
- next
  - BDictMap, [80](#)
  - BList, [150](#)
  - BNode, [166](#)
- nextFd
  - BPoll, [236](#)
- Node
  - BList::Node, [340](#)
- nodeCreate
  - BList, [151](#)
- nodeGet
  - BList, [151](#)
- nprintf
  - BDebug.h, [357](#)
- num
  - BError, [101](#)
- number
  - BArray, [21](#)
  - BList, [151](#)
  - BoapMc1.h, [391](#)
  - BoapMc1Error, [183](#)
- oaddress
  - BSocketAddress, [270](#)
- oaddressFrom
  - BoapMc1Comms, [180](#)
  - BoapMcClientObject, [188](#)
  - BoapMcComms, [193](#)
- oaddressTo
  - BoapMc1Comms, [180](#)
  - BoapMcClientObject, [188](#)
  - BoapMcComms, [193](#)
- oapiVersion
  - BoapClientObject, [171](#)
  - BoapMc1Comms, [180](#)
  - BoapMcClientObject, [188](#)
  - BoapMcComms, [193](#)
  - BoapMcServiceObject, [199](#)
  - BoapServiceObject, [228](#)
- oarg
  - BEvent, [107](#)
- oaverage
  - BTimer, [313](#)
- oboapNs
  - BoapServer, [217](#)
- oboapServer
  - BoapServerConnection, [222](#)
- oboapns
  - BoapServer, [218](#)
- oclientGoneEvent
  - BoapServer, [218](#)
- oclients
  - BoapServer, [218](#)
- ocmd
  - BoapFuncEntry, [175](#)
- ocolumnWidths
  - BTable, [299](#)
- ocomments
  - BEntryFile, [94](#)
- ocomms
  - BoapMc1Comms, [180](#)
  - BoapMcClientObject, [188](#)
  - BoapMcComms, [193](#)
  - BoapMcSignalObject, [200](#)
- ocond
  - BCond, [43](#)
  - BCondBool, [45](#)
  - BCondInt, [49](#)
  - BCondResource, [52](#)
  - BCondValue, [56](#)
  - BCondWrap, [60](#)
  - BRtcThreaded, [247](#)
- oconnected
  - BComms, [40](#)
  - BoapClientObject, [171](#)
- odata
  - BBuffer, [29](#)
  - BFifo, [123](#)
  - BFifoCirc, [128](#)
  - BRefData, [243](#)
  - BTable, [299](#)
  - BoapPacket, [209](#)
- odataSize
  - BBuffer, [29](#)
- odb
  - BMysql, [162](#)
- odebug
  - BMysql, [163](#)
- odev
  - BSpi, [275](#)
- odevName
  - BSpi, [275](#)
- odirname
  - BDir, [84](#)
- oendTime
  - BTimer, [313](#)
- oerrNo
  - BError, [102](#)

- BErrorTime, [105](#)
- oerrStr
  - BError, [102](#)
  - BErrorTime, [105](#)
- oerrTime
  - BErrorTime, [105](#)
- oerror
  - BDir, [84](#)
  - BoapMc1Comms, [180](#)
- oevent
  - BComms, [40](#)
- oeventNum
  - BComms, [41](#)
- oeventQueue
  - BComms, [41](#)
- ofd
  - BRtc, [245](#)
- ofds
  - BEvent1Int, [112](#)
  - BEvent1Pipe, [114](#)
  - BEventPipe, [116](#)
  - BPoll, [237](#)
- ofdsNext
  - BPoll, [237](#)
- ofdsNum
  - BPoll, [237](#)
- ofile
  - BConfig, [62](#)
  - BFile, [138](#)
- ofileName
  - BConfig, [62](#)
  - BFile, [138](#)
- ofilename
  - BEntryFile, [94](#)
  - BFileData, [142](#)
- ofunc
  - BoapFuncEntry, [175](#)
- ofuncList
  - BoapServiceObject, [228](#)
- ohalfDuplex
  - BoapMc1Comms, [180](#)
