

Beam-lib

2.16.3

Generated by Doxygen 1.8.15



<b>1 Main Page</b>	<b>1</b>
1.1 Introduction	1
1.2 Components	2
1.3 API Examples	2
<b>2 Namespace Index</b>	<b>3</b>
2.1 Namespace List	3
<b>3 Hierarchical Index</b>	<b>5</b>
3.1 Class Hierarchy	5
<b>4 Class Index</b>	<b>9</b>
4.1 Class List	9
<b>5 File Index</b>	<b>13</b>
5.1 File List	13
<b>6 Namespace Documentation</b>	<b>17</b>
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	22
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]	23
7.2.3.6 operator--() [1/2]	24
7.2.3.7 operator--() [2/2]	24
7.3 BAtomicCount Class Reference	24
7.3.1 Detailed Description	24
7.3.2 Constructor & Destructor Documentation	24
7.3.2.1 BAtomicCount()	24
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]	25
7.3.3.6 operator--() [1/2]	25
7.3.3.7 operator--() [2/2]	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()	27
7.4.2.2 resize()	27
7.4.2.3 setData()	27
7.4.2.4 setSize()	28
7.4.2.5 size()	28
7.4.2.6 writeData()	28
7.4.3 Member Data Documentation	28
7.4.3.1 odata	28
7.4.3.2 odataSize	28
7.4.3.3 osize	28
7.5 BBufferStore Class Reference	29
7.5.1 Constructor & Destructor Documentation	30
7.5.1.1 BBufferStore()	30
7.5.1.2 ~BBufferStore()	30
7.5.2 Member Function Documentation	30
7.5.2.1 getHexString()	30
7.5.2.2 getPos()	30
7.5.2.3 pop() [1/15]	30

7.5.2.4 pop() [2/15]	31
7.5.2.5 pop() [3/15]	31
7.5.2.6 pop() [4/15]	31
7.5.2.7 pop() [5/15]	31
7.5.2.8 pop() [6/15]	31
7.5.2.9 pop() [7/15]	31
7.5.2.10 pop() [8/15]	31
7.5.2.11 pop() [9/15]	32
7.5.2.12 pop() [10/15]	32
7.5.2.13 pop() [11/15]	32
7.5.2.14 pop() [12/15]	32
7.5.2.15 pop() [13/15]	32
7.5.2.16 pop() [14/15]	32
7.5.2.17 pop() [15/15]	32
7.5.2.18 push() [1/15]	33
7.5.2.19 push() [2/15]	33
7.5.2.20 push() [3/15]	33
7.5.2.21 push() [4/15]	33
7.5.2.22 push() [5/15]	33
7.5.2.23 push() [6/15]	33
7.5.2.24 push() [7/15]	33
7.5.2.25 push() [8/15]	34
7.5.2.26 push() [9/15]	34
7.5.2.27 push() [10/15]	34
7.5.2.28 push() [11/15]	34
7.5.2.29 push() [12/15]	34
7.5.2.30 push() [13/15]	34
7.5.2.31 push() [14/15]	34
7.5.2.32 push() [15/15]	35
7.5.2.33 setHexString()	35
7.5.2.34 setPos()	35
7.5.3 Member Data Documentation	35
7.5.3.1 opos	35
7.5.3.2 oswapBytes	35
7.6 BComms Class Reference	35
7.6.1 Member Enumeration Documentation	36
7.6.1.1 Flush	36
7.6.2 Constructor & Destructor Documentation	37
7.6.2.1 BComms()	37
7.6.2.2 ~BComms()	37
7.6.3 Member Function Documentation	37
7.6.3.1 byteRate()	37

7.6.3.2 close()	37
7.6.3.3 connect()	37
7.6.3.4 disconnect()	38
7.6.3.5 eventEnable()	38
7.6.3.6 eventQueue()	38
7.6.3.7 flush()	38
7.6.3.8 init()	38
7.6.3.9 isConnected()	38
7.6.3.10 name()	39
7.6.3.11 packetMode()	39
7.6.3.12 read()	39
7.6.3.13 readAvailable()	39
7.6.3.14 setPacketMode()	39
7.6.3.15 setTimeout()	39
7.6.3.16 wait()	40
7.6.3.17 write()	40
7.6.3.18 writeAvailable()	40
7.6.3.19 writeChunks()	40
7.6.4 Member Data Documentation	40
7.6.4.1 oconnected	40
7.6.4.2 oevent	40
7.6.4.3 oeventEnabled	41
7.6.4.4 oeventNum	41
7.6.4.5 oeventQueue	41
7.6.4.6 oeventSet	41
7.6.4.7 opacketMode	41
7.6.4.8 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()	43
7.8 BCondBool Class Reference	43
7.8.1 Detailed Description	43
7.8.2 Constructor & Destructor Documentation	43
7.8.2.1 BCondBool()	43
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()	44
7.8.3.5 value()	44
7.8.3.6 wait()	45
7.9 BCondInt Class Reference	45
7.9.1 Detailed Description	45
7.9.2 Constructor & Destructor Documentation	46
7.9.2.1 BCondInt()	46
7.9.2.2 ~BCondInt()	46
7.9.3 Member Function Documentation	46
7.9.3.1 decrement()	46
7.9.3.2 increment()	46
7.9.3.3 operator++()	46
7.9.3.4 operator+=()	47
7.9.3.5 operator--()	47
7.9.3.6 operator-=()	47
7.9.3.7 setValue()	47
7.9.3.8 value()	47
7.9.3.9 waitLessThan()	47
7.9.3.10 waitLessThanOrEqual()	48
7.9.3.11 waitMoreThanOrEqual()	48
7.10 BCondResource Class Reference	48
7.10.1 Detailed Description	48
7.10.2 Constructor & Destructor Documentation	49
7.10.2.1 BCondResource()	49
7.10.2.2 ~BCondResource()	49
7.10.3 Member Function Documentation	49
7.10.3.1 end()	49
7.10.3.2 inUse()	49
7.10.3.3 lock()	49
7.10.3.4 locked()	49
7.10.3.5 start()	50
7.10.3.6 unlock()	50
7.11 BCondValue Class Reference	50
7.11.1 Detailed Description	51
7.11.2 Constructor & Destructor Documentation	51
7.11.2.1 BCondValue()	51
7.11.2.2 ~BCondValue()	51
7.11.3 Member Function Documentation	51
7.11.3.1 decrement()	51
7.11.3.2 increment()	51

7.11.3.3 operator++()	51
7.11.3.4 operator+=()	52
7.11.3.5 operator--()	52
7.11.3.6 operator-=()	52
7.11.3.7 setValue()	52
7.11.3.8 value()	52
7.11.3.9 waitLessThan()	52
7.11.3.10 waitLessThanOrEqual()	53
7.11.3.11 waitMoreThanOrEqual()	53
7.12 BCondWrap Class Reference	53
7.12.1 Constructor & Destructor Documentation	54
7.12.1.1 BCondWrap()	54
7.12.1.2 ~BCondWrap()	54
7.12.2 Member Function Documentation	54
7.12.2.1 decrement()	54
7.12.2.2 increment()	54
7.12.2.3 operator++()	54
7.12.2.4 operator+=()	55
7.12.2.5 operator--()	55
7.12.2.6 operator-=()	55
7.12.2.7 setValue()	55
7.12.2.8 value()	55
7.12.2.9 waitLessThan()	55
7.12.2.10 waitLessThanOrEqual()	56
7.12.2.11 waitMoreThanOrEqual()	56
7.13 BConfig Class Reference	56
7.13.1 Detailed Description	57
7.13.2 Member Function Documentation	57
7.13.2.1 close()	57
7.13.2.2 fileName()	57
7.13.2.3 findValue()	57
7.13.2.4 open()	57
7.13.2.5 read()	57
7.13.2.6 write()	58
7.14 BDataChunk Class Reference	58
7.14.1 Constructor & Destructor Documentation	58
7.14.1.1 BDataChunk()	58
7.14.2 Member Data Documentation	58
7.14.2.1 data	58
7.14.2.2 size	59
7.15 BDate Class Reference	59
7.15.1 Constructor & Destructor Documentation	60



7.15.1.1 BDate() [1/2]	60
7.15.1.2 BDate() [2/2]	60
7.15.1.3 ~BDate()	60
7.15.2 Member Function Documentation	60
7.15.2.1 clear()	60
7.15.2.2 compare()	61
7.15.2.3 day()	61
7.15.2.4 daysInMonth()	61
7.15.2.5 getDate()	61
7.15.2.6 getString()	61
7.15.2.7 getStringFormatted()	61
7.15.2.8 isLeap()	62
7.15.2.9 isSet()	62
7.15.2.10 month()	62
7.15.2.11 operator BString()	62
7.15.2.12 operator"!="()	62
7.15.2.13 operator<()	62
7.15.2.14 operator<=()	62
7.15.2.15 operator==()	63
7.15.2.16 operator>()	63
7.15.2.17 operator>=()	63
7.15.2.18 set() [1/2]	63
7.15.2.19 set() [2/2]	63
7.15.2.20 setFirst()	63
7.15.2.21 setLast()	64
7.15.2.22 setNow()	64
7.15.2.23 setString()	64
7.15.2.24 setYDay()	64
7.15.2.25 yday()	64
7.15.2.26 year()	64
7.15.3 Member Data Documentation	64
7.15.3.1 oyday	65
7.15.3.2 oyear	65
7.16 BDebugBacktrace Class Reference	65
7.16.1 Constructor & Destructor Documentation	65
7.16.1.1 BDebugBacktrace()	65
7.16.1.2 ~BDebugBacktrace()	65
7.16.2 Member Function Documentation	66
7.16.2.1 dumpBacktrace()	66
7.16.2.2 dumpBacktraceFile()	66
7.16.2.3 dumpBacktraceStdout()	66
7.16.2.4 dumpBacktraceSyslog()	66

7.17 BDict< Type > Class Template Reference	66
7.17.1 Member Typedef Documentation	67
7.17.1.1 iterator	67
7.17.2 Constructor & Destructor Documentation	67
7.17.2.1 BDict() [1/2]	68
7.17.2.2 BDict() [2/2]	68
7.17.3 Member Function Documentation	68
7.17.3.1 append() [1/2]	68
7.17.3.2 append() [2/2]	68
7.17.3.3 clear()	68
7.17.3.4 del() [1/2]	69
7.17.3.5 del() [2/2]	69
7.17.3.6 find()	69
7.17.3.7 hashPrint()	69
7.17.3.8 hasKey()	69
7.17.3.9 insert()	69
7.17.3.10 key()	70
7.17.3.11 operator+()	70
7.17.3.12 operator=()	70
7.17.3.13 operator[]() [1/6]	70
7.17.3.14 operator[]() [2/6]	70
7.17.3.15 operator[]() [3/6]	70
7.17.3.16 operator[]() [4/6]	71
7.17.3.17 operator[]() [5/6]	71
7.17.3.18 operator[]() [6/6]	71
7.18 BDictItem< Type > Class Template Reference	71
7.18.1 Detailed Description	71
7.18.2 Constructor & Destructor Documentation	72
7.18.2.1 BDictItem()	72
7.18.3 Member Data Documentation	72
7.18.3.1 key	72
7.18.3.2 value	72
7.19 BDictMap< Value > Class Template Reference	72
7.19.1 Detailed Description	73
7.19.2 Member Typedef Documentation	73
7.19.2.1 iterator	73
7.19.3 Member Function Documentation	73
7.19.3.1 clear()	73
7.19.3.2 del() [1/2]	73
7.19.3.3 del() [2/2]	74
7.19.3.4 hasKey()	74
7.19.3.5 isEnd()	74

7.19.3.6 key()	74
7.19.3.7 next()	74
7.19.3.8 operator[]() [1/2]	74
7.19.3.9 operator[]() [2/2]	75
7.19.3.10 size()	75
7.19.3.11 start()	75
7.20 BDir Class Reference	75
7.20.1 Detailed Description	76
7.20.2 Constructor & Destructor Documentation	76
7.20.2.1 BDir() [1/2]	76
7.20.2.2 BDir() [2/2]	76
7.20.2.3 ~BDir()	77
7.20.3 Member Function Documentation	77
7.20.3.1 clear()	77
7.20.3.2 entryName()	77
7.20.3.3 entryStat()	77
7.20.3.4 entryStat64()	77
7.20.3.5 error()	78
7.20.3.6 open()	78
7.20.3.7 read()	78
7.20.3.8 setSort()	78
7.20.3.9 setWild()	78
7.21 BDuration Class Reference	78
7.21.1 Constructor & Destructor Documentation	79
7.21.1.1 BDuration() [1/2]	79
7.21.1.2 BDuration() [2/2]	79
7.21.1.3 ~BDuration()	80
7.21.2 Member Function Documentation	80
7.21.2.1 addMicroSeconds()	80
7.21.2.2 addMilliSeconds()	80
7.21.2.3 addSeconds()	80
7.21.2.4 clear()	80
7.21.2.5 getMicroSeconds()	80
7.21.2.6 getSeconds()	81
7.21.2.7 getString()	81
7.21.2.8 hour()	81
7.21.2.9 microSecond()	81
7.21.2.10 minute()	81
7.21.2.11 second()	81
7.21.2.12 set()	81
7.21.2.13 setString()	82
7.22 BEntry Class Reference	82

7.22.1 Detailed Description	82
7.22.2 Constructor & Destructor Documentation	82
7.22.2.1 BEntry() [1/3]	83
7.22.2.2 BEntry() [2/3]	83
7.22.2.3 BEntry() [3/3]	83
7.22.3 Member Function Documentation	83
7.22.3.1 getName()	83
7.22.3.2 getValue()	83
7.22.3.3 line()	83
7.22.3.4 print()	84
7.22.3.5 setLine()	84
7.22.3.6 setName()	84
7.22.3.7 setValue()	84
7.23 BEntryFile Class Reference	84
7.23.1 Detailed Description	85
7.23.2 Constructor & Destructor Documentation	85
7.23.2.1 BEntryFile() [1/2]	85
7.23.2.2 BEntryFile() [2/2]	85
7.23.2.3 ~BEntryFile()	85
7.23.3 Member Function Documentation	86
7.23.3.1 clear()	86
7.23.3.2 filename()	86
7.23.3.3 open()	86
7.23.3.4 read()	86
7.23.3.5 write()	86
7.23.3.6 writeList()	87
7.24 BEntryList Class Reference	87
7.24.1 Detailed Description	88
7.24.2 Constructor & Destructor Documentation	88
7.24.2.1 BEntryList()	88
7.24.3 Member Function Documentation	88
7.24.3.1 clear()	88
7.24.3.2 del()	88
7.24.3.3 deleteEntry()	88
7.24.3.4 find()	89
7.24.3.5 findValue()	89
7.24.3.6 getString()	89
7.24.3.7 insert()	89
7.24.3.8 isSet()	89
7.24.3.9 operator=()	90
7.24.3.10 print()	90
7.24.3.11 setValue()	90

7.24.3.12 setValueRaw()	90
7.25 BError Class Reference	90
7.25.1 Detailed Description	91
7.25.2 Constructor & Destructor Documentation	91
7.25.2.1 BError() [1/2]	91
7.25.2.2 BError() [2/2]	92
7.25.3 Member Function Documentation	92
7.25.3.1 clear()	92
7.25.3.2 copy()	92
7.25.3.3 getErrorNo()	92
7.25.3.4 getNumber()	92
7.25.3.5 getString()	92
7.25.3.6 num()	93
7.25.3.7 operator int()	93
7.25.3.8 set()	93
7.25.3.9 setError()	93
7.25.3.10 str()	93
7.26 BErrorTime Class Reference	94
7.26.1 Detailed Description	94
7.26.2 Member Enumeration Documentation	94
7.26.2.1 Type	94
7.26.3 Constructor & Destructor Documentation	95
7.26.3.1 BErrorTime()	95
7.26.4 Member Function Documentation	95
7.26.4.1 clear()	95
7.26.4.2 copy()	95
7.26.4.3 getErrorNo()	95
7.26.4.4 getString()	95
7.26.4.5 getTime()	96
7.26.4.6 operator int()	96
7.26.4.7 set()	96
7.27 BEvent Class Reference	96
7.27.1 Constructor & Destructor Documentation	96
7.27.1.1 BEvent()	97
7.27.2 Member Function Documentation	97
7.27.2.1 arg()	97
7.27.2.2 type()	97
7.28 BEvent1 Class Reference	97
7.28.1 Detailed Description	98
7.28.2 Constructor & Destructor Documentation	98
7.28.2.1 BEvent1()	98
7.28.2.2 ~BEvent1()	98

7.28.3 Member Function Documentation	98
7.28.3.1 getBinary()	98
7.28.3.2 getType()	98
7.28.3.3 setBinary()	99
7.29 BEvent1Error Class Reference	99
7.29.1 Constructor & Destructor Documentation	99
7.29.1.1 BEvent1Error()	99
7.29.2 Member Function Documentation	99
7.29.2.1 getBinary()	100
7.29.2.2 setBinary()	100
7.30 BEvent1Int Class Reference	100
7.30.1 Detailed Description	100
7.30.2 Constructor & Destructor Documentation	101
7.30.2.1 BEvent1Int()	101
7.30.2.2 ~BEvent1Int()	101
7.30.3 Member Function Documentation	101
7.30.3.1 clear()	101
7.30.3.2 getEvent()	101
7.30.3.3 getFd()	101
7.30.3.4 sendEvent()	102
7.31 BEvent1Pipe Class Reference	102
7.31.1 Detailed Description	102
7.31.2 Constructor & Destructor Documentation	102
7.31.2.1 BEvent1Pipe()	102
7.31.2.2 ~BEvent1Pipe()	103
7.31.3 Member Function Documentation	103
7.31.3.1 clear()	103
7.31.3.2 getEvent()	103
7.31.3.3 getReceiveFd()	103
7.31.3.4 sendEvent()	103
7.32 BEventPipe Class Reference	104
7.32.1 Detailed Description	104
7.32.2 Constructor & Destructor Documentation	104
7.32.2.1 BEventPipe()	104
7.32.2.2 ~BEventPipe()	104
7.32.3 Member Function Documentation	104
7.32.3.1 clear()	105
7.32.3.2 getFd()	105
7.32.3.3 read()	105
7.32.3.4 readAvailable()	105
7.32.3.5 write()	105
7.32.3.6 writeAvailable()	105

7.33 Bfifo< Type > Class Template Reference	106
7.33.1 Constructor & Destructor Documentation	107
7.33.1.1 Bfifo()	107
7.33.1.2 ~Bfifo()	107
7.33.2 Member Function Documentation	107
7.33.2.1 clear()	107
7.33.2.2 operator[]()	107
7.33.2.3 read() [1/2]	108
7.33.2.4 read() [2/2]	108
7.33.2.5 readAvailable()	108
7.33.2.6 readAvailableChunk()	108
7.33.2.7 readData() [1/2]	108
7.33.2.8 readData() [2/2]	109
7.33.2.9 readDone()	109
7.33.2.10 readPos()	109
7.33.2.11 rebase()	109
7.33.2.12 resize()	109
7.33.2.13 size()	110
7.33.2.14 write() [1/2]	110
7.33.2.15 write() [2/2]	110
7.33.2.16 writeAvailable()	110
7.33.2.17 writeAvailableChunk()	110
7.33.2.18 writeBackup()	111
7.33.2.19 writeData() [1/2]	111
7.33.2.20 writeData() [2/2]	111
7.33.2.21 writeDone()	111
7.33.2.22 writePos()	111
7.33.3 Member Data Documentation	112
7.33.3.1 odata	112
7.33.3.2 oreadPos	112
7.33.3.3 osize	112
7.33.3.4 owritePos	112
7.34 BfifoCirc< Type > Class Template Reference	112
7.34.1 Detailed Description	114
7.34.2 Member Enumeration Documentation	114
7.34.2.1 anonymous enum	114
7.34.3 Constructor & Destructor Documentation	114
7.34.3.1 BfifoCirc()	114
7.34.3.2 ~BfifoCirc()	114
7.34.4 Member Function Documentation	114
7.34.4.1 clear()	115
7.34.4.2 mapCircularBuffer()	115

7.34.4.3 operator[]()	115
7.34.4.4 read()	115
7.34.4.5 readAvailable()	115
7.34.4.6 readData()	115
7.34.4.7 readDone()	116
7.34.4.8 readWaitAvailable()	116
7.34.4.9 size()	116
7.34.4.10 unmapCircularBuffer()	116
7.34.4.11 write()	116
7.34.4.12 writeAvailable()	117
7.34.4.13 writeData()	117
7.34.4.14 writeDone()	117
7.34.4.15 writeWaitAvailable()	117
7.34.5 Member Data Documentation	117
7.34.5.1 odata	117
7.34.5.2 olock	118
7.34.5.3 oreadPos	118
7.34.5.4 osize	118
7.34.5.5 ovmSize	118
7.34.5.6 owriteNumFifoSamples	118
7.34.5.7 owritePos	118
7.35 BFifoCircPos Class Reference	119
7.35.1 Detailed Description	119
7.35.2 Constructor & Destructor Documentation	119
7.35.2.1 BFifoCircPos()	119
7.35.3 Member Function Documentation	119
7.35.3.1 difference()	119
7.35.3.2 increment()	120
7.35.3.3 operator int()	120
7.35.3.4 operator"!=()	120
7.35.3.5 operator+=()	120
7.35.3.6 operator==()	120
7.35.3.7 pos()	120
7.35.3.8 set()	121
7.35.3.9 setSize()	121
7.36 BFile Class Reference	121
7.36.1 Detailed Description	122
7.36.2 Constructor & Destructor Documentation	122
7.36.2.1 BFile() [1/2]	122
7.36.2.2 BFile() [2/2]	123
7.36.2.3 ~BFile()	123
7.36.3 Member Function Documentation	123



7.36.3.1 close()	123
7.36.3.2 fgets()	123
7.36.3.3 fileName()	123
7.36.3.4 flush()	123
7.36.3.5 getFd()	124
7.36.3.6 isEnd()	124
7.36.3.7 isOpen()	124
7.36.3.8 length()	124
7.36.3.9 open() [1/3]	124
7.36.3.10 open() [2/3]	124
7.36.3.11 open() [3/3]	125
7.36.3.12 operator=()	125
7.36.3.13 position()	125
7.36.3.14 printf()	125
7.36.3.15 read()	125
7.36.3.16 readString()	125
7.36.3.17 seek()	126
7.36.3.18 setVBuf()	126
7.36.3.19 truncate()	126
7.36.3.20 write()	126
7.36.3.21 writeString()	126
7.37 BFileCsv Class Reference	127
7.37.1 Constructor & Destructor Documentation	127
7.37.1.1 BFileCsv()	127
7.37.2 Member Function Documentation	127
7.37.2.1 readCsv()	127
7.37.2.2 writeCsv()	127
7.38 BFileData Class Reference	128
7.38.1 Member Function Documentation	128
7.38.1.1 del()	128
7.38.1.2 find()	128
7.38.1.3 getNextId()	128
7.38.1.4 open()	129
7.38.1.5 write()	129
7.39 BFirmwareFileHeader Struct Reference	129
7.39.1 Member Data Documentation	129
7.39.1.1 checksum	129
7.39.1.2 fileLength	130
7.39.1.3 format	130
7.39.1.4 itemType	130
7.39.1.5 magic	130
7.39.1.6 numSegments	130

7.39.1.7 platform	130
7.39.1.8 special	130
7.39.1.9 startAddress	130
7.39.1.10 ver0	131
7.39.1.11 ver1	131
7.39.1.12 ver2	131
7.39.1.13 ver3	131
7.40 BFirmwareInfo Struct Reference	131
7.40.1 Member Data Documentation	131
7.40.1.1 checksum	132
7.40.1.2 length	132
7.40.1.3 magic	132
7.40.1.4 type	132
7.40.1.5 ver0	132
7.40.1.6 ver1	132
7.40.1.7 ver2	132
7.41 BFirmwareSegHeader Struct Reference	133
7.41.1 Member Data Documentation	133
7.41.1.1 address	133
7.41.1.2 checksum	133
7.41.1.3 dataLength	133
7.41.1.4 fileLength	133
7.41.1.5 format	134
7.41.1.6 itemType	134
7.41.1.7 length	134
7.41.1.8 magic	134
7.41.1.9 platform	134
7.41.1.10 special	134
7.42 Blter Class Reference	134
7.42.1 Detailed Description	135
7.42.2 Constructor & Destructor Documentation	135
7.42.2.1 Blter()	135
7.42.3 Member Function Documentation	135
7.42.3.1 operator BNode *()	135
7.42.3.2 operator==( )	135
7.42.3.3 valid()	135
7.43 BList< T > Class Template Reference	136
7.43.1 Detailed Description	138
7.43.2 Member Typedef Documentation	138
7.43.2.1 SortFunc	138
7.43.3 Constructor & Destructor Documentation	138
7.43.3.1 BList() [1/2]	138

7.43.3.2 BList() [2/2]	138
7.43.3.3 ~BList()	139
7.43.4 Member Function Documentation	139
7.43.4.1 append() [1/2]	139
7.43.4.2 append() [2/2]	139
7.43.4.3 begin()	139
7.43.4.4 clear()	139
7.43.4.5 del()	140
7.43.4.6 deleteFirst()	140
7.43.4.7 deleteLast()	140
7.43.4.8 end() [1/2]	140
7.43.4.9 end() [2/2]	140
7.43.4.10 front()	141
7.43.4.11 get() [1/2]	141
7.43.4.12 get() [2/2]	141
7.43.4.13 goTo()	141
7.43.4.14 has()	141
7.43.4.15 insert()	142
7.43.4.16 insertAfter()	142
7.43.4.17 isEnd()	142
7.43.4.18 isStart()	142
7.43.4.19 next()	142
7.43.4.20 nodeCreate()	143
7.43.4.21 nodeGet() [1/2]	143
7.43.4.22 nodeGet() [2/2]	143
7.43.4.23 number()	143
7.43.4.24 operator+()	143
7.43.4.25 operator=()	143
7.43.4.26 operator[]() [1/4]	144
7.43.4.27 operator[]() [2/4]	144
7.43.4.28 operator[]() [3/4]	144
7.43.4.29 operator[]() [4/4]	144
7.43.4.30 pop()	144
7.43.4.31 position()	144
7.43.4.32 prev()	145
7.43.4.33 push()	145
7.43.4.34 queueAdd()	145
7.43.4.35 queueGet()	145
7.43.4.36 rear()	145
7.43.4.37 size()	146
7.43.4.38 sort() [1/2]	146
7.43.4.39 sort() [2/2]	146

7.43.4.40 start()	146
7.43.4.41 swap()	146
7.43.5 Member Data Documentation	146
7.43.5.1 olength	147
7.43.5.2 onodes	147
7.44 BMutex Class Reference	147
7.44.1 Detailed Description	148
7.44.2 Member Enumeration Documentation	148
7.44.2.1 Type	148
7.44.3 Constructor & Destructor Documentation	148
7.44.3.1 BMutex() [1/2]	148
7.44.3.2 BMutex() [2/2]	148
7.44.3.3 ~BMutex()	148
7.44.4 Member Function Documentation	148
7.44.4.1 lock()	149
7.44.4.2 operator=()	149
7.44.4.3 timedLock()	149
7.44.4.4 tryLock()	149
7.44.4.5 unlock()	149
7.45 BMutexLock Class Reference	149
7.45.1 Constructor & Destructor Documentation	150
7.45.1.1 BMutexLock()	150
7.45.1.2 ~BMutexLock()	150
7.45.2 Member Function Documentation	150
7.45.2.1 lock()	150
7.45.2.2 unlock()	150
7.46 BMySQL Class Reference	150
7.46.1 Constructor & Destructor Documentation	151
7.46.1.1 BMySQL()	151
7.46.1.2 ~BMySQL()	151
7.46.2 Member Function Documentation	151
7.46.2.1 close()	151
7.46.2.2 db()	151
7.46.2.3 del()	152
7.46.2.4 escapeString()	152
7.46.2.5 flush()	152
7.46.2.6 get()	152
7.46.2.7 insert()	152
7.46.2.8 open()	153
7.46.2.9 query()	153
7.46.2.10 setDebug()	153
7.46.2.11 update()	153

7.47 BNameValue< T > Class Template Reference . . . . .	153
7.47.1 Constructor & Destructor Documentation . . . . .	154
7.47.1.1 BNameValue() [1/2] . . . . .	154
7.47.1.2 BNameValue() [2/2] . . . . .	154
7.47.2 Member Function Documentation . . . . .	154
7.47.2.1 getName() . . . . .	154
7.47.2.2 getValue() . . . . .	154
7.48 BNameValueList< T > Class Template Reference . . . . .	154
7.48.1 Member Function Documentation . . . . .	155
7.48.1.1 find() . . . . .	155
7.48.1.2 findPos() . . . . .	155
7.49 BNode Class Reference . . . . .	155
7.49.1 Constructor & Destructor Documentation . . . . .	156
7.49.1.1 BNode() . . . . .	156
7.49.2 Member Data Documentation . . . . .	156
7.49.2.1 next . . . . .	156
7.49.2.2 prev . . . . .	156
7.50 BoapClientObject Class Reference . . . . .	156
7.50.1 Constructor & Destructor Documentation . . . . .	158
7.50.1.1 BoapClientObject() [1/2] . . . . .	158
7.50.1.2 ~BoapClientObject() . . . . .	158
7.50.1.3 BoapClientObject() [2/2] . . . . .	158
7.50.2 Member Function Documentation . . . . .	158
7.50.2.1 checkApiVersion() . . . . .	158
7.50.2.2 connectService() [1/2] . . . . .	158
7.50.2.3 connectService() [2/2] . . . . .	159
7.50.2.4 disconnectService() . . . . .	159
7.50.2.5 getServiceName() . . . . .	159
7.50.2.6 handleReconnect() . . . . .	159
7.50.2.7 performCall() [1/2] . . . . .	159
7.50.2.8 performCall() [2/2] . . . . .	159
7.50.2.9 performRecv() [1/2] . . . . .	160
7.50.2.10 performRecv() [2/2] . . . . .	160
7.50.2.11 performSend() [1/2] . . . . .	160
7.50.2.12 performSend() [2/2] . . . . .	160
7.50.2.13 ping() . . . . .	160
7.50.2.14 pingLocked() . . . . .	160
7.50.2.15 setConnectionPriority() . . . . .	161
7.50.2.16 setMaxLength() . . . . .	161
7.50.2.17 setTimeout() . . . . .	161
7.50.3 Member Data Documentation . . . . .	161
7.50.3.1 oapiVersion . . . . .	161

7.50.3.2 oconnected	161
7.50.3.3 olock	161
7.50.3.4 omaxLength	162
7.50.3.5 oname	162
7.50.3.6 opriority	162
7.50.3.7 oreconnect	162
7.50.3.8 orx	162
7.50.3.9 oservice	162
7.50.3.10 otimeout	162
7.50.3.11 otx	163
7.51 Boapns::BoapEntry Class Reference	163
7.51.1 Constructor & Destructor Documentation	163
7.51.1.1 BoapEntry()	163
7.51.2 Member Data Documentation	163
7.51.2.1 addressList	164
7.51.2.2 hostName	164
7.51.2.3 name	164
7.51.2.4 port	164
7.51.2.5 service	164
7.52 BoapFuncEntry Class Reference	164
7.52.1 Constructor & Destructor Documentation	165
7.52.1.1 BoapFuncEntry() [1/2]	165
7.52.1.2 BoapFuncEntry() [2/2]	165
7.52.2 Member Data Documentation	165
7.52.2.1 ocmd [1/2]	165
7.52.2.2 ocmd [2/2]	165
7.52.2.3 ofunc	165
7.53 BoapMc1Comms Class Reference	166
7.53.1 Constructor & Destructor Documentation	167
7.53.1.1 BoapMc1Comms()	167
7.53.1.2 ~BoapMc1Comms()	167
7.53.2 Member Function Documentation	167
7.53.2.1 getApiVersion()	167
7.53.2.2 packetRx()	168
7.53.2.3 packetRxData()	168
7.53.2.4 packetRxEnd()	168
7.53.2.5 packetTx()	168
7.53.2.6 processRequest()	168
7.53.2.7 processRequests()	168
7.53.2.8 processRx()	169
7.53.2.9 setAddress()	169
7.53.2.10 setComms() [1/2]	169

7.53.2.11 setComms() [2/2]	169
7.53.2.12 setCommsMode()	169
7.53.2.13 setTimeout()	169
7.53.2.14 validate()	170
7.53.3 Member Data Documentation	170
7.53.3.1 oaddressFrom	170
7.53.3.2 oaddressTo	170
7.53.3.3 oapiVersion	170
7.53.3.4 ocomms	170
7.53.3.5 oerror	170
7.53.3.6 ohalfDuplex	171
7.53.3.7 olockCall	171
7.53.3.8 olockTx	171
7.53.3.9 opacketRpcCmd	171
7.53.3.10 opacketRpcDoneSema	171
7.53.3.11 opacketRpcSema	171
7.53.3.12 opacketRx	172
7.53.3.13 opacketRxBase	172
7.53.3.14 opacketTx	172
7.53.3.15 opacketTxBase	172
7.53.3.16 oreqSize	172
7.53.3.17 othreaded	172
7.53.3.18 otimeout	173
7.54 BoapMc1Error Struct Reference	173
7.54.1 Member Data Documentation	173
7.54.1.1 number	173
7.54.1.2 string	173
7.55 BoapMc1Packet Class Reference	174
7.55.1 Member Data Documentation	174
7.55.1.1 data	174
7.55.1.2 head	174
7.56 BoapMc1PacketHead Struct Reference	174
7.56.1 Member Data Documentation	175
7.56.1.1 addressFrom	175
7.56.1.2 addressTo	175
7.56.1.3 checksum	175
7.56.1.4 cmd	175
7.56.1.5 error	175
7.56.1.6 length	176
7.56.1.7 magic	176
7.57 BoapMcClientObject Class Reference	176
7.57.1 Constructor & Destructor Documentation	177

7.57.1.1 BoapMcClientObject()	177
7.57.1.2 ~BoapMcClientObject()	177
7.57.2 Member Function Documentation	177
7.57.2.1 getApiVersion()	177
7.57.2.2 performCall()	177
7.57.2.3 performRecv()	177
7.57.2.4 performSend()	178
7.57.2.5 setAddress()	178
7.57.3 Member Data Documentation	178
7.57.3.1 oaddressFrom	178
7.57.3.2 oaddressTo	178
7.57.3.3 oapiVersion	178
7.57.3.4 ocomms	178
7.57.3.5 opacket	179
7.58 BoapMcComms Class Reference	179
7.58.1 Constructor & Destructor Documentation	180
7.58.1.1 BoapMcComms()	180
7.58.1.2 ~BoapMcComms()	181
7.58.2 Member Function Documentation	181
7.58.2.1 getApiVersion()	181
7.58.2.2 packetRecv()	181
7.58.2.3 packetSend()	181
7.58.2.4 performCall()	181
7.58.2.5 performSend()	181
7.58.2.6 processPacket()	182
7.58.2.7 processRequest()	182
7.58.2.8 processRequests()	182
7.58.2.9 processRx()	182
7.58.2.10 setAddress()	182
7.58.2.11 setComms() [1/2]	183
7.58.2.12 setComms() [2/2]	183
7.58.2.13 setCommsMode()	183
7.58.2.14 setTimeout()	183
7.58.3 Member Data Documentation	183
7.58.3.1 oaddressFrom	183
7.58.3.2 oaddressTo	184
7.58.3.3 oapiVersion	184
7.58.3.4 ocomms	184
7.58.3.5 olockCall	184
7.58.3.6 olockTx	184
7.58.3.7 opacket	184
7.58.3.8 opacketReqQueue	184



7.58.3.9 opacketReqRx . . . . .	185
7.58.3.10 opacketReqTx . . . . .	185
7.58.3.11 opacketRx . . . . .	185
7.58.3.12 opacketRxSema . . . . .	185
7.58.3.13 opacketTx . . . . .	185
7.58.3.14 opacketTxQueue . . . . .	185
7.58.3.15 opacketTxQueueWriteNum . . . . .	186
7.58.3.16 opacketTxSema . . . . .	186
7.58.3.17 oslave . . . . .	186
7.58.3.18 othreaded . . . . .	186
7.58.3.19 otimeout . . . . .	186
7.59 BoapMcPacket Class Reference . . . . .	186
7.59.1 Member Data Documentation . . . . .	187
7.59.1.1 data . . . . .	187
7.59.1.2 head . . . . .	187
7.60 BoapMcPacketHead Struct Reference . . . . .	187
7.60.1 Member Data Documentation . . . . .	187
7.60.1.1 addressFrom . . . . .	187
7.60.1.2 addressTo . . . . .	188
7.60.1.3 checksum . . . . .	188
7.60.1.4 cmd . . . . .	188
7.60.1.5 error . . . . .	188
7.60.1.6 length . . . . .	188
7.61 BoapMcServiceObject Class Reference . . . . .	188
7.61.1 Constructor & Destructor Documentation . . . . .	189
7.61.1.1 BoapMcServiceObject() . . . . .	189
7.61.1.2 ~BoapMcServiceObject() . . . . .	189
7.61.2 Member Function Documentation . . . . .	189
7.61.2.1 process() . . . . .	189
7.61.2.2 processEvent() . . . . .	189
7.61.2.3 sendEvent() . . . . .	189
7.61.3 Member Data Documentation . . . . .	190
7.61.3.1 oapiVersion . . . . .	190
7.62 BoapMcSignalObject Class Reference . . . . .	190
7.62.1 Constructor & Destructor Documentation . . . . .	190
7.62.1.1 BoapMcSignalObject() . . . . .	190
7.62.2 Member Function Documentation . . . . .	190
7.62.2.1 performSend() . . . . .	191
7.62.3 Member Data Documentation . . . . .	191
7.62.3.1 ocomms . . . . .	191
7.63 Boapns::Boapns Class Reference . . . . .	191
7.63.1 Constructor & Destructor Documentation . . . . .	191

7.63.1.1 Boapns()	192
7.63.2 Member Function Documentation	192
7.63.2.1 addEntry()	192
7.63.2.2 delEntry()	192
7.63.2.3 getEntry()	192
7.63.2.4 getEntryList()	192
7.63.2.5 getNewName()	192
7.63.2.6 getVersion()	193
7.64 BoapPacket Class Reference	193
7.64.1 Constructor & Destructor Documentation	194
7.64.1.1 BoapPacket() [1/2]	194
7.64.1.2 ~BoapPacket() [1/2]	194
7.64.1.3 BoapPacket() [2/2]	194
7.64.1.4 ~BoapPacket() [2/2]	194
7.64.2 Member Function Documentation	194
7.64.2.1 data()	195
7.64.2.2 getCmd()	195
7.64.2.3 nbytes()	195
7.64.2.4 peekHead()	195
7.64.2.5 pop() [1/10]	195
7.64.2.6 pop() [2/10]	195
7.64.2.7 pop() [3/10]	195
7.64.2.8 pop() [4/10]	196
7.64.2.9 pop() [5/10]	196
7.64.2.10 pop() [6/10]	196
7.64.2.11 pop() [7/10]	196
7.64.2.12 pop() [8/10]	196
7.64.2.13 pop() [9/10]	196
7.64.2.14 pop() [10/10]	196
7.64.2.15 popHead() [1/2]	197
7.64.2.16 popHead() [2/2]	197
7.64.2.17 push() [1/10]	197
7.64.2.18 push() [2/10]	197
7.64.2.19 push() [3/10]	197
7.64.2.20 push() [4/10]	197
7.64.2.21 push() [5/10]	197
7.64.2.22 push() [6/10]	198
7.64.2.23 push() [7/10]	198
7.64.2.24 push() [8/10]	198
7.64.2.25 push() [9/10]	198
7.64.2.26 push() [10/10]	198
7.64.2.27 pushHead() [1/2]	198

7.64.2.28 pushHead() [2/2]	198
7.64.2.29 resize()	199
7.64.2.30 setData()	199
7.64.2.31 updateHead()	199
7.65 BoapPacketHead Struct Reference	199
7.65.1 Member Data Documentation	199
7.65.1.1 cmd [1/2]	200
7.65.1.2 cmd [2/2]	200
7.65.1.3 length [1/2]	200
7.65.1.4 length [2/2]	200
7.65.1.5 reserved	200
7.65.1.6 service [1/2]	200
7.65.1.7 service [2/2]	200
7.65.1.8 type [1/2]	200
7.65.1.9 type [2/2]	201
7.66 BoapServer Class Reference	201
7.66.1 Member Enumeration Documentation	202
7.66.1.1 anonymous enum	202
7.66.2 Constructor & Destructor Documentation	202
7.66.2.1 BoapServer() [1/2]	202
7.66.2.2 ~BoapServer()	202
7.66.2.3 BoapServer() [2/2]	202
7.66.3 Member Function Documentation	202
7.66.3.1 addObject() [1/2]	203
7.66.3.2 addObject() [2/2]	203
7.66.3.3 clientGone()	203
7.66.3.4 closeConnections()	203
7.66.3.5 getConnectionsNumber()	203
7.66.3.6 getEventSocket() [1/2]	203
7.66.3.7 getEventSocket() [2/2]	203
7.66.3.8 getHostName() [1/2]	204
7.66.3.9 getHostName() [2/2]	204
7.66.3.10 getSocket() [1/2]	204
7.66.3.11 getSocket() [2/2]	204
7.66.3.12 init() [1/2]	204
7.66.3.13 init() [2/2]	204
7.66.3.14 newConnection()	204
7.66.3.15 process() [1/2]	205
7.66.3.16 process() [2/2]	205
7.66.3.17 processEvent() [1/4]	205
7.66.3.18 processEvent() [2/4]	205
7.66.3.19 processEvent() [3/4]	205

7.66.3.20 processEvent() [4/4]	205
7.66.3.21 run() [1/2]	205
7.66.3.22 run() [2/2]	206
7.66.3.23 sendEvent() [1/2]	206
7.66.3.24 sendEvent() [2/2]	206
7.66.4 Member Data Documentation	206
7.66.4.1 onumOperations	206
7.67 BoapServerConnection Class Reference	206
7.67.1 Constructor & Destructor Documentation	207
7.67.1.1 BoapServerConnection()	207
7.67.1.2 ~BoapServerConnection()	207
7.67.2 Member Function Documentation	207
7.67.2.1 getHead()	207
7.67.2.2 getSocket()	207
7.67.2.3 init()	208
7.67.2.4 process()	208
7.67.2.5 setMaxLength()	208
7.67.2.6 validate()	208
7.68 BoapServiceEntry Class Reference	208
7.68.1 Constructor & Destructor Documentation	209
7.68.1.1 BoapServiceEntry() [1/2]	209
7.68.1.2 BoapServiceEntry() [2/2]	209
7.68.2 Member Data Documentation	209
7.68.2.1 oobject	209
7.68.2.2 oservice	209
7.69 BoapServiceObject Class Reference	209
7.69.1 Constructor & Destructor Documentation	210
7.69.1.1 BoapServiceObject() [1/2]	210
7.69.1.2 ~BoapServiceObject() [1/2]	210
7.69.1.3 BoapServiceObject() [2/2]	211
7.69.1.4 ~BoapServiceObject() [2/2]	211
7.69.2 Member Function Documentation	211
7.69.2.1 doConnectionPriority()	211
7.69.2.2 doPing()	211
7.69.2.3 name() [1/2]	211
7.69.2.4 name() [2/2]	211
7.69.2.5 process() [1/2]	212
7.69.2.6 process() [2/2]	212
7.69.2.7 processEvent() [1/4]	212
7.69.2.8 processEvent() [2/4]	212
7.69.2.9 processEvent() [3/4]	212
7.69.2.10 processEvent() [4/4]	212

7.69.2.11 sendEvent() [1/4]	213
7.69.2.12 sendEvent() [2/4]	213
7.69.2.13 sendEvent() [3/4]	213
7.69.2.14 sendEvent() [4/4]	213
7.69.2.15 setName()	213
7.69.3 Member Data Documentation	213
7.69.3.1 oapiVersion	213
7.69.3.2 ofuncList	214
7.69.3.3 oname	214
7.69.3.4 oserver	214
7.70 BoapSignalObject Class Reference	214
7.70.1 Constructor & Destructor Documentation	215
7.70.1.1 BoapSignalObject() [1/2]	215
7.70.1.2 BoapSignalObject() [2/2]	215
7.70.2 Member Function Documentation	215
7.70.2.1 performSend() [1/2]	215
7.70.2.2 performSend() [2/2]	215
7.70.3 Member Data Documentation	215
7.70.3.1 orx	216
7.70.3.2 otx	216
7.71 BObj Class Reference	216
7.71.1 Constructor & Destructor Documentation	216
7.71.1.1 BObj()	216
7.71.1.2 ~BObj()	217
7.71.2 Member Function Documentation	217
7.71.2.1 getDebugString()	217
7.71.2.2 getMember()	217
7.71.2.3 getMembers() [1/2]	217
7.71.2.4 getMembers() [2/2]	217
7.71.2.5 getType()	217
7.71.2.6 membersPrint()	218
7.71.2.7 setMember()	218
7.71.2.8 setMembers()	218
7.72 BObjMember Struct Reference	218
7.72.1 Member Data Documentation	218
7.72.1.1 dataOffset	218
7.72.1.2 name	219
7.72.1.3 size	219
7.72.1.4 type	219
7.72.1.5 typeComp	219
7.72.1.6 typeName	219
7.73 BPoll Class Reference	219

7.73.1 Detailed Description	220
7.73.2 Member Typedef Documentation	220
7.73.2.1 PollFd	220
7.73.3 Constructor & Destructor Documentation	220
7.73.3.1 BPoll()	220
7.73.3.2 ~BPoll()	220
7.73.4 Member Function Documentation	220
7.73.4.1 append()	221
7.73.4.2 clear()	221
7.73.4.3 delFd()	221
7.73.4.4 doPoll()	221
7.73.4.5 doPollEvents()	221
7.73.4.6 getPollFds()	221
7.73.4.7 getPollFdsNum()	222
7.74 BQueue< T > Class Template Reference	222
7.74.1 Detailed Description	222
7.74.2 Constructor & Destructor Documentation	222
7.74.2.1 BQueue()	223
7.74.2.2 ~BQueue()	223
7.74.3 Member Function Documentation	223
7.74.3.1 clear()	223
7.74.3.2 read()	223
7.74.3.3 readAvailable()	223
7.74.3.4 write()	224
7.74.3.5 writeAvailable()	224
7.75 BRefData Class Reference	224
7.75.1 Detailed Description	224
7.75.2 Constructor & Destructor Documentation	225
7.75.2.1 BRefData() [1/3]	225
7.75.2.2 BRefData() [2/3]	225
7.75.2.3 BRefData() [3/3]	225
7.75.2.4 ~BRefData()	225
7.75.3 Member Function Documentation	225
7.75.3.1 addRef()	225
7.75.3.2 copy()	225
7.75.3.3 data()	226
7.75.3.4 deleteRef()	226
7.75.3.5 len()	226
7.75.3.6 operator=()	226
7.75.3.7 setLen()	226
7.76 BRtc Class Reference	226
7.76.1 Detailed Description	227

7.76.2 Constructor & Destructor Documentation	227
7.76.2.1 BRtc()	227
7.76.2.2 ~BRtc()	227
7.76.3 Member Function Documentation	227
7.76.3.1 init()	227
7.76.3.2 wait()	228
7.77 BRtcThreaded Class Reference	228
7.77.1 Detailed Description	228
7.77.2 Constructor & Destructor Documentation	228
7.77.2.1 BRtcThreaded()	228
7.77.2.2 ~BRtcThreaded()	229
7.77.3 Member Function Documentation	229
7.77.3.1 init()	229
7.77.3.2 wait()	229
7.78 BRWLock Class Reference	229
7.78.1 Detailed Description	230
7.78.2 Constructor & Destructor Documentation	230
7.78.2.1 BRWLock() [1/2]	230
7.78.2.2 BRWLock() [2/2]	230
7.78.2.3 ~BRWLock()	230
7.78.3 Member Function Documentation	230
7.78.3.1 operator=()	230
7.78.3.2 rdLock()	230
7.78.3.3 tryRdLock()	231
7.78.3.4 tryWrLock()	231
7.78.3.5 unlock()	231
7.78.3.6 wrLock()	231
7.79 BSema Class Reference	231
7.79.1 Detailed Description	232
7.79.2 Constructor & Destructor Documentation	232
7.79.2.1 BSema() [1/2]	232
7.79.2.2 BSema() [2/2]	232
7.79.2.3 ~BSema()	232
7.79.3 Member Function Documentation	232
7.79.3.1 getValue()	233
7.79.3.2 operator=()	233
7.79.3.3 post()	233
7.79.3.4 timedWait()	233
7.79.3.5 tryWait()	233
7.79.3.6 wait()	233
7.80 BSemaphore Class Reference	234
7.80.1 Detailed Description	234

7.80.2 Constructor & Destructor Documentation	234
7.80.2.1 BSemaphore() [1/2]	234
7.80.2.2 BSemaphore() [2/2]	234
7.80.2.3 ~BSemaphore()	234
7.80.3 Member Function Documentation	235
7.80.3.1 getValue()	235
7.80.3.2 operator=()	235
7.80.3.3 set()	235
7.80.3.4 wait()	235
7.81 BSemaphoreBool Class Reference	235
7.81.1 Constructor & Destructor Documentation	236
7.81.1.1 BSemaphoreBool() [1/2]	236
7.81.1.2 BSemaphoreBool() [2/2]	236
7.81.1.3 ~BSemaphoreBool()	236
7.81.2 Member Function Documentation	236
7.81.2.1 clear()	236
7.81.2.2 operator int()	237
7.81.2.3 operator=()	237
7.81.2.4 operator==()	237
7.81.2.5 set()	237
7.81.2.6 value()	237
7.81.2.7 wait()	237
7.82 BSemaphoreCount Class Reference	238
7.82.1 Constructor & Destructor Documentation	238
7.82.1.1 BSemaphoreCount() [1/2]	238
7.82.1.2 BSemaphoreCount() [2/2]	238
7.82.1.3 ~BSemaphoreCount()	238
7.82.2 Member Function Documentation	238
7.82.2.1 add()	239
7.82.2.2 operator=()	239
7.82.2.3 setValue()	239
7.82.2.4 take()	239
7.82.2.5 value()	239
7.82.2.6 wait()	239
7.83 BSocket Class Reference	240
7.83.1 Member Enumeration Documentation	241
7.83.1.1 NType	241
7.83.1.2 Priority	241
7.83.2 Constructor & Destructor Documentation	241
7.83.2.1 BSocket() [1/4]	241
7.83.2.2 BSocket() [2/4]	241
7.83.2.3 BSocket() [3/4]	242



