

cryomesh

Generated by Doxygen 1.7.6.1

Tue Mar 6 2012 06:20:34



# Contents

<b>1</b>	<b>Namespace Index</b>	<b>1</b>
1.1	Namespace List . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class Hierarchy . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class List . . . . .	5
<b>4</b>	<b>File Index</b>	<b>9</b>
4.1	File List . . . . .	9
<b>5</b>	<b>Namespace Documentation</b>	<b>13</b>
5.1	cryomesh Namespace Reference . . . . .	13
5.1.1	Detailed Description . . . . .	13
5.2	cryomesh::common Namespace Reference . . . . .	14
5.2.1	Function Documentation . . . . .	14
5.2.1.1	operator<< . . . . .	14
5.3	cryomesh::components Namespace Reference . . . . .	15
5.3.1	Function Documentation . . . . .	15
5.3.1.1	operator<< . . . . .	15
5.3.1.2	operator<< . . . . .	16
5.3.1.3	operator<< . . . . .	16
5.3.1.4	operator<< . . . . .	16
5.4	cryomesh::dataobjects Namespace Reference . . . . .	17
5.5	cryomesh::manager Namespace Reference . . . . .	17
5.6	cryomesh::manipulators Namespace Reference . . . . .	18

5.7	<a href="#">cryomesh::state Namespace Reference</a>	18
5.7.1	<a href="#">Function Documentation</a>	18
5.7.1.1	<a href="#">operator&lt;&lt;</a>	18
5.7.1.2	<a href="#">operator&lt;&lt;</a>	18
5.7.1.3	<a href="#">operator&lt;&lt;</a>	19
5.7.1.4	<a href="#">operator&lt;&lt;</a>	19
5.8	<a href="#">cryomesh::structures Namespace Reference</a>	19
5.8.1	<a href="#">Typedef Documentation</a>	20
5.8.1.1	<a href="#">NeighbourhoodMap</a>	20
5.8.1.2	<a href="#">NeighbourhoodMapConstIterator</a>	20
5.8.2	<a href="#">Function Documentation</a>	20
5.8.2.1	<a href="#">operator&lt;&lt;</a>	20
5.8.2.2	<a href="#">operator&lt;&lt;</a>	20
5.8.2.3	<a href="#">operator&lt;&lt;</a>	21
5.8.2.4	<a href="#">operator&lt;&lt;</a>	21
5.9	<a href="#">cryomesh::utilities Namespace Reference</a>	22
5.9.1	<a href="#">Function Documentation</a>	22
5.9.1.1	<a href="#">operator&lt;&lt;</a>	22
5.9.1.2	<a href="#">operator&lt;&lt;</a>	22
<b>6</b>	<b><a href="#">Class Documentation</a></b>	<b>23</b>
6.1	<a href="#">cryomesh::state::ActivityPattern Class Reference</a>	23
6.1.1	<a href="#">Detailed Description</a>	23
6.1.2	<a href="#">Member Enumeration Documentation</a>	24
6.1.2.1	<a href="#">ActivityComparison</a>	24
6.1.3	<a href="#">Constructor &amp; Destructor Documentation</a>	24
6.1.3.1	<a href="#">ActivityPattern</a>	24
6.1.3.2	<a href="#">~ActivityPattern</a>	24
6.1.4	<a href="#">Member Function Documentation</a>	24
6.1.4.1	<a href="#">toBooleanVector</a>	24
6.1.4.2	<a href="#">toPlusBooleanList</a>	24
6.1.4.3	<a href="#">toPlusBooleanString</a>	25
6.2	<a href="#">cryomesh::components::ActivityTimer Class Reference</a>	25
6.2.1	<a href="#">Detailed Description</a>	26

6.2.2	Constructor & Destructor Documentation . . . . .	26
6.2.2.1	ActivityTimer . . . . .	26
6.2.2.2	~ActivityTimer . . . . .	26
6.2.3	Member Function Documentation . . . . .	26
6.2.3.1	print . . . . .	26
6.2.3.2	reset . . . . .	26
6.2.4	Friends And Related Function Documentation . . . . .	26
6.2.4.1	operator<< . . . . .	26
6.3	cryomesh::components::ActivityTimerDistance Class Reference . . . . .	27
6.3.1	Detailed Description . . . . .	28
6.3.2	Constructor & Destructor Documentation . . . . .	29
6.3.2.1	ActivityTimerDistance . . . . .	29
6.3.2.2	ActivityTimerDistance . . . . .	29
6.3.2.3	~ActivityTimerDistance . . . . .	29
6.3.3	Member Function Documentation . . . . .	29
6.3.3.1	checkConstraints . . . . .	29
6.3.3.2	enableDebug . . . . .	29
6.3.3.3	getDecrement . . . . .	29
6.3.3.4	getDelay . . . . .	30
6.3.3.5	getRandom . . . . .	30
6.3.3.6	getStartingDelay . . . . .	30
6.3.3.7	operator+ . . . . .	31
6.3.3.8	operator+= . . . . .	31
6.3.3.9	operator-- . . . . .	31
6.3.3.10	operator-- . . . . .	31
6.3.3.11	operator< . . . . .	32
6.3.3.12	operator= . . . . .	32
6.3.3.13	operator> . . . . .	32
6.3.3.14	print . . . . .	33
6.3.3.15	reset . . . . .	33
6.3.4	Friends And Related Function Documentation . . . . .	33
6.3.4.1	operator<< . . . . .	33
6.3.5	Member Data Documentation . . . . .	33
6.3.5.1	decrement . . . . .	34

6.3.5.2	<a href="#">distance</a>	34
6.3.5.3	<a href="#">distance_remaining</a>	34
6.3.5.4	<a href="#">MAX_DECREMENT_FRACTION</a>	34
6.3.5.5	<a href="#">MAX_DISTANCE</a>	34
6.3.5.6	<a href="#">MIN_DECREMENT_FRACTION</a>	34
6.3.5.7	<a href="#">MIN_DISTANCE</a>	34
6.4	<a href="#">cryomesh::state::BinaryString Class Reference</a>	35
6.4.1	<a href="#">Detailed Description</a>	36
6.4.2	<a href="#">Member Enumeration Documentation</a>	36
6.4.2.1	<a href="#">Type</a>	36
6.4.3	<a href="#">Constructor &amp; Destructor Documentation</a>	37
6.4.3.1	<a href="#">BinaryString</a>	37
6.4.3.2	<a href="#">BinaryString</a>	37
6.4.3.3	<a href="#">BinaryString</a>	37
6.4.3.4	<a href="#">BinaryString</a>	37
6.4.3.5	<a href="#">~BinaryString</a>	37
6.4.4	<a href="#">Member Function Documentation</a>	37
6.4.4.1	<a href="#">charToBinaryString</a>	37
6.4.4.2	<a href="#">formatBinaryStringsToText</a>	37
6.4.4.3	<a href="#">formatIntsToText</a>	38
6.4.4.4	<a href="#">formatTextToBinaryStrings</a>	38
6.4.4.5	<a href="#">formatTextToInts</a>	38
6.4.4.6	<a href="#">getBinaryString</a>	38
6.4.4.7	<a href="#">getBools</a>	38
6.4.4.8	<a href="#">getSignBit</a>	38
6.4.4.9	<a href="#">getWidth</a>	38
6.4.4.10	<a href="#">isAllZeroes</a>	39
6.4.4.11	<a href="#">isValidBinary</a>	39
6.4.4.12	<a href="#">resize</a>	39
6.4.4.13	<a href="#">serialize</a>	39
6.4.4.14	<a href="#">setBinaryString</a>	39
6.4.4.15	<a href="#">setBinaryString</a>	39
6.4.4.16	<a href="#">setSignBit</a>	40
6.4.4.17	<a href="#">toInt</a>	40

6.4.4.18	<a href="#">toInts</a>	40
6.4.4.19	<a href="#">toText</a>	40
6.4.5	<a href="#">Friends And Related Function Documentation</a>	40
6.4.5.1	<a href="#">boost::serialization::access</a>	40
6.4.5.2	<a href="#">operator&lt;&lt;</a>	40
6.4.6	<a href="#">Member Data Documentation</a>	40
6.4.6.1	<a href="#">BINARY_CHAR_LENGTH</a>	40
6.4.6.2	<a href="#">binaryString</a>	41
6.4.6.3	<a href="#">MAX_BINARY_INTEGER_SIZE</a>	41
6.4.6.4	<a href="#">signBit</a>	41
6.5	<a href="#">cryomesh::structures::Bundle Class Reference</a>	41
6.5.1	<a href="#">Detailed Description</a>	45
6.5.2	<a href="#">Member Enumeration Documentation</a>	45
6.5.2.1	<a href="#">LoggingDepth</a>	45
6.5.3	<a href="#">Constructor &amp; Destructor Documentation</a>	46
6.5.3.1	<a href="#">Bundle</a>	46
6.5.3.2	<a href="#">~Bundle</a>	46
6.5.4	<a href="#">Member Function Documentation</a>	46
6.5.4.1	<a href="#">autoConnectPrimaryInputClusters</a>	46
6.5.4.2	<a href="#">autoConnectPrimaryInputClusters</a>	46
6.5.4.3	<a href="#">autoConnectPrimaryOutputClusters</a>	46
6.5.4.4	<a href="#">autoConnectPrimaryOutputClusters</a>	47
6.5.4.5	<a href="#">checkChannelStructure</a>	47
6.5.4.6	<a href="#">checkFibreStructure</a>	47
6.5.4.7	<a href="#">checkStructure</a>	47
6.5.4.8	<a href="#">connectCluster</a>	48
6.5.4.9	<a href="#">connectCluster</a>	48
6.5.4.10	<a href="#">connectLoopbackCluster</a>	49
6.5.4.11	<a href="#">connectPrimaryInputCluster</a>	49
6.5.4.12	<a href="#">connectPrimaryOutputCluster</a>	49
6.5.4.13	<a href="#">createCluster</a>	50
6.5.4.14	<a href="#">enableDebug</a>	50
6.5.4.15	<a href="#">enableDebugClusters</a>	50
6.5.4.16	<a href="#">enableDebugFibres</a>	51

6.5.4.17	<a href="#">getActualFibrePatternChannelMap</a>	51
6.5.4.18	<a href="#">getActualInputChannelsMap</a>	51
6.5.4.19	<a href="#">getActualOutputChannelsMap</a>	51
6.5.4.20	<a href="#">getActualPrimaryInputChannelByFibre</a>	51
6.5.4.21	<a href="#">getActualPrimaryOutputChannelByFibre</a>	52
6.5.4.22	<a href="#">getClusters</a>	52
6.5.4.23	<a href="#">getDisconnectedRealInputPatternChannels</a>	52
6.5.4.24	<a href="#">getDisconnectedRealOutputPatternChannels</a>	52
6.5.4.25	<a href="#">getEnergy</a>	53
6.5.4.26	<a href="#">getFibres</a>	53
6.5.4.27	<a href="#">getInputFibres</a>	53
6.5.4.28	<a href="#">getMutableClusters</a>	53
6.5.4.29	<a href="#">getMutableInputFibres</a>	54
6.5.4.30	<a href="#">getMutableOutputFibres</a>	54
6.5.4.31	<a href="#">getMutableStatistician</a>	54
6.5.4.32	<a href="#">getOutputFibres</a>	54
6.5.4.33	<a href="#">getPrimaryChannelByFibre</a>	55
6.5.4.34	<a href="#">getPrimaryFibreByChannel</a>	55
6.5.4.35	<a href="#">getPrimaryInputFibreByActualChannel</a>	56
6.5.4.36	<a href="#">getPrimaryInputFibreByRealChannel</a>	56
6.5.4.37	<a href="#">getPrimaryOutputFibreByActualChannel</a>	56
6.5.4.38	<a href="#">getPrimaryOutputFibreByRealChannel</a>	56
6.5.4.39	<a href="#">getRealFibrePatternChannelMap</a>	57
6.5.4.40	<a href="#">getRealInputChannelsMap</a>	57
6.5.4.41	<a href="#">getRealOutputChannelsMap</a>	57
6.5.4.42	<a href="#">getRealPrimaryInputChannelByFibre</a>	58
6.5.4.43	<a href="#">getRealPrimaryOutputChannelByFibre</a>	58
6.5.4.44	<a href="#">getStatistician</a>	58
6.5.4.45	<a href="#">loadChannels</a>	58
6.5.4.46	<a href="#">matchOutputChannelsSum</a>	59
6.5.4.47	<a href="#">print</a>	59
6.5.4.48	<a href="#">printChannels</a>	59
6.5.4.49	<a href="#">printFibreMap</a>	60
6.5.4.50	<a href="#">printFibreMaps</a>	60



6.5.4.51	<a href="#">printSearch</a>	60
6.5.4.52	<a href="#">setEnergy</a>	60
6.5.4.53	<a href="#">update</a>	60
6.5.4.54	<a href="#">updatePrimaryInputFibres</a>	61
6.5.4.55	<a href="#">updatePrimaryOutputFibres</a>	61
6.5.4.56	<a href="#">updateStatistician</a>	61
6.5.5	<a href="#">Friends And Related Function Documentation</a>	61
6.5.5.1	<a href="#">operator&lt;&lt;</a>	61
6.5.6	<a href="#">Member Data Documentation</a>	62
6.5.6.1	<a href="#">actualFibrePatternChannelMap</a>	62
6.5.6.2	<a href="#">actualInputChannelsMap</a>	62
6.5.6.3	<a href="#">actualOutputChannelsMap</a>	62
6.5.6.4	<a href="#">clusters</a>	62
6.5.6.5	<a href="#">energy</a>	62
6.5.6.6	<a href="#">energyFractionMethod</a>	62
6.5.6.7	<a href="#">fibres</a>	63
6.5.6.8	<a href="#">inputFibres</a>	63
6.5.6.9	<a href="#">outputFibres</a>	63
6.5.6.10	<a href="#">realFibrePatternChannelMap</a>	63
6.5.6.11	<a href="#">realInputChannelsMap</a>	63
6.5.6.12	<a href="#">realOutputChannelsMap</a>	63
6.5.6.13	<a href="#">statistician</a>	64
6.6	<a href="#">cryomesh::structures::Cluster Class Reference</a>	64
6.6.1	<a href="#">Detailed Description</a>	66
6.6.2	<a href="#">Member Enumeration Documentation</a>	66
6.6.2.1	<a href="#">EnergyFractionMethod</a>	66
6.6.2.2	<a href="#">ValueTypeSpecifier</a>	67
6.6.3	<a href="#">Constructor &amp; Destructor Documentation</a>	67
6.6.3.1	<a href="#">Cluster</a>	67
6.6.3.2	<a href="#">Cluster</a>	67
6.6.3.3	<a href="#">Cluster</a>	67
6.6.3.4	<a href="#">~Cluster</a>	67
6.6.4	<a href="#">Member Function Documentation</a>	68
6.6.4.1	<a href="#">createConnectivity</a>	68

6.6.4.2	<a href="#">enableDebug</a>	68
6.6.4.3	<a href="#">getActiveNodeCount</a>	68
6.6.4.4	<a href="#">getClusterArchitect</a>	68
6.6.4.5	<a href="#">getConnectionMap</a>	68
6.6.4.6	<a href="#">getConnections</a>	69
6.6.4.7	<a href="#">getConnector</a>	69
6.6.4.8	<a href="#">getEnergy</a>	69
6.6.4.9	<a href="#">getEnergyFractionMethod</a>	69
6.6.4.10	<a href="#">getLiveNodeCount</a>	70
6.6.4.11	<a href="#">getMaxEnergyFraction</a>	70
6.6.4.12	<a href="#">getMesh</a>	70
6.6.4.13	<a href="#">getMutableClusterArchitect</a>	70
6.6.4.14	<a href="#">getMutableConnectionMap</a>	70
6.6.4.15	<a href="#">getMutableConnector</a>	71
6.6.4.16	<a href="#">getMutableMesh</a>	71
6.6.4.17	<a href="#">getMutableNodeMap</a>	71
6.6.4.18	<a href="#">getNodeMap</a>	71
6.6.4.19	<a href="#">getNodes</a>	72
6.6.4.20	<a href="#">getTriggeredNodeCount</a>	72
6.6.4.21	<a href="#">runAnalyser</a>	72
6.6.4.22	<a href="#">setEnergy</a>	73
6.6.4.23	<a href="#">setEnergyFractionMethod</a>	73
6.6.4.24	<a href="#">update</a>	73
6.6.4.25	<a href="#">updateConnectivity</a>	73
6.6.4.26	<a href="#">updateEnergy</a>	73
6.6.4.27	<a href="#">warpNodes</a>	73
6.6.5	<a href="#">Friends And Related Function Documentation</a>	73
6.6.5.1	<a href="#">operator&lt;&lt;</a>	74
6.6.6	<a href="#">Member Data Documentation</a>	74
6.6.6.1	<a href="#">clusterArchitect</a>	74
6.6.6.2	<a href="#">connections</a>	74
6.6.6.3	<a href="#">connector</a>	74
6.6.6.4	<a href="#">energy</a>	74
6.6.6.5	<a href="#">energyFractionMethod</a>	75

6.6.6.6	<a href="#">maxEnergyFraction</a>	75
6.6.6.7	<a href="#">mesh</a>	75
6.6.6.8	<a href="#">nodes</a>	75
6.6.6.9	<a href="#">SELF_CONNECTED_NODES_FRACTION</a>	75
6.7	<a href="#">cryomesh::manipulators::ClusterAnalyserBasic Class Reference</a>	75
6.7.1	<a href="#">Detailed Description</a>	77
6.7.2	<a href="#">Member Enumeration Documentation</a>	77
6.7.2.1	<a href="#">EnergyVariation</a>	77
6.7.3	<a href="#">Constructor &amp; Destructor Documentation</a>	77
6.7.3.1	<a href="#">ClusterAnalyserBasic</a>	77
6.7.3.2	<a href="#">~ClusterAnalyserBasic</a>	77
6.7.4	<a href="#">Member Function Documentation</a>	78
6.7.4.1	<a href="#">analyseCluster</a>	78
6.7.4.2	<a href="#">calculateRangeEnergies</a>	78
6.7.4.3	<a href="#">getConnectionCreationEnabledCountdown</a>	79
6.7.4.4	<a href="#">getConnectionDestructionEnabledCountdown</a>	79
6.7.4.5	<a href="#">getConnectionRestructuring</a>	79
6.7.4.6	<a href="#">getEnergyVariationMap</a>	79
6.7.4.7	<a href="#">getNodeCreationEnabledCountdown</a>	80
6.7.4.8	<a href="#">getNodeDestructionEnabledCountdown</a>	80
6.7.4.9	<a href="#">getNodeRestructuring</a>	80
6.7.4.10	<a href="#">setConnectionDestructionEnabledCountdown</a>	80
6.7.5	<a href="#">Member Data Documentation</a>	80
6.7.5.1	<a href="#">clusterArchitect</a>	80
6.7.5.2	<a href="#">connectionRestructuring</a>	80
6.7.5.3	<a href="#">nodeRestructuring</a>	81
6.8	<a href="#">cryomesh::manipulators::ClusterAnalysisData Class Reference</a>	81
6.8.1	<a href="#">Detailed Description</a>	82
6.8.2	<a href="#">Constructor &amp; Destructor Documentation</a>	82
6.8.2.1	<a href="#">ClusterAnalysisData</a>	82
6.8.2.2	<a href="#">ClusterAnalysisData</a>	82
6.8.2.3	<a href="#">ClusterAnalysisData</a>	83
6.8.2.4	<a href="#">ClusterAnalysisData</a>	83
6.8.2.5	<a href="#">~ClusterAnalysisData</a>	83

6.8.3	Member Function Documentation	83
6.8.3.1	getClusterRangeEnergy	83
6.8.3.2	getConnectionCreationWeight	83
6.8.3.3	getConnectionDestructionWeight	83
6.8.3.4	getConnectionsToCreate	84
6.8.3.5	getConnectionsToDestroy	84
6.8.3.6	getNodeCreationWeight	84
6.8.3.7	getNodeDestructionWeight	84
6.8.3.8	getNodesToCreate	84
6.8.3.9	getNodesToDestroy	85
6.8.3.10	setClusterRangeEnergy	85
6.8.3.11	setConnectionCreationWeight	85
6.8.3.12	setConnectionDestructionWeight	85
6.8.3.13	setConnectionsToCreate	85
6.8.3.14	setConnectionsToDestroy	86
6.8.3.15	setNodeCreationWeight	86
6.8.3.16	setNodeDestructionWeight	86
6.8.3.17	setNodesToCreate	86
6.8.3.18	setNodesToDestroy	86
6.8.4	Friends And Related Function Documentation	86
6.8.4.1	operator<<	86
6.8.5	Member Data Documentation	87
6.8.5.1	clusterRangeEnergy	87
6.8.5.2	connectionCreationWeight	87
6.8.5.3	connectionDestructionWeight	87
6.8.5.4	connectionsToCreate	87
6.8.5.5	connectionsToDestroy	87
6.8.5.6	nodeCreationWeight	87
6.8.5.7	nodeDestructionWeight	88
6.8.5.8	nodesToCreate	88
6.8.5.9	nodesToDestroy	88
6.9	cryomesh::manipulators::ClusterArchitect Class Reference	88
6.9.1	Detailed Description	90
6.9.2	Member Enumeration Documentation	90

6.9.2.1	ConnectionStrategy	90
6.9.3	Constructor & Destructor Documentation	91
6.9.3.1	ClusterArchitect	91
6.9.3.2	~ClusterArchitect	91
6.9.4	Member Function Documentation	91
6.9.4.1	addHistoryEntry	91
6.9.4.2	createConnection	91
6.9.4.3	createRandomConnections	91
6.9.4.4	createRandomNodes	92
6.9.4.5	deleteConnection	92
6.9.4.6	destroyRandomConnections	92
6.9.4.7	destroyRandomNodes	92
6.9.4.8	getCluster	93
6.9.4.9	getClusterAnalyser	93
6.9.4.10	getCurrentClusterAnalysisData	93
6.9.4.11	getCurrentHistory	93
6.9.4.12	getHistories	93
6.9.4.13	getHistoryEntriesInRange	94
6.9.4.14	getHistorySteppingFactor	94
6.9.4.15	getMaxHistorySize	94
6.9.4.16	getRandomConnections	94
6.9.4.17	getRandomNodes	95
6.9.4.18	printAllHistory	95
6.9.4.19	reduceContainerSize	96
6.9.4.20	runAnalysis	96
6.9.4.21	setClusterAnalyser	96
6.9.4.22	setCurrentClusterAnalysisData	96
6.9.4.23	setCurrentHistory	96
6.9.4.24	setHistories	96
6.9.4.25	setMaxHistorySize	97
6.9.4.26	splitHistoryByValue	97
6.9.5	Member Data Documentation	97
6.9.5.1	cluster	97
6.9.5.2	clusterAnalyser	97

6.9.5.3	<a href="#">currentClusterAnalysisData</a>	97
6.9.5.4	<a href="#">currentHistory</a>	97
6.9.5.5	<a href="#">DEFAULT_CONNECTIVITY_FRACTION</a>	98
6.9.5.6	<a href="#">DEFAULT_HISTORY_STEPPING_FACTOR</a>	98
6.9.5.7	<a href="#">DEFAULT_MAX_HISTORY_SIZE</a>	98
6.9.5.8	<a href="#">histories</a>	98
6.9.5.9	<a href="#">historiesNewEntries</a>	98
6.9.5.10	<a href="#">historySteppingFactor</a>	99
6.9.5.11	<a href="#">maxHistorySize</a>	99
6.10	<a href="#">cryomesh::components::Connection Class Reference</a>	99
6.10.1	<a href="#">Detailed Description</a>	100
6.10.2	<a href="#">Constructor &amp; Destructor Documentation</a>	101
6.10.2.1	<a href="#">Connection</a>	101
6.10.2.2	<a href="#">~Connection</a>	101
6.10.3	<a href="#">Member Function Documentation</a>	101
6.10.3.1	<a href="#">add</a>	101
6.10.3.2	<a href="#">connectInput</a>	102
6.10.3.3	<a href="#">connectOutput</a>	102
6.10.3.4	<a href="#">disconnect</a>	102
6.10.3.5	<a href="#">disconnectInput</a>	102
6.10.3.6	<a href="#">disconnectOutput</a>	102
6.10.3.7	<a href="#">enableDebug</a>	102
6.10.3.8	<a href="#">getActivityTimer</a>	102
6.10.3.9	<a href="#">getConnector</a>	103
6.10.3.10	<a href="#">getDatabaseObject</a>	103
6.10.3.11	<a href="#">getImpulses</a>	103
6.10.3.12	<a href="#">getMutableActivityTimer</a>	104
6.10.3.13	<a href="#">getMutableConnector</a>	104
6.10.3.14	<a href="#">getMutableImpulses</a>	104
6.10.3.15	<a href="#">isPrimaryInputConnection</a>	105
6.10.3.16	<a href="#">isPrimaryOutputConnection</a>	105
6.10.3.17	<a href="#">remove</a>	105
6.10.3.18	<a href="#">remove</a>	106
6.10.3.19	<a href="#">update</a>	106