- ohashLists
  - BDict, [76](#)
- ohashSize
  - BDict, [76](#)
- ohostName
  - BoapServer, [218](#)
- ohour
  - BDuration, [88](#)
  - BTimeStamp, [323](#)
- oi
  - Blter, [143](#)
- oinit
  - BUrl, [339](#)
- oisBoapns
  - BoapServer, [218](#)
- olastPos
  - BEntryList, [98](#)
- olen
  - BRefData, [243](#)
  - BSocketAddress, [270](#)
- olength
  - BList, [155](#)
- olock
  - BCondResource, [52](#)
  - BConfig, [62](#)
  - BFifoCirc, [128](#)
  - BMutexLock, [159](#)
  - BMysql, [163](#)
  - BQueue, [240](#)
  - BRWLock, [250](#)
  - BSemaphoreCount, [259](#)
  - BStringLocked, [296](#)
  - BTimer, [314](#)
  - BoapClientObject, [171](#)
  - BoapServer, [218](#)
- olockCall
  - BoapMc1Comms, [181](#)
  - BoapMcComms, [193](#)
- olockTx
  - BoapMc1Comms, [181](#)
  - BoapMcComms, [194](#)
- omaxLength
  - BoapClientObject, [172](#)
  - BoapServerConnection, [222](#)
- omicroSecond
  - BDuration, [88](#)
  - BTimeStamp, [323](#)
- ominute
  - BDuration, [88](#)
  - BTimeStamp, [323](#)
- omode
  - BFile, [139](#)
- omutex
  - BCond, [43](#)
  - BCondBool, [45](#)
  - BCondInt, [49](#)
  - BCondResource, [52](#)
  - BCondValue, [56](#)
  - BCondWrap, [60](#)
  - BMutex, [158](#)
- oname
  - BEntry, [92](#)
  - BNameValue, [164](#)
  - BTask, [301](#)
  - BoapClientObject, [172](#)
  - BoapServiceObject, [228](#)
- onbytes
  - BoapPacket, [209](#)
- onet
  - BoapServer, [218](#)
- onetEvent
  - BoapServer, [218](#)
- onetEventAddress
  - BoapServer, [219](#)
- onodes

- BList, [155](#)
- onum
  - BTimer, [314](#)
- onumOperations
  - BoapServer, [219](#)
- onumber
  - BQueue, [240](#)
- oobject
  - BoapServiceEntry, [223](#)
- oopened
  - BMysql, [163](#)
- opacket
  - BoapMcClientObject, [188](#)
  - BoapMcComms, [194](#)
- opacketMode
  - BComms, [41](#)
- opacketReqQueue
  - BoapMcComms, [194](#)
- opacketReqRx
  - BoapMcComms, [194](#)
- opacketReqTx
  - BoapMcComms, [194](#)
- opacketRpcCmd
  - BoapMc1Comms, [181](#)
- opacketRpcDoneSema
  - BoapMc1Comms, [181](#)
- opacketRpcSema
  - BoapMc1Comms, [181](#)
- opacketRx
  - BoapMc1Comms, [181](#)
  - BoapMcComms, [194](#)
- opacketRxBase
  - BoapMc1Comms, [182](#)
- opacketRxSema
  - BoapMcComms, [195](#)
- opacketTx
  - BoapMc1Comms, [182](#)
  - BoapMcComms, [195](#)
- opacketTxBase
  - BoapMc1Comms, [182](#)
- opacketTxQueue
  - BoapMcComms, [195](#)
- opacketTxQueueWriteNum
  - BoapMcComms, [195](#)
- opacketTxSema
  - BoapMcComms, [195](#)
- opeak
  - BTimer, [314](#)
- open
  - BConfig, [62](#)
  - BDir, [83](#)
  - BEntryFile, [94](#)
  - BFile, [136](#)
  - BFileData, [141](#)
  - BMysql, [162](#)
- operator BNode \*
  - BIter, [143](#)