7.83.2.4 BSocket() [ 4 / 4 ] . . . . .	242
7.83.2.5 ~BSocket() . . . . .	242
7.83.3 Member Function Documentation . . . . .	242
7.83.3.1 accept() [ 1 / 2 ] . . . . .	242
7.83.3.2 accept() [ 2 / 2 ] . . . . .	242
7.83.3.3 bind() . . . . .	242
7.83.3.4 close() . . . . .	243
7.83.3.5 connect() . . . . .	243
7.83.3.6 getAddress() . . . . .	243
7.83.3.7 getFd() . . . . .	243
7.83.3.8 getMTU() . . . . .	243
7.83.3.9 getSockOpt() . . . . .	243
7.83.3.10 init() [ 1 / 2 ] . . . . .	243
7.83.3.11 init() [ 2 / 2 ] . . . . .	244
7.83.3.12 listen() . . . . .	244
7.83.3.13 recv() . . . . .	244
7.83.3.14 recvAvailable() . . . . .	244
7.83.3.15 recvFrom() . . . . .	244
7.83.3.16 recvFromWithTimeout() . . . . .	244
7.83.3.17 recvWithTimeout() . . . . .	245
7.83.3.18 send() . . . . .	245
7.83.3.19 sendChunks() . . . . .	245
7.83.3.20 sendTo() . . . . .	245
7.83.3.21 setBroadCast() . . . . .	245
7.83.3.22 setFd() . . . . .	246
7.83.3.23 setPriority() . . . . .	246
7.83.3.24 setReuseAddress() . . . . .	246
7.83.3.25 setSockOpt() . . . . .	246
7.83.3.26 shutdown() . . . . .	246
7.84 BSocketAddress Class Reference . . . . .	246
7.84.1 Detailed Description . . . . .	247
7.84.2 Member Typedef Documentation . . . . .	247
7.84.2.1 SockAddr . . . . .	247
7.84.3 Constructor & Destructor Documentation . . . . .	247
7.84.3.1 BSocketAddress() [ 1 / 3 ] . . . . .	247
7.84.3.2 BSocketAddress() [ 2 / 3 ] . . . . .	248
7.84.3.3 BSocketAddress() [ 3 / 3 ] . . . . .	248
7.84.3.4 ~BSocketAddress() . . . . .	248
7.84.4 Member Function Documentation . . . . .	248
7.84.4.1 getString() . . . . .	248
7.84.4.2 len() . . . . .	248
7.84.4.3 operator const SockAddr *() . . . . .	248

7.84.4.4 operator!=(())	249
7.84.4.5 operator=()	249
7.84.4.6 operator==(())	249
7.84.4.7 raw()	249
7.84.4.8 set()	249
7.85 BSocketAddressINET Class Reference	249
7.85.1 Detailed Description	250
7.85.2 Member Typedef Documentation	250
7.85.2.1 SockAddrIP	250
7.85.3 Member Function Documentation	250
7.85.3.1 address()	251
7.85.3.2 getHostName()	251
7.85.3.3 getIpAddresses()	251
7.85.3.4 getIpAddressList()	251
7.85.3.5 getIpAddressListAll()	251
7.85.3.6 getString()	251
7.85.3.7 port()	252
7.85.3.8 set() [1/3]	252
7.85.3.9 set() [2/3]	252
7.85.3.10 set() [3/3]	252
7.85.3.11 setPort()	252
7.86 BSpI Class Reference	252
7.86.1 Detailed Description	253
7.86.2 Member Enumeration Documentation	253
7.86.2.1 Mode	253
7.86.3 Constructor & Destructor Documentation	253
7.86.3.1 BSpI()	253
7.86.4 Member Function Documentation	253
7.86.4.1 init()	254
7.86.4.2 transact()	254
7.87 BString Class Reference	254
7.87.1 Constructor & Destructor Documentation	257
7.87.1.1 BString() [1/9]	257
7.87.1.2 BString() [2/9]	257
7.87.1.3 BString() [3/9]	258
7.87.1.4 BString() [4/9]	258
7.87.1.5 BString() [5/9]	258
7.87.1.6 BString() [6/9]	258
7.87.1.7 BString() [7/9]	258
7.87.1.8 BString() [8/9]	258
7.87.1.9 BString() [9/9]	258
7.87.1.10 ~BString()	259

7.87.2 Member Function Documentation	259
7.87.2.1 add()	259
7.87.2.2 append()	259
7.87.2.3 base64Decode()	259
7.87.2.4 base64Encode()	259
7.87.2.5 basename()	259
7.87.2.6 clear()	260
7.87.2.7 compare()	260
7.87.2.8 compareRegex()	260
7.87.2.9 compareWild()	260
7.87.2.10 compareWildExpression()	260
7.87.2.11 convert() [1/5]	260
7.87.2.12 convert() [2/5]	261
7.87.2.13 convert() [3/5]	261
7.87.2.14 convert() [4/5]	261
7.87.2.15 convert() [5/5]	261
7.87.2.16 convertHex() [1/2]	261
7.87.2.17 convertHex() [2/2]	262
7.87.2.18 copy()	262
7.87.2.19 csvDecode()	262
7.87.2.20 csvEncode()	262
7.87.2.21 del()	262
7.87.2.22 dirname()	262
7.87.2.23 extension()	263
7.87.2.24 field()	263
7.87.2.25 fields()	263
7.87.2.26 find() [1/2]	263
7.87.2.27 find() [2/2]	263
7.87.2.28 findReverse()	263
7.87.2.29 firstLine()	264
7.87.2.30 fixedLen()	264
7.87.2.31 get() [1/2]	264
7.87.2.32 get() [2/2]	264
7.87.2.33 getTokenList() [1/2]	264
7.87.2.34 getTokenList() [2/2]	264
7.87.2.35 hash()	265
7.87.2.36 insert()	265
7.87.2.37 justify()	265
7.87.2.38 len()	265
7.87.2.39 lowerFirst()	265
7.87.2.40 operator const char *()	265
7.87.2.41 operator"!="( [1/2]	266

7.87.2.42 operator"!=()	[ 2/2]	266
7.87.2.43 operator+()	[ 1/6]	266
7.87.2.44 operator+()	[ 2/6]	266
7.87.2.45 operator+()	[ 3/6]	266
7.87.2.46 operator+()	[ 4/6]	266
7.87.2.47 operator+()	[ 5/6]	266
7.87.2.48 operator+()	[ 6/6]	267
7.87.2.49 operator+=()	[ 1/2]	267
7.87.2.50 operator+=()	[ 2/2]	267
7.87.2.51 operator<()	[ 1/2]	267
7.87.2.52 operator<()	[ 2/2]	267
7.87.2.53 operator<=()		267
7.87.2.54 operator=()		267
7.87.2.55 operator==(())	[ 1/2]	268
7.87.2.56 operator==(())	[ 2/2]	268
7.87.2.57 operator>()	[ 1/2]	268
7.87.2.58 operator>()	[ 2/2]	268
7.87.2.59 operator>=()		268
7.87.2.60 operator[]()		268
7.87.2.61 pad()		268
7.87.2.62 printf()		269
7.87.2.63 pullLine()		269
7.87.2.64 pullSeparators()		269
7.87.2.65 pullToken()		269
7.87.2.66 pullWord()		269
7.87.2.67 removeNL()		269
7.87.2.68 removeSeparators()		270
7.87.2.69 retDouble()		270
7.87.2.70 retFloat64()		270
7.87.2.71 retInt()		270
7.87.2.72 retStr()		270
7.87.2.73 retStrDup()		270
7.87.2.74 retUInt()		271
7.87.2.75 reverse()		271
7.87.2.76 split()		271
7.87.2.77 str()		271
7.87.2.78 subString()		271
7.87.2.79 toLower()		271
7.87.2.80 toUpper()		272
7.87.2.81 translateChar()		272
7.87.2.82 truncate()		272
7.87.3 Member Data Documentation		272

7.87.3.1 ostr	272
7.88 BStringLocked Class Reference	272
7.88.1 Constructor & Destructor Documentation	273
7.88.1.1 BStringLocked() [1/3]	273
7.88.1.2 BStringLocked() [2/3]	273
7.88.1.3 BStringLocked() [3/3]	273
7.88.2 Member Function Documentation	273
7.88.2.1 len()	273
7.88.2.2 operator BString()	274
7.88.2.3 operator+()	274
7.88.2.4 operator=()	274
7.89 BStringMutex Class Reference	274
7.89.1 Constructor & Destructor Documentation	274
7.89.1.1 BStringMutex()	275
7.90 BTable Class Reference	275
7.90.1 Constructor & Destructor Documentation	275
7.90.1.1 BTable()	275
7.90.1.2 ~BTable()	275
7.90.2 Member Function Documentation	275
7.90.2.1 addRow()	275
7.90.2.2 clear()	276
7.90.2.3 print()	276
7.90.2.4 setTitle()	276
7.91 BTask Class Reference	276
7.91.1 Constructor & Destructor Documentation	277
7.91.1.1 BTask()	277
7.91.1.2 ~BTask()	277
7.91.2 Member Function Documentation	277
7.91.2.1 init()	277
7.91.2.2 run()	277
7.91.2.3 setPriority()	278
7.91.2.4 start()	278
7.91.2.5 stop()	278
7.91.2.6 taskFunc()	278
7.91.2.7 waitForCompletion()	278
7.91.3 Member Data Documentation	278
7.91.3.1 oname	278
7.91.3.2 opolicy	279
7.91.3.3 opriority	279
7.91.3.4 orunning	279
7.91.3.5 ostackSize	279
7.91.3.6 othread	279

7.92 BThread Class Reference	279
7.92.1 Constructor & Destructor Documentation	280
7.92.1.1 BThread()	280
7.92.1.2 ~BThread()	280
7.92.2 Member Function Documentation	280
7.92.2.1 cancel()	280
7.92.2.2 function()	280
7.92.2.3 getThread()	281
7.92.2.4 result()	281
7.92.2.5 running()	281
7.92.2.6 setInitPriority()	281
7.92.2.7 setInitStackSize()	281
7.92.2.8 setPriority()	281
7.92.2.9 start()	281
7.92.2.10 waitForCompletion()	282
7.93 BTime Class Reference	282
7.93.1 Constructor & Destructor Documentation	283
7.93.1.1 BTime()	283
7.93.2 Member Function Documentation	283
7.93.2.1 addSeconds()	283
7.93.2.2 getDate()	283
7.93.2.3 getSeconds()	283
7.93.2.4 getString()	283
7.93.2.5 getTime()	284
7.93.2.6 isLeapYear()	284
7.93.2.7 isSet()	284
7.93.2.8 operator"!="()	284
7.93.2.9 operator+()	284
7.93.2.10 operator+=()	284
7.93.2.11 operator<()	285
7.93.2.12 operator<=()	285
7.93.2.13 operator==()	285
7.93.2.14 operator>()	285
7.93.2.15 operator>=()	285
7.93.2.16 set() [1/2]	285
7.93.2.17 set() [2/2]	286
7.93.2.18 setString()	286
7.93.2.19 setYearDay()	286
7.94 BTimer Class Reference	286
7.94.1 Detailed Description	287
7.94.2 Constructor & Destructor Documentation	287
7.94.2.1 BTimer()	287

7.94.2.2 ~BTimer()	287
7.94.3 Member Function Documentation	287
7.94.3.1 add()	287
7.94.3.2 average()	288
7.94.3.3 clear()	288
7.94.3.4 getElapsedTime()	288
7.94.3.5 peak()	288
7.94.3.6 start()	288
7.94.3.7 stop()	288
7.95 BTimeStamp Class Reference	289
7.95.1 Constructor & Destructor Documentation	290
7.95.1.1 BTimeStamp() [1/3]	290
7.95.1.2 BTimeStamp() [2/3]	291
7.95.1.3 BTimeStamp() [3/3]	291
7.95.1.4 ~BTimeStamp()	291
7.95.2 Member Function Documentation	291
7.95.2.1 addMicroSeconds()	291
7.95.2.2 addMilliSeconds()	291
7.95.2.3 addSeconds()	292
7.95.2.4 clear()	292
7.95.2.5 compare()	292
7.95.2.6 day()	292
7.95.2.7 difference()	292
7.95.2.8 getDate()	292
7.95.2.9 getString()	293
7.95.2.10 getStringFormatted()	293
7.95.2.11 getStringNoMs()	293
7.95.2.12 getYearMicroSeconds()	293
7.95.2.13 getYearSeconds()	293
7.95.2.14 hour()	293
7.95.2.15 isLeap()	294
7.95.2.16 isSet()	294
7.95.2.17 microSecond()	294
7.95.2.18 minute()	294
7.95.2.19 month()	294
7.95.2.20 operator BString()	294
7.95.2.21 operator"!="()	294
7.95.2.22 operator<()	295
7.95.2.23 operator<=()	295
7.95.2.24 operator=()	295
7.95.2.25 operator==(())	295
7.95.2.26 operator>()	295

7.95.2.27 operator>=()	295
7.95.2.28 second()	295
7.95.2.29 set() [1/3]	296
7.95.2.30 set() [2/3]	296
7.95.2.31 set() [3/3]	296
7.95.2.32 setFirst()	296
7.95.2.33 setLast()	296
7.95.2.34 setNow()	297
7.95.2.35 setString()	297
7.95.2.36 setTime()	297
7.95.2.37 setYDay()	297
7.95.2.38 yday()	297
7.95.2.39 year()	297
7.95.3 Member Data Documentation	298
7.95.3.1 ohour	298
7.95.3.2 omicroSecond	298
7.95.3.3 ominute	298
7.95.3.4 osecond	298
7.95.3.5 ospare	298
7.95.3.6 oyday	299
7.95.3.7 oyear	299
7.96 BTimeStampMs Class Reference	299
7.96.1 Constructor & Destructor Documentation	300
7.96.1.1 BTimeStampMs()	301
7.96.1.2 ~BTimeStampMs()	301
7.96.2 Member Function Documentation	301
7.96.2.1 addMilliseconds()	301
7.96.2.2 addSeconds()	301
7.96.2.3 clear()	301
7.96.2.4 compare()	301
7.96.2.5 difference()	302
7.96.2.6 getDate()	302
7.96.2.7 getDurationString()	302
7.96.2.8 getDurationStringNoMs()	302
7.96.2.9 getString()	302
7.96.2.10 getStringNoMs()	303
7.96.2.11 getStringRaw()	303
7.96.2.12 getYearMilliseconds()	303
7.96.2.13 getYearSeconds()	303
7.96.2.14 isLeap()	303
7.96.2.15 operator<()	303
7.96.2.16 operator<=()	304



7.96.2.17 operator>()	304
7.96.2.18 operator>=()	304
7.96.2.19 set()	304
7.96.2.20 setDurationString()	304
7.96.2.21 setFirst()	304
7.96.2.22 setLast()	305
7.96.2.23 setNow()	305
7.96.2.24 setString()	305
7.96.2.25 setTime()	305
7.96.2.26 setYDay()	305
7.96.2.27 subMilliseconds()	306
7.96.2.28 subSeconds()	306
7.96.3 Member Data Documentation	306
7.96.3.1 hour	306
7.96.3.2 milliSecond	306
7.96.3.3 minute	306
7.96.3.4 sampleNumber	307
7.96.3.5 second	307
7.96.3.6 yday	307
7.96.3.7 year	307
7.97 BTimeUs Class Reference	307
7.97.1 Constructor & Destructor Documentation	308
7.97.1.1 BTimeUs() [1/2]	309
7.97.1.2 BTimeUs() [2/2]	309
7.97.2 Member Function Documentation	309
7.97.2.1 addMicroSeconds()	309
7.97.2.2 addSeconds()	309
7.97.2.3 getDate()	309
7.97.2.4 getMicroSeconds()	310
7.97.2.5 getSeconds()	310
7.97.2.6 getString()	310
7.97.2.7 getStringUs()	310
7.97.2.8 getTime()	310
7.97.2.9 isLeapYear()	310
7.97.2.10 isSet()	311
7.97.2.11 operator BTime()	311
7.97.2.12 operator"!=(	311
7.97.2.13 operator+()	311
7.97.2.14 operator+=(	311
7.97.2.15 operator<()	311
7.97.2.16 operator<=()	311
7.97.2.17 operator==(	312

7.97.2.18 operator>()	312
7.97.2.19 operator>=()	312
7.97.2.20 set() [1/2]	312
7.97.2.21 set() [2/2]	312
7.97.2.22 setString()	312
7.97.2.23 setYearDay()	313
7.98 BUrl Class Reference	313
7.98.1 Detailed Description	313
7.98.2 Constructor & Destructor Documentation	313
7.98.2.1 BUrl()	313
7.98.2.2 ~BUrl()	313
7.98.3 Member Function Documentation	314
7.98.3.1 readString()	314
7.99 BList< T >::Node Class Reference	314
7.99.1 Constructor & Destructor Documentation	314
7.99.1.1 Node()	314
7.99.2 Member Data Documentation	315
7.99.2.1 item	315
<b>8 File Documentation</b>	<b>317</b>
8.1 /src/bdev/beam-lib/doc/overview.dox File Reference	317
8.2 BArray.h File Reference	317
8.2.1 Macro Definition Documentation	317
8.2.1.1 BArrayLoop	317
8.3 BAtomic.h File Reference	317
8.3.1 Typedef Documentation	318
8.3.1.1 BAtomicInt32	318
8.3.1.2 BAtomicInt64	318
8.3.1.3 BAtomicUInt32	318
8.3.1.4 BAtomicUInt64	318
8.4 BAtomicCount.h File Reference	318
8.5 BBuffer.cpp File Reference	319
8.5.1 Variable Documentation	319
8.5.1.1 roundSize	319
8.6 BBuffer.h File Reference	319
8.6.1 Macro Definition Documentation	320
8.6.1.1 BBigEndian	320
8.7 BComms.cpp File Reference	320
8.8 BComms.h File Reference	320
8.9 BComplex.h File Reference	320
8.9.1 Typedef Documentation	320
8.9.1.1 BComplex	321

8.9.1.2 BComplex32 . . . . .	321
8.9.1.3 BComplex64 . . . . .	321
8.10 BCond.cpp File Reference . . . . .	321
8.11 BCond.h File Reference . . . . .	321
8.12 BCondInt.cpp File Reference . . . . .	321
8.12.1 Function Documentation . . . . .	322
8.12.1.1 getTimeout() . . . . .	322
8.13 BCondInt.h File Reference . . . . .	322
8.14 BConfig.cpp File Reference . . . . .	322
8.15 BConfig.h File Reference . . . . .	322
8.16 BCrc16.cpp File Reference . . . . .	323
8.16.1 Function Documentation . . . . .	323
8.16.1.1 bcrc16() . . . . .	323
8.16.2 Variable Documentation . . . . .	323
8.16.2.1 table_crc_hi . . . . .	323
8.16.2.2 table_crc_lo . . . . .	324
8.17 BCrc16.h File Reference . . . . .	324
8.17.1 Function Documentation . . . . .	324
8.17.1.1 bcrc16() . . . . .	324
8.18 BCrc32.cpp File Reference . . . . .	324
8.18.1 Function Documentation . . . . .	325
8.18.1.1 bcrc32() . . . . .	325
8.18.2 Variable Documentation . . . . .	325
8.18.2.1 crc32_tab . . . . .	325
8.19 BCrc32.h File Reference . . . . .	325
8.19.1 Function Documentation . . . . .	325
8.19.1.1 bcrc32() . . . . .	326
8.20 BDate-1.cpp File Reference . . . . .	326
8.20.1 Function Documentation . . . . .	326
8.20.1.1 fromBString() . . . . .	326
8.20.1.2 toBString() . . . . .	326
8.20.2 Variable Documentation . . . . .	326
8.20.2.1 mon_yday . . . . .	327
8.21 BDate.cpp File Reference . . . . .	327
8.21.1 Function Documentation . . . . .	327
8.21.1.1 fromBString() . . . . .	327
8.21.1.2 toBString() . . . . .	327
8.21.2 Variable Documentation . . . . .	327
8.21.2.1 mon_yday . . . . .	328
8.22 BDate.h File Reference . . . . .	328
8.22.1 Function Documentation . . . . .	328
8.22.1.1 fromBString() . . . . .	328

8.22.1.2 toString()	328
8.23 BDebug.cpp File Reference	329
8.23.1 Function Documentation	329
8.23.1.1 bhd32()	329
8.23.1.2 bhd8()	329
8.23.1.3 bhd8a()	330
8.23.1.4 bhda32()	330
8.23.1.5 bhda8()	330
8.23.1.6 getTime()	330
8.23.1.7 setDebug()	330
8.23.2 Variable Documentation	330
8.23.2.1 bdebug	330
8.24 BDebug.h File Reference	331
8.24.1 Macro Definition Documentation	331
8.24.1.1 BDebug_STD	332
8.24.1.2 dl1printf	332
8.24.1.3 dl2printf	332
8.24.1.4 dl3printf	332
8.24.1.5 dl4printf	332
8.24.1.6 dprintf	332
8.24.1.7 eprintf	333
8.24.1.8 nprintf	333
8.24.1.9 wprintf	333
8.24.2 Function Documentation	333
8.24.2.1 bgettid()	333
8.24.2.2 bhd32()	333
8.24.2.3 bhd8()	333
8.24.2.4 bhd8a()	334
8.24.2.5 bhda8()	334
8.24.2.6 bhds32()	334
8.24.2.7 getTime()	334
8.24.2.8 setDebug()	334
8.24.2.9 tprintf()	334
8.24.3 Variable Documentation	334
8.24.3.1 bdebug	335
8.25 BDict.cpp File Reference	335
8.25.1 Function Documentation	335
8.25.1.1 bdictStringToString()	335
8.25.1.2 fromBString()	335
8.25.1.3 toBString()	335
8.26 BDict.h File Reference	335
8.26.1 Typedef Documentation	336

8.26.1.1 BDictString	336
8.26.2 Function Documentation	336
8.26.2.1 bdictStringToString()	336
8.26.2.2 fromBString()	336
8.26.2.3 toBString()	337
8.27 BDictMap.h File Reference	337
8.27.1 Typedef Documentation	337
8.27.1.1 BDictMapString	337
8.28 BDir.cpp File Reference	337
8.28.1 Function Documentation	338
8.28.1.1 wild()	338
8.28.2 Variable Documentation	338
8.28.2.1 wildString	338
8.29 BDir.h File Reference	338
8.30 BDuration.cpp File Reference	338
8.31 BDuration.h File Reference	338
8.32 BEndian.cpp File Reference	339
8.32.1 Function Documentation	339
8.32.1.1 bswap_copy()	339
8.33 BEndian.h File Reference	339
8.33.1 Macro Definition Documentation	340
8.33.1.1 be16toh	340
8.33.1.2 be32toh	341
8.33.1.3 be64toh	341
8.33.1.4 htobe16	341
8.33.1.5 htobe32	341
8.33.1.6 htobe64	341
8.33.1.7 htole16	341
8.33.1.8 htole32	341
8.33.1.9 htole64	342
8.33.1.10 le16toh	342
8.33.1.11 le32toh	342
8.33.1.12 le64toh	342
8.33.2 Function Documentation	342
8.33.2.1 betoh() [1/8]	342
8.33.2.2 betoh() [2/8]	342
8.33.2.3 betoh() [3/8]	343
8.33.2.4 betoh() [4/8]	343
8.33.2.5 betoh() [5/8]	343
8.33.2.6 betoh() [6/8]	343
8.33.2.7 betoh() [7/8]	343
8.33.2.8 betoh() [8/8]	343

8.33.2.9 bswap_copy()	343
8.33.2.10 bswap_p16()	344
8.33.2.11 bswap_p32()	344
8.33.2.12 bswap_p64()	344
8.33.2.13 bswap_p8()	344
8.33.2.14 htobe() [1/8]	344
8.33.2.15 htobe() [2/8]	344
8.33.2.16 htobe() [3/8]	345
8.33.2.17 htobe() [4/8]	345
8.33.2.18 htobe() [5/8]	345
8.33.2.19 htobe() [6/8]	345
8.33.2.20 htobe() [7/8]	345
8.33.2.21 htobe() [8/8]	345
8.33.2.22 htobe() [1/8]	345
8.33.2.23 htobe() [2/8]	346
8.33.2.24 htobe() [3/8]	346
8.33.2.25 htobe() [4/8]	346
8.33.2.26 htobe() [5/8]	346
8.33.2.27 htobe() [6/8]	346
8.33.2.28 htobe() [7/8]	346
8.33.2.29 htobe() [8/8]	346
8.33.2.30 letoh() [1/8]	347
8.33.2.31 letoh() [2/8]	347
8.33.2.32 letoh() [3/8]	347
8.33.2.33 letoh() [4/8]	347
8.33.2.34 letoh() [5/8]	347
8.33.2.35 letoh() [6/8]	347
8.33.2.36 letoh() [7/8]	347
8.33.2.37 letoh() [8/8]	348
8.34 BEntry.cpp File Reference	348
8.35 BEntry.h File Reference	348
8.36 BError.cpp File Reference	348
8.37 BError.h File Reference	348
8.37.1 Enumeration Type Documentation	349
8.37.1.1 BErrorNum	349
8.38 BErrorTime.cpp File Reference	350
8.39 BErrorTime.h File Reference	350
8.40 BEvent.cpp File Reference	350
8.41 BEvent.h File Reference	350
8.41.1 Typedef Documentation	351
8.41.1.1 BEventQueue	351
8.42 BEvent1.cpp File Reference	351

8.43 BEvent1.h File Reference	351
8.43.1 Enumeration Type Documentation	351
8.43.1.1 BEvent1Type	351
8.44 B Fifo.h File Reference	352
8.45 B Fifo.inc File Reference	352
8.46 B FifoCirc.cpp File Reference	352
8.46.1 Macro Definition Documentation	352
8.46.1.1 dprintf	352
8.47 B FifoCirc.h File Reference	353
8.48 B FifoCirc.inc File Reference	353
8.49 B File.cpp File Reference	353
8.49.1 Macro Definition Documentation	353
8.49.1.1 STRBUF	353
8.50 B File.h File Reference	353
8.51 B FileCsv.cpp File Reference	354
8.52 B FileCsv.h File Reference	354
8.53 B FileData.cpp File Reference	354
8.54 B FileData.h File Reference	354
8.55 B Firmware.h File Reference	354
8.55.1 Typedef Documentation	356
8.55.1.1 B FirmwareFirmwareHeader	356
8.55.2 Function Documentation	356
8.55.2.1 __attribute__()	356
8.55.2.2 bfirmwareBoot()	356
8.55.2.3 bfirmwareValid()	356
8.55.3 Variable Documentation	356
8.55.3.1 __attribute__	356
8.55.3.2 address	357
8.55.3.3 B FirmwareFormatGzip	357
8.55.3.4 B FirmwareFormatRaw	357
8.55.3.5 B FirmwareInfoEncrypt1	357
8.55.3.6 B FirmwareInfoMagic	357
8.55.3.7 B FirmwareMagic	357
8.55.3.8 B FirmwarePlatformBMeasure125	357
8.55.3.9 B FirmwarePlatformBMeasure125Boot	357
8.55.3.10 B FirmwarePlatformBMeasure125Cpu	358
8.55.3.11 B FirmwarePlatformBMeasure125Fpga	358
8.55.3.12 B FirmwarePlatformBMeasure125Wifi	358
8.55.3.13 B FirmwareTypeFile	358
8.55.3.14 B FirmwareTypeFirmware	358
8.55.3.15 B FirmwareTypeSegment	358
8.55.3.16 checksum	358

8.55.3.17 dataLength . . . . .	358
8.55.3.18 fileLength . . . . .	359
8.55.3.19 format . . . . .	359
8.55.3.20 itemType . . . . .	359
8.55.3.21 length . . . . .	359
8.55.3.22 magic . . . . .	359
8.55.3.23 numSegments . . . . .	359
8.55.3.24 platform . . . . .	359
8.55.3.25 special . . . . .	359
8.55.3.26 startAddress . . . . .	360
8.55.3.27 ver0 . . . . .	360
8.55.3.28 ver1 . . . . .	360
8.55.3.29 ver2 . . . . .	360
8.55.3.30 ver3 . . . . .	360
8.56 BList.h File Reference . . . . .	360
8.56.1 Macro Definition Documentation . . . . .	361
8.56.1.1 BListLoop . . . . .	361
8.57 BList_func.h File Reference . . . . .	361
8.58 BMutex.cpp File Reference . . . . .	361
8.58.1 Macro Definition Documentation . . . . .	361
8.58.1.1 MDEBUG . . . . .	361
8.59 BMutex.h File Reference . . . . .	361
8.60 BMySQL.cpp File Reference . . . . .	362
8.61 BMySQL.h File Reference . . . . .	362
8.62 BNameValue.h File Reference . . . . .	362
8.63 Boap.cpp File Reference . . . . .	362
8.63.1 Macro Definition Documentation . . . . .	363
8.63.1.1 APIVERSION_TEST . . . . .	363
8.63.1.2 DEBUG . . . . .	363
8.63.1.3 dprintf . . . . .	363
8.63.1.4 IS_BIG_ENDIAN . . . . .	363
8.63.2 Variable Documentation . . . . .	363
8.63.2.1 boapPort . . . . .	364
8.64 Boap.d File Reference . . . . .	364
8.65 build_x86_64/Boap.d File Reference . . . . .	364
8.66 Boap.h File Reference . . . . .	364
8.66.1 Typedef Documentation . . . . .	365
8.66.1.1 BoapFunc . . . . .	365
8.66.1.2 BoapService . . . . .	365
8.66.2 Enumeration Type Documentation . . . . .	365
8.66.2.1 BoapPriority . . . . .	365
8.66.2.2 BoapType . . . . .	365



8.66.3 Variable Documentation . . . . .	366
8.66.3.1 BoapMagic . . . . .	366
8.67 BoapMc.cpp File Reference . . . . .	366
8.67.1 Macro Definition Documentation . . . . .	366
8.67.1.1 DEBUG_LOCAL . . . . .	366
8.67.1.2 DEBUG_LOCAL1 . . . . .	367
8.67.1.3 dlprintf . . . . .	367
8.67.1.4 dlprintf . . . . .	367
8.68 BoapMc.h File Reference . . . . .	367
8.68.1 Enumeration Type Documentation . . . . .	368
8.68.1.1 BoapMcType . . . . .	368
8.68.2 Function Documentation . . . . .	368
8.68.2.1 __attribute__() . . . . .	368
8.68.3 Variable Documentation . . . . .	368
8.68.3.1 __attribute__ . . . . .	368
8.68.3.2 addressFrom . . . . .	369
8.68.3.3 addressTo . . . . .	369
8.68.3.4 checksum . . . . .	369
8.68.3.5 cmd . . . . .	369
8.68.3.6 error . . . . .	369
8.68.3.7 length . . . . .	369
8.69 BoapMc1.cpp File Reference . . . . .	369
8.69.1 Macro Definition Documentation . . . . .	370
8.69.1.1 BDEBUGL1 . . . . .	370
8.69.1.2 BDEBUGL2 . . . . .	370
8.70 BoapMc1.h File Reference . . . . .	370
8.70.1 Enumeration Type Documentation . . . . .	371
8.70.1.1 BoapMc1Type . . . . .	371
8.70.2 Function Documentation . . . . .	371
8.70.2.1 __attribute__() . . . . .	371
8.70.2.2 boapMc1CommsRoundupLen() . . . . .	372
8.70.3 Variable Documentation . . . . .	372
8.70.3.1 __attribute__ . . . . .	372
8.70.3.2 addressFrom . . . . .	372
8.70.3.3 addressTo . . . . .	372
8.70.3.4 BoapMc1Magic . . . . .	372
8.70.3.5 checksum . . . . .	372
8.70.3.6 cmd . . . . .	373
8.70.3.7 data . . . . .	373
8.70.3.8 error . . . . .	373
8.70.3.9 head . . . . .	373
8.70.3.10 length . . . . .	373

8.70.3.11 magic	373
8.70.3.12 number	374
8.70.3.13 string	374
8.71 BoapnsC.cpp File Reference	374
8.72 BoapnsC.h File Reference	374
8.73 BoapnsD.cpp File Reference	375
8.74 BoapnsD.h File Reference	375
8.75 BoapSimple.cc File Reference	375
8.75.1 Macro Definition Documentation	376
8.75.1.1 DEBUG	376
8.75.1.2 dprintf	376
8.75.2 Variable Documentation	376
8.75.2.1 roundSize	376
8.76 BoapSimple.h File Reference	376
8.76.1 Typedef Documentation	377
8.76.1.1 BoapFunc	377
8.76.1.2 BoapService	377
8.76.1.3 Double	377
8.76.1.4 Int16	377
8.76.1.5 Int32	378
8.76.1.6 Int8	378
8.76.1.7 UInt16	378
8.76.1.8 UInt32	378
8.76.1.9 UInt8	378
8.76.2 Enumeration Type Documentation	378
8.76.2.1 BoapType	378
8.77 BObj.cpp File Reference	379
8.78 BObj.h File Reference	379
8.79 BObjStringFormat.cpp File Reference	379
8.79.1 Function Documentation	380
8.79.1.1 toBDictStringFromJson()	380
8.79.1.2 toBString() [1/18]	380
8.79.1.3 toBString() [2/18]	380
8.79.1.4 toBString() [3/18]	380
8.79.1.5 toBString() [4/18]	381
8.79.1.6 toBString() [5/18]	381
8.79.1.7 toBString() [6/18]	381
8.79.1.8 toBString() [7/18]	381
8.79.1.9 toBString() [8/18]	381
8.79.1.10 toBString() [9/18]	381
8.79.1.11 toBString() [10/18]	382
8.79.1.12 toBString() [11/18]	382

8.79.1.13 toBString()	[12/18]	382
8.79.1.14 toBString()	[13/18]	382
8.79.1.15 toBString()	[14/18]	382
8.79.1.16 toBString()	[15/18]	382
8.79.1.17 toBString()	[16/18]	383
8.79.1.18 toBString()	[17/18]	383
8.79.1.19 toBString()	[18/18]	383
8.79.1.20 toBStringJson()	[1/18]	383
8.79.1.21 toBStringJson()	[2/18]	383
8.79.1.22 toBStringJson()	[3/18]	383
8.79.1.23 toBStringJson()	[4/18]	384
8.79.1.24 toBStringJson()	[5/18]	384
8.79.1.25 toBStringJson()	[6/18]	384
8.79.1.26 toBStringJson()	[7/18]	384
8.79.1.27 toBStringJson()	[8/18]	384
8.79.1.28 toBStringJson()	[9/18]	384
8.79.1.29 toBStringJson()	[10/18]	385
8.79.1.30 toBStringJson()	[11/18]	385
8.79.1.31 toBStringJson()	[12/18]	385
8.79.1.32 toBStringJson()	[13/18]	385
8.79.1.33 toBStringJson()	[14/18]	385
8.79.1.34 toBStringJson()	[15/18]	385
8.79.1.35 toBStringJson()	[16/18]	386
8.79.1.36 toBStringJson()	[17/18]	386
8.79.1.37 toBStringJson()	[18/18]	386
8.80 BObjStringFormat.h File Reference		386
8.80.1 Function Documentation		387
8.80.1.1 base64_decode()		387
8.80.1.2 base64_encode()		387
8.80.1.3 toBDictStringFromJson()		387
8.80.1.4 toBString()	[1/18]	388
8.80.1.5 toBString()	[2/18]	388
8.80.1.6 toBString()	[3/18]	388
8.80.1.7 toBString()	[4/18]	388
8.80.1.8 toBString()	[5/18]	388
8.80.1.9 toBString()	[6/18]	388
8.80.1.10 toBString()	[7/18]	389
8.80.1.11 toBString()	[8/18]	389
8.80.1.12 toBString()	[9/18]	389
8.80.1.13 toBString()	[10/18]	389
8.80.1.14 toBString()	[11/18]	389
8.80.1.15 toBString()	[12/18]	389

8.80.1.16 toBString() [13/18]	390
8.80.1.17 toBString() [14/18]	390
8.80.1.18 toBString() [15/18]	390
8.80.1.19 toBString() [16/18]	390
8.80.1.20 toBString() [17/18]	390
8.80.1.21 toBString() [18/18]	390
8.80.1.22 toBStringJson() [1/18]	391
8.80.1.23 toBStringJson() [2/18]	391
8.80.1.24 toBStringJson() [3/18]	391
8.80.1.25 toBStringJson() [4/18]	391
8.80.1.26 toBStringJson() [5/18]	391
8.80.1.27 toBStringJson() [6/18]	391
8.80.1.28 toBStringJson() [7/18]	392
8.80.1.29 toBStringJson() [8/18]	392
8.80.1.30 toBStringJson() [9/18]	392
8.80.1.31 toBStringJson() [10/18]	392
8.80.1.32 toBStringJson() [11/18]	392
8.80.1.33 toBStringJson() [12/18]	392
8.80.1.34 toBStringJson() [13/18]	393
8.80.1.35 toBStringJson() [14/18]	393
8.80.1.36 toBStringJson() [15/18]	393
8.80.1.37 toBStringJson() [16/18]	393
8.80.1.38 toBStringJson() [17/18]	393
8.80.1.39 toBStringJson() [18/18]	393
8.81 BPoll.cpp File Reference	394
8.82 BPoll.h File Reference	394
8.83 BQueue.h File Reference	394
8.83.1 Typedef Documentation	394
8.83.1.1 BQueueInt	394
8.84 BRefData.cpp File Reference	395
8.84.1 Macro Definition Documentation	395
8.84.1.1 CHUNK	395
8.85 BRefData.h File Reference	395
8.86 BRtc.cpp File Reference	395
8.87 BRtc.h File Reference	396
8.88 BRWLock.cpp File Reference	396
8.89 BRWLock.h File Reference	396
8.90 BSema.cpp File Reference	396
8.91 BSema.h File Reference	396
8.92 BSemaphore.cpp File Reference	397
8.93 BSemaphore.h File Reference	397
8.94 BSocket.cpp File Reference	397

8.94.1 Macro Definition Documentation	398
8.94.1.1 IP_MTU	398
8.95 BSocket.h File Reference	398
8.95.1 Macro Definition Documentation	398
8.95.1.1 MSG_NOSIGNAL	398
8.95.1.2 SO_PRIORITY	399
8.95.1.3 SOL_IP	399
8.96 BSpI.cpp File Reference	399
8.97 BSpI.h File Reference	399
8.98 BString.cpp File Reference	399
8.98.1 Macro Definition Documentation	400
8.98.1.1 MINUS	400
8.98.1.2 STRIP	400
8.98.2 Function Documentation	401
8.98.2.1 barrayToString()	401
8.98.2.2 blistToString()	401
8.98.2.3 bstringListinList()	401
8.98.2.4 bstringToArray()	401
8.98.2.5 bstringToList()	401
8.98.2.6 bstrncpy()	401
8.98.2.7 bstrtrim()	402
8.98.2.8 charToArray()	402
8.98.2.9 charToList()	402
8.98.2.10 floatToString()	402
8.98.2.11 fromBString() [1/6]	402
8.98.2.12 fromBString() [2/6]	402
8.98.2.13 fromBString() [3/6]	403
8.98.2.14 fromBString() [4/6]	403
8.98.2.15 fromBString() [5/6]	403
8.98.2.16 fromBString() [6/6]	403
8.98.2.17 gmatch()	403
8.98.2.18 int64ToString()	403
8.98.2.19 intToString()	404
8.98.2.20 operator<<()	404
8.98.2.21 operator>>()	404
8.98.2.22 toBString() [1/6]	404
8.98.2.23 toBString() [2/6]	404
8.98.2.24 toBString() [3/6]	404
8.98.2.25 toBString() [4/6]	405
8.98.2.26 toBString() [5/6]	405
8.98.2.27 toBString() [6/6]	405
8.98.3 Variable Documentation	405

8.98.3.1 <a href="#">base64_decode_table</a>	405
8.99 <a href="#">BString.h</a> File Reference	405
8.99.1 Function Documentation	406
8.99.1.1 <a href="#">bstrncpy()</a>	406
8.99.1.2 <a href="#">bstrtrim()</a>	406
8.99.1.3 <a href="#">floatToString()</a>	407
8.99.1.4 <a href="#">from_hex()</a>	407
8.99.1.5 <a href="#">fromBString()</a> [1/6]	407
8.99.1.6 <a href="#">fromBString()</a> [2/6]	407
8.99.1.7 <a href="#">fromBString()</a> [3/6]	407
8.99.1.8 <a href="#">fromBString()</a> [4/6]	407
8.99.1.9 <a href="#">fromBString()</a> [5/6]	408
8.99.1.10 <a href="#">fromBString()</a> [6/6]	408
8.99.1.11 <a href="#">int64ToString()</a>	408
8.99.1.12 <a href="#">intToString()</a>	408
8.99.1.13 <a href="#">operator&lt;&lt;()</a>	408
8.99.1.14 <a href="#">operator&gt;&gt;()</a>	408
8.99.1.15 <a href="#">to_hex()</a>	409
8.99.1.16 <a href="#">toBString()</a> [1/6]	409
8.99.1.17 <a href="#">toBString()</a> [2/6]	409
8.99.1.18 <a href="#">toBString()</a> [3/6]	409
8.99.1.19 <a href="#">toBString()</a> [4/6]	409
8.99.1.20 <a href="#">toBString()</a> [5/6]	409
8.99.1.21 <a href="#">toBString()</a> [6/6]	410
8.100 <a href="#">BStringLocked.h</a> File Reference	410
8.101 <a href="#">BSys.cpp</a> File Reference	410
8.101.1 Function Documentation	410
8.101.1.1 <a href="#">delayMs()</a>	410
8.101.1.2 <a href="#">delayUs()</a>	411
8.102 <a href="#">BSys.h</a> File Reference	411
8.102.1 Function Documentation	411
8.102.1.1 <a href="#">delayMs()</a>	411
8.102.1.2 <a href="#">delayUs()</a>	411
8.103 <a href="#">BTable.cpp</a> File Reference	411
8.104 <a href="#">BTable.h</a> File Reference	412
8.105 <a href="#">BTask.cpp</a> File Reference	412
8.106 <a href="#">BTask.h</a> File Reference	412
8.107 <a href="#">BThread.cpp</a> File Reference	412
8.108 <a href="#">BThread.h</a> File Reference	412
8.109 <a href="#">BTime.cpp</a> File Reference	413
8.109.1 Function Documentation	413
8.109.1.1 <a href="#">yearDays()</a>	413

8.109.1.2 yearIsLeap()	413
8.109.2 Variable Documentation	413
8.109.2.1 monDays	413
8.110 BTime.h File Reference	414
8.111 BTimer.cpp File Reference	414
8.112 BTimer.h File Reference	414
8.113 BTimeStamp.cpp File Reference	414
8.113.1 Function Documentation	415
8.113.1.1 fromBString()	415
8.113.1.2 toBString()	415
8.113.2 Variable Documentation	415
8.113.2.1 mon_yday	415
8.114 BTimeStamp.h File Reference	415
8.114.1 Function Documentation	416
8.114.1.1 fromBString()	416
8.114.1.2 toBString()	416
8.115 BTimeStampMs.cpp File Reference	416
8.115.1 Variable Documentation	416
8.115.1.1 mon_yday	416
8.116 BTimeStampMs.h File Reference	417
8.117 BTimeUs.cpp File Reference	417
8.117.1 Function Documentation	417
8.117.1.1 yearDays()	417
8.117.1.2 yearIsLeap()	417
8.117.2 Variable Documentation	417
8.117.2.1 monDays	418
8.118 BTimeUs.h File Reference	418
8.119 BTypes.h File Reference	418
8.119.1 Typedef Documentation	419
8.119.1.1 BArrayDouble	419
8.119.1.2 BArrayFloat	419
8.119.1.3 BChar	420
8.119.1.4 BDouble	420
8.119.1.5 BFloat	420
8.119.1.6 BFloat32	420
8.119.1.7 BFloat64	420
8.119.1.8 BInt	420
8.119.1.9 BInt16	420
8.119.1.10 BInt32	420
8.119.1.11 BInt64	421
8.119.1.12 BInt8	421
8.119.1.13 Bool	421

8.119.1.14 BSize	421
8.119.1.15 BTimeout	421
8.119.1.16 BUInt	421
8.119.1.17 BUInt16	421
8.119.1.18 BUInt32	421
8.119.1.19 BUInt64	422
8.119.1.20 BUInt8	422
8.119.2 Enumeration Type Documentation	422
8.119.2.1 BEventType	422
8.119.2.2 BEventWaitSet	422
8.119.2.3 BType	423
8.119.2.4 BTypeComp	423
8.119.3 Function Documentation	423
8.119.3.1 byteSwap16()	424
8.119.3.2 byteSwap32()	424
8.119.3.3 byteSwap64()	424
8.119.3.4 byteSwap8()	424
8.119.3.5 timeoutTicks()	424
8.119.4 Variable Documentation	424
8.119.4.1 BTimeoutForever	424
8.120 build_x86_64/BBuffer.d File Reference	425
8.121 build_x86_64/BComms.d File Reference	425
8.122 build_x86_64/BCond.d File Reference	425
8.123 build_x86_64/BCondInt.d File Reference	425
8.124 build_x86_64/BConfig.d File Reference	425
8.125 build_x86_64/BCrc16.d File Reference	425
8.126 build_x86_64/BCrc32.d File Reference	425
8.127 build_x86_64/BDate.d File Reference	425
8.128 build_x86_64/BDebug.d File Reference	425
8.129 build_x86_64/BDict.d File Reference	425
8.130 build_x86_64/BDir.d File Reference	425
8.131 build_x86_64/BDuration.d File Reference	425
8.132 build_x86_64/BEndian.d File Reference	425
8.133 build_x86_64/BEntry.d File Reference	425
8.134 build_x86_64/BError.d File Reference	425
8.135 build_x86_64/BErrorTime.d File Reference	425
8.136 build_x86_64/BEvent.d File Reference	425
8.137 build_x86_64/BEvent1.d File Reference	426
8.138 build_x86_64/BFifoCirc.d File Reference	426
8.139 build_x86_64/BFile.d File Reference	426
8.140 build_x86_64/BFileCsv.d File Reference	426
8.141 build_x86_64/BFileData.d File Reference	426



8.142 build_x86_64/BMutex.d File Reference . . . . .	426
8.143 build_x86_64/BMySQL.d File Reference . . . . .	426
8.144 build_x86_64/BoapMc.d File Reference . . . . .	426
8.145 build_x86_64/BoapMc1.d File Reference . . . . .	426
8.146 build_x86_64/BoapnsC.d File Reference . . . . .	426
8.147 build_x86_64/BoapnsD.d File Reference . . . . .	426
8.148 build_x86_64/BObj.d File Reference . . . . .	426
8.149 build_x86_64/BObjStringFormat.d File Reference . . . . .	426
8.150 build_x86_64/BPoll.d File Reference . . . . .	426
8.151 build_x86_64/BRefData.d File Reference . . . . .	426
8.152 build_x86_64/BRtc.d File Reference . . . . .	426
8.153 build_x86_64/BRWLock.d File Reference . . . . .	426
8.154 build_x86_64/BSema.d File Reference . . . . .	427
8.155 build_x86_64/BSemaphore.d File Reference . . . . .	427
8.156 build_x86_64/BSocket.d File Reference . . . . .	427
8.157 build_x86_64/BSpi.d File Reference . . . . .	427
8.158 build_x86_64/BString.d File Reference . . . . .	427
8.159 build_x86_64/BSys.d File Reference . . . . .	427
8.160 build_x86_64/BTable.d File Reference . . . . .	427
8.161 build_x86_64/BTask.d File Reference . . . . .	427
8.162 build_x86_64/BThread.d File Reference . . . . .	427
8.163 build_x86_64/BTime.d File Reference . . . . .	427
8.164 build_x86_64/BTimer.d File Reference . . . . .	427
8.165 build_x86_64/BTimeStamp.d File Reference . . . . .	427
8.166 build_x86_64/BTimeStampMs.d File Reference . . . . .	427
8.167 build_x86_64/BTimeUs.d File Reference . . . . .	427
8.168 build_x86_64/BUrl.d File Reference . . . . .	427
8.169 BUrl.cpp File Reference . . . . .	427
8.170 BUrl.h File Reference . . . . .	428

## Index

429



# Chapter 1

## Main Page

### Author

Dr Terry Barnaby

### Version

2.16.3

### Date

2020-06-29

## 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 Bbeam-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:

<a href="#">Boapns</a>	.....	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 . . . . .	193
BComms . . . . .	35
BCond . . . . .	41
BCondBool . . . . .	43
BCondInt . . . . .	45
BCondResource . . . . .	48
BCondValue . . . . .	50
BCondWrap . . . . .	53
BDataChunk . . . . .	58
BDate . . . . .	59
BDebugBacktrace . . . . .	65
BDictItem< Type > . . . . .	71
BDuration . . . . .	78
BEntry . . . . .	82
BError . . . . .	90
BEvent1Error . . . . .	99
BErrorTime . . . . .	94
BEvent . . . . .	96
BEvent1 . . . . .	97
BEvent1Error . . . . .	99
BEvent1Int . . . . .	100
BEvent1Pipe . . . . .	102
BEventPipe . . . . .	104
BFifo< Type > . . . . .	106
BFifo< BoapMcPacket > . . . . .	106
BFifoCirc< Type > . . . . .	112
BFifoCircPos . . . . .	119
BFile . . . . .	121
BFileCsv . . . . .	127
BFirmwareFileHeader . . . . .	129

BFirmwareInfo . . . . .	131
BFirmwareSegHeader . . . . .	133
BIter . . . . .	134
BList< T > . . . . .	136
BQueue< T > . . . . .	222
BList< BArray< BString > > . . . . .	136
BList< BDictItem< Type > > . . . . .	136
BDict< Type > . . . . .	66
BConfig . . . . .	56
BList< BEntry > . . . . .	136
BEntryList . . . . .	87
BEntryFile . . . . .	84
BList< BNameValue< T > > . . . . .	136
BNameValueList< T > . . . . .	154
BList< BoapFuncEntry > . . . . .	136
BList< BoapMcPacket > . . . . .	136
BQueue< BoapMcPacket > . . . . .	222
BList< BoapServerConnection * > . . . . .	136
BList< BoapServiceEntry > . . . . .	136
BList< BString > . . . . .	136
BList< BStringList > . . . . .	136
BFileData . . . . .	128
BList< struct dirent * > . . . . .	136
BDir . . . . .	75
BMutex . . . . .	147
BStringMutex . . . . .	274
BMutexLock . . . . .	149
BMysql . . . . .	150
BNameValue< T > . . . . .	153
BNode . . . . .	155
BList< T >::Node . . . . .	314
Boapns::BoapEntry . . . . .	163
BoapFuncEntry . . . . .	164
BoapMc1Comms . . . . .	166
BoapMc1Error . . . . .	173
BoapMc1Packet . . . . .	174
BoapMc1PacketHead . . . . .	174
BoapMcClientObject . . . . .	176
BoapMcComms . . . . .	179
BoapMcPacket . . . . .	186
BoapMcPacketHead . . . . .	187
BoapMcServiceObject . . . . .	188
BoapMcSignalObject . . . . .	190
BoapPacketHead . . . . .	199
BoapServiceEntry . . . . .	208
BoapServiceObject . . . . .	209
BObj . . . . .	216
BObjMember . . . . .	218
BPoll . . . . .	219
BRefData . . . . .	224
BRtc . . . . .	226
BRWLock . . . . .	229
BSema . . . . .	231
BSemaphore . . . . .	234
BSemaphoreBool . . . . .	235