6.10.3.20	updatePosition	106
6.10.4	Friends And Related Function Documentation	106
6.10.4.1	operator<<	106
6.10.5	Member Data Documentation	107
6.10.5.1	activityTimer	107
6.10.5.2	connector	107
6.10.5.3	impulses	107
6.11	cryomesh::manager::ConnectionDatabaseObject Class Reference	107
6.11.1	Detailed Description	109
6.11.2	Constructor & Destructor Documentation	109
6.11.2.1	ConnectionDatabaseObject	109
6.11.2.2	ConnectionDatabaseObject	109
6.11.2.3	~ConnectionDatabaseObject	110
6.11.3	Member Function Documentation	110
6.11.3.1	findValue	110
6.11.3.2	getColumnMapFromEntry	110
6.11.3.3	getCycle	111
6.11.3.4	getImpulseCount	111
6.11.3.5	getInputNodeUUID	111
6.11.3.6	getInsert	111
6.11.3.7	getKey	112
6.11.3.8	getOutputNodeUUID	112
6.11.3.9	getUUID	112
6.11.3.10	toString	113
6.11.4	Member Data Documentation	113
6.11.4.1	columns	113
6.11.4.2	cycle	113
6.11.4.3	CYCLE_TAG	113
6.11.4.4	ID_TAG	113
6.11.4.5	IMPULSE_COUNT_TAG	114
6.11.4.6	impulseCount	114
6.11.4.7	INPUT_ID_TAG	114
6.11.4.8	inputNodeUUID	114
6.11.4.9	OUTPUT_ID_TAG	114

6.11.4.10	outputNodeUUID	114
6.11.4.11	uuid	114
6.12	cryomesh::components::ConnectionMap Class Reference	115
6.12.1	Detailed Description	115
6.12.2	Constructor & Destructor Documentation	116
6.12.2.1	ConnectionMap	116
6.12.2.2	~ConnectionMap	116
6.12.3	Member Function Documentation	116
6.12.3.1	getActivityPattern	116
6.12.3.2	getActivityPattern	116
6.12.3.3	getAllPrimaryInputConnections	116
6.12.3.4	getAllPrimaryOutputConnections	117
6.12.3.5	update	117
6.12.4	Friends And Related Function Documentation	117
6.12.4.1	operator<<	117
6.13	cryomesh::manager::ConnectionTableFormat Struct Reference	117
6.13.1	Detailed Description	118
6.13.2	Constructor & Destructor Documentation	118
6.13.2.1	ConnectionTableFormat	118
6.13.3	Member Function Documentation	119
6.13.3.1	getTable	119
6.13.3.2	getKey	119
6.13.3.3	getName	119
6.13.4	Member Data Documentation	120
6.13.4.1	columns	120
6.13.4.2	name	120
6.14	cryomesh::common::Connector< U, T > Class Template Reference	120
6.14.1	Detailed Description	122
6.14.2	Constructor & Destructor Documentation	123
6.14.2.1	Connector	123
6.14.2.2	~Connector	123
6.14.3	Member Function Documentation	123
6.14.3.1	connect	123
6.14.3.2	connectInput	123



6.14.3.3	<a href="#">connectInputs</a>	124
6.14.3.4	<a href="#">connectInputs</a>	124
6.14.3.5	<a href="#">connectOutput</a>	125
6.14.3.6	<a href="#">connectOutputs</a>	125
6.14.3.7	<a href="#">connectOutputs</a>	126
6.14.3.8	<a href="#">disconnect</a>	126
6.14.3.9	<a href="#">disconnect</a>	127
6.14.3.10	<a href="#">disconnectAllInputs</a>	127
6.14.3.11	<a href="#">disconnectAllOutputs</a>	127
6.14.3.12	<a href="#">disconnectInput</a>	128
6.14.3.13	<a href="#">disconnectInput</a>	128
6.14.3.14	<a href="#">disconnectInputs</a>	128
6.14.3.15	<a href="#">disconnectInputs</a>	129
6.14.3.16	<a href="#">disconnectInputs</a>	129
6.14.3.17	<a href="#">disconnectInputs</a>	130
6.14.3.18	<a href="#">disconnectOutput</a>	130
6.14.3.19	<a href="#">disconnectOutput</a>	130
6.14.3.20	<a href="#">disconnectOutputs</a>	131
6.14.3.21	<a href="#">disconnectOutputs</a>	131
6.14.3.22	<a href="#">disconnectOutputs</a>	132
6.14.3.23	<a href="#">disconnectOutputs</a>	132
6.14.3.24	<a href="#">getInputs</a>	132
6.14.3.25	<a href="#">getInputsUUID</a>	133
6.14.3.26	<a href="#">getMutableInputs</a>	133
6.14.3.27	<a href="#">getMutableOutputs</a>	133
6.14.3.28	<a href="#">getOutputs</a>	133
6.14.3.29	<a href="#">getOutputsUUID</a>	134
6.14.4	<a href="#">Friends And Related Function Documentation</a>	134
6.14.4.1	<a href="#">operator&lt;&lt;</a>	134
6.14.5	<a href="#">Member Data Documentation</a>	134
6.14.5.1	<a href="#">maxInputs</a>	134
6.14.5.2	<a href="#">maxOutputs</a>	134
6.14.5.3	<a href="#">mininputs</a>	135
6.14.5.4	<a href="#">moutputs</a>	135

6.15 cryomesh::manager::Creator Class Reference . . . . .	135
6.15.1 Detailed Description . . . . .	137
6.15.2 Constructor & Destructor Documentation . . . . .	138
6.15.2.1 Creator . . . . .	138
6.15.2.2 Creator . . . . .	138
6.15.2.3 ~Creator . . . . .	138
6.15.3 Member Function Documentation . . . . .	138
6.15.3.1 analyseConfig . . . . .	138
6.15.3.2 autoConnectPrimaryInputs . . . . .	139
6.15.3.3 autoConnectPrimaryOutputs . . . . .	139
6.15.3.4 checkConfigEntry . . . . .	139
6.15.3.5 checkConfigStructure . . . . .	140
6.15.3.6 connectCluster . . . . .	140
6.15.3.7 connectPrimaryInputChannel . . . . .	140
6.15.3.8 connectPrimaryOutputChannel . . . . .	141
6.15.3.9 createCluster . . . . .	141
6.15.3.10 createFromConfigFile . . . . .	141
6.15.3.11 createFromConfigStream . . . . .	142
6.15.3.12 getAcceptedCommandList . . . . .	142
6.15.3.13 getBundle . . . . .	142
6.15.3.14 getClusterIDMap . . . . .	142
6.15.3.15 getClusterRealID . . . . .	143
6.15.3.16 getFibreIDMap . . . . .	143
6.15.3.17 getFibreRealID . . . . .	143
6.15.3.18 getMutableBundle . . . . .	144
6.15.3.19 getPatternChannelIDMap . . . . .	144
6.15.3.20 getPatternChannelRealID . . . . .	144
6.15.3.21 getRealID . . . . .	145
6.15.3.22 initialise . . . . .	145
6.15.3.23 loadData . . . . .	145
6.15.3.24 runCommand . . . . .	146
6.15.4 Member Data Documentation . . . . .	146
6.15.4.1 acceptedCommandList . . . . .	146
6.15.4.2 bundle . . . . .	146

6.15.4.3	clusterIDMap	146
6.15.4.4	databaseFilename	146
6.15.4.5	DEFAULT_DATABASE_FILENAME	146
6.15.4.6	fibreIDMap	147
6.15.4.7	patternChannelIDMap	147
6.16	cryomesh::common::Cycle Class Reference	147
6.16.1	Detailed Description	148
6.16.2	Constructor & Destructor Documentation	148
6.16.2.1	Cycle	148
6.16.2.2	Cycle	149
6.16.2.3	Cycle	149
6.16.3	Member Function Documentation	149
6.16.3.1	getMP	149
6.16.3.2	operator!=	149
6.16.3.3	operator+	150
6.16.3.4	operator++	150
6.16.3.5	operator++	150
6.16.3.6	operator+=	150
6.16.3.7	operator-	151
6.16.3.8	operator--	151
6.16.3.9	operator--	151
6.16.3.10	operator-=	151
6.16.3.11	operator<	152
6.16.3.12	operator<=	152
6.16.3.13	operator=	152
6.16.3.14	operator==	153
6.16.3.15	operator>	153
6.16.3.16	operator>=	153
6.16.3.17	toLInt	154
6.16.3.18	toULInt	154
6.16.4	Friends And Related Function Documentation	155
6.16.4.1	operator<<	155
6.16.5	Member Data Documentation	155
6.16.5.1	cycle	155

6.17 cryomesh::manager::DatabaseManager Class Reference . . . . .	155
6.17.1 Detailed Description . . . . .	159
6.17.2 Constructor & Destructor Documentation . . . . .	159
6.17.2.1 DatabaseManager . . . . .	159
6.17.2.2 DatabaseManager . . . . .	159
6.17.2.3 ~DatabaseManager . . . . .	159
6.17.3 Member Function Documentation . . . . .	159
6.17.3.1 addHistoryEntry . . . . .	160
6.17.3.2 addHistoryEntry . . . . .	160
6.17.3.3 clearTable . . . . .	161
6.17.3.4 clearTables . . . . .	161
6.17.3.5 countConnections . . . . .	161
6.17.3.6 countNodes . . . . .	161
6.17.3.7 countRows . . . . .	162
6.17.3.8 createTables . . . . .	162
6.17.3.9 databaseCallback . . . . .	162
6.17.3.10 deleteAll . . . . .	163
6.17.3.11 deleteAllByCycle . . . . .	163
6.17.3.12 deleteByCycle . . . . .	163
6.17.3.13 deleteConnection . . . . .	164
6.17.3.14 deleteConnections . . . . .	164
6.17.3.15 deleteConnectionsByCycle . . . . .	164
6.17.3.16 deleteNode . . . . .	165
6.17.3.17 deleteNodes . . . . .	165
6.17.3.18 deleteNodesByCycle . . . . .	166
6.17.3.19 deleteOutputPattern . . . . .	166
6.17.3.20 deleteOutputPatterns . . . . .	166
6.17.3.21 deleteSelected . . . . .	166
6.17.3.22 dropTable . . . . .	167
6.17.3.23 insertConnection . . . . .	167
6.17.3.24 insertNode . . . . .	167
6.17.3.25 insertOutputPattern . . . . .	168
6.17.3.26 isDatabaseAccessible . . . . .	168
6.17.3.27 operator= . . . . .	168

6.17.3.28	<a href="#">printHistory</a>	169
6.17.3.29	<a href="#">printHistory</a>	169
6.17.3.30	<a href="#">select</a>	169
6.17.3.31	<a href="#">selectConnection</a>	170
6.17.3.32	<a href="#">selectConnections</a>	170
6.17.3.33	<a href="#">selectConnectionValue</a>	170
6.17.3.34	<a href="#">selectNode</a>	171
6.17.3.35	<a href="#">selectNodes</a>	171
6.17.3.36	<a href="#">selectNodeValue</a>	172
6.17.3.37	<a href="#">selectOutputPattern</a>	172
6.17.3.38	<a href="#">selectOutputPatterns</a>	172
6.17.3.39	<a href="#">selectOutputPatternValue</a>	172
6.17.3.40	<a href="#">selectValue</a>	172
6.17.3.41	<a href="#">sqlCommand</a>	173
6.17.3.42	<a href="#">sqlCommandBySelection</a>	173
6.17.3.43	<a href="#">updateByUUID</a>	174
6.17.3.44	<a href="#">updateConnection</a>	174
6.17.3.45	<a href="#">updateNode</a>	174
6.17.4	<a href="#">Member Data Documentation</a>	175
6.17.4.1	<a href="#">CONNECTIONS_TABLE_FORMAT</a>	175
6.17.4.2	<a href="#">database</a>	175
6.17.4.3	<a href="#">databaseAccess</a>	175
6.17.4.4	<a href="#">DEFAULT_DATABASE</a>	175
6.17.4.5	<a href="#">DEFAULT_DATABASE_PATH</a>	176
6.17.4.6	<a href="#">errorCode</a>	176
6.17.4.7	<a href="#">errorMessage</a>	176
6.17.4.8	<a href="#">MAX_COMMAND_HISTORY</a>	176
6.17.4.9	<a href="#">NODES_TABLE_FORMAT</a>	176
6.17.4.10	<a href="#">OUTPUT_PATTERNS_TABLE_FORMAT</a>	176
6.17.4.11	<a href="#">sqlResults</a>	176
6.17.4.12	<a href="#">sqlResultsBuffer</a>	177
6.18	<a href="#">cryomesh::manager::DatabaseObject Class Reference</a>	177
6.18.1	<a href="#">Detailed Description</a>	178
6.18.2	<a href="#">Constructor &amp; Destructor Documentation</a>	178

6.18.2.1	DatabaseObject	178
6.18.2.2	~DatabaseObject	178
6.18.3	Member Function Documentation	178
6.18.3.1	findValue	178
6.18.3.2	getColumnMapFromEntry	179
6.18.3.3	getInsert	179
6.18.3.4	getKey	179
6.18.3.5	toString	179
6.18.4	Member Data Documentation	180
6.18.4.1	columns	180
6.19	cryomesh::dataobjects::DataObject< U, T > Class Template Reference	180
6.19.1	Detailed Description	182
6.19.2	Member Enumeration Documentation	182
6.19.2.1	ComparisonType	182
6.19.3	Constructor & Destructor Documentation	182
6.19.3.1	DataObject	182
6.19.3.2	DataObject	182
6.19.3.3	~DataObject	183
6.19.4	Member Function Documentation	183
6.19.4.1	clear	183
6.19.4.2	enableLogging	183
6.19.4.3	getAverageValue	183
6.19.4.4	getByKey	183
6.19.4.5	getDatasetMaximumSize	184
6.19.4.6	getMap	184
6.19.4.7	getMap	184
6.19.4.8	getMap	185
6.19.4.9	getMaximumValue	185
6.19.4.10	getMinimumValue	185
6.19.4.11	getMutableMap	185
6.19.4.12	getValueComparison	186
6.19.4.13	insert	186
6.19.4.14	isLoggingEnabled	186
6.19.4.15	setDatasetMaximumSize	187

6.19.5	Friends And Related Function Documentation	187
6.19.5.1	operator<<	187
6.19.6	Member Data Documentation	187
6.19.6.1	datasetMaximumSize	187
6.19.6.2	DEFAULT_DATASET_SIZE	188
6.19.6.3	loggingEnabled	188
6.19.6.4	valueMap	188
6.20	cryomesh::dataobjects::DataObjectController< U, T > Class Template Reference	188
6.20.1	Detailed Description	189
6.20.2	Constructor & Destructor Documentation	189
6.20.2.1	DataObjectController	189
6.20.2.2	DataObjectController	189
6.20.2.3	~DataObjectController	189
6.20.3	Member Function Documentation	190
6.20.3.1	enableLogging	190
6.20.3.2	getDataObject	190
6.20.3.3	getMap	190
6.20.3.4	refreshDataObject	190
6.20.4	Member Data Documentation	191
6.20.4.1	dataObject	191
6.21	cryomesh::manipulators::IClusterAnalyser::EnergyVariationWeighting-Map Struct Reference	191
6.21.1	Detailed Description	191
6.21.2	Constructor & Destructor Documentation	191
6.21.2.1	EnergyVariationWeightingMap	191
6.21.3	Member Data Documentation	192
6.21.3.1	variationMap	192
6.22	cryomesh::structures::Fibre Class Reference	192
6.22.1	Detailed Description	194
6.22.2	Member Enumeration Documentation	195
6.22.2.1	ClusterConnectionType	195
6.22.2.2	FibreType	195
6.22.3	Constructor & Destructor Documentation	195

6.22.3.1	Fibre	195
6.22.3.2	Fibre	196
6.22.3.3	~Fibre	196
6.22.4	Member Function Documentation	196
6.22.4.1	connectAllConnections	196
6.22.4.2	countConnections	197
6.22.4.3	createConnections	197
6.22.4.4	disconnectAllConnections	198
6.22.4.5	enableDebug	198
6.22.4.6	forceFireInputNodes	198
6.22.4.7	forceFireNodes	198
6.22.4.8	forceFireOutputNodes	199
6.22.4.9	getConnections	199
6.22.4.10	getConnector	199
6.22.4.11	getInputNodes	200
6.22.4.12	getInputNodesPattern	200
6.22.4.13	getMutableConnections	200
6.22.4.14	getMutableConnector	200
6.22.4.15	getNodes	201
6.22.4.16	getNodesPattern	201
6.22.4.17	getOutputNodes	202
6.22.4.18	getOutputNodesPattern	202
6.22.4.19	getType	202
6.22.4.20	getWidth	202
6.22.4.21	isConnected	203
6.22.4.22	setType	203
6.22.4.23	trigger	203
6.22.4.24	trigger	204
6.22.4.25	trigger	204
6.22.4.26	trigger	204
6.22.4.27	update	204
6.22.5	Friends And Related Function Documentation	205
6.22.5.1	operator<<	205
6.22.6	Member Data Documentation	205



6.22.6.1	connections	205
6.22.6.2	connector	205
6.22.6.3	fibreType	205
6.23	cryomesh::structures::FibreMap Class Reference	206
6.23.1	Detailed Description	206
6.23.2	Constructor & Destructor Documentation	206
6.23.2.1	FibreMap	206
6.23.2.2	~FibreMap	206
6.23.3	Member Function Documentation	206
6.23.3.1	update	206
6.23.4	Friends And Related Function Documentation	207
6.23.4.1	operator<<	207
6.24	cryomesh::manipulators::IClusterAnalyser Class Reference	207
6.24.1	Detailed Description	208
6.24.2	Member Enumeration Documentation	208
6.24.2.1	EnergyVariation	208
6.24.3	Constructor & Destructor Documentation	209
6.24.3.1	IClusterAnalyser	209
6.24.3.2	~IClusterAnalyser	209
6.24.4	Member Function Documentation	209
6.24.4.1	analyseCluster	209
6.24.4.2	calculateRangeEnergies	209
6.24.4.3	getConnectionRestructuring	210
6.24.4.4	getEnergyVariationMap	210
6.24.4.5	getNodeRestructuring	210
6.24.5	Member Data Documentation	210
6.24.5.1	connectionRestructuring	210
6.24.5.2	nodeRestructuring	210
6.25	cryomesh::components::Impulse Class Reference	210
6.25.1	Detailed Description	213
6.25.2	Constructor & Destructor Documentation	213
6.25.2.1	Impulse	213
6.25.2.2	Impulse	213
6.25.2.3	Impulse	214

6.25.2.4	Impulse	214
6.25.2.5	~Impulse	214
6.25.3	Member Function Documentation	214
6.25.3.1	enableDebug	215
6.25.3.2	getActivities	215
6.25.3.3	getActivity	215
6.25.3.4	getActivity	215
6.25.3.5	getActivityBoundary	216
6.25.3.6	getActivityDelay	216
6.25.3.7	getActivityMaximum	216
6.25.3.8	getActivityMinimum	216
6.25.3.9	getActivityTimer	216
6.25.3.10	getFirstActiveCycle	217
6.25.3.11	getLastActiveCycle	217
6.25.3.12	getMutableActivityTimer	217
6.25.3.13	getRandom	218
6.25.3.14	getTriggerImpulse	218
6.25.3.15	invert	218
6.25.3.16	isActive	218
6.25.3.17	isActive	219
6.25.3.18	isActive	219
6.25.3.19	operator!=	219
6.25.3.20	operator+	220
6.25.3.21	operator+=	220
6.25.3.22	operator=	220
6.25.3.23	operator==	221
6.25.3.24	randomise	221
6.25.3.25	setActivityDelay	221
6.25.3.26	setActivityTimer	221
6.25.3.27	setFirstActiveCycle	222
6.25.4	Friends And Related Function Documentation	222
6.25.4.1	operator<<	222
6.25.5	Member Data Documentation	222
6.25.5.1	activityDelay	222

6.25.5.2	<a href="#">activityTimer</a>	222
6.25.5.3	<a href="#">firstActiveCycle</a>	223
6.25.5.4	<a href="#">FORCED_TRIGGER_ACTIVITY</a>	223
6.25.5.5	<a href="#">lastActiveCycle</a>	223
6.25.5.6	<a href="#">MAX_ACTIVITY</a>	223
6.25.5.7	<a href="#">MAX_ACTIVITY_DELAY</a>	223
6.25.5.8	<a href="#">MAX_ACTIVITY_LENGTH</a>	224
6.25.5.9	<a href="#">MIN_ACTIVITY</a>	224
6.25.5.10	<a href="#">MIN_ACTIVITY_DELAY</a>	224
6.25.5.11	<a href="#">MIN_ACTIVITY_LENGTH</a>	224
6.25.5.12	<a href="#">MIN_ACTIVITY_MAGNITUDE</a>	224
6.26	<a href="#">cryomesh::components::ImpulseCollection Class Reference</a>	224
6.26.1	<a href="#">Detailed Description</a>	227
6.26.2	<a href="#">Member Enumeration Documentation</a>	227
6.26.2.1	<a href="#">Comparison</a>	227
6.26.3	<a href="#">Constructor &amp; Destructor Documentation</a>	227
6.26.3.1	<a href="#">ImpulseCollection</a>	227
6.26.3.2	<a href="#">~ImpulseCollection</a>	227
6.26.4	<a href="#">Member Function Documentation</a>	227
6.26.4.1	<a href="#">clearActiveImpulses</a>	227
6.26.4.2	<a href="#">clearActiveImpulses</a>	228
6.26.4.3	<a href="#">clearActiveImpulses</a>	228
6.26.4.4	<a href="#">clearActivitiesByMaximum</a>	229
6.26.4.5	<a href="#">clearActivitiesByMinimum</a>	229
6.26.4.6	<a href="#">clearActivitiesByValue</a>	230
6.26.4.7	<a href="#">clearImpulses</a>	230
6.26.4.8	<a href="#">clearImpulses</a>	230
6.26.4.9	<a href="#">clearImpulses</a>	231
6.26.4.10	<a href="#">decrementActivityTimers</a>	231
6.26.4.11	<a href="#">enableDebug</a>	231
6.26.4.12	<a href="#">enableLogging</a>	232
6.26.4.13	<a href="#">getActivity</a>	232
6.26.4.14	<a href="#">getActivity</a>	232
6.26.4.15	<a href="#">getByActivityTimerValue</a>	233

6.26.4.16	<a href="#">getDataObject</a>	233
6.26.4.17	<a href="#">getMap</a>	233
6.26.4.18	<a href="#">operator!=</a>	234
6.26.4.19	<a href="#">operator+</a>	234
6.26.4.20	<a href="#">operator+=</a>	234
6.26.4.21	<a href="#">operator=</a>	235
6.26.4.22	<a href="#">operator==</a>	235
6.26.4.23	<a href="#">refreshDataObject</a>	235
6.26.4.24	<a href="#">removeByActivityTimerValue</a>	236
6.26.5	<a href="#">Friends And Related Function Documentation</a>	236
6.26.5.1	<a href="#">operator&lt;&lt;</a>	236
6.26.6	<a href="#">Member Data Documentation</a>	236
6.26.6.1	<a href="#">dataObject</a>	236
6.27	<a href="#">cryomesh::manager::InputPatternsTableFormat Struct Reference</a>	237
6.27.1	<a href="#">Detailed Description</a>	237
6.27.2	<a href="#">Constructor &amp; Destructor Documentation</a>	238
6.27.2.1	<a href="#">InputPatternsTableFormat</a>	238
6.27.3	<a href="#">Member Function Documentation</a>	238
6.27.3.1	<a href="#">getCreateTable</a>	238
6.27.3.2	<a href="#">getKey</a>	238
6.27.3.3	<a href="#">getName</a>	238
6.27.4	<a href="#">Member Data Documentation</a>	239
6.27.4.1	<a href="#">columns</a>	239
6.27.4.2	<a href="#">name</a>	239
6.28	<a href="#">cryomesh::common::Loggable Class Reference</a>	239
6.28.1	<a href="#">Detailed Description</a>	240
6.28.2	<a href="#">Member Enumeration Documentation</a>	240
6.28.2.1	<a href="#">LoggingDepth</a>	240
6.28.3	<a href="#">Constructor &amp; Destructor Documentation</a>	240
6.28.3.1	<a href="#">Loggable</a>	240
6.28.3.2	<a href="#">~Loggable</a>	241
6.28.4	<a href="#">Member Function Documentation</a>	241
6.28.4.1	<a href="#">print</a>	241
6.29	<a href="#">cryomesh::structures::Mesh Class Reference</a>	241

6.29.1	Detailed Description	242
6.29.2	Member Enumeration Documentation	242
6.29.2.1	BlendingMethod	242
6.29.3	Constructor & Destructor Documentation	242
6.29.3.1	Mesh	242
6.29.3.2	Mesh	243
6.29.3.3	~Mesh	243
6.29.4	Member Function Documentation	243
6.29.4.1	getActivityGrid	243
6.29.4.2	getBlendedActivity	243
6.29.4.3	getCluster	243
6.29.4.4	update	243
6.29.4.5	warp	244
6.29.4.6	warp	244
6.29.4.7	warp	244
6.29.5	Member Data Documentation	245
6.29.5.1	cluster	245
6.29.5.2	DEFAULT_BLEND_FORCE	245
6.29.5.3	DEFAULT_MESH_GRANULARITY	245
6.29.5.4	grid	245
6.30	cryomesh::structures::NodeMesh::NeighbourhoodRanges Struct - Reference	245
6.30.1	Detailed Description	246
6.30.2	Member Data Documentation	246
6.30.2.1	maximumNeighbourCount	246
6.30.2.2	maximumNeighbourDistance	246
6.30.2.3	minimumNeighbourCount	246
6.30.2.4	minimumNeighbourDistance	246
6.31	cryomesh::components::Node Class Reference	247
6.31.1	Detailed Description	250
6.31.2	Member Enumeration Documentation	251
6.31.2.1	ActivationState	251
6.31.2.2	RecoverySetting	251
6.31.3	Constructor & Destructor Documentation	251