- operator BString
  - BDate, [67](#)
  - BStringLocked, [295](#)
  - BTimeStamp, [320](#)
- operator BTime
  - BTimeUs, [335](#)
- operator const char \*
  - BString, [287](#)
- operator const SockAddr \*
  - BSocketAddress, [269](#)
- operator int
  - BCondBool, [44](#)
  - BError, [101](#)
  - BErrorTime, [105](#)
  - BFifoCircPos, [131](#)
  - BSemaphoreBool, [256](#)
- operator long
  - BAtomicCount, [25](#)
- operator Type
  - BAtomic, [23](#)
- operator!=
  - BDate, [67](#)
  - BFifoCircPos, [131](#)
  - BSocketAddress, [269](#)
  - BString, [287](#)
  - BTime, [308](#)
  - BTimeStamp, [320](#)
  - BTimeUs, [335](#)
- operator<
  - BDate, [67](#)
  - BString, [289](#)
  - BTime, [309](#)
  - BTimeStamp, [320](#)
  - BTimeStampMs, [329](#)
  - BTimeUs, [336](#)
- operator<<
  - BString.cpp, [422](#)
  - BString.h, [426](#)
- operator<=
  - BDate, [67](#)
  - BString, [289](#)
  - BTime, [309](#)
  - BTimeStamp, [320](#)
  - BTimeStampMs, [329](#)
  - BTimeUs, [336](#)
- operator>
  - BDate, [68](#)
  - BString, [290](#)
  - BTime, [309](#)
  - BTimeStamp, [321](#)
  - BTimeStampMs, [329](#)
  - BTimeUs, [336](#)
- operator>>
  - BString.cpp, [422](#)
  - BString.h, [426](#)
- operator>=
  - BDate, [68](#)
  - BString, [290](#)
  - BTime, [309](#)



- BTimeStamp, 321
- BTimeStampMs, 329
- BTimeUs, 336
- operator+
  - BDict, 75
  - BList, 151
  - BString, 288
  - BStringLocked, 295
  - BTime, 308
  - BTimeUs, 335
- operator++
  - BAtomic, 23
  - BAtomicCount, 25
  - BCondInt, 47
  - BCondValue, 54
  - BCondWrap, 58
- operator+=
  - BCondInt, 47
  - BCondValue, 54
  - BCondWrap, 58
  - BFifoCircPos, 131
  - BString, 288, 289
  - BTime, 309
  - BTimeUs, 335
- operator--
  - BAtomic, 24
  - BAtomicCount, 26
  - BCondInt, 47
  - BCondValue, 54
  - BCondWrap, 59
- operator-=
  - BCondInt, 48
  - BCondValue, 55
  - BCondWrap, 59
- operator=
  - BDict, 75
  - BEntryList, 98
  - BFile, 136
  - BList, 152
  - BMutex, 157
  - BRWLock, 249
  - BRefData, 242
  - BSema, 251
  - BSemaphore, 254
  - BSemaphoreBool, 256
  - BSemaphoreCount, 258
  - BSocketAddress, 269
  - BString, 289
  - BStringLocked, 296
  - BTimeStamp, 320
- operator==
  - BDate, 67
  - BFifoCircPos, 131
  - Blter, 143
  - BSemaphoreBool, 256
  - BSocketAddress, 269
  - BString, 289
  - BTime, 309
  - BTimeStamp, 320
  - BTimeUs, 336
- operator[]
  - BDict, 76
  - BDictMap, 80
  - BFifo, 118
  - BFifoCirc, 126
  - BList, 152
  - BString, 290
- opolicy
  - BTask, 301
  - BThread, 305
- opoll
  - BoapServer, 219
- opos
  - BBufferStore, 36
  - BFifoCircPos, 132
  - BoapPacket, 209
- opriority
  - BTask, 302
  - BThread, 305
  - BoapClientObject, 172