BSemaphoreCount . . . . .	238
BSocket . . . . .	240
BoapClientObject . . . . .	156
Boapns::Boapns . . . . .	191
BoapClientObject . . . . .	156
BoapSignalObject . . . . .	214
BoapSignalObject . . . . .	214
BSocketAddress . . . . .	246
BSocketAddressINET . . . . .	249
BSpi . . . . .	252
BString . . . . .	254
BStringLocked . . . . .	272
BTable . . . . .	275
BTask . . . . .	276
BThread . . . . .	279
BoapServer . . . . .	201
BoapServerConnection . . . . .	206
BRtcThreaded . . . . .	228
BTime . . . . .	282
BTimer . . . . .	286
BTimeStamp . . . . .	289
BTimeStampMs . . . . .	299
BTimeUs . . . . .	307
BUrl . . . . .	313
map	
BDictMap< Value > . . . . .	72
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>	35
<a href="#">BCond</a>	41
<a href="#">BCondBool</a>	
Thread conditional boolean	43
<a href="#">BCondInt</a>	
Thread conditional value	45
<a href="#">BCondResource</a>	
Resource lock	48
<a href="#">BCondValue</a>	
Thread conditional value	50
<a href="#">BCondWrap</a>	53
<a href="#">BConfig</a>	
This class implements the configuration file access	56
<a href="#">BDataChunk</a>	58
<a href="#">BDate</a>	59
<a href="#">BDebugBacktrace</a>	65
<a href="#">BDict&lt; Type &gt;</a>	66
<a href="#">BDictItem&lt; Type &gt;</a>	
Template based Dictionary class	71
<a href="#">BDictMap&lt; Value &gt;</a>	72
<a href="#">BDir</a>	
File system directory class	75
<a href="#">BDuration</a>	78
<a href="#">BEntry</a>	
Manipulate a name value pair	82
<a href="#">BEntryFile</a>	
File of Entries	84
<a href="#">BEntryList</a>	
List of Entries. Where an entry is a name value pair	87

<a href="#">BError</a>	90
<a href="#">BErrorTime</a>	
Error return class	94
<a href="#">BEvent</a>	96
<a href="#">BEvent1</a>	
This class provides a base class for all event objects that can be sent over the events interface	97
<a href="#">BEvent1Error</a>	99
<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	100
<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	102
<a href="#">BEventPipe</a>	
This class provides an interface for sending simple integer events via a pipe file descriptor	104
<a href="#">BFifo&lt; Type &gt;</a>	106
<a href="#">BFifoCirc&lt; Type &gt;</a>	
This class implements a thread safe FIFO buffer	112
<a href="#">BFifoCircPos</a>	
This class implements a pointer into the Fifo's circular buffer	119
<a href="#">BFile</a>	
File operations class	121
<a href="#">BFileCsv</a>	127
<a href="#">BFileData</a>	128
<a href="#">BFirmwareFileHeader</a>	129
<a href="#">BFirmwareInfo</a>	131
<a href="#">BFirmwareSegHeader</a>	133
<a href="#">BIter</a>	
Iterator for <a href="#">BList</a>	134
<a href="#">BList&lt; T &gt;</a>	
Template based list class	136
<a href="#">BMutex</a>	147
<a href="#">BMutexLock</a>	149
<a href="#">BMySQL</a>	150
<a href="#">BNameValue&lt; T &gt;</a>	153
<a href="#">BNameValueList&lt; T &gt;</a>	154
<a href="#">BNode</a>	155
<a href="#">BoapClientObject</a>	156
<a href="#">Boapns::BoapEntry</a>	163
<a href="#">BoapFuncEntry</a>	164
<a href="#">BoapMc1Comms</a>	166
<a href="#">BoapMc1Error</a>	173
<a href="#">BoapMc1Packet</a>	174
<a href="#">BoapMc1PacketHead</a>	174
<a href="#">BoapMcClientObject</a>	176
<a href="#">BoapMcComms</a>	179
<a href="#">BoapMcPacket</a>	186
<a href="#">BoapMcPacketHead</a>	187
<a href="#">BoapMcServiceObject</a>	188
<a href="#">BoapMcSignalObject</a>	190
<a href="#">Boapns::Boapns</a>	191
<a href="#">BoapPacket</a>	193
<a href="#">BoapPacketHead</a>	199
<a href="#">BoapServer</a>	201
<a href="#">BoapServerConnection</a>	206
<a href="#">BoapServiceEntry</a>	208
<a href="#">BoapServiceObject</a>	209
<a href="#">BoapSignalObject</a>	214

BObj	216
BObjMember	218
BPoll	
This class provides an interface for polling a number of file descriptors. It uses round robin polling	219
BQueue< T >	
Queue class	222
BRefData	224
BRtc	
Realtime clock	226
BRtcThreaded	
Threaded real time clock	228
BRWLock	
Thread read-write locks	229
BSema	
Sempahore class	231
BSemaphore	
Semaphore class	234
BSemaphoreBool	235
BSemaphoreCount	238
BSocket	240
BSocketAddress	
Socket Address	246
BSocketAddressINET	
IP aware socket address	249
BSpi	
BSpi class	252
BString	254
BStringLocked	272
BStringMutex	274
BTable	275
BTask	276
BThread	279
BTime	282
BTimer	
Stopwatch style timer	286
BTimeStamp	289
BTimeStampMs	299
BTimeUs	307
BUrl	
Basic access to a Url	313
BList< T >::Node	314



## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

BArray.h	317
BAtomic.h	317
BAtomicCount.h	318
BBuffer.cpp	319
BBuffer.h	319
BComms.cpp	320
BComms.h	320
BComplex.h	320
BCond.cpp	321
BCond.h	321
BCondInt.cpp	321
BCondInt.h	322
BConfig.cpp	322
BConfig.h	322
BCrc16.cpp	323
BCrc16.h	324
BCrc32.cpp	324
BCrc32.h	325
BDate-1.cpp	326
BDate.cpp	327
BDate.h	328
BDebug.cpp	329
BDebug.h	331
BDict.cpp	335
BDict.h	335
BDictMap.h	337
BDir.cpp	337
BDir.h	338
BDuration.cpp	338
BDuration.h	338
BEndian.cpp	339
BEndian.h	339
BEntry.cpp	348
BEntry.h	348
BError.cpp	348

BError.h	348
BErrorTime.cpp	350
BErrorTime.h	350
BEvent.cpp	350
BEvent.h	350
BEvent1.cpp	351
BEvent1.h	351
BFifo.h	352
BFifo.inc	352
BFifoCirc.cpp	352
BFifoCirc.h	353
BFifoCirc.inc	353
BFile.cpp	353
BFile.h	353
BFileCsv.cpp	354
BFileCsv.h	354
BFileData.cpp	354
BFileData.h	354
BFirmware.h	354
BList.h	360
BList_func.h	361
BMutex.cpp	361
BMutex.h	361
BMysql.cpp	362
BMysql.h	362
BNameValue.h	362
Boap.cpp	362
Boap.d	364
Boap.h	364
BoapMc.cpp	366
BoapMc.h	367
BoapMc1.cpp	369
BoapMc1.h	370
BoapnsC.cpp	374
BoapnsC.h	374
BoapnsD.cpp	375
BoapnsD.h	375
BoapSimple.cc	375
BoapSimple.h	376
BObj.cpp	379
BObj.h	379
BObjStringFormat.cpp	379
BObjStringFormat.h	386
BPoll.cpp	394
BPoll.h	394
BQueue.h	394
BRefData.cpp	395
BRefData.h	395
BRtc.cpp	395
BRtc.h	396
BRWLock.cpp	396
BRWLock.h	396
BSema.cpp	396
BSema.h	396
BSemaphore.cpp	397
BSemaphore.h	397
BSocket.cpp	397
BSocket.h	398



BSpi.cpp	399
BSpi.h	399
BString.cpp	399
BString.h	405
BStringLocked.h	410
BSys.cpp	410
BSys.h	411
BTable.cpp	411
BTable.h	412
BTask.cpp	412
BTask.h	412
BThread.cpp	412
BThread.h	412
BTime.cpp	413
BTime.h	414
BTimer.cpp	414
BTimer.h	414
BTimeStamp.cpp	414
BTimeStamp.h	415
BTimeStampMs.cpp	416
BTimeStampMs.h	417
BTimeUs.cpp	417
BTimeUs.h	418
BTypes.h	418
BUrl.cpp	427
BUrl.h	428
build_x86_64/BBuffer.d	425
build_x86_64/BComms.d	425
build_x86_64/BCond.d	425
build_x86_64/BCondInt.d	425
build_x86_64/BConfig.d	425
build_x86_64/BCrc16.d	425
build_x86_64/BCrc32.d	425
build_x86_64/BDate.d	425
build_x86_64/BDebug.d	425
build_x86_64/BDict.d	425
build_x86_64/BDir.d	425
build_x86_64/BDuration.d	425
build_x86_64/BEndian.d	425
build_x86_64/BEntry.d	425
build_x86_64/BError.d	425
build_x86_64/BErrorTime.d	425
build_x86_64/BEvent.d	425
build_x86_64/BEvent1.d	426
build_x86_64/BFifoCirc.d	426
build_x86_64/BFile.d	426
build_x86_64/BFileCsv.d	426
build_x86_64/BFileData.d	426
build_x86_64/BMutex.d	426
build_x86_64/BMySQL.d	426
build_x86_64/Boap.d	364
build_x86_64/BoapMc.d	426
build_x86_64/BoapMc1.d	426
build_x86_64/BoapnsC.d	426
build_x86_64/BoapnsD.d	426
build_x86_64/BObj.d	426
build_x86_64/BObjStringFormat.d	426
build_x86_64/BPoll.d	426

build_x86_64/BRefData.d	426
build_x86_64/BRtc.d	426
build_x86_64/BRWLock.d	426
build_x86_64/BSema.d	427
build_x86_64/BSemaphore.d	427
build_x86_64/BSocket.d	427
build_x86_64/BSpi.d	427
build_x86_64/BString.d	427
build_x86_64/BSys.d	427
build_x86_64/BTable.d	427
build_x86_64/BTask.d	427
build_x86_64/BThread.d	427
build_x86_64/BTime.d	427
build_x86_64/BTimer.d	427
build_x86_64/BTimeStamp.d	427
build_x86_64/BTimeStampMs.d	427
build_x86_64/BTimeUs.d	427
build_x86_64/BUrl.d	427

## 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
```



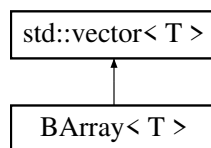
## Chapter 7

# 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

#### 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]
```

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

#### 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]
```

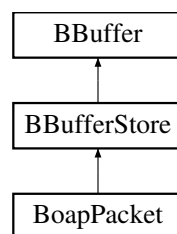
The documentation for this class was generated from the following file:

- [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]
```

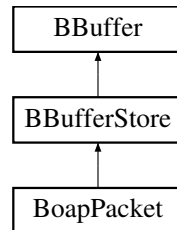
The documentation for this class was generated from the following files:

- [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 void [close](#) ()
- virtual const char \* [name](#) ()  
*The name of this interface.*
- virtual [BUInt32](#) [byteRate](#) ()  
*The byte rate of this interface.*
- 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](#) ([BUInt32](#) [eventSet](#), [BTimeout](#) [timeoutUs](#)=[BTimeoutForever](#), [BUInt32](#) [num](#)=1)
- virtual void [eventQueue](#) ([BEventQueue](#) \*[eventQueue](#), [BUInt32](#) [event](#), [BUInt32](#) [eventSet](#), [BUInt](#) [num](#)=1)
- virtual void [eventEnable](#) ([Bool](#) [on](#))  
*Enable events to be sent.*

## Protected Attributes

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

## 7.6.1 Member Enumeration Documentation

### 7.6.1.1 Flush

```
enum BComms::Flush
```

## Enumerator

FlushRead	
FlushWrite	
FlushReadWrite	

## 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 byteRate()

```
BUInt32 BComms::byteRate ( ) [virtual]
```

The byte rate of this interface.

### 7.6.3.2 close()

```
void BComms::close ( ) [virtual]
```

### 7.6.3.3 connect()

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

Create a connection.

#### 7.6.3.4 disconnect()

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

Disconnect.

#### 7.6.3.5 eventEnable()

```
void BComms::eventEnable (
    Bool on ) [virtual]
```

Enable events to be sent.

#### 7.6.3.6 eventQueue()

```
void BComms::eventQueue (
    BEventQueue * eventQueue,
    BUInt32 event,
    BUInt32 eventSet,
    BUInt num = 1 ) [virtual]
```

#### 7.6.3.7 flush()

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

#### 7.6.3.8 init()

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

#### 7.6.3.9 isConnected()

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



#### 7.6.3.10 name()

```
const char * BComms::name ( ) [virtual]
```

The name of this interface.

#### 7.6.3.11 packetMode()

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

Device is in packet mode.

#### 7.6.3.12 read()

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

#### 7.6.3.13 readAvailable()

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

#### 7.6.3.14 setPacketMode()

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

Set packet mode.

#### 7.6.3.15 setTimeout()

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

Set communication timeout.

#### 7.6.3.16 wait()

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

#### 7.6.3.17 write()

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

#### 7.6.3.18 writeAvailable()

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

#### 7.6.3.19 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

```
BUInt32 BComms::oevent [protected]
```

#### 7.6.4.3 oeventEnabled

```
Bool BComms::oeventEnabled [protected]
```

#### 7.6.4.4 oeventNum

```
BUInt BComms::oeventNum [protected]
```

#### 7.6.4.5 oeventQueue

```
BEventQueue* BComms::oeventQueue [protected]
```

#### 7.6.4.6 oeventSet

```
BUInt32 BComms::oeventSet [protected]
```

#### 7.6.4.7 opacketMode

```
Bool BComms::opacketMode [protected]
```

#### 7.6.4.8 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)

## 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 ( )
```

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](#) ()

### 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.

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.*

## 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.

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](#) ()

### 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.

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

### 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.

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.*

## 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 increment()

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

Increment. Wakes waiting.

### 7.12.2.3 operator++()

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

Increment value. Wakes waiting.



#### 7.12.2.4 operator+=( )

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

Add to value. Wakes waiting.

#### 7.12.2.5 operator--( )

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

Decrement value. Wakes waiting.

#### 7.12.2.6 operator-= ( )

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

Subtract from value. Wakes waiting.

#### 7.12.2.7 setValue( )

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

Set the value. Wakes waiting.

#### 7.12.2.8 value( )

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

Current value.

#### 7.12.2.9 waitLessThan( )

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

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

#### 7.12.2.10 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.

#### 7.12.2.11 waitMoreThanOrEqual()

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

Wait until value is at least the value given.

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

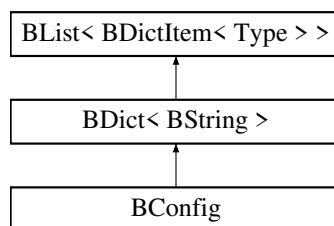
- [BCondInt.h](#)
- [BCondInt.cpp](#)

## 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](#) ()

## 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 ( )
```

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 &amp; 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 )
```

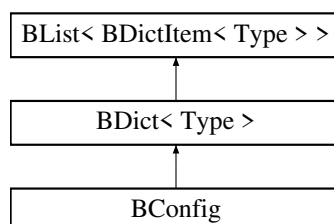
The documentation for this class was generated from the following file:

- [BDebug.h](#)

## 7.17 BDict< Type > Class Template Reference

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

Inheritance diagram for BDict< Type >:



## Public Types

- typedef [Blter](#) iterator

## Public Member Functions

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

## Additional Inherited Members

### 7.17.1 Member Typedef Documentation

#### 7.17.1.1 iterator

```
template<class Type>
typedef Blter 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 hashPrint()

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

#### 7.17.3.8 hasKey()

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

#### 7.17.3.9 insert()

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

Insert item before item.

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

#### 7.17.3.10 key()

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

#### 7.17.3.11 operator+()

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

#### 7.17.3.12 operator=()

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

#### 7.17.3.13 operator[]() [1/6]

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

#### 7.17.3.14 operator[]() [2/6]

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

#### 7.17.3.15 operator[]() [3/6]

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



## 7.17.3.16 operator[]() [4/6]

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

## 7.17.3.17 operator[]() [5/6]

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

## 7.17.3.18 operator[]() [6/6]

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

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
```

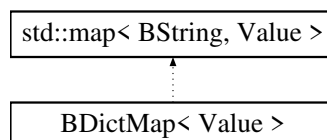
The documentation for this class was generated from the following file:

- [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:

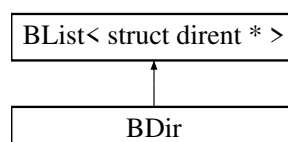
- [BDictMap.h](#)

## 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.*

## 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.

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.*

### 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.

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.*

### 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.

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

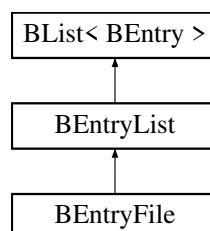
- [BEntry.h](#)
- [BEntry.cpp](#)

## 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.*

## 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.

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

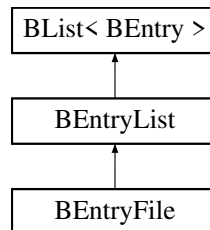
- [BEntry.h](#)
- [BEntry.cpp](#)

## 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)

## 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.

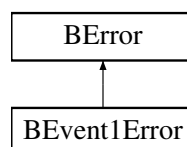
The documentation for this class was generated from the following files:

- [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.*

### 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.

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.*

### 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.

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 ()

#### 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 ( )
```

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

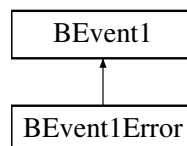
- [BEvent.h](#)
- [BEvent.cpp](#)

## 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)

### 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](#).

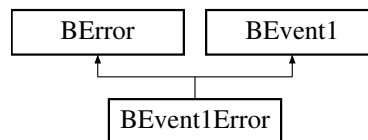
The documentation for this class was generated from the following files:

- [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 &amp; 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](#) ()

### 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.

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*

### 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.

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.*

### 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
```

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)

### 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 )
```

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

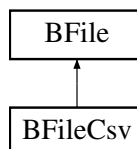
- [BFifoCirc.h](#)
- [BFifoCirc.cpp](#)

## 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)`
- `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)`

### 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.

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

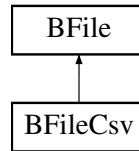
- [BFile.h](#)
- [BFile.cpp](#)



## 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)

### 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 )
```

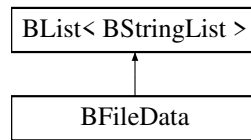
The documentation for this class was generated from the following files:

- [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)

### 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 write()

```
BError BFileData::write (
    int id,
    BStringList & csvList )
```

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

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

## 7.39 BFirmwareFileHeader Struct Reference

```
#include <BFirmware.h>
```

### Public Attributes

- [BUInt32 magic](#)
- [BUInt32 itemType](#)
- [BUInt32 fileLength](#)
- [BUInt32 checksum](#)
- [BUInt32 platform](#)
- [BUInt32 format](#)
- [BUInt32 numSegments](#)
- [BUInt32 startAddress](#)
- [BUInt8 ver0](#)
- [BUInt8 ver1](#)
- [BUInt8 ver2](#)
- [BUInt8 ver3](#)
- [BUInt32 special \[7\]](#)

### 7.39.1 Member Data Documentation

#### 7.39.1.1 checksum

```
BUInt32 BFirmwareFileHeader::checksum
```

#### 7.39.1.2 fileLength

`BUInt32 BFirmwareFileHeader::fileLength`

#### 7.39.1.3 format

`BUInt32 BFirmwareFileHeader::format`

#### 7.39.1.4 itemType

`BUInt32 BFirmwareFileHeader::itemType`

#### 7.39.1.5 magic

`BUInt32 BFirmwareFileHeader::magic`

#### 7.39.1.6 numSegments

`BUInt32 BFirmwareFileHeader::numSegments`

#### 7.39.1.7 platform

`BUInt32 BFirmwareFileHeader::platform`

#### 7.39.1.8 special

`BUInt32 BFirmwareFileHeader::special[7]`

#### 7.39.1.9 startAddress

`BUInt32 BFirmwareFileHeader::startAddress`

#### 7.39.1.10 ver0

`BUInt8 BFirmwareFileHeader::ver0`

#### 7.39.1.11 ver1

`BUInt8 BFirmwareFileHeader::ver1`

#### 7.39.1.12 ver2

`BUInt8 BFirmwareFileHeader::ver2`

#### 7.39.1.13 ver3

`BUInt8 BFirmwareFileHeader::ver3`

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

- [BFirmware.h](#)

## 7.40 BFirmwareInfo Struct Reference

```
#include <BFirmware.h>
```

### Public Attributes

- [BUInt32 magic](#)
- [BUInt32 length](#)
- [BUInt32 checksum](#)
- [BUInt8 type](#)
- [BUInt8 ver0](#)
- [BUInt8 ver1](#)
- [BUInt8 ver2](#)

### 7.40.1 Member Data Documentation

#### 7.40.1.1 checksum

`BUInt32` `BFirmwareInfo::checksum`

#### 7.40.1.2 length

`BUInt32` `BFirmwareInfo::length`

#### 7.40.1.3 magic

`BUInt32` `BFirmwareInfo::magic`

#### 7.40.1.4 type

`BUInt8` `BFirmwareInfo::type`

#### 7.40.1.5 ver0

`BUInt8` `BFirmwareInfo::ver0`

#### 7.40.1.6 ver1

`BUInt8` `BFirmwareInfo::ver1`

#### 7.40.1.7 ver2

`BUInt8` `BFirmwareInfo::ver2`

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

- [BFirmware.h](#)

## 7.41 BFirmwareSegHeader Struct Reference

```
#include <BFirmware.h>
```

### Public Attributes

- [BUInt32 magic](#)
- [BUInt32 itemType](#)
- [BUInt32 fileLength](#)
- [BUInt32 checksum](#)
- [BUInt32 platform](#)
- [BUInt32 format](#)
- [BUInt32 dataLength](#)
- [BUInt32 address](#)
- [BUInt32 length](#)
- [BUInt32 special](#) [7]

### 7.41.1 Member Data Documentation

#### 7.41.1.1 address

[BUInt32](#) BFirmwareSegHeader::address

#### 7.41.1.2 checksum

[BUInt32](#) BFirmwareSegHeader::checksum

#### 7.41.1.3 dataLength

[BUInt32](#) BFirmwareSegHeader::dataLength

#### 7.41.1.4 fileLength

[BUInt32](#) BFirmwareSegHeader::fileLength

#### 7.41.1.5 `format`

`BUInt32` `BFirmwareSegHeader::format`

#### 7.41.1.6 `itemType`

`BUInt32` `BFirmwareSegHeader::itemType`

#### 7.41.1.7 `length`

`BUInt32` `BFirmwareSegHeader::length`

#### 7.41.1.8 `magic`

`BUInt32` `BFirmwareSegHeader::magic`

#### 7.41.1.9 `platform`

`BUInt32` `BFirmwareSegHeader::platform`

#### 7.41.1.10 `special`

`BUInt32` `BFirmwareSegHeader::special[7]`

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

- [BFirmware.h](#)

## 7.42 `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](#) ()

### 7.42.1 Detailed Description

Iterator for [BList](#).

### 7.42.2 Constructor & Destructor Documentation

#### 7.42.2.1 BIter()

```
BIter::BIter (  
    BNode * i = 0 ) [inline]
```

### 7.42.3 Member Function Documentation

#### 7.42.3.1 operator BNode \*()

```
BIter::operator BNode * ( ) [inline]
```

#### 7.42.3.2 operator==( )

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

#### 7.42.3.3 valid()

```
int BIter::valid ( ) [inline]
```

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

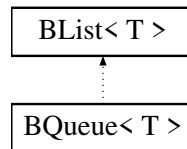
- [BList.h](#)

## 7.43 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](#) ([Blter](#) &i) const  
*Iterator to start of list.*
- [Blter begin](#) () const  
*Iterator for start of list.*
- [Blter end](#) () const  
*Iterator for end of list.*
- [Blter end](#) ([Blter](#) &i) const  
*Iterator for end of list.*
- void [next](#) ([Blter](#) &i) const  
*Iterator for next item in list.*
- void [prev](#) ([Blter](#) &i)  
*Iterator for previous item in list.*
- [Blter goTo](#) (int pos) const  
*Iterator for pos item in list.*
- int [position](#) ([Blter](#) i)  
*Position 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 [isStart](#) ([Blter](#) &i) const

- True if iterator refers to first item.*
- 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 first item.*
- void **push** (const T &i)
- Push item onto list.*
- T **pop** ()
- Pop item from list deleting item.*
- void **queueAdd** (const T &i)
- Add item to end of list.*
- T **queueGet** ()
- Get item from front of list deleting 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[]** (**BIter** i)
- const T & **operator[]** (const **BIter** &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](#)

### 7.43.1 Detailed Description

```
template<class T>
class BList< T >
```

Template based list class.

### 7.43.2 Member Typedef Documentation

#### 7.43.2.1 SortFunc

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

Prototype for sorting function.

### 7.43.3 Constructor & Destructor Documentation

#### 7.43.3.1 [BList](#)() [1/2]

```
template<class T >
BList< T >::BList ( )
```

#### 7.43.3.2 [BList](#)() [2/2]

```
template<class T>
BList< T >::BList (
    const BList< T > & l )
```

### 7.43.3.3 ~BList()

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

## 7.43.4 Member Function Documentation

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

```
template<class T>
void BList< T >::append (
    const T & item )
```

Append item to list.

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

```
template<class T>
void BList< T >::append (
    const BList< T > & l )
```

Append list to list.

### 7.43.4.3 begin()

```
template<class T >
BIter BList< T >::begin ( ) const
```

Iterator for start of list.

### 7.43.4.4 clear()

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

Clear the list.

Reimplemented in [BEntryFile](#), [BDir](#), [BEntryList](#), [BDict< Type >](#), [BQueue< T >](#), and [BQueue< BoapMcPacket >](#).

#### 7.43.4.5 del()

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

Delete specified item.

Reimplemented in [BEntryList](#), and [BDict< Type >](#).

#### 7.43.4.6 deleteFirst()

```
template<class T >
void BList< T >::deleteFirst ( )
```

Delete first item.

#### 7.43.4.7 deleteLast()

```
template<class T >
void BList< T >::deleteLast ( )
```

Delete last item.

#### 7.43.4.8 end() [1/2]

```
template<class T >
BIter BList< T >::end ( ) const
```

Iterator for end of list.

#### 7.43.4.9 end() [2/2]

```
template<class T >
BIter BList< T >::end (
    BIter & i ) const
```

Iterator for end of list.

#### 7.43.4.10 front()

```
template<class T >
T & BList< T >::front ( )
```

Get first item in list.

#### 7.43.4.11 get() [1/2]

```
template<class T >
T & BList< T >::get (
    BIter i )
```

Get item specified by iterator in list.

#### 7.43.4.12 get() [2/2]

```
template<class T >
const T & BList< T >::get (
    BIter i ) const
```

Get item specified by iterator in list.

#### 7.43.4.13 goTo()

```
template<class T >
BIter BList< T >::goTo (
    int pos ) const
```

Iterator for pos item in list.

#### 7.43.4.14 has()

```
template<class T>
int BList< T >::has (
    const T & i ) const
```

Checks if the item is in the list.

#### 7.43.4.15 insert()

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

Insert item before item.

Reimplemented in [BEntryList](#), and [BDict< Type >](#).

#### 7.43.4.16 insertAfter()

```
template<class T>
void BList< T >::insertAfter (
    BIter & i,
    const T & item )
```

Insert item after item.

#### 7.43.4.17 isEnd()

```
template<class T >
int BList< T >::isEnd (
    BIter & i ) const
```

True if iterator refers to last item.

#### 7.43.4.18 isStart()

```
template<class T >
int BList< T >::isStart (
    BIter & i ) const
```

True if iterator refers to first item.

#### 7.43.4.19 next()

```
template<class T >
void BList< T >::next (
    BIter & i ) const
```

Iterator for next item in list.



## 7.43.4.20 nodeCreate()

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

## 7.43.4.21 nodeGet() [1/2]

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

## 7.43.4.22 nodeGet() [2/2]

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

## 7.43.4.23 number()

```
template<class T >
unsigned int BList< T >::number ( ) const
```

Number of items in list.

## 7.43.4.24 operator+()

```
template<class T>
BList< T > BList< T >::operator+ (
    const BList< T > & l ) const
```

## 7.43.4.25 operator=()

```
template<class T>
BList< T > & BList< T >::operator= (
    const BList< T > & l )
```

**7.43.4.26 operator[]()** [1/4]

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

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

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

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

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

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

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

**7.43.4.30 pop()**

```
template<class T >
T BList< T >::pop ( )
```

Pop item from list deleteing item.

**7.43.4.31 position()**

```
template<class T >
int BList< T >::position (
    BIter i )
```

Postition in list item with iterator i.

#### 7.43.4.32 prev()

```
template<class T >
void BList< T >::prev (
    BIter & i )
```

Iterator for previous item in list.

#### 7.43.4.33 push()

```
template<class T>
void BList< T >::push (
    const T & i )
```

Push item onto list.

#### 7.43.4.34 queueAdd()

```
template<class T>
void BList< T >::queueAdd (
    const T & i )
```

Add item to end of list.

#### 7.43.4.35 queueGet()

```
template<class T >
T BList< T >::queueGet ( )
```

Get item from front of list deleteing item.

#### 7.43.4.36 rear()

```
template<class T >
T & BList< T >::rear ( )
```

Get last item in list.

#### 7.43.4.37 size()

```
template<class T >
unsigned int BList< T >::size ( ) const
```

Number of items in list.

#### 7.43.4.38 sort() [1/2]

```
template<class T >
void BList< T >::sort ( )
```

Sort list based on get(i) values.

#### 7.43.4.39 sort() [2/2]

```
template<class T >
void BList< T >::sort (
    SortFunc func )
```

Sort list based on Sort func.

#### 7.43.4.40 start()

```
template<class T >
void BList< T >::start (
    BIter & i ) const
```

Iterator to start of list.

#### 7.43.4.41 swap()

```
template<class T >
void BList< T >::swap (
    BIter i1,
    BIter i2 )
```

Swap two items in list.

### 7.43.5 Member Data Documentation

## 7.43.5.1 olength

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

## 7.43.5.2 onodes

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

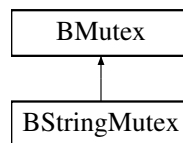
The documentation for this class was generated from the following files:

- [BList.h](#)
- [BList\\_func.h](#)

## 7.44 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)

### 7.44.1 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.

### 7.44.2 Member Enumeration Documentation

#### 7.44.2.1 Type

```
enum BMutex::Type
```

##### Enumerator

Normal	
Recursive	

### 7.44.3 Constructor & Destructor Documentation

#### 7.44.3.1 BMutex() [1/2]

```
BMutex::BMutex (
    Type type = Normal )
```

#### 7.44.3.2 BMutex() [2/2]

```
BMutex::BMutex (
    const BMutex & mutex )
```

#### 7.44.3.3 ~BMutex()

```
BMutex::~~BMutex ( )
```

### 7.44.4 Member Function Documentation

#### 7.44.4.1 lock()

```
int BMutex::lock ( )
```

Set lock, wait as necessary.

#### 7.44.4.2 operator=()

```
BMutex & BMutex::operator= (
    const BMutex & mutex )
```

#### 7.44.4.3 timedLock()

```
int BMutex::timedLock (
    int timeoutUs )
```

Set lock, wait as necessary but timeout after given time.

#### 7.44.4.4 tryLock()

```
int BMutex::tryLock ( )
```

Test the lock.

#### 7.44.4.5 unlock()

```
int BMutex::unlock ( )
```

Unlock the lock.

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

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

## 7.45 BMutexLock Class Reference

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

## Public Member Functions

- [BMutexLock](#) ([BMutex](#) &[lock](#), int doLock=0)
- [~BMutexLock](#) ()
- int [lock](#) ()
- int [unlock](#) ()

### 7.45.1 Constructor & Destructor Documentation

#### 7.45.1.1 BMutexLock()

```
BMutexLock::BMutexLock (
    BMutex & lock,
    int doLock = 0 ) [inline]
```

#### 7.45.1.2 ~BMutexLock()

```
BMutexLock::~~BMutexLock ( ) [inline]
```

### 7.45.2 Member Function Documentation

#### 7.45.2.1 lock()

```
int BMutexLock::lock ( ) [inline]
```

#### 7.45.2.2 unlock()

```
int BMutexLock::unlock ( ) [inline]
```

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

- [BMutex.h](#)

## 7.46 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)

## 7.46.1 Constructor & Destructor Documentation

### 7.46.1.1 Bmysql()

```
Bmysql::Bmysql ( )
```

### 7.46.1.2 ~Bmysql()

```
Bmysql::~~Bmysql ( )
```

## 7.46.2 Member Function Documentation

### 7.46.2.1 close()

```
void Bmysql::close ( )
```

### 7.46.2.2 db()

```
MYSQL & Bmysql::db ( )
```

#### 7.46.2.3 del()

```
BError Bmysql::del (
    BString table,
    BUInt32 id )
```

Delete record from table.

#### 7.46.2.4 escapeString()

```
BString Bmysql::escapeString (
    BString str )
```

Escapes special characters in the string.

#### 7.46.2.5 flush()

```
BError Bmysql::flush ( )
```

Flush all data to disk.

#### 7.46.2.6 get()

```
BError Bmysql::get (
    BString table,
    BString where,
    BDictString & fields )
```

#### 7.46.2.7 insert()

```
BError Bmysql::insert (
    BString table,
    BDictString fields,
    BUInt32 * id = 0 )
```

## 7.46.2.8 open()

```
BError Bmysql::open (
    BString hostName,
    BString dataBase,
    BString userName,
    BString password )
```

## 7.46.2.9 query()

```
BError Bmysql::query (
    BString cmd,
    BList< BDictString > & result )
```

## 7.46.2.10 setDebug()

```
void Bmysql::setDebug (
    int debug )
```

## 7.46.2.11 update()

```
BError Bmysql::update (
    BString table,
    BUInt32 id,
    BDictString fields )
```

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

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

## 7.47 BNameValue< T > Class Template Reference

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

### Public Member Functions

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

## 7.47.1 Constructor & Destructor Documentation

### 7.47.1.1 BNameValue() [1/2]

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

### 7.47.1.2 BNameValue() [2/2]

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

## 7.47.2 Member Function Documentation

### 7.47.2.1 getName()

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

### 7.47.2.2 getValue()

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

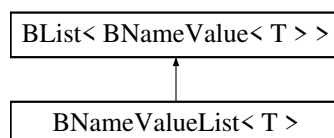
The documentation for this class was generated from the following file:

- [BNameValue.h](#)

## 7.48 BNameValueList< T > 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

### 7.48.1 Member Function Documentation

#### 7.48.1.1 find()

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

#### 7.48.1.2 findPos()

```
template<class T >
Blter BNameValueList< T >::findPos (
    BString name ) [inline]
```

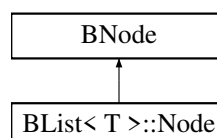
The documentation for this class was generated from the following file:

- [BNameValue.h](#)

## 7.49 BNode Class Reference

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

Inheritance diagram for BNode:



## Public Member Functions

- [BNode](#) ()

## Public Attributes

- [BNode](#) \* [next](#)
- [BNode](#) \* [prev](#)

## 7.49.1 Constructor & Destructor Documentation

### 7.49.1.1 BNode()

```
BNode::BNode ( ) [inline]
```

## 7.49.2 Member Data Documentation

### 7.49.2.1 next

```
BNode* BNode::next
```

### 7.49.2.2 prev

```
BNode* BNode::prev
```

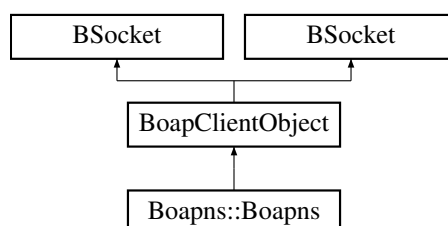
The documentation for this class was generated from the following file:

- [BList.h](#)

## 7.50 BoapClientObject Class Reference

```
#include <Boap.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 olock
- int otimeout
- int oreconnect  
*Handle an automatic reconnect on timeout.*

## Additional Inherited Members

### 7.50.1 Constructor & Destructor Documentation

#### 7.50.1.1 BoapClientObject() [1/2]

```
BoapClientObject::BoapClientObject (
    BString name = "" )
```

#### 7.50.1.2 ~BoapClientObject()

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

#### 7.50.1.3 BoapClientObject() [2/2]

```
BoapClientObject::BoapClientObject (
    BString name )
```

### 7.50.2 Member Function Documentation

#### 7.50.2.1 checkApiVersion()

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

#### 7.50.2.2 connectService() [1/2]

```
BError BoapClientObject::connectService (
    BString name )
```

Connects to the named service.



### 7.50.2.3 connectService() [2/2]

```
BError BoapClientObject::connectService (
    BString name )
```

### 7.50.2.4 disconnectService()

```
BError BoapClientObject::disconnectService ( )
```

Disconnects from the named service.

### 7.50.2.5 getServiceName()

```
BString BoapClientObject::getServiceName ( )
```

Get the name of the service.

### 7.50.2.6 handleReconnect()

```
BError BoapClientObject::handleReconnect (
    BError err ) [protected], [virtual]
```

Handle a reconnect performing autorisaztion if required.

### 7.50.2.7 performCall() [1/2]

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

Performs a RPC call to the named service.

### 7.50.2.8 performCall() [2/2]

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

#### 7.50.2.9 performRecv() [1/2]

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

Performs a receive.

#### 7.50.2.10 performRecv() [2/2]

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

#### 7.50.2.11 performSend() [1/2]

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

Performs a send to the named service.

#### 7.50.2.12 performSend() [2/2]

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

#### 7.50.2.13 ping()

```
BError BoapClientObject::ping (  
    BUInt32 & apiVersion )
```

Pings the connection and finds the remotes version number.

#### 7.50.2.14 pingLocked()

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

#### 7.50.2.15 setConnectionPriority()

```
BError BoapClientObject::setConnectionPriority (
    BoapPriority priority )
```

Sets the connection priority.

#### 7.50.2.16 setMaxLength()

```
void BoapClientObject::setMaxLength (
    BUInt32 maxLength )
```

Sets the maximum packet length.

#### 7.50.2.17 setTimeout()

```
void BoapClientObject::setTimeout (
    int timeout )
```

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

### 7.50.3 Member Data Documentation

#### 7.50.3.1 oapiVersion

```
BUInt32 BoapClientObject::oapiVersion [protected]
```

#### 7.50.3.2 oconnected

```
int BoapClientObject::oconnected [protected]
```

#### 7.50.3.3 olock

```
BMutex BoapClientObject::olock [protected]
```

#### 7.50.3.4 omaxLength

`BUInt32` BoapClientObject::omaxLength [protected]

#### 7.50.3.5 oname

`BString` BoapClientObject::oname [protected]

#### 7.50.3.6 opriority

`BoapPriority` BoapClientObject::opriority [protected]

#### 7.50.3.7 oreconnect

`int` BoapClientObject::oreconnect [protected]

Handle an automatic reconnect on timeout.

#### 7.50.3.8 orx

`BoapPacket` BoapClientObject::orx [protected]

#### 7.50.3.9 oservice

`BoapService` BoapClientObject::oservice [protected]

#### 7.50.3.10 otimeout

`int` BoapClientObject::otimeout [protected]

## 7.50.3.11 otx

`BoapPacket` `BoapClientObject::otx` [protected]

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.51 Boapns::BoapEntry Class Reference

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

### Public Member Functions

- [BoapEntry](#) ([BString](#) pname=[BString](#)(), [BString](#) phostName=[BString](#)(), [BList](#)< [BString](#) > paddressList=[BList](#)< [BString](#) >(), [BUInt32](#) pport=[BUInt32](#)(), [BUInt32](#) pservice=[BUInt32](#)())

### Public Attributes

- [BString](#) name
- [BString](#) hostName
- [BList](#)< [BString](#) > addressList
- [BUInt32](#) port
- [BUInt32](#) service

### 7.51.1 Constructor & Destructor Documentation

#### 7.51.1.1 BoapEntry()

```
Boapns::BoapEntry::BoapEntry (
    BString pname = BString(),
    BString phostName = BString(),
    BList< BString > paddressList = BList<BString >(),
    BUInt32 pport = BUInt32(),
    BUInt32 pservice = BUInt32() )
```

### 7.51.2 Member Data Documentation

#### 7.51.2.1 addressList

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

#### 7.51.2.2 hostName

`BString Boapns::BoapEntry::hostName`

#### 7.51.2.3 name

`BString Boapns::BoapEntry::name`

#### 7.51.2.4 port

`BUInt32 Boapns::BoapEntry::port`

#### 7.51.2.5 service

`BUInt32 Boapns::BoapEntry::service`

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

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

## 7.52 BoapFuncEntry Class Reference

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

### Public Member Functions

- [BoapFuncEntry](#) (int cmd, [BoapFunc](#) func)
- [BoapFuncEntry](#) (int cmd, [BoapFunc](#) func)

## Public Attributes

- [BUInt32 ocmd](#)
- [BoapFunc ofunc](#)
- [UInt32 ocmd](#)

## 7.52.1 Constructor & Destructor Documentation

### 7.52.1.1 BoapFuncEntry() [1/2]

```
BoapFuncEntry::BoapFuncEntry (
    int cmd,
    BoapFunc func )
```

### 7.52.1.2 BoapFuncEntry() [2/2]

```
BoapFuncEntry::BoapFuncEntry (
    int cmd,
    BoapFunc func )
```

## 7.52.2 Member Data Documentation

### 7.52.2.1 ocmd [1/2]

```
UInt32 BoapFuncEntry::ocmd
```

### 7.52.2.2 ocmd [2/2]

```
BUInt32 BoapFuncEntry::ocmd
```

### 7.52.2.3 ofunc

```
BoapFunc BoapFuncEntry::ofunc
```

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.53 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.*
- virtual [BError](#) [validate](#) ()
  - Validate the request.*
- [BoapMc1Packet](#) \* [packetRx](#) ()
  - Returns a reference to the current RX packet.*
- virtual [BError](#) [processRx](#) ()
  - Process any RX packets queuing them as needed.*

### Protected Member Functions

- virtual [BError](#) [processRequests](#) ()
  - Check and process any requests.*
- virtual [BError](#) [processRequest](#) ()
  - 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;.*

### 7.53.1 Constructor & Destructor Documentation

#### 7.53.1.1 BoapMc1Comms()

```
BoapMc1Comms::BoapMc1Comms (
    Bool threaded = 0,
    BUInt reqSize = 512 )
```

#### 7.53.1.2 ~BoapMc1Comms()

```
BoapMc1Comms::~~BoapMc1Comms ( ) [virtual]
```

### 7.53.2 Member Function Documentation

#### 7.53.2.1 getApiVersion()

```
BUInt32 BoapMc1Comms::getApiVersion ( )
```

Returns the API version.

#### 7.53.2.2 packetRx()

```
BoapMclPacket * BoapMclComms::packetRx ( )
```

Returns a reference to the current RX packet.

#### 7.53.2.3 packetRxData()

```
BError BoapMclComms::packetRxData (
    void * data,
    BUInt nBytes ) [protected]
```

#### 7.53.2.4 packetRxEnd()

```
BError BoapMclComms::packetRxEnd ( ) [protected]
```

#### 7.53.2.5 packetTx()

```
BError BoapMclComms::packetTx (
    BDataChunk * chunks,
    BUInt nChunks,
    BUInt16 waitCmdReply ) [protected]
```

#### 7.53.2.6 processRequest()

```
BError BoapMclComms::processRequest ( ) [protected], [virtual]
```

Check and process any request.

#### 7.53.2.7 processRequests()

```
BError BoapMclComms::processRequests ( ) [protected], [virtual]
```

Check and process any requests.

#### 7.53.2.8 processRx()

```
BError BoapMc1Comms::processRx ( ) [virtual]
```

Process any RX packets queuing them as needed.

#### 7.53.2.9 setAddress()

```
void BoapMc1Comms::setAddress (
    BUInt16 addressTo,
    BUInt16 addressFrom )
```

Sets the to and from addresses.

#### 7.53.2.10 setComms() [1/2]

```
void BoapMc1Comms::setComms (
    BComms & comms )
```

Sets the communications interface to use.

#### 7.53.2.11 setComms() [2/2]

```
void BoapMc1Comms::setComms (
    BComms * comms )
```

Sets the communications interface to use.

#### 7.53.2.12 setCommsMode()

```
void BoapMc1Comms::setCommsMode (
    Bool halfDuplex )
```

Sets half duplex mode.

#### 7.53.2.13 setTimeout()

```
BUInt32 BoapMc1Comms::setTimeout (
    BUInt32 timeoutUs )
```

Sets the call timeout returning the current value.

#### 7.53.2.14 validate()

`BError BoapMc1Comms::validate ( ) [virtual]`

Validate the request.

### 7.53.3 Member Data Documentation

#### 7.53.3.1 oaddressFrom

`BUInt16 BoapMc1Comms::oaddressFrom [protected]`

#### 7.53.3.2 oaddressTo

`BUInt16 BoapMc1Comms::oaddressTo [protected]`

#### 7.53.3.3 oapiVersion

`BUInt32 BoapMc1Comms::oapiVersion [protected]`

#### 7.53.3.4 ocomms

`BComms* BoapMc1Comms::ocomms [protected]`

#### 7.53.3.5 oerror

`BoapMc1Error BoapMc1Comms::oerror [protected]`

The call return error;.

#### 7.53.3.6 ohalfDuplex

`Bool` BoapMc1Comms::ohalfDuplex [protected]

Half duplex mode.

#### 7.53.3.7 olockCall

`BMutex` BoapMc1Comms::olockCall [protected]

Lock for RPC calls. Only one at a time.

#### 7.53.3.8 olockTx

`BMutex` BoapMc1Comms::olockTx [protected]

Lock for TX.

#### 7.53.3.9 opacketRpcCmd

`BUInt` BoapMc1Comms::opacketRpcCmd [protected]

Waiting for RPC reply to cmd.

#### 7.53.3.10 opacketRpcDoneSema

`BSemaphore` BoapMc1Comms::opacketRpcDoneSema [protected]

Wait RPC complete semaphore.

#### 7.53.3.11 opacketRpcSema

`BSemaphore` BoapMc1Comms::opacketRpcSema [protected]

Wait RPC reply semaphore.

#### 7.53.3.12 opacketRx

`BoapMc1Packet*` `BoapMc1Comms::opacketRx` [protected]

The RX packet.

#### 7.53.3.13 opacketRxBase

`BoapMc1Packet` `BoapMc1Comms::opacketRxBase` [protected]

#### 7.53.3.14 opacketTx

`BoapMc1Packet*` `BoapMc1Comms::opacketTx` [protected]

The TX packet.

#### 7.53.3.15 opacketTxBase

`BoapMc1Packet` `BoapMc1Comms::opacketTxBase` [protected]

#### 7.53.3.16 oreqSize

`BUInt32` `BoapMc1Comms::oreqSize` [protected]

The maximum request size.

#### 7.53.3.17 othreaded

`Bool` `BoapMc1Comms::othreaded` [protected]

Threaded operation.

#### 7.53.3.18 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](#)

## 7.54 BoapMc1Error Struct Reference

```
#include <BoapMc1.h>
```

### Public Attributes

- [BInt16 number](#)  
*The error number.*
- `char string [32]`  
*The error string.*

### 7.54.1 Member Data Documentation

#### 7.54.1.1 number

`BInt16 BoapMc1Error::number`

The error number.

#### 7.54.1.2 string

`char BoapMc1Error::string[32]`

The error string.

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

- [BoapMc1.h](#)

## 7.55 BoapMc1Packet Class Reference

```
#include <BoapMc1.h>
```

### Public Attributes

- [BoapMc1PacketHead](#) `head`
- char `data` [8]

### 7.55.1 Member Data Documentation

#### 7.55.1.1 data

```
char BoapMc1Packet::data[8]
```

#### 7.55.1.2 head

```
BoapMc1PacketHead BoapMc1Packet::head
```

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

- [BoapMc1.h](#)

## 7.56 BoapMc1PacketHead Struct Reference

```
#include <BoapMc1.h>
```

### Public Attributes

- [BUInt16](#) `magic`  
*Packet magic pattern.*
- [BUInt16](#) `length`  
*Total packet length including the header.*
- [BUInt16](#) `addressTo`  
*Address to send to.*
- [BUInt16](#) `addressFrom`  
*Address packet is from.*
- [BUInt16](#) `cmd`  
*The RPC command or reply number.*
- [BInt16](#) `error`  
*Error number.*
- [BUInt32](#) `checksum`  
*Packet checksum, when used.*



## 7.56.1 Member Data Documentation

### 7.56.1.1 addressFrom

`BUInt16 BoapMc1PacketHead::addressFrom`

Address packet is from.

### 7.56.1.2 addressTo

`BUInt16 BoapMc1PacketHead::addressTo`

Address to send to.

### 7.56.1.3 checksum

`BUInt32 BoapMc1PacketHead::checksum`

Packet checksum, when used.

### 7.56.1.4 cmd

`BUInt16 BoapMc1PacketHead::cmd`

The RPC command or reply number.

### 7.56.1.5 error

`BInt16 BoapMc1PacketHead::error`

Error number.

#### 7.56.1.6 length

`BUInt16 BoapMc1PacketHead::length`

Total packet length including the header.

#### 7.56.1.7 magic

`BUInt16 BoapMc1PacketHead::magic`

Packet magic pattern.