6.31.3.1	Node	251
6.31.3.2	~Node	251
6.31.4	Member Function Documentation	252
6.31.4.1	addActivity	252
6.31.4.2	addImpulse	252
6.31.4.3	addImpulses	252
6.31.4.4	checkActivationState	253
6.31.4.5	checkFire	253
6.31.4.6	connectInput	253
6.31.4.7	connectOutput	254
6.31.4.8	destroyAllConnections	254
6.31.4.9	destroyAllInputConnections	254
6.31.4.10	destroyAllOutputConnections	254
6.31.4.11	emitImpulse	254
6.31.4.12	emitImpulseNegative	255
6.31.4.13	emitImpulsePositive	255
6.31.4.14	enableDebug	255
6.31.4.15	enableLogging	255
6.31.4.16	enterRecovery	255
6.31.4.17	forceFire	256
6.31.4.18	getActivities	256
6.31.4.19	getActivity	256
6.31.4.20	getActivity	256
6.31.4.21	getActivityThreshold	257
6.31.4.22	getConnector	257
6.31.4.23	getDatabaseObject	257
6.31.4.24	getDataObject	258
6.31.4.25	getEmittedImpulse	258
6.31.4.26	getImpulses	258
6.31.4.27	getLastActivationState	258
6.31.4.28	getMap	259
6.31.4.29	getMutableConnector	259
6.31.4.30	getMutableEmittedImpulse	259
6.31.4.31	getMutableImpulses	260

6.31.4.32	getPosition	260
6.31.4.33	getPrimaryInputConnections	260
6.31.4.34	getPrimaryOutputConnections	260
6.31.4.35	getRandom	260
6.31.4.36	isActive	261
6.31.4.37	isInputIsolated	261
6.31.4.38	isLive	261
6.31.4.39	isOutputIsolated	261
6.31.4.40	isPrimaryInputAttachedNode	262
6.31.4.41	isPrimaryOutputAttachedNode	262
6.31.4.42	isTriggered	262
6.31.4.43	printConnections	262
6.31.4.44	randomise	262
6.31.4.45	refreshDataObject	263
6.31.4.46	setActivity	263
6.31.4.47	setActivity	263
6.31.4.48	setPosition	263
6.31.4.49	update	264
6.31.4.50	updateActivity	264
6.31.4.51	updateActivity	264
6.31.4.52	updateImpulses	264
6.31.4.53	updatePosition	265
6.31.5	Friends And Related Function Documentation	265
6.31.5.1	operator<<	265
6.31.6	Member Data Documentation	265
6.31.6.1	activities	265
6.31.6.2	activityThreshold	265
6.31.6.3	connector	266
6.31.6.4	dataObject	266
6.31.6.5	emittedImpulse	266
6.31.6.6	impulses	266
6.31.6.7	lastActivationState	266
6.31.6.8	MAX_ACTIVITIES_LENGTH	266
6.31.6.9	MAX_ACTIVITY_THRESHOLD	267

6.31.6.10	MAX_BOUNDING_BOX_POINT	267
6.31.6.11	MIN_ACTIVITY_THRESHOLD	267
6.31.6.12	position	267
6.32	cryomesh::manager::NodeDatabaseObject Class Reference	267
6.32.1	Detailed Description	269
6.32.2	Constructor & Destructor Documentation	269
6.32.2.1	NodeDatabaseObject	269
6.32.2.2	NodeDatabaseObject	269
6.32.2.3	~NodeDatabaseObject	270
6.32.3	Member Function Documentation	270
6.32.3.1	findValue	270
6.32.3.2	getActivity	270
6.32.3.3	getColumnMapFromEntry	271
6.32.3.4	getCycle	271
6.32.3.5	getInsert	271
6.32.3.6	getKey	271
6.32.3.7	getPoint	272
6.32.3.8	getUUID	272
6.32.3.9	toString	272
6.32.4	Member Data Documentation	272
6.32.4.1	activity	273
6.32.4.2	ACTIVITY_TAG	273
6.32.4.3	columns	273
6.32.4.4	cycle	273
6.32.4.5	CYCLE_TAG	273
6.32.4.6	ID_TAG	273
6.32.4.7	point	273
6.32.4.8	uuid	274
6.32.4.9	X_TAG	274
6.32.4.10	Y_TAG	274
6.32.4.11	Z_TAG	274
6.33	cryomesh::utilities::SequencerGeneric::NodeEntry Struct Reference	274
6.33.1	Detailed Description	275
6.33.2	Constructor & Destructor Documentation	275



6.33.2.1	<a href="#">NodeEntry</a>	275
6.33.3	<a href="#">Friends And Related Function Documentation</a>	275
6.33.3.1	<a href="#">operator&lt;&lt;</a>	275
6.33.4	<a href="#">Member Data Documentation</a>	275
6.33.4.1	<a href="#">childNodes</a>	275
6.33.4.2	<a href="#">info</a>	275
6.33.4.3	<a href="#">name</a>	276
6.33.4.4	<a href="#">parentNode</a>	276
6.34	<a href="#">cryomesh::components::NodeMap Class Reference</a>	276
6.34.1	<a href="#">Detailed Description</a>	277
6.34.2	<a href="#">Constructor &amp; Destructor Documentation</a>	277
6.34.2.1	<a href="#">NodeMap</a>	277
6.34.2.2	<a href="#">~NodeMap</a>	277
6.34.3	<a href="#">Member Function Documentation</a>	277
6.34.3.1	<a href="#">addRandomImpulses</a>	277
6.34.3.2	<a href="#">getAllConnections</a>	277
6.34.3.3	<a href="#">getAllInputConnections</a>	277
6.34.3.4	<a href="#">getAllOutputConnections</a>	277
6.34.3.5	<a href="#">getAllPrimaryInputNodes</a>	278
6.34.3.6	<a href="#">getAllPrimaryOutputNodes</a>	278
6.34.3.7	<a href="#">update</a>	278
6.34.4	<a href="#">Friends And Related Function Documentation</a>	278
6.34.4.1	<a href="#">operator&lt;&lt;</a>	278
6.35	<a href="#">cryomesh::structures::NodeMesh Class Reference</a>	278
6.35.1	<a href="#">Detailed Description</a>	280
6.35.2	<a href="#">Member Enumeration Documentation</a>	280
6.35.2.1	<a href="#">InterpolationStyle</a>	280
6.35.3	<a href="#">Constructor &amp; Destructor Documentation</a>	281
6.35.3.1	<a href="#">NodeMesh</a>	281
6.35.3.2	<a href="#">NodeMesh</a>	281
6.35.3.3	<a href="#">~NodeMesh</a>	281
6.35.4	<a href="#">Member Function Documentation</a>	281
6.35.4.1	<a href="#">getDecayRate</a>	281
6.35.4.2	<a href="#">getInterpolatedActivity</a>	282

6.35.4.3	<a href="#">getNeighbourhoodActivities</a>	282
6.35.4.4	<a href="#">getNeighbourRanges</a>	283
6.35.4.5	<a href="#">getNodeNeighbourhoodMap</a>	283
6.35.4.6	<a href="#">printNeighbourhoodActivities</a>	283
6.35.4.7	<a href="#">printNeighbourhoods</a>	284
6.35.4.8	<a href="#">regenerateActivities</a>	284
6.35.4.9	<a href="#">regenerateNeighbourhoods</a>	284
6.35.4.10	<a href="#">update</a>	285
6.35.4.11	<a href="#">warpNodes</a>	285
6.35.5	<a href="#">Friends And Related Function Documentation</a>	285
6.35.5.1	<a href="#">operator&lt;&lt;</a>	285
6.35.6	<a href="#">Member Data Documentation</a>	285
6.35.6.1	<a href="#">cluster</a>	285
6.35.6.2	<a href="#">decayRate</a>	285
6.35.6.3	<a href="#">INTERPOLATED_ACTIVITY_SCALING_FACTOR</a>	286
6.35.6.4	<a href="#">MAX_RADIUS_FRACTION_OF_BOUNDING_BOX</a>	286
6.35.6.5	<a href="#">maximumNeighbourhoodRadius</a>	286
6.35.6.6	<a href="#">neighbourhoodActivities</a>	286
6.35.6.7	<a href="#">nodeNeighbourhoodMap</a>	286
6.36	<a href="#">cryomesh::manager::NodeTableFormat Struct Reference</a>	286
6.36.1	<a href="#">Detailed Description</a>	287
6.36.2	<a href="#">Constructor &amp; Destructor Documentation</a>	287
6.36.2.1	<a href="#">NodeTableFormat</a>	287
6.36.3	<a href="#">Member Function Documentation</a>	288
6.36.3.1	<a href="#">getCreateTable</a>	288
6.36.3.2	<a href="#">getKey</a>	288
6.36.3.3	<a href="#">getName</a>	288
6.36.4	<a href="#">Member Data Documentation</a>	289
6.36.4.1	<a href="#">columns</a>	289
6.36.4.2	<a href="#">name</a>	289
6.37	<a href="#">cryomesh::manager::OutputPatternsTableFormat Struct Reference</a>	289
6.37.1	<a href="#">Detailed Description</a>	290
6.37.2	<a href="#">Constructor &amp; Destructor Documentation</a>	290
6.37.2.1	<a href="#">OutputPatternsTableFormat</a>	290

6.37.3	Member Function Documentation	290
6.37.3.1	getCreateTable	290
6.37.3.2	getKey	291
6.37.3.3	getName	291
6.37.4	Member Data Documentation	291
6.37.4.1	columns	291
6.37.4.2	name	291
6.38	cryomesh::state::Pattern Class Reference	292
6.38.1	Detailed Description	293
6.38.2	Constructor & Destructor Documentation	293
6.38.2.1	Pattern	293
6.38.2.2	Pattern	293
6.38.2.3	Pattern	294
6.38.2.4	Pattern	294
6.38.2.5	Pattern	294
6.38.2.6	~Pattern	294
6.38.3	Member Function Documentation	294
6.38.3.1	assignIds	294
6.38.3.2	compare	294
6.38.3.3	getBinaryString	294
6.38.3.4	getDatabaseObject	294
6.38.3.5	getId	295
6.38.3.6	getIds	295
6.38.3.7	getMutableBinaryString	295
6.38.3.8	getPattern	295
6.38.3.9	getPatternTag	295
6.38.3.10	getRandom	295
6.38.3.11	getSize	296
6.38.3.12	getString	296
6.38.3.13	getWidth	296
6.38.3.14	initialise	296
6.38.3.15	isAllZeroes	296
6.38.3.16	operator<	296
6.38.3.17	operator=	297

6.38.3.18	<a href="#">operator==</a>	297
6.38.3.19	<a href="#">patternToString</a>	297
6.38.3.20	<a href="#">serialize</a>	297
6.38.3.21	<a href="#">setId</a>	297
6.38.3.22	<a href="#">setIds</a>	297
6.38.3.23	<a href="#">setPattern</a>	297
6.38.3.24	<a href="#">setPatternTag</a>	297
6.38.3.25	<a href="#">stringToPattern</a>	298
6.38.4	<a href="#">Friends And Related Function Documentation</a>	298
6.38.4.1	<a href="#">boost::serialization::access</a>	298
6.38.4.2	<a href="#">operator&lt;&lt;</a>	298
6.38.5	<a href="#">Member Data Documentation</a>	298
6.38.5.1	<a href="#">binaryString</a>	298
6.38.5.2	<a href="#">id</a>	298
6.38.5.3	<a href="#">ids</a>	298
6.38.5.4	<a href="#">patternTag</a>	298
6.39	<a href="#">cryomesh::state::PatternChannel Class Reference</a>	299
6.39.1	<a href="#">Detailed Description</a>	301
6.39.2	<a href="#">Member Enumeration Documentation</a>	301
6.39.2.1	<a href="#">ChannelDataType</a>	301
6.39.2.2	<a href="#">PrintFormat</a>	301
6.39.3	<a href="#">Constructor &amp; Destructor Documentation</a>	301
6.39.3.1	<a href="#">PatternChannel</a>	301
6.39.3.2	<a href="#">PatternChannel</a>	302
6.39.3.3	<a href="#">~PatternChannel</a>	302
6.39.3.4	<a href="#">PatternChannel</a>	302
6.39.4	<a href="#">Member Function Documentation</a>	302
6.39.4.1	<a href="#">addPattern</a>	302
6.39.4.2	<a href="#">addPatterns</a>	302
6.39.4.3	<a href="#">addPatterns</a>	302
6.39.4.4	<a href="#">clearPatternList</a>	303
6.39.4.5	<a href="#">enableDebug</a>	303
6.39.4.6	<a href="#">forcePatternListSize</a>	303
6.39.4.7	<a href="#">forcePatternListSize</a>	303

6.39.4.8	<a href="#">getChannelDataType</a>	303
6.39.4.9	<a href="#">getCurrentPattern</a>	303
6.39.4.10	<a href="#">getLength</a>	303
6.39.4.11	<a href="#">getMaxPatternListSize</a>	304
6.39.4.12	<a href="#">getMutablePatternByUUID</a>	304
6.39.4.13	<a href="#">getPatternByCycle</a>	304
6.39.4.14	<a href="#">getPatternByTag</a>	304
6.39.4.15	<a href="#">getPatternByTagMap</a>	304
6.39.4.16	<a href="#">getPatternByUUID</a>	304
6.39.4.17	<a href="#">getPatternList</a>	305
6.39.4.18	<a href="#">getPatternListIterator</a>	305
6.39.4.19	<a href="#">getPatternMap</a>	305
6.39.4.20	<a href="#">getPatternPosition</a>	305
6.39.4.21	<a href="#">getRefID</a>	305
6.39.4.22	<a href="#">getRefIDS</a>	305
6.39.4.23	<a href="#">getWidth</a>	306
6.39.4.24	<a href="#">matchGlobally</a>	306
6.39.4.25	<a href="#">matchSequentially</a>	306
6.39.4.26	<a href="#">nextPattern</a>	306
6.39.4.27	<a href="#">operator=</a>	306
6.39.4.28	<a href="#">previousPattern</a>	307
6.39.4.29	<a href="#">printBinaryFormattedPatternList</a>	307
6.39.4.30	<a href="#">printFormattedPatternList</a>	307
6.39.4.31	<a href="#">printIntegerFormattedPatternList</a>	307
6.39.4.32	<a href="#">printPatternList</a>	307
6.39.4.33	<a href="#">printTextFormattedPatternList</a>	307
6.39.4.34	<a href="#">removePatterns</a>	308
6.39.4.35	<a href="#">removePatterns</a>	308
6.39.4.36	<a href="#">setMaxPatternListSize</a>	308
6.39.4.37	<a href="#">setRefID</a>	308
6.39.4.38	<a href="#">setWidth</a>	308
6.39.5	<a href="#">Friends And Related Function Documentation</a>	308
6.39.5.1	<a href="#">operator&lt;&lt;</a>	308
6.39.6	<a href="#">Member Data Documentation</a>	308

6.39.6.1	<a href="#">channelDataType</a>	309
6.39.6.2	<a href="#">DEFAULT_MAX_PATTERN_LIST_SIZE</a>	309
6.39.6.3	<a href="#">length</a>	309
6.39.6.4	<a href="#">maxPatternListSize</a>	309
6.39.6.5	<a href="#">patternByTagMap</a>	309
6.39.6.6	<a href="#">patternList</a>	309
6.39.6.7	<a href="#">patternListIterator</a>	310
6.39.6.8	<a href="#">patternMap</a>	310
6.39.6.9	<a href="#">patternPosition</a>	310
6.39.6.10	<a href="#">refID</a>	310
6.39.6.11	<a href="#">REFID_CREATE_START</a>	310
6.39.6.12	<a href="#">refIDS</a>	310
6.39.6.13	<a href="#">width</a>	310
6.40	<a href="#">cryomesh::state::PatternChannelMap Class Reference</a>	311
6.40.1	<a href="#">Detailed Description</a>	311
6.40.2	<a href="#">Constructor &amp; Destructor Documentation</a>	311
6.40.2.1	<a href="#">PatternChannelMap</a>	311
6.40.2.2	<a href="#">~PatternChannelMap</a>	311
6.40.3	<a href="#">Member Function Documentation</a>	311
6.40.3.1	<a href="#">getPatterns</a>	312
6.41	<a href="#">cryomesh::manager::PatternDatabaseObject Class Reference</a>	312
6.41.1	<a href="#">Detailed Description</a>	313
6.41.2	<a href="#">Constructor &amp; Destructor Documentation</a>	313
6.41.2.1	<a href="#">PatternDatabaseObject</a>	313
6.41.2.2	<a href="#">PatternDatabaseObject</a>	314
6.41.2.3	<a href="#">~PatternDatabaseObject</a>	314
6.41.3	<a href="#">Member Function Documentation</a>	314
6.41.3.1	<a href="#">findValue</a>	314
6.41.3.2	<a href="#">getColumnMapFromEntry</a>	315
6.41.3.3	<a href="#">getCycle</a>	315
6.41.3.4	<a href="#">getInsert</a>	315
6.41.3.5	<a href="#">getKey</a>	315
6.41.3.6	<a href="#">getPattern</a>	316
6.41.3.7	<a href="#">getUUID</a>	316

6.41.3.8	<a href="#">toString</a>	316
6.41.4	<a href="#">Member Data Documentation</a>	316
6.41.4.1	<a href="#">columns</a>	317
6.41.4.2	<a href="#">cycle</a>	317
6.41.4.3	<a href="#">CYCLE_TAG</a>	317
6.41.4.4	<a href="#">ID_TAG</a>	317
6.41.4.5	<a href="#">pattern</a>	317
6.41.4.6	<a href="#">PATTERN_TAG</a>	317
6.41.4.7	<a href="#">uuid</a>	317
6.42	<a href="#">cryomesh::state::PatternTag Class Reference</a>	318
6.42.1	<a href="#">Detailed Description</a>	318
6.42.2	<a href="#">Constructor &amp; Destructor Documentation</a>	318
6.42.2.1	<a href="#">PatternTag</a>	318
6.42.2.2	<a href="#">~PatternTag</a>	319
6.42.3	<a href="#">Member Function Documentation</a>	319
6.42.3.1	<a href="#">getEndTag</a>	319
6.42.3.2	<a href="#">getGlobalTag</a>	319
6.42.3.3	<a href="#">getStartTag</a>	319
6.42.3.4	<a href="#">getTag</a>	319
6.42.3.5	<a href="#">moveTag</a>	319
6.42.3.6	<a href="#">moveTag</a>	319
6.42.3.7	<a href="#">setEndTag</a>	320
6.42.3.8	<a href="#">setStartTag</a>	320
6.42.3.9	<a href="#">setTag</a>	320
6.43	<a href="#">cryomesh::state::PatternTagByDate Class Reference</a>	320
6.43.1	<a href="#">Detailed Description</a>	322
6.43.2	<a href="#">Member Enumeration Documentation</a>	322
6.43.2.1	<a href="#">DateType</a>	322
6.43.3	<a href="#">Constructor &amp; Destructor Documentation</a>	322
6.43.3.1	<a href="#">PatternTagByDate</a>	322
6.43.3.2	<a href="#">PatternTagByDate</a>	322
6.43.3.3	<a href="#">~PatternTagByDate</a>	322
6.43.4	<a href="#">Member Function Documentation</a>	322
6.43.4.1	<a href="#">getEndTag</a>	322

6.43.4.2	<a href="#">getGlobalTag</a>	323
6.43.4.3	<a href="#">getStartTag</a>	323
6.43.4.4	<a href="#">getTag</a>	323
6.43.4.5	<a href="#">isLeapYear</a>	323
6.43.4.6	<a href="#">moveDay</a>	323
6.43.4.7	<a href="#">moveHour</a>	323
6.43.4.8	<a href="#">moveMonth</a>	324
6.43.4.9	<a href="#">moveTag</a>	324
6.43.4.10	<a href="#">moveTag</a>	324
6.43.4.11	<a href="#">moveWeek</a>	324
6.43.4.12	<a href="#">moveYear</a>	324
6.43.4.13	<a href="#">serialize</a>	325
6.43.4.14	<a href="#">setEndTag</a>	325
6.43.4.15	<a href="#">setStartTag</a>	325
6.43.4.16	<a href="#">setTag</a>	325
6.43.4.17	<a href="#">tagToTm</a>	325
6.43.4.18	<a href="#">tmToTag</a>	325
6.43.5	<a href="#">Friends And Related Function Documentation</a>	326
6.43.5.1	<a href="#">boost::serialization::access</a>	326
6.43.6	<a href="#">Member Data Documentation</a>	326
6.43.6.1	<a href="#">currentTime</a>	326
6.43.6.2	<a href="#">DateFormat</a>	326
6.43.6.3	<a href="#">dateType</a>	326
6.43.6.4	<a href="#">endTime</a>	326
6.43.6.5	<a href="#">GlobalCurrentTag</a>	326
6.43.6.6	<a href="#">GlobalEndTag</a>	326
6.43.6.7	<a href="#">GlobalStartTag</a>	327
6.43.6.8	<a href="#">globalTag</a>	327
6.43.6.9	<a href="#">startTime</a>	327
6.44	<a href="#">cryomesh::state::PatternTagByld Class Reference</a>	327
6.44.1	<a href="#">Detailed Description</a>	328
6.44.2	<a href="#">Constructor &amp; Destructor Documentation</a>	328
6.44.2.1	<a href="#">PatternTagByld</a>	328
6.44.2.2	<a href="#">~PatternTagByld</a>	328



6.44.3	Member Function Documentation	328
6.44.3.1	getEndTag	328
6.44.3.2	getGlobalTag	328
6.44.3.3	getStartTag	329
6.44.3.4	getTag	329
6.44.3.5	moveTag	329
6.44.3.6	moveTag	329
6.44.3.7	setEndTag	329
6.44.3.8	setStartTag	330
6.44.3.9	setTag	330
6.44.4	Member Data Documentation	330
6.44.4.1	globalTag	330
6.44.4.2	id	330
6.45	cryomesh::Pointer< T > Struct Template Reference	330
6.45.1	Detailed Description	331
6.45.2	Member Typedef Documentation	331
6.45.2.1	scoped_ptr	331
6.45.2.2	shared_ptr	331
6.46	cryomesh::manipulators::ClusterAnalysisData::RangeEnergy Struct - Reference	331
6.46.1	Detailed Description	332
6.46.2	Constructor & Destructor Documentation	332
6.46.2.1	RangeEnergy	332
6.46.2.2	RangeEnergy	332
6.46.2.3	RangeEnergy	332
6.46.2.4	RangeEnergy	332
6.46.2.5	~RangeEnergy	333
6.46.3	Member Function Documentation	333
6.46.3.1	operator+	333
6.46.3.2	operator+=	333
6.46.3.3	operator/	333
6.46.3.4	operator/=	334
6.46.4	Friends And Related Function Documentation	334
6.46.4.1	operator<<	334

6.46.5	Member Data Documentation . . . . .	334
6.46.5.1	endCycle . . . . .	334
6.46.5.2	energy . . . . .	334
6.46.5.3	energyFraction . . . . .	334
6.46.5.4	energyMax . . . . .	335
6.46.5.5	energyMin . . . . .	335
6.46.5.6	startCycle . . . . .	335
6.47	cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown Struct Reference . . . . .	335
6.47.1	Detailed Description . . . . .	336
6.47.2	Constructor & Destructor Documentation . . . . .	336
6.47.2.1	RestructuringCountdown . . . . .	336
6.47.3	Member Function Documentation . . . . .	336
6.47.3.1	isAllLongRestructuringEnabled . . . . .	336
6.47.3.2	isAllMediumRestructuringEnabled . . . . .	337
6.47.3.3	isAllRestructuringEnabled . . . . .	337
6.47.3.4	isAllShortRestructuringEnabled . . . . .	337
6.47.3.5	isAnyLongRestructuringEnabled . . . . .	337
6.47.3.6	isAnyMediumRestructuringEnabled . . . . .	337
6.47.3.7	isAnyRestructuringEnabled . . . . .	338
6.47.3.8	isAnyShortRestructuringEnabled . . . . .	338
6.47.3.9	isRestructuringEnabled . . . . .	338
6.47.3.10	operator-- . . . . .	338
6.47.3.11	setLongCountdown . . . . .	338
6.47.3.12	setMediumCountdown . . . . .	339
6.47.3.13	setShortCountdown . . . . .	339
6.47.4	Friends And Related Function Documentation . . . . .	339
6.47.4.1	operator<< . . . . .	339
6.47.5	Member Data Documentation . . . . .	339
6.47.5.1	longCreation . . . . .	339
6.47.5.2	longDestruction . . . . .	340
6.47.5.3	mediumCreation . . . . .	340
6.47.5.4	mediumDestruction . . . . .	340
6.47.5.5	shortCreation . . . . .	340