- orate
  - BRtc, 245
  - BRtcThreaded, 247
- oreadPos
  - BFifo, 123
  - BFifoCirc, 129
- oreconnect
  - BoapClientObject, 172
- orefCount
  - BRefData, 243
- oreqSize
  - BoapMc1Comms, 182
- ores
  - BUrl, 339
- oresult
  - BThread, 305
- ortc
  - BRtcThreaded, 247
- orunning
  - BTask, 302
  - BThread, 305
- orx
  - BoapClientObject, 172
  - BoapServer, 219
  - BoapServerConnection, 222
  - BoapSignalObject, 230
- osecond
  - BDuration, 88
  - BTimeStamp, 323
- osema
  - BSema, 252
  - BSemaphore, 254
  - BSemaphoreBool, 257
  - BSemaphoreCount, 259
- oseparator
  - BFileCsv, 140

- oserver
  - BoapServiceObject, 228
- oservice
  - BoapClientObject, 172
  - BoapServiceEntry, 224
- oservices
  - BoapServer, 219
- osize
  - BBuffer, 29
  - BFifo, 123
  - BFifoCirc, 129
  - BFifoCircPos, 132
  - BQueue, 240
  - BoapPacket, 209
- oslave
  - BoapMcComms, 195
- osocket
  - BSocket, 266
  - BoapServerConnection, 222
- osort
  - BDir, 84
- ospare
  - BDuration, 89
  - BTimeStamp, 324
- ostackSize
  - BTask, 302
  - BThread, 306
- ostartTime
  - BTimer, 314
- ostr
  - BString, 294
  - BStringLocked, 296
- oswapBytes
  - BBufferStore, 36
- othread
  - BTask, 302
  - BThread, 306
- othreaded
  - BoapMc1Comms, 182
  - BoapMcComms, 196
  - BoapServer, 219
- otime
  - BTime, 310
  - BTimeUs, 337
- otimeout
  - BComms, 41
  - BoapClientObject, 172
  - BoapMc1Comms, 182
  - BoapMcComms, 196
- otitle
  - BTable, 299
- otx
  - BoapClientObject, 173
  - BoapServer, 219
  - BoapServerConnection, 222
  - BoapSignalObject, 230
- otype
  - BEvent, 107
  - BEvent1, 109
- ouse
  - BCondResource, 52
- ovalue
  - BAtomic, 24
  - BAtomicCount, 26
  - BCondBool, 45
  - BCondInt, 49
  - BCondValue, 56
  - BCondWrap, 60
  - BEntry, 92
  - BNameValue, 164
  - BSemaphoreBool, 257
  - BSemaphoreCount, 259
- overview.dox, 443
- ovmSize
  - BFifoCirc, 129
- owild
  - BDir, 84
- owriteNumFifoSamples
  - BFifoCirc, 129
- owritePos
  - BFifo, 123
  - BFifoCirc, 129
- oyday
  - BDate, 69
  - BTimeStamp, 324
- oyear
  - BDate, 70
  - BTimeStamp, 324
- packetMode
  - BComms, 39
- packetRecv
  - BoapMcComms, 190
- packetRx
  - BoapMc1Comms, 177
- packetRxData
  - BoapMc1Comms, 178
- packetRxEnd
  - BoapMc1Comms, 178
- packetSend
  - BoapMcComms, 191
- packetTx
  - BoapMc1Comms, 178
- pad
  - BString, 290
- peak
  - BTimer, 313
- peekHead
  - BoapPacket, 205
- performCall
  - BoapClientObject, 169
  - BoapMcClientObject, 187
  - BoapMcComms, 191
- performRecv
  - BoapClientObject, 170
  - BoapMcClientObject, 187
- performSend

- BoapClientObject, [170](#)
- BoapMcClientObject, [187](#)
- BoapMcComms, [191](#)
- BoapMcSignalObject, [200](#)