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

- [BoapMc1.h](#)

## 7.57 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

## 7.57.1 Constructor & Destructor Documentation

### 7.57.1.1 BoapMcClientObject()

```
BoapMcClientObject::BoapMcClientObject (
    BComms & comms )
```

### 7.57.1.2 ~BoapMcClientObject()

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

## 7.57.2 Member Function Documentation

### 7.57.2.1 getApiVersion()

```
BUInt32 BoapMcClientObject::getApiVersion ( )
```

Returns the API version.

### 7.57.2.2 performCall()

```
BError BoapMcClientObject::performCall ( ) [protected]
```

Performs a RPC call to the named service.

### 7.57.2.3 performRecv()

```
BError BoapMcClientObject::performRecv ( ) [protected]
```

Performs a receive.

#### 7.57.2.4 performSend()

`BError` BoapMcClientObject::performSend ( ) [protected]

Performs a send to the named service.

#### 7.57.2.5 setAddress()

```
void BoapMcClientObject::setAddress (
    BUInt8 addressTo,
    BUInt8 addressFrom )
```

### 7.57.3 Member Data Documentation

#### 7.57.3.1 oaddressFrom

`BUInt8` BoapMcClientObject::oaddressFrom [protected]

#### 7.57.3.2 oaddressTo

`BUInt8` BoapMcClientObject::oaddressTo [protected]

#### 7.57.3.3 oapiVersion

`BUInt32` BoapMcClientObject::oapiVersion [protected]

#### 7.57.3.4 ocomms

`BComms&` BoapMcClientObject::ocomms [protected]

## 7.57.3.5 opacket

`BoapMcPacket` `BoapMcClientObject::opacket` [protected]

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

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

## 7.58 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.*

## 7.58.1 Constructor & Destructor Documentation

### 7.58.1.1 BoapMcComms()

```
BoapMcComms::BoapMcComms (
    Bool threaded = 0,
    BUInt rxQueueSize = 4 )
```

### 7.58.1.2 ~BoapMcComms()

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

## 7.58.2 Member Function Documentation

### 7.58.2.1 getApiVersion()

```
BUInt32 BoapMcComms::getApiVersion ( )
```

Returns the API version.

### 7.58.2.2 packetRecv()

```
BError BoapMcComms::packetRecv (
    BoapMcPacket & packet ) [protected]
```

Receives a packet.

### 7.58.2.3 packetSend()

```
BError BoapMcComms::packetSend (
    BoapMcPacket & packet ) [protected]
```

Receives a packet.

### 7.58.2.4 performCall()

```
BError BoapMcComms::performCall ( ) [protected]
```

Performs a RPC call to the remote side.

### 7.58.2.5 performSend()

```
BError BoapMcComms::performSend ( ) [protected]
```

Performs a RPC send to the remote side.

#### 7.58.2.6 processPacket()

```
BError BoapMcComms::processPacket (
    BoapMcPacket & rx,
    BoapMcPacket & tx ) [virtual]
```

Process a recieved packet.

#### 7.58.2.7 processRequest()

```
BError BoapMcComms::processRequest (
    BTimeout timeoutUs = BTimeoutForever ) [virtual]
```

Check and process any request.

#### 7.58.2.8 processRequests()

```
BError BoapMcComms::processRequests (
    BTimeout timeoutUs = BTimeoutForever ) [virtual]
```

Check and process all requests.

#### 7.58.2.9 processRx()

```
BError BoapMcComms::processRx (
    BTimeout timeoutUs = BTimeoutForever ) [virtual]
```

Process any RX packets queuing them as needed.

!!! This should wait on comms for timeoutUs !!!

#### 7.58.2.10 setAddress()

```
void BoapMcComms::setAddress (
    BUInt8 addressTo,
    BUInt8 addressFrom )
```

Sets the to and from addresses.



#### 7.58.2.11 setComms() [1/2]

```
void BoapMcComms::setComms (
    BComms & comms )
```

Sets the communications interface to use.

#### 7.58.2.12 setComms() [2/2]

```
void BoapMcComms::setComms (
    BComms * comms )
```

Sets the communications interface to use.

#### 7.58.2.13 setCommsMode()

```
void BoapMcComms::setCommsMode (
    Bool slave,
    BUInt txQueueSize )
```

Sets slave mode.

#### 7.58.2.14 setTimeout()

```
BUInt32 BoapMcComms::setTimeout (
    BUInt32 timeoutUs )
```

Sets the call timeout returning the current value.

### 7.58.3 Member Data Documentation

#### 7.58.3.1 oaddressFrom

```
BUInt8 BoapMcComms::oaddressFrom [protected]
```

### 7.58.3.2 oaddressTo

`BUInt8` BoapMcComms::oaddressTo [protected]

### 7.58.3.3 oapiVersion

`BUInt32` BoapMcComms::oapiVersion [protected]

### 7.58.3.4 ocomms

`BComms*` BoapMcComms::ocomms [protected]

### 7.58.3.5 olockCall

`BMutex` BoapMcComms::olockCall [protected]

Lock for RPC calls. Only one at a time.

### 7.58.3.6 olockTx

`BMutex` BoapMcComms::olockTx [protected]

Lock for TX.

### 7.58.3.7 opacket

`BoapMcPacket` BoapMcComms::opacket [protected]

Packet RX buffer.

### 7.58.3.8 opacketReqQueue

`BQueue<BoapMcPacket>` BoapMcComms::opacketReqQueue [protected]

Packet RX buffer queue for requests.

### 7.58.3.9 opacketReqRx

`BoapMcPacket` `BoapMcComms::opacketReqRx` [protected]

Packet RX buffer for requests.

### 7.58.3.10 opacketReqTx

`BoapMcPacket` `BoapMcComms::opacketReqTx` [protected]

Packet TX buffer for requests.

### 7.58.3.11 opacketRx

`BoapMcPacket` `BoapMcComms::opacketRx` [protected]

Packet RX buffer for calls.

### 7.58.3.12 opacketRxSema

`BSemaphore` `BoapMcComms::opacketRxSema` [protected]

Wait RX semaphore.

### 7.58.3.13 opacketTx

`BoapMcPacket` `BoapMcComms::opacketTx` [protected]

Packet TX buffer for calls.

### 7.58.3.14 opacketTxQueue

`BFifo<BoapMcPacket>` `BoapMcComms::opacketTxQueue` [protected]

Packet TX Queue.

#### 7.58.3.15 opacketTxQueueWriteNum

[BSemaphoreCount](#) BoapMcComms::opacketTxQueueWriteNum [protected]

Packet TX Queue number.

#### 7.58.3.16 opacketTxSema

[BSemaphore](#) BoapMcComms::opacketTxSema [protected]

Wait for TX semaphore.

#### 7.58.3.17 oslave

[Bool](#) BoapMcComms::oslave [protected]

Set slave mode.

#### 7.58.3.18 othreaded

[Bool](#) BoapMcComms::othreaded [protected]

#### 7.58.3.19 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](#)

## 7.59 BoapMcPacket Class Reference

```
#include <BoapMc.h>
```

## Public Attributes

- [BoapMcPacketHead head](#)
- char [data](#) [256 - sizeof([BoapMcPacketHead](#))]

### 7.59.1 Member Data Documentation

#### 7.59.1.1 data

```
char BoapMcPacket::data[256 - sizeof(BoapMcPacketHead) ]
```

#### 7.59.1.2 head

```
BoapMcPacketHead BoapMcPacket::head
```

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

- [BoapMc.h](#)

## 7.60 BoapMcPacketHead Struct Reference

```
#include <BoapMc.h>
```

## Public Attributes

- [BUInt8 length](#)
- [BUInt8 addressTo](#)
- [BUInt8 addressFrom](#)
- [BUInt8 cmd](#)
- [BUInt16 error](#)
- [BUInt16 checksum](#)

### 7.60.1 Member Data Documentation

#### 7.60.1.1 addressFrom

```
BUInt8 BoapMcPacketHead::addressFrom
```

#### 7.60.1.2 addressTo

`BUInt8 BoapMcPacketHead::addressTo`

#### 7.60.1.3 checksum

`BUInt16 BoapMcPacketHead::checksum`

#### 7.60.1.4 cmd

`BUInt8 BoapMcPacketHead::cmd`

#### 7.60.1.5 error

`BUInt16 BoapMcPacketHead::error`

#### 7.60.1.6 length

`BUInt8 BoapMcPacketHead::length`

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

- [BoapMc.h](#)

## 7.61 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](#)

## 7.61.1 Constructor & Destructor Documentation

### 7.61.1.1 BoapMcServiceObject()

```
BoapMcServiceObject::BoapMcServiceObject ( )
```

### 7.61.1.2 ~BoapMcServiceObject()

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

## 7.61.2 Member Function Documentation

### 7.61.2.1 process()

```
BError BoapMcServiceObject::process (
    BoapMcPacket & rx,
    BoapMcPacket & tx ) [virtual]
```

### 7.61.2.2 processEvent()

```
BError BoapMcServiceObject::processEvent (
    BoapMcPacket & rx ) [virtual]
```

### 7.61.2.3 sendEvent()

```
BError BoapMcServiceObject::sendEvent (
    BoapMcPacket & tx ) [protected]
```

### 7.61.3 Member Data Documentation

#### 7.61.3.1 oapiVersion

`BUInt32 BoapMcServiceObject::oapiVersion [protected]`

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

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

## 7.62 BoapMcSignalObject Class Reference

```
#include <BoapMc.h>
```

### Public Member Functions

- [BoapMcSignalObject](#) ([BComms](#) &comms)

### Protected Member Functions

- [BError performSend](#) ([BoapMcPacket](#) &tx)

### Protected Attributes

- [BComms](#) & [ocomms](#)

### 7.62.1 Constructor & Destructor Documentation

#### 7.62.1.1 BoapMcSignalObject()

```
BoapMcSignalObject::BoapMcSignalObject (
    BComms & comms )
```

### 7.62.2 Member Function Documentation



## 7.62.2.1 performSend()

```
BError BoapMcSignalObject::performSend (
    BoapMcPacket & tx ) [protected]
```

## 7.62.3 Member Data Documentation

## 7.62.3.1 ocomms

```
BComms& BoapMcSignalObject::ocomms [protected]
```

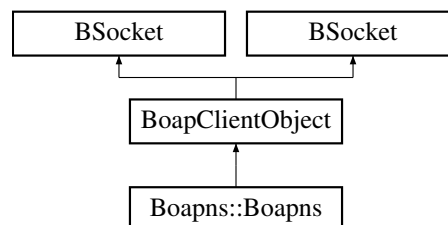
The documentation for this class was generated from the following files:

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

## 7.63 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

## 7.63.1 Constructor &amp; Destructor Documentation

### 7.63.1.1 Boapns()

```
Boapns::Boapns::Boapns (
    BString name = "" )
```

## 7.63.2 Member Function Documentation

### 7.63.2.1 addEntry()

```
BError Boapns::Boapns::addEntry (
    BoapEntry entry )
```

### 7.63.2.2 delEntry()

```
BError Boapns::Boapns::delEntry (
    BString name )
```

### 7.63.2.3 getEntry()

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

### 7.63.2.4 getEntryList()

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

### 7.63.2.5 getNewName()

```
BError Boapns::Boapns::getNewName (
    BString & name )
```

## 7.63.2.6 getVersion()

```
BError Boapns::Boapns::getVersion (
    BString & version )
```

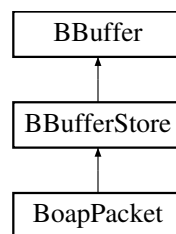
The documentation for this class was generated from the following files:

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

## 7.64 BoapPacket Class Reference

```
#include <Boap.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](#))

## Additional Inherited Members

### 7.64.1 Constructor & Destructor Documentation

#### 7.64.1.1 [BoapPacket\(\)](#) [1/2]

`BoapPacket::BoapPacket ( )`

#### 7.64.1.2 [~BoapPacket\(\)](#) [1/2]

`BoapPacket::~~BoapPacket ( )`

#### 7.64.1.3 [BoapPacket\(\)](#) [2/2]

`BoapPacket::BoapPacket ( )`

#### 7.64.1.4 [~BoapPacket\(\)](#) [2/2]

`BoapPacket::~~BoapPacket ( )`

### 7.64.2 Member Function Documentation

#### 7.64.2.1 data()

```
char * BoapPacket::data ( )
```

#### 7.64.2.2 getCmd()

```
BUInt32 BoapPacket::getCmd ( )
```

#### 7.64.2.3 nbytes()

```
int BoapPacket::nbytes ( )
```

#### 7.64.2.4 peekHead()

```
int BoapPacket::peekHead (
    BoapPacketHead & head )
```

#### 7.64.2.5 pop() [1/10]

```
int BoapPacket::pop (
    Int8 & v )
```

#### 7.64.2.6 pop() [2/10]

```
int BoapPacket::pop (
    UInt8 & v )
```

#### 7.64.2.7 pop() [3/10]

```
int BoapPacket::pop (
    Int16 & v )
```

**7.64.2.8 pop()** [4/10]

```
int BoapPacket::pop (
    UInt16 & v )
```

**7.64.2.9 pop()** [5/10]

```
int BoapPacket::pop (
    Int32 & v )
```

**7.64.2.10 pop()** [6/10]

```
int BoapPacket::pop (
    UInt32 & v )
```

**7.64.2.11 pop()** [7/10]

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

**7.64.2.12 pop()** [8/10]

```
int BoapPacket::pop (
    Double & v )
```

**7.64.2.13 pop()** [9/10]

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

**7.64.2.14 pop()** [10/10]

```
int BoapPacket::pop (
    UInt32 nBytes,
    void * data )
```

**7.64.2.15 popHead()** [1/2]

```
int BoapPacket::popHead (
    BoapPacketHead & head )
```

**7.64.2.16 popHead()** [2/2]

```
int BoapPacket::popHead (
    BoapPacketHead & head )
```

**7.64.2.17 push()** [1/10]

```
int BoapPacket::push (
    Int8 v )
```

**7.64.2.18 push()** [2/10]

```
int BoapPacket::push (
    UInt8 v )
```

**7.64.2.19 push()** [3/10]

```
int BoapPacket::push (
    Int16 v )
```

**7.64.2.20 push()** [4/10]

```
int BoapPacket::push (
    UInt16 v )
```

**7.64.2.21 push()** [5/10]

```
int BoapPacket::push (
    Int32 v )
```

**7.64.2.22** `push()` [6/10]

```
int BoapPacket::push (
    UInt32 v )
```

**7.64.2.23** `push()` [7/10]

```
int BoapPacket::push (
    BString & v )
```

**7.64.2.24** `push()` [8/10]

```
int BoapPacket::push (
    Double v )
```

**7.64.2.25** `push()` [9/10]

```
int BoapPacket::push (
    BError & v )
```

**7.64.2.26** `push()` [10/10]

```
int BoapPacket::push (
    UInt32 nBytes,
    const void * data )
```

**7.64.2.27** `pushHead()` [1/2]

```
int BoapPacket::pushHead (
    BoapPacketHead & head )
```

**7.64.2.28** `pushHead()` [2/2]

```
int BoapPacket::pushHead (
    BoapPacketHead & head )
```



#### 7.64.2.29 `resize()`

```
int BoapPacket::resize (
    int size )
```

#### 7.64.2.30 `setData()`

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

#### 7.64.2.31 `updateHead()`

```
void BoapPacket::updateHead ( )
```

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.65 BoapPacketHead Struct Reference

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

### Public Attributes

- [BUInt32 type](#)
- [BUInt32 length](#)
- [BUInt32 service](#)
- [BUInt32 cmd](#)
- [UInt32 length](#)
- [BoapType type](#)
- [BoapService service](#)
- [UInt32 cmd](#)
- [UInt32 reserved](#) [12]

### 7.65.1 Member Data Documentation

**7.65.1.1 cmd** [1/2]

[UInt32](#) BoapPacketHead::cmd

**7.65.1.2 cmd** [2/2]

[BUInt32](#) BoapPacketHead::cmd

**7.65.1.3 length** [1/2]

[UInt32](#) BoapPacketHead::length

**7.65.1.4 length** [2/2]

[BUInt32](#) BoapPacketHead::length

**7.65.1.5 reserved**

[UInt32](#) BoapPacketHead::reserved[12]

**7.65.1.6 service** [1/2]

[BoapService](#) BoapPacketHead::service

**7.65.1.7 service** [2/2]

[BUInt32](#) BoapPacketHead::service

**7.65.1.8 type** [1/2]

[BUInt32](#) BoapPacketHead::type

7.65.1.9 `type` [2/2]

`BoapType` `BoapPacketHead::type`

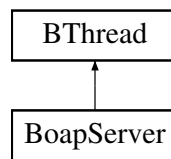
The documentation for this struct was generated from the following files:

- [Boap.h](#)
- [BoapSimple.h](#)

## 7.66 BoapServer Class Reference

```
#include <Boap.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](#)

## 7.66.1 Member Enumeration Documentation

### 7.66.1.1 anonymous enum

anonymous enum

#### Enumerator

NOTHEADS	
THREADED	

## 7.66.2 Constructor & Destructor Documentation

### 7.66.2.1 BoapServer() [1/2]

BoapServer::BoapServer ( )

### 7.66.2.2 ~BoapServer()

BoapServer::~~BoapServer ( ) [virtual]

### 7.66.2.3 BoapServer() [2/2]

BoapServer::BoapServer ( )

## 7.66.3 Member Function Documentation

### 7.66.3.1 addObject() [1/2]

```
BError BoapServer::addObject (
    BoapServiceObject * object )
```

### 7.66.3.2 addObject() [2/2]

```
BError BoapServer::addObject (
    BoapServiceObject * object ) [virtual]
```

### 7.66.3.3 clientGone()

```
void BoapServer::clientGone (
    BoapServerConnection * client ) [virtual]
```

### 7.66.3.4 closeConnections()

```
void BoapServer::closeConnections ( )
```

### 7.66.3.5 getConnectionsNumber()

```
int BoapServer::getConnectionsNumber ( )
```

### 7.66.3.6 getEventSocket() [1/2]

```
BSocket& BoapServer::getEventSocket ( )
```

### 7.66.3.7 getEventSocket() [2/2]

```
BSocket & BoapServer::getEventSocket ( )
```

**7.66.3.8** `getHostName()` [1/2]

```
BString BoapServer::getHostName ( )
```

**7.66.3.9** `getHostName()` [2/2]

```
BString BoapServer::getHostName ( )
```

**7.66.3.10** `getSocket()` [1/2]

```
BSocket& BoapServer::getSocket ( )
```

**7.66.3.11** `getSocket()` [2/2]

```
BSocket & BoapServer::getSocket ( )
```

**7.66.3.12** `init()` [1/2]

```
BError BoapServer::init (
    int boapNs = 0 )
```

**7.66.3.13** `init()` [2/2]

```
BError BoapServer::init (
    BString boapNsHost = "",
    int port = 0,
    int threaded = 0,
    int isBoapns = 0 ) [virtual]
```

**7.66.3.14** `newConnection()`

```
BoapServerConnection * BoapServer::newConnection (
    int fd,
    BSocketAddressINET address ) [virtual]
```

**7.66.3.15 process()** [1/2]

```
BError BoapServer::process (
    int fd )
```

**7.66.3.16 process()** [2/2]

```
BError BoapServer::process (
    BoapServerConnection * conn,
    BoapPacket & rx,
    BoapPacket & tx ) [virtual]
```

**7.66.3.17 processEvent()** [1/4]

```
BError BoapServer::processEvent (
    BoapPacket & rx )
```

**7.66.3.18 processEvent()** [2/4]

```
BError BoapServer::processEvent (
    int fd )
```

**7.66.3.19 processEvent()** [3/4]

```
BError BoapServer::processEvent (
    BoapPacket & rx ) [virtual]
```

**7.66.3.20 processEvent()** [4/4]

```
BError BoapServer::processEvent (
    int fd ) [virtual]
```

**7.66.3.21 run()** [1/2]

```
BError BoapServer::run ( )
```

#### 7.66.3.22 `run()` [2/2]

```
BError BoapServer::run (
    int inThread = 0 ) [virtual]
```

#### 7.66.3.23 `sendEvent()` [1/2]

```
BError BoapServer::sendEvent (
    BoapPacket & tx )
```

#### 7.66.3.24 `sendEvent()` [2/2]

```
BError BoapServer::sendEvent (
    BoapPacket & tx ) [virtual]
```

### 7.66.4 Member Data Documentation

#### 7.66.4.1 `onumOperations`

```
BUInt64 BoapServer::onumOperations
```

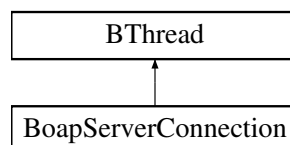
The documentation for this class was generated from the following files:

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

### 7.67 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.*

## 7.67.1 Constructor & Destructor Documentation

### 7.67.1.1 BoapServerConnection()

```
BoapServerConnection::BoapServerConnection (
    BoapServer & boapServer,
    int fd )
```

### 7.67.1.2 ~BoapServerConnection()

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

## 7.67.2 Member Function Documentation

### 7.67.2.1 getHead()

```
BError BoapServerConnection::getHead (
    BoapPacketHead & head ) [virtual]
```

### 7.67.2.2 getSocket()

```
BSocket & BoapServerConnection::getSocket ( ) [virtual]
```

### 7.67.2.3 init()

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

Initialise connection.

### 7.67.2.4 process()

```
BError BoapServerConnection::process ( ) [virtual]
```

### 7.67.2.5 setMaxLength()

```
void BoapServerConnection::setMaxLength (
    BUInt32 maxLength ) [virtual]
```

### 7.67.2.6 validate()

```
BError BoapServerConnection::validate ( ) [virtual]
```

Validate the connection.

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

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

## 7.68 BoapServiceEntry Class Reference

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

### Public Member Functions

- [BoapServiceEntry](#) ([BoapService](#) service=0, [BoapServiceObject](#) \*object=0)
- [BoapServiceEntry](#) ([BoapService](#) service=0, [BoapServiceObject](#) \*object=0)

### Public Attributes

- [BoapService](#) oservice
- [BoapServiceObject](#) \* oobject

## 7.68.1 Constructor & Destructor Documentation

### 7.68.1.1 BoapServiceEntry() [1/2]

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

### 7.68.1.2 BoapServiceEntry() [2/2]

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

## 7.68.2 Member Data Documentation

### 7.68.2.1 oobject

```
BoapServiceObject * BoapServiceEntry::oobject
```

### 7.68.2.2 oservice

```
BoapService BoapServiceEntry::oservice
```

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

- [Boap.h](#)
- [BoapSimple.h](#)

## 7.69 BoapServiceObject Class Reference

```
#include <Boap.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

## 7.69.1 Constructor & Destructor Documentation

### 7.69.1.1 [BoapServiceObject](#)() [1/2]

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

### 7.69.1.2 [~BoapServiceObject](#)() [1/2]

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

### 7.69.1.3 BoapServiceObject() [2/2]

```
BoapServiceObject::BoapServiceObject (
    BoapServer & server,
    BString name )
```

### 7.69.1.4 ~BoapServiceObject() [2/2]

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

## 7.69.2 Member Function Documentation

### 7.69.2.1 doConnectionPriority()

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

### 7.69.2.2 doPing()

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

### 7.69.2.3 name() [1/2]

```
BString BoapServiceObject::name ( )
```

### 7.69.2.4 name() [2/2]

```
BString BoapServiceObject::name ( )
```

#### 7.69.2.5 process() [1/2]

```
BError BoapServiceObject::process (
    BoapPacket & rx,
    BoapPacket & tx )
```

#### 7.69.2.6 process() [2/2]

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

#### 7.69.2.7 processEvent() [1/4]

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

#### 7.69.2.8 processEvent() [2/4]

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

#### 7.69.2.9 processEvent() [3/4]

```
BError BoapServiceObject::processEvent (
    BString objectName,
    BString name,
    BInt32 arg ) [virtual]
```

#### 7.69.2.10 processEvent() [4/4]

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

**7.69.2.11 sendEvent()** [1/4]

```
BError BoapServiceObject::sendEvent (
    BString signalName,
    Int32 arg )
```

**7.69.2.12 sendEvent()** [2/4]

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

**7.69.2.13 sendEvent()** [3/4]

```
BError BoapServiceObject::sendEvent (
    BString signalName,
    BInt32 arg )
```

**7.69.2.14 sendEvent()** [4/4]

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

**7.69.2.15 setName()**

```
BError BoapServiceObject::setName (
    BString name )
```

**7.69.3 Member Data Documentation****7.69.3.1 oapiVersion**

```
BUInt32 BoapServiceObject::oapiVersion [protected]
```

### 7.69.3.2 ofuncList

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

### 7.69.3.3 oname

```
BString BoapServiceObject::oname [protected]
```

### 7.69.3.4 oserver

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

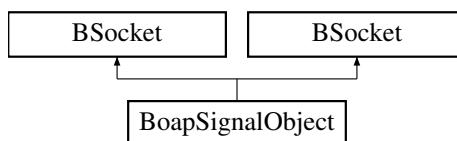
The documentation for this class was generated from the following files:

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.70 BoapSignalObject Class Reference

```
#include <Boap.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

### 7.70.1 Constructor & Destructor Documentation

#### 7.70.1.1 BoapSignalObject() [1/2]

```
BoapSignalObject::BoapSignalObject ( )
```

#### 7.70.1.2 BoapSignalObject() [2/2]

```
BoapSignalObject::BoapSignalObject ( )
```

### 7.70.2 Member Function Documentation

#### 7.70.2.1 performSend() [1/2]

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

#### 7.70.2.2 performSend() [2/2]

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

### 7.70.3 Member Data Documentation

### 7.70.3.1 orx

`BoapPacket` `BoapSignalObject::orx` [protected]

### 7.70.3.2 otx

`BoapPacket` `BoapSignalObject::otx` [protected]

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

- [Boap.h](#)
- [BoapSimple.h](#)
- [Boap.cpp](#)
- [BoapSimple.cc](#)

## 7.71 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.*

### 7.71.1 Constructor & Destructor Documentation

#### 7.71.1.1 BObj()

```
BObj::BObj ( )
```

### 7.71.1.2 ~BObj()

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

## 7.71.2 Member Function Documentation

### 7.71.2.1 getDebugString()

```
BString BObj::getDebugString ( ) [virtual]
```

Returns contents as a debug string.

### 7.71.2.2 getMember()

```
BError BObj::getMember (
    BString name,
    BString & value ) [virtual]
```

### 7.71.2.3 getMembers() [1/2]

```
const BObjMember * BObj::getMembers ( ) const [virtual]
```

### 7.71.2.4 getMembers() [2/2]

```
BError BObj::getMembers (
    BDictString & members ) [virtual]
```

### 7.71.2.5 getType()

```
const char * BObj::getType ( ) const [virtual]
```

#### 7.71.2.6 membersPrint()

```
void BObj::membersPrint ( ) const [virtual]
```

Prints out members.

#### 7.71.2.7 setMember()

```
BError BObj::setMember (
    BString name,
    BString value ) [virtual]
```

#### 7.71.2.8 setMembers()

```
BError BObj::setMembers (
    BDictString & members ) [virtual]
```

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

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

## 7.72 BObjMember Struct Reference

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

### Public Attributes

- [BType type](#)
- [BTypeComp typeComp](#)
- [BUInt16 dataOffset](#)
- [BUInt16 size](#)
- [const char \\* typeName](#)
- [const char \\* name](#)

### 7.72.1 Member Data Documentation

#### 7.72.1.1 dataOffset

```
BUInt16 BObjMember::dataOffset
```

#### 7.72.1.2 name

```
const char* BObjMember::name
```

#### 7.72.1.3 size

```
BUInt16 BObjMember::size
```

#### 7.72.1.4 type

```
BType BObjMember::type
```

#### 7.72.1.5 typeComp

```
BTypeComp BObjMember::typeComp
```

#### 7.72.1.6 typeName

```
const char* BObjMember::typeName
```

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

- [BTypes.h](#)

## 7.73 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](#) ()

### 7.73.1 Detailed Description

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

### 7.73.2 Member Typedef Documentation

#### 7.73.2.1 PollFd

```
typedef struct pollfd BPoll::PollFd
```

### 7.73.3 Constructor & Destructor Documentation

#### 7.73.3.1 BPoll()

```
BPoll::BPoll ( )
```

#### 7.73.3.2 ~BPoll()

```
BPoll::~~BPoll ( )
```

### 7.73.4 Member Function Documentation

#### 7.73.4.1 append()

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

Append a file descriptor to polling list.

#### 7.73.4.2 clear()

```
void BPoll::clear ( )
```

#### 7.73.4.3 delFd()

```
void BPoll::delFd (
    int fd )
```

Remove a file descriptor from polling list.

#### 7.73.4.4 doPoll()

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

Perform polling operation.

#### 7.73.4.5 doPollEvents()

```
BError BPoll::doPollEvents (
    int & fd,
    int & events,
    int timeoutUs = -1 )
```

Perform polling operation and return events.

#### 7.73.4.6 getPollFds()

```
BPoll::PollFd * BPoll::getPollFds ( )
```

#### 7.73.4.7 getPollFdsNum()

```
int BPoll::getPollFdsNum ( )
```

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

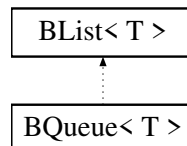
- [BPoll.h](#)
- [BPoll.cpp](#)

## 7.74 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.*

#### 7.74.1 Detailed Description

```
template<class T>
class BQueue< T >
```

Queue class.

#### 7.74.2 Constructor & Destructor Documentation



#### 7.74.2.1 BQueue()

```
template<class T >
BQueue< T >::BQueue (
    BUInt size )
```

#### 7.74.2.2 ~BQueue()

```
template<class T >
BQueue< T >::~~BQueue ( )
```

### 7.74.3 Member Function Documentation

#### 7.74.3.1 clear()

```
template<class T >
void BQueue< T >::clear ( ) [virtual]
```

Clear the queue.

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

#### 7.74.3.2 read()

```
template<class T>
BError BQueue< T >::read (
    T & v,
    BTimeout timeout = BTimeoutForever )
```

Get an item from the queue.

#### 7.74.3.3 readAvailable()

```
template<class T >
BUInt BQueue< T >::readAvailable ( ) const
```

#### 7.74.3.4 write()

```
template<class T>
BError BQueue< T >::write (
    const T & v,
    BTimeout timeout = BTimeoutForever )
```

Append an item onto the queue.

#### 7.74.3.5 writeAvailable()

```
template<class T >
BUInt BQueue< T >::writeAvailable ( ) const
```

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

- [BQueue.h](#)

## 7.75 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 refCount = 1.*

### 7.75.1 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.

## 7.75.2 Constructor & Destructor Documentation

### 7.75.2.1 BRefData() [1/3]

```
BRefData::BRefData ( )
```

### 7.75.2.2 BRefData() [2/3]

```
BRefData::BRefData (
    int len )
```

### 7.75.2.3 BRefData() [3/3]

```
BRefData::BRefData (
    const BRefData & refData )
```

### 7.75.2.4 ~BRefData()

```
BRefData::~BRefData ( )
```

## 7.75.3 Member Function Documentation

### 7.75.3.1 addRef()

```
BRefData * BRefData::addRef ( )
```

Increment the reference counter.

### 7.75.3.2 copy()

```
BRefData * BRefData::copy ( )
```

Create a copy of this reference for writing, if necessary.

#### 7.75.3.3 data()

```
char* BRefData::data ( ) [inline]
```

Return the raw data pointer.

#### 7.75.3.4 deleteRef()

```
int BRefData::deleteRef ( )
```

Decrement the reference counter.

#### 7.75.3.5 len()

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

Return the length in bytes.

#### 7.75.3.6 operator=()

```
BRefData & BRefData::operator= (
    const BRefData & refData )
```

#### 7.75.3.7 setLen()

```
void BRefData::setLen (
    int len )
```

Set the length in bytes. Note should only be used if orefCount = 1.

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

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

## 7.76 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.*

### 7.76.1 Detailed Description

Realtime clock.

### 7.76.2 Constructor & Destructor Documentation

#### 7.76.2.1 BRtc()

```
BRtc::BRtc ( )
```

#### 7.76.2.2 ~BRtc()

```
BRtc::~~BRtc ( )
```

### 7.76.3 Member Function Documentation

#### 7.76.3.1 init()

```
BError BRtc::init (
    int rate )
```

Setup interrupt rate.

### 7.76.3.2 wait()

```
void BRtc::wait (
    int delayUs )
```

Wait specified uS.

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

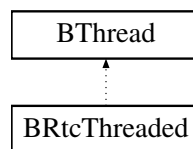
- [BRtc.h](#)
- [BRtc.cpp](#)

## 7.77 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.*

### 7.77.1 Detailed Description

Threaded real time clock.

### 7.77.2 Constructor & Destructor Documentation

#### 7.77.2.1 BRtcThreaded()

```
BRtcThreaded::BRtcThreaded ( )
```

## 7.77.2.2 ~BRtcThreaded()

```
BRtcThreaded::~BRtcThreaded ( )
```

## 7.77.3 Member Function Documentation

## 7.77.3.1 init()

```
BError BRtcThreaded::init (
    int rate )
```

Setup interrupt rate.

## 7.77.3.2 wait()

```
void BRtcThreaded::wait (
    int delayUs )
```

Wait specified uS.

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

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

## 7.78 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)

### 7.78.1 Detailed Description

thread read-write locks

### 7.78.2 Constructor & Destructor Documentation

#### 7.78.2.1 BRWLock() [1/2]

```
BRWLock::BRWLock ( )
```

#### 7.78.2.2 BRWLock() [2/2]

```
BRWLock::BRWLock (
    const BRWLock & rlock )
```

#### 7.78.2.3 ~BRWLock()

```
BRWLock::~BRWLock ( )
```

### 7.78.3 Member Function Documentation

#### 7.78.3.1 operator=()

```
BRWLock & BRWLock::operator= (
    const BRWLock & rlock )
```

#### 7.78.3.2 rdLock()

```
int BRWLock::rdLock ( )
```

Set lock, wait if necessary.



### 7.78.3.3 tryRdLock()

```
int BRWLock::tryRdLock ( )
```

Test the lock.

### 7.78.3.4 tryWrLock()

```
int BRWLock::tryWrLock ( )
```

Test the lock.

### 7.78.3.5 unlock()

```
int BRWLock::unlock ( )
```

Unlock the lock.

### 7.78.3.6 wrLock()

```
int BRWLock::wrLock ( )
```

Set lock, wait if necessary.

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

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

## 7.79 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)

### 7.79.1 Detailed Description

Sempahore class.

### 7.79.2 Constructor & Destructor Documentation

#### 7.79.2.1 [BSema\(\)](#) [1/2]

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

#### 7.79.2.2 [BSema\(\)](#) [2/2]

```
BSema::BSema (
    const BSema & sema )
```

#### 7.79.2.3 [~BSema\(\)](#)

```
BSema::~~BSema ( )
```

### 7.79.3 Member Function Documentation

#### 7.79.3.1 getValue()

```
int BSema::getValue ( ) const
```

#### 7.79.3.2 operator=()

```
BSema & BSema::operator= (
    const BSema & sema )
```

#### 7.79.3.3 post()

```
int BSema::post ( )
```

Post condition.

#### 7.79.3.4 timedWait()

```
int BSema::timedWait (
    int timeUs )
```

Wait for condition with timeout.

#### 7.79.3.5 tryWait()

```
int BSema::tryWait ( )
```

Test for the condition.

#### 7.79.3.6 wait()

```
int BSema::wait ( )
```

Wait for contition.

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

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

## 7.80 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)

### 7.80.1 Detailed Description

Semaphore class.

### 7.80.2 Constructor & Destructor Documentation

#### 7.80.2.1 BSemaphore() [1/2]

```
BSemaphore::BSemaphore ( )
```

#### 7.80.2.2 BSemaphore() [2/2]

```
BSemaphore::BSemaphore (
    const BSemaphore & semaphore )
```

#### 7.80.2.3 ~BSemaphore()

```
BSemaphore::~~BSemaphore ( )
```

### 7.80.3 Member Function Documentation

#### 7.80.3.1 getValue()

```
int BSemaphore::getValue ( ) const
```

#### 7.80.3.2 operator=()

```
BSemaphore & BSemaphore::operator= (
    const BSemaphore & semaphore )
```

#### 7.80.3.3 set()

```
void BSemaphore::set ( )
```

Set the semaphore.

#### 7.80.3.4 wait()

```
Bool BSemaphore::wait (
    BTimeout timeoutUs = BTimeoutForever )
```

Wait for the semaphore.

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

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

## 7.81 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)

## 7.81.1 Constructor & Destructor Documentation

### 7.81.1.1 [BSemaphoreBool](#)() [1/2]

```
BSemaphoreBool::BSemaphoreBool ( )
```

### 7.81.1.2 [BSemaphoreBool](#)() [2/2]

```
BSemaphoreBool::BSemaphoreBool (
    const BSemaphoreBool & semaphore )
```

### 7.81.1.3 [~BSemaphoreBool](#)()

```
BSemaphoreBool::~~BSemaphoreBool ( )
```

## 7.81.2 Member Function Documentation

### 7.81.2.1 [clear](#)()

```
void BSemaphoreBool::clear ( )
```

### 7.81.2.2 operator int()

```
BSemaphoreBool::operator int ( )
```

### 7.81.2.3 operator=()

```
BSemaphoreBool & BSemaphoreBool::operator= (
    Bool on )
```

### 7.81.2.4 operator==()

```
int BSemaphoreBool::operator== (
    Bool on )
```

### 7.81.2.5 set()

```
void BSemaphoreBool::set (
    Bool on = 1 )
```

### 7.81.2.6 value()

```
Bool BSemaphoreBool::value ( )
```

### 7.81.2.7 wait()

```
Bool BSemaphoreBool::wait (
    Bool v = 1,
    BTimeout timeoutUs = BTimeoutForever )
```

Wait for the semaphore.

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

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

## 7.82 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)

### 7.82.1 Constructor & Destructor Documentation

#### 7.82.1.1 BSemaphoreCount() [1/2]

```
BSemaphoreCount::BSemaphoreCount ( )
```

#### 7.82.1.2 BSemaphoreCount() [2/2]

```
BSemaphoreCount::BSemaphoreCount (
    const BSemaphoreCount & semaphore )
```

#### 7.82.1.3 ~BSemaphoreCount()

```
BSemaphoreCount::~~BSemaphoreCount ( )
```

### 7.82.2 Member Function Documentation



### 7.82.2.1 add()

```
void BSemaphoreCount::add (
    int v = 1 )
```

Set the semaphore.

### 7.82.2.2 operator=()

```
BSemaphoreCount & BSemaphoreCount::operator= (
    const BSemaphoreCount & semaphore )
```

### 7.82.2.3 setValue()

```
void BSemaphoreCount::setValue (
    BUInt v )
```

### 7.82.2.4 take()

```
Bool BSemaphoreCount::take (
    BUInt v = 1,
    BTimeout timeoutUs = BTimeoutForever )
```

Take for the semaphore.

### 7.82.2.5 value()

```
BUInt BSemaphoreCount::value ( )
```

### 7.82.2.6 wait()

```
Bool BSemaphoreCount::wait (
    BUInt v = 1,
    BTimeout timeoutUs = BTimeoutForever )
```

Wait for the semaphore.

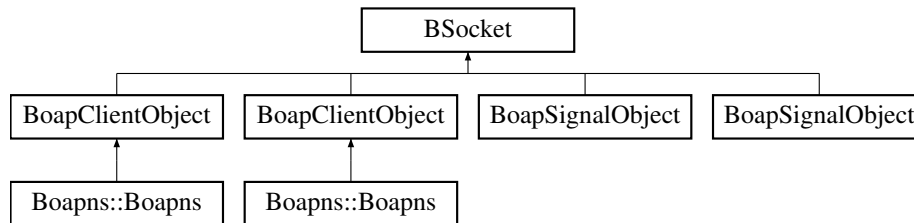
The documentation for this class was generated from the following files:

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

## 7.83 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)

## 7.83.1 Member Enumeration Documentation

### 7.83.1.1 NType

enum `BSocket::NType`

#### Enumerator

STREAM	
DGRAM	

### 7.83.1.2 Priority

enum `BSocket::Priority`

#### Enumerator

PriorityLow	
PriorityNormal	
PriorityHigh	

## 7.83.2 Constructor & Destructor Documentation

### 7.83.2.1 BSocket() [1/4]

`BSocket::BSocket ( )`

### 7.83.2.2 BSocket() [2/4]

`BSocket::BSocket (`  
`int fd )`

### 7.83.2.3 BSocket() [3/4]

```
BSocket::BSocket (
    NType type )
```

### 7.83.2.4 BSocket() [4/4]

```
BSocket::BSocket (
    int domain,
    int type,
    int protocol )
```

### 7.83.2.5 ~BSocket()

```
BSocket::~~BSocket ( )
```

## 7.83.3 Member Function Documentation

### 7.83.3.1 accept() [1/2]

```
BError BSocket::accept (
    int & fd )
```

### 7.83.3.2 accept() [2/2]

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

### 7.83.3.3 bind()

```
BError BSocket::bind (
    const BSocketAddress & add )
```

#### 7.83.3.4 close()

```
BError BSocket::close ( )
```

#### 7.83.3.5 connect()

```
BError BSocket::connect (
    const BSocketAddress & add )
```

#### 7.83.3.6 getAddress()

```
BError BSocket::getAddress (
    BSocketAddress & address )
```

#### 7.83.3.7 getFd()

```
int BSocket::getFd ( )
```

#### 7.83.3.8 getMTU()

```
BError BSocket::getMTU (
    uint32_t & mtu )
```

#### 7.83.3.9 getSockOpt()

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

#### 7.83.3.10 init() [1/2]

```
BError BSocket::init (
    int domain,
    int type,
    int protocol )
```

**7.83.3.11** `init()` [2/2]

```
BError BSocket::init (
    NType type )
```

**7.83.3.12** `listen()`

```
BError BSocket::listen (
    int backlog = 5 )
```

**7.83.3.13** `recv()`

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

**7.83.3.14** `recvAvailable()`

```
BUInt BSocket::recvAvailable ( )
```

**7.83.3.15** `recvFrom()`

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

**7.83.3.16** `recvFromWithTimeout()`

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

#### 7.83.3.17 recvWithTimeout()

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

#### 7.83.3.18 send()

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

#### 7.83.3.19 sendChunks()

```
BError BSocket::sendChunks (
    const BDataChunk * chunks,
    BSize nChunks,
    BSize & nbytesSent,
    int flags = 0 )
```

#### 7.83.3.20 sendTo()

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

#### 7.83.3.21 setBroadCast()

```
BError BSocket::setBroadCast (
    int on )
```

#### 7.83.3.22 setFd()

```
void BSocket::setFd (
    int fd )
```

#### 7.83.3.23 setPriority()

```
BError BSocket::setPriority (
    Priority priority )
```

#### 7.83.3.24 setReuseAddress()

```
BError BSocket::setReuseAddress (
    int on )
```

#### 7.83.3.25 setSockOpt()

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

#### 7.83.3.26 shutdown()

```
BError BSocket::shutdown (
    int how )
```

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

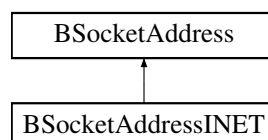
- [BSocket.h](#)
- [BSocket.cpp](#)

## 7.84 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

### 7.84.1 Detailed Description

Socket Address.

### 7.84.2 Member Typedef Documentation

#### 7.84.2.1 SockAddr

```
typedef struct sockaddr BSocketAddress::SockAddr
```

### 7.84.3 Constructor & Destructor Documentation

#### 7.84.3.1 BSocketAddress() [1/3]

```
BSocketAddress::BSocketAddress ( )
```

#### 7.84.3.2 BSocketAddress() [2/3]

```
BSocketAddress::BSocketAddress (
    const BSocketAddress & add )
```

#### 7.84.3.3 BSocketAddress() [3/3]

```
BSocketAddress::BSocketAddress (
    SockAddr * address,
    int len )
```

#### 7.84.3.4 ~BSocketAddress()

```
BSocketAddress::~~BSocketAddress ( )
```

### 7.84.4 Member Function Documentation

#### 7.84.4.1 getString()

```
BString BSocketAddress::getString ( ) const
```

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

#### 7.84.4.2 len()

```
int BSocketAddress::len ( ) const
```

#### 7.84.4.3 operator const SockAddr \*()

```
BSocketAddress::operator const SockAddr * ( ) const [inline]
```

## 7.84.4.4 operator!=(())

```
int BSocketAddress::operator!= (
    const BSocketAddress & add ) const
```

## 7.84.4.5 operator=()

```
BSocketAddress & BSocketAddress::operator= (
    const BSocketAddress & add )
```

## 7.84.4.6 operator==(())

```
int BSocketAddress::operator== (
    const BSocketAddress & add ) const
```

## 7.84.4.7 raw()

```
const BSocketAddress::SockAddr * BSocketAddress::raw ( ) const
```

## 7.84.4.8 set()

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

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

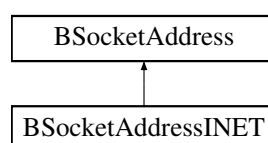
- [BSocket.h](#)
- [BSocket.cpp](#)

## 7.85 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.*

### 7.85.1 Detailed Description

IP aware socket address.

### 7.85.2 Member Typedef Documentation

#### 7.85.2.1 SockAddrIP

```
typedef struct sockaddr_in BSocketAddressINET::SockAddrIP
```

### 7.85.3 Member Function Documentation

### 7.85.3.1 address()

```
uint32_t BSocketAddressINET::address ( )
```

Returns socket ip address.

### 7.85.3.2 getHostName()

```
BString BSocketAddressINET::getHostName ( ) [static]
```

Get this hosts network name.

### 7.85.3.3 getIpAddresses()

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

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

### 7.85.3.4 getIpAddressList()

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

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

### 7.85.3.5 getIpAddressListAll()

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

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

### 7.85.3.6 getString()

```
BString BSocketAddressINET::getString ( )
```

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

#### 7.85.3.7 port()

```
uint32_t BSocketAddressINET::port ( )
```

Returns socket port.

#### 7.85.3.8 set() [1/3]

```
BError BSocketAddressINET::set (
    BString hostName,
    uint32_t port )
```

#### 7.85.3.9 set() [2/3]

```
BError BSocketAddressINET::set (
    uint32_t address,
    uint32_t port )
```

#### 7.85.3.10 set() [3/3]

```
BError BSocketAddressINET::set (
    BString hostName,
    BString service,
    BString type )
```

#### 7.85.3.11 setPort()

```
void BSocketAddressINET::setPort (
    uint32_t port )
```

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

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

## 7.86 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)

### 7.86.1 Detailed Description

[BSpI](#) class.

### 7.86.2 Member Enumeration Documentation

#### 7.86.2.1 Mode

enum [BSpI::Mode](#)

##### Enumerator

<a href="#">Mode0</a>	
<a href="#">Mode1</a>	
<a href="#">Mode2</a>	
<a href="#">Mode3</a>	

### 7.86.3 Constructor & Destructor Documentation

#### 7.86.3.1 BSpI()

[BSpI::BSpI](#) ( )

### 7.86.4 Member Function Documentation

## 7.86.4.1 init()

```
BError BSpi::init (
    BString devName,
    BUInt speed = 1000000,
    Mode mode = Model,
    Bool csActive = 0 )
```

## 7.86.4.2 transact()

```
BError BSpi::transact (
    BUInt8 dev,
    void * txBuf,
    int txLen,
    int pad,
    void * rxBuf,
    int rxLen )
```

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

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

## 7.87 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.*
- const char \* [str](#) () 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](#)

## 7.87.1 Constructor & Destructor Documentation

### 7.87.1.1 BString() [1/9]

```
BString::BString ( )
```

### 7.87.1.2 BString() [2/9]

```
BString::BString (
    const BString & string )
```

**7.87.1.3 BString()** [3/9]

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

**7.87.1.4 BString()** [4/9]

```
BString::BString (
    const char * str,
    unsigned int len )
```

**7.87.1.5 BString()** [5/9]

```
BString::BString (
    char ch )
```

**7.87.1.6 BString()** [6/9]

```
BString::BString (
    BInt v )
```

**7.87.1.7 BString()** [7/9]

```
BString::BString (
    BUInt v )
```

**7.87.1.8 BString()** [8/9]

```
BString::BString (
    BUInt64 v )
```

**7.87.1.9 BString()** [9/9]

```
BString::BString (
    double v )
```

#### 7.87.1.10 ~BString()

```
BString::~~BString ( )
```

### 7.87.2 Member Function Documentation

#### 7.87.2.1 add()

```
BString BString::add (
    const BString & str ) const
```

Add strings returning result.

#### 7.87.2.2 append()

```
int BString::append (
    const BString & str )
```

Append a string.

#### 7.87.2.3 base64Decode()

```
BError BString::base64Decode (
    BString & str ) const
```

Decode a string from base64.

#### 7.87.2.4 base64Encode()

```
BString BString::base64Encode ( ) const
```

Encode a string to base64.

#### 7.87.2.5 basename()

```
BString BString::basename ( )
```

#### 7.87.2.6 clear()

```
void BString::clear ( )
```

Clear the string.

#### 7.87.2.7 compare()

```
int BString::compare (
    const BString & string ) const
```

Compare strings.

#### 7.87.2.8 compareRegex()

```
int BString::compareRegex (
    const BString & pattern,
    int ignoreCase = 0 ) const
```

Compare strings.

#### 7.87.2.9 compareWild()

```
int BString::compareWild (
    const BString & string ) const
```

Compare string to string with wildcards.

#### 7.87.2.10 compareWildExpression()

```
int BString::compareWildExpression (
    const BString & string ) const
```

Compare string to space delimited patterns.

#### 7.87.2.11 convert() [1/5]

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

Converts char to string.

**7.87.2.12 convert()** [2/5]

```
BString BString::convert (
    BInt value ) [static]
```

Converts int to string.

**7.87.2.13 convert()** [3/5]

```
BString BString::convert (
    BUInt value ) [static]
```

Converts uint to string.

**7.87.2.14 convert()** [4/5]

```
BString BString::convert (
    double value,
    int eFormat = 0 ) [static]
```

Converts double to string.

**7.87.2.15 convert()** [5/5]

```
BString BString::convert (
    BUInt64 value ) [static]
```

Converts long long to string.

**7.87.2.16 convertHex()** [1/2]

```
BString BString::convertHex (
    BInt value ) [static]
```

Converts int to string as hex value.

#### 7.87.2.17 `convertHex()` [2/2]

```
BString BString::convertHex (
    BUInt value ) [static]
```

Converts uint to string as hex value.

#### 7.87.2.18 `copy()`

```
BString BString::copy ( ) const
```

Return an independant copy.