6.47.5.6	shortDestruction	340
6.48	cryomesh::state::Sequence Class Reference	340
6.48.1	Detailed Description	342
6.48.2	Constructor & Destructor Documentation	342
6.48.2.1	Sequence	342
6.48.2.2	~Sequence	342
6.48.2.3	Sequence	342
6.48.2.4	Sequence	342
6.48.2.5	Sequence	342
6.48.3	Member Function Documentation	343
6.48.3.1	addEntry	343
6.48.3.2	clear	343
6.48.3.3	compare	343
6.48.3.4	compare	343
6.48.3.5	compareInput	343
6.48.3.6	compareOutput	343
6.48.3.7	getAndAdvanceCurrentInputPattern	343
6.48.3.8	getAndAdvanceCurrentOutputPattern	344
6.48.3.9	getCurrentInputPattern	344
6.48.3.10	getCurrentInputPatternId	344
6.48.3.11	getCurrentIterator	344
6.48.3.12	getCurrentOutputPattern	344
6.48.3.13	getCurrentOutputPatternId	344
6.48.3.14	getNextInputPattern	344
6.48.3.15	getPatterns	345
6.48.3.16	initialise	345
6.48.3.17	isAllZeroes	345
6.48.3.18	isInputAllZeroes	345
6.48.3.19	isOutputAllZeroes	345
6.48.3.20	loadFromFile	345
6.48.3.21	operator=	346
6.48.3.22	operator==	346
6.48.3.23	saveToFile	346
6.48.3.24	serialize	346

6.48.3.25	<a href="#">setCurrentIterator</a>	346
6.48.3.26	<a href="#">setPatterns</a>	346
6.48.4	<a href="#">Friends And Related Function Documentation</a>	346
6.48.4.1	<a href="#">boost::serialization::access</a>	346
6.48.4.2	<a href="#">operator&lt;&lt;</a>	347
6.48.5	<a href="#">Member Data Documentation</a>	347
6.48.5.1	<a href="#">INPUT_TAG</a>	347
6.48.5.2	<a href="#">it_patterns</a>	347
6.48.5.3	<a href="#">OUTPUT_TAG</a>	347
6.48.5.4	<a href="#">patterns</a>	347
6.49	<a href="#">cryomesh::utilities::SequencerChannels Class Reference</a>	347
6.49.1	<a href="#">Detailed Description</a>	348
6.49.2	<a href="#">Constructor &amp; Destructor Documentation</a>	349
6.49.2.1	<a href="#">SequencerChannels</a>	349
6.49.2.2	<a href="#">~SequencerChannels</a>	349
6.49.3	<a href="#">Member Function Documentation</a>	349
6.49.3.1	<a href="#">readSequences</a>	349
6.49.3.2	<a href="#">writeSequences</a>	349
6.49.4	<a href="#">Member Data Documentation</a>	349
6.49.4.1	<a href="#">DESCRIPTION_STRING</a>	349
6.49.4.2	<a href="#">in_channels_filtered</a>	349
6.49.4.3	<a href="#">out_channels_filtered</a>	350
6.49.4.4	<a href="#">PATTERN_BINARY_STRING</a>	350
6.49.4.5	<a href="#">PATTERN_CHANNEL_DEPTH_STRING</a>	350
6.49.4.6	<a href="#">PATTERN_CHANNEL_INPUT_STRING</a>	350
6.49.4.7	<a href="#">PATTERN_CHANNEL_NOTE_STRING</a>	350
6.49.4.8	<a href="#">PATTERN_CHANNEL_OUTPUT_STRING</a>	351
6.49.4.9	<a href="#">PATTERN_CHANNEL_REFID_STRING</a>	351
6.49.4.10	<a href="#">PATTERN_CHANNEL_STRING</a>	351
6.49.4.11	<a href="#">PATTERN_CHANNEL_TYPE_STRING</a>	351
6.49.4.12	<a href="#">PATTERN_CHANNEL_WIDTH_STRING</a>	351
6.49.4.13	<a href="#">PATTERN_STRING</a>	351
6.49.4.14	<a href="#">PATTERN_TAG_STRING</a>	352
6.49.4.15	<a href="#">VERSION_STRING</a>	352

6.50 cryomesh::utilities::SequencerGeneric Class Reference . . . . .	352
6.50.1 Detailed Description . . . . .	353
6.50.2 Constructor & Destructor Documentation . . . . .	353
6.50.2.1 SequencerGeneric . . . . .	353
6.50.2.2 ~SequencerGeneric . . . . .	353
6.50.3 Member Function Documentation . . . . .	353
6.50.3.1 getNodeEntries . . . . .	353
6.50.3.2 on_comment . . . . .	353
6.50.3.3 on_end_document . . . . .	353
6.50.3.4 on_end_element . . . . .	353
6.50.3.5 on_error . . . . .	354
6.50.3.6 on_fatal_error . . . . .	354
6.50.3.7 on_start_document . . . . .	354
6.50.3.8 on_start_element . . . . .	354
6.50.3.9 on_warning . . . . .	354
6.50.4 Friends And Related Function Documentation . . . . .	354
6.50.4.1 operator<< . . . . .	354
6.50.5 Member Data Documentation . . . . .	354
6.50.5.1 elementCount . . . . .	354
6.50.5.2 nodeEntries . . . . .	355
6.50.5.3 nodeStack . . . . .	355
6.51 cryomesh::utilities::Statistician Class Reference . . . . .	355
6.51.1 Detailed Description . . . . .	356
6.51.2 Constructor & Destructor Documentation . . . . .	356
6.51.2.1 Statistician . . . . .	356
6.51.2.2 ~Statistician . . . . .	356
6.51.3 Member Function Documentation . . . . .	357
6.51.3.1 getActiveNodesPerCluster . . . . .	357
6.51.3.2 getActiveNodesTotal . . . . .	357
6.51.3.3 getBundle . . . . .	357
6.51.3.4 getBundleUUID . . . . .	357
6.51.3.5 getClusterCount . . . . .	357
6.51.3.6 getInputChannelsCount . . . . .	357
6.51.3.7 getInputFibresCount . . . . .	358

6.51.3.8	<a href="#">getNormalFibresCount</a>	358
6.51.3.9	<a href="#">getOutputChannelsCount</a>	358
6.51.3.10	<a href="#">getOutputFibresCount</a>	358
6.51.3.11	<a href="#">getTriggeredNodesPerCluster</a>	358
6.51.3.12	<a href="#">getTriggeredNodesTotal</a>	358
6.51.3.13	<a href="#">update</a>	358
6.51.4	<a href="#">Friends And Related Function Documentation</a>	359
6.51.4.1	<a href="#">operator&lt;&lt;</a>	359
6.51.5	<a href="#">Member Data Documentation</a>	359
6.51.5.1	<a href="#">bundle</a>	359
6.51.5.2	<a href="#">bundleuuid</a>	359
6.51.5.3	<a href="#">clusterCount</a>	359
6.51.5.4	<a href="#">inputChannelsCount</a>	359
6.51.5.5	<a href="#">inputFibresCount</a>	360
6.51.5.6	<a href="#">nodesActive</a>	360
6.51.5.7	<a href="#">nodesTotal</a>	360
6.51.5.8	<a href="#">nodesTriggered</a>	360
6.51.5.9	<a href="#">normalFibresCount</a>	360
6.51.5.10	<a href="#">outputChannelsCount</a>	360
6.51.5.11	<a href="#">outputFibresCount</a>	360
6.52	<a href="#">cryomesh::manager::TableFormat Struct Reference</a>	361
6.52.1	<a href="#">Detailed Description</a>	361
6.52.2	<a href="#">Constructor &amp; Destructor Documentation</a>	361
6.52.2.1	<a href="#">TableFormat</a>	361
6.52.2.2	<a href="#">~TableFormat</a>	361
6.52.3	<a href="#">Member Function Documentation</a>	362
6.52.3.1	<a href="#">getCreateTable</a>	362
6.52.3.2	<a href="#">getKey</a>	362
6.52.3.3	<a href="#">getName</a>	362
6.52.4	<a href="#">Member Data Documentation</a>	362
6.52.4.1	<a href="#">columns</a>	363
6.52.4.2	<a href="#">name</a>	363
6.53	<a href="#">cryomesh::common::TimeKeeper Class Reference</a>	363
6.53.1	<a href="#">Detailed Description</a>	364

6.53.2	Constructor & Destructor Documentation . . . . .	364
6.53.2.1	~TimeKeeper . . . . .	364
6.53.2.2	TimeKeeper . . . . .	365
6.53.2.3	TimeKeeper . . . . .	365
6.53.3	Member Function Documentation . . . . .	365
6.53.3.1	getCycle . . . . .	365
6.53.3.2	getStartTime . . . . .	365
6.53.3.3	getTimeKeeper . . . . .	366
6.53.3.4	getTimer . . . . .	366
6.53.3.5	getTiming . . . . .	366
6.53.3.6	operator= . . . . .	367
6.53.3.7	operator== . . . . .	367
6.53.3.8	reset . . . . .	367
6.53.3.9	update . . . . .	367
6.53.4	Member Data Documentation . . . . .	367
6.53.4.1	cycle . . . . .	368
6.53.4.2	last_timing . . . . .	368
6.53.4.3	start_time . . . . .	368
6.53.4.4	this_timing . . . . .	368
6.53.4.5	timekeeper . . . . .	368
6.53.4.6	timer . . . . .	368
<b>7</b>	<b>File Documentation</b>	<b>369</b>
7.1	/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Connector.h File Reference . . . . .	369
7.2	/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Cycle.cpp File Reference . . . . .	369
7.3	/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Cycle.h File - Reference . . . . .	370
7.4	/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Loggable.h File Reference . . . . .	370
7.5	/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/TimeKeeper.cpp File Reference . . . . .	371
7.6	/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/TimeKeeper.h File Reference . . . . .	371

7.7	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Activity-Timer.h File Reference . . . . .	371
7.8	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Activity-TimerDistance.cpp File Reference . . . . .	372
7.8.1	Define Documentation . . . . .	372
7.8.1.1	ACTIVITYTIMERDISTANCE_DEBUG . . . . .	372
7.9	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Activity-TimerDistance.h File Reference . . . . .	372
7.10	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection.cpp File Reference . . . . .	373
7.11	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection.h File Reference . . . . .	373
7.12	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection-Map.h File Reference . . . . .	374
7.13	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Impulse.cpp File Reference . . . . .	374
7.14	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Impulse.h File Reference . . . . .	375
7.15	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Impulse-Collection.cpp File Reference . . . . .	375
7.16	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Impulse-Collection.h File Reference . . . . .	375
7.17	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Node.cpp File Reference . . . . .	376
7.18	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Node.h File Reference . . . . .	376
7.19	/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Node-Map.h File Reference . . . . .	377
7.20	/home/niall/Projects/Eclipse/CPP/cryomesh/src/dataobjects/Data-Object.h File Reference . . . . .	377
7.21	/home/niall/Projects/Eclipse/CPP/cryomesh/src/dataobjects/Data-ObjectController.h File Reference . . . . .	378
7.22	/home/niall/Projects/Eclipse/CPP/cryomesh/src/Defs.h File Reference . . . . .	378
7.22.1	Typedef Documentation . . . . .	379
7.22.1.1	NeighbourhoodMap . . . . .	379
7.23	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Connection-DatabaseObject.cpp File Reference . . . . .	379
7.24	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Connection-DatabaseObject.h File Reference . . . . .	379



7.25	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Creator.cpp File Reference . . . . .	380
7.26	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Creator.h File Reference . . . . .	380
7.27	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/CryoManager.cpp File Reference . . . . .	380
7.28	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/CryoManager.h File Reference . . . . .	381
7.28.1	Define Documentation . . . . .	381
7.28.1.1	CRYOMANAGER_DEBUG . . . . .	381
7.29	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Database- Manager.cpp File Reference . . . . .	381
7.30	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Database- Manager.h File Reference . . . . .	382
7.31	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Database- Object.h File Reference . . . . .	382
7.32	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/NodeDatabase- Object.cpp File Reference . . . . .	382
7.33	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/NodeDatabase- Object.h File Reference . . . . .	383
7.34	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Pattern- DatabaseObject.cpp File Reference . . . . .	383
7.35	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Pattern- DatabaseObject.h File Reference . . . . .	383
7.36	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/TableFormats.h File Reference . . . . .	384
7.37	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/Cluster- AnalyserBasic.cpp File Reference . . . . .	384
7.38	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/Cluster- AnalyserBasic.h File Reference . . . . .	385
7.39	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/Cluster- AnalysisData.cpp File Reference . . . . .	385
7.40	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/Cluster- AnalysisData.h File Reference . . . . .	385
7.41	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/Cluster- Architect.cpp File Reference . . . . .	386
7.42	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/Cluster- Architect.h File Reference . . . . .	386
7.43	/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ICluster- Analyser.h File Reference . . . . .	387

7.44	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/ActivityPattern.cpp File Reference . . . . .	387
7.45	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/ActivityPattern.h - File Reference . . . . .	388
7.46	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/BinaryString.cpp - File Reference . . . . .	388
7.47	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/BinaryString.h - File Reference . . . . .	388
7.48	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern.cpp File - Reference . . . . .	389
7.49	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern.h File - Reference . . . . .	389
7.50	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannel.cpp File Reference . . . . .	390
7.51	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannel.h File Reference . . . . .	390
7.52	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannel- Map.h File Reference . . . . .	391
7.53	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTag.h File - Reference . . . . .	391
7.54	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagBy- Date.cpp File Reference . . . . .	391
7.55	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagBy- Date.h File Reference . . . . .	392
7.56	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagBy- Id.cpp File Reference . . . . .	392
7.57	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagById.h File Reference . . . . .	392
7.58	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Sequence.cpp - File Reference . . . . .	393
7.59	/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Sequence.h File - Reference . . . . .	393
7.60	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Bundle.cpp - File Reference . . . . .	394
7.61	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Bundle.h File Reference . . . . .	394
7.62	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Cluster.cpp - File Reference . . . . .	395
7.63	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Cluster.h File Reference . . . . .	395

7.64	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Cluster-Map.h File Reference . . . . .	396
7.64.1	Define Documentation . . . . .	396
7.64.1.1	CLUSTERMAP_DEBUG . . . . .	396
7.65	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Fibre.cpp - File Reference . . . . .	396
7.66	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Fibre.h File - Reference . . . . .	396
7.67	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/FibreMap.h - File Reference . . . . .	397
7.68	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Mesh.cpp File Reference . . . . .	397
7.69	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Mesh.h File - Reference . . . . .	398
7.70	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Node-Mesh.cpp File Reference . . . . .	398
7.71	/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/NodeMesh.h File Reference . . . . .	398
7.72	/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/Sequencer-Channels.cpp File Reference . . . . .	399
7.73	/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/Sequencer-Channels.h File Reference . . . . .	399
7.74	/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/Sequencer-Generic.cpp File Reference . . . . .	400
7.75	/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/Sequencer-Generic.h File Reference . . . . .	400
7.76	/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/Statistician.cpp File Reference . . . . .	401
7.77	/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/Statistician.h File Reference . . . . .	401



# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">cryomesh</a>	
<a href="#">Connector.h</a>	13
<a href="#">cryomesh::common</a>	14
<a href="#">cryomesh::components</a>	15
<a href="#">cryomesh::dataobjects</a>	17
<a href="#">cryomesh::manager</a>	17
<a href="#">cryomesh::manipulators</a>	18
<a href="#">cryomesh::state</a>	18
<a href="#">cryomesh::structures</a>	19
<a href="#">cryomesh::utilities</a>	22



## Chapter 2

# Class Index

### 2.1 Class Hierarchy

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

cryomesh::state::ActivityPattern . . . . .	23
cryomesh::components::ActivityTimer . . . . .	25
cryomesh::components::ActivityTimerDistance . . . . .	27
cryomesh::state::BinaryString . . . . .	35
cryomesh::structures::Cluster . . . . .	64
cryomesh::manipulators::ClusterAnalysisData . . . . .	81
cryomesh::manipulators::ClusterArchitect . . . . .	88
cryomesh::components::Connection . . . . .	99
cryomesh::components::ConnectionMap . . . . .	115
cryomesh::common::Connector< U, T > . . . . .	120
cryomesh::manager::Creator . . . . .	135
cryomesh::common::Cycle . . . . .	147
cryomesh::manager::DatabaseManager . . . . .	155
cryomesh::manager::DatabaseObject . . . . .	177
cryomesh::manager::ConnectionDatabaseObject . . . . .	107
cryomesh::manager::NodeDatabaseObject . . . . .	267
cryomesh::manager::PatternDatabaseObject . . . . .	312
cryomesh::dataobjects::DataObject< U, T > . . . . .	180
cryomesh::dataobjects::DataObject< unsigned long int, double > . . . . .	180
cryomesh::dataobjects::DataObjectController< U, T > . . . . .	188
cryomesh::dataobjects::DataObjectController< unsigned long int, double > . . . . .	188
cryomesh::components::ImpulseCollection . . . . .	224
cryomesh::components::Node . . . . .	247
cryomesh::manipulators::IClusterAnalyser::EnergyVariationWeightingMap . . . . .	191
cryomesh::structures::Fibre . . . . .	192
cryomesh::structures::FibreMap . . . . .	206
cryomesh::manipulators::IClusterAnalyser . . . . .	207
cryomesh::manipulators::ClusterAnalyserBasic . . . . .	75

cryomesh::components::Impulse . . . . .	210
cryomesh::common::Loggable . . . . .	239
cryomesh::structures::Bundle . . . . .	41
cryomesh::structures::Mesh . . . . .	241
cryomesh::structures::NodeMesh::NeighbourhoodRanges . . . . .	245
cryomesh::utilities::SequencerGeneric::NodeEntry . . . . .	274
cryomesh::components::NodeMap . . . . .	276
cryomesh::structures::NodeMesh . . . . .	278
cryomesh::state::Pattern . . . . .	292
cryomesh::state::PatternChannel . . . . .	299
cryomesh::state::PatternChannelMap . . . . .	311
cryomesh::state::PatternTag . . . . .	318
cryomesh::state::PatternTagByDate . . . . .	320
cryomesh::state::PatternTagById . . . . .	327
cryomesh::Pointer< T > . . . . .	330
cryomesh::manipulators::ClusterAnalysisData::RangeEnergy . . . . .	331
cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown . . . . .	335
cryomesh::state::Sequence . . . . .	340
cryomesh::utilities::SequencerChannels . . . . .	347
cryomesh::utilities::SequencerGeneric . . . . .	352
cryomesh::utilities::Statistician . . . . .	355
cryomesh::manager::TableFormat . . . . .	361
cryomesh::manager::ConnectionTableFormat . . . . .	117
cryomesh::manager::InputPatternsTableFormat . . . . .	237
cryomesh::manager::NodeTableFormat . . . . .	286
cryomesh::manager::OutputPatternsTableFormat . . . . .	289
cryomesh::common::TimeKeeper . . . . .	363



## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">cryomesh::state::ActivityPattern</a>	
A simple collection of doubles representing a pattern of activities . . .	23
<a href="#">cryomesh::components::ActivityTimer</a>	
Simple interface class for activity timers . . . . .	25
<a href="#">cryomesh::components::ActivityTimerDistance</a> . . . . .	27
<a href="#">cryomesh::state::BinaryString</a> . . . . .	35
<a href="#">cryomesh::structures::Bundle</a>	
A <a href="#">Bundle</a> is the collection of clusters and fibres, it represents the system as a whole . . . . .	41
<a href="#">cryomesh::structures::Cluster</a>	
A <a href="#">Cluster</a> is a collection of self-contained nodes and connections along with an associated <a href="#">Mesh</a> , that can be connected up to one another . . . . .	64
<a href="#">cryomesh::manipulators::ClusterAnalyserBasic</a> . . . . .	75
<a href="#">cryomesh::manipulators::ClusterAnalysisData</a> . . . . .	81
<a href="#">cryomesh::manipulators::ClusterArchitect</a> . . . . .	88
<a href="#">cryomesh::components::Connection</a>	
<a href="#">Connection</a> class to manage the transfer of Impulses between Nodes	99
<a href="#">cryomesh::manager::ConnectionDatabaseObject</a> . . . . .	107
<a href="#">cryomesh::components::ConnectionMap</a>	
Helper class for <a href="#">ConnectionMap</a> to KeyMappedCollection mapping .	115
<a href="#">cryomesh::manager::ConnectionTableFormat</a>	
Struct representing a connections table structure . . . . .	117
<a href="#">cryomesh::common::Connector&lt; U, T &gt;</a>	
<a href="#">Connector</a> is a template to add connectable functionality between two classes . . . . .	120
<a href="#">cryomesh::manager::Creator</a>	
Class to take in a config file of ConfigTranslator form and parse the commands to create a full cryomesh object . . . . .	135

<a href="#">cryomesh::common::Cycle</a>	147
<a href="#">cryomesh::manager::DatabaseManager</a>	
Database manager creates and maintains a database of mesh re-	
lated objects and data	155
<a href="#">cryomesh::manager::DatabaseObject</a>	177
<a href="#">cryomesh::dataobjects::DataObject&lt; U, T &gt;</a>	
Class to contain all the useful data about an object	180
<a href="#">cryomesh::dataobjects::DataObjectController&lt; U, T &gt;</a>	
Class used to interface with data objects	188
<a href="#">cryomesh::manipulators::IClusterAnalyser::EnergyVariationWeightingMap</a>	191
<a href="#">cryomesh::structures::Fibre</a>	
A <a href="#">Fibre</a> is a collection of connections that connect one structure to	
another	192
<a href="#">cryomesh::structures::FibreMap</a>	206
<a href="#">cryomesh::manipulators::IClusterAnalyser</a>	207
<a href="#">cryomesh::components::Impulse</a>	
Impulse is a mobile information packet to be passed between Nodes	210
<a href="#">cryomesh::components::ImpulseCollection</a>	
ImpulseCollection represents a collection of <a href="#">Impulse</a> objects	224
<a href="#">cryomesh::manager::InputPatternsTableFormat</a>	
Struct representing input pattern table structure	237
<a href="#">cryomesh::common::Loggable</a>	239
<a href="#">cryomesh::structures::Mesh</a>	
Mesh is the fabric of connection space and warps and is warped by it	241
<a href="#">cryomesh::structures::NodeMesh::NeighbourhoodRanges</a>	
Struct to capture some statistics data on a nodes neighbourhood	245
<a href="#">cryomesh::components::Node</a>	
Node is an accumulation and computational nodal point of impulses	247
<a href="#">cryomesh::manager::NodeDatabaseObject</a>	267
<a href="#">cryomesh::utilities::SequencerGeneric::NodeEntry</a>	274
<a href="#">cryomesh::components::NodeMap</a>	
Helper class for <a href="#">NodeMap</a> to KeyMappedCollection mapping	276
<a href="#">cryomesh::structures::NodeMesh</a>	
Mesh of nodes and their neighbouring nodes and distances	278
<a href="#">cryomesh::manager::NodeTableFormat</a>	
Struct representing a node table structure	286
<a href="#">cryomesh::manager::OutputPatternsTableFormat</a>	
Struct representing output pattern table structure	289
<a href="#">cryomesh::state::Pattern</a>	292
<a href="#">cryomesh::state::PatternChannel</a>	299
<a href="#">cryomesh::state::PatternChannelMap</a>	311
<a href="#">cryomesh::manager::PatternDatabaseObject</a>	312
<a href="#">cryomesh::state::PatternTag</a>	318
<a href="#">cryomesh::state::PatternTagByDate</a>	320
<a href="#">cryomesh::state::PatternTagById</a>	327
<a href="#">cryomesh::Pointer&lt; T &gt;</a>	
Pointer struct to allow typedef of templated smart pointers	330
<a href="#">cryomesh::manipulators::ClusterAnalysisData::RangeEnergy</a>	
Struct representing the value extrapolated over a history range	331

<a href="#">cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown</a>	
Class to hold together information on whether we can act to restructure items . . . . .	335
<a href="#">cryomesh::state::Sequence</a> . . . . .	340
<a href="#">cryomesh::utilities::SequencerChannels</a> . . . . .	347
<a href="#">cryomesh::utilities::SequencerGeneric</a> . . . . .	352
<a href="#">cryomesh::utilities::Statistician</a>	
Class to draw together lots of useful statistics and monitoring data for a Bundle and its components . . . . .	355
<a href="#">cryomesh::manager::TableFormat</a>	
General structure of a table . . . . .	361
<a href="#">cryomesh::common::TimeKeeper</a>	
TimeKeeper is a class keep track of the cycle state and timing . . . .	363



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

/home/niall/Projects/Eclipse/CPP/cryomesh/src/Defs.h . . . . .	378
/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Connector.h . . . .	369
/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Cycle.cpp . . . . .	369
/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Cycle.h . . . . .	370
/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Loggable.h . . . . .	370
/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/TimeKeeper.cpp . .	371
/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/TimeKeeper.h . . .	371
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ActivityTimer.h .	371
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ActivityTimer- Distance.cpp . . . . .	372
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ActivityTimer- Distance.h . . . . .	372
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection.cpp .	373
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection.h . .	373
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection- Map.h . . . . .	374
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Impulse.cpp . . .	374
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Impulse.h . . . .	375
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ImpulseCollection.- cpp . . . . .	375
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ImpulseCollection.- h . . . . .	375
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Node.cpp . . . .	376
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Node.h . . . . .	376
/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/NodeMap.h . . .	377
/home/niall/Projects/Eclipse/CPP/cryomesh/src/dataobjects/DataObject.h . .	377
/home/niall/Projects/Eclipse/CPP/cryomesh/src/dataobjects/DataObject- Controller.h . . . . .	378

/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/ConnectionDatabase-Object.cpp	379
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/ConnectionDatabase-Object.h	379
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Creator.cpp	380
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Creator.h	380
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/CryoManager.cpp	380
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/CryoManager.h	381
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/DatabaseManager.cpp	381
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/DatabaseManager.h	382
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/DatabaseObject.h	382
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/NodeDatabase-Object.cpp	382
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/NodeDatabase-Object.h	383
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/PatternDatabase-Object.cpp	383
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/PatternDatabase-Object.h	383
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/TableFormats.h	384
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterAnalyser-Basic.cpp	384
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterAnalyser-Basic.h	385
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterAnalysis-Data.cpp	385
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterAnalysis-Data.h	385
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterArchitect.cpp	386
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterArchitect.h	386
/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/IClusterAnalyser.h	387
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/ActivityPattern.cpp	387
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/ActivityPattern.h	388
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/BinaryString.cpp	388
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/BinaryString.h	388
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern.cpp	389
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern.h	389
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannel.cpp	390
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannel.h	390
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannelMap.h	391
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTag.h	391
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagByDate.cpp	391
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagByDate.h	392
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagById.cpp	392
/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagById.h	392

/home/niall/Projects/Eclipse/Cpp/cryomesh/src/state/Sequence.cpp . . . . .	393
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/state/Sequence.h . . . . .	393
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/Bundle.cpp . . . . .	394
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/Bundle.h . . . . .	394
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/Cluster.cpp . . . . .	395
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/Cluster.h . . . . .	395
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/ClusterMap.h . . . . .	396
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/Fibre.cpp . . . . .	396
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/Fibre.h . . . . .	396
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/FibreMap.h . . . . .	397
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/Mesh.cpp . . . . .	397
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/Mesh.h . . . . .	398
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/NodeMesh.cpp . . . . .	398
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/structures/NodeMesh.h . . . . .	398
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/utilities/SequencerChannels.- cpp . . . . .	399
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/utilities/SequencerChannels.- h . . . . .	399
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/utilities/SequencerGeneric.- cpp . . . . .	400
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/utilities/SequencerGeneric.h . . . . .	400
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/utilities/Statistician.cpp . . . . .	401
/home/niall/Projects/Eclipse/Cpp/cryomesh/src/utilities/Statistician.h . . . . .	401





## Chapter 5

# Namespace Documentation

### 5.1 cryomesh Namespace Reference

[Connector.h](#).