- BoapSignalObject, [230](#)
- ping
  - BoapClientObject, [170](#)
- pingLocked
  - BoapClientObject, [170](#)
- PollFd
  - BPoll, [235](#)
- pop
  - BBufferStore, [31–33](#)
  - BList, [152](#)
  - BoapPacket, [205, 206](#)
- popHead
  - BoapPacket, [206, 207](#)
- port
  - BSocketAddressINET, [272](#)
  - Boapns::BoapEntry, [174](#)
- pos
  - BFifoCircPos, [131](#)
- position
  - BFile, [137](#)
  - BList, [153](#)
- post
  - BSema, [251](#)
- prev
  - BList, [153](#)
  - BNode, [166](#)
- print
  - BEntry, [91](#)
  - BEntryList, [98](#)
  - BTable, [298](#)
- printLine
  - BTable, [298](#)
- printf
  - BFile, [137](#)
  - BString, [290](#)
- Priority
  - BSocket, [261](#)
- process
  - BoapMcServiceObject, [199](#)
  - BoapServer, [216](#)
  - BoapServerConnection, [221](#)
  - BoapServiceObject, [226](#)
- processEvent
  - BoapMcServiceObject, [199](#)
  - BoapServer, [216, 217](#)
  - BoapServiceObject, [226, 227](#)
- processPacket
  - BoapMcComms, [191](#)
- processRequest
  - BoapMc1Comms, [178](#)
  - BoapMcComms, [191](#)
- processRequests
  - BoapMc1Comms, [178](#)
  - BoapMcComms, [191](#)
- processRx
  - BoapMc1Comms, [178](#)
  - BoapMcComms, [192](#)
- pullLine
  - BString, [291](#)
- pullSeparators
  - BString, [291](#)
- pullToken
  - BString, [291](#)
- pullWord
  - BString, [291](#)
- push
  - BBufferStore, [33–35](#)
  - BList, [153](#)
  - BoapPacket, [207, 208](#)
- pushHead
  - BoapPacket, [208](#)
- query
  - BMySQL, [162](#)
- queueAdd
  - BList, [153](#)
- queueGet
  - BList, [153](#)
- raw
  - BSocketAddress, [269](#)
- rdLock
  - BRWLock, [249](#)
- read
  - BComms, [39](#)
  - BConfig, [62](#)
  - BDir, [83](#)
  - BEntryFile, [94](#)
  - BEventPipe, [115](#)
  - BFifo, [118, 119](#)
  - BFifoCirc, [126](#)
  - BFile, [137](#)
  - BFileData, [141](#)
  - BQueue, [239](#)
- readAvailable
  - BComms, [39](#)
  - BEventPipe, [116](#)
  - BFifo, [119](#)
  - BFifoCirc, [126](#)
  - BQueue, [239](#)
- readAvailableChunk
  - BFifo, [119](#)
- readCsv
  - BFileCsv, [140](#)
- readData
  - BFifo, [119](#)
  - BFifoCirc, [126](#)
- readDone
  - BFifo, [120](#)
  - BFifoCirc, [126](#)
- readPos
  - BFifo, [120](#)
- readString

- BFile, [137](#)
- BUrl, [339](#)
- readWaitAvailable
  - BFifoCirc, [127](#)
- rear
  - BArray, [21](#)
  - BList, [154](#)
- rebase
  - BFifo, [120](#)
- recv
  - BSocket, [264](#)
- recvAvailable
  - BSocket, [264](#)
- recvFrom
  - BSocket, [264](#)
- recvFromWithTimeout
  - BSocket, [264](#)
- recvWithTimeout
  - BSocket, [264](#)
- removeNL
  - BString, [291](#)
- removeSeparators
  - BString, [291](#)
- reserved
  - BoapPacketHead, [211](#)