#### 7.87.2.19 `csvDecode()`

```
BString & BString::csvDecode (
    const BString str )
```

Decode a string from CSV.

#### 7.87.2.20 `csvEncode()`

```
BString BString::csvEncode ( ) const
```

Encode a string for CSV.

#### 7.87.2.21 `del()`

```
int BString::del (
    int start,
    int len )
```

Delete substring.

#### 7.87.2.22 `dirname()`

```
BString BString::dirname ( )
```



#### 7.87.2.23 extension()

```
BString BString::extension ( )
```

#### 7.87.2.24 field()

```
BString BString::field (
    int field ) const
```

#### 7.87.2.25 fields()

```
char ** BString::fields ( )
```

#### 7.87.2.26 find() [1/2]

```
int BString::find (
    char ch ) const
```

Find *ch* in string searching forwards.

#### 7.87.2.27 find() [2/2]

```
int BString::find (
    BString str ) const
```

Find *string* in string searching forwards.

#### 7.87.2.28 findReverse()

```
int BString::findReverse (
    char ch ) const
```

Find *ch* in string searching backwards.

#### 7.87.2.29 firstLine()

```
BString BString::firstLine ( )
```

Return first line.

#### 7.87.2.30 fixedLen()

```
BString BString::fixedLen (
    int length,
    int rightJustify = 0 )
```

return string formatted to fixed length

#### 7.87.2.31 get() [1/2]

```
char & BString::get (
    int pos )
```

#### 7.87.2.32 get() [2/2]

```
const char & BString::get (
    int pos ) const
```

#### 7.87.2.33 getTokenList() [1/2]

```
BList< BString > BString::getTokenList (
    BString separators )
```

Break string into tokens.

#### 7.87.2.34 getTokenList() [2/2]

```
BList< BString > BString::getTokenList (
    char separator )
```

Break string into tokens.

#### 7.87.2.35 hash()

```
BUInt32 BString::hash ( ) const
```

#### 7.87.2.36 insert()

```
int BString::insert (
    int start,
    BString str )
```

Insert substring.

#### 7.87.2.37 justify()

```
BString BString::justify (
    int leftMargin,
    int width )
```

Justify the string to the given width.

#### 7.87.2.38 len()

```
int BString::len ( ) const
```

Length of string.

#### 7.87.2.39 lowerFirst()

```
BString BString::lowerFirst ( )
```

Return string with lowercase first character.

#### 7.87.2.40 operator const char \*()

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

**7.87.2.41 operator!=( )** [1/2]

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

**7.87.2.42 operator!=( )** [2/2]

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

**7.87.2.43 operator+( )** [1/6]

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

**7.87.2.44 operator+( )** [2/6]

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

**7.87.2.45 operator+( )** [3/6]

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

**7.87.2.46 operator+( )** [4/6]

```
BString BString::operator+ (
    BInt i ) const [inline]
```

**7.87.2.47 operator+( )** [5/6]

```
BString BString::operator+ (
    BUInt i ) const [inline]
```

**7.87.2.48 operator+()** [6/6]

```
BString BString::operator+ (
    BUInt64 i ) const [inline]
```

**7.87.2.49 operator+=()** [1/2]

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

**7.87.2.50 operator+=()** [2/2]

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

**7.87.2.51 operator<()** [1/2]

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

**7.87.2.52 operator<()** [2/2]

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

**7.87.2.53 operator<=()**

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

**7.87.2.54 operator=()**

```
BString & BString::operator= (
    const BString & string )
```

**7.87.2.55 operator==(** [1/2]

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

**7.87.2.56 operator==(** [2/2]

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

**7.87.2.57 operator>(** [1/2]

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

**7.87.2.58 operator>(** [2/2]

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

**7.87.2.59 operator>=(**

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

**7.87.2.60 operator[](**

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

**7.87.2.61 pad()**

```
BString & BString::pad (
    int len )
```

Pad to length len.

#### 7.87.2.62 printf()

```
BString & BString::printf (
    const char * fmt,
    ... )
```

Formatted print into the string.

#### 7.87.2.63 pullLine()

```
BString BString::pullLine ( )
```

Pull a line out of the head of the string.

#### 7.87.2.64 pullSeparators()

```
BString BString::pullSeparators (
    BString separators )
```

Pull separators from start of string.

#### 7.87.2.65 pullToken()

```
BString BString::pullToken (
    BString terminators )
```

Pull token from start of string.

#### 7.87.2.66 pullWord()

```
BString BString::pullWord ( )
```

Pull a word out of the head of the string.

#### 7.87.2.67 removeNL()

```
void BString::removeNL ( )
```

Remove if present NL from last char.

#### 7.87.2.68 removeSeparators()

```
BString BString::removeSeparators (
    BString separators )
```

Remove any char from sepatators from string.

#### 7.87.2.69 retDouble()

```
double BString::retDouble ( ) const
```

Return string as a double.

#### 7.87.2.70 retFloat64()

```
BFloat64 BString::retFloat64 ( ) const
```

Return string as a BFloat64.

#### 7.87.2.71 retInt()

```
int BString::retInt ( ) const
```

Return string as a int.

#### 7.87.2.72 retStr()

```
const char * BString::retStr ( ) const
```

Ptr to char\* representation.

#### 7.87.2.73 retStrDup()

```
char * BString::retStrDup ( ) const
```

Ptr to newly malloc'd char\*.



#### 7.87.2.74 retUInt()

```
unsigned int BString::retUInt ( ) const
```

Return string as a int.

#### 7.87.2.75 reverse()

```
BString BString::reverse ( ) const
```

Reverse character order.

#### 7.87.2.76 split()

```
BList< BString > BString::split (
    char splitChar )
```

Split string into an array based on the character separator.

#### 7.87.2.77 str()

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

Ptr to char\* representation.

#### 7.87.2.78 subString()

```
BString BString::subString (
    int start,
    int len ) const
```

Returns substring.

#### 7.87.2.79 toLower()

```
BString & BString::toLower ( )
```

Convert to lowercase.

#### 7.87.2.80 toUpper()

```
BString & BString::toUpper ( )
```

Convert to uppercase.

#### 7.87.2.81 translateChar()

```
BString BString::translateChar (
    char ch,
    BString replace = " " )
```

Translate character converting them to the given string.

#### 7.87.2.82 truncate()

```
BString & BString::truncate (
    int len )
```

Truncate to length len.

### 7.87.3 Member Data Documentation

#### 7.87.3.1 ostr

```
BRefData* BString::ostr [protected]
```

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

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

## 7.88 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)

## 7.88.1 Constructor & Destructor Documentation

### 7.88.1.1 BStringLocked() [1/3]

```
BStringLocked::BStringLocked ( ) [inline]
```

### 7.88.1.2 BStringLocked() [2/3]

```
BStringLocked::BStringLocked (
    const BStringLocked & s ) [inline]
```

### 7.88.1.3 BStringLocked() [3/3]

```
BStringLocked::BStringLocked (
    const BString & s ) [inline]
```

## 7.88.2 Member Function Documentation

### 7.88.2.1 len()

```
int BStringLocked::len ( ) const [inline]
```

Length of string.

### 7.88.2.2 operator BString()

```
BStringLocked::operator BString ( ) const [inline]
```

### 7.88.2.3 operator+()

```
BStringLocked BStringLocked::operator+ (
    const BStringLocked & s ) const [inline]
```

### 7.88.2.4 operator=()

```
BStringLocked& BStringLocked::operator= (
    const BStringLocked & s ) [inline]
```

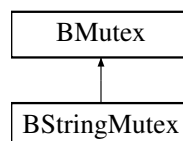
The documentation for this class was generated from the following file:

- [BStringLocked.h](#)

## 7.89 BStringMutex Class Reference

```
#include <BStringLocked.h>
```

Inheritance diagram for BStringMutex:



### Public Member Functions

- [BStringMutex \(\)](#)

### Additional Inherited Members

### 7.89.1 Constructor & Destructor Documentation

### 7.89.1.1 BStringMutex()

```
BStringMutex::BStringMutex ( ) [inline]
```

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

- [BStringLocked.h](#)

## 7.90 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](#) ()

### 7.90.1 Constructor & Destructor Documentation

#### 7.90.1.1 BTable()

```
BTable::BTable ( )
```

#### 7.90.1.2 ~BTable()

```
BTable::~~BTable ( )
```

### 7.90.2 Member Function Documentation

#### 7.90.2.1 addRow()

```
void BTable::addRow (
    BArray< BString > data )
```

#### 7.90.2.2 clear()

```
void BTable::clear ( )
```

#### 7.90.2.3 print()

```
void BTable::print ( )
```

#### 7.90.2.4 setTitle()

```
void BTable::setTitle (
    BArray< BString > title )
```

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

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

## 7.91 BTask Class Reference

```
#include <BTask.h>
```

### Public Member Functions

- [BTask](#) (const char \*name="", [BUInt](#) stackSize=0, [BUInt](#) priority=1)
- virtual [~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`

## 7.91.1 Constructor & Destructor Documentation

### 7.91.1.1 BTask()

```
BTask::BTask (
    const char * name = "",
    BUInt stackSize = 0,
    BUInt priority = 1 )
```

### 7.91.1.2 ~BTask()

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

## 7.91.2 Member Function Documentation

### 7.91.2.1 init()

```
void BTask::init (
    const char * name,
    BUInt stackSize = 0,
    BUInt priority = 1 )
```

### 7.91.2.2 run()

```
void BTask::run ( ) [virtual]
```

#### 7.91.2.3 setPriority()

```
int BTask::setPriority (
    BUInt priority )
```

Set the priority of the task: 0 upwards.

#### 7.91.2.4 start()

```
BError BTask::start ( )
```

Starts the task running.

#### 7.91.2.5 stop()

```
void BTask::stop ( )
```

#### 7.91.2.6 taskFunc()

```
void * BTask::taskFunc (
    void * arg ) [static], [protected]
```

#### 7.91.2.7 waitForCompletion()

```
void BTask::waitForCompletion ( )
```

### 7.91.3 Member Data Documentation

#### 7.91.3.1 oname

```
const char* BTask::oname [protected]
```



### 7.91.3.2 opolicy

```
BUInt BTask::opolicy [protected]
```

### 7.91.3.3 opriority

```
BUInt BTask::opriority [protected]
```

### 7.91.3.4 orunning

```
Bool BTask::orunning [protected]
```

### 7.91.3.5 ostackSize

```
BUInt BTask::ostackSize [protected]
```

### 7.91.3.6 othread

```
pthread_t BTask::othread [protected]
```

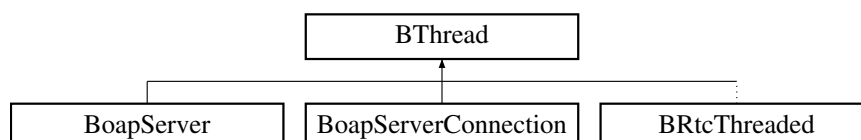
The documentation for this class was generated from the following files:

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

## 7.92 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](#) ()

## 7.92.1 Constructor & Destructor Documentation

### 7.92.1.1 BThread()

```
BThread::BThread ( )
```

### 7.92.1.2 ~BThread()

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

## 7.92.2 Member Function Documentation

### 7.92.2.1 cancel()

```
int BThread::cancel ( )
```

### 7.92.2.2 function()

```
void * BThread::function ( ) [virtual]
```

### 7.92.2.3 getThread()

```
pthread_t BThread::getThread ( )
```

### 7.92.2.4 result()

```
void * BThread::result ( )
```

### 7.92.2.5 running()

```
int BThread::running ( )
```

### 7.92.2.6 setInitPriority()

```
int BThread::setInitPriority (
    int policy,
    int priority )
```

### 7.92.2.7 setInitStackSize()

```
int BThread::setInitStackSize (
    size_t stackSize )
```

### 7.92.2.8 setPriority()

```
int BThread::setPriority (
    int policy,
    int priority )
```

### 7.92.2.9 start()

```
int BThread::start ( )
```

### 7.92.2.10 waitForCompletion()

```
void * BThread::waitForCompletion ( )
```

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

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

## 7.93 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)

## 7.93.1 Constructor & Destructor Documentation

### 7.93.1.1 BTime()

```
BTime::BTime (
    BUInt32 t = 0 )
```

## 7.93.2 Member Function Documentation

### 7.93.2.1 addSeconds()

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

Add the given number of seconds.

### 7.93.2.2 getDate()

```
void BTime::getDate (
    BUInt & year,
    BUInt & month,
    BUInt & day ) const
```

Return the date information.

### 7.93.2.3 getSeconds()

```
BUInt32 BTime::getSeconds ( ) const
```

Return the number of seconds.

### 7.93.2.4 getString()

```
BString BTime::getString (
    BString format = "isoT" ) const
```

Gets the date/time in string format.

#### 7.93.2.5 getTime()

```
void BTime::getTime (
    BUInt & hour,
    BUInt & minute,
    BUInt & second ) const
```

Return the time information.

#### 7.93.2.6 isLeapYear()

```
int BTime::isLeapYear ( )
```

Returns if a leap year.

#### 7.93.2.7 isSet()

```
int BTime::isSet ( ) const [inline]
```

Check if set.

#### 7.93.2.8 operator!=(())

```
int BTime::operator!= (
    const BTime & time ) const [inline]
```

#### 7.93.2.9 operator+()

```
BTime BTime::operator+ (
    int seconds ) const [inline]
```

#### 7.93.2.10 operator+=(())

```
BTime& BTime::operator+= (
    int seconds ) [inline]
```

#### 7.93.2.11 operator<()

```
int BTime::operator< (
    const BTime & time ) const [inline]
```

#### 7.93.2.12 operator<=()

```
int BTime::operator<= (
    const BTime & time ) const [inline]
```

#### 7.93.2.13 operator==( )

```
int BTime::operator==(
    const BTime & time ) const [inline]
```

#### 7.93.2.14 operator>()

```
int BTime::operator> (
    const BTime & time ) const [inline]
```

#### 7.93.2.15 operator>=()

```
int BTime::operator>= (
    const BTime & time ) const [inline]
```

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

```
void BTime::set (
    BUInt32 seconds )
```

Set the date and time.

#### 7.93.2.17 `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.

#### 7.93.2.18 `setString()`

```
BError BTime::setString (
    const BString dateTime )
```

Sets the date/time from string format.

#### 7.93.2.19 `setYearDay()`

```
void BTime::setYearDay (
    BUInt year,
    BUInt yearDay,
    BUInt hour = 0,
    BUInt minute = 0,
    BUInt second = 0 )
```

Set the date and time.

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

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

## 7.94 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.*

### 7.94.1 Detailed Description

Stopwatch style timer.

### 7.94.2 Constructor & Destructor Documentation

#### 7.94.2.1 BTimer()

```
BTimer::BTimer ( )
```

#### 7.94.2.2 ~BTimer()

```
BTimer::~~BTimer ( )
```

### 7.94.3 Member Function Documentation

#### 7.94.3.1 add()

```
void BTimer::add (  
    BTimer & timer )
```

Add two timers.

#### 7.94.3.2 `average()`

```
double BTimer::average ( )
```

Average time is duration between [start\(\)](#) and [stop\(\)](#) / number of stops.

#### 7.94.3.3 `clear()`

```
void BTimer::clear ( )
```

Clear timer.

#### 7.94.3.4 `getElapsedTime()`

```
double BTimer::getElapsedTime ( )
```

Returns the elapsed time from the last start.

#### 7.94.3.5 `peak()`

```
double BTimer::peak ( )
```

Peak time.

#### 7.94.3.6 `start()`

```
void BTimer::start ( )
```

Start timer.

#### 7.94.3.7 `stop()`

```
void BTimer::stop ( )
```

Stop timer.

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

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

## 7.95 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 than a year.*
- void [addMicroSeconds](#) (int64\_t [microSeconds](#))
  - Add the given number of micro seconds. This should be less than a year.*
- void [addSeconds](#) (int [seconds](#))
  - Add the given number of seconds. This should be less than 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 `omminute`  
*Minute (0 .. 59)*
- uint8\_t `osecond`  
*Second (0 .. 59)*
- uint8\_t `ospare`  
*Padding.*
- uint32\_t `omicroSecond`  
*MicroSecond (0 .. 999999)*

## 7.95.1 Constructor & Destructor Documentation

### 7.95.1.1 `BTimeStamp()` [1/3]

```
BTimeStamp::BTimeStamp ( )
```

### 7.95.1.2 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 )
```

### 7.95.1.3 BTimeStamp() [3/3]

```
BTimeStamp::BTimeStamp (
    const BString str )
```

### 7.95.1.4 ~BTimeStamp()

```
BTimeStamp::~BTimeStamp ( )
```

## 7.95.2 Member Function Documentation

### 7.95.2.1 addMicroSeconds()

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

Add the given number of micro seconds. This should be less that a year.

### 7.95.2.2 addMilliSeconds()

```
void BTimeStamp::addMilliSeconds (
    int milliSeconds )
```

Add the given number of milli seconds. This should be less that a year.

### 7.95.2.3 addSeconds()

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

Add the given number of seconds. This should be less than a year.

### 7.95.2.4 clear()

```
void BTimeStamp::clear ( )
```

Clear the date/time.

### 7.95.2.5 compare()

```
int BTimeStamp::compare (
    const BTimeStamp & timeStamp ) const
```

Compare two dates.

### 7.95.2.6 day()

```
int BTimeStamp::day ( ) const
```

### 7.95.2.7 difference()

```
BInt64 BTimeStamp::difference (
    BTimeStamp t2,
    BTimeStamp t1 ) [static]
```

### 7.95.2.8 getDate()

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

#### 7.95.2.9 getString()

```
BString BTimeStamp::getString (
    BString separator = "T" ) const
```

Get the time as an ISO date/time string.

#### 7.95.2.10 getStringFormatted()

```
BString BTimeStamp::getStringFormatted (
    BString format ) const
```

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

#### 7.95.2.11 getStringNoMs()

```
BString BTimeStamp::getStringNoMs (
    BString separator = "T" ) const
```

Get the time as an ISO date/time string without microseconds.

#### 7.95.2.12 getYearMicroSeconds()

```
uint64_t BTimeStamp::getYearMicroSeconds ( ) const
```

Get number of micro seconds within the year.

#### 7.95.2.13 getYearSeconds()

```
uint32_t BTimeStamp::getYearSeconds ( ) const
```

Get number of seconds within the year.

#### 7.95.2.14 hour()

```
int BTimeStamp::hour ( ) const
```

#### 7.95.2.15 isLeap()

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

#### 7.95.2.16 isSet()

```
int BTimeStamp::isSet ( ) const [inline]
```

#### 7.95.2.17 microSecond()

```
int BTimeStamp::microSecond ( ) const
```

#### 7.95.2.18 minute()

```
int BTimeStamp::minute ( ) const
```

#### 7.95.2.19 month()

```
int BTimeStamp::month ( ) const
```

#### 7.95.2.20 operator BString()

```
BTimeStamp::operator BString ( ) const [inline]
```

#### 7.95.2.21 operator!=(=)

```
int BTimeStamp::operator!= (
    const BTimeStamp & timeStamp ) const [inline]
```



#### 7.95.2.22 operator<()

```
int BTimeStamp::operator< (
    const BTimeStamp & timeStamp ) const [inline]
```

#### 7.95.2.23 operator<=()

```
int BTimeStamp::operator<= (
    const BTimeStamp & timeStamp ) const [inline]
```

#### 7.95.2.24 operator=()

```
BTimeStamp& BTimeStamp::operator= (
    const BTimeStampMs & timeStamp ) [inline]
```

#### 7.95.2.25 operator==()

```
int BTimeStamp::operator== (
    const BTimeStamp & timeStamp ) const [inline]
```

#### 7.95.2.26 operator>()

```
int BTimeStamp::operator> (
    const BTimeStamp & timeStamp ) const [inline]
```

#### 7.95.2.27 operator>=()

```
int BTimeStamp::operator>= (
    const BTimeStamp & timeStamp ) const [inline]
```

#### 7.95.2.28 second()

```
int BTimeStamp::second ( ) const
```

**7.95.2.29 set()** [1/3]

```
void BTimeStamp::set (
    time_t time,
    int microseconds )
```

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

**7.95.2.30 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 )
```

**7.95.2.31 set()** [3/3]

```
void BTimeStamp::set (
    const BTimeStampMs & timeStamp )
```

Set the timeStamp to given MS time stamp.

**7.95.2.32 setFirst()**

```
void BTimeStamp::setFirst ( )
```

Set the first date available.

**7.95.2.33 setLast()**

```
void BTimeStamp::setLast ( )
```

Set the last date available.

#### 7.95.2.34 setNow()

```
void BTimeStamp::setNow ( )
```

Set the timeStamp to now.

#### 7.95.2.35 setString()

```
BError BTimeStamp::setString (
    const BString dateTime )
```

Set the time from an ISO date/time.

#### 7.95.2.36 setTime()

```
void BTimeStamp::setTime (
    int hour = 0,
    int minute = 0,
    int second = 0,
    int microsecond = 0 )
```

#### 7.95.2.37 setYDay()

```
void BTimeStamp::setYDay (
    int year = 0,
    int yday = 0,
    int hour = 0,
    int minute = 0,
    int second = 0,
    int microsecond = 0 )
```

#### 7.95.2.38 yday()

```
int BTimeStamp::yday ( ) const
```

#### 7.95.2.39 year()

```
int BTimeStamp::year ( ) const
```

### 7.95.3 Member Data Documentation

#### 7.95.3.1 ohour

`uint8_t BTimeStamp::ohour`

Hour (0 .. 23)

#### 7.95.3.2 omicroSecond

`uint32_t BTimeStamp::omicroSecond`

MicroSecond (0 .. 999999)

#### 7.95.3.3 ominute

`uint8_t BTimeStamp::ominute`

Minute (0 .. 59)

#### 7.95.3.4 osecond

`uint8_t BTimeStamp::osecond`

Second (0 .. 59)

#### 7.95.3.5 ospare

`uint8_t BTimeStamp::ospare`

Padding.

## 7.95.3.6 oyday

```
uint16_t BTimeStamp::oyday
```

Day in year (0 .. 365)

## 7.95.3.7 oyear

```
uint16_t BTimeStamp::oyear
```

Year (0 .. 65535)

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

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

## 7.96 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.*
- void [setFirst](#) ()  
*Set the first date available.*
- void [setLast](#) ()  
*Set the last date available.*
- void [set](#) (time\_t time, int milliseconds=0)  
*Set time using Unix time (seconds from 1970-01-01)*
- void [setYDay](#) (int year=0, int yday=0, int hour=0, int minute=0, int second=0, int milliSecond=0)
- void [setTime](#) (int hour=0, int minute=0, int second=0, int milliSecond=0)
- [BTimeStampMs](#) & [addMilliseconds](#) (int milliseconds)  
*Add the given number of milli seconds. This should be less that a year.*
- [BTimeStampMs](#) & [subMilliseconds](#) (int milliseconds)  
*Add the given number of milli seconds. This should be less that a year.*
- [BTimeStampMs](#) & [addSeconds](#) (int seconds)  
*Add the given number of seconds. This should be less that a year.*
- [BTimeStampMs](#) & [subSeconds](#) (int seconds)  
*Subtract the given number of seconds. This should be less that 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.*

## 7.96.1 Constructor & Destructor Documentation

### 7.96.1.1 BTimeStampMs()

```
BTimeStampMs::BTimeStampMs (
    BString str = "" )
```

### 7.96.1.2 ~BTimeStampMs()

```
BTimeStampMs::~~BTimeStampMs ( )
```

## 7.96.2 Member Function Documentation

### 7.96.2.1 addMilliseconds()

```
BTimeStampMs & BTimeStampMs::addMilliseconds (
    int milliseconds )
```

Add the given number of milli seconds. This should be less that a year.

### 7.96.2.2 addSeconds()

```
BTimeStampMs & BTimeStampMs::addSeconds (
    int seconds )
```

Add the given number of seconds. This should be less that a year.

### 7.96.2.3 clear()

```
void BTimeStampMs::clear ( )
```

Clear the date/time.

### 7.96.2.4 compare()

```
int BTimeStampMs::compare (
    const BTimeStampMs & timeStamp )
```

Compare two dates.

#### 7.96.2.5 difference()

```
BUInt64 BTimeStampMs::difference (
    BTimeStamp t2,
    BTimeStamp t1 ) [static]
```

#### 7.96.2.6 getDate()

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

Get the year, month and day.

#### 7.96.2.7 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.

#### 7.96.2.8 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.

#### 7.96.2.9 getString()

```
BString BTimeStampMs::getString (
    BString separator = "T" )
```

Get the time as an ISO date/time string.



#### 7.96.2.10 getStringNoMs()

```
BString BTimeStampMs::getStringNoMs (
    BString separator = "T" )
```

Get the time as an ISO date/time string with no ms.

#### 7.96.2.11 getStringRaw()

```
BString BTimeStampMs::getStringRaw ( )
```

#### 7.96.2.12 getYearMilliseconds()

```
uint64_t BTimeStampMs::getYearMilliseconds ( )
```

Get number of seconds within the year.

#### 7.96.2.13 getYearSeconds()

```
uint32_t BTimeStampMs::getYearSeconds ( )
```

Get number of seconds within the year.

#### 7.96.2.14 isLeap()

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

#### 7.96.2.15 operator<()

```
int BTimeStampMs::operator< (
    const BTimeStamp & timeStamp ) [inline]
```

#### 7.96.2.16 operator<=()

```
int BTimeStampMs::operator<= (
    const BTimeStampMs & timeStamp ) [inline]
```

#### 7.96.2.17 operator>()

```
int BTimeStampMs::operator> (
    const BTimeStampMs & timeStamp ) [inline]
```

#### 7.96.2.18 operator>=()

```
int BTimeStampMs::operator>= (
    const BTimeStampMs & timeStamp ) [inline]
```

#### 7.96.2.19 set()

```
void BTimeStampMs::set (
    time_t time,
    int milliseconds = 0 )
```

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

#### 7.96.2.20 setDurationString()

```
BError BTimeStampMs::setDurationString (
    BString dateTime )
```

Set the time from an ISO date/time string but with month's and days starting from 0.

#### 7.96.2.21 setFirst()

```
void BTimeStampMs::setFirst ( )
```

Set the first date available.

#### 7.96.2.22 setLast()

```
void BTimeStampMs::setLast ( )
```

Set the last date available.

#### 7.96.2.23 setNow()

```
void BTimeStampMs::setNow ( )
```

Set the timeStamp to now.

#### 7.96.2.24 setString()

```
BError BTimeStampMs::setString (
    BString dateTime )
```

Set the time from an ISO date/time.

#### 7.96.2.25 setTime()

```
void BTimeStampMs::setTime (
    int hour = 0,
    int minute = 0,
    int second = 0,
    int milliSecond = 0 )
```

#### 7.96.2.26 setYDay()

```
void BTimeStampMs::setYDay (
    int year = 0,
    int yday = 0,
    int hour = 0,
    int minute = 0,
    int second = 0,
    int milliSecond = 0 )
```

#### 7.96.2.27 subMilliseconds()

```
BTimeStampMs & BTimeStampMs::subMilliseconds (
    int milliseconds )
```

Add the given number of milli seconds. This should be less that a year.

#### 7.96.2.28 subSeconds()

```
BTimeStampMs & BTimeStampMs::subSeconds (
    int seconds )
```

Subtract the given number of seconds. This should be less that a year.

### 7.96.3 Member Data Documentation

#### 7.96.3.1 hour

```
uint16_t BTimeStampMs::hour
```

Hour (0 .. 23)

#### 7.96.3.2 milliSecond

```
uint16_t BTimeStampMs::milliSecond
```

MilliSecond (0 .. 999)

#### 7.96.3.3 minute

```
uint16_t BTimeStampMs::minute
```

Minute (0 .. 59)

#### 7.96.3.4 sampleNumber

```
int32_t BTimeStampMs::sampleNumber
```

The sample number this time refers to.

#### 7.96.3.5 second

```
uint16_t BTimeStampMs::second
```

Second (0 .. 59)

#### 7.96.3.6 yday

```
uint16_t BTimeStampMs::yday
```

Day in year (0 .. 365)

#### 7.96.3.7 year

```
uint16_t BTimeStampMs::year
```

Year (2000 .. 3000)

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

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

## 7.97 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](#) microSeconds=0)  
*Set the date and time from UTC.*
- void [setYearDay](#) ([BUInt](#) year, [BUInt](#) yearDay, [BUInt](#) hour=0, [BUInt](#) minute=0, [BUInt](#) second=0, [BUInt](#) microSeconds=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)

### 7.97.1 Constructor & Destructor Documentation

### 7.97.1.1 BTimeUs() [1/2]

```
BTimeUs::BTimeUs (
    BUInt64 t = 0 )
```

### 7.97.1.2 BTimeUs() [2/2]

```
BTimeUs::BTimeUs (
    BTime t )
```

## 7.97.2 Member Function Documentation

### 7.97.2.1 addMicroSeconds()

```
void BTimeUs::addMicroSeconds (
    BInt64 microSeconds )
```

Add the given number of seconds.

### 7.97.2.2 addSeconds()

```
void BTimeUs::addSeconds (
    BInt64 seconds )
```

Add the given number of seconds.

### 7.97.2.3 getDate()

```
void BTimeUs::getDate (
    BUInt & year,
    BUInt & month,
    BUInt & day ) const
```

Return the date information UTC.

#### 7.97.2.4 getMicroSeconds()

```
BUInt64 BTimeUs::getMicroSeconds ( ) const
```

Return the number of micro seconds TAI.

#### 7.97.2.5 getSeconds()

```
BUInt64 BTimeUs::getSeconds ( ) const
```

Return the number of seconds TAI.

#### 7.97.2.6 getString()

```
BString BTimeUs::getString (
    BString format = "isoT" ) const
```

Gets the date/time in string format.

#### 7.97.2.7 getStringUs()

```
BString BTimeUs::getStringUs (
    BString format = "isoT" ) const
```

Gets the date/time in string format.

#### 7.97.2.8 getTime()

```
void BTimeUs::getTime (
    BUInt & hour,
    BUInt & minute,
    BUInt & second ) const
```

Return the time information UTC.

#### 7.97.2.9 isLeapYear()

```
int BTimeUs::isLeapYear ( )
```

Returns if a leap year.



#### 7.97.2.10 isSet()

```
int BTimeUs::isSet ( ) const [inline]
```

Check if set.

#### 7.97.2.11 operator BTime()

```
BTimeUs::operator BTime ( ) const [inline]
```

#### 7.97.2.12 operator"!="()

```
int BTimeUs::operator!= (
    const BTimeUs & time ) const [inline]
```

#### 7.97.2.13 operator+()

```
BTimeUs BTimeUs::operator+ (
    BInt64 microseconds ) const [inline]
```

#### 7.97.2.14 operator+=()

```
BTimeUs& BTimeUs::operator+= (
    BInt64 microseconds ) [inline]
```

#### 7.97.2.15 operator<()

```
int BTimeUs::operator< (
    const BTimeUs & time ) const [inline]
```

#### 7.97.2.16 operator<=()

```
int BTimeUs::operator<= (
    const BTimeUs & time ) const [inline]
```

#### 7.97.2.17 operator==( )

```
int BTimeUs::operator==(
    const BTimeUs & time ) const [inline]
```

#### 7.97.2.18 operator>( )

```
int BTimeUs::operator> (
    const BTimeUs & time ) const [inline]
```

#### 7.97.2.19 operator>=( )

```
int BTimeUs::operator>= (
    const BTimeUs & time ) const [inline]
```

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

```
void BTimeUs::set (
    BUInt64 microSeconds )
```

Set the time to TAI us.

#### 7.97.2.21 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.

#### 7.97.2.22 setString( )

```
BError BTimeUs::setString (
    const BString dateTime )
```

Sets the date/time from string format.

### 7.97.2.23 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.

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

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

## 7.98 BUrl Class Reference

Basic access to a Url.

```
#include <BUrl.h>
```

### Public Member Functions

- [BUrl \(\)](#)
- [~BUrl \(\)](#)
- [BError readString \(BString url, BString &str\)](#)  
*Reads URL.*

### 7.98.1 Detailed Description

Basic access to a Url.

### 7.98.2 Constructor & Destructor Documentation

#### 7.98.2.1 BUrl()

```
BUrl::BUrl ( )
```

#### 7.98.2.2 ~BUrl()

```
BUrl::~~BUrl ( )
```

### 7.98.3 Member Function Documentation

#### 7.98.3.1 readString()

```
BError BUrl::readString (
    BString url,
    BString & str )
```

Reads URL.

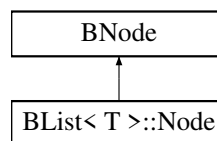
The documentation for this class was generated from the following files:

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

### 7.99 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](#)

### 7.99.1 Constructor & Destructor Documentation

#### 7.99.1.1 Node()

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

## 7.99.2 Member Data Documentation

### 7.99.2.1 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 /src/bdev/beam-lib/doc/overview.dox File Reference

### 8.2 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++)

#### 8.2.1 Macro Definition Documentation

##### 8.2.1.1 BArrayLoop

```
#define BArrayLoop(  
    list,  
    i ) for(BUInt i = 0; i < list.number\(\); i++)
```

### 8.3 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](#)

## 8.3.1 Typedef Documentation

### 8.3.1.1 BAtomicInt32

```
typedef BAtomic<BInt32> BAtomicInt32
```

### 8.3.1.2 BAtomicInt64

```
typedef BAtomic<BInt64> BAtomicInt64
```

### 8.3.1.3 BAtomicUInt32

```
typedef BAtomic<BUInt32> BAtomicUInt32
```

### 8.3.1.4 BAtomicUInt64

```
typedef BAtomic<BUInt64> BAtomicUInt64
```

## 8.4 BAtomicCount.h File Reference

```
#include <bits/atomicity.h>
```



## Classes

- class [BAtomicCount](#)  
*BAtomicCount* class.

## 8.5 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

### 8.5.1 Variable Documentation

#### 8.5.1.1 roundSize

```
const int roundSize = 256
```

## 8.6 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

## 8.6.1 Macro Definition Documentation

### 8.6.1.1 BBigEndian

```
#define BBigEndian 0
```

## 8.7 BComms.cpp File Reference

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

## 8.8 BComms.h File Reference

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

### Classes

- class [BComms](#)

## 8.9 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](#)

### 8.9.1 Typedef Documentation

### 8.9.1.1 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.

### 8.9.1.2 BComplex32

```
typedef std::complex<float> BComplex32
```

### 8.9.1.3 BComplex64

```
typedef std::complex<double> BComplex64
```

## 8.10 BCond.cpp File Reference

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

## 8.11 BCond.h File Reference

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

### Classes

- class [BCond](#)

## 8.12 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)

## 8.12.1 Function Documentation

### 8.12.1.1 getTimeout()

```
static struct timespec getTimeout (
    uint32_t timeOutUs ) [static]
```

## 8.13 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.*

## 8.14 BConfig.cpp File Reference

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

## 8.15 BConfig.h File Reference

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

### Classes

- class [BConfig](#)  
*This class implements the configuration file access.*



### 8.16.2.2 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
}
```

## 8.17 BCrc16.h File Reference

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

### Functions

- [BUInt16 bcrc16](#) (void \*buf, [BUInt16](#) len)

### 8.17.1 Function Documentation

#### 8.17.1.1 bcrc16()

```
BUInt16 bcrc16 (
    void * buf,
    BUInt16 len )
```

## 8.18 BCrc32.cpp File Reference

```
#include <BCrc32.h>
```

## Functions

- [BUInt32 bcrc32](#) ([BUInt32](#) crc, const void \*buf, [BUInt32](#) len)

## Variables

- static [BUInt32 crc32\\_tab](#) []

### 8.18.1 Function Documentation

#### 8.18.1.1 bcrc32()

```
BUInt32 bcrc32 (  
    BUInt32 crc,  
    const void * buf,  
    BUInt32 len )
```

### 8.18.2 Variable Documentation

#### 8.18.2.1 crc32\_tab

```
BUInt32 crc32_tab[]  [static]
```

## 8.19 BCrc32.h File Reference

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

## Functions

- [BUInt32 bcrc32](#) ([BUInt32](#) crc, const void \*buf, [BUInt32](#) len)

### 8.19.1 Function Documentation

#### 8.19.1.1 bcrc32()

```
BUInt32 bcrc32 (
    BUInt32 crc,
    const void * buf,
    BUInt32 len )
```

## 8.20 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]

### 8.20.1 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.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]

## 8.21.1 Function Documentation

### 8.21.1.1 fromBString()

```
void fromBString (  
    BString & s,  
    BDate & v )
```

### 8.21.1.2 toBString()

```
void toBString (  
    BDate & v,  
    BString & s )
```

## 8.21.2 Variable Documentation

### 8.21.2.1 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 }  
}
```

## 8.22 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)

### 8.22.1 Function Documentation

#### 8.22.1.1 fromBString()

```
void fromBString (  
    BString & s,  
    BDate & v )
```

#### 8.22.1.2 toBString()

```
void toBString (  
    BDate & v,  
    BString & s )
```

## 8.23 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](#)

#### 8.23.1 Function Documentation

##### 8.23.1.1 [bhd32\(\)](#)

```
void bhd32 (
    const void * data,
    unsigned int n )
```

##### 8.23.1.2 [bhd8\(\)](#)

```
void bhd8 (
    const void * data,
    unsigned int n )
```

#### 8.23.1.3 bhd8a()

```
void bhd8a (
    const void * data,
    unsigned int n )
```

#### 8.23.1.4 bhda32()

```
void bhda32 (
    const void * data,
    unsigned int n )
```

#### 8.23.1.5 bhda8()

```
void bhda8 (
    const void * data,
    unsigned int n )
```

#### 8.23.1.6 getTime()

```
double getTime ( )
```

#### 8.23.1.7 setDebug()

```
void setDebug (
    int d )
```

### 8.23.2 Variable Documentation

#### 8.23.2.1 bdebug

```
int bdebug
```

## 8.24 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 [bgettid](#) ()

### Variables

- int [bdebug](#)

#### 8.24.1 Macro Definition Documentation

#### 8.24.1.1 BDebug\_STD

```
#define BDebug_STD 0x000001
```

#### 8.24.1.2 dl1printf

```
#define dl1printf(  
    fmt,  
    a... )
```

#### 8.24.1.3 dl2printf

```
#define dl2printf(  
    fmt,  
    a... )
```

#### 8.24.1.4 dl3printf

```
#define dl3printf(  
    fmt,  
    a... )
```

#### 8.24.1.5 dl4printf

```
#define dl4printf(  
    fmt,  
    a... )
```

#### 8.24.1.6 dprintf

```
#define dprintf(  
    level,  
    fmt,  
    a... )
```

General debug functions.

#### 8.24.1.7 eprintf

```
#define eprintf(  
    fmt,  
    a... ) syslog(LOG_ERR, fmt, ##a)
```

#### 8.24.1.8 nprintf

```
#define nprintf(  
    fmt,  
    a... ) syslog(LOG_NOTICE, fmt, ##a)
```

Warnings and errors logging.

#### 8.24.1.9 wprintf

```
#define wprintf(  
    fmt,  
    a... ) syslog(LOG_WARNING, fmt, ##a)
```

### 8.24.2 Function Documentation

#### 8.24.2.1 bgettid()

```
pid_t bgettid ( )
```

#### 8.24.2.2 bhd32()

```
void bhd32 (  
    const void * data,  
    unsigned int n )
```

#### 8.24.2.3 bhd8()

```
void bhd8 (  
    const void * data,  
    unsigned int n )
```

#### 8.24.2.4 bhd8a()

```
void bhd8a (
    const void * data,
    unsigned int n )
```

#### 8.24.2.5 bhda8()

```
void bhda8 (
    const void * data,
    unsigned int n )
```

#### 8.24.2.6 bhds32()

```
void bhds32 (
    const void * data,
    unsigned int n )
```

#### 8.24.2.7 getTime()

```
double getTime ( )
```

#### 8.24.2.8 setDebug()

```
void setDebug (
    int debug )
```

#### 8.24.2.9 tprintf()

```
void tprintf (
    int log,
    const char * fmt,
    ... )
```

### 8.24.3 Variable Documentation



### 8.24.3.1 bdebug

```
int bdebug
```

## 8.25 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)

### 8.25.1 Function Documentation

#### 8.25.1.1 bdictStringToString()

```
BString bdictStringToString (
    const BDictString & dict )
```

#### 8.25.1.2 fromBString()

```
void fromBString (
    const BString & str,
    BDictString & v )
```

#### 8.25.1.3 toBString()

```
void toBString (
    const BDictString & v,
    BString & s )
```

## 8.26 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)

### 8.26.1 Typedef Documentation

#### 8.26.1.1 BDictString

```
typedef BDict<BString> BDictString
```

### 8.26.2 Function Documentation

#### 8.26.2.1 bdictStringToString()

```
BString bdictStringToString (  
    const BDictString & dict )
```

#### 8.26.2.2 fromBString()

```
void fromBString (  
    const BString & s,  
    BDictString & v )
```

### 8.26.2.3 toBString()

```
void toBString (
    const BDictString & v,
    BString & s )
```

## 8.27 BDictMap.h File Reference

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

### Classes

- class [BDictMap< Value >](#)

### Typedefs

- typedef [BDictMap< BString >](#) [BDictMapString](#)

### 8.27.1 Typedef Documentation

#### 8.27.1.1 BDictMapString

```
typedef BDictMap<BString> BDictMapString
```

## 8.28 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](#)

## 8.28.1 Function Documentation

### 8.28.1.1 wild()

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

## 8.28.2 Variable Documentation

### 8.28.2.1 wildString

```
BString wildString [static]
```

## 8.29 BDir.h File Reference

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

### Classes

- class [BDir](#)  
*File system directory class.*

## 8.30 BDuration.cpp File Reference

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

## 8.31 BDuration.h File Reference

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

## Classes

- class [BDuration](#)

## 8.32 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)

### 8.32.1 Function Documentation

#### 8.32.1.1 bswap\_copy()

```
void bswap_copy (
    int swap,
    const void * src,
    void * dst,
    BUInt32 nBytes,
    const char * swapType )
```

## 8.33 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)

### 8.33.1 Macro Definition Documentation

#### 8.33.1.1 `be16toh`

```
#define be16toh(  
    x ) __bswap_16 (x)
```

#### 8.33.1.2 be32toh

```
#define be32toh(  
    x ) __bswap_32 (x)
```

#### 8.33.1.3 be64toh

```
#define be64toh(  
    x ) __bswap_64 (x)
```

#### 8.33.1.4 htobe16

```
#define htobe16(  
    x ) __bswap_16 (x)
```

#### 8.33.1.5 htobe32

```
#define htobe32(  
    x ) __bswap_32 (x)
```

#### 8.33.1.6 htobe64

```
#define htobe64(  
    x ) __bswap_64 (x)
```

#### 8.33.1.7 htole16

```
#define htole16(  
    x ) (x)
```

#### 8.33.1.8 htole32

```
#define htole32(  
    x ) (x)
```

#### 8.33.1.9 htole64

```
#define htole64(  
    x ) (x)
```

#### 8.33.1.10 le16toh

```
#define le16toh(  
    x ) (x)
```

#### 8.33.1.11 le32toh

```
#define le32toh(  
    x ) (x)
```

#### 8.33.1.12 le64toh

```
#define le64toh(  
    x ) (x)
```

### 8.33.2 Function Documentation

#### 8.33.2.1 betoh() [1/8]

```
uint16_t betoh (  
    uint16_t v ) [inline]
```

#### 8.33.2.2 betoh() [2/8]

```
int16_t betoh (  
    int16_t v ) [inline]
```



**8.33.2.3 betoh()** [3/8]

```
uint32_t betoh (  
    uint32_t v ) [inline]
```

**8.33.2.4 betoh()** [4/8]

```
int32_t betoh (  
    int32_t v ) [inline]
```

**8.33.2.5 betoh()** [5/8]

```
uint64_t betoh (  
    uint64_t v ) [inline]
```

**8.33.2.6 betoh()** [6/8]

```
int64_t betoh (  
    int64_t v ) [inline]
```

**8.33.2.7 betoh()** [7/8]

```
double betoh (  
    double v ) [inline]
```

**8.33.2.8 betoh()** [8/8]

```
float betoh (  
    float v ) [inline]
```

**8.33.2.9 bswap\_copy()**

```
void bswap_copy (  
    int swap,  
    const void * src,  
    void * dst,  
    BUInt32 nBytes,  
    const char * swapType )
```

#### 8.33.2.10 bswap\_p16()

```
void bswap_p16 (
    const void * s,
    void * d ) [inline]
```

#### 8.33.2.11 bswap\_p32()

```
void bswap_p32 (
    const void * s,
    void * d ) [inline]
```

#### 8.33.2.12 bswap\_p64()

```
void bswap_p64 (
    const void * s,
    void * d ) [inline]
```

#### 8.33.2.13 bswap\_p8()

```
void bswap_p8 (
    const void * s,
    void * d ) [inline]
```

#### 8.33.2.14 htobe() [1/8]

```
uint16_t htobe (
    uint16_t v ) [inline]
```

#### 8.33.2.15 htobe() [2/8]

```
int16_t htobe (
    int16_t v ) [inline]
```

**8.33.2.16 htobe()** [3/8]

```
uint32_t htobe (  
    uint32_t v ) [inline]
```

**8.33.2.17 htobe()** [4/8]

```
int32_t htobe (  
    int32_t v ) [inline]
```

**8.33.2.18 htobe()** [5/8]

```
uint64_t htobe (  
    uint64_t v ) [inline]
```

**8.33.2.19 htobe()** [6/8]

```
int64_t htobe (  
    int64_t v ) [inline]
```

**8.33.2.20 htobe()** [7/8]

```
double htobe (  
    double v ) [inline]
```

**8.33.2.21 htobe()** [8/8]

```
float htobe (  
    float v ) [inline]
```

**8.33.2.22 htobe()** [1/8]

```
uint16_t htobe (  
    uint16_t v ) [inline]
```

**8.33.2.23 htole()** [2/8]

```
int16_t htole (
    int16_t v ) [inline]
```

**8.33.2.24 htole()** [3/8]

```
uint32_t htole (
    uint32_t v ) [inline]
```

**8.33.2.25 htole()** [4/8]

```
int32_t htole (
    int32_t v ) [inline]
```

**8.33.2.26 htole()** [5/8]

```
uint64_t htole (
    uint64_t v ) [inline]
```

**8.33.2.27 htole()** [6/8]

```
int64_t htole (
    int64_t v ) [inline]
```

**8.33.2.28 htole()** [7/8]

```
double htole (
    double v ) [inline]
```

**8.33.2.29 htole()** [8/8]

```
float htole (
    float v ) [inline]
```

**8.33.2.30** `letoh()` [1/8]

```
uint16_t letoh (  
    uint16_t v )    [inline]
```

**8.33.2.31** `letoh()` [2/8]

```
int16_t letoh (  
    int16_t v )    [inline]
```

**8.33.2.32** `letoh()` [3/8]

```
uint32_t letoh (  
    uint32_t v )    [inline]
```

**8.33.2.33** `letoh()` [4/8]

```
int32_t letoh (  
    int32_t v )    [inline]
```

**8.33.2.34** `letoh()` [5/8]

```
uint64_t letoh (  
    uint64_t v )    [inline]
```

**8.33.2.35** `letoh()` [6/8]

```
int64_t letoh (  
    int64_t v )    [inline]
```

**8.33.2.36** `letoh()` [7/8]

```
double letoh (  
    double v )    [inline]
```

### 8.33.2.37 letoh() [8/8]

```
float letoh (
    float v ) [inline]
```

## 8.34 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>
```

## 8.35 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.*

## 8.36 BError.cpp File Reference

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

## 8.37 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, `ErrorDataTruncated` = 22, `ErrorApiVersion` = 23,  
`ErrorAppBase` = 64, `ErrorUserBase` = 96 }

### 8.37.1 Enumeration Type Documentation

#### 8.37.1.1 BErrorNum

enum `BErrorNum`

##### Enumerator

<code>ErrorOk</code>	
<code>ErrorMisc</code>	
<code>ErrorWarning</code>	
<code>ErrorParam</code>	
<code>ErrorTimeout</code>	
<code>ErrorNotAvailable</code>	
<code>ErrorData</code>	
<code>ErrorChecksum</code>	
<code>ErrorOverrun</code>	
<code>ErrorUnderrun</code>	
<code>ErrorInit</code>	
<code>ErrorConfig</code>	
<code>ErrorNotImplemented</code>	
<code>ErrorResourceLimit</code>	
<code>ErrorEndOfFile</code>	
<code>ErrorFile</code>	
<code>ErrorFormat</code>	
<code>ErrorComms</code>	
<code>ErrorAccessDenied</code>	
<code>ErrorNoData</code>	
<code>ErrorEndOfData</code>	
<code>ErrorDataPresent</code>	
<code>ErrorDataTruncated</code>	
<code>ErrorApiVersion</code>	
<code>ErrorAppBase</code>	
<code>ErrorUserBase</code>	

## 8.38 BErrorTime.cpp File Reference

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

## 8.39 BErrorTime.h File Reference

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

### Classes

- class [BErrorTime](#)  
*Error return class.*

## 8.40 BEvent.cpp File Reference

```
#include <BEvent.h>  
#include <BPoll.h>  
#include <stdlib.h>  
#include <unistd.h>  
#include <sys/ioctl.h>
```

## 8.41 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](#).*



### 8.41.1 Typedef Documentation

#### 8.41.1.1 BEventQueue

```
typedef BQueue<BEvent> BEventQueue
```

This class provides an interface for sending simple integer events via a [BQueue](#).

## 8.42 BEvent1.cpp File Reference

```
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <BEvent1.h>
#include <BPoll.h>
```

## 8.43 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](#) }

### 8.43.1 Enumeration Type Documentation

#### 8.43.1.1 BEvent1Type

```
enum BEvent1Type
```

#### Enumerator

BEvent1TypeNone	
BEvent1TypeInt	
BEvent1TypeError	

## 8.44 BFifo.h File Reference

```
#include <BTypes.h>
#include <BError.h>
#include <BFifo.inc>
```

#### Classes

- class [BFifo< Type >](#)

## 8.45 BFifo.inc File Reference

## 8.46 BFifoCirc.cpp File Reference

```
#include <BFifoCirc.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/mman.h>
```

#### Macros

- #define [dprintf](#)(fmt, a...)

### 8.46.1 Macro Definition Documentation

#### 8.46.1.1 dprintf

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

## 8.47 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.*

## 8.48 BFifoCirc.inc File Reference

## 8.49 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

### 8.49.1 Macro Definition Documentation

#### 8.49.1.1 STRBUF