#### Namespaces

- namespace [common](#)
- namespace [components](#)
- namespace [dataobjects](#)
- namespace [manager](#)
- namespace [manipulators](#)
- namespace [state](#)
- namespace [structures](#)
- namespace [utilities](#)

#### Classes

- struct [Pointer](#)  
*[Pointer](#) struct to allow typedef of templated smart pointers.*

#### 5.1.1 Detailed Description

[Connector.h](#). [Mesh.h](#).

[Node.h](#).

[ImpulseCollection.h](#).

[ImpulseCollection.cpp](#).

[Impulse.h](#).

[Connection.h](#).

[Cycle.h](#).

[Cycle.cpp](#).

Created on: 19 Jan 2011 Author: SevenMachines<[SevenMachines@yahoo.co.uk](#)>

Created on: 1 Feb 2011 Author: SevenMachines<[SevenMachines@yahoo.co.uk](#)>

Created on: 1 Feb 2011 Author: SevenMachines<[SevenMachines@yahoo.co.uk](#)> Wrapper class around implementation of a cycle

Created on: 3 Jan 2011 Author: SevenMachines<[SevenMachines@yahoo.co.uk](#)>

Created on: 20 Jan 2011 Author: SevenMachines<[SevenMachines@yahoo.co.uk](#)>

Created on: 20 Jan 2011 Author: SevenMachines<[SevenMachines@yahoo.co.uk](#)>

A collection of Impulses that allows for Impulses to be held, 'moved forward' in time, and summated in some way

## 5.2 cryomesh::common Namespace Reference

### Classes

- class [Connector](#)  
*[Connector](#) is a template to add connectable functionality between two classes.*
- class [Cycle](#)
- class [Loggable](#)
- class [TimeKeeper](#)  
*[TimeKeeper](#) is a class keep track of the cycle state and timing.*

### Functions

- `std::ostream & operator<< (std::ostream &os, const Cycle &obj)`

#### 5.2.1 Function Documentation

5.2.1.1 `std::ostream& cryomesh::common::operator<< ( std::ostream & os, const Cycle & obj )`

#### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Cycle</a> & obj The object to stream

## Returns

std::ostream & The output stream

Definition at line 110 of file Cycle.cpp.

References cryomesh::common::Cycle::toLInt().

## 5.3 cryomesh::components Namespace Reference

### Classes

- class [ActivityTimer](#)  
*Simple interface class for activity timers.*
- class [ActivityTimerDistance](#)
- class [Connection](#)  
*[Connection](#) class to manage the transfer of Impulses between Nodes.*
- class [ConnectionMap](#)  
*Helper class for [ConnectionMap](#) to KeyMappedCollection mapping.*
- class [Impulse](#)  
*[Impulse](#) is a mobile information packet to be passed between Nodes.*
- class [ImpulseCollection](#)  
*[ImpulseCollection](#) represents a collection of [Impulse](#) objects.*
- class [Node](#)  
*[Node](#) is an accumulation and computational nodal point of impulses.*
- class [NodeMap](#)  
*Helper class for [NodeMap](#) to KeyMappedCollection mapping.*

### Functions

- std::ostream & [operator<<](#) (std::ostream &os, const [Connection](#) &obj)
- std::ostream & [operator<<](#) (std::ostream &os, const [Impulse](#) &obj)
- std::ostream & [operator<<](#) (std::ostream &os, const [ImpulseCollection](#) &obj)
- std::ostream & [operator<<](#) (std::ostream &os, const [Node](#) &obj)

#### 5.3.1 Function Documentation

5.3.1.1 std::ostream& cryomesh::components::operator<< ( std::ostream & os, const [Connection](#) & obj )

## Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Connection</a> & obj The object to stream

**Returns**

`std::ostream` & The output stream

Definition at line 245 of file `Connection.cpp`.

References `cryomesh::components::Connection::getConnector()`, `cryomesh::components::Connection::getImpulses()`, `cryomesh::components::Connection::isPrimaryInputConnection()`, and `cryomesh::components::Connection::isPrimaryOutputConnection()`.

**5.3.1.2** `std::ostream& cryomesh::components::operator<< ( std::ostream & os, const Impulse & obj )`

**Parameters**

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Impulse</a> & obj The object to stream

**Returns**

`std::ostream` & The output stream

Definition at line 339 of file `Impulse.cpp`.

References `cryomesh::components::Impulse::getActivityDelay()`, `cryomesh::components::Impulse::getActivityTimer()`, `cryomesh::components::Impulse::getFirstActiveCycle()`, and `cryomesh::components::Impulse::getLastActiveCycle()`.

**5.3.1.3** `std::ostream& cryomesh::components::operator<< ( std::ostream & os, const ImpulseCollection & obj )`

**Parameters**

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">ImpulseCollection</a> & obj The object to stream

**Returns**

`std::ostream` & The output stream

Definition at line 375 of file `ImpulseCollection.cpp`.

**5.3.1.4** `std::ostream& cryomesh::components::operator<< ( std::ostream & os, const Node & obj )`

**Parameters**

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Node</a> & obj The object to stream

### Returns

std::ostream & The output stream

Definition at line 548 of file Node.cpp.

References cryomesh::components::Node::getActivityThreshold(), cryomesh::components::Node::getConnector(), cryomesh::components::Node::getImpulses(), cryomesh::components::Node::isPrimaryInputAttachedNode(), cryomesh::components::Node::isPrimaryOutputAttachedNode(), and cryomesh::components::Node::printConnections().

## 5.4 cryomesh::dataobjects Namespace Reference

### Classes

- class [DataObject](#)  
*Class to contain all the useful data about an object.*
- class [DataObjectController](#)  
*Class used to interface with data objects.*

## 5.5 cryomesh::manager Namespace Reference

### Classes

- class [ConnectionDatabaseObject](#)
- class [Creator](#)  
*Class to take in a config file of ConfigTranslator form and parse the commands to create a full cryomesh object.*
- class [DatabaseManager](#)  
*Database manager creates and maintains a database of mesh related objects and data.*
- class [DatabaseObject](#)
- class [NodeDatabaseObject](#)
- class [PatternDatabaseObject](#)
- struct [TableFormat](#)  
*General structure of a table.*
- struct [NodeTableFormat](#)  
*Struct representing a node table structure.*
- struct [ConnectionTableFormat](#)  
*Struct representing a connections table structure.*
- struct [InputPatternsTableFormat](#)  
*Struct representing input pattern table structure.*
- struct [OutputPatternsTableFormat](#)  
*Struct representing output pattern table structure.*

## 5.6 cryomesh::manipulators Namespace Reference

### Classes

- class [ClusterAnalyserBasic](#)
- class [ClusterAnalysisData](#)
- class [ClusterArchitect](#)
- class [IClusterAnalyser](#)

## 5.7 cryomesh::state Namespace Reference

### Classes

- class [ActivityPattern](#)  
*A simple collection of doubles representing a pattern of activities.*
- class [BinaryString](#)
- class [Pattern](#)
- class [PatternChannel](#)
- class [PatternChannelMap](#)
- class [PatternTag](#)
- class [PatternTagByDate](#)
- class [PatternTagById](#)
- class [Sequence](#)

### Functions

- `std::ostream & operator<< (std::ostream &os, const BinaryString &obj)`
- `std::ostream & operator<< (std::ostream &os, const Pattern &obj)`
- `std::ostream & operator<< (std::ostream &os, const PatternChannel &obj)`
- `std::ostream & operator<< (std::ostream &os, const Sequence &obj)`

### 5.7.1 Function Documentation

#### 5.7.1.1 `std::ostream& cryomesh::state::operator<< ( std::ostream & os, const BinaryString & obj )`

Definition at line 223 of file `BinaryString.cpp`.

References `cryomesh::state::BinaryString::getBinaryString()`.

#### 5.7.1.2 `std::ostream& cryomesh::state::operator<< ( std::ostream & os, const Pattern & obj )`

Definition at line 242 of file `Pattern.cpp`.

References `cryomesh::state::Pattern::getId()`, and `cryomesh::state::Pattern::getString()`.

5.7.1.3 `std::ostream& cryomesh::state::operator<< ( std::ostream & os, const Sequence & obj )`

Definition at line 380 of file Sequence.cpp.

References `cryomesh::state::Sequence::getPatterns()`.

5.7.1.4 `std::ostream& cryomesh::state::operator<< ( std::ostream & os, const PatternChannel & obj )`

Definition at line 639 of file PatternChannel.cpp.

References `cryomesh::state::PatternChannel::length`, `cryomesh::state::PatternChannel::maxPatternListSize`, `cryomesh::state::PatternChannel::patternMap`, `cryomesh::state::PatternChannel::patternPosition`, `cryomesh::state::PatternChannel::printBinaryFormattedPatternList()`, `cryomesh::state::PatternChannel::refID`, and `cryomesh::state::PatternChannel::width`.

## 5.8 cryomesh::structures Namespace Reference

### Classes

- class [Bundle](#)  
*A [Bundle](#) is the collection of clusters and fibres, it represents the system as a whole.*
- class [Cluster](#)  
*A [Cluster](#) is a collection of self-contained nodes and connections along with an associated [Mesh](#), that can be connected up to one another.*
- class [Fibre](#)  
*A [Fibre](#) is a collection of connections that connect one structure to another.*
- class [FibreMap](#)
- class [Mesh](#)  
*[Mesh](#) is the fabric of connection space and warps and is warped by it.*
- class [NodeMesh](#)  
*[Mesh](#) of nodes and their neighbouring nodes and distances.*

### Typedefs

- typedef `std::map < boost::shared_ptr < cryomesh::components::Node > , std::map< boost::shared_ptr < cryomesh::components::Node > , double > > - NeighbourhoodMap`  
*Typedef to simplify neighbourhood map structure.*
- typedef `std::map < boost::shared_ptr < cryomesh::components::Node > , std::map< boost::shared_ptr < cryomesh::components::Node > , double > >::const_iterator NeighbourhoodMapConstIterator`  
*Typdef for iterator to neighbourhood map.*

## Functions

- `std::ostream & operator<< (std::ostream &os, const Bundle &obj)`
- `std::ostream & operator<< (std::ostream &os, const Cluster &obj)`
- `std::ostream & operator<< (std::ostream &os, const Fibre &obj)`
- `std::ostream & operator<< (std::ostream &os, const NodeMesh &obj)`

### 5.8.1 Typedef Documentation

#### 5.8.1.1 `cryomesh::structures::NodeMesh::NeighbourhoodMap`

Typedef to simplify neighbourhood map structure.

Map of Nodes to other nodes within their neighbourhood and the distances to them.

Definition at line 19 of file `NodeMesh.h`.

```
5.8.1.2 typedef std::map<boost::shared_ptr<cryomesh::components::Node>,
std::map<boost::shared_ptr<cryomesh::components::Node>, double>
>::const_iterator cryomesh::structures::NeighbourhoodMapConstIterator
```

Typdef for iterator to neighbourhood map.

Definition at line 30 of file `NodeMesh.h`.

### 5.8.2 Function Documentation

```
5.8.2.1 std::ostream& cryomesh::structures::operator<< ( std::ostream & os, const
NodeMesh & obj )
```

#### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">NodeMesh</a> & obj The object to stream

#### Returns

`std::ostream &` The output stream

Definition at line 306 of file `NodeMesh.cpp`.

References `cryomesh::structures::NodeMesh::printNeighbourhoodActivities()`, and `cryomesh::structures::NodeMesh::printNeighbourhoods()`.

```
5.8.2.2 std::ostream& cryomesh::structures::operator<< ( std::ostream & os, const Cluster &
obj )
```



## Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Cluster</a> & obj The object to stream

## Returns

std::ostream & The output stream

Definition at line 331 of file Cluster.cpp.

References [cryomesh::structures::Cluster::getConnections\(\)](#), [cryomesh::structures::Cluster::getNodeMap\(\)](#), and [cryomesh::structures::Cluster::getNodes\(\)](#).

### 5.8.2.3 std::ostream& cryomesh::structures::operator<< ( std::ostream & os, const Fibre & obj )

## Returns

Pattern To stream operator

## Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Fibre</a> & obj The object to stream

## Returns

std::ostream & The output stream

Definition at line 449 of file Fibre.cpp.

References [cryomesh::structures::Fibre::getConnections\(\)](#), and [cryomesh::structures::Fibre::getOutputNodesPattern\(\)](#).

### 5.8.2.4 std::ostream& cryomesh::structures::operator<< ( std::ostream & os, const Bundle & obj )

## Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Bundle</a> & obj The object to stream

## Returns

std::ostream & The output stream

Definition at line 973 of file Bundle.cpp.

References [cryomesh::common::Loggable::MAX](#), and [cryomesh::structures::Bundle::print\(\)](#).

## 5.9 cryomesh::utilities Namespace Reference

### Classes

- class [SequencerChannels](#)
- class [SequencerGeneric](#)
- class [Statistician](#)

*Class to draw together lots of useful statistics and monitoring data for a Bundle and its components.*

### Functions

- `std::ostream & operator<< (std::ostream &os, const SequencerGeneric &obj)`
- `std::ostream & operator<< (std::ostream &os, const Statistician &obj)`

#### 5.9.1 Function Documentation

**5.9.1.1** `std::ostream& cryomesh::utilities::operator<< ( std::ostream & os, const SequencerGeneric & obj )`

Definition at line 109 of file SequencerGeneric.cpp.

References `cryomesh::utilities::SequencerGeneric::nodeEntries`.

**5.9.1.2** `std::ostream& cryomesh::utilities::operator<< ( std::ostream & os, const Statistician & obj )`

#### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Statistician</a> & obj The object to stream

#### Returns

`std::ostream &` The output stream

Definition at line 131 of file Statistician.cpp.

References `cryomesh::utilities::Statistician::getActiveNodesPerCluster()`, `cryomesh::utilities::Statistician::getActiveNodesTotal()`, `cryomesh::utilities::Statistician::getBundleUUID()`, `cryomesh::utilities::Statistician::getClusterCount()`, `cryomesh::common::TimeKeeper::getCycle()`, `cryomesh::common::TimeKeeper::getTimeKeeper()`, `cryomesh::utilities::Statistician::getTriggeredNodesTotal()`, `cryomesh::utilities::Statistician::inputChannelsCount`, `cryomesh::utilities::Statistician::inputFibresCount`, `cryomesh::utilities::Statistician::normalFibresCount`, `cryomesh::utilities::Statistician::outputChannelsCount`, and `cryomesh::utilities::Statistician::outputFibresCount`.

## Chapter 6

# Class Documentation

### 6.1 cryomesh::state::ActivityPattern Class Reference

A simple collection of doubles representing a pattern of activities.

```
#include <ActivityPattern.h>
```

#### Public Types

- enum [ActivityComparison](#) { [GreaterThan](#), [LessThan](#), [EqualTo](#) }

#### Public Member Functions

- [ActivityPattern](#) ()
- virtual [~ActivityPattern](#) ()
- virtual std::string [toPlusBooleanString](#) () const  
*Return a string of booleans representing with each element is > 0 or not.*
- virtual std::list< bool > [toPlusBooleanList](#) () const  
*Return a vector of booleans representing with each element > 0 or not.*

#### Protected Member Functions

- std::list< bool > [toBooleanVector](#) (const double cutoff, const [ActivityComparison](#) compare) const

#### 6.1.1 Detailed Description

A simple collection of doubles representing a pattern of activities.

Definition at line 20 of file ActivityPattern.h.

## 6.1.2 Member Enumeration Documentation

### 6.1.2.1 enum cryomesh::state::ActivityPattern::ActivityComparison

Enumerator:

***GreaterThan***

***LessThan***

***EqualTo***

Definition at line 22 of file ActivityPattern.h.

## 6.1.3 Constructor & Destructor Documentation

### 6.1.3.1 cryomesh::state::ActivityPattern::ActivityPattern ( )

Definition at line 18 of file ActivityPattern.cpp.

### 6.1.3.2 cryomesh::state::ActivityPattern::~~ActivityPattern ( ) [virtual]

Definition at line 22 of file ActivityPattern.cpp.

## 6.1.4 Member Function Documentation

### 6.1.4.1 std::list< bool > cryomesh::state::ActivityPattern::toBooleanVector ( const double *cutoff*, const ActivityComparison *compare* ) const [protected]

Definition at line 35 of file ActivityPattern.cpp.

References EqualTo, GreaterThan, and LessThan.

Referenced by toPlusBooleanList().

### 6.1.4.2 std::list< bool > cryomesh::state::ActivityPattern::toPlusBooleanList ( ) const [virtual]

Return a vector of booleans representing with each element > 0 or not.

Returns

std::vector<bool> The vector of booleans

Definition at line 31 of file ActivityPattern.cpp.

References GreaterThan, and toBooleanVector().

Referenced by toPlusBooleanString().

6.1.4.3 `std::string cryomesh::state::ActivityPattern::toPlusBooleanString ( ) const`  
`[virtual]`

Return a string of booleans representing with each element is > 0 or not.

#### Returns

`std::string` The string of booleans

Definition at line 25 of file `ActivityPattern.cpp`.

References `toPlusBooleanList()`.

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

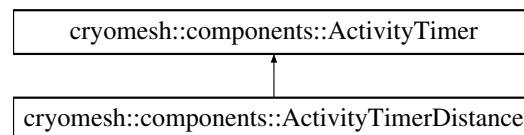
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/ActivityPattern.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/ActivityPattern.cpp`

## 6.2 cryomesh::components::ActivityTimer Class Reference

Simple interface class for activity timers.

```
#include <ActivityTimer.h>
```

Inheritance diagram for `cryomesh::components::ActivityTimer`:



### Public Member Functions

- `ActivityTimer ( )`  
*Default constructor.*
- `virtual ~ActivityTimer ( )`  
*Default destructor.*
- `virtual void reset ()=0`

### Protected Member Functions

- `virtual std::ostream & print (std::ostream &os) const =0`

### Friends

- `std::ostream & operator<< (std::ostream &os, const ActivityTimer &obj)`  
*To stream operator.*

### 6.2.1 Detailed Description

Simple interface class for activity timers.

Definition at line 20 of file ActivityTimer.h.

### 6.2.2 Constructor & Destructor Documentation

#### 6.2.2.1 `cryomesh::components::ActivityTimer::ActivityTimer ( )` [inline]

Default constructor.

Definition at line 25 of file ActivityTimer.h.

#### 6.2.2.2 `virtual cryomesh::components::ActivityTimer::~~ActivityTimer ( )` [inline, virtual]

Default destructor.

Definition at line 31 of file ActivityTimer.h.

### 6.2.3 Member Function Documentation

#### 6.2.3.1 `virtual std::ostream& cryomesh::components::ActivityTimer::print ( std::ostream & os ) const` [protected, pure virtual]

Implemented in [cryomesh::components::ActivityTimerDistance](#).

#### 6.2.3.2 `virtual void cryomesh::components::ActivityTimer::reset ( )` [pure virtual]

Implemented in [cryomesh::components::ActivityTimerDistance](#).

### 6.2.4 Friends And Related Function Documentation

#### 6.2.4.1 `std::ostream& operator<< ( std::ostream & os, const ActivityTimer & obj )` [friend]

To stream operator.

##### Parameters

<code>std::ostream</code>	& os The output stream
<code>const</code>	<a href="#">ActivityTimer</a> & obj The object to stream

**Returns**

std::ostream & The output stream

Definition at line 47 of file ActivityTimer.h.

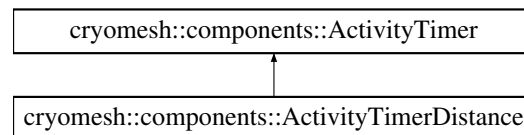
The documentation for this class was generated from the following file:

- /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/[ActivityTimer.h](#)

**6.3 cryomesh::components::ActivityTimerDistance Class Reference**

```
#include <ActivityTimerDistance.h>
```

Inheritance diagram for cryomesh::components::ActivityTimerDistance:

**Public Member Functions**

- [ActivityTimerDistance](#) ()  
*Constructor.*
- [ActivityTimerDistance](#) (double dist, double dec)  
*Construct from an initial distance, and a rate of decrement.*
- virtual [~ActivityTimerDistance](#) ()
- [ActivityTimerDistance](#) & [operator=](#) (const [ActivityTimerDistance](#) &obj)  
*Assignment operator.*
- bool [operator<](#) (const [ActivityTimerDistance](#) &obj) const  
*Less than operator.*
- bool [operator>](#) (const [ActivityTimerDistance](#) &obj) const  
*Greater than operator.*
- [ActivityTimerDistance](#) & [operator--](#) ()  
*Prefix increment operator.*
- [ActivityTimerDistance](#) [operator--](#) (int)  
*Postfix decrement operator.*
- const [ActivityTimerDistance](#) [operator+](#) (const [ActivityTimerDistance](#) &obj) const  
*Non-destructive addition operator.*
- [ActivityTimerDistance](#) & [operator+=](#) (const [ActivityTimerDistance](#) &obj)  
*Destructive addition and assignment operator.*
- double [getDelay](#) () const  
*Get the delay.*

- double [getStartingDelay](#) () const  
*Get the starting delay.*
- double [getDecrement](#) () const  
*Get the decrement.*
- void [reset](#) ()  
*Reset the countdown.*
- virtual bool [checkConstraints](#) () const
- virtual void [enableDebug](#) (bool b)

### Static Public Member Functions

- static boost::shared\_ptr < [ActivityTimerDistance](#) > [getRandom](#) ()  
*Get a random object.*

### Static Public Attributes

- static const double [MIN\\_DECREMENT\\_FRACTION](#) = 0.1
- static const double [MAX\\_DECREMENT\\_FRACTION](#) = 1
- static const double [MIN\\_DISTANCE](#) = 1.0
- static const double [MAX\\_DISTANCE](#) = 100.0

### Protected Member Functions

- virtual std::ostream & [print](#) (std::ostream &os) const

### Private Attributes

- double [distance](#)
- double [distance\\_remaining](#)
- double [decrement](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [ActivityTimer](#) &obj)  
*To stream operator.*

## 6.3.1 Detailed Description

Definition at line 19 of file ActivityTimerDistance.h.



### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 cryomesh::components::ActivityTimerDistance::ActivityTimerDistance ( )

Constructor.

Definition at line 36 of file ActivityTimerDistance.cpp.

Referenced by getRandom().

#### 6.3.2.2 cryomesh::components::ActivityTimerDistance::ActivityTimerDistance ( double *dist*, double *dec* )

Construct from an initial distance, and a rate of decrement.