- resize
  - BBuffer, [28](#)
  - BFifo, [120](#)
  - BoapPacket, [208](#)
- result
  - BThread, [304](#)
- retDouble
  - BString, [292](#)
- retFloat64
  - BString, [292](#)
- retInt
  - BString, [292](#)
- retStr
  - BString, [292](#)
- retStrDup
  - BString, [292](#)
- retUInt
  - BString, [292](#)
- reverse
  - BString, [293](#)
- roundSize
  - BBuffer.cpp, [343](#)
  - BoapSimple.cc, [394](#)
- run
  - BTask, [300](#)
  - BoapServer, [217](#)
- running
  - BThread, [304](#)
- SO\_PRIORITY
  - BSocket.h, [416](#)
- SOL\_IP
  - BSocket.h, [417](#)
- STRBUF
  - BFile.cpp, [377](#)
- STRIP
  - BString.cpp, [418](#)
- sampleNumber
  - BTimeStampMs, [331](#)
- second
  - BDuration, [87](#)
  - BTimeStamp, [321](#)
  - BTimeStampMs, [331](#)
- seek
  - BFile, [137](#)
- send
  - BSocket, [265](#)
- sendChunks
  - BSocket, [265](#)
- sendEvent
  - BEvent1Int, [112](#)
  - BEvent1Pipe, [114](#)
  - BoapMcServiceObject, [199](#)
  - BoapServer, [217](#)
  - BoapServiceObject, [227](#)
- sendTo
  - BSocket, [265](#)
- service
  - BoapPacketHead, [211](#)
  - Boapns::BoapEntry, [174](#)
- set
  - BCondBool, [44](#)
  - BDate, [68](#)
  - BDuration, [88](#)
  - BError, [101](#)
  - BErrorTime, [105](#)
  - BFifoCircPos, [132](#)
  - BSemaphore, [254](#)
  - BSemaphoreBool, [256](#)
  - BSocketAddress, [269](#)
  - BSocketAddressINET, [272](#), [273](#)
  - BTime, [309](#), [310](#)
  - BTimeStamp, [321](#)
  - BTimeUs, [336](#)
- setAddress
  - BoapMc1Comms, [179](#)
  - BoapMcClientObject, [187](#)
  - BoapMcComms, [192](#)
- setBinary
  - BEvent1, [108](#)
  - BEvent1Error, [110](#)
- setBroadCast
  - BSocket, [265](#)
- setComms
  - BoapMc1Comms, [179](#)
  - BoapMcComms, [192](#)
- setCommsMode
  - BoapMc1Comms, [179](#)
  - BoapMcComms, [192](#)
- setConnectionPriority
  - BoapClientObject, [171](#)
- setData

- BBuffer, 28
- BoapPacket, 209
- setDebug
  - BDebug.cpp, 354
  - BDebug.h, 358
  - BMySQL, 162
- setDurationString
  - BTimeStampMs, 329
- setError
  - BError, 102
- setFd
  - BSocket, 265
- setFirst
  - BDate, 68
  - BTimeStamp, 322
- setHexString
  - BBufferStore, 35
- setInitPriority
  - BThread, 304
- setInitStackSize
  - BThread, 304
- setLast
  - BDate, 68
  - BTimeStamp, 322
- setLen
  - BRefData, 242
- setLine
  - BEntry, 91
- setMaxLength
  - BoapClientObject, 171
  - BoapServerConnection, 222
- setMember
  - BObj, 232
- setMembers
  - BObj, 232
- setName
  - BEntry, 91
  - BoapServiceObject, 228
- setNow
  - BDate, 69
  - BTimeStamp, 322
  - BTimeStampMs, 329
- setPacketMode
  - BComms, 39
- setPort
  - BSocketAddressINET, 273
- setPos
  - BBufferStore, 36
- setPriority
  - BSocket, 266