```
#define STRBUF 10240
```

## 8.50 BFile.h File Reference

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

## Classes

- class [BFile](#)  
*File operations class.*

## 8.51 BFileCsv.cpp File Reference

```
#include <BFileCsv.h>
#include <errno.h>
```

## 8.52 BFileCsv.h File Reference

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

## Classes

- class [BFileCsv](#)

## 8.53 BFileData.cpp File Reference

```
#include <BFileCsv.h>
#include <BFileData.h>
#include <errno.h>
```

## 8.54 BFileData.h File Reference

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

## Classes

- class [BFileData](#)

## 8.55 BFirmware.h File Reference

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

## Classes

- struct [BFirmwareFileHeader](#)
- struct [BFirmwareSegHeader](#)
- struct [BFirmwareInfo](#)

## Typedefs

- typedef [BFirmwareFileHeader](#) [BFirmwareFirmwareHeader](#)

## Functions

- struct [BFirmwareFileHeader](#) [\\_\\_attribute\\_\\_](#) ((packed))
- int [bfirmwareValid](#) ([BUInt32](#) baseAddress, [BUInt](#) type, [Bool](#) checkChecksum, char \*version=0)
- void [bfirmwareBoot](#) ([BUInt32](#) baseAddress)

## Variables

- const [BUInt32](#) [BFirmwareMagic](#) = 0x01414542
- const [BUInt32](#) [BFirmwareTypeFile](#) = 1
- const [BUInt32](#) [BFirmwareTypeFirmware](#) = 2
- const [BUInt32](#) [BFirmwareTypeSegment](#) = 3
- const [BUInt32](#) [BFirmwareFormatRaw](#) = 0
- const [BUInt32](#) [BFirmwareFormatGzip](#) = 1
- const [BUInt32](#) [BFirmwarePlatformBMeasure125](#) = 33
- const [BUInt32](#) [BFirmwarePlatformBMeasure125Cpu](#) = 34
- const [BUInt32](#) [BFirmwarePlatformBMeasure125Fpga](#) = 35
- const [BUInt32](#) [BFirmwarePlatformBMeasure125Wifi](#) = 36
- const [BUInt32](#) [BFirmwarePlatformBMeasure125Boot](#) = 37
- [BUInt32](#) magic
- [BUInt32](#) itemType
- [BUInt32](#) fileLength
- [BUInt32](#) checksum
- [BUInt32](#) platform
- [BUInt32](#) format
- [BUInt32](#) numSegments
- [BUInt32](#) startAddress
- [BUInt8](#) ver0
- [BUInt8](#) ver1
- [BUInt8](#) ver2
- [BUInt8](#) ver3
- [BUInt32](#) special [7]
- [BUInt32](#) dataLength
- [BUInt32](#) address
- [BUInt32](#) length
- const [BUInt32](#) [BFirmwareInfoMagic](#) = 0xBBEEAA00
- const [BUInt8](#) [BFirmwareInfoEncrypt1](#) = 0x40
- struct [BFirmwareInfo](#) [\\_\\_attribute\\_\\_](#)

## 8.55.1 Typedef Documentation

### 8.55.1.1 BFirmwareFirmwareHeader

```
typedef BFirmwareFileHeader BFirmwareFirmwareHeader
```

## 8.55.2 Function Documentation

### 8.55.2.1 \_\_attribute\_\_()

```
struct BFirmwareFileHeader __attribute__ (  
    (packed) )
```

### 8.55.2.2 bfirmwareBoot()

```
void bfirmwareBoot (  
    BUInt32 baseAddress )
```

### 8.55.2.3 bfirmwareValid()

```
int bfirmwareValid (  
    BUInt32 baseAddress,  
    BUInt type,  
    Bool checkChecksum,  
    char * version = 0 )
```

## 8.55.3 Variable Documentation

### 8.55.3.1 \_\_attribute\_\_

```
struct BoapMc1Error __attribute__
```

### 8.55.3.2 address

`BUInt32` address

### 8.55.3.3 BFirmwareFormatGzip

`const BUInt32 BFirmwareFormatGzip = 1`

### 8.55.3.4 BFirmwareFormatRaw

`const BUInt32 BFirmwareFormatRaw = 0`

### 8.55.3.5 BFirmwareInfoEncrypt1

`const BUInt8 BFirmwareInfoEncrypt1 = 0x40`

### 8.55.3.6 BFirmwareInfoMagic

`const BUInt32 BFirmwareInfoMagic = 0xBBEEAA00`

### 8.55.3.7 BFirmwareMagic

`const BUInt32 BFirmwareMagic = 0x01414542`

### 8.55.3.8 BFirmwarePlatformBMeasure125

`const BUInt32 BFirmwarePlatformBMeasure125 = 33`

### 8.55.3.9 BFirmwarePlatformBMeasure125Boot

`const BUInt32 BFirmwarePlatformBMeasure125Boot = 37`

#### 8.55.3.10 BFirmwarePlatformBMeasure125Cpu

```
const BUInt32 BFirmwarePlatformBMeasure125Cpu = 34
```

#### 8.55.3.11 BFirmwarePlatformBMeasure125Fpga

```
const BUInt32 BFirmwarePlatformBMeasure125Fpga = 35
```

#### 8.55.3.12 BFirmwarePlatformBMeasure125Wifi

```
const BUInt32 BFirmwarePlatformBMeasure125Wifi = 36
```

#### 8.55.3.13 BFirmwareTypeFile

```
const BUInt32 BFirmwareTypeFile = 1
```

#### 8.55.3.14 BFirmwareTypeFirmware

```
const BUInt32 BFirmwareTypeFirmware = 2
```

#### 8.55.3.15 BFirmwareTypeSegment

```
const BUInt32 BFirmwareTypeSegment = 3
```

#### 8.55.3.16 checksum

```
BUInt32 checksum
```

#### 8.55.3.17 dataLength

```
BUInt32 dataLength
```



**8.55.3.18   fileLength**

BUInt32 fileLength

**8.55.3.19   format**

BUInt32 format

**8.55.3.20   itemType**

BUInt32 itemType

**8.55.3.21   length**

BUInt32 length

**8.55.3.22   magic**

BUInt32 magic

**8.55.3.23   numSegments**

BUInt32 numSegments

**8.55.3.24   platform**

BUInt32 platform

**8.55.3.25   special**

BUInt32 special

#### 8.55.3.26 startAddress

`BUInt32 startAddress`

#### 8.55.3.27 ver0

`BUInt8 ver0`

#### 8.55.3.28 ver1

`BUInt8 ver1`

#### 8.55.3.29 ver2

`BUInt8 ver2`

#### 8.55.3.30 ver3

`BUInt8 ver3`

## 8.56 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))`

## 8.56.1 Macro Definition Documentation

### 8.56.1.1 BListLoop

```
#define BListLoop(  
    list,  
    i ) for(BIter i = list.begin(); !list.isEnd(i); list.next(i))
```

## 8.57 BList\_func.h File Reference

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

## 8.58 BMutex.cpp File Reference

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

### Macros

- #define MDEBUG 0

## 8.58.1 Macro Definition Documentation

### 8.58.1.1 MDEBUG

```
#define MDEBUG 0
```

## 8.59 BMutex.h File Reference

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

## Classes

- class [BMutex](#)
- class [BMutexLock](#)

## 8.60 BMySQL.cpp File Reference

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

## 8.61 BMySQL.h File Reference

```
#include <BTypes.h>
#include <BError.h>
#include <BDict.h>
#include <BMutex.h>
#include <mysql/mysql.h>
```

## Classes

- class [BMySQL](#)

## 8.62 BNameValue.h File Reference

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

## Classes

- class [BNameValue< T >](#)
- class [BNameValueList< T >](#)

## 8.63 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.*

### 8.63.1 Macro Definition Documentation

#### 8.63.1.1 APIVERSION\_TEST

```
#define APIVERSION_TEST 1
```

#### 8.63.1.2 DEBUG

```
#define DEBUG 0
```

#### 8.63.1.3 dprintf

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

#### 8.63.1.4 IS\_BIG\_ENDIAN

```
#define IS_BIG_ENDIAN 1
```

### 8.63.2 Variable Documentation

### 8.63.2.1 boapPort

```
const int boapPort = 12000
```

The default BOAP connection port.

## 8.64 Boap.d File Reference

## 8.65 build\_x86\_64/Boap.d File Reference

## 8.66 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

### 8.66.1 Typedef Documentation

#### 8.66.1.1 BoapFunc

```
typedef BError(BoapServiceObject::* BoapFunc) (BoapServerConnection *conn, BoapPacket &rx,  
BoapPacket &tx)
```

#### 8.66.1.2 BoapService

```
typedef BUInt32 BoapService
```

### 8.66.2 Enumeration Type Documentation

#### 8.66.2.1 BoapPriority

```
enum BoapPriority
```

##### Enumerator

<code>BoapPriorityLow</code>	
<code>BoapPriorityNormal</code>	
<code>BoapPriorityHigh</code>	

#### 8.66.2.2 BoapType

```
enum BoapType
```

### Enumerator

BoapTypeRpc	
BoapTypeRpcReply	
BoapTypeSignal	
BoapTypeRpcError	
BoapTypeRpc	
BoapTypeSignal	

## 8.66.3 Variable Documentation

### 8.66.3.1 BoapMagic

```
const BUInt32 BoapMagic = 0x424F4100
```

## 8.67 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...)

### 8.67.1 Macro Definition Documentation

#### 8.67.1.1 DEBUG\_LOCAL

```
#define DEBUG_LOCAL 0
```



### 8.67.1.2 `DEBUG_LOCAL1`

```
#define DEBUG_LOCAL1 0
```

### 8.67.1.3 `dl1printf`

```
#define dl1printf(  
    fmt,  
    a... )
```

### 8.67.1.4 `dlprintf`

```
#define dlprintf(  
    fmt,  
    a... )
```

## 8.68 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\\_\\_](#)

## 8.68.1 Enumeration Type Documentation

### 8.68.1.1 BoapMcType

enum [BoapMcType](#)

#### Enumerator

<a href="#">BoapMcTypeRequest</a>	
<a href="#">BoapMcTypeReply</a>	

## 8.68.2 Function Documentation

### 8.68.2.1 [\\_\\_attribute\\_\\_\(\)](#)

```
struct BoapMcPacketHead \_\_attribute\_\_ (  
    (aligned(8), packed) )
```

## 8.68.3 Variable Documentation

### 8.68.3.1 [\\_\\_attribute\\_\\_](#)

```
class BoapMcPacket \_\_attribute\_\_
```

#### 8.68.3.2 addressFrom

`BUInt8` addressFrom

#### 8.68.3.3 addressTo

`BUInt8` addressTo

#### 8.68.3.4 checksum

`BUInt16` checksum

#### 8.68.3.5 cmd

`BUInt8` cmd

#### 8.68.3.6 error

`BUInt16` error

#### 8.68.3.7 length

`BUInt8` length

## 8.69 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`

### 8.69.1 Macro Definition Documentation

#### 8.69.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

#### 8.69.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

## 8.70 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](#)  
*Packet magic pattern.*
- [BUInt16 length](#)  
*Total packet length including the header.*
- [BUInt16 addressTo](#)  
*Address to send to.*
- [BUInt16 addressFrom](#)  
*Address packet is from.*
- [BUInt16 cmd](#)  
*The RPC command or reply number.*
- [BInt16 error](#)  
*Error number.*
- [BUInt32 checksum](#)  
*Packet checksum, when used.*
- [BoapMc1PacketHead head](#)
- char [data](#) [8]
- [BInt16 number](#)  
*The error number.*
- char [string](#) [32]  
*The error string.*
- class [BoapMc1Comms \\_\\_attribute\\_\\_](#)

## 8.70.1 Enumeration Type Documentation

### 8.70.1.1 BoapMc1Type

enum [BoapMc1Type](#)

#### Enumerator

<a href="#">BoapMc1TypeRequest</a>	
<a href="#">BoapMc1TypeReply</a>	

## 8.70.2 Function Documentation

### 8.70.2.1 \_\_attribute\_\_()

```
struct BoapMc1PacketHead __attribute__ (
    (aligned(8), packed) )
```

### 8.70.2.2 boapMc1CommsRoundupLen()

```
BUInt32 boapMc1CommsRoundupLen (
    BUInt32 len ) [inline]
```

## 8.70.3 Variable Documentation

### 8.70.3.1 \_\_attribute\_\_

```
class BoapMc1Comms __attribute__
```

### 8.70.3.2 addressFrom

```
BUInt16 addressFrom
```

Address packet is from.

### 8.70.3.3 addressTo

```
BUInt16 addressTo
```

Address to send to.

### 8.70.3.4 BoapMc1Magic

```
const BUInt16 BoapMc1Magic = 0x5542
```

### 8.70.3.5 checksum

```
BUInt32 checksum
```

Packet checksum, when used.

#### 8.70.3.6 cmd

`BUInt16 cmd`

The RPC command or reply number.

#### 8.70.3.7 data

`char data[8]`

#### 8.70.3.8 error

`BInt16 error`

Error number.

#### 8.70.3.9 head

`BoapMc1PacketHead head`

#### 8.70.3.10 length

`BUInt16 length`

Total packet length including the header.

#### 8.70.3.11 magic

`BUInt16 magic`

Packet magic pattern.

#### 8.70.3.12 number

`BInt16` number

The error number.

#### 8.70.3.13 string

`char` string[32]

The error string.

## 8.71 BoapnsC.cpp File Reference

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

### Namespaces

- [Boapns](#)

## 8.72 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



## 8.73 BoapnsD.cpp File Reference

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

### Namespaces

- [Boapns](#)

## 8.74 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](#)

## 8.75 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

## 8.75.1 Macro Definition Documentation

### 8.75.1.1 DEBUG

```
#define DEBUG 0
```

### 8.75.1.2 dprintf

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

## 8.75.2 Variable Documentation

### 8.75.2.1 roundSize

```
const int roundSize = 256
```

## 8.76 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](#) }

### 8.76.1 Typedef Documentation

#### 8.76.1.1 BoapFunc

```
typedef BError(BoapServiceObject::* BoapFunc) (BoapPacket &rx, BoapPacket &tx)
```

#### 8.76.1.2 BoapService

```
typedef uint32_t BoapService
```

#### 8.76.1.3 Double

```
typedef double Double
```

#### 8.76.1.4 Int16

```
typedef int16_t Int16
```

#### 8.76.1.5 Int32

```
typedef int32_t Int32
```

#### 8.76.1.6 Int8

```
typedef int8_t Int8
```

#### 8.76.1.7 UInt16

```
typedef uint16_t UInt16
```

#### 8.76.1.8 UInt32

```
typedef uint32_t UInt32
```

#### 8.76.1.9 UInt8

```
typedef uint8_t UInt8
```

### 8.76.2 Enumeration Type Documentation

#### 8.76.2.1 BoapType

```
enum BoapType
```

##### Enumerator

BoapTypeRpc	
BoapTypeRpcReply	
BoapTypeSignal	
BoapTypeRpcError	
BoapTypeRpc	
BoapTypeSignal	

## 8.77 BObj.cpp File Reference

```
#include <BObj.h>
```

## 8.78 BObj.h File Reference

```
#include <BTypes.h>
#include <BDict.h>
#include <BString.h>
#include <BError.h>
```

### Classes

- class [BObj](#)

## 8.79 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)

## 8.79.1 Function Documentation

### 8.79.1.1 toBDictStringFromJson()

```
BError toBDictStringFromJson (
    BString json,
    BDictString & ds )
```

### 8.79.1.2 toBString() [1/18]

```
BString toBString (
    BString n,
    Bool v )
```

### 8.79.1.3 toBString() [2/18]

```
BString toBString (
    BString n,
    BInt8 v )
```

### 8.79.1.4 toBString() [3/18]

```
BString toBString (
    BString n,
    BUInt8 v )
```

**8.79.1.5 toBString()** [4/18]

```
BString toBString (  
    BString n,  
    BInt16 v )
```

**8.79.1.6 toBString()** [5/18]

```
BString toBString (  
    BString n,  
    BUInt16 v )
```

**8.79.1.7 toBString()** [6/18]

```
BString toBString (  
    BString n,  
    BInt32 v )
```

**8.79.1.8 toBString()** [7/18]

```
BString toBString (  
    BString n,  
    BUInt32 v )
```

**8.79.1.9 toBString()** [8/18]

```
BString toBString (  
    BString n,  
    BInt64 v )
```

**8.79.1.10 toBString()** [9/18]

```
BString toBString (  
    BString n,  
    BUInt64 v )
```

**8.79.1.11 toBString()** [10/18]

```
BString toBString (
    BString n,
    BFloat32 v )
```

**8.79.1.12 toBString()** [11/18]

```
BString toBString (
    BString n,
    BFloat64 v )
```

**8.79.1.13 toBString()** [12/18]

```
BString toBString (
    BString n,
    BChar v )
```

**8.79.1.14 toBString()** [13/18]

```
BString toBString (
    BString n,
    const BChar * v )
```

**8.79.1.15 toBString()** [14/18]

```
BString toBString (
    BString n,
    BString v )
```

**8.79.1.16 toBString()** [15/18]

```
BString toBString (
    BString n,
    BError v )
```



**8.79.1.17 toBString()** [16/18]

```
BString toBString (
    BString n,
    BTime v )
```

**8.79.1.18 toBString()** [17/18]

```
BString toBString (
    BString name,
    const BObjMember * m,
    const void * obj,
    BStringList ignoreFields )
```

**8.79.1.19 toBString()** [18/18]

```
BString toBString (
    BString n,
    BObj & obj )
```

**8.79.1.20 toBStringJson()** [1/18]

```
BString toBStringJson (
    BString n,
    Bool v )
```

**8.79.1.21 toBStringJson()** [2/18]

```
BString toBStringJson (
    BString n,
    BInt8 v )
```

**8.79.1.22 toBStringJson()** [3/18]

```
BString toBStringJson (
    BString n,
    BUInt8 v )
```

**8.79.1.23 toBStringJson()** [4/18]

```
BString toBStringJson (
    BString n,
    BInt16 v )
```

**8.79.1.24 toBStringJson()** [5/18]

```
BString toBStringJson (
    BString n,
    BUInt16 v )
```

**8.79.1.25 toBStringJson()** [6/18]

```
BString toBStringJson (
    BString n,
    BInt32 v )
```

**8.79.1.26 toBStringJson()** [7/18]

```
BString toBStringJson (
    BString n,
    BUInt32 v )
```

**8.79.1.27 toBStringJson()** [8/18]

```
BString toBStringJson (
    BString n,
    BInt64 v )
```

**8.79.1.28 toBStringJson()** [9/18]

```
BString toBStringJson (
    BString n,
    BUInt64 v )
```

**8.79.1.29 toBStringJson()** [10/18]

```
BString toBStringJson (
    BString n,
    BFloat32 v )
```

**8.79.1.30 toBStringJson()** [11/18]

```
BString toBStringJson (
    BString n,
    BFloat64 v )
```

**8.79.1.31 toBStringJson()** [12/18]

```
BString toBStringJson (
    BString n,
    BChar v )
```

**8.79.1.32 toBStringJson()** [13/18]

```
BString toBStringJson (
    BString n,
    const BChar * v )
```

**8.79.1.33 toBStringJson()** [14/18]

```
BString toBStringJson (
    BString n,
    BString v )
```

**8.79.1.34 toBStringJson()** [15/18]

```
BString toBStringJson (
    BString n,
    BError v )
```

**8.79.1.35 toBStringJson()** [16/18]

```
BString toBStringJson (
    BString n,
    BTime v )
```

**8.79.1.36 toBStringJson()** [17/18]

```
BString toBStringJson (
    BString n,
    const BObjMember * m,
    const void * obj,
    BStringList ignoreFields )
```

**8.79.1.37 toBStringJson()** [18/18]

```
BString toBStringJson (
    BString n,
    BObj & obj )
```

**8.80 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)

## 8.80.1 Function Documentation

### 8.80.1.1 base64\_decode()

```
BError base64_decode (
    BString strIn,
    BString & strOut )
```

### 8.80.1.2 base64\_encode()

```
BString base64_encode (
    void * data,
    BUInt len )
```

### 8.80.1.3 toBDictStringFromJson()

```
BError toBDictStringFromJson (
    BString json,
    BDictString & ds )
```

**8.80.1.4 toBString()** [1/18]

```
BString toBString (
    BString name,
    Bool value )
```

**8.80.1.5 toBString()** [2/18]

```
BString toBString (
    BString name,
    BInt8 value )
```

**8.80.1.6 toBString()** [3/18]

```
BString toBString (
    BString name,
    BUInt8 value )
```

**8.80.1.7 toBString()** [4/18]

```
BString toBString (
    BString name,
    BInt16 value )
```

**8.80.1.8 toBString()** [5/18]

```
BString toBString (
    BString name,
    BUInt16 value )
```

**8.80.1.9 toBString()** [6/18]

```
BString toBString (
    BString name,
    BInt32 value )
```

**8.80.1.10 toBString()** [7/18]

```
BString toBString (
    BString name,
    BUInt32 value )
```

**8.80.1.11 toBString()** [8/18]

```
BString toBString (
    BString name,
    BInt64 value )
```

**8.80.1.12 toBString()** [9/18]

```
BString toBString (
    BString name,
    BUInt64 value )
```

**8.80.1.13 toBString()** [10/18]

```
BString toBString (
    BString name,
    BFloat32 value )
```

**8.80.1.14 toBString()** [11/18]

```
BString toBString (
    BString name,
    BFloat64 value )
```

**8.80.1.15 toBString()** [12/18]

```
BString toBString (
    BString name,
    BChar value )
```

**8.80.1.16 toBString()** [13/18]

```
BString toBString (
    BString name,
    const BChar * value )
```

**8.80.1.17 toBString()** [14/18]

```
BString toBString (
    BString name,
    BString value )
```

**8.80.1.18 toBString()** [15/18]

```
BString toBString (
    BString name,
    BError value )
```

**8.80.1.19 toBString()** [16/18]

```
BString toBString (
    BString name,
    BTime time )
```

**8.80.1.20 toBString()** [17/18]

```
BString toBString (
    BString name,
    const BObjMember * members,
    const void * obj,
    BStringList ignoreFields = BStringList() )
```

**8.80.1.21 toBString()** [18/18]

```
BString toBString (
    BString name,
    BObj & obj )
```



**8.80.1.22 toBStringJson()** [1/18]

```
BString toBStringJson (
    BString name,
    Bool value )
```

**8.80.1.23 toBStringJson()** [2/18]

```
BString toBStringJson (
    BString name,
    BInt8 value )
```

**8.80.1.24 toBStringJson()** [3/18]

```
BString toBStringJson (
    BString name,
    BUInt8 value )
```

**8.80.1.25 toBStringJson()** [4/18]

```
BString toBStringJson (
    BString name,
    BInt16 value )
```

**8.80.1.26 toBStringJson()** [5/18]

```
BString toBStringJson (
    BString name,
    BUInt16 value )
```

**8.80.1.27 toBStringJson()** [6/18]

```
BString toBStringJson (
    BString name,
    BInt32 value )
```

**8.80.1.28 toBStringJson()** [7/18]

```
BString toBStringJson (
    BString name,
    BUInt32 value )
```

**8.80.1.29 toBStringJson()** [8/18]

```
BString toBStringJson (
    BString name,
    BInt64 value )
```

**8.80.1.30 toBStringJson()** [9/18]

```
BString toBStringJson (
    BString name,
    BUInt64 value )
```

**8.80.1.31 toBStringJson()** [10/18]

```
BString toBStringJson (
    BString name,
    BFloat32 value )
```

**8.80.1.32 toBStringJson()** [11/18]

```
BString toBStringJson (
    BString name,
    BFloat64 value )
```

**8.80.1.33 toBStringJson()** [12/18]

```
BString toBStringJson (
    BString name,
    BChar value )
```

**8.80.1.34 toBStringJson()** [13/18]

```
BString toBStringJson (
    BString name,
    const BChar * value )
```

**8.80.1.35 toBStringJson()** [14/18]

```
BString toBStringJson (
    BString name,
    BString value )
```

**8.80.1.36 toBStringJson()** [15/18]

```
BString toBStringJson (
    BString name,
    BError value )
```

**8.80.1.37 toBStringJson()** [16/18]

```
BString toBStringJson (
    BString name,
    BTime time )
```

**8.80.1.38 toBStringJson()** [17/18]

```
BString toBStringJson (
    BString name,
    const BObjMember * members,
    const void * obj,
    BStringList ignoreFields = BStringList() )
```

**8.80.1.39 toBStringJson()** [18/18]

```
BString toBStringJson (
    BString name,
    BObj & obj )
```

## 8.81 BPoll.cpp File Reference

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

## 8.82 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.*

## 8.83 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](#)

### 8.83.1 Typedef Documentation

#### 8.83.1.1 BQueueInt

```
typedef BQueue<BInt32> BQueueInt
```

## 8.84 BRefData.cpp File Reference

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

### Macros

- `#define` [CHUNK](#) 16

### 8.84.1 Macro Definition Documentation

#### 8.84.1.1 [CHUNK](#)

```
#define CHUNK 16
```

## 8.85 BRefData.h File Reference

```
#include <BAAtomicCount.h>
```

### Classes

- class [BRefData](#)

## 8.86 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>
```

## 8.87 BRtc.h File Reference

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

### Classes

- class [BRtc](#)  
*Realtime clock.*
- class [BRtcThreaded](#)  
*Threaded real time clock.*

## 8.88 BRWLock.cpp File Reference

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

## 8.89 BRWLock.h File Reference

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

### Classes

- class [BRWLock](#)  
*thread read-write locks*

## 8.90 BSema.cpp File Reference

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

## 8.91 BSema.h File Reference

```
#include <sys/types.h>
#include <semaphore.h>
```

## Classes

- class [BSema](#)  
*Sempahore class.*

## 8.92 BSemaphore.cpp File Reference

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

## 8.93 BSemaphore.h File Reference

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

## Classes

- class [BSemaphore](#)  
*Semaphore class.*
- class [BSemaphoreBool](#)
- class [BSemaphoreCount](#)

## 8.94 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`

## 8.94.1 Macro Definition Documentation

### 8.94.1.1 IP\_MTU

```
#define IP_MTU 14
```

## 8.95 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

## 8.95.1 Macro Definition Documentation

### 8.95.1.1 MSG\_NOSIGNAL

```
#define MSG_NOSIGNAL 0
```



### 8.95.1.2 SO\_PRIORITY

```
#define SO_PRIORITY 12
```

### 8.95.1.3 SOL\_IP

```
#define SOL_IP 0
```

## 8.96 BSpI.cpp File Reference

```
#include <BSpI.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/ioctl.h>
#include <linux/spi/spidev.h>
```

## 8.97 BSpI.h File Reference

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

### Classes

- class [BSpI](#)  
*BSpI class.*

## 8.98 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](#) []

### 8.98.1 Macro Definition Documentation

#### 8.98.1.1 MINUS

```
#define MINUS '-'
```

#### 8.98.1.2 STRIP

```
#define STRIP 0x7f
```

## 8.98.2 Function Documentation

### 8.98.2.1 barrayToString()

```
BString barrayToString (
    const BStringArray & list )
```

### 8.98.2.2 blistToString()

```
BString blistToString (
    const BStringList & list )
```

### 8.98.2.3 bstringListinList()

```
int bstringListinList (
    BStringList & list,
    BString s )
```

### 8.98.2.4 bstringToArray()

```
BStringArray bstringToArray (
    BString str,
    int stripSpaces )
```

### 8.98.2.5 bstringToList()

```
BStringList bstringToList (
    BString str,
    int stripSpaces )
```

### 8.98.2.6 bstrncpy()

```
char* bstrncpy (
    char * dest,
    const char * src,
    size_t n )
```

**8.98.2.7 bstrtrim()**

```
char* bstrtrim (
    char * str )
```

**8.98.2.8 charToArray()**

```
BStringArray charToArray (
    const char ** str )
```

**8.98.2.9 charToList()**

```
BStringList charToList (
    const char ** str )
```

**8.98.2.10 floatToString()**

```
const char* floatToString (
    char * str,
    BUInt strLen,
    BFloat32 f,
    BUInt precision )
```

**8.98.2.11 fromBString()** [1/6]

```
void fromBString (
    BString & s,
    BString & v )
```

**8.98.2.12 fromBString()** [2/6]

```
void fromBString (
    BString & s,
    BStringList & v )
```

**8.98.2.13 fromBString()** [3/6]

```
void fromBString (
    BString & s,
    BInt32 & v )
```

**8.98.2.14 fromBString()** [4/6]

```
void fromBString (
    BString & s,
    BUInt32 & v )
```

**8.98.2.15 fromBString()** [5/6]

```
void fromBString (
    BString & s,
    BUInt64 & v )
```

**8.98.2.16 fromBString()** [6/6]

```
void fromBString (
    BString & s,
    BFloat64 & v )
```

**8.98.2.17 gmatch()**

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

**8.98.2.18 int64ToString()**

```
const char* int64ToString (
    char * str,
    BUInt strLen,
    BInt64 value,
    int base )
```

#### 8.98.2.19 intToString()

```
const char* intToString (
    char * str,
    BUInt strLen,
    int value,
    int base )
```

#### 8.98.2.20 operator<<()

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

#### 8.98.2.21 operator>>()

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

#### 8.98.2.22 toBString() [1/6]

```
void toBString (
    BString & v,
    BString & s )
```

#### 8.98.2.23 toBString() [2/6]

```
void toBString (
    BStringList & v,
    BString & s )
```

#### 8.98.2.24 toBString() [3/6]

```
void toBString (
    BInt32 & v,
    BString & s )
```

```
void toBString (
    BUInt32 & v,
    BString & s )
```

```
void toBString (
    BUInt64 & v,
    BString & s )
```

```
void toBString (
    BFloat64 & v,
    BString & s )
```

### 8.98.3.1 base64\_decode\_table

**Initial value:**

[illegible]

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

## 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)`

## 8.99.1 Function Documentation

### 8.99.1.1 [bstrncpy\(\)](#)

```
char* bstrncpy (
    char * dest,
    const char * src,
    size_t n )
```

### 8.99.1.2 [bstrtrim\(\)](#)

```
char* bstrtrim (
    char * str )
```



### 8.99.1.3 floatToString()

```
const char* floatToString (
    char * str,
    BUInt strLen,
    BFloat32 f,
    BUInt precision )
```

### 8.99.1.4 from\_hex()

```
char from_hex (
    char ch ) [inline]
```

### 8.99.1.5 fromBString() [1/6]

```
void fromBString (
    BString & s,
    BString & v )
```

### 8.99.1.6 fromBString() [2/6]

```
void fromBString (
    BString & s,
    BStringList & v )
```

### 8.99.1.7 fromBString() [3/6]

```
void fromBString (
    BString & s,
    BInt32 & v )
```

### 8.99.1.8 fromBString() [4/6]

```
void fromBString (
    BString & s,
    BUInt32 & v )
```

**8.99.1.9 fromBString()** [5/6]

```
void fromBString (
    BString & s,
    BUInt64 & v )
```

**8.99.1.10 fromBString()** [6/6]

```
void fromBString (
    BString & s,
    BFloat64 & v )
```

**8.99.1.11 int64ToString()**

```
const char* int64ToString (
    char * str,
    BUInt strLen,
    BInt64 value,
    int base = 10 )
```

**8.99.1.12 intToString()**

```
const char* intToString (
    char * str,
    BUInt strLen,
    int value,
    int base = 10 )
```

**8.99.1.13 operator<<()**

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

**8.99.1.14 operator>>()**

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

#### 8.99.1.15 to\_hex()

```
char to_hex (
    char code )    [inline]
```

#### 8.99.1.16 toBString() [1/6]

```
void toBString (
    BString & v,
    BString & s )
```

#### 8.99.1.17 toBString() [2/6]

```
void toBString (
    BStringList & v,
    BString & s )
```

#### 8.99.1.18 toBString() [3/6]

```
void toBString (
    BInt32 & v,
    BString & s )
```

#### 8.99.1.19 toBString() [4/6]

```
void toBString (
    BUInt32 & v,
    BString & s )
```

#### 8.99.1.20 toBString() [5/6]

```
void toBString (
    BUInt64 & v,
    BString & s )
```

#### 8.99.1.21 toBString() [ 6 / 6 ]

```
void toBString (
    BFloat64 & v,
    BString & s )
```

### 8.100 BStringLocked.h File Reference

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

#### Classes

- class [BStringMutex](#)
- class [BStringLocked](#)

### 8.101 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.*

#### 8.101.1 Function Documentation

##### 8.101.1.1 delayMs()

```
void delayMs (
    BUInt ms )
```

Will delay for given time in ms, if tasks running task will sleep.

### 8.101.1.2 delayUs()

```
void delayUs (  
    BUInt us )
```

Will delay for given time in us, if tasks running task will sleep.

## 8.102 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.*

### 8.102.1 Function Documentation

#### 8.102.1.1 delayMs()

```
void delayMs (  
    BUInt ms )
```

Will delay for given time in ms, if tasks running task will sleep.

#### 8.102.1.2 delayUs()

```
void delayUs (  
    BUInt us )
```

Will delay for given time in us, if tasks running task will sleep.

## 8.103 BTable.cpp File Reference

```
#include <BTable.h>
```

## 8.104 BTable.h File Reference

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

### Classes

- class [BTable](#)

## 8.105 BTask.cpp File Reference

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

## 8.106 BTask.h File Reference

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

### Classes

- class [BTask](#)

## 8.107 BThread.cpp File Reference

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

## 8.108 BThread.h File Reference

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

### Classes

- class [BThread](#)

## 8.109 BTime.cpp File Reference

```
#include <BTime.h>
```

### Functions

- static bool [yearIsLeap](#) ([BUInt16](#) year)
- static [BUInt16](#) [yearDays](#) ([BUInt16](#) year)

### Variables

- static [BUInt16](#) [monDays](#) [2][13]

### 8.109.1 Function Documentation

#### 8.109.1.1 yearDays()

```
static BUInt16 yearDays (  
    BUInt16 year )    [inline], [static]
```

#### 8.109.1.2 yearIsLeap()

```
static bool yearIsLeap (  
    BUInt16 year )    [inline], [static]
```

### 8.109.2 Variable Documentation

#### 8.109.2.1 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 }  
}
```

## 8.110 BTime.h File Reference

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

### Classes

- class [BTime](#)

## 8.111 BTimer.cpp File Reference

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

## 8.112 BTimer.h File Reference

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

### Classes

- class [BTimer](#)  
*Stopwatch style timer.*

## 8.113 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]



### 8.113.1 Function Documentation

#### 8.113.1.1 fromBString()

```
void fromBString (
    BString & s,
    BTimeStamp & v )
```

#### 8.113.1.2 toBString()

```
void toBString (
    BTimeStamp & v,
    BString & s )
```

### 8.113.2 Variable Documentation

#### 8.113.2.1 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 }
}
```

## 8.114 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)

### 8.114.1 Function Documentation

#### 8.114.1.1 fromBString()

```
void fromBString (
    BString & s,
    BTimeStamp & v )
```

#### 8.114.1.2 toBString()

```
void toBString (
    BTimeStamp & v,
    BString & s )
```

## 8.115 BTimeStampMs.cpp File Reference

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

### Variables

- static int `mon_yday` [2][13]

### 8.115.1 Variable Documentation

#### 8.115.1.1 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 }
}
```

## 8.116 BTimeStampMs.h File Reference

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

### Classes

- class [BTimeStampMs](#)

## 8.117 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]

### 8.117.1 Function Documentation

#### 8.117.1.1 yearDays()

```
static BUInt16 yearDays (
    BUInt16 year ) [inline], [static]
```

#### 8.117.1.2 yearIsLeap()

```
static bool yearIsLeap (
    BUInt16 year ) [inline], [static]
```

### 8.117.2 Variable Documentation

### 8.117.2.1 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 }
}
```

## 8.118 BTimeUs.h File Reference

```
#include <BTypes.h>
#include <BError.h>
#include <BString.h>
#include <BTime.h>
```

### Classes

- class [BTimeUs](#)

## 8.119 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`, `BEventTypeReadLine`,  
`BEventTypeWrite`, `BEventTypeConnect`, `BEventTypeDisconnect`, `BEventTypeClientConnect`,  
`BEventTypeClientDisconnect` }
- enum `BEventWaitSet` {  
`BEventWaitNone` = 0x00, `BEventWaitError` = 0x01, `BEventWaitRead` = 0x02, `BEventWaitReadLine` = 0x04,  
`BEventWaitWrite` = 0x08, `BEventWaitConnect` = 0x10, `BEventWaitDisconnect` = 0x20, `BEventWaitClientConnect`  
= 0x40,  
`BEventWaitClientDisconnect` = 0x80, `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

### 8.119.1 Typedef Documentation

#### 8.119.1.1 BArrayDouble

```
typedef std::vector<BFloat64> BArrayDouble
```

#### 8.119.1.2 BArrayFloat

```
typedef std::vector<BFloat32> BArrayFloat
```

**8.119.1.3 BChar**

```
typedef char BChar
```

**8.119.1.4 BDouble**

```
typedef BFloat64 BDouble
```

**8.119.1.5 BFloat**

```
typedef BFloat32 BFloat
```

**8.119.1.6 BFloat32**

```
typedef float BFloat32
```

**8.119.1.7 BFloat64**

```
typedef double BFloat64
```

**8.119.1.8 Bint**

```
typedef Bint32 Bint
```

**8.119.1.9 Bint16**

```
typedef int16_t Bint16
```

**8.119.1.10 Bint32**

```
typedef int32_t Bint32
```

#### 8.119.1.11 BInt64

```
typedef int64_t BInt64
```

#### 8.119.1.12 BInt8

```
typedef int8_t BInt8
```

#### 8.119.1.13 Bool

```
typedef bool Bool
```

#### 8.119.1.14 BSize

```
typedef size_t BSize
```

#### 8.119.1.15 BTimeout

```
typedef BUInt32 BTimeout
```

#### 8.119.1.16 BUInt

```
typedef BUInt32 BUInt
```

#### 8.119.1.17 BUInt16

```
typedef uint16_t BUInt16
```

#### 8.119.1.18 BUInt32

```
typedef uint32_t BUInt32
```

### 8.119.1.19 BUInt64

```
typedef uint64_t BUInt64
```

### 8.119.1.20 BUInt8

```
typedef uint8_t BUInt8
```

## 8.119.2 Enumeration Type Documentation

### 8.119.2.1 BEventType

```
enum BEventType
```

#### Enumerator

BEventTypeNone	
BEventTypeError	
BEventTypeRead	
BEventTypeReadLine	
BEventTypeWrite	
BEventTypeConnect	
BEventTypeDisconnect	
BEventTypeClientConnect	
BEventTypeClientDisconnect	

### 8.119.2.2 BEventWaitSet

```
enum BEventWaitSet
```

#### Enumerator

BEventWaitNone	
BEventWaitError	
BEventWaitRead	
BEventWaitReadLine	
BEventWaitWrite	
BEventWaitConnect	
BEventWaitDisconnect	
BEventWaitClientConnect	
BEventWaitClientDisconnect	
BEventWaitAny	



### 8.119.2.3 BType

enum [BType](#)

#### Enumerator

BTypeNone	
BTypeBool	
BTypeInt8	
BTypeUInt8	
BTypeInt16	
BTypeUInt16	
BTypeInt32	
BTypeUInt32	
BTypeInt64	
BTypeUInt64	
BTypeFloat32	
BTypeFloat64	
BTypeChar	
BTypeString	
BTypeError	
BTypeTime	
BTypeTimeUs	
BTypeObj	

### 8.119.2.4 BTypeComp

enum [BTypeComp](#)

#### Enumerator

BTypeCompSingle	
BTypeCompArray	
BTypeCompArrayFixed	
BTypeCompList	
BTypeCompDict	

## 8.119.3 Function Documentation

#### 8.119.3.1 byteSwap16()

```
void byteSwap16 (
    void * d,
    void * s ) [inline]
```

#### 8.119.3.2 byteSwap32()

```
void byteSwap32 (
    void * d,
    void * s ) [inline]
```

#### 8.119.3.3 byteSwap64()

```
void byteSwap64 (
    void * d,
    void * s ) [inline]
```

#### 8.119.3.4 byteSwap8()

```
void byteSwap8 (
    void * d,
    void * s ) [inline]
```

#### 8.119.3.5 timeoutTicks()

```
BTimeout timeoutTicks (
    BTimeout timeoutUs ) [inline]
```

### 8.119.4 Variable Documentation

#### 8.119.4.1 BTimeoutForever

```
const BTimeout BTimeoutForever = 0xFFFFFFFF
```

8.120 build\_x86\_64/BBuffer.d File Reference

8.121 build\_x86\_64/BComms.d File Reference

8.122 build\_x86\_64/BCond.d File Reference

8.123 build\_x86\_64/BCondInt.d File Reference

8.124 build\_x86\_64/BConfig.d File Reference

8.125 build\_x86\_64/BCrc16.d File Reference

8.126 build\_x86\_64/BCrc32.d File Reference

8.127 build\_x86\_64/BDate.d File Reference

8.128 build\_x86\_64/BDebug.d File Reference

8.129 build\_x86\_64/BDict.d File Reference

8.130 build\_x86\_64/BDir.d File Reference

8.131 build\_x86\_64/BDuration.d File Reference

8.132 build\_x86\_64/BEndian.d File Reference

8.133 build\_x86\_64/BEntry.d File Reference

8.134 build\_x86\_64/BError.d File Reference

8.135 build\_x86\_64/BErrorTime.d File Reference

8.136 build\_x86\_64/BEvent.d File Reference

- 8.137 [build\\_x86\\_64/BEvent1.d File Reference](#)
- 8.138 [build\\_x86\\_64/BFifoCirc.d File Reference](#)
- 8.139 [build\\_x86\\_64/BFile.d File Reference](#)
- 8.140 [build\\_x86\\_64/BFileCsv.d File Reference](#)
- 8.141 [build\\_x86\\_64/BFileData.d File Reference](#)
- 8.142 [build\\_x86\\_64/BMutex.d File Reference](#)
- 8.143 [build\\_x86\\_64/BMysql.d File Reference](#)
- 8.144 [build\\_x86\\_64/BoapMc.d File Reference](#)
- 8.145 [build\\_x86\\_64/BoapMc1.d File Reference](#)
- 8.146 [build\\_x86\\_64/BoapnsC.d File Reference](#)
- 8.147 [build\\_x86\\_64/BoapnsD.d File Reference](#)
- 8.148 [build\\_x86\\_64/BObj.d File Reference](#)
- 8.149 [build\\_x86\\_64/BObjStringFormat.d File Reference](#)
- 8.150 [build\\_x86\\_64/BPoll.d File Reference](#)
- 8.151 [build\\_x86\\_64/BRefData.d File Reference](#)
- 8.152 [build\\_x86\\_64/BRtc.d File Reference](#)
- 8.153 [build\\_x86\\_64/BRWLock.d File Reference](#)

**8.154** build\_x86\_64/BSema.d File Reference

**8.155** build\_x86\_64/BSemaphore.d File Reference

**8.156** build\_x86\_64/BSocket.d File Reference

**8.157** build\_x86\_64/BSpi.d File Reference

**8.158** build\_x86\_64/BString.d File Reference

**8.159** build\_x86\_64/BSys.d File Reference

**8.160** build\_x86\_64/BTable.d File Reference

**8.161** build\_x86\_64/BTask.d File Reference

**8.162** build\_x86\_64/BThread.d File Reference

**8.163** build\_x86\_64/BTime.d File Reference

**8.164** build\_x86\_64/BTimer.d File Reference

**8.165** build\_x86\_64/BTimeStamp.d File Reference

**8.166** build\_x86\_64/BTimeStampMs.d File Reference

**8.167** build\_x86\_64/BTimeUs.d File Reference

**8.168** build\_x86\_64/BUrl.d File Reference

**8.169** BUrl.cpp File Reference

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

## 8.170 BUrl.h File Reference

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

### Classes

- class [BUrl](#)

*Basic access to a Url.*

# Index

[/src/bdev/beam-lib/doc/overview.dox](#), 317

\_\_attribute\_\_

BFirmware.h, 356

BoapMc.h, 368

BoapMc1.h, 371, 372

~BBuffer

BBuffer, 27

~BBufferStore

BBufferStore, 30

~BComms

BComms, 37

~BCond

BCond, 42

~BCondBool

BCondBool, 43

~BCondInt

BCondInt, 46

~BCondResource

BCondResource, 49

~BCondValue

BCondValue, 51

~BCondWrap

BCondWrap, 54

~BDate

BDate, 60

~BDebugBacktrace

BDebugBacktrace, 65

~BDir

BDir, 76

~BDuration

BDuration, 79

~BEntryFile

BEntryFile, 85

~BEvent1

BEvent1, 98

~BEvent1Int

BEvent1Int, 101

~BEvent1Pipe

BEvent1Pipe, 102

~BEventPipe

BEventPipe, 104

~BFifo

BFifo< Type >, 107

~BFifoCirc

BFifoCirc< Type >, 114

~BFile

BFile, 123

~BList

BList< T >, 138

~BMutex

BMutex, 148

~BMutexLock

BMutexLock, 150

~BMysql

BMysql, 151

~BObj

BObj, 216

~BPoll

BPoll, 220

~BQueue

BQueue< T >, 223

~BRWLock

BRWLock, 230

~BRefData

BRefData, 225

~BRtc

BRtc, 227

~BRtcThreaded

BRtcThreaded, 228

~BSema

BSema, 232

~BSemaphore

BSemaphore, 234

~BSemaphoreBool

BSemaphoreBool, 236

~BSemaphoreCount

BSemaphoreCount, 238

~BSocket

BSocket, 242

~BSocketAddress

BSocketAddress, 248

~BString

BString, 258

~BTable

BTable, 275

~BTask

BTask, 277

~BThread

BThread, 280

~BTimeStamp

BTimeStamp, 291

~BTimeStampMs

BTimeStampMs, 301

~BTimer

BTimer, 287

~BUrl

BUrl, 313

~BoapClientObject

- BoapClientObject, 158
- ~BoapMc1Comms
  - BoapMc1Comms, 167
- ~BoapMcClientObject
  - BoapMcClientObject, 177
- ~BoapMcComms
  - BoapMcComms, 180
- ~BoapMcServiceObject
  - BoapMcServiceObject, 189
- ~BoapPacket
  - BoapPacket, 194
- ~BoapServer
  - BoapServer, 202
- ~BoapServerConnection
  - BoapServerConnection, 207
- ~BoapServiceObject
  - BoapServiceObject, 210, 211
- accept
  - BSocket, 242
- add
  - BAtomic< Type >, 23
  - BAtomicCount, 25
  - BSemaphoreCount, 238
  - BString, 259
  - BTimer, 287
- addEntry
  - Boapns::Boapns, 192
- addMicroSeconds
  - BDuration, 80
  - BTimeStamp, 291
  - BTimeUs, 309
- addMilliSeconds
  - BDuration, 80
  - BTimeStamp, 291
  - BTimeStampMs, 301
- addObject
  - BoapServer, 202, 203
- addRef
  - BRefData, 225
- address
  - BFirmware.h, 356
  - BFirmwareSegHeader, 133
  - BSocketAddressINET, 250
- addressFrom
  - BoapMc.h, 368
  - BoapMc1.h, 372
  - BoapMc1PacketHead, 175
  - BoapMcPacketHead, 187
- addressList
  - Boapns::BoapEntry, 163
- addressTo
  - BoapMc.h, 369
  - BoapMc1.h, 372
  - BoapMc1PacketHead, 175
  - BoapMcPacketHead, 187
- addRow
  - BTable, 275
- addSeconds
  - BDuration, 80
  - BTime, 283
  - BTimeStamp, 291
  - BTimeStampMs, 301
  - BTimeUs, 309
- apiVersion
  - Boapns, 17
- APIVERSION\_TEST
  - Boap.cpp, 363
- append
  - BArray< T >, 21
  - BDict< Type >, 68
  - BList< T >, 139
  - BPoll, 220
  - BString, 259
- arg
  - BEvent, 97
- average
  - BTimer, 287
- BArray
  - BArray< T >, 20
- BArray< T >, 19
  - append, 21
  - BArray, 20
  - del, 21
  - insert, 21
  - number, 21
  - rear, 21
  - sort, 22
  - SortFunc, 20
- BArray.h, 317
  - BArrayLoop, 317
- BArrayDouble
  - BTypes.h, 419
- BArrayFloat
  - BTypes.h, 419
- BArrayLoop
  - BArray.h, 317
- barrayToString
  - BString.cpp, 401
- base64\_decode
  - BObjStringFormat.h, 387
- base64\_decode\_table
  - BString.cpp, 405
- base64\_encode
  - BObjStringFormat.h, 387
- base64Decode
  - BString, 259
- base64Encode
  - BString, 259
- basename
  - BString, 259
- BAtomic
  - BAtomic< Type >, 22
- BAtomic< Type >, 22
  - add, 23
  - BAtomic, 22
  - getValue, 23



- operator Type, 23
- operator++, 23
- operator--, 23, 24
- BAtomic.h, 317
  - BAtomicInt32, 318
  - BAtomicInt64, 318
  - BAtomicUInt32, 318
  - BAtomicUInt64, 318
- BAtomicCount, 24
  - add, 25
  - BAtomicCount, 24
  - getValue, 25
  - operator long, 25
  - operator++, 25
  - operator--, 25
- BAtomicCount.h, 318
- BAtomicInt32
  - BAtomic.h, 318
- BAtomicInt64
  - BAtomic.h, 318
- BAtomicUInt32
  - BAtomic.h, 318
- BAtomicUInt64
  - BAtomic.h, 318
- BBigEndian
  - BBuffer.h, 320
- BBuffer, 26
  - ~BBuffer, 27
  - BBuffer, 27
  - data, 27
  - odata, 28
  - odataSize, 28
  - osize, 28
  - resize, 27
  - setData, 27
  - setSize, 27
  - size, 28
  - writeData, 28
- BBuffer.cpp, 319
  - roundSize, 319
- BBuffer.h, 319
  - BBigEndian, 320
- BBufferStore, 29
  - ~BBufferStore, 30
  - BBufferStore, 30
  - getHexString, 30
  - getPos, 30
  - opos, 35
  - oswapBytes, 35
  - pop, 30–32
  - push, 32–34
  - setHexString, 35
  - setPos, 35
- BChar
  - BTypes.h, 419
- BComms, 35
  - ~BComms, 37
  - BComms, 37
  - byteRate, 37
  - close, 37
  - connect, 37
  - disconnect, 37
  - eventEnable, 38
  - eventQueue, 38
  - Flush, 36
  - flush, 38
  - FlushRead, 37
  - FlushReadWrite, 37
  - FlushWrite, 37
  - init, 38
  - isConnected, 38
  - name, 38
  - oconnected, 40
  - oevent, 40
  - oeventEnabled, 40
  - oeventNum, 41
  - oeventQueue, 41
  - oeventSet, 41
  - opacketMode, 41
  - otimeout, 41
  - packetMode, 39
  - read, 39
  - readAvailable, 39
  - setPacketMode, 39
  - setTimeout, 39
  - wait, 39
  - write, 40
  - writeAvailable, 40
  - writeChunks, 40
- BComms.cpp, 320
- BComms.h, 320
- BComplex
  - BComplex.h, 320
- BComplex.h, 320
  - BComplex, 320
  - BComplex32, 321
  - BComplex64, 321
- BComplex32
  - BComplex.h, 321
- BComplex64
  - BComplex.h, 321
- BCond, 41
  - ~BCond, 42
  - BCond, 42
  - signal, 42
  - timedWait, 42
  - wait, 42
- BCond.cpp, 321
- BCond.h, 321
- BCondBool, 43
  - ~BCondBool, 43
  - BCondBool, 43
  - clear, 44
  - operator int, 44
  - set, 44
  - timedWait, 44