Definition at line 40 of file ActivityTimerDistance.cpp.

#### 6.3.2.3 cryomesh::components::ActivityTimerDistance::~~ActivityTimerDistance ( ) [virtual]

Definition at line 44 of file ActivityTimerDistance.cpp.

### 6.3.3 Member Function Documentation

#### 6.3.3.1 bool cryomesh::components::ActivityTimerDistance::checkConstraints ( ) const [virtual]

Definition at line 113 of file ActivityTimerDistance.cpp.

References getDecrement(), getStartingDelay(), and MIN\_DISTANCE.

#### 6.3.3.2 void cryomesh::components::ActivityTimerDistance::enableDebug ( bool *b* ) [virtual]

Definition at line 101 of file ActivityTimerDistance.cpp.

#### 6.3.3.3 double cryomesh::components::ActivityTimerDistance::getDecrement ( ) const

Get the decrement.

Returns

double The decrement of the timer

Definition at line 98 of file ActivityTimerDistance.cpp.

References decrement.

Referenced by checkConstraints(), and print().

#### 6.3.3.4 `double cryomesh::components::ActivityTimerDistance::getDelay ( ) const`

Get the delay.

##### Returns

double The distance delay

Definition at line 90 of file ActivityTimerDistance.cpp.

References distance\_remaining.

Referenced by print().

#### 6.3.3.5 `boost::shared_ptr< ActivityTimerDistance > cryomesh- ::components::ActivityTimerDistance::getRandom ( ) [static]`

Get a random object.

##### Returns

boost::shared\_ptr<ActivityTimerDistance> The random object

Definition at line 22 of file ActivityTimerDistance.cpp.

References ActivityTimerDistance(), MAX\_DECREMENT\_FRACTION, MAX\_DISTANCE, MIN\_DECREMENT\_FRACTION, and MIN\_DISTANCE.

#### 6.3.3.6 `double cryomesh::components::ActivityTimerDistance::getStartingDelay ( ) const`

Get the starting delay.

##### Returns

double The starting distance delay

Definition at line 94 of file ActivityTimerDistance.cpp.

References distance.

Referenced by checkConstraints(), and print().

**6.3.3.7** `const ActivityTimerDistance cryomesh::components::ActivityTimerDistance::operator+ ( const ActivityTimerDistance & obj )`  
`const`

Non-destructive addition operator.

#### Parameters

<code>const</code>	<a href="#">ActivityTimerDistance</a> & obj RHS addition
--------------------	--

#### Returns

[ActivityTimerDistance](#) New object after addition

Definition at line 76 of file ActivityTimerDistance.cpp.

**6.3.3.8** `ActivityTimerDistance & cryomesh::components::ActivityTimerDistance::operator+= ( const ActivityTimerDistance & obj )`

Destructive addition and assignment operator.

#### Parameters

<code>const</code>	<a href="#">ActivityTimerDistance</a> & obj RHS addition
--------------------	--

#### Returns

[ActivityTimerDistance](#) & This object after addition and assignment

Definition at line 82 of file ActivityTimerDistance.cpp.

References decrement, distance, and distance\_remaining.

**6.3.3.9** `ActivityTimerDistance & cryomesh::components::ActivityTimerDistance::operator-- ( )`

Prefix increment operator.

#### Returns

[ActivityTimerDistance](#) & Return this

Definition at line 62 of file ActivityTimerDistance.cpp.

References decrement, and distance\_remaining.

**6.3.3.10** `ActivityTimerDistance cryomesh::components::ActivityTimerDistance::operator-- ( int )`

Postfix decrement operator.

**Returns**

[ActivityTimerDistance](#) & Return this

Definition at line 70 of file ActivityTimerDistance.cpp.

**6.3.3.11** `bool cryomesh::components::ActivityTimerDistance::operator< ( const ActivityTimerDistance & obj ) const`

Less than operator.

**Parameters**

<i>const</i>	<a href="#">ActivityTimerDistance</a> & obj RHS
--------------	---

**Returns**

bool True if < than obj, false otherwise

Definition at line 54 of file ActivityTimerDistance.cpp.

References distance.

**6.3.3.12** `ActivityTimerDistance & cryomesh::components::ActivityTimerDistance::operator= ( const ActivityTimerDistance & obj )`

Assignment operator.

**Parameters**

<i>const</i>	<a href="#">ActivityTimerDistance</a> & obj RHS assignment
--------------	--

**Returns**

[ActivityTimerDistance](#) & This object after assignment

Definition at line 47 of file ActivityTimerDistance.cpp.

References decrement, distance, and distance\_remaining.

**6.3.3.13** `bool cryomesh::components::ActivityTimerDistance::operator> ( const ActivityTimerDistance & obj ) const`

Greater than operator.

**Parameters**

<i>const</i>	<a href="#">ActivityTimerDistance</a> & obj RHS
--------------	---

**Returns**

bool True if > than obj, false otherwise

Definition at line 58 of file ActivityTimerDistance.cpp.

References `distance_remaining`.

**6.3.3.14** `std::ostream & cryomesh::components::ActivityTimerDistance::print (`  
`std::ostream & os ) const` `[protected, virtual]`

Implements [cryomesh::components::ActivityTimer](#).

Definition at line 107 of file ActivityTimerDistance.cpp.

References `getDecrement()`, `getDelay()`, and `getStartingDelay()`.

**6.3.3.15** `void cryomesh::components::ActivityTimerDistance::reset ( )`  
`[virtual]`

Reset the countdown.

Implements [cryomesh::components::ActivityTimer](#).

Definition at line 104 of file ActivityTimerDistance.cpp.

References `distance`, and `distance_remaining`.

**6.3.4 Friends And Related Function Documentation**

**6.3.4.1** `std::ostream& operator<< ( std::ostream & os, const ActivityTimer & obj )`  
`[friend, inherited]`

To stream operator.

**Parameters**

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">ActivityTimer</a> & obj The object to stream

**Returns**

std::ostream & The output stream

Definition at line 47 of file ActivityTimer.h.

**6.3.5 Member Data Documentation**

#### 6.3.5.1 `double cryomesh::components::ActivityTimerDistance::decrement` [private]

Definition at line 196 of file ActivityTimerDistance.h.

Referenced by `getDecrement()`, `operator+=()`, `operator--()`, and `operator=()`.

#### 6.3.5.2 `double cryomesh::components::ActivityTimerDistance::distance` [private]

Definition at line 181 of file ActivityTimerDistance.h.

Referenced by `getStartingDelay()`, `operator+=()`, `operator<()`, `operator=()`, and `reset()`.

#### 6.3.5.3 `double cryomesh::components::ActivityTimerDistance::distance_` `remaining` [private]

Definition at line 188 of file ActivityTimerDistance.h.

Referenced by `getDelay()`, `operator+=()`, `operator--()`, `operator=()`, `operator>()`, and `reset()`.

#### 6.3.5.4 `const double cryomesh::components::ActivityTimerDistance::MAX_DECR-` `EMENT_FRACTION = 1` [static]

Definition at line 156 of file ActivityTimerDistance.h.

Referenced by `getRandom()`.

#### 6.3.5.5 `const double cryomesh::components::ActivityTimerDistance::MAX_DISTA-` `NCE = 100.0` [static]

Definition at line 170 of file ActivityTimerDistance.h.

Referenced by `getRandom()`.

#### 6.3.5.6 `const double cryomesh::components::ActivityTimerDistance::MIN_DECRE-` `MENT_FRACTION = 0.1` [static]

Definition at line 149 of file ActivityTimerDistance.h.

Referenced by `getRandom()`.

#### 6.3.5.7 `const double cryomesh::components::ActivityTimerDistance::MIN_DISTA-` `NCE = 1.0` [static]

Definition at line 163 of file ActivityTimerDistance.h.

Referenced by `checkConstraints()`, `getRandom()`, and `cryomesh::components::Connection::updatePosition()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ActivityTimer-Distance.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ActivityTimer-Distance.cpp`

## 6.4 cryomesh::state::BinaryString Class Reference

```
#include <BinaryString.h>
```

### Public Types

- enum `Type` { `TXT`, `BIN` }

### Public Member Functions

- `template<class Archive > void serialize (Archive &ar, const unsigned int version)`
- `BinaryString ()`
- `BinaryString (const std::string &str, bool sign_bit=false, Type tp=TXT)`
- `BinaryString (const std::vector< bool > &binvec, bool sign_bit)`
- `BinaryString (const BinaryString &obj)`
- `virtual ~BinaryString ()`
- `void setSignBit (bool b)`
- `bool getSignBit () const`
- `const std::string & getBinaryString () const`
- `const std::vector< bool > getBools () const`
- `void setBinaryString (const std::string &str)`
- `void setBinaryString (const std::vector< bool > &binvec)`
- `bool isValidBinary () const`
- `bool isAllZeroes () const`
- `unsigned int getWidth () const`
- `std::list< int > toInts () const`
- `std::string toText () const`
- `std::string resize (const unsigned int size)`
- `int toInt () const`

### Static Public Member Functions

- static std::list< int > [formatTextToInts](#) (const std::string &str)
- static std::string [formatIntsToText](#) (const std::list< int > &charints)
- static std::list< [BinaryString](#) > [formatTextToBinaryStrings](#) (const std::string &str, std::string &allbins)
- static std::string [formatBinaryStringsToText](#) (const std::list< std::string > &strs)

### Static Public Attributes

- static const unsigned int [BINARY\\_CHAR\\_LENGTH](#) = 8
- static const unsigned int [MAX\\_BINARY\\_INTEGER\\_SIZE](#) = 32

### Static Private Member Functions

- static std::string [charToBinaryString](#) (const char &ch, bool sign\_bit)

### Private Attributes

- std::string [binaryString](#)
- bool [signBit](#)

### Friends

- class [boost::serialization::access](#)
- std::ostream & [operator<<](#) (std::ostream &os, const [BinaryString](#) &obj)

## 6.4.1 Detailed Description

Definition at line 21 of file [BinaryString.h](#).

## 6.4.2 Member Enumeration Documentation

### 6.4.2.1 enum cryomesh::state::BinaryString::Type

Enumerator:

***TXT***

***BIN***

Definition at line 29 of file [BinaryString.h](#).



### 6.4.3 Constructor & Destructor Documentation

#### 6.4.3.1 cryomesh::state::BinaryString::BinaryString ( )

Definition at line 21 of file BinaryString.cpp.

#### 6.4.3.2 cryomesh::state::BinaryString::BinaryString ( const std::string & *str*, bool *sign\_bit* = false, Type *tp* = TXT )

Definition at line 25 of file BinaryString.cpp.

References BIN, BINARY\_CHAR\_LENGTH, binaryString, charToBinaryString(), formatTextToInts(), getSignBit(), and resize().

#### 6.4.3.3 cryomesh::state::BinaryString::BinaryString ( const std::vector< bool > & *binvec*, bool *sign\_bit* )

Definition at line 59 of file BinaryString.cpp.

References binaryString.

#### 6.4.3.4 cryomesh::state::BinaryString::BinaryString ( const BinaryString & *obj* )

Definition at line 78 of file BinaryString.cpp.

References binaryString, getBinaryString(), getSignBit(), and signBit.

#### 6.4.3.5 cryomesh::state::BinaryString::~BinaryString ( ) [virtual]

Definition at line 83 of file BinaryString.cpp.

### 6.4.4 Member Function Documentation

#### 6.4.4.1 std::string cryomesh::state::BinaryString::charToBinaryString ( const char & *ch*, bool *sign\_bit* ) [static, private]

Definition at line 327 of file BinaryString.cpp.

References BIN, BINARY\_CHAR\_LENGTH, and resize().

Referenced by BinaryString(), and formatTextToBinaryStrings().

#### 6.4.4.2 static std::string cryomesh::state::BinaryString::formatBinaryStringsToText ( const std::list< std::string > & *strs* ) [static]

**6.4.4.3** `std::string cryomesh::state::BinaryString::formatIntsToText ( const std::list< int > & charints ) [static]`

Definition at line 290 of file BinaryString.cpp.

**6.4.4.4** `std::list< BinaryString > cryomesh::state::BinaryString::formatTextToBinaryStrings ( const std::string & str, std::string & allbins ) [static]`

Definition at line 304 of file BinaryString.cpp.

References BIN, and charToBinaryString().

Referenced by toInts().

**6.4.4.5** `std::list< int > cryomesh::state::BinaryString::formatTextToInts ( const std::string & str ) [static]`

Definition at line 277 of file BinaryString.cpp.

Referenced by BinaryString().

**6.4.4.6** `const std::string & cryomesh::state::BinaryString::getBinaryString ( ) const`

Definition at line 93 of file BinaryString.cpp.

References binaryString.

Referenced by BinaryString(), cryomesh::state::operator<<(), resize(), toInts(), and toText().

**6.4.4.7** `const std::vector< bool > cryomesh::state::BinaryString::getBools ( ) const`

Definition at line 97 of file BinaryString.cpp.

References binaryString, and isValidBinary().

Referenced by cryomesh::state::Pattern::getPattern().

**6.4.4.8** `bool cryomesh::state::BinaryString::getSignBit ( ) const`

Definition at line 90 of file BinaryString.cpp.

References signBit.

Referenced by BinaryString(), resize(), toInt(), and toText().

**6.4.4.9** `unsigned int cryomesh::state::BinaryString::getWidth ( ) const`

Definition at line 177 of file BinaryString.cpp.

References binaryString.

Referenced by cryomesh::state::Pattern::getSize().

#### 6.4.4.10 bool cryomesh::state::BinaryString::isAllZeroes ( ) const

Definition at line 162 of file BinaryString.cpp.

References binaryString.

Referenced by cryomesh::state::Pattern::isAllZeroes().

#### 6.4.4.11 bool cryomesh::state::BinaryString::isValidBinary ( ) const

Definition at line 147 of file BinaryString.cpp.

References binaryString.

Referenced by getBools(), and toInt().

#### 6.4.4.12 std::string cryomesh::state::BinaryString::resize ( const unsigned int *size* )

Definition at line 374 of file BinaryString.cpp.

References binaryString, getBinaryString(), and getSignBit().

Referenced by BinaryString(), charToBinaryString(), and toText().

#### 6.4.4.13 template<class Archive > void cryomesh::state::BinaryString::serialize ( Archive & *ar*, const unsigned int *version* ) [inline]

Definition at line 25 of file BinaryString.h.

References binaryString, and signBit.

#### 6.4.4.14 void cryomesh::state::BinaryString::setBinaryString ( const std::string & *str* )

Definition at line 125 of file BinaryString.cpp.

References binaryString.

Referenced by cryomesh::state::Pattern::setPattern().

#### 6.4.4.15 void cryomesh::state::BinaryString::setBinaryString ( const std::vector< bool > & *binvec* )

Definition at line 128 of file BinaryString.cpp.

References binaryString.

#### 6.4.4.16 void cryomesh::state::BinaryString::setSignBit ( bool *b* )

Definition at line 87 of file BinaryString.cpp.

References signBit.

#### 6.4.4.17 int cryomesh::state::BinaryString::toInt ( ) const

Definition at line 181 of file BinaryString.cpp.

References binaryString, getSignBit(), isValidBinary(), MAX\_BINARY\_INTEGER\_SIZE, and signBit.

Referenced by toText().

#### 6.4.4.18 std::list< int > cryomesh::state::BinaryString::toInts ( ) const

Definition at line 227 of file BinaryString.cpp.

References formatTextToBinaryStrings(), and getBinaryString().

#### 6.4.4.19 std::string cryomesh::state::BinaryString::toText ( ) const

Definition at line 242 of file BinaryString.cpp.

References BIN, BINARY\_CHAR\_LENGTH, binaryString, getBinaryString(), getSignBit(), resize(), and toInt().

### 6.4.5 Friends And Related Function Documentation

#### 6.4.5.1 friend class boost::serialization::access [friend]

Definition at line 23 of file BinaryString.h.

#### 6.4.5.2 std::ostream& operator<< ( std::ostream & *os*, const BinaryString & *obj* ) [friend]

Definition at line 223 of file BinaryString.cpp.

### 6.4.6 Member Data Documentation

#### 6.4.6.1 const unsigned int cryomesh::state::BinaryString::BINARY\_CHAR\_LENGTH = 8 [static]

Definition at line 61 of file BinaryString.h.

Referenced by BinaryString(), charToBinaryString(), and toText().

6.4.6.2 `std::string cryomesh::state::BinaryString::binaryString` [private]

Definition at line 64 of file BinaryString.h.

Referenced by `BinaryString()`, `getBinaryString()`, `getBools()`, `getWidth()`, `isAllZeroes()`, `isValidBinary()`, `resize()`, `serialize()`, `setBinaryString()`, `toInt()`, and `toText()`.

6.4.6.3 `const unsigned int cryomesh::state::BinaryString::MAX_BINARY_INTEGER_SIZE = 32` [static]

Definition at line 62 of file BinaryString.h.

Referenced by `toInt()`.

6.4.6.4 `bool cryomesh::state::BinaryString::signBit` [private]

Definition at line 65 of file BinaryString.h.

Referenced by `BinaryString()`, `getSignBit()`, `serialize()`, `setSignBit()`, and `toInt()`.

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

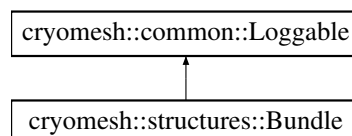
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/BinaryString.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/BinaryString.cpp](#)

## 6.5 cryomesh::structures::Bundle Class Reference

A [Bundle](#) is the collection of clusters and fibres, it represents the system as a whole.

```
#include <Bundle.h>
```

Inheritance diagram for `cryomesh::structures::Bundle`:



### Public Types

- enum [LoggingDepth](#) { [SUMMARY](#), [MAX](#) }  
*Enum representing print detail.*

### Public Member Functions

- [Bundle](#) ()

*Default constructor.*

- virtual [~Bundle](#) ()

*Default destructor.*

- void [update](#) ()

*Update all bundle components.*

- virtual boost::shared\_ptr < [Cluster](#) > [createCluster](#) (int nodeSize, int nodeConnectivity)

*Create a cluster with a size and connectivity.*

- virtual boost::shared\_ptr< [Fibre](#) > [connectCluster](#) (boost::uuids::uuid inputClusterUUID, boost::uuids::uuid outputClusterUUID, int fibreWidth)

*Connect clusters specified by uuids with a fibre of width.*

- virtual boost::shared\_ptr< [Fibre](#) > [connectCluster](#) (boost::uuids::uuid clusterUUID, const [Fibre::FibreType](#) &type, int fibreWidth)

*Connect clusters specified by uuids with a fibre of width.*

- virtual boost::shared\_ptr< [Fibre](#) > [connectPrimaryInputCluster](#) (boost::uuids::uuid patchanid, boost::uuids::uuid clusterUUID)

*Helper access function for specialised connection.*

- virtual boost::shared\_ptr< [Fibre](#) > [connectPrimaryOutputCluster](#) (boost::uuids::uuid patchanid, boost::uuids::uuid clusterUUID)

*Helper access function for specialised connection.*

- std::vector< boost::shared\_ptr < [Fibre](#) > > [autoConnectPrimaryInputClusters](#) (const std::vector< boost::uuids::uuid > &cluster\_uuids)

- std::vector< boost::shared\_ptr < [Fibre](#) > > [autoConnectPrimaryOutputClusters](#) (const std::vector< boost::uuids::uuid > &cluster\_uuids)

- virtual std::vector < boost::shared\_ptr< [Fibre](#) > > [autoConnectPrimaryInputClusters](#) (std::vector< boost::shared\_ptr< [Cluster](#) > > list)

- virtual std::vector < boost::shared\_ptr< [Fibre](#) > > [autoConnectPrimaryOutputClusters](#) (std::vector< boost::shared\_ptr< [Cluster](#) > > list)

- virtual std::vector < boost::shared\_ptr < [state::PatternChannel](#) > > [getDisconnectedRealInputPatternChannels](#) ()

- virtual std::vector < boost::shared\_ptr < [state::PatternChannel](#) > > [getDisconnectedRealOutputPatternChannels](#) ()

- virtual boost::shared\_ptr< [Fibre](#) > [connectLoopbackCluster](#) (boost::uuids::uuid clusterUUID, int fibreWidth)

*Helper access function for specialised connection.*

- virtual void [loadChannels](#) (const std::string &ifstr)

*Load in the pattern channels from a filename of a pattern dataset.*

- const ClusterMap & [getClusters](#) () const

*Get the clusters in this bundle.*

- const [FibreMap](#) & [getFibres](#) () const

*Get the mutable clusters in this bundle.*

- const [FibreMap](#) & [getInputFibres](#) () const

*Get the input fibres of this bundle.*

- [FibreMap](#) & [getMutableInputFibres](#) ()

*Get the mutable input fibres of this bundle.*

- const [FibreMap](#) & [getOutputFibres](#) () const  
*Get the output fibres of this bundle.*
- [FibreMap](#) & [getMutableOutputFibres](#) ()  
*Get the mutable output fibres of this bundle.*
- const [state::PatternChannelMap](#) & [getRealInputChannelsMap](#) () const  
*Get the real input pattern channels of this bundle.*
- const [state::PatternChannelMap](#) & [getRealOutputChannelsMap](#) () const  
*Get the real output pattern channels of this bundle.*
- const [state::PatternChannelMap](#) & [getActualInputChannelsMap](#) () const  
*Get the actual input pattern channels of this bundle.*
- const [state::PatternChannelMap](#) & [getActualOutputChannelsMap](#) () const  
*Get the real output pattern channels of this bundle.*
- const std::map < boost::uuids::uuid, boost::uuids::uuid > & [getRealFibre-PatternChannelMap](#) () const  
*Get the map of fibres to their associated pattern channels.*
- const std::map < boost::uuids::uuid, boost::uuids::uuid > & [getActualFibre-PatternChannelMap](#) () const
- const boost::shared\_ptr < [utilities::Statistician](#) > [getStatistician](#) () const  
*Get the Statistician.*
- boost::shared\_ptr < [utilities::Statistician](#) > [getMutableStatistician](#) ()
- double [getEnergy](#) () const  
*Get the last calculated energy of the bundle.*
- virtual bool [checkStructure](#) () const  
*Run a system structure check.*
- virtual bool [checkFibreStructure](#) () const  
*Check the structure of Fibres.*
- virtual bool [checkChannelStructure](#) () const  
*Check the structure of Channels.*
- virtual void [enableDebug](#) (bool b)
- virtual void [enableDebugClusters](#) (bool b)
- virtual void [enableDebugFibres](#) (bool b)
- virtual std::ostream & [print](#) (std::ostream &os, const [common::Loggable::Logging-Depth](#) depth=SUMMARY) const  
*Print the bundle to stream.*
- std::ostream & [printChannels](#) (std::ostream &os) const  
*Print the channels to stream.*
- std::ostream & [printFibreMaps](#) (std::ostream &os) const
- std::ostream & [printFibreMap](#) (std::ostream &os, const std::map< boost::uuids::uuid, boost::uuids::uuid > &fibre\_map) const

## Protected Member Functions

- virtual void [updatePrimaryInputFibres](#) ()  
*Get the next patterns from channels and apply them to their mapped fibres.*
- virtual void [updatePrimaryOutputFibres](#) ()  
*Get the patterns from primary output fibres and apply them to their mapped pattern channels.*
- virtual double [matchOutputChannelsSum](#) () const  
*Compare the output channels of primary output fibres to expected output channel patterns.*
- virtual void [updateStatistician](#) ()  
*Update the statistician if debugging is enabled.*
- void [setEnergy](#) (double d)  
*Set the energy of the bundle.*
- ClusterMap & [getMutableClusters](#) ()
- template<class T >  
std::ostream & [printSearch](#) (std::ostream &os, const boost::uuids::uuid &uuid, const std::map< boost::uuids::uuid, boost::shared\_ptr< T > > &map) const  
*Print out a uuid search.*

## Private Member Functions

- const boost::shared\_ptr < [state::PatternChannel](#) > [getRealPrimaryInputChannelByFibre](#) (const boost::uuids::uuid fibre\_uuid) const
- const boost::shared\_ptr < [state::PatternChannel](#) > [getRealPrimaryOutputChannelByFibre](#) (const boost::uuids::uuid fibre\_uuid) const
- const boost::shared\_ptr < [state::PatternChannel](#) > [getActualPrimaryInputChannelByFibre](#) (const boost::uuids::uuid fibre\_uuid) const
- const boost::shared\_ptr < [state::PatternChannel](#) > [getActualPrimaryOutputChannelByFibre](#) (const boost::uuids::uuid fibre\_uuid) const
- const boost::shared\_ptr< [Fibre](#) > [getPrimaryInputFibreByRealChannel](#) (const boost::uuids::uuid pattern\_channel\_uuid) const  
*Helper method to take a uuid and find its correspondingly mapped object Take an input PatternChannel uuid and find the input Fibre its mapped to.*
- const boost::shared\_ptr< [Fibre](#) > [getPrimaryOutputFibreByRealChannel](#) (const boost::uuids::uuid pattern\_channel\_uuid) const  
*Helper method to take a uuid and find its correspondingly mapped object Take an output PatternChannel uuid and find the output Fibre its mapped to.*
- const boost::shared\_ptr< [Fibre](#) > [getPrimaryInputFibreByActualChannel](#) (const boost::uuids::uuid pattern\_channel\_uuid) const
- const boost::shared\_ptr< [Fibre](#) > [getPrimaryOutputFibreByActualChannel](#) (const boost::uuids::uuid pattern\_channel\_uuid) const
- const boost::shared\_ptr< [Fibre](#) > [getPrimaryFibreByChannel](#) (const boost::uuids::uuid id, const [FibreMap](#) &map, const std::map< boost::uuids::uuid, boost::uuids::uuid > &fibrepattern\_channelmap) const



*Helper method to take a uuid and find its correspondingly mapped object Take an channel uuid and find the [Fibre](#) its mapped to inside the supplied map.*

- `const boost::shared_ptr < state::PatternChannel > getPrimaryChannelByFibre`  
`(const boost::uuids::uuid id, const state::PatternChannelMap &map, const std::map< boost::uuids::uuid, boost::uuids::uuid > &fibrepattern_channelmap)`  
`const`

*Helper method to take a uuid and find its correspondingly mapped object Take an [Fibre](#) uuid and find the [PatternChannel](#) its mapped to inside the supplied map.*

## Private Attributes

- `ClusterMap` [clusters](#)
- `FibreMap` [fibres](#)
- `state::PatternChannelMap` [realInputChannelsMap](#)
- `state::PatternChannelMap` [realOutputChannelsMap](#)
- `state::PatternChannelMap` [actualInputChannelsMap](#)
- `state::PatternChannelMap` [actualOutputChannelsMap](#)
- `FibreMap` [inputFibres](#)
- `FibreMap` [outputFibres](#)
- `boost::shared_ptr < utilities::Statistician > statistician`

*Statistics object to generate useful info on the bundle.*

- `double` [energy](#)

*Last energy calculation of the output channel matching.*

- `std::map< boost::uuids::uuid, boost::uuids::uuid > realFibrePatternChannelMap`
- `std::map< boost::uuids::uuid, boost::uuids::uuid > actualFibrePatternChannelMap`
- `Cluster::EnergyFractionMethod` [energyFractionMethod](#)

## Friends

- `std::ostream & operator<< (std::ostream &os, const Bundle &obj)`

*To stream operator.*

### 6.5.1 Detailed Description

A [Bundle](#) is the collection of clusters and fibres, it represents the system as a whole.