  - BTask, 300
  - BThread, 304
- setReuseAddress
  - BSocket, 266
- setSize
  - BBuffer, 28
  - BFifoCircPos, 132
- setSockOpt
  - BSocket, 266
- setSort
  - BDir, 83
- setString
  - BDate, 69
  - BDuration, 88
  - BTime, 310
  - BTimeStamp, 322
  - BTimeStampMs, 330
  - BTimeUs, 337
- setTime
  - BTimeStamp, 322
- setTimeout
  - BComms, 39
  - BoapClientObject, 171
  - BoapMc1Comms, 179
  - BoapMcComms, 193
- setTitle
  - BTable, 298
- setVBuf
  - BFile, 137
- setValue
  - BCondInt, 48
  - BCondValue, 55
  - BCondWrap, 59
  - BEntry, 91
  - BEntryList, 98
  - BSemaphoreCount, 258
- setValueRaw
  - BEntryList, 98
- setWild
  - BDir, 84
- setYDay
  - BDate, 69
  - BTimeStamp, 322
- setYearDay
  - BTime, 310
  - BTimeUs, 337
- shutdown
  - BSocket, 266
- signal
  - BCond, 42
- size
  - BBuffer, 28
  - BDataChunk, 63
  - BDictMap, 80
  - BFifo, 120
  - BFifoCirc, 127
  - BList, 154
  - BObjMember, 233
- SockAddr
  - BSocketAddress, 267
- SockAddrIP
  - BSocketAddressINET, 271
- sort
  - BArray, 22
  - BList, 154
- SortFunc

- BArray, 20
- BList, 146
- split
  - BString, 293
- start
  - BCondResource, 51
  - BDictMap, 81
  - BList, 154
  - BTask, 301
  - BThread, 305
  - BTimer, 313
- startFunc
  - BThread, 305
- stop
  - BTask, 301
  - BTimer, 313
- str
  - BError, 102
- string
  - BoapMc1.h, 392
  - BoapMc1Error, 183
- subMilliseconds
  - BTimeStampMs, 330
- subSeconds
  - BTimeStampMs, 330
- subString
  - BString, 293
- swap
  - BList, 155
- table\_crc\_hi
  - BCrc16.cpp, 347
- table\_crc\_lo
  - BCrc16.cpp, 347
- take
  - BSemaphoreCount, 258
- taskFunc
  - BTask, 301
- timedLock
  - BMutex, 157
- timedWait
  - BCond, 42
  - BCondBool, 44
  - BSema, 251
- timeoutTicks
  - BTypes.h, 442
- to\_hex
  - BString.h, 426
- toBDictStringFromJson
  - BObjStringFormat.cpp, 398
  - BObjStringFormat.h, 405
- toBString
  - BDate-1.cpp, 350
  - BDate.cpp, 351
  - BDate.h, 352
  - BDict.cpp, 359
  - BDict.h, 361
  - BObjStringFormat.cpp, 398–401
  - BObjStringFormat.h, 405–408
  - BString.cpp, 422, 423
  - BString.h, 427
  - BTimeStamp.cpp, 433
  - BTimeStamp.h, 434
- toBStringJson
  - BObjStringFormat.cpp, 401–404
  - BObjStringFormat.h, 408–411
- toLower
  - BString, 293
- toUpper
  - BString, 293
- tprintf
  - BDebug.h, 358
- transact
  - BSpi, 274
- translateChar
  - BString, 293
- truncate
  - BFile, 138
  - BString, 294
- tryLock
  - BMutex, 157
- tryRdLock
  - BRWLock, 249
- tryWait
  - BSema, 252
- tryWrLock
  - BRWLock, 249
- Type
  - BErrorTime, 103
  - BMutex, 156
- type
  - BEvent, 106
  - BObjMember, 233
  - BoapPacketHead, 211
- typeComp
  - BObjMember, 233
- typeName
  - BObjMember, 233
- UInt16