- value, [44](#)
- wait, [44](#)
- BCondInt, [45](#)
  - ~BCondInt, [46](#)
  - BCondInt, [46](#)
  - decrement, [46](#)
  - increment, [46](#)
  - operator++, [46](#)
  - operator+==, [46](#)
  - operator--, [47](#)
  - operator-=, [47](#)
  - setValue, [47](#)
  - value, [47](#)
  - waitLessThan, [47](#)
  - waitLessThanOrEqual, [47](#)
  - waitMoreThanOrEqual, [48](#)
- BCondInt.cpp, [321](#)
  - getTimeout, [322](#)
- BCondInt.h, [322](#)
- BCondResource, [48](#)
  - ~BCondResource, [49](#)
  - BCondResource, [49](#)
  - end, [49](#)
  - inUse, [49](#)
  - lock, [49](#)
  - locked, [49](#)
  - start, [49](#)
  - unlock, [50](#)
- BCondValue, [50](#)
  - ~BCondValue, [51](#)
  - BCondValue, [51](#)
  - decrement, [51](#)
  - increment, [51](#)
  - operator++, [51](#)
  - operator+==, [51](#)
  - operator--, [52](#)
  - operator-=, [52](#)
  - setValue, [52](#)
  - value, [52](#)
  - waitLessThan, [52](#)
  - waitLessThanOrEqual, [52](#)
  - waitMoreThanOrEqual, [53](#)
- BCondWrap, [53](#)
  - ~BCondWrap, [54](#)
  - BCondWrap, [54](#)
  - decrement, [54](#)
  - increment, [54](#)
  - operator++, [54](#)
  - operator+==, [54](#)
  - operator--, [55](#)
  - operator-=, [55](#)
  - setValue, [55](#)
  - value, [55](#)
  - waitLessThan, [55](#)
  - waitLessThanOrEqual, [55](#)
  - waitMoreThanOrEqual, [56](#)
- BConfig, [56](#)
  - close, [57](#)
  - fileName, [57](#)
  - findValue, [57](#)
  - open, [57](#)
  - read, [57](#)
  - write, [57](#)
- BConfig.cpp, [322](#)
- BConfig.h, [322](#)
- bcrc16
  - BCrc16.cpp, [323](#)
  - BCrc16.h, [324](#)
- BCrc16.cpp, [323](#)
  - bcrc16, [323](#)
  - table\_crc\_hi, [323](#)
  - table\_crc\_lo, [323](#)
- BCrc16.h, [324](#)
  - bcrc16, [324](#)
- bcrc32
  - BCrc32.cpp, [325](#)
  - BCrc32.h, [325](#)
- BCrc32.cpp, [324](#)
  - bcrc32, [325](#)
  - crc32\_tab, [325](#)
- BCrc32.h, [325](#)
  - bcrc32, [325](#)
- BDataChunk, [58](#)
  - BDataChunk, [58](#)
  - data, [58](#)
  - size, [58](#)
- BDate, [59](#)
  - ~BDate, [60](#)
  - BDate, [60](#)
  - clear, [60](#)
  - compare, [60](#)
  - day, [61](#)
  - daysInMonth, [61](#)
  - getDate, [61](#)
  - getString, [61](#)
  - getStringFormatted, [61](#)
  - isLeap, [61](#)
  - isSet, [62](#)
  - month, [62](#)
  - operator BString, [62](#)
  - operator!=, [62](#)
  - operator<, [62](#)
  - operator<=, [62](#)
  - operator>, [63](#)
  - operator>=, [63](#)
  - operator==, [62](#)
  - oyday, [64](#)
  - oyear, [65](#)
  - set, [63](#)
  - setFirst, [63](#)
  - setLast, [63](#)
  - setNow, [64](#)
  - setString, [64](#)
  - setYDay, [64](#)
  - yday, [64](#)
  - year, [64](#)

- BDate-1.cpp, 326
  - fromBString, 326
  - mon\_yday, 326
  - toBString, 326
- BDate.cpp, 327
  - fromBString, 327
  - mon\_yday, 327
  - toBString, 327
- BDate.h, 328
  - fromBString, 328
  - toBString, 328
- bdebug
  - BDebug.cpp, 330
  - BDebug.h, 334
- BDebug.cpp, 329
  - bdebug, 330
  - bhd32, 329
  - bhd8, 329
  - bhd8a, 329
  - bhda32, 330
  - bhda8, 330
  - getTime, 330
  - setDebug, 330
- BDebug.h, 331
  - bdebug, 334
  - BDebug\_STD, 331
  - bgettid, 333
  - bhd32, 333
  - bhd8, 333
  - bhd8a, 333
  - bhda8, 334
  - bhds32, 334
  - dl1printf, 332
  - dl2printf, 332
  - dl3printf, 332
  - dl4printf, 332
  - dprintf, 332
  - eprintf, 332
  - getTime, 334
  - nprintf, 333
  - setDebug, 334
  - tprintf, 334
  - wprintf, 333
- BDebug\_STD
  - BDebug.h, 331
- BDebugBacktrace, 65
  - ~BDebugBacktrace, 65
  - BDebugBacktrace, 65
  - dumpBacktrace, 66
  - dumpBacktraceFile, 66
  - dumpBacktraceStdout, 66
  - dumpBacktraceSyslog, 66
- BDEBUGL1
  - BoapMc1.cpp, 370
- BDEBUGL2
  - BoapMc1.cpp, 370
- BDict
  - BDict< Type >, 67, 68
- BDict< Type >, 66
  - append, 68
  - BDict, 67, 68
  - clear, 68
  - del, 68, 69
  - find, 69
  - hashPrint, 69
  - hasKey, 69
  - insert, 69
  - iterator, 67
  - key, 69
  - operator+, 70
  - operator=, 70
  - operator[], 70, 71
- BDict.cpp, 335
  - bdictStringToString, 335
  - fromBString, 335
  - toBString, 335
- BDict.h, 335
  - BDictString, 336
  - bdictStringToString, 336
  - fromBString, 336
  - toBString, 336
- BDictItem
  - BDictItem< Type >, 72
- BDictItem< Type >, 71
  - BDictItem, 72
  - key, 72
  - value, 72
- BDictMap< Value >, 72
  - clear, 73
  - del, 73
  - hasKey, 74
  - isEnd, 74
  - iterator, 73
  - key, 74
  - next, 74
  - operator[], 74
  - size, 75
  - start, 75
- BDictMap.h, 337
  - BDictMapString, 337
- BDictMapString
  - BDictMap.h, 337
- BDictString
  - BDict.h, 336
- bdictStringToString
  - BDict.cpp, 335
  - BDict.h, 336
- BDir, 75
  - ~BDir, 76
  - BDir, 76
  - clear, 77
  - entryName, 77
  - entryStat, 77
  - entryStat64, 77
  - error, 77
  - open, 78

- read, 78
- setSort, 78
- setWild, 78
- BDir.cpp, 337
  - wild, 338
  - wildString, 338
- BDir.h, 338
- BDouble
  - BTypes.h, 420
- BDuration, 78
  - ~BDuration, 79
  - addMicroSeconds, 80
  - addMilliSeconds, 80
  - addSeconds, 80
  - BDuration, 79
  - clear, 80
  - getMicroSeconds, 80
  - getSeconds, 80
  - getString, 81
  - hour, 81
  - microSecond, 81
  - minute, 81
  - second, 81
  - set, 81
  - setString, 81
- BDuration.cpp, 338
- BDuration.h, 338
- be16toh
  - BEndian.h, 340
- be32toh
  - BEndian.h, 340
- be64toh
  - BEndian.h, 341
- begin
  - BList< T >, 139
- BEndian.cpp, 339
  - bswap\_copy, 339
- BEndian.h, 339
  - be16toh, 340
  - be32toh, 340
  - be64toh, 341
  - betoh, 342, 343
  - bswap\_copy, 343
  - bswap\_p16, 343
  - bswap\_p32, 344
  - bswap\_p64, 344
  - bswap\_p8, 344
  - htobe, 344, 345
  - htobe16, 341
  - htobe32, 341
  - htobe64, 341
  - htole, 345, 346
  - htole16, 341
  - htole32, 341
  - htole64, 341
  - le16toh, 342
  - le32toh, 342
  - le64toh, 342
  - letoh, 346, 347
- BEntry, 82
  - BEntry, 82, 83
  - getName, 83
  - getValue, 83
  - line, 83
  - print, 83
  - setLine, 84
  - setName, 84
  - setValue, 84
- BEntry.cpp, 348
- BEntry.h, 348
- BEntryFile, 84
  - ~BEntryFile, 85
  - BEntryFile, 85
  - clear, 86
  - filename, 86
  - open, 86
  - read, 86
  - write, 86
  - writeList, 86
- BEntryList, 87
  - BEntryList, 88
  - clear, 88
  - del, 88
  - deleteEntry, 88
  - find, 88
  - findValue, 89
  - getString, 89
  - insert, 89
  - isSet, 89
  - operator=, 89
  - print, 90
  - setValue, 90
  - setValueRaw, 90
- BError, 90
  - BError, 91
  - clear, 92
  - copy, 92
  - getErrorNo, 92
  - getNumber, 92
  - getString, 92
  - num, 92
  - operator int, 93
  - set, 93
  - setError, 93
  - str, 93
- BError.cpp, 348
- BError.h, 348
  - BErrorNum, 349
  - ErrorAccessDenied, 349
  - ErrorApiVersion, 349
  - ErrorAppBase, 349
  - ErrorChecksum, 349
  - ErrorComms, 349
  - ErrorConfig, 349
  - ErrorData, 349
  - ErrorDataPresent, 349

- ErrorDataTruncated, 349
- ErrorEndOfData, 349
- ErrorEndOfFile, 349
- ErrorFile, 349
- ErrorFormat, 349
- ErrorInit, 349
- ErrorMisc, 349
- ErrorNoData, 349
- ErrorNotAvailable, 349
- ErrorNotImplemented, 349
- ErrorOk, 349
- ErrorOverrun, 349
- ErrorParam, 349
- ErrorResourceLimit, 349
- ErrorTimeout, 349
- ErrorUnderrun, 349
- ErrorUserBase, 349
- ErrorWarning, 349
- BErrorNum
  - BError.h, 349
- BErrorTime, 94
  - BErrorTime, 95
  - clear, 95
  - copy, 95
  - Error, 94
  - getErrorNo, 95
  - getString, 95
  - getTime, 95
  - None, 94
  - operator int, 96
  - set, 96
  - Type, 94
- BErrorTime.cpp, 350
- BErrorTime.h, 350
- betoh
  - BEndian.h, 342, 343
- BEvent, 96
  - arg, 97
  - BEvent, 96
  - type, 97
- BEvent.cpp, 350
- BEvent.h, 350
  - BEventQueue, 351
- BEvent1, 97
  - ~BEvent1, 98
  - BEvent1, 98
  - getBinary, 98
  - getType, 98
  - setBinary, 98
- BEvent1.cpp, 351
- BEvent1.h, 351
  - BEvent1 Type, 351
  - BEvent1 TypeError, 352
  - BEvent1 TypeInt, 352
  - BEvent1 TypeNone, 352
- BEvent1Error, 99
  - BEvent1Error, 99
  - getBinary, 99
  - setBinary, 100
- BEvent1Int, 100
  - ~BEvent1Int, 101
  - BEvent1Int, 101
  - clear, 101
  - getEvent, 101
  - getFd, 101
  - sendEvent, 101
- BEvent1Pipe, 102
  - ~BEvent1Pipe, 102
  - BEvent1Pipe, 102
  - clear, 103
  - getEvent, 103
  - getReceiveFd, 103
  - sendEvent, 103
- BEvent1 Type
  - BEvent1.h, 351
- BEvent1 TypeError
  - BEvent1.h, 352
- BEvent1 TypeInt
  - BEvent1.h, 352
- BEvent1 TypeNone
  - BEvent1.h, 352
- BEventPipe, 104
  - ~BEventPipe, 104
  - BEventPipe, 104
  - clear, 104
  - getFd, 105
  - read, 105
  - readAvailable, 105
  - write, 105
  - writeAvailable, 105
- BEventQueue
  - BEvent.h, 351
- BEventType
  - BTypes.h, 422
- BEventTypeClientConnect
  - BTypes.h, 422
- BEventTypeClientDisconnect
  - BTypes.h, 422
- BEventTypeConnect
  - BTypes.h, 422
- BEventTypeDisconnect
  - BTypes.h, 422
- BEventTypeError
  - BTypes.h, 422
- BEventTypeNone
  - BTypes.h, 422
- BEventTypeRead
  - BTypes.h, 422
- BEventTypeReadLine
  - BTypes.h, 422
- BEventTypeWrite
  - BTypes.h, 422
- BEventWaitAny
  - BTypes.h, 422
- BEventWaitClientConnect
  - BTypes.h, 422

- BEventWaitClientDisconnect
  - BTypes.h, 422
- BEventWaitConnect
  - BTypes.h, 422
- BEventWaitDisconnect
  - BTypes.h, 422
- BEventWaitError
  - BTypes.h, 422
- BEventWaitNone
  - BTypes.h, 422
- BEventWaitRead
  - BTypes.h, 422
- BEventWaitReadLine
  - BTypes.h, 422
- BEventWaitSet
  - BTypes.h, 422
- BEventWaitWrite
  - BTypes.h, 422
- BFifo
  - BFifo< Type >, 107
- BFifo< Type >, 106
  - ~BFifo, 107
  - BFifo, 107
  - clear, 107
  - odata, 112
  - operator[], 107
  - oreadPos, 112
  - osize, 112
  - owritePos, 112
  - read, 107, 108
  - readAvailable, 108
  - readAvailableChunk, 108
  - readData, 108
  - readDone, 109
  - readPos, 109
  - rebase, 109
  - resize, 109
  - size, 109
  - write, 110
  - writeAvailable, 110
  - writeAvailableChunk, 110
  - writeBackup, 110
  - writeData, 111
  - writeDone, 111
  - writePos, 111
- BFifo.h, 352
- BFifo.inc, 352
- BFifoCirc
  - BFifoCirc< Type >, 114
- BFifoCirc< Type >, 112
  - ~BFifoCirc, 114
  - BFifoCirc, 114
  - clear, 114
  - defaultSize, 114
  - mapCircularBuffer, 115
  - odata, 117
  - oclock, 117
  - operator[], 115
  - oreadPos, 118
  - osize, 118
  - ovmSize, 118
  - owriteNumFifoSamples, 118
  - owritePos, 118
  - read, 115
  - readAvailable, 115
  - readData, 115
  - readDone, 115
  - readWaitAvailable, 116
  - size, 116
  - unmapCircularBuffer, 116
  - write, 116
  - writeAvailable, 116
  - writeData, 117
  - writeDone, 117
  - writeWaitAvailable, 117
- BFifoCirc.cpp, 352
  - dprintf, 352
- BFifoCirc.h, 353
- BFifoCirc.inc, 353
- BFifoCircPos, 119
  - BFifoCircPos, 119
  - difference, 119
  - increment, 119
  - operator int, 120
  - operator!=, 120
  - operator+==, 120
  - operator==, 120
  - pos, 120
  - set, 120
  - setSize, 121
- BFile, 121
  - ~BFile, 123
  - BFile, 122
  - close, 123
  - fgets, 123
  - fileName, 123
  - flush, 123
  - getFd, 123
  - isEnd, 124
  - isOpen, 124
  - length, 124
  - open, 124
  - operator=, 125
  - position, 125
  - printf, 125
  - read, 125
  - readString, 125
  - seek, 125
  - setVBuf, 126
  - truncate, 126
  - write, 126
  - writeString, 126
- BFile.cpp, 353
  - STRBUF, 353
- BFile.h, 353
- BFileCsv, 127

- BFileCsv, [127](#)
- readCsv, [127](#)
- writeCsv, [127](#)
- BFileCsv.cpp, [354](#)
- BFileCsv.h, [354](#)
- BFileData, [128](#)
- del, [128](#)
- find, [128](#)
- getNextId, [128](#)
- open, [128](#)
- write, [129](#)
- BFileData.cpp, [354](#)
- BFileData.h, [354](#)
- BFirmware.h, [354](#)
- \_\_attribute\_\_, [356](#)
- address, [356](#)
- bfirmwareBoot, [356](#)
- BFirmwareFirmwareHeader, [356](#)
- BFirmwareFormatGzip, [357](#)
- BFirmwareFormatRaw, [357](#)
- BFirmwareInfoEncrypt1, [357](#)
- BFirmwareInfoMagic, [357](#)
- BFirmwareMagic, [357](#)
- BFirmwarePlatformBMeasure125, [357](#)
- BFirmwarePlatformBMeasure125Boot, [357](#)
- BFirmwarePlatformBMeasure125Cpu, [357](#)
- BFirmwarePlatformBMeasure125Fpga, [358](#)
- BFirmwarePlatformBMeasure125Wifi, [358](#)
- BFirmwareTypeFile, [358](#)
- BFirmwareTypeFirmware, [358](#)
- BFirmwareTypeSegment, [358](#)
- bfirmwareValid, [356](#)
- checksum, [358](#)
- dataLength, [358](#)
- fileLength, [358](#)
- format, [359](#)
- itemType, [359](#)
- length, [359](#)
- magic, [359](#)
- numSegments, [359](#)
- platform, [359](#)
- special, [359](#)
- startAddress, [359](#)
- ver0, [360](#)
- ver1, [360](#)
- ver2, [360](#)
- ver3, [360](#)
- bfirmwareBoot
- BFirmware.h, [356](#)
- BFirmwareFileHeader, [129](#)
- checksum, [129](#)
- fileLength, [129](#)
- format, [130](#)
- itemType, [130](#)
- magic, [130](#)
- numSegments, [130](#)
- platform, [130](#)
- special, [130](#)
- startAddress, [130](#)
- ver0, [130](#)
- ver1, [131](#)
- ver2, [131](#)
- ver3, [131](#)
- BFirmwareFirmwareHeader
- BFirmware.h, [356](#)
- BFirmwareFormatGzip
- BFirmware.h, [357](#)
- BFirmwareFormatRaw
- BFirmware.h, [357](#)
- BFirmwareInfo, [131](#)
- checksum, [131](#)
- length, [132](#)
- magic, [132](#)
- type, [132](#)
- ver0, [132](#)
- ver1, [132](#)
- ver2, [132](#)
- BFirmwareInfoEncrypt1
- BFirmware.h, [357](#)
- BFirmwareInfoMagic
- BFirmware.h, [357](#)
- BFirmwareMagic
- BFirmware.h, [357](#)
- BFirmwarePlatformBMeasure125
- BFirmware.h, [357](#)
- BFirmwarePlatformBMeasure125Boot
- BFirmware.h, [357](#)
- BFirmwarePlatformBMeasure125Cpu
- BFirmware.h, [357](#)
- BFirmwarePlatformBMeasure125Fpga
- BFirmware.h, [358](#)
- BFirmwarePlatformBMeasure125Wifi
- BFirmware.h, [358](#)
- BFirmwareSegHeader, [133](#)
- address, [133](#)
- checksum, [133](#)
- dataLength, [133](#)
- fileLength, [133](#)
- format, [133](#)
- itemType, [134](#)
- length, [134](#)
- magic, [134](#)
- platform, [134](#)
- special, [134](#)
- BFirmwareTypeFile
- BFirmware.h, [358](#)
- BFirmwareTypeFirmware
- BFirmware.h, [358](#)
- BFirmwareTypeSegment
- BFirmware.h, [358](#)
- bfirmwareValid
- BFirmware.h, [356](#)
- BFloat
- BTypes.h, [420](#)
- BFloat32
- BTypes.h, [420](#)

- BFloat64
  - BTypes.h, 420
- bgettid
  - BDebug.h, 333
- bhd32
  - BDebug.cpp, 329
  - BDebug.h, 333
- bhd8
  - BDebug.cpp, 329
  - BDebug.h, 333
- bhd8a
  - BDebug.cpp, 329
  - BDebug.h, 333
- bhda32
  - BDebug.cpp, 330
- bhda8
  - BDebug.cpp, 330
  - BDebug.h, 334
- bhds32
  - BDebug.h, 334
- bind
  - BSocket, 242
- BInt
  - BTypes.h, 420
- BInt16
  - BTypes.h, 420
- BInt32
  - BTypes.h, 420
- BInt64
  - BTypes.h, 420
- BInt8
  - BTypes.h, 421
- Blter, 134
  - Blter, 135
  - operator BNode \*, 135
  - operator==, 135
  - valid, 135
- BList
  - BList< T >, 138
- BList< T >, 136
  - ~BList, 138
  - append, 139
  - begin, 139
  - BList, 138
  - clear, 139
  - del, 139
  - deleteFirst, 140
  - deleteLast, 140
  - end, 140
  - front, 140
  - get, 141
  - goTo, 141
  - has, 141
  - insert, 141
  - insertAfter, 142
  - isEnd, 142
  - isStart, 142
  - next, 142
  - nodeCreate, 142
  - nodeGet, 143
  - number, 143
  - olength, 146
  - onodes, 147
  - operator+, 143
  - operator=, 143
  - operator[], 143, 144
  - pop, 144
  - position, 144
  - prev, 144
  - push, 145
  - queueAdd, 145
  - queueGet, 145
  - rear, 145
  - size, 145
  - sort, 146
  - SortFunc, 138
  - start, 146
  - swap, 146
- BList< T >::Node, 314
  - item, 315
  - Node, 314
- BList.h, 360
  - BListLoop, 361
- BList\_func.h, 361
- BListLoop
  - BList.h, 361
- blstToString
  - BString.cpp, 401
- BMutex, 147
  - ~BMutex, 148
  - BMutex, 148
  - lock, 148
  - Normal, 148
  - operator=, 149
  - Recursive, 148
  - timedLock, 149
  - tryLock, 149
  - Type, 148
  - unlock, 149
- BMutex.cpp, 361
  - MDEBUG, 361
- BMutex.h, 361
- BMutexLock, 149
  - ~BMutexLock, 150
  - BMutexLock, 150
  - lock, 150
  - unlock, 150
- BMysql, 150
  - ~BMysql, 151
  - BMysql, 151
  - close, 151
  - db, 151
  - del, 151
  - escapeString, 152
  - flush, 152
  - get, 152



- insert, [152](#)
- open, [152](#)
- query, [153](#)
- setDebug, [153](#)
- update, [153](#)
- BMySQL.cpp, [362](#)
- BMySQL.h, [362](#)
- BNameValue
  - BNameValue< T >, [154](#)
- BNameValue< T >, [153](#)
  - BNameValue, [154](#)
  - getName, [154](#)
  - getValue, [154](#)
- BNameValue.h, [362](#)
- BNameValueList< T >, [154](#)
  - find, [155](#)
  - findPos, [155](#)
- BNode, [155](#)
  - BNode, [156](#)
  - next, [156](#)
  - prev, [156](#)
- Boap.cpp, [362](#)
  - APIVERSION\_TEST, [363](#)
  - boapPort, [363](#)
  - DEBUG, [363](#)
  - dprintf, [363](#)
  - IS\_BIG\_ENDIAN, [363](#)
- Boap.d, [364](#)
- Boap.h, [364](#)
  - BoapFunc, [365](#)
  - BoapMagic, [366](#)
  - BoapPriority, [365](#)
  - BoapPriorityHigh, [365](#)
  - BoapPriorityLow, [365](#)
  - BoapPriorityNormal, [365](#)
  - BoapService, [365](#)
  - BoapType, [365](#)
  - BoapTypeRpc, [366](#)
  - BoapTypeRpcError, [366](#)
  - BoapTypeRpcReply, [366](#)
  - BoapTypeSignal, [366](#)
- BoapClientObject, [156](#)
  - ~BoapClientObject, [158](#)
- BoapClientObject, [158](#)
- checkApiVersion, [158](#)
- connectService, [158](#)
- disconnectService, [159](#)
- getServiceName, [159](#)
- handleReconnect, [159](#)
- oapiVersion, [161](#)
- oconnected, [161](#)
- oclock, [161](#)
- omaxLength, [161](#)
- oname, [162](#)
- opriority, [162](#)
- oreconnect, [162](#)
- orx, [162](#)
- oservice, [162](#)
- otimeout, [162](#)
- otx, [162](#)
- performCall, [159](#)
- performRecv, [159](#), [160](#)
- performSend, [160](#)
- ping, [160](#)
- pingLocked, [160](#)
- setConnectionPriority, [160](#)
- setMaxLength, [161](#)
- setTimeout, [161](#)
- BoapEntry
  - Boapns::BoapEntry, [163](#)
- BoapFunc
  - Boap.h, [365](#)
  - BoapSimple.h, [377](#)
- BoapFuncEntry, [164](#)
  - BoapFuncEntry, [165](#)
  - ocmd, [165](#)
  - ofunc, [165](#)
- BoapMagic
  - Boap.h, [366](#)
- BoapMc.cpp, [366](#)
  - DEBUG\_LOCAL, [366](#)
  - DEBUG\_LOCAL1, [366](#)
  - dl1printf, [367](#)
  - dlprintf, [367](#)
- BoapMc.h, [367](#)
  - \_\_attribute\_\_, [368](#)
  - addressFrom, [368](#)
  - addressTo, [369](#)
  - BoapMcType, [368](#)
  - BoapMcTypeReply, [368](#)
  - BoapMcTypeRequest, [368](#)
  - checksum, [369](#)
  - cmd, [369](#)
  - error, [369](#)
  - length, [369](#)
- BoapMc1.cpp, [369](#)
  - BDEBUGL1, [370](#)
  - BDEBUGL2, [370](#)
- BoapMc1.h, [370](#)
  - \_\_attribute\_\_, [371](#), [372](#)
  - addressFrom, [372](#)
  - addressTo, [372](#)
  - boapMc1CommsRoundupLen, [371](#)
  - BoapMc1Magic, [372](#)
  - BoapMc1Type, [371](#)
  - BoapMc1TypeReply, [371](#)
  - BoapMc1TypeRequest, [371](#)
  - checksum, [372](#)
  - cmd, [372](#)
  - data, [373](#)
  - error, [373](#)
  - head, [373](#)
  - length, [373](#)
  - magic, [373](#)
  - number, [373](#)
  - string, [374](#)

- BoapMc1Comms, 166
  - ~BoapMc1Comms, 167
  - BoapMc1Comms, 167
  - getApiVersion, 167
  - oaddressFrom, 170
  - oaddressTo, 170
  - oapiVersion, 170
  - ocomms, 170
  - oerror, 170
  - ohalfDuplex, 170
  - oclockCall, 171
  - oclockTx, 171
  - opacketRpcCmd, 171
  - opacketRpcDoneSema, 171
  - opacketRpcSema, 171
  - opacketRx, 171
  - opacketRxBase, 172
  - opacketTx, 172
  - opacketTxBase, 172
  - oreqSize, 172
  - othreaded, 172
  - otimeout, 172
  - packetRx, 167
  - packetRxData, 168
  - packetRxEnd, 168
  - packetTx, 168
  - processRequest, 168
  - processRequests, 168
  - processRx, 168
  - setAddress, 169
  - setComms, 169
  - setCommsMode, 169
  - setTimeout, 169
  - validate, 169
- boapMc1CommsRoundupLen
  - BoapMc1.h, 371
- BoapMc1Error, 173
  - number, 173
  - string, 173
- BoapMc1Magic
  - BoapMc1.h, 372
- BoapMc1Packet, 174
  - data, 174
  - head, 174
- BoapMc1PacketHead, 174
  - addressFrom, 175
  - addressTo, 175
  - checksum, 175
  - cmd, 175
  - error, 175
  - length, 175
  - magic, 176
- BoapMc1Type
  - BoapMc1.h, 371
- BoapMc1TypeReply
  - BoapMc1.h, 371
- BoapMc1TypeRequest
  - BoapMc1.h, 371
- BoapMcClientObject, 176
  - ~BoapMcClientObject, 177
  - BoapMcClientObject, 177
  - getApiVersion, 177
  - oaddressFrom, 178
  - oaddressTo, 178
  - oapiVersion, 178
  - ocomms, 178
  - opacket, 178
  - performCall, 177
  - performRecv, 177
  - performSend, 177
  - setAddress, 178
- BoapMcComms, 179
  - ~BoapMcComms, 180
  - BoapMcComms, 180
  - getApiVersion, 181
  - oaddressFrom, 183
  - oaddressTo, 183
  - oapiVersion, 184
  - ocomms, 184
  - oclockCall, 184
  - oclockTx, 184
  - opacket, 184
  - opacketReqQueue, 184
  - opacketReqRx, 184
  - opacketReqTx, 185
  - opacketRx, 185
  - opacketRxSema, 185
  - opacketTx, 185
  - opacketTxQueue, 185
  - opacketTxQueueWriteNum, 185
  - opacketTxSema, 186
  - oslave, 186
  - othreaded, 186
  - otimeout, 186
  - packetRecv, 181
  - packetSend, 181
  - performCall, 181
  - performSend, 181
  - processPacket, 181
  - processRequest, 182
  - processRequests, 182
  - processRx, 182
  - setAddress, 182
  - setComms, 182, 183
  - setCommsMode, 183
  - setTimeout, 183
- BoapMcPacket, 186
  - data, 187
  - head, 187
- BoapMcPacketHead, 187
  - addressFrom, 187
  - addressTo, 187
  - checksum, 188
  - cmd, 188
  - error, 188
  - length, 188

- BoapMcServiceObject, 188
  - ~BoapMcServiceObject, 189
  - BoapMcServiceObject, 189
  - oapiVersion, 190
  - process, 189
  - processEvent, 189
  - sendEvent, 189
- BoapMcSignalObject, 190
  - BoapMcSignalObject, 190
  - ocomms, 191
  - performSend, 190
- BoapMcType
  - BoapMc.h, 368
- BoapMcTypeReply
  - BoapMc.h, 368
- BoapMcTypeRequest
  - BoapMc.h, 368
- Boapns, 17
  - apiVersion, 17
  - Boapns::Boapns, 191
- Boapns::BoapEntry, 163
  - addressList, 163
  - BoapEntry, 163
  - hostName, 164
  - name, 164
  - port, 164
  - service, 164
- Boapns::Boapns, 191
  - addEntry, 192
  - Boapns, 191
  - delEntry, 192
  - getEntry, 192
  - getEntryList, 192
  - getNewName, 192
  - getVersion, 192
- BoapnsC.cpp, 374
- BoapnsC.h, 374
- BoapnsD.cpp, 375
- BoapnsD.h, 375
- BoapPacket, 193
  - ~BoapPacket, 194
  - BoapPacket, 194
  - data, 194
  - getCmd, 195
  - nbytes, 195
  - peekHead, 195
  - pop, 195, 196
  - popHead, 196, 197
  - push, 197, 198
  - pushHead, 198
  - resize, 198
  - setData, 199
  - updateHead, 199
- BoapPacketHead, 199
  - cmd, 199, 200
  - length, 200
  - reserved, 200
  - service, 200
  - type, 200
- boapPort
  - Boap.cpp, 363
- BoapPriority
  - Boap.h, 365
- BoapPriorityHigh
  - Boap.h, 365
- BoapPriorityLow
  - Boap.h, 365
- BoapPriorityNormal
  - Boap.h, 365
- BoapServer, 201
  - ~BoapServer, 202
  - addObject, 202, 203
  - BoapServer, 202
  - clientGone, 203
  - closeConnections, 203
  - getConnectionsNumber, 203
  - getEventSocket, 203
  - getHostName, 203, 204
  - getSocket, 204
  - init, 204
  - newConnection, 204
  - NOTHEADS, 202
  - onumOperations, 206
  - process, 204, 205
  - processEvent, 205
  - run, 205
  - sendEvent, 206
  - THREADED, 202
- BoapServerConnection, 206
  - ~BoapServerConnection, 207
  - BoapServerConnection, 207
  - getHead, 207
  - getSocket, 207
  - init, 207
  - process, 208
  - setMaxLength, 208
  - validate, 208
- BoapService
  - Boap.h, 365
  - BoapSimple.h, 377
- BoapServiceEntry, 208
  - BoapServiceEntry, 209
  - oobject, 209
  - oservice, 209
- BoapServiceObject, 209
  - ~BoapServiceObject, 210, 211
  - BoapServiceObject, 210
  - doConnectionPriority, 211
  - doPing, 211
  - name, 211
  - oapiVersion, 213
  - ofuncList, 213
  - oname, 214
  - oserver, 214
  - process, 211, 212
  - processEvent, 212

- sendEvent, [212](#), [213](#)
- setName, [213](#)
- BoapSignalObject, [214](#)
  - BoapSignalObject, [215](#)
  - orx, [215](#)
  - otx, [216](#)
  - performSend, [215](#)
- BoapSimple.cc, [375](#)
  - DEBUG, [376](#)
  - dprintf, [376](#)
  - roundSize, [376](#)
- BoapSimple.h, [376](#)
  - BoapFunc, [377](#)
  - BoapService, [377](#)
  - BoapType, [378](#)
  - BoapTypeRpc, [378](#)
  - BoapTypeRpcError, [378](#)
  - BoapTypeRpcReply, [378](#)
  - BoapTypeSignal, [378](#)
  - Double, [377](#)
  - Int16, [377](#)
  - Int32, [377](#)
  - Int8, [378](#)
  - UInt16, [378](#)
  - UInt32, [378](#)
  - UInt8, [378](#)
- BoapType
  - Boap.h, [365](#)
  - BoapSimple.h, [378](#)
- BoapTypeRpc
  - Boap.h, [366](#)
  - BoapSimple.h, [378](#)
- BoapTypeRpcError
  - Boap.h, [366](#)
  - BoapSimple.h, [378](#)
- BoapTypeRpcReply
  - Boap.h, [366](#)
  - BoapSimple.h, [378](#)
- BoapTypeSignal
  - Boap.h, [366](#)
  - BoapSimple.h, [378](#)
- BObj, [216](#)
  - ~BObj, [216](#)
  - BObj, [216](#)
  - getDebugString, [217](#)
  - getMember, [217](#)
  - getMembers, [217](#)
  - getType, [217](#)
  - membersPrint, [217](#)
  - setMember, [218](#)
  - setMembers, [218](#)
- BObj.cpp, [379](#)
- BObj.h, [379](#)
- BObjMember, [218](#)
  - dataOffset, [218](#)
  - name, [218](#)
  - size, [219](#)
  - type, [219](#)
- typeComp, [219](#)
- typeName, [219](#)
- BObjStringFormat.cpp, [379](#)
  - toBDictStringFromJson, [380](#)
  - toBString, [380–383](#)
  - toBStringJson, [383–386](#)
- BObjStringFormat.h, [386](#)
  - base64\_decode, [387](#)
  - base64\_encode, [387](#)
  - toBDictStringFromJson, [387](#)
  - toBString, [387–390](#)
  - toBStringJson, [390–393](#)
- Bool
  - BTypes.h, [421](#)
- BPoll, [219](#)
  - ~BPoll, [220](#)
  - append, [220](#)
  - BPoll, [220](#)
  - clear, [221](#)
  - delFd, [221](#)
  - doPoll, [221](#)
  - doPollEvents, [221](#)
  - getPollFds, [221](#)
  - getPollFdsNum, [221](#)
  - PollFd, [220](#)
- BPoll.cpp, [394](#)
- BPoll.h, [394](#)
- BQueue
  - BQueue< T >, [222](#)
- BQueue< T >, [222](#)
  - ~BQueue, [223](#)
  - BQueue, [222](#)
  - clear, [223](#)
  - read, [223](#)
  - readAvailable, [223](#)
  - write, [223](#)
  - writeAvailable, [224](#)
- BQueue.h, [394](#)
  - BQueueInt, [394](#)
- BQueueInt
  - BQueue.h, [394](#)
- BRefData, [224](#)
  - ~BRefData, [225](#)
  - addRef, [225](#)
  - BRefData, [225](#)
  - copy, [225](#)
  - data, [225](#)
  - deleteRef, [226](#)
  - len, [226](#)
  - operator=, [226](#)
  - setLen, [226](#)
- BRefData.cpp, [395](#)
- BRefData.h, [395](#)
- BRtc, [226](#)
  - ~BRtc, [227](#)
  - BRtc, [227](#)
  - init, [227](#)

- wait, [227](#)
- BRtc.cpp, [395](#)
- BRtc.h, [396](#)
- BRtcThreaded, [228](#)
  - ~BRtcThreaded, [228](#)
  - BRtcThreaded, [228](#)
  - init, [229](#)
  - wait, [229](#)
- BRWLock, [229](#)
  - ~BRWLock, [230](#)
  - BRWLock, [230](#)
  - operator=, [230](#)
  - rdLock, [230](#)
  - tryRdLock, [230](#)
  - tryWrLock, [231](#)
  - unlock, [231](#)
  - wrLock, [231](#)
- BRWLock.cpp, [396](#)
- BRWLock.h, [396](#)
- BSema, [231](#)
  - ~BSema, [232](#)
  - BSema, [232](#)
  - getValue, [232](#)
  - operator=, [233](#)
  - post, [233](#)
  - timedWait, [233](#)
  - tryWait, [233](#)
  - wait, [233](#)
- BSema.cpp, [396](#)
- BSema.h, [396](#)
- BSemaphore, [234](#)
  - ~BSemaphore, [234](#)
  - BSemaphore, [234](#)
  - getValue, [235](#)
  - operator=, [235](#)
  - set, [235](#)
  - wait, [235](#)
- BSemaphore.cpp, [397](#)
- BSemaphore.h, [397](#)
- BSemaphoreBool, [235](#)
  - ~BSemaphoreBool, [236](#)
  - BSemaphoreBool, [236](#)
  - clear, [236](#)
  - operator int, [236](#)
  - operator=, [237](#)
  - operator==, [237](#)
  - set, [237](#)
  - value, [237](#)
  - wait, [237](#)
- BSemaphoreCount, [238](#)
  - ~BSemaphoreCount, [238](#)
  - add, [238](#)
  - BSemaphoreCount, [238](#)
  - operator=, [239](#)
  - setValue, [239](#)
  - take, [239](#)
  - value, [239](#)
  - wait, [239](#)
- BSize
  - BTypes.h, [421](#)
- BSocket, [240](#)
  - ~BSocket, [242](#)
  - accept, [242](#)
  - bind, [242](#)
  - BSocket, [241](#), [242](#)
  - close, [242](#)
  - connect, [243](#)
  - DGRAM, [241](#)
  - getAddress, [243](#)
  - getFd, [243](#)
  - getMTU, [243](#)
  - getSockOpt, [243](#)
  - init, [243](#)
  - listen, [244](#)
  - NType, [241](#)
  - Priority, [241](#)
  - PriorityHigh, [241](#)
  - PriorityLow, [241](#)
  - PriorityNormal, [241](#)
  - recv, [244](#)
  - recvAvailable, [244](#)
  - recvFrom, [244](#)
  - recvFromWithTimeout, [244](#)
  - recvWithTimeout, [244](#)
  - send, [245](#)
  - sendChunks, [245](#)
  - sendTo, [245](#)
  - setBroadCast, [245](#)
  - setFd, [245](#)
  - setPriority, [246](#)
  - setReuseAddress, [246](#)
  - setSockOpt, [246](#)
  - shutdown, [246](#)
  - STREAM, [241](#)
- BSocket.cpp, [397](#)
  - IP\_MTU, [398](#)
- BSocket.h, [398](#)
  - MSG\_NOSIGNAL, [398](#)
  - SO\_PRIORITY, [398](#)
  - SOL\_IP, [399](#)
- BSocketAddress, [246](#)
  - ~BSocketAddress, [248](#)
  - BSocketAddress, [247](#), [248](#)
  - getString, [248](#)
  - len, [248](#)
  - operator const SockAddr \*, [248](#)
  - operator!=, [248](#)
  - operator=, [249](#)
  - operator==, [249](#)
  - raw, [249](#)
  - set, [249](#)
  - SockAddr, [247](#)
- BSocketAddressINET, [249](#)
  - address, [250](#)
  - getHostName, [251](#)
  - getIpAddresses, [251](#)

- getIpAddressList, [251](#)
- getIpAddressListAll, [251](#)
- getString, [251](#)
- port, [251](#)
- set, [252](#)
- setPort, [252](#)
- SocketAddrIP, [250](#)
- BSpi, [252](#)
  - BSpi, [253](#)
  - init, [253](#)
  - Mode, [253](#)
  - Mode0, [253](#)
  - Mode1, [253](#)
  - Mode2, [253](#)
  - Mode3, [253](#)
  - transact, [254](#)
- BSpi.cpp, [399](#)
- BSpi.h, [399](#)
- BString, [254](#)
  - ~BString, [258](#)
  - add, [259](#)
  - append, [259](#)
  - base64Decode, [259](#)
  - base64Encode, [259](#)
  - basename, [259](#)
  - BString, [257](#), [258](#)
  - clear, [259](#)
  - compare, [260](#)
  - compareRegex, [260](#)
  - compareWild, [260](#)
  - compareWildExpression, [260](#)
  - convert, [260](#), [261](#)
  - convertHex, [261](#)
  - copy, [262](#)
  - csvDecode, [262](#)
  - csvEncode, [262](#)
  - del, [262](#)
  - dirname, [262](#)
  - extension, [262](#)
  - field, [263](#)
  - fields, [263](#)
  - find, [263](#)
  - findReverse, [263](#)
  - firstLine, [263](#)
  - fixedLen, [264](#)
  - get, [264](#)
  - getTokenList, [264](#)
  - hash, [264](#)
  - insert, [265](#)
  - justify, [265](#)
  - len, [265](#)
  - lowerFirst, [265](#)
  - operator const char \*, [265](#)
  - operator!=, [265](#), [266](#)
  - operator<, [267](#)
  - operator<=, [267](#)
  - operator>, [268](#)
  - operator>=, [268](#)
  - operator+, [266](#)
  - operator+=, [267](#)
  - operator=, [267](#)
  - operator==, [267](#), [268](#)
  - operator[], [268](#)
  - ostr, [272](#)
  - pad, [268](#)
  - printf, [268](#)
  - pullLine, [269](#)
  - pullSeparators, [269](#)
  - pullToken, [269](#)
  - pullWord, [269](#)
  - removeNL, [269](#)
  - removeSeparators, [269](#)
  - retDouble, [270](#)
  - retFloat64, [270](#)
  - retInt, [270](#)
  - retStr, [270](#)
  - retStrDup, [270](#)
  - retUInt, [270](#)
  - reverse, [271](#)
  - split, [271](#)
  - str, [271](#)
  - subString, [271](#)
  - toLowerCase, [271](#)
  - toUpperCase, [271](#)
  - translateChar, [272](#)
  - truncate, [272](#)
- BString.cpp, [399](#)
  - barrayToString, [401](#)
  - base64\_decode\_table, [405](#)
  - blistToString, [401](#)
  - bstringListinList, [401](#)
  - bstringToArray, [401](#)
  - bstringToList, [401](#)
  - bstrncpy, [401](#)
  - bstrtrim, [401](#)
  - charToArray, [402](#)
  - charToList, [402](#)
  - floatToString, [402](#)
  - fromBString, [402](#), [403](#)
  - gmatch, [403](#)
  - int64ToString, [403](#)
  - intToString, [403](#)
  - MINUS, [400](#)
  - operator<<, [404](#)
  - operator>>, [404](#)
  - STRIP, [400](#)
  - toBString, [404](#), [405](#)
- BString.h, [405](#)
  - bstrncpy, [406](#)
  - bstrtrim, [406](#)
  - floatToString, [406](#)
  - from\_hex, [407](#)
  - fromBString, [407](#), [408](#)
  - int64ToString, [408](#)
  - intToString, [408](#)
  - operator<<, [408](#)

- operator>>, 408
- to\_hex, 408
- toBString, 409
- bstringListinList
  - BString.cpp, 401
- BStringLocked, 272
  - BStringLocked, 273
  - len, 273
  - operator BString, 273
  - operator+, 274
  - operator=, 274
- BStringLocked.h, 410
- BStringMutex, 274
  - BStringMutex, 274
- bstringToArray
  - BString.cpp, 401
- bstringToList
  - BString.cpp, 401
- bstrncpy
  - BString.cpp, 401
  - BString.h, 406
- bstrtrim
  - BString.cpp, 401
  - BString.h, 406
- bswap\_copy
  - BEndian.cpp, 339
  - BEndian.h, 343
- bswap\_p16
  - BEndian.h, 343
- bswap\_p32
  - BEndian.h, 344
- bswap\_p64
  - BEndian.h, 344
- bswap\_p8
  - BEndian.h, 344
- BSys.cpp, 410
  - delayMs, 410
  - delayUs, 410
- BSys.h, 411
  - delayMs, 411
  - delayUs, 411
- BTable, 275
  - ~BTable, 275
  - addRow, 275
  - BTable, 275
  - clear, 275
  - print, 276
  - setTitle, 276
- BTable.cpp, 411
- BTable.h, 412
- BTask, 276
  - ~BTask, 277
  - BTask, 277
  - init, 277
  - oname, 278
  - opolicy, 278
  - opriority, 279
  - orunning, 279
  - ostackSize, 279
  - othread, 279
  - run, 277
  - setPriority, 277
  - start, 278
  - stop, 278
  - taskFunc, 278
  - waitForCompletion, 278
- BTask.cpp, 412
- BTask.h, 412
- BThread, 279
  - ~BThread, 280
  - BThread, 280
  - cancel, 280
  - function, 280
  - getThread, 280
  - result, 281
  - running, 281
  - setInitPriority, 281
  - setInitStackSize, 281
  - setPriority, 281
  - start, 281
  - waitForCompletion, 281
- BThread.cpp, 412
- BThread.h, 412
- BTime, 282
  - addSeconds, 283
  - BTime, 283
  - getDate, 283
  - getSeconds, 283
  - getString, 283
  - getTime, 283
  - isLeapYear, 284
  - isSet, 284
  - operator!=, 284
  - operator<, 284
  - operator<=, 285
  - operator>, 285
  - operator>=, 285
  - operator+, 284
  - operator+=, 284
  - operator==, 285
  - set, 285
  - setString, 286
  - setYearDay, 286
- BTime.cpp, 413
  - monDays, 413
  - yearDays, 413
  - yearIsLeap, 413
- BTime.h, 414
- BTimeout
  - BTypes.h, 421
- BTimeoutForever
  - BTypes.h, 424
- BTimer, 286
  - ~BTimer, 287
  - add, 287
  - average, 287

- BTimer, 287
- clear, 288
- getElapsedTime, 288
- peak, 288
- start, 288
- stop, 288
- BTimer.cpp, 414
- BTimer.h, 414
- BTimeStamp, 289
  - ~BTimeStamp, 291
  - addMicroSeconds, 291
  - addMilliSeconds, 291
  - addSeconds, 291
  - BTimeStamp, 290, 291
  - clear, 292
  - compare, 292
  - day, 292
  - difference, 292
  - getDate, 292
  - getString, 292
  - getStringFormatted, 293
  - getStringNoMs, 293
  - getYearMicroSeconds, 293
  - getYearSeconds, 293
  - hour, 293
  - isLeap, 293
  - isSet, 294
  - microSecond, 294
  - minute, 294
  - month, 294
  - ohour, 298
  - omicroSecond, 298
  - omminute, 298
  - operator BString, 294
  - operator!=, 294
  - operator<, 294
  - operator<=, 295
  - operator>, 295
  - operator>=, 295
  - operator=, 295
  - operator==, 295
  - osecond, 298
  - ospare, 298
  - oyday, 298
  - oyear, 299
  - second, 295
  - set, 295, 296
  - setFirst, 296
  - setLast, 296
  - setNow, 296
  - setString, 297
  - setTime, 297
  - setYDay, 297
  - yday, 297
  - year, 297
- BTimeStamp.cpp, 414
  - fromBString, 415
  - mon\_yday, 415
- toBString, 415
- BTimeStamp.h, 415
  - fromBString, 416
  - toBString, 416
- BTimeStampMs, 299
  - ~BTimeStampMs, 301
  - addMilliSeconds, 301
  - addSeconds, 301
  - BTimeStampMs, 300
  - clear, 301
  - compare, 301
  - difference, 301
  - getDate, 302
  - getDurationString, 302
  - getDurationStringNoMs, 302
  - getString, 302
  - getStringNoMs, 302
  - getStringRaw, 303
  - getYearMilliSeconds, 303
  - getYearSeconds, 303
  - hour, 306
  - isLeap, 303
  - milliSecond, 306
  - minute, 306
  - operator<, 303
  - operator<=, 303
  - operator>, 304
  - operator>=, 304
  - sampleNumber, 306
  - second, 307
  - set, 304
  - setDurationString, 304
  - setFirst, 304
  - setLast, 304
  - setNow, 305
  - setString, 305
  - setTime, 305
  - setYDay, 305
  - subMilliSeconds, 305
  - subSeconds, 306
  - yday, 307
  - year, 307
- BTimeStampMs.cpp, 416
  - mon\_yday, 416
- BTimeStampMs.h, 417
- BTimeUs, 307
  - addMicroSeconds, 309
  - addSeconds, 309
  - BTimeUs, 308, 309
  - getDate, 309
  - getMicroSeconds, 309
  - getSeconds, 310
  - getString, 310
  - getStringUs, 310
  - getTime, 310
  - isLeapYear, 310
  - isSet, 310
  - operator BTime, 311