Definition at line 29 of file `Bundle.h`.

### 6.5.2 Member Enumeration Documentation

#### 6.5.2.1 `enum cryomesh::common::Loggable::LoggingDepth` [[inherited](#)]

Enum representing print detail.

Enumerator:

***SUMMARY***

***MAX***

Definition at line 23 of file Loggable.h.

### 6.5.3 Constructor & Destructor Documentation

#### 6.5.3.1 cryomesh::structures::Bundle::Bundle ( )

Default constructor.

Definition at line 17 of file Bundle.cpp.

#### 6.5.3.2 cryomesh::structures::Bundle::~~Bundle ( ) [virtual]

Default destructor.

Definition at line 23 of file Bundle.cpp.

### 6.5.4 Member Function Documentation

#### 6.5.4.1 std::vector< boost::shared\_ptr< Fibre > > cryomesh::structures::Bundle::autoConnectPrimaryInputClusters ( const std::vector< boost::uuids::uuid > & cluster\_uuids )

Definition at line 206 of file Bundle.cpp.

References getMutableClusters().

#### 6.5.4.2 std::vector< boost::shared\_ptr< Fibre > > cryomesh::structures::Bundle::autoConnectPrimaryInputClusters ( std::vector< boost::shared\_ptr< Cluster > > list ) [virtual]

Definition at line 254 of file Bundle.cpp.

References connectPrimaryInputCluster(), and getDisconnectedRealInputPatternChannels().

#### 6.5.4.3 std::vector< boost::shared\_ptr< Fibre > > cryomesh::structures::Bundle::autoConnectPrimaryOutputClusters ( const std::vector< boost::uuids::uuid > & cluster\_uuids )

Definition at line 231 of file Bundle.cpp.

References getMutableClusters().

**6.5.4.4** `std::vector< boost::shared_ptr< Fibre > > cryomesh::structures::Bundle::autoConnectPrimaryOutputClusters ( std::vector< boost::shared_ptr< Cluster > > list ) [virtual]`

Definition at line 301 of file Bundle.cpp.

References `connectPrimaryOutputCluster()`, and `getDisconnectedRealOutputPatternChannels()`.

**6.5.4.5** `bool cryomesh::structures::Bundle::checkChannelStructure ( ) const [virtual]`

Check the structure of Channels.

#### Returns

bool True if structure tests pass, false otherwise

Definition at line 558 of file Bundle.cpp.

Referenced by `checkStructure()`.

**6.5.4.6** `bool cryomesh::structures::Bundle::checkFibreStructure ( ) const [virtual]`

Check the structure of Fibres.

#### Returns

bool True if structure tests pass, false otherwise

Definition at line 477 of file Bundle.cpp.

References `getClusters()`, `getFibres()`, `getInputFibres()`, and `getOutputFibres()`.

Referenced by `checkStructure()`.

**6.5.4.7** `bool cryomesh::structures::Bundle::checkStructure ( ) const [virtual]`

Run a system structure check.

#### Returns

bool True if system passes all tests, false otherwise

Definition at line 561 of file Bundle.cpp.

References `checkChannelStructure()`, and `checkFibreStructure()`.

6.5.4.8 `boost::shared_ptr< Fibre > cryomesh::structures::Bundle::connectCluster ( boost::uuids::uuid inputClusterUUID, boost::uuids::uuid outputClusterUUID, int fibreWidth ) [virtual]`

Connect clusters specified by uuids with a fibre of width.

#### Parameters

<code>boost::uuids::uuid</code>	inputClusterUUID UUID of input cluster
<code>boost::uuids::uuid</code>	outputClusterUUID UUID of output cluster
<code>int</code>	width Width of fibre to create

#### Returns

The new fibre created, possible null

Definition at line 61 of file Bundle.cpp.

References clusters, and fibres.

Referenced by connectLoopbackCluster(), connectPrimaryInputCluster(), and connectPrimaryOutputCluster().

6.5.4.9 `boost::shared_ptr< Fibre > cryomesh::structures::Bundle::connectCluster ( boost::uuids::uuid clusterUUID, const Fibre::FibreType & type, int fibreWidth ) [virtual]`

Connect clusters specified by uuids with a fibre of width.

#### Parameters

<code>boost::uuids::uuid</code>	clusterUUID UUID of cluster to connect to fibre
<code>const</code>	<a href="#">Fibre::FibreType</a> & type Type of fibre connection to make
<code>int</code>	width Width of fibre to create

#### Returns

The new fibre created, possible null

Definition at line 78 of file Bundle.cpp.

References clusters, fibres, inputFibres, cryomesh::structures::Fibre::LoopbackFibre, outputFibres, cryomesh::structures::Fibre::PrimaryInputFibre, and cryomesh::structures::Fibre::PrimaryOutputFibre.

6.5.4.10 `boost::shared_ptr< Fibre > cryomesh::structures::Bundle::connect-LoopbackCluster ( boost::uuids::uuid clusterUUID, int fibreWidth )`  
`[virtual]`

Helper access function for specialised connection.

#### Parameters

<i>const</i>	<a href="#">Fibre::FibreType</a> & type Type of fibre connection to make
<i>int</i>	width Width of fibre to create

#### Returns

The new fibre created, possible null

Definition at line 392 of file Bundle.cpp.

References `connectCluster()`, and `cryomesh::structures::Fibre::LoopbackFibre`.

6.5.4.11 `boost::shared_ptr< Fibre > cryomesh::structures::Bundle::connect-PrimaryInputCluster ( boost::uuids::uuid patchanid, boost::uuids::uuid clusterUUID )`  
`[virtual]`

Helper access function for specialised connection.

#### Parameters

<i>boost::uuids- ::uuid</i>	PatternChannel to map the fibre to
<i>const</i>	<a href="#">Fibre::FibreType</a> & type Type of fibre connection to make

#### Returns

The new fibre created, possible null

Definition at line 110 of file Bundle.cpp.

References `actualFibrePatternChannelMap`, `actualInputChannelsMap`, `connectCluster()`, `cryomesh::state::PatternChannel::Input`, `cryomesh::structures::Fibre::PrimaryInputFibre`, `realFibrePatternChannelMap`, and `realInputChannelsMap`.

Referenced by `autoConnectPrimaryInputClusters()`.

6.5.4.12 `boost::shared_ptr< Fibre > cryomesh::structures::Bundle::connect-PrimaryOutputCluster ( boost::uuids::uuid patchanid, boost::uuids::uuid clusterUUID )`  
`[virtual]`

Helper access function for specialised connection.

## Parameters

<i>boost::uuids- ::uuid</i>	PatternChannel to map the fibre to
<i>const</i>	<a href="#">Fibre::FibreType</a> & type Type of fibre connection to make

## Returns

The new fibre created, possible null

Definition at line 157 of file Bundle.cpp.

References [actualFibrePatternChannelMap](#), [actualOutputChannelsMap](#), [connectCluster\(\)](#), [cryomesh::state::PatternChannel::Output](#), [cryomesh::structures::Fibre::PrimaryOutputFibre](#), [realFibrePatternChannelMap](#), and [realOutputChannelsMap](#).

Referenced by [autoConnectPrimaryOutputClusters\(\)](#).

**6.5.4.13** `boost::shared_ptr< Cluster > cryomesh::structures::Bundle::createCluster  
( int nodeSize, int nodeConnectivity ) [virtual]`

Create a cluster with a size and connectivity.

## Parameters

<i>int</i>	The number of nodes to create
<i>int</i>	The connectivity of the nodes

## Returns

`boost::shared_ptr<Cluster>` The cluster that was created

Definition at line 54 of file Bundle.cpp.

References [clusters](#), and [energyFractionMethod](#).

**6.5.4.14** `void cryomesh::structures::Bundle::enableDebug ( bool b ) [virtual]`

Definition at line 412 of file Bundle.cpp.

References [enableDebugClusters\(\)](#), and [enableDebugFibres\(\)](#).

**6.5.4.15** `void cryomesh::structures::Bundle::enableDebugClusters ( bool b )  
[virtual]`

Definition at line 417 of file Bundle.cpp.

References [clusters](#).

Referenced by [enableDebug\(\)](#).

**6.5.4.16** `void cryomesh::structures::Bundle::enableDebugFibres ( bool b )`  
`[virtual]`

Definition at line 420 of file Bundle.cpp.

References fibres.

Referenced by enableDebug().

**6.5.4.17** `const std::map< boost::uuids::uuid, boost::uuids::uuid > &`  
`cryomesh::structures::Bundle::getActualFibrePatternChannelMap ( )`  
`const`

Definition at line 463 of file Bundle.cpp.

References actualFibrePatternChannelMap.

Referenced by printFibreMaps().

**6.5.4.18** `const state::PatternChannelMap & cryomesh::structures::Bundle::get-`  
`ActualInputChannelsMap ( ) const`

Get the actual input pattern channels of this bundle.

Returns

PatternChannelMap The map of actual input pattern channels of this bundle

Definition at line 453 of file Bundle.cpp.

References actualInputChannelsMap.

**6.5.4.19** `const state::PatternChannelMap & cryomesh::structures-`  
`::Bundle::getActualOutputChannelsMap ( )`  
`const`

Get the real output pattern channels of this bundle.

Returns

PatternChannelMap The map of actual output pattern channels of this bundle

Definition at line 456 of file Bundle.cpp.

References actualOutputChannelsMap.

**6.5.4.20** `const boost::shared_ptr< state::PatternChannel > cryomesh::structures:-`  
`Bundle::getActualPrimaryInputChannelByFibre ( const boost::uuids::uuid`  
`fibre_uuid ) const` `[private]`

Definition at line 711 of file Bundle.cpp.

References `actualFibrePatternChannelMap`, `actualInputChannelsMap`, and `getPrimaryChannelByFibre()`.

Referenced by `updatePrimaryInputFibres()`.

```
6.5.4.21  const boost::shared_ptr< state::PatternChannel > cryomesh::structures::-
          Bundle::getActualPrimaryOutputChannelByFibre ( const boost::uuids::uuid
          fibre_uuid ) const [private]
```

Definition at line 715 of file `Bundle.cpp`.

References `actualFibrePatternChannelMap`, `actualOutputChannelsMap`, and `getPrimaryChannelByFibre()`.

Referenced by `matchOutputChannelsSum()`, and `updatePrimaryOutputFibres()`.

```
6.5.4.22  const ClusterMap & cryomesh::structures::Bundle::getClusters ( ) const
```

Get the clusters in this bundle.

#### Returns

ClusterMap The map of clusters in this bundle

Definition at line 408 of file `Bundle.cpp`.

References `clusters`.

Referenced by `checkFibreStructure()`, `cryomesh::utilities::Statistician::getActiveNodesPerCluster()`, `cryomesh::utilities::Statistician::getTriggeredNodesPerCluster()`, `print()`, and `cryomesh::utilities::Statistician::update()`.

```
6.5.4.23  std::vector< boost::shared_ptr< state::PatternChannel > > cryomesh-
          ::structures::Bundle::getDisconnectedRealInputPatternChannels ( )
          [virtual]
```

Definition at line 342 of file `Bundle.cpp`.

References `getPrimaryInputFibreByRealChannel()`, and `realInputChannelsMap`.

Referenced by `autoConnectPrimaryInputClusters()`.

```
6.5.4.24  std::vector< boost::shared_ptr< state::PatternChannel > > cryomesh-
          ::structures::Bundle::getDisconnectedRealOutputPatternChannels ( )
          [virtual]
```

Definition at line 368 of file `Bundle.cpp`.

References `getPrimaryOutputFibreByRealChannel()`, and `realOutputChannelsMap`.

Referenced by `autoConnectPrimaryOutputClusters()`.



**6.5.4.25 double cryomesh::structures::Bundle::getEnergy ( ) const**

Get the last calculated energy of the bundle.

**Returns**

double The last calculated energy

Definition at line 470 of file Bundle.cpp.

References energy.

Referenced by update().

**6.5.4.26 const FibreMap & cryomesh::structures::Bundle::getFibres ( ) const**

Get the mutable clusters in this bundle.

**Returns**

ClusterMap The mutable map of clusters in this bundle

Definition at line 427 of file Bundle.cpp.

References fibres.

Referenced by checkFibreStructure(), print(), and cryomesh::utilities::Statistician::update().

**6.5.4.27 const FibreMap & cryomesh::structures::Bundle::getInputFibres ( ) const**

Get the input fibres of this bundle.

**Returns**

[FibreMap](#) The map of input fibres of this bundle

Definition at line 431 of file Bundle.cpp.

References inputFibres.

Referenced by checkFibreStructure(), print(), and cryomesh::utilities::Statistician::update().

**6.5.4.28 ClusterMap & cryomesh::structures::Bundle::getMutableClusters ( )  
[protected]**

Definition at line 424 of file Bundle.cpp.

References clusters.

Referenced by autoConnectPrimaryInputClusters(), and autoConnectPrimaryOutputClusters().

#### 6.5.4.29 **FibreMap & cryomesh::structures::Bundle::getMutableInputFibres ( )**

Get the mutable input fibres of this bundle.

##### Returns

[FibreMap](#) The map of mutable input fibres of this bundle

Definition at line 435 of file Bundle.cpp.

References inputFibres.

#### 6.5.4.30 **FibreMap & cryomesh::structures::Bundle::getMutableOutputFibres ( )**

Get the mutable output fibres of this bundle.

##### Returns

[FibreMap](#) The map of mutable output fibres of this bundle

Definition at line 443 of file Bundle.cpp.

References outputFibres.

#### 6.5.4.31 **boost::shared\_ptr< utilities::Statistician > cryomesh::structures::Bundle::getMutableStatistician ( )**

Definition at line 473 of file Bundle.cpp.

References statistician.

#### 6.5.4.32 **const FibreMap & cryomesh::structures::Bundle::getOutputFibres ( )** **const**

Get the output fibres of this bundle.

##### Returns

[FibreMap](#) The map of output fibres of this bundle

Definition at line 439 of file Bundle.cpp.

References outputFibres.

Referenced by checkFibreStructure(), matchOutputChannelsSum(), print(), and cryomesh::utilities::Statistician::update().

6.5.4.33 `const boost::shared_ptr< state::PatternChannel > cryomesh::structures::Bundle::getPrimaryChannelByFibre ( const boost::uuids::uuid id, const state::PatternChannelMap & map, const std::map< boost::uuids::uuid, boost::uuids::uuid > & fibrepattern_channelmap ) const [private]`

Helper method to take a uuid and find its correspondingly mapped object Take an [Fibre](#) uuid and find the PatternChannel its mapped to inside the supplied map.

#### Parameters

<code>boost::uuids::uuid</code>	The uuid of the <a href="#">Fibre</a>
<code>PatternChannelMap</code>	The map to search for a mapping from

#### Returns

`boost::shared_ptr<PatternChannel>` The PatternChannel object the [Fibre](#) with this uuid is mapped to, null if not found

Definition at line 787 of file Bundle.cpp.

Referenced by `getActualPrimaryInputChannelByFibre()`, `getActualPrimaryOutputChannelByFibre()`, `getRealPrimaryInputChannelByFibre()`, and `getRealPrimaryOutputChannelByFibre()`.

6.5.4.34 `const boost::shared_ptr< Fibre > cryomesh::structures::Bundle::getPrimaryFibreByChannel ( const boost::uuids::uuid id, const FibreMap & map, const std::map< boost::uuids::uuid, boost::uuids::uuid > & fibrepattern_channelmap ) const [private]`

Helper method to take a uuid and find its correspondingly mapped object Take an channel uuid and find the [Fibre](#) its mapped to inside the supplied map.

#### Parameters

<code>boost::uuids::uuid</code>	The uuid of the PatternChannel
<code>FibreMap</code>	The map to search for a mapping from

#### Returns

`boost::shared_ptr<Fibre>` The [Fibre](#) object the PatternChannel with this uuid is mapped to, null if not found

Definition at line 736 of file Bundle.cpp.

Referenced by `getPrimaryInputFibreByActualChannel()`, `getPrimaryInputFibreByRealChannel()`, `getPrimaryOutputFibreByActualChannel()`, and `getPrimaryOutputFibreByRealChannel()`.

```
6.5.4.35  const boost::shared_ptr< Fibre > cryomesh::structures::Bundle-
::getPrimaryInputFibreByActualChannel ( const boost::uuids::uuid
pattern_channel_uuid ) const    [private]
```

Definition at line 728 of file Bundle.cpp.

References `actualFibrePatternChannelMap`, `getPrimaryFibreByChannel()`, and `inputFibres`.

```
6.5.4.36  const boost::shared_ptr< Fibre > cryomesh::structures::Bundle-
::getPrimaryInputFibreByRealChannel ( const boost::uuids::uuid
pattern_channel_uuid ) const    [private]
```

Helper method to take a uuid and find its correspondingly mapped object Take an input `PatternChannel` uuid and find the input `Fibre` its mapped to.

#### Parameters

<i>boost::uuids- ::uuid</i>	The uuid of the input <code>PatternChannel</code>
---------------------------------	---

#### Returns

`boost::shared_ptr<Fibre>` The input `Fibre` object the input `PatternChannel` with this uuid is mapped to, null if not found

Definition at line 720 of file Bundle.cpp.

References `getPrimaryFibreByChannel()`, `inputFibres`, and `realFibrePatternChannelMap`.

Referenced by `getDisconnectedRealInputPatternChannels()`, and `updatePrimaryInputFibres()`.

```
6.5.4.37  const boost::shared_ptr< Fibre > cryomesh::structures::Bundle-
::getPrimaryOutputFibreByActualChannel ( const boost::uuids::uuid
pattern_channel_uuid ) const    [private]
```

Definition at line 732 of file Bundle.cpp.

References `actualFibrePatternChannelMap`, `getPrimaryFibreByChannel()`, and `outputFibres`.

```
6.5.4.38  const boost::shared_ptr< Fibre > cryomesh::structures::Bundle-
::getPrimaryOutputFibreByRealChannel ( const boost::uuids::uuid
pattern_channel_uuid ) const    [private]
```

Helper method to take a uuid and find its correspondingly mapped object Take an output `PatternChannel` uuid and find the output `Fibre` its mapped to.

## Parameters

<i>boost::uuids- ::uuid</i>	The uuid of the output PatternChannel
---------------------------------	---------------------------------------

## Returns

`boost::shared_ptr<Fibre>` The output [Fibre](#) object the output PatternChannel with this uuid is mapped to, null if not found

Definition at line 724 of file Bundle.cpp.

References `getPrimaryFibreByChannel()`, `outputFibres`, and `realFibrePatternChannelMap`.

Referenced by `getDisconnectedRealOutputPatternChannels()`.

```
6.5.4.39  const std::map< boost::uuids::uuid, boost::uuids::uuid > &
          cryomesh::structures::Bundle::getRealFibrePatternChannelMap ( )
          const
```

Get the map of fibres to their associated pattern channels.

## Returns

`std::map<boost::uuids::uuid, boost::uuids::uuid>` The map of fibres to their associated pattern channels

Definition at line 460 of file Bundle.cpp.

References `realFibrePatternChannelMap`.

Referenced by `printFibreMaps()`.

```
6.5.4.40  const state::PatternChannelMap & cryomesh::structures::Bundle::get-
          RealInputChannelsMap ( ) const
```

Get the real input pattern channels of this bundle.

## Returns

`PatternChannelMap` The map of real input pattern channels of this bundle

Definition at line 447 of file Bundle.cpp.

References `realInputChannelsMap`.

Referenced by `cryomesh::utilities::Statistician::update()`.

```
6.5.4.41  const state::PatternChannelMap & cryomesh::structures::Bundle::get-
          RealOutputChannelsMap ( ) const
```

Get the real output pattern channels of this bundle.

**Returns**

PatternChannelMap The map of real output pattern channels of this bundle

Definition at line 450 of file Bundle.cpp.

References realOutputChannelsMap.

Referenced by cryomesh::utilities::Statistician::update().

```
6.5.4.42  const boost::shared_ptr< state::PatternChannel > cryomesh::structures::-
          Bundle::getRealPrimaryInputChannelByFibre ( const boost::uuids::uuid
          fibre_uuid ) const    [private]
```

Definition at line 703 of file Bundle.cpp.

References getPrimaryChannelByFibre(), realFibrePatternChannelMap, and realInputChannelsMap.

```
6.5.4.43  const boost::shared_ptr< state::PatternChannel > cryomesh::structures::-
          Bundle::getRealPrimaryOutputChannelByFibre ( const boost::uuids::uuid
          fibre_uuid ) const    [private]
```

Definition at line 707 of file Bundle.cpp.

References getPrimaryChannelByFibre(), realFibrePatternChannelMap, and realOutputChannelsMap.

Referenced by matchOutputChannelsSum().

```
6.5.4.44  const boost::shared_ptr< utilities::Statistician >
          cryomesh::structures::Bundle::getStatistician ( ) const
```

Get the Statistician.

**Returns**

boost::shared\_ptr< Statistician > The current statistician, null pointer if we dont have one

Definition at line 466 of file Bundle.cpp.

References statistician.

```
6.5.4.45  void cryomesh::structures::Bundle::loadChannels ( const std::string & ifstr )
          [virtual]
```

Load in the pattern channels from a filename of a pattern dataset.

## Parameters

<code>std::string</code>	The full path filename of the pattern data set
--------------------------	--

Definition at line 396 of file Bundle.cpp.

References `cryomesh::utilities::SequencerChannels::readSequences()`, `realInputChannelsMap`, and `realOutputChannelsMap`.

**6.5.4.46** `double cryomesh::structures::Bundle::matchOutputChannelsSum ( )`  
`const` [`protected`, `virtual`]

Compare the output channels of primary output fibres to expected output channel patterns.

## Returns

double The double representing the accumulated 'energy' of all matches

Definition at line 652 of file Bundle.cpp.

References `getActualPrimaryOutputChannelByFibre()`, `getOutputFibres()`, and `getRealPrimaryOutputChannelByFibre()`.

Referenced by `update()`.

**6.5.4.47** `std::ostream & cryomesh::structures::Bundle::print ( std::ostream & os,`  
`const common::Loggable::LoggingDepth depth = SUMMARY ) const`  
`[virtual]`

Print the bundle to stream.

Implements [cryomesh::common::Loggable](#).

Definition at line 836 of file Bundle.cpp.

References `getClusters()`, `cryomesh::common::TimeKeeper::getCycle()`, `getFibres()`, `getInputFibres()`, `getOutputFibres()`, `cryomesh::common::TimeKeeper::getTimeKeeper()`, `cryomesh::common::Loggable::MAX`, `printChannels()`, `printFibreMaps()`, and `cryomesh::common::Loggable::SUMMARY`.

Referenced by `cryomesh::structures::operator<<()`.

**6.5.4.48** `std::ostream & cryomesh::structures::Bundle::printChannels ( std::ostream`  
`& os ) const`

Print the channels to stream.

Definition at line 868 of file Bundle.cpp.

References `actualInputChannelsMap`, `actualOutputChannelsMap`, `realInputChannelsMap`, and `realOutputChannelsMap`.

Referenced by `print()`.