  - BoapSimple.h, 396
- UInt32
  - BoapSimple.h, 396
- UInt8
  - BoapSimple.h, 396
- unlock
  - BCondResource, 51
  - BMutex, 158
  - BMutexLock, 159
  - BRWLock, 249
- unmapCircularBuffer
  - BFifoCirc, 127
- update
  - BMysql, 162
- updateHead
  - BoapPacket, 209
- updateLen
  - BoapPacket, 209

- valid
  - [Blter](#), [143](#)
- validate
  - [BoapServerConnection](#), [222](#)
- value
  - [BCondBool](#), [45](#)
  - [BCondInt](#), [48](#)
  - [BCondValue](#), [55](#)
  - [BCondWrap](#), [59](#)
  - [BDictItem](#), [78](#)
  - [BSemaphoreBool](#), [256](#)
  - [BSemaphoreCount](#), [259](#)
- wait
  - [BComms](#), [40](#)
  - [BCond](#), [42](#)
  - [BCondBool](#), [45](#)
  - [BRtc](#), [244](#)
  - [BRtcThreaded](#), [247](#)
  - [BSema](#), [252](#)
  - [BSemaphore](#), [254](#)
  - [BSemaphoreBool](#), [256](#)
  - [BSemaphoreCount](#), [259](#)
- waitForCompletion
  - [BTask](#), [301](#)
  - [BThread](#), [305](#)
- waitLessThan
  - [BCondInt](#), [48](#)
  - [BCondValue](#), [55](#)
  - [BCondWrap](#), [59](#)
- waitLessThanOrEqual
  - [BCondInt](#), [48](#)
  - [BCondValue](#), [55](#)
  - [BCondWrap](#), [59](#)
- waitMoreThanOrEqual
  - [BCondInt](#), [49](#)
  - [BCondValue](#), [56](#)
  - [BCondWrap](#), [60](#)
- wild
  - [BDir.cpp](#), [362](#)
- wildString
  - [BDir.cpp](#), [362](#)
- wprintf
  - [BDebug.h](#), [357](#)
- wrLock
  - [BRWLock](#), [249](#)
- write
  - [BComms](#), [40](#)
  - [BConfig](#), [62](#)
  - [BEntryFile](#), [94](#)
  - [BEventPipe](#), [116](#)
  - [BFifo](#), [121](#)
  - [BFifoCirc](#), [127](#)
  - [BFile](#), [138](#)
  - [BFileData](#), [142](#)
  - [BQueue](#), [239](#)
- writeAvailable
  - [BComms](#), [40](#)
  - [BEventPipe](#), [116](#)
  - [BFifo](#), [121](#)
  - [BFifoCirc](#), [127](#)
  - [BQueue](#), [239](#)
- writeAvailableChunk
  - [BFifo](#), [121](#)
- writeBackup
  - [BFifo](#), [121](#)
- writeChunks
  - [BComms](#), [40](#)
- writeCsv
  - [BFileCsv](#), [140](#)
- writeData
  - [BBuffer](#), [28](#)
  - [BFifo](#), [122](#)
  - [BFifoCirc](#), [128](#)
  - [BUrl](#), [339](#)
- writeDone
  - [BFifo](#), [122](#)
  - [BFifoCirc](#), [128](#)
- writeList
  - [BEntryFile](#), [94](#)
- writePos
  - [BFifo](#), [122](#)
- writeString
  - [BFile](#), [138](#)
- writeWaitAvailable
  - [BFifoCirc](#), [128](#)
- yday
  - [BDate](#), [69](#)
  - [BTimeStamp](#), [323](#)
  - [BTimeStampMs](#), [331](#)
- year
  - [BDate](#), [69](#)
  - [BTimeStamp](#), [323](#)
  - [BTimeStampMs](#), [331](#)
- yearDays
  - [BTime.cpp](#), [431](#)
  - [BTimeUs.cpp](#), [435](#)
- yearsLeap
  - [BTime.cpp](#), [431](#)
  - [BTimeUs.cpp](#), [435](#)