- operator!=, [311](#)
- operator<, [311](#)
- operator<=, [311](#)
- operator>, [312](#)
- operator>=, [312](#)
- operator+, [311](#)
- operator+=, [311](#)
- operator==, [311](#)
- set, [312](#)
- setString, [312](#)
- setYearDay, [312](#)
- BTimeUs.cpp, [417](#)
- monDays, [417](#)
- yearDays, [417](#)
- yearIsLeap, [417](#)
- BTimeUs.h, [418](#)
- BType
- BTypes.h, [423](#)
- BTypeBool
- BTypes.h, [423](#)
- BTypeChar
- BTypes.h, [423](#)
- BTypeComp
- BTypes.h, [423](#)
- BTypeCompArray
- BTypes.h, [423](#)
- BTypeCompArrayFixed
- BTypes.h, [423](#)
- BTypeCompDict
- BTypes.h, [423](#)
- BTypeCompList
- BTypes.h, [423](#)
- BTypeCompSingle
- BTypes.h, [423](#)
- BTypeError
- BTypes.h, [423](#)
- BTypeFloat32
- BTypes.h, [423](#)
- BTypeFloat64
- BTypes.h, [423](#)
- BTypeInt16
- BTypes.h, [423](#)
- BTypeInt32
- BTypes.h, [423](#)
- BTypeInt64
- BTypes.h, [423](#)
- BTypeInt8
- BTypes.h, [423](#)
- BTypeNone
- BTypes.h, [423](#)
- BTypeObj
- BTypes.h, [423](#)
- BTypes.h, [418](#)
- BArrayDouble, [419](#)
- BArrayFloat, [419](#)
- BChar, [419](#)
- BDouble, [420](#)
- BEventType, [422](#)
- BEventTypeClientConnect, [422](#)
- BEventTypeClientDisconnect, [422](#)
- BEventTypeConnect, [422](#)
- BEventTypeDisconnect, [422](#)
- BEventTypeError, [422](#)
- BEventTypeNone, [422](#)
- BEventTypeRead, [422](#)
- BEventTypeReadLine, [422](#)
- BEventTypeWrite, [422](#)
- BEventWaitAny, [422](#)
- BEventWaitClientConnect, [422](#)
- BEventWaitClientDisconnect, [422](#)
- BEventWaitConnect, [422](#)
- BEventWaitDisconnect, [422](#)
- BEventWaitError, [422](#)
- BEventWaitNone, [422](#)
- BEventWaitRead, [422](#)
- BEventWaitReadLine, [422](#)
- BEventWaitSet, [422](#)
- BEventWaitWrite, [422](#)
- BFloat, [420](#)
- BFloat32, [420](#)
- BFloat64, [420](#)
- BInt, [420](#)
- BInt16, [420](#)
- BInt32, [420](#)
- BInt64, [420](#)
- BInt8, [421](#)
- Bool, [421](#)
- BSize, [421](#)
- BTimeout, [421](#)
- BTimeoutForever, [424](#)
- BType, [423](#)
- BTypeBool, [423](#)
- BTypeChar, [423](#)
- BTypeComp, [423](#)
- BTypeCompArray, [423](#)
- BTypeCompArrayFixed, [423](#)
- BTypeCompDict, [423](#)
- BTypeCompList, [423](#)
- BTypeCompSingle, [423](#)
- BTypeError, [423](#)
- BTypeFloat32, [423](#)
- BTypeFloat64, [423](#)
- BTypeInt16, [423](#)
- BTypeInt32, [423](#)
- BTypeInt64, [423](#)
- BTypeInt8, [423](#)
- BTypeNone, [423](#)
- BTypeObj, [423](#)
- BTypeString, [423](#)
- BTypeTime, [423](#)
- BTypeTimeUs, [423](#)
- BTypeUInt16, [423](#)
- BTypeUInt32, [423](#)
- BTypeUInt64, [423](#)
- BTypeUInt8, [423](#)
- BUInt, [421](#)

- BUInt16, [421](#)
- BUInt32, [421](#)
- BUInt64, [421](#)
- BUInt8, [422](#)
- byteSwap16, [423](#)
- byteSwap32, [424](#)
- byteSwap64, [424](#)
- byteSwap8, [424](#)
- timeoutTicks, [424](#)
- BTypeString
  - BTypes.h, [423](#)
- BTypeTime
  - BTypes.h, [423](#)
- BTypeTimeUs
  - BTypes.h, [423](#)
- BTypeUInt16
  - BTypes.h, [423](#)
- BTypeUInt32
  - BTypes.h, [423](#)
- BTypeUInt64
  - BTypes.h, [423](#)
- BTypeUInt8
  - BTypes.h, [423](#)
- build\_x86\_64/BBuffer.d, [425](#)
- build\_x86\_64/BComms.d, [425](#)
- build\_x86\_64/BCond.d, [425](#)
- build\_x86\_64/BCondInt.d, [425](#)
- build\_x86\_64/BConfig.d, [425](#)
- build\_x86\_64/BCrc16.d, [425](#)
- build\_x86\_64/BCrc32.d, [425](#)
- build\_x86\_64/BDate.d, [425](#)
- build\_x86\_64/BDebug.d, [425](#)
- build\_x86\_64/BDict.d, [425](#)
- build\_x86\_64/BDir.d, [425](#)
- build\_x86\_64/BDuration.d, [425](#)
- build\_x86\_64/BEndian.d, [425](#)
- build\_x86\_64/BEntry.d, [425](#)
- build\_x86\_64/BError.d, [425](#)
- build\_x86\_64/BErrorTime.d, [425](#)
- build\_x86\_64/BEvent.d, [425](#)
- build\_x86\_64/BEvent1.d, [426](#)
- build\_x86\_64/BFifoCirc.d, [426](#)
- build\_x86\_64/BFile.d, [426](#)
- build\_x86\_64/BFileCsv.d, [426](#)
- build\_x86\_64/BFileData.d, [426](#)
- build\_x86\_64/BMutex.d, [426](#)
- build\_x86\_64/BMysql.d, [426](#)
- build\_x86\_64/Boap.d, [364](#)
- build\_x86\_64/BoapMc.d, [426](#)
- build\_x86\_64/BoapMc1.d, [426](#)
- build\_x86\_64/BoapnsC.d, [426](#)
- build\_x86\_64/BoapnsD.d, [426](#)
- build\_x86\_64/BObj.d, [426](#)
- build\_x86\_64/BObjStringFormat.d, [426](#)
- build\_x86\_64/BPoll.d, [426](#)
- build\_x86\_64/BRefData.d, [426](#)
- build\_x86\_64/BRtc.d, [426](#)
- build\_x86\_64/BRWLock.d, [426](#)
- build\_x86\_64/BSema.d, [427](#)
- build\_x86\_64/BSemaphore.d, [427](#)
- build\_x86\_64/BSocket.d, [427](#)
- build\_x86\_64/BSpi.d, [427](#)
- build\_x86\_64/BString.d, [427](#)
- build\_x86\_64/BSys.d, [427](#)
- build\_x86\_64/BTable.d, [427](#)
- build\_x86\_64/BTask.d, [427](#)
- build\_x86\_64/BThread.d, [427](#)
- build\_x86\_64/BTime.d, [427](#)
- build\_x86\_64/BTimer.d, [427](#)
- build\_x86\_64/BTimeStamp.d, [427](#)
- build\_x86\_64/BTimeStampMs.d, [427](#)
- build\_x86\_64/BTimeUs.d, [427](#)
- build\_x86\_64/BUrl.d, [427](#)
- BUInt
  - BTypes.h, [421](#)
- BUInt16
  - BTypes.h, [421](#)
- BUInt32
  - BTypes.h, [421](#)
- BUInt64
  - BTypes.h, [421](#)
- BUInt8
  - BTypes.h, [422](#)
- BUrl, [313](#)
  - ~BUrl, [313](#)
  - BUrl, [313](#)
  - readString, [314](#)
- BUrl.cpp, [427](#)
- BUrl.h, [428](#)
- byteRate
  - BComms, [37](#)
- byteSwap16
  - BTypes.h, [423](#)
- byteSwap32
  - BTypes.h, [424](#)
- byteSwap64
  - BTypes.h, [424](#)
- byteSwap8
  - BTypes.h, [424](#)
- cancel
  - BThread, [280](#)
- charToArray
  - BString.cpp, [402](#)
- charToList
  - BString.cpp, [402](#)
- checkApiVersion
  - BoapClientObject, [158](#)
- checksum
  - BFirmware.h, [358](#)
  - BFirmwareFileHeader, [129](#)
  - BFirmwareInfo, [131](#)
  - BFirmwareSegHeader, [133](#)
  - BoapMc.h, [369](#)
  - BoapMc1.h, [372](#)
  - BoapMc1PacketHead, [175](#)
  - BoapMcPacketHead, [188](#)

CHUNK  
  BRefData.cpp, 395  
clear  
  BCondBool, 44  
  BDate, 60  
  BDict< Type >, 68  
  BDictMap< Value >, 73  
  BDir, 77  
  BDuration, 80  
  BEntryFile, 86  
  BEntryList, 88  
  BError, 92  
  BErrorTime, 95  
  BEvent1Int, 101  
  BEvent1Pipe, 103  
  BEventPipe, 104  
  BFifo< Type >, 107  
  BFifoCirc< Type >, 114  
  BList< T >, 139  
  BPoll, 221  
  BQueue< T >, 223  
  BSemaphoreBool, 236  
  BString, 259  
  BTable, 275  
  BTimer, 288  
  BTimeStamp, 292  
  BTimeStampMs, 301  
clientGone  
  BoapServer, 203  
close  
  BComms, 37  
  BConfig, 57  
  BFile, 123  
  BMySQL, 151  
  BSocket, 242  
closeConnections  
  BoapServer, 203  
cmd  
  BoapMc.h, 369  
  BoapMc1.h, 372  
  BoapMc1PacketHead, 175  
  BoapMcPacketHead, 188  
  BoapPacketHead, 199, 200  
compare  
  BDate, 60  
  BString, 260  
  BTimeStamp, 292  
  BTimeStampMs, 301  
compareRegex  
  BString, 260  
compareWild  
  BString, 260  
compareWildExpression  
  BString, 260  
connect  
  BComms, 37  
  BSocket, 243  
connectService  
  BoapClientObject, 158  
convert  
  BString, 260, 261  
convertHex  
  BString, 261  
copy  
  BError, 92  
  BErrorTime, 95  
  BRefData, 225  
  BString, 262  
crc32\_tab  
  BCrc32.cpp, 325  
csvDecode  
  BString, 262  
csvEncode  
  BString, 262  
data  
  BBuffer, 27  
  BDataChunk, 58  
  BoapMc1.h, 373  
  BoapMc1Packet, 174  
  BoapMcPacket, 187  
  BoapPacket, 194  
  BRefData, 225  
dataLength  
  BFirmware.h, 358  
  BFirmwareSegHeader, 133  
dataOffset  
  BObjMember, 218  
day  
  BDate, 61  
  BTimeStamp, 292  
daysInMonth  
  BDate, 61  
db  
  BMySQL, 151  
DEBUG  
  Boap.cpp, 363  
  BoapSimple.cc, 376  
DEBUG\_LOCAL  
  BoapMc.cpp, 366  
DEBUG\_LOCAL1  
  BoapMc.cpp, 366  
decrement  
  BCondInt, 46  
  BCondValue, 51  
  BCondWrap, 54  
defaultSize  
  BFifoCirc< Type >, 114  
del  
  BArray< T >, 21  
  BDict< Type >, 68, 69  
  BDictMap< Value >, 73  
  BEntryList, 88  
  BFileData, 128  
  BList< T >, 139  
  BMySQL, 151  
  BString, 262

- delayMs
  - BSys.cpp, 410
  - BSys.h, 411
- delayUs
  - BSys.cpp, 410
  - BSys.h, 411
- delEntry
  - Boapns::Boapns, 192
- deleteEntry
  - BEntryList, 88
- deleteFirst
  - BList< T >, 140
- deleteLast
  - BList< T >, 140
- deleteRef
  - BRefData, 226
- delFd
  - BPoll, 221
- DGRAM
  - BSocket, 241
- difference
  - BFifoCircPos, 119
  - BTimeStamp, 292
  - BTimeStampMs, 301
- dirname
  - BString, 262
- disconnect
  - BComms, 37
- disconnectService
  - BoapClientObject, 159
- dl1printf
  - BDebug.h, 332
  - BoapMc.cpp, 367
- dl2printf
  - BDebug.h, 332
- dl3printf
  - BDebug.h, 332
- dl4printf
  - BDebug.h, 332
- dlprintf
  - BoapMc.cpp, 367
- doConnectionPriority
  - BoapServiceObject, 211
- doPing
  - BoapServiceObject, 211
- doPoll
  - BPoll, 221
- doPollEvents
  - BPoll, 221
- Double
  - BoapSimple.h, 377
- dprintf
  - BDebug.h, 332
  - BFifoCirc.cpp, 352
  - Boap.cpp, 363
  - BoapSimple.cc, 376
- dumpBacktrace
  - BDebugBacktrace, 66
- dumpBacktraceFile
  - BDebugBacktrace, 66
- dumpBacktraceStdout
  - BDebugBacktrace, 66
- dumpBacktraceSyslog
  - BDebugBacktrace, 66
- end
  - BCondResource, 49
  - BList< T >, 140
- entryName
  - BDir, 77
- entryStat
  - BDir, 77
- entryStat64
  - BDir, 77
- eprintf
  - BDebug.h, 332
- Error
  - BErrorTime, 94
- error
  - BDir, 77
  - BoapMc.h, 369
  - BoapMc1.h, 373
  - BoapMc1PacketHead, 175
  - BoapMcPacketHead, 188
- ErrorAccessDenied
  - BError.h, 349
- ErrorApiVersion
  - BError.h, 349
- ErrorAppBase
  - BError.h, 349
- ErrorChecksum
  - BError.h, 349
- ErrorComms
  - BError.h, 349
- ErrorConfig
  - BError.h, 349
- ErrorData
  - BError.h, 349
- ErrorDataPresent
  - BError.h, 349
- ErrorDataTruncated
  - BError.h, 349
- ErrorEndOfData
  - BError.h, 349
- ErrorEndOfFile
  - BError.h, 349
- ErrorFile
  - BError.h, 349
- ErrorFormat
  - BError.h, 349
- ErrorInit
  - BError.h, 349
- ErrorMisc
  - BError.h, 349
- ErrorNoData
  - BError.h, 349
- ErrorNotAvailable

- BError.h, [349](#)
- ErrorNotImplemented
  - BError.h, [349](#)
- ErrorOk
  - BError.h, [349](#)
- ErrorOverrun
  - BError.h, [349](#)
- ErrorParam
  - BError.h, [349](#)
- ErrorResourceLimit
  - BError.h, [349](#)
- ErrorTimeout
  - BError.h, [349](#)
- ErrorUnderrun
  - BError.h, [349](#)
- ErrorUserBase
  - BError.h, [349](#)
- ErrorWarning
  - BError.h, [349](#)
- escapeString
  - BMySQL, [152](#)
- eventEnable
  - BComms, [38](#)
- eventQueue
  - BComms, [38](#)
- extension
  - BString, [262](#)
- fgets
  - BFile, [123](#)
- field
  - BString, [263](#)
- fields
  - BString, [263](#)
- fileLength
  - BFirmware.h, [358](#)
  - BFirmwareFileHeader, [129](#)
  - BFirmwareSegHeader, [133](#)
- fileName
  - BConfig, [57](#)
  - BFile, [123](#)
- filename
  - BEntryFile, [86](#)
- find
  - BDict< Type >, [69](#)
  - BEntryList, [88](#)
  - BFileData, [128](#)
  - BNameValueList< T >, [155](#)
  - BString, [263](#)
- findPos
  - BNameValueList< T >, [155](#)
- findReverse
  - BString, [263](#)
- findValue
  - BConfig, [57](#)
  - BEntryList, [89](#)
- firstLine
  - BString, [263](#)
- fixedLen
  - BString, [264](#)
- floatToString
  - BString.cpp, [402](#)
  - BString.h, [406](#)
- Flush
  - BComms, [36](#)
- flush
  - BComms, [38](#)
  - BFile, [123](#)
  - BMySQL, [152](#)
- FlushRead
  - BComms, [37](#)
- FlushReadWrite
  - BComms, [37](#)
- FlushWrite
  - BComms, [37](#)
- format
  - BFirmware.h, [359](#)
  - BFirmwareFileHeader, [130](#)
  - BFirmwareSegHeader, [133](#)
- from\_hex
  - BString.h, [407](#)
- fromBString
  - BDate-1.cpp, [326](#)
  - BDate.cpp, [327](#)
  - BDate.h, [328](#)
  - BDict.cpp, [335](#)
  - BDict.h, [336](#)
  - BString.cpp, [402](#), [403](#)
  - BString.h, [407](#), [408](#)
  - BTimeStamp.cpp, [415](#)
  - BTimeStamp.h, [416](#)
- front
  - BList< T >, [140](#)
- function
  - BThread, [280](#)
- get
  - BList< T >, [141](#)
  - BMySQL, [152](#)
  - BString, [264](#)
- getAddress
  - BSocket, [243](#)
- getApiVersion
  - BoapMc1Comms, [167](#)
  - BoapMcClientObject, [177](#)
  - BoapMcComms, [181](#)
- getBinary
  - BEvent1, [98](#)
  - BEvent1Error, [99](#)
- getCmd
  - BoapPacket, [195](#)
- getConnectionsNumber
  - BoapServer, [203](#)
- getDate
  - BDate, [61](#)
  - BTime, [283](#)
  - BTimeStamp, [292](#)
  - BTimeStampMs, [302](#)

- BTimeUs, 309
- getDebugString
  - BObj, 217
- getDurationString
  - BTimeStampMs, 302
- getDurationStringNoMs
  - BTimeStampMs, 302
- getElapsedTime
  - BTimer, 288
- getEntry
  - Boapns::Boapns, 192
- getEntryList
  - Boapns::Boapns, 192
- getErrorNo
  - BError, 92
  - BErrorTime, 95
- getEvent
  - BEvent1Int, 101
  - BEvent1Pipe, 103
- getEventSocket
  - BoapServer, 203
- getFd
  - BEvent1Int, 101
  - BEventPipe, 105
  - BFile, 123
  - BSocket, 243
- getHead
  - BoapServerConnection, 207
- getHexString
  - BBufferStore, 30
- getHostName
  - BoapServer, 203, 204
  - BSocketAddressINET, 251
- getIpAddresses
  - BSocketAddressINET, 251
- getIpAddressList
  - BSocketAddressINET, 251
- getIpAddressListAll
  - BSocketAddressINET, 251
- getMember
  - BObj, 217
- getMembers
  - BObj, 217
- getMicroSeconds
  - BDuration, 80
  - BTimeUs, 309
- getMTU
  - BSocket, 243
- getName
  - BEntry, 83
  - BNameValue< T >, 154
- getNewName
  - Boapns::Boapns, 192
- getNextId
  - BFileData, 128
- getNumber
  - BError, 92
- getPollFds
  - BPoll, 221
- getPollFdsNum
  - BPoll, 221
- getPos
  - BBufferStore, 30
- getReceiveFd
  - BEvent1Pipe, 103
- getSeconds
  - BDuration, 80
  - BTime, 283
  - BTimeUs, 310
- getServiceName
  - BoapClientObject, 159
- getSocket
  - BoapServer, 204
  - BoapServerConnection, 207
- getSockOpt
  - BSocket, 243
- getString
  - BDate, 61
  - BDuration, 81
  - BEntryList, 89
  - BError, 92
  - BErrorTime, 95
  - BSocketAddress, 248
  - BSocketAddressINET, 251
  - BTime, 283
  - BTimeStamp, 292
  - BTimeStampMs, 302
  - BTimeUs, 310
- getStringFormatted
  - BDate, 61
  - BTimeStamp, 293
- getStringNoMs
  - BTimeStamp, 293
  - BTimeStampMs, 302
- getStringRaw
  - BTimeStampMs, 303
- getStringUs
  - BTimeUs, 310
- getThread
  - BThread, 280
- getTime
  - BDebug.cpp, 330
  - BDebug.h, 334
  - BErrorTime, 95
  - BTime, 283
  - BTimeUs, 310
- getTimeout
  - BCondInt.cpp, 322
- getTokenList
  - BString, 264
- getType
  - BEvent1, 98
  - BObj, 217
- getValue
  - BAtomic< Type >, 23
  - BAtomicCount, 25

- BEntry, [83](#)
- BNameValue< T >, [154](#)
- BSema, [232](#)
- BSemaphore, [235](#)
- getVersion
  - Boapns::Boapns, [192](#)
- getYearMicroSeconds
  - BTimeStamp, [293](#)
- getYearMilliSeconds
  - BTimeStampMs, [303](#)
- getYearSeconds
  - BTimeStamp, [293](#)
  - BTimeStampMs, [303](#)
- gmatch
  - BString.cpp, [403](#)
- goTo
  - BList< T >, [141](#)
- handleReconnect
  - BoapClientObject, [159](#)
- has
  - BList< T >, [141](#)
- hash
  - BString, [264](#)
- hashPrint
  - BDict< Type >, [69](#)
- hasKey
  - BDict< Type >, [69](#)
  - BDictMap< Value >, [74](#)
- head
  - BoapMc1.h, [373](#)
  - BoapMc1Packet, [174](#)
  - BoapMcPacket, [187](#)
- hostName
  - Boapns::BoapEntry, [164](#)
- hour
  - BDuration, [81](#)
  - BTimeStamp, [293](#)
  - BTimeStampMs, [306](#)
- htobe
  - BEndian.h, [344](#), [345](#)
- htobe16
  - BEndian.h, [341](#)
- htobe32
  - BEndian.h, [341](#)
- htobe64
  - BEndian.h, [341](#)
- htole
  - BEndian.h, [345](#), [346](#)
- htole16
  - BEndian.h, [341](#)
- htole32
  - BEndian.h, [341](#)
- htole64
  - BEndian.h, [341](#)
- increment
  - BCondInt, [46](#)
  - BCondValue, [51](#)
- BCondWrap, [54](#)
- BFifoCircPos, [119](#)
- init
  - BComms, [38](#)
  - BoapServer, [204](#)
  - BoapServerConnection, [207](#)
  - BRtc, [227](#)
  - BRtcThreaded, [229](#)
  - BSocket, [243](#)
  - BSpi, [253](#)
  - BTask, [277](#)
- insert
  - BArray< T >, [21](#)
  - BDict< Type >, [69](#)
  - BEntryList, [89](#)
  - BList< T >, [141](#)
  - BMySQL, [152](#)
  - BString, [265](#)
- insertAfter
  - BList< T >, [142](#)
- Int16
  - BoapSimple.h, [377](#)
- Int32
  - BoapSimple.h, [377](#)
- int64ToString
  - BString.cpp, [403](#)
  - BString.h, [408](#)
- Int8
  - BoapSimple.h, [378](#)
- intToString
  - BString.cpp, [403](#)
  - BString.h, [408](#)
- inUse
  - BCondResource, [49](#)
- IP\_MTU
  - BSocket.cpp, [398](#)
- IS\_BIG\_ENDIAN
  - Boap.cpp, [363](#)
- isConnected
  - BComms, [38](#)
- isEnd
  - BDictMap< Value >, [74](#)
  - BFile, [124](#)
  - BList< T >, [142](#)
- isLeap
  - BDate, [61](#)
  - BTimeStamp, [293](#)
  - BTimeStampMs, [303](#)
- isLeapYear
  - BTime, [284](#)
  - BTimeUs, [310](#)
- isOpen
  - BFile, [124](#)
- isSet
  - BDate, [62](#)
  - BEntryList, [89](#)
  - BTime, [284](#)
  - BTimeStamp, [294](#)

- BTimeUs, 310
- isStart
  - BList< T >, 142
- item
  - BList< T >::Node, 315
- itemType
  - BFirmware.h, 359
  - BFirmwareFileHeader, 130
  - BFirmwareSegHeader, 134
- iterator
  - BDict< Type >, 67
  - BDictMap< Value >, 73
- justify
  - BString, 265
- key
  - BDict< Type >, 69
  - BDictItem< Type >, 72
  - BDictMap< Value >, 74
- le16toh
  - BEndian.h, 342
- le32toh
  - BEndian.h, 342
- le64toh
  - BEndian.h, 342
- len
  - BRefData, 226
  - BSocketAddress, 248
  - BString, 265
  - BStringLocked, 273
- length
  - BFile, 124
  - BFirmware.h, 359
  - BFirmwareInfo, 132
  - BFirmwareSegHeader, 134
  - BoapMc.h, 369
  - BoapMc1.h, 373
  - BoapMc1PacketHead, 175
  - BoapMcPacketHead, 188
  - BoapPacketHead, 200
- letoh
  - BEndian.h, 346, 347
- line
  - BEntry, 83
- listen
  - BSocket, 244
- lock
  - BCondResource, 49
  - BMutex, 148
  - BMutexLock, 150
- locked
  - BCondResource, 49
- lowerFirst
  - BString, 265
- magic
  - BFirmware.h, 359
- BFirmwareFileHeader, 130
- BFirmwareInfo, 132
- BFirmwareSegHeader, 134
- BoapMc1.h, 373
- BoapMc1PacketHead, 176
- mapCircularBuffer
  - BFifoCirc< Type >, 115
- MDEBUG
  - BMutex.cpp, 361
- membersPrint
  - BObj, 217
- microSecond
  - BDuration, 81
  - BTimeStamp, 294
- milliSecond
  - BTimeStampMs, 306
- MINUS
  - BString.cpp, 400
- minute
  - BDuration, 81
  - BTimeStamp, 294
  - BTimeStampMs, 306
- Mode
  - BSpi, 253
- Mode0
  - BSpi, 253
- Mode1
  - BSpi, 253
- Mode2
  - BSpi, 253
- Mode3
  - BSpi, 253
- mon\_yday
  - BDate-1.cpp, 326
  - BDate.cpp, 327
  - BTimeStamp.cpp, 415
  - BTimeStampMs.cpp, 416
- monDays
  - BTime.cpp, 413
  - BTimeUs.cpp, 417
- month
  - BDate, 62
  - BTimeStamp, 294
- MSG\_NOSIGNAL
  - BSocket.h, 398
- name
  - BComms, 38
  - Boapns::BoapEntry, 164
  - BoapServiceObject, 211
  - BObjMember, 218
- nbytes
  - BoapPacket, 195
- newConnection
  - BoapServer, 204
- next
  - BDictMap< Value >, 74
  - BList< T >, 142
  - BNode, 156



- Node
  - BList< T >::Node, [314](#)
- nodeCreate
  - BList< T >, [142](#)
- nodeGet
  - BList< T >, [143](#)
- None
  - BErrorTime, [94](#)
- Normal
  - BMutex, [148](#)
- NOTHREADS
  - BoapServer, [202](#)
- nprintf
  - BDebug.h, [333](#)
- NType
  - BSocket, [241](#)
- num
  - BError, [92](#)
- number
  - BArray< T >, [21](#)
  - BList< T >, [143](#)
  - BoapMc1.h, [373](#)
  - BoapMc1Error, [173](#)
- numSegments
  - BFirmware.h, [359](#)
  - BFirmwareFileHeader, [130](#)
- oaddressFrom
  - BoapMc1Comms, [170](#)
  - BoapMcClientObject, [178](#)
  - BoapMcComms, [183](#)
- oaddressTo
  - BoapMc1Comms, [170](#)
  - BoapMcClientObject, [178](#)
  - BoapMcComms, [183](#)
- oapiVersion
  - BoapClientObject, [161](#)
  - BoapMc1Comms, [170](#)
  - BoapMcClientObject, [178](#)
  - BoapMcComms, [184](#)
  - BoapMcServiceObject, [190](#)
  - BoapServiceObject, [213](#)
- ocmd
  - BoapFuncEntry, [165](#)
- ocomms
  - BoapMc1Comms, [170](#)
  - BoapMcClientObject, [178](#)
  - BoapMcComms, [184](#)
  - BoapMcSignalObject, [191](#)
- oconnected
  - BComms, [40](#)
  - BoapClientObject, [161](#)
- odata
  - BBuffer, [28](#)
  - BFifo< Type >, [112](#)
  - BFifoCirc< Type >, [117](#)
- odataSize
  - BBuffer, [28](#)
- oerror
  - BoapMc1Comms, [170](#)
- oevent
  - BComms, [40](#)
- oeventEnabled
  - BComms, [40](#)
- oeventNum
  - BComms, [41](#)
- oeventQueue
  - BComms, [41](#)
- oeventSet
  - BComms, [41](#)
- ofunc
  - BoapFuncEntry, [165](#)
- ofuncList
  - BoapServiceObject, [213](#)
- ohalfDuplex
  - BoapMc1Comms, [170](#)
- ohour
  - BTimeStamp, [298](#)
- olength
  - BList< T >, [146](#)
- olock
  - BFifoCirc< Type >, [117](#)
  - BoapClientObject, [161](#)
- olockCall
  - BoapMc1Comms, [171](#)
  - BoapMcComms, [184](#)
- olockTx
  - BoapMc1Comms, [171](#)
  - BoapMcComms, [184](#)
- omaxLength
  - BoapClientObject, [161](#)
- omicroSecond
  - BTimeStamp, [298](#)
- ominute
  - BTimeStamp, [298](#)
- oname
  - BoapClientObject, [162](#)
  - BoapServiceObject, [214](#)
  - BTask, [278](#)
- onodes
  - BList< T >, [147](#)
- onumOperations
  - BoapServer, [206](#)
- oobject
  - BoapServiceEntry, [209](#)
- opacket
  - BoapMcClientObject, [178](#)
  - BoapMcComms, [184](#)
- opacketMode
  - BComms, [41](#)
- opacketReqQueue
  - BoapMcComms, [184](#)
- opacketReqRx
  - BoapMcComms, [184](#)
- opacketReqTx
  - BoapMcComms, [185](#)
- opacketRpcCmd

- BoapMc1Comms, 171
- opacketRpcDoneSema
  - BoapMc1Comms, 171
- opacketRpcSema
  - BoapMc1Comms, 171
- opacketRx
  - BoapMc1Comms, 171
  - BoapMcComms, 185
- opacketRxBase
  - BoapMc1Comms, 172
- opacketRxSema
  - BoapMcComms, 185
- opacketTx
  - BoapMc1Comms, 172
  - BoapMcComms, 185
- opacketTxBase
  - BoapMc1Comms, 172
- opacketTxQueue
  - BoapMcComms, 185
- opacketTxQueueWriteNum
  - BoapMcComms, 185
- opacketTxSema
  - BoapMcComms, 186
- open
  - BConfig, 57
  - BDir, 78
  - BEntryFile, 86
  - BFile, 124
  - BFileData, 128
  - BMysql, 152
- operator BNode \*
  - BIter, 135
- operator BString
  - BDate, 62
  - BStringLocked, 273
  - BTimeStamp, 294
- operator BTime
  - BTimeUs, 311
- operator const char \*
  - BString, 265
- operator const SockAddr \*
  - BSocketAddress, 248
- operator int
  - BCondBool, 44
  - BError, 93
  - BErrorTime, 96
  - BFifoCircPos, 120
  - BSemaphoreBool, 236
- operator long
  - BAtomicCount, 25
- operator Type
  - BAtomic< Type >, 23
- operator!=
  - BDate, 62
  - BFifoCircPos, 120
  - BSocketAddress, 248
  - BString, 265, 266
  - BTime, 284
  - BTimeStamp, 294
  - BTimeUs, 311
- operator<
  - BDate, 62
  - BString, 267
  - BTime, 284
  - BTimeStamp, 294
  - BTimeStampMs, 303
  - BTimeUs, 311
- operator<<
  - BString.cpp, 404
  - BString.h, 408
- operator<=
  - BDate, 62
  - BString, 267
  - BTime, 285
  - BTimeStamp, 295
  - BTimeStampMs, 303
  - BTimeUs, 311
- operator>
  - BDate, 63
  - BString, 268
  - BTime, 285
  - BTimeStamp, 295
  - BTimeStampMs, 304
  - BTimeUs, 312
- operator>>
  - BString.cpp, 404
  - BString.h, 408
- operator>=
  - BDate, 63
  - BString, 268
  - BTime, 285
  - BTimeStamp, 295
  - BTimeStampMs, 304
  - BTimeUs, 312
- operator+
  - BDict< Type >, 70
  - BList< T >, 143
  - BString, 266
  - BStringLocked, 274
  - BTime, 284
  - BTimeUs, 311
- operator++
  - BAtomic< Type >, 23
  - BAtomicCount, 25
  - BCondInt, 46
  - BCondValue, 51
  - BCondWrap, 54
- operator+=
  - BCondInt, 46
  - BCondValue, 51
  - BCondWrap, 54
  - BFifoCircPos, 120
  - BString, 267
  - BTime, 284
  - BTimeUs, 311
- operator--

- BAtomic< Type >, 23, 24
- BAtomicCount, 25
- BCondInt, 47
- BCondValue, 52
- BCondWrap, 55
- operator==
  - BCondInt, 47
  - BCondValue, 52
  - BCondWrap, 55
- operator=
  - BDict< Type >, 70
  - BEntryList, 89
  - BFile, 125
  - BList< T >, 143
  - BMutex, 149
  - BRefData, 226
  - BRWLock, 230
  - BSema, 233
  - BSemaphore, 235
  - BSemaphoreBool, 237
  - BSemaphoreCount, 239
  - BSocketAddress, 249
  - BString, 267
  - BStringLocked, 274
  - BTimeStamp, 295
- operator==
  - BDate, 62
  - BFifoCircPos, 120
  - BIter, 135
  - BSemaphoreBool, 237
  - BSocketAddress, 249
  - BString, 267, 268
  - BTime, 285
  - BTimeStamp, 295
  - BTimeUs, 311
- operator[]
  - BDict< Type >, 70, 71
  - BDictMap< Value >, 74
  - BFifo< Type >, 107
  - BFifoCirc< Type >, 115
  - BList< T >, 143, 144
  - BString, 268
- opolicy
  - BTask, 278
- opos
  - BBufferStore, 35
- opriority
  - BoapClientObject, 162
  - BTask, 279
- oreadPos
  - BFifo< Type >, 112
  - BFifoCirc< Type >, 118
- oreconnect
  - BoapClientObject, 162
- oreqSize
  - BoapMc1Comms, 172
- orunning
  - BTask, 279
- orx
  - BoapClientObject, 162
  - BoapSignalObject, 215
- osecond
  - BTimeStamp, 298
- oserver
  - BoapServiceObject, 214
- oservice
  - BoapClientObject, 162
  - BoapServiceEntry, 209
- osize
  - BBuffer, 28
  - BFifo< Type >, 112
  - BFifoCirc< Type >, 118
- oslave
  - BoapMcComms, 186
- ospare
  - BTimeStamp, 298
- ostackSize
  - BTask, 279
- ostr
  - BString, 272
- oswapBytes
  - BBufferStore, 35
- othread
  - BTask, 279
- othreaded
  - BoapMc1Comms, 172
  - BoapMcComms, 186
- otimeout
  - BComms, 41
  - BoapClientObject, 162
  - BoapMc1Comms, 172
  - BoapMcComms, 186
- otx
  - BoapClientObject, 162
  - BoapSignalObject, 216
- ovmSize
  - BFifoCirc< Type >, 118
- owriteNumFifoSamples
  - BFifoCirc< Type >, 118
- owritePos
  - BFifo< Type >, 112
  - BFifoCirc< Type >, 118
- oyday
  - BDate, 64
  - BTimeStamp, 298
- oyear
  - BDate, 65
  - BTimeStamp, 299
- packetMode
  - BComms, 39
- packetRecv
  - BoapMcComms, 181
- packetRx
  - BoapMc1Comms, 167
- packetRxData
  - BoapMc1Comms, 168

- packetRxEnd
  - BoapMc1Comms, 168
- packetSend
  - BoapMcComms, 181
- packetTx
  - BoapMc1Comms, 168
- pad
  - BString, 268
- peak
  - BTimer, 288
- peekHead
  - BoapPacket, 195
- performCall
  - BoapClientObject, 159
  - BoapMcClientObject, 177
  - BoapMcComms, 181
- performRecv
  - BoapClientObject, 159, 160
  - BoapMcClientObject, 177
- performSend
  - BoapClientObject, 160
  - BoapMcClientObject, 177
  - BoapMcComms, 181
  - BoapMcSignalObject, 190
  - BoapSignalObject, 215
- ping
  - BoapClientObject, 160
- pingLocked
  - BoapClientObject, 160
- platform
  - BFirmware.h, 359
  - BFirmwareFileHeader, 130
  - BFirmwareSegHeader, 134
- PollFd
  - BPoll, 220
- pop
  - BBufferStore, 30–32
  - BList< T >, 144
  - BoapPacket, 195, 196
- popHead
  - BoapPacket, 196, 197
- port
  - Boapns::BoapEntry, 164
  - BSocketAddressINET, 251
- pos
  - BFifoCircPos, 120
- position
  - BFile, 125
  - BList< T >, 144
- post
  - BSema, 233
- prev
  - BList< T >, 144
  - BNode, 156
- print
  - BEntry, 83
  - BEntryList, 90
  - BTable, 276
- printf
  - BFile, 125
  - BString, 268
- Priority
  - BSocket, 241
- PriorityHigh
  - BSocket, 241
- PriorityLow
  - BSocket, 241
- PriorityNormal
  - BSocket, 241
- process
  - BoapMcServiceObject, 189
  - BoapServer, 204, 205
  - BoapServerConnection, 208
  - BoapServiceObject, 211, 212
- processEvent
  - BoapMcServiceObject, 189
  - BoapServer, 205
  - BoapServiceObject, 212
- processPacket
  - BoapMcComms, 181
- processRequest
  - BoapMc1Comms, 168
  - BoapMcComms, 182
- processRequests
  - BoapMc1Comms, 168
  - BoapMcComms, 182
- processRx
  - BoapMc1Comms, 168
  - BoapMcComms, 182
- pullLine
  - BString, 269
- pullSeparators
  - BString, 269
- pullToken
  - BString, 269
- pullWord
  - BString, 269
- push
  - BBufferStore, 32–34
  - BList< T >, 145
  - BoapPacket, 197, 198
- pushHead
  - BoapPacket, 198
- query
  - BMySQL, 153
- queueAdd
  - BList< T >, 145
- queueGet
  - BList< T >, 145
- raw
  - BSocketAddress, 249
- rdLock
  - BRWLock, 230
- read
  - BComms, 39

- BConfig, 57
- BDir, 78
- BEntryFile, 86
- BEventPipe, 105
- BFifo< Type >, 107, 108
- BFifoCirc< Type >, 115
- BFile, 125
- BQueue< T >, 223
- readAvailable
  - BComms, 39
  - BEventPipe, 105
  - BFifo< Type >, 108
  - BFifoCirc< Type >, 115
  - BQueue< T >, 223
- readAvailableChunk
  - BFifo< Type >, 108
- readCsv
  - BFileCsv, 127
- readData
  - BFifo< Type >, 108
  - BFifoCirc< Type >, 115
- readDone
  - BFifo< Type >, 109
  - BFifoCirc< Type >, 115
- readPos
  - BFifo< Type >, 109
- readString
  - BFile, 125
  - BUrl, 314
- readWaitAvailable
  - BFifoCirc< Type >, 116
- rear
  - BArray< T >, 21
  - BList< T >, 145
- rebase
  - BFifo< Type >, 109
- Recursive
  - BMutex, 148
- recv
  - BSocket, 244
- recvAvailable
  - BSocket, 244
- recvFrom
  - BSocket, 244
- recvFromWithTimeout
  - BSocket, 244
- recvWithTimeout
  - BSocket, 244
- removeNL
  - BString, 269
- removeSeparators
  - BString, 269
- reserved
  - BoapPacketHead, 200
- resize
  - BBuffer, 27
  - BFifo< Type >, 109
  - BoapPacket, 198
- result
  - BThread, 281
- retDouble
  - BString, 270
- retFloat64
  - BString, 270
- retInt
  - BString, 270
- retStr
  - BString, 270
- retStrDup
  - BString, 270
- retUInt
  - BString, 270
- reverse
  - BString, 271
- roundSize
  - BBuffer.cpp, 319
  - BoapSimple.cc, 376
- run
  - BoapServer, 205
  - BTask, 277
- running
  - BThread, 281
- sampleNumber
  - BTimeStampMs, 306
- second
  - BDuration, 81
  - BTimeStamp, 295
  - BTimeStampMs, 307
- seek
  - BFile, 125
- send
  - BSocket, 245
- sendChunks
  - BSocket, 245
- sendEvent
  - BEvent1Int, 101
  - BEvent1Pipe, 103
  - BoapMcServiceObject, 189
  - BoapServer, 206
  - BoapServiceObject, 212, 213
- sendTo
  - BSocket, 245
- service
  - Boapns::BoapEntry, 164
  - BoapPacketHead, 200
- set
  - BCondBool, 44
  - BDate, 63
  - BDuration, 81
  - BError, 93
  - BErrorTime, 96
  - BFifoCircPos, 120
  - BSemaphore, 235
  - BSemaphoreBool, 237
  - BSocketAddress, 249
  - BSocketAddressINET, 252

- BTime, 285
- BTimeStamp, 295, 296
- BTimeStampMs, 304
- BTimeUs, 312
- setAddress
  - BoapMc1Comms, 169
  - BoapMcClientObject, 178
  - BoapMcComms, 182
- setBinary
  - BEvent1, 98
  - BEvent1Error, 100
- setBroadCast
  - BSocket, 245
- setComms
  - BoapMc1Comms, 169
  - BoapMcComms, 182, 183
- setCommsMode
  - BoapMc1Comms, 169
  - BoapMcComms, 183
- setConnectionPriority
  - BoapClientObject, 160
- setData
  - BBuffer, 27
  - BoapPacket, 199
- setDebug
  - BDebug.cpp, 330
  - BDebug.h, 334
  - BMysql, 153
- setDurationString
  - BTimeStampMs, 304
- setError
  - BError, 93
- setFd
  - BSocket, 245
- setFirst
  - BDate, 63
  - BTimeStamp, 296
  - BTimeStampMs, 304
- setHexString
  - BBufferStore, 35
- setInitPriority
  - BThread, 281
- setInitStackSize
  - BThread, 281
- setLast
  - BDate, 63
  - BTimeStamp, 296
  - BTimeStampMs, 304
- setLen
  - BRefData, 226
- setLine
  - BEntry, 84
- setMaxLength
  - BoapClientObject, 161
  - BoapServerConnection, 208
- setMember
  - BObj, 218
- setMembers
  - BObj, 218
- setName
  - BEntry, 84
  - BoapServiceObject, 213
- setNow
  - BDate, 64
  - BTimeStamp, 296
  - BTimeStampMs, 305
- setPacketMode
  - BComms, 39
- setPort
  - BSocketAddressINET, 252
- setPos
  - BBufferStore, 35
- setPriority
  - BSocket, 246
  - BTask, 277
  - BThread, 281
- setReuseAddress
  - BSocket, 246
- setSize
  - BBuffer, 27
  - BFifoCircPos, 121
- setSockOpt
  - BSocket, 246
- setSort
  - BDir, 78
- setString
  - BDate, 64
  - BDuration, 81
  - BTime, 286
  - BTimeStamp, 297
  - BTimeStampMs, 305
  - BTimeUs, 312
- setTime
  - BTimeStamp, 297
  - BTimeStampMs, 305
- setTimeout
  - BComms, 39
  - BoapClientObject, 161
  - BoapMc1Comms, 169
  - BoapMcComms, 183
- setTitle
  - BTable, 276
- setValue
  - BCondInt, 47
  - BCondValue, 52
  - BCondWrap, 55
  - BEntry, 84
  - BEntryList, 90
  - BSemaphoreCount, 239
- setValueRaw
  - BEntryList, 90
- setVBuf
  - BFile, 126
- setWild
  - BDir, 78
- setYDay

- BDate, 64
- BTimeStamp, 297
- BTimeStampMs, 305
- setYearDay
  - BTime, 286
  - BTimeUs, 312
- shutdown
  - BSocket, 246
- signal
  - BCond, 42
- size
  - BBuffer, 28
  - BDataChunk, 58
  - BDictMap< Value >, 75
  - BFifo< Type >, 109
  - BFifoCirc< Type >, 116
  - BList< T >, 145
  - BObjMember, 219
- SO\_PRIORITY
  - BSocket.h, 398
- SockAddr
  - BSocketAddress, 247
- SockAddrIP
  - BSocketAddressINET, 250
- SOL\_IP
  - BSocket.h, 399
- sort
  - BArray< T >, 22
  - BList< T >, 146
- SortFunc
  - BArray< T >, 20
  - BList< T >, 138
- special
  - BFirmware.h, 359
  - BFirmwareFileHeader, 130
  - BFirmwareSegHeader, 134
- split
  - BString, 271
- start
  - BCondResource, 49
  - BDictMap< Value >, 75
  - BList< T >, 146
  - BTask, 278
  - BThread, 281
  - BTimer, 288
- startAddress
  - BFirmware.h, 359
  - BFirmwareFileHeader, 130
- stop
  - BTask, 278
  - BTimer, 288
- str
  - BError, 93
  - BString, 271
- STRBUF
  - BFile.cpp, 353
- STREAM
  - BSocket, 241
- string
  - BoapMc1.h, 374
  - BoapMc1Error, 173
- STRIP
  - BString.cpp, 400
- subMilliseconds
  - BTimeStampMs, 305
- subSeconds
  - BTimeStampMs, 306
- subString
  - BString, 271
- swap
  - BList< T >, 146
- table\_crc\_hi
  - BCrc16.cpp, 323
- table\_crc\_lo
  - BCrc16.cpp, 323
- take
  - BSemaphoreCount, 239
- taskFunc
  - BTask, 278
- THREADED
  - BoapServer, 202
- timedLock
  - BMutex, 149
- timedWait
  - BCond, 42
  - BCondBool, 44
  - BSema, 233
- timeoutTicks
  - BTypes.h, 424
- to\_hex
  - BString.h, 408
- toBDictStringFromJson
  - BObjStringFormat.cpp, 380
  - BObjStringFormat.h, 387
- toBString
  - BDate-1.cpp, 326
  - BDate.cpp, 327
  - BDate.h, 328
  - BDict.cpp, 335
  - BDict.h, 336
  - BObjStringFormat.cpp, 380–383
  - BObjStringFormat.h, 387–390
  - BString.cpp, 404, 405
  - BString.h, 409
  - BTimeStamp.cpp, 415
  - BTimeStamp.h, 416
- toBStringJson
  - BObjStringFormat.cpp, 383–386
  - BObjStringFormat.h, 390–393
- toLower
  - BString, 271
- toUpper
  - BString, 271
- tprintf
  - BDebug.h, 334
- transact

- BSpi, 254
- translateChar
  - BString, 272
- truncate
  - BFile, 126
  - BString, 272
- tryLock
  - BMutex, 149
- tryRdLock
  - BRWLock, 230
- tryWait
  - BSema, 233
- tryWrLock
  - BRWLock, 231
- Type
  - BErrorTime, 94
  - BMutex, 148
- type
  - BEvent, 97
  - BFirmwareInfo, 132
  - BoapPacketHead, 200
  - BObjMember, 219
- typeComp
  - BObjMember, 219
- typeName
  - BObjMember, 219
- UInt16
  - BoapSimple.h, 378
- UInt32
  - BoapSimple.h, 378
- UInt8
  - BoapSimple.h, 378
- unlock
  - BCondResource, 50
  - BMutex, 149
  - BMutexLock, 150
  - BRWLock, 231
- unmapCircularBuffer
  - BFifoCirc< Type >, 116
- update
  - BMysql, 153
- updateHead
  - BoapPacket, 199
- valid
  - Blter, 135
- validate
  - BoapMc1Comms, 169
  - BoapServerConnection, 208
- value
  - BCondBool, 44
  - BCondInt, 47
  - BCondValue, 52
  - BCondWrap, 55
  - BDictItem< Type >, 72
  - BSemaphoreBool, 237
  - BSemaphoreCount, 239
- ver0
  - BFirmware.h, 360
  - BFirmwareFileHeader, 130
  - BFirmwareInfo, 132
- ver1
  - BFirmware.h, 360
  - BFirmwareFileHeader, 131
  - BFirmwareInfo, 132
- ver2
  - BFirmware.h, 360
  - BFirmwareFileHeader, 131
  - BFirmwareInfo, 132
- ver3
  - BFirmware.h, 360
  - BFirmwareFileHeader, 131
- wait
  - BComms, 39
  - BCond, 42
  - BCondBool, 44
  - BRtc, 227
  - BRtcThreaded, 229
  - BSema, 233
  - BSemaphore, 235
  - BSemaphoreBool, 237
  - BSemaphoreCount, 239
- waitForCompletion
  - BTask, 278
  - BThread, 281
- waitLessThan
  - BCondInt, 47
  - BCondValue, 52
  - BCondWrap, 55
- waitLessThanOrEqual
  - BCondInt, 47
  - BCondValue, 52
  - BCondWrap, 55
- waitMoreThanOrEqual
  - BCondInt, 48
  - BCondValue, 53
  - BCondWrap, 56
- wild
  - BDir.cpp, 338
- wildString
  - BDir.cpp, 338
- wprintf
  - BDebug.h, 333
- write
  - BComms, 40
  - BConfig, 57
  - BEntryFile, 86
  - BEventPipe, 105
  - BFifo< Type >, 110
  - BFifoCirc< Type >, 116
  - BFile, 126
  - BFileData, 129
  - BQueue< T >, 223
- writeAvailable
  - BComms, 40
  - BEventPipe, 105



- BFifo< Type >, [110](#)
- BFifoCirc< Type >, [116](#)
- BQueue< T >, [224](#)
- writeAvailableChunk
  - BFifo< Type >, [110](#)
- writeBackup
  - BFifo< Type >, [110](#)
- writeChunks
  - BComms, [40](#)
- writeCsv
  - BFileCsv, [127](#)
- writeData
  - BBuffer, [28](#)
  - BFifo< Type >, [111](#)
  - BFifoCirc< Type >, [117](#)
- writeDone
  - BFifo< Type >, [111](#)
  - BFifoCirc< Type >, [117](#)
- writeList
  - BEntryFile, [86](#)
- writePos
  - BFifo< Type >, [111](#)
- writeString
  - BFile, [126](#)
- writeWaitAvailable
  - BFifoCirc< Type >, [117](#)
- wrLock
  - BRWLock, [231](#)
- yday
  - BDate, [64](#)
  - BTimeStamp, [297](#)
  - BTimeStampMs, [307](#)
- year
  - BDate, [64](#)
  - BTimeStamp, [297](#)
  - BTimeStampMs, [307](#)
- yearDays
  - BTime.cpp, [413](#)
  - BTimeUs.cpp, [417](#)
- yearIsLeap
  - BTime.cpp, [413](#)
  - BTimeUs.cpp, [417](#)