**6.5.4.49** `std::ostream & cryomesh::structures::Bundle::printFibreMap ( std::ostream & os, const std::map< boost::uuids::uuid, boost::uuids::uuid > & fibre_map ) const`

Definition at line 959 of file Bundle.cpp.

Referenced by printFibreMaps().

**6.5.4.50** `std::ostream & cryomesh::structures::Bundle::printFibreMaps ( std::ostream & os ) const`

Definition at line 950 of file Bundle.cpp.

References getActualFibrePatternChannelMap(), getRealFibrePatternChannelMap(), and printFibreMap().

Referenced by print(), and updatePrimaryOutputFibres().

**6.5.4.51** `template<class T > std::ostream & cryomesh::structures::Bundle::printSearch ( std::ostream & os, const boost::uuids::uuid & uuid, const std::map< boost::uuids::uuid, boost::shared_ptr< T > > & map ) const [protected]`

Print out a uuid search.

Definition at line 814 of file Bundle.cpp.

**6.5.4.52** `void cryomesh::structures::Bundle::setEnergy ( double d ) [protected]`

Set the energy of the bundle.

#### Parameters

<i>double</i>	The energy to set
---------------	-------------------

Definition at line 686 of file Bundle.cpp.

References energy.

Referenced by update().

**6.5.4.53** `void cryomesh::structures::Bundle::update ( )`

Update all bundle components.

Definition at line 26 of file Bundle.cpp.

References clusters, fibres, getEnergy(), cryomesh::common::TimeKeeper::getTimeKeeper(), matchOutputChannelsSum(), setEnergy(), cryomesh::structures::FibreMap::update(), cryomesh::common::TimeKeeper::update(), updatePrimaryInputFibres(), updatePrimaryOutputFibres(), and updateStatistician().



**6.5.4.54** `void cryomesh::structures::Bundle::updatePrimaryInputFibres ( )`  
`[protected, virtual]`

Get the next patterns from channels and apply them to their mapped fibres.

Definition at line 568 of file Bundle.cpp.

References `getActualPrimaryInputChannelByFibre()`, `getPrimaryInputFibreByRealChannel()`, `inputFibres`, `realInputChannelsMap`, and `cryomesh::structures::FibreMap::update()`.

Referenced by `update()`.

**6.5.4.55** `void cryomesh::structures::Bundle::updatePrimaryOutputFibres ( )`  
`[protected, virtual]`

Get the patterns from primary output fibres and apply them to their mapped pattern channels.

Definition at line 618 of file Bundle.cpp.

References `getActualPrimaryOutputChannelByFibre()`, `outputFibres`, `printFibreMaps()`, and `cryomesh::structures::FibreMap::update()`.

Referenced by `update()`.

**6.5.4.56** `void cryomesh::structures::Bundle::updateStatistician ( )`  
`[protected, virtual]`

Update the statistician if debugging is enabled.

Definition at line 690 of file Bundle.cpp.

References `statistician`.

Referenced by `update()`.

## 6.5.5 Friends And Related Function Documentation

**6.5.5.1** `std::ostream& operator<< ( std::ostream & os, const Bundle & obj )` `[friend]`

To stream operator.

### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Bundle</a> & obj The object to stream

### Returns

`std::ostream` & The output stream

Definition at line 973 of file Bundle.cpp.

### 6.5.6 Member Data Documentation

**6.5.6.1** `std::map<boost::uuids::uuid, boost::uuids::uuid> cryomesh-  
::structures::Bundle::actualFibrePatternChannelMap`  
[private]

Definition at line 430 of file Bundle.h.

Referenced by `connectPrimaryInputCluster()`, `connectPrimaryOutputCluster()`, `getActualFibrePatternChannelMap()`, `getActualPrimaryInputChannelByFibre()`, `getActualPrimaryOutputChannelByFibre()`, `getPrimaryInputFibreByActualChannel()`, and `getPrimaryOutputFibreByActualChannel()`.

**6.5.6.2** `state::PatternChannelMap cryomesh::structures::Bundle::actualInput-  
ChannelsMap` [private]

Definition at line 385 of file Bundle.h.

Referenced by `connectPrimaryInputCluster()`, `getActualInputChannelsMap()`, `getActualPrimaryInputChannelByFibre()`, and `printChannels()`.

**6.5.6.3** `state::PatternChannelMap cryomesh::structures::Bundle::actualOutput-  
ChannelsMap` [private]

Definition at line 392 of file Bundle.h.

Referenced by `connectPrimaryOutputCluster()`, `getActualOutputChannelsMap()`, `getActualPrimaryOutputChannelByFibre()`, and `printChannels()`.

**6.5.6.4** `ClusterMap cryomesh::structures::Bundle::clusters` [private]

Definition at line 357 of file Bundle.h.

Referenced by `connectCluster()`, `createCluster()`, `enableDebugClusters()`, `getClusters()`, `getMutableClusters()`, and `update()`.

**6.5.6.5** `double cryomesh::structures::Bundle::energy` [private]

Last energy calculation of the output channel matching.

Definition at line 416 of file Bundle.h.

Referenced by `getEnergy()`, and `setEnergy()`.

**6.5.6.6** `Cluster::EnergyFractionMethod cryomesh::structures::Bundle::energy-  
FractionMethod` [private]

Definition at line 432 of file Bundle.h.

Referenced by createCluster().

#### 6.5.6.7 FibreMap cryomesh::structures::Bundle::fibres [private]

Definition at line 364 of file Bundle.h.

Referenced by connectCluster(), enableDebugFibres(), getFibres(), and update().

#### 6.5.6.8 FibreMap cryomesh::structures::Bundle::inputFibres [private]

Definition at line 399 of file Bundle.h.

Referenced by connectCluster(), getInputFibres(), getMutableInputFibres(), getPrimaryInputFibreByActualChannel(), getPrimaryInputFibreByRealChannel(), and updatePrimaryInputFibres().

#### 6.5.6.9 FibreMap cryomesh::structures::Bundle::outputFibres [private]

Definition at line 406 of file Bundle.h.

Referenced by connectCluster(), getMutableOutputFibres(), getOutputFibres(), getPrimaryOutputFibreByActualChannel(), getPrimaryOutputFibreByRealChannel(), and updatePrimaryOutputFibres().

#### 6.5.6.10 std::map<boost::uuids::uuid, boost::uuids::uuid> cryomesh::structures::Bundle::realFibrePatternChannelMap [private]

Definition at line 423 of file Bundle.h.

Referenced by connectPrimaryInputCluster(), connectPrimaryOutputCluster(), getPrimaryInputFibreByRealChannel(), getPrimaryOutputFibreByRealChannel(), getRealFibrePatternChannelMap(), getRealPrimaryInputChannelByFibre(), and getRealPrimaryOutputChannelByFibre().

#### 6.5.6.11 state::PatternChannelMap cryomesh::structures::Bundle::realInputChannelsMap [private]

Definition at line 371 of file Bundle.h.

Referenced by connectPrimaryInputCluster(), getDisconnectedRealInputPatternChannels(), getRealInputChannelsMap(), getRealPrimaryInputChannelByFibre(), loadChannels(), printChannels(), and updatePrimaryInputFibres().

#### 6.5.6.12 state::PatternChannelMap cryomesh::structures::Bundle::realOutputChannelsMap [private]

Definition at line 378 of file Bundle.h.

Referenced by `connectPrimaryOutputCluster()`, `getDisconnectedRealOutputPatternChannels()`, `getRealOutputChannelsMap()`, `getRealPrimaryOutputChannelByFibre()`, `loadChannels()`, and `printChannels()`.

#### 6.5.6.13 `boost::shared_ptr<utilities::Statistician> cryomesh::structures::Bundle::statistician` [private]

Statistics object to generate useful info on the bundle.

Definition at line 411 of file `Bundle.h`.

Referenced by `getMutableStatistician()`, `getStatistician()`, and `updateStatistician()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Bundle.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Bundle.cpp`

## 6.6 `cryomesh::structures::Cluster` Class Reference

A `Cluster` is a collection of self-contained nodes and connections along with an associated `Mesh`, that can be connected up to one another.

```
#include <Cluster.h>
```

### Public Types

- enum `EnergyFractionMethod` { `ENERGY_FRACTION_BY_CLUSTER_COUNT`, `ENERGY_FRACTION_BY_NODE_COUNT`, `ENERGY_FRACTION_NULL` }
- enum `ValueTypeSpecifier` { `AsIncrement`, `AsDecrement`, `AsMinumum`, `AsAbsolute` }

*Enum to force value type of a parameter.*

### Public Member Functions

- `Cluster` ()  
*Default constructor.*
- `Cluster` (int nodeCount, int connectivity)  
*Constructor with number of nodes and connectivity.*
- `Cluster` (int nodeCount, int connectivity, const spacial::Point bounding\_box)
- virtual `~Cluster` ()  
*Destructor.*
- void `update` ()  
*Update all elements.*
- void `updateEnergy` (double total\_energy)

- void [setEnergyFractionMethod](#) ([EnergyFractionMethod](#) method, double max\_energy\_fraction)
- [EnergyFractionMethod](#) [getEnergyFractionMethod](#) () const
- void [warpNodes](#) ()
- void [runAnalyser](#) ()
- virtual void [createConnectivity](#) (const int connectivity=1)  
*Create connections between all nodes.*
- const std::map < boost::uuids::uuid, boost::shared\_ptr < [components::Node](#) > > & [getNodes](#) () const  
*Get all nodes.*
- const std::map < boost::uuids::uuid, boost::shared\_ptr < [components::Connection](#) > > & [getConnections](#) () const  
*Get all connections.*
- const [common::Connector](#) < [Cluster](#), [Fibre](#) > & [getConnector](#) () const  
*Get the connector for this Cluster.*
- [common::Connector](#) < [Cluster](#), [Fibre](#) > & [getMutableConnector](#) ()  
*Get the mutable connector for this Cluster.*
- const [components::NodeMap](#) & [getNodeMap](#) () const  
*Get the NodeMap for this Cluster.*
- const [components::ConnectionMap](#) & [getConnectionMap](#) () const
- [components::NodeMap](#) & [getMutableNodeMap](#) ()  
*Get the mutable NodeMap for this Cluster.*
- [components::ConnectionMap](#) & [getMutableConnectionMap](#) ()
- int [getTriggeredNodeCount](#) (const int indicator=0) const  
*Get the total fired nodes in this cluster currently.*
- int [getActiveNodeCount](#) (const int indicator=0) const  
*Get the total active nodes in this cluster currently.*
- int [getLiveNodeCount](#) () const  
*Get the total live nodes in this cluster currently, ie those with at least one impulse.*
- double [getEnergy](#) () const
- void [setEnergy](#) (double d)
- double [getMaxEnergyFraction](#) () const
- const boost::shared\_ptr < [manipulators::ClusterArchitect](#) > [getClusterArchitect](#) () const
- boost::shared\_ptr < [manipulators::ClusterArchitect](#) > [getMutableClusterArchitect](#) ()
- const boost::shared\_ptr < [NodeMesh](#) > [getMesh](#) () const
- boost::shared\_ptr < [NodeMesh](#) > [getMutableMesh](#) ()
- virtual void [enableDebug](#) (bool b)

### Static Public Attributes

- static const double [SELF\\_CONNECTED\\_NODES\\_FRACTION](#) = 0.1

## Protected Member Functions

- virtual void `updateConnectivity` (const int connectivity, `ValueTypeSpecifier` as-Value=`AsIncrement`)

*Update connectivity so that each node has at least param number of connections.*

## Private Attributes

- double `energy`
- `components::NodeMap` `nodes`
- `components::ConnectionMap` `connections`
- `boost::shared_ptr< NodeMesh >` `mesh`
- `common::Connector< Cluster, Fibre >` `connector`
- `boost::shared_ptr < manipulators::ClusterArchitect >` `clusterArchitect`
- `EnergyFractionMethod` `energyFractionMethod`
- double `maxEnergyFraction`

## Friends

- `std::ostream & operator<<` (`std::ostream &os`, const `Cluster` &obj)

*To stream operator.*

### 6.6.1 Detailed Description

A `Cluster` is a collection of self-contained nodes and connections along with an associated `Mesh`, that can be connected up to one another.

Definition at line 41 of file `Cluster.h`.

### 6.6.2 Member Enumeration Documentation

#### 6.6.2.1 enum `cryomesh::structures::Cluster::EnergyFractionMethod`

Enumerator:

**`ENERGY_FRACTION_BY_CLUSTER_COUNT`**  
**`ENERGY_FRACTION_BY_NODE_COUNT`**  
**`ENERGY_FRACTION_NULL`**

Definition at line 45 of file `Cluster.h`.

### 6.6.2.2 enum cryomesh::structures::Cluster::ValueTypeSpecifier

Enum to force value type of a parameter.

Enumerator:

***AsIncrement***

***AsDecrement***

***AsMinumum***

***AsAbsolute***

Definition at line 53 of file Cluster.h.

## 6.6.3 Constructor & Destructor Documentation

### 6.6.3.1 cryomesh::structures::Cluster::Cluster ( )

Default constructor.

Definition at line 22 of file Cluster.cpp.

### 6.6.3.2 cryomesh::structures::Cluster::Cluster ( int *nodeCount*, int *connectivity* )

Constructor with number of nodes and connectivity.

Parameters

<i>int</i>	nodeCount	Number of nodes to make
<i>int</i>	connectivity	Connectivity to start with

Definition at line 35 of file Cluster.cpp.

References clusterArchitect.

### 6.6.3.3 cryomesh::structures::Cluster::Cluster ( int *nodeCount*, int *connectivity*, const spacial::Point *bounding\_box* )

Definition at line 28 of file Cluster.cpp.

References clusterArchitect.

### 6.6.3.4 cryomesh::structures::Cluster::~Cluster ( ) [virtual]

Destructor.

Definition at line 42 of file Cluster.cpp.

### 6.6.4 Member Function Documentation

**6.6.4.1** `void cryomesh::structures::Cluster::createConnectivity ( const int connectivity = 1 ) [virtual]`

Create connections between all nodes.

#### Parameters

<i>int</i>	connectivity The number of connections between each node
------------	--

Definition at line 89 of file Cluster.cpp.

References `AsIncrement`, and `updateConnectivity()`.

**6.6.4.2** `void cryomesh::structures::Cluster::enableDebug ( bool b ) [virtual]`

Definition at line 306 of file Cluster.cpp.

References `connections`, and `nodes`.

**6.6.4.3** `int cryomesh::structures::Cluster::getActiveNodeCount ( const int indicator = 0 ) const`

Get the total active nodes in this cluster currently.

#### Parameters

<i>int</i>	Set >0 for only positive active nodes, <0 for negative, 0 for all (default)
------------	---

#### Returns

`int` The total count of currently active nodes

Definition at line 263 of file Cluster.cpp.

References `getNodes()`.

**6.6.4.4** `const boost::shared_ptr< manipulators::ClusterArchitect > cryomesh::structures::Cluster::getClusterArchitect ( ) const`

Definition at line 72 of file Cluster.cpp.

References `clusterArchitect`.

**6.6.4.5** `const components::ConnectionMap & cryomesh::structures::Cluster::getConnectionMap ( ) const`

Definition at line 222 of file Cluster.cpp.



References connections.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster(), and cryomesh::manipulators::ClusterArchitect::getRandomConnections().

**6.6.4.6** `const std::map< boost::uuids::uuid, boost::shared_ptr< components::Connection > > & cryomesh::structures::Cluster::getConnections ( ) const`

Get all connections.

Returns

`std::map<boost::uuids::uuid, boost::shared_ptr< components::Connection > > -`  
Return all Connections

Definition at line 204 of file Cluster.cpp.

References connections.

Referenced by cryomesh::structures::operator<<().

**6.6.4.7** `const common::Connector< Cluster, Fibre > & cryomesh::structures::Cluster::getConnector ( ) const`

Get the connector for this [Cluster](#).

Returns

`common::Connector<Cluster, Fibre>` The connector for this [Cluster](#)

Definition at line 208 of file Cluster.cpp.

References connector.

**6.6.4.8** `double cryomesh::structures::Cluster::getEnergy ( ) const`

Definition at line 54 of file Cluster.cpp.

References energy.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster(), and cryomesh::structures::NodeMesh::warpNodes().

**6.6.4.9** `Cluster::EnergyFractionMethod cryomesh::structures::Cluster::get-EnergyFractionMethod ( ) const`

Definition at line 68 of file Cluster.cpp.

References energyFractionMethod.

#### 6.6.4.10 `int cryomesh::structures::Cluster::getLiveNodeCount ( ) const`

Get the total live nodes in this cluster currently, ie those with at least one impulse.

##### Returns

int The total count of currently live nodes

Definition at line 312 of file Cluster.cpp.

References `getNodes()`.

#### 6.6.4.11 `double cryomesh::structures::Cluster::getMaxEnergyFraction ( ) const`

Definition at line 60 of file Cluster.cpp.

References `maxEnergyFraction`.

Referenced by `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`.

#### 6.6.4.12 `const boost::shared_ptr<NodeMesh> cryomesh::structures::Cluster::getMesh ( ) const [inline]`

Definition at line 184 of file Cluster.h.

References `mesh`.

#### 6.6.4.13 `boost::shared_ptr< manipulators::ClusterArchitect > cryomesh::structures::Cluster::getMutableClusterArchitect ( )`

Definition at line 76 of file Cluster.cpp.

References `clusterArchitect`.

#### 6.6.4.14 `components::ConnectionMap & cryomesh::structures::Cluster::getMutableConnectionMap ( )`

Definition at line 225 of file Cluster.cpp.

References `connections`.

Referenced by `cryomesh::manipulators::ClusterArchitect::createConnection()`, `cryomesh::manipulators::ClusterArchitect::createRandomConnections()`, `cryomesh::manipulators::ClusterArchitect::deleteConnection()`, `cryomesh::manipulators::ClusterArchitect::destroyRandomConnections()`, and `cryomesh::manipulators::ClusterArchitect::getRandomConnections()`.

**6.6.4.15 common::Connector< Cluster, Fibre > & cryomesh::structures::Cluster::getMutableConnector ( )**

Get the mutable connector for this [Cluster](#).

**Returns**

[common::Connector<Cluster, Fibre>](#) The mutable onnector for this [Cluster](#)

Definition at line 212 of file Cluster.cpp.

References [connector](#).

**6.6.4.16 boost::shared\_ptr<NodeMesh> cryomesh::structures::Cluster::getMutableMesh ( ) [inline]**

Definition at line 187 of file Cluster.h.

References [mesh](#).

**6.6.4.17 components::NodeMap & cryomesh::structures::Cluster::getMutableNodeMap ( )**

Get the mutable NodeMap for this [Cluster](#).

**Returns**

[components::NodeMap](#) The mutable NodeMap for this [Cluster](#)

Definition at line 219 of file Cluster.cpp.

References [nodes](#).

Referenced by [cryomesh::manipulators::ClusterArchitect::createRandomConnections\(\)](#), [cryomesh::manipulators::ClusterArchitect::createRandomNodes\(\)](#), [cryomesh::manipulators::ClusterArchitect::destroyRandomNodes\(\)](#), and [cryomesh::manipulators::ClusterArchitect::getRandomNodes\(\)](#).

**6.6.4.18 const components::NodeMap & cryomesh::structures::Cluster::getNodeMap ( ) const**

Get the NodeMap for this [Cluster](#).

**Returns**

[components::NodeMap](#) The NodeMap for this [Cluster](#)

Definition at line 216 of file Cluster.cpp.

References [nodes](#).

Referenced by `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`, `cryomesh::manipulators::ClusterArchitect::createRandomNodes()`, `cryomesh::manipulators::ClusterArchitect::getRandomNodes()`, and `cryomesh::structures::operator<<()`.

**6.6.4.19** `const std::map< boost::uuids::uuid, boost::shared_ptr< components::Node > > & cryomesh::structures::Cluster::getNodes ( ) const`

Get all nodes.

#### Returns

`std::map<boost::uuids::uuid, boost::shared_ptr< components::Node > >` Return all nodes

Definition at line 200 of file `Cluster.cpp`.

References nodes.

Referenced by `getActiveNodeCount()`, `getLiveNodeCount()`, `getTriggeredNodeCount()`, `cryomesh::structures::NodeMesh::NodeMesh()`, `cryomesh::structures::operator<<()`, `cryomesh::structures::NodeMesh::regenerateNeighbourhoods()`, and `cryomesh::structures::Mesh::update()`.

**6.6.4.20** `int cryomesh::structures::Cluster::getTriggeredNodeCount ( const int indicator = 0 ) const`

Get the total fired nodes in this cluster currently.

#### Parameters

<i>int</i>	Set >0 for only positive triggered nodes, <0 for negative, 0 for all (default)
------------	--

#### Returns

`int` The total count of currently triggered nodes

Definition at line 228 of file `Cluster.cpp`.

References `getNodes()`, `cryomesh::components::Node::Negative`, `cryomesh::components::Node::None`, and `cryomesh::components::Node::Positive`.

**6.6.4.21** `void cryomesh::structures::Cluster::runAnalyser ( )`

Definition at line 85 of file `Cluster.cpp`.

References `clusterArchitect`.

**6.6.4.22 void cryomesh::structures::Cluster::setEnergy ( double *d* )**

Definition at line 57 of file Cluster.cpp.

References energy.

**6.6.4.23 void cryomesh::structures::Cluster::setEnergyFractionMethod ( EnergyFractionMethod *method*, double *max\_energy\_fraction* )**

Definition at line 63 of file Cluster.cpp.

References energyFractionMethod, and maxEnergyFraction.

**6.6.4.24 void cryomesh::structures::Cluster::update ( )**

Update all elements.

Definition at line 45 of file Cluster.cpp.

References connections, mesh, nodes, cryomesh::components::NodeMap::update(), and cryomesh::components::ConnectionMap::update().

**6.6.4.25 void cryomesh::structures::Cluster::updateConnectivity ( const int *connectivity*, ValueTypeSpecifier *asValue* = AsIncrement ) [protected, virtual]**

Update connectivity so that each node has at least param number of connections.

**Parameters**

<i>int</i>	connectivity The least connectivity to ensure
------------	---

Definition at line 93 of file Cluster.cpp.

References AsIncrement, AsMinumum, clusterArchitect, nodes, and SELF\_CONNECTED\_NODES\_FRACTION.

Referenced by createConnectivity().

**6.6.4.26 void cryomesh::structures::Cluster::updateEnergy ( double *total\_energy* )****6.6.4.27 void cryomesh::structures::Cluster::warpNodes ( )**

Definition at line 80 of file Cluster.cpp.

References mesh.

**6.6.5 Friends And Related Function Documentation**

**6.6.5.1** `std::ostream& operator<< ( std::ostream & os, const Cluster & obj )` `[friend]`

To stream operator.

#### Parameters

<code>std::ostream</code>	& <i>os</i> The output stream
<code>const</code>	<a href="#">Cluster</a> & <i>obj</i> The object to stream

#### Returns

`std::ostream &` The output stream

Definition at line 331 of file `Cluster.cpp`.

### 6.6.6 Member Data Documentation

**6.6.6.1** `boost::shared_ptr<manipulators::ClusterArchitect>`  
`cryomesh::structures::Cluster::clusterArchitect` `[private]`

Definition at line 247 of file `Cluster.h`.

Referenced by `Cluster()`, `getClusterArchitect()`, `getMutableClusterArchitect()`, `runAnalyser()`, and `updateConnectivity()`.

**6.6.6.2** `components::ConnectionMap` `cryomesh::structures::Cluster::connections` `[private]`

Definition at line 231 of file `Cluster.h`.

Referenced by `enableDebug()`, `getConnectionMap()`, `getConnections()`, `getMutableConnectionMap()`, and `update()`.

**6.6.6.3** `common::Connector<Cluster, Fibre>` `cryomesh::structures::Cluster::connector` `[private]`

Definition at line 245 of file `Cluster.h`.

Referenced by `getConnector()`, and `getMutableConnector()`.

**6.6.6.4** `double` `cryomesh::structures::Cluster::energy` `[private]`

Definition at line 217 of file `Cluster.h`.

Referenced by `getEnergy()`, and `setEnergy()`.

**6.6.6.5 EnergyFractionMethod cryomesh::structures::Cluster::energyFractionMethod** [private]

Definition at line 248 of file Cluster.h.

Referenced by `getEnergyFractionMethod()`, and `setEnergyFractionMethod()`.

**6.6.6.6 double cryomesh::structures::Cluster::maxEnergyFraction** [private]

Definition at line 249 of file Cluster.h.

Referenced by `getMaxEnergyFraction()`, and `setEnergyFractionMethod()`.

**6.6.6.7 boost::shared\_ptr<NodeMesh> cryomesh::structures::Cluster::mesh** [private]

Definition at line 238 of file Cluster.h.

Referenced by `getMesh()`, `getMutableMesh()`, `update()`, and `warpNodes()`.

**6.6.6.8 components::NodeMap cryomesh::structures::Cluster::nodes** [private]

Definition at line 224 of file Cluster.h.

Referenced by `enableDebug()`, `getMutableNodeMap()`, `getNodeMap()`, `getNodes()`, `update()`, and `updateConnectivity()`.

**6.6.6.9 const double cryomesh::structures::Cluster::SELF\_CONNECTED\_NODES\_FRACTION = 0.1** [static]

Definition at line 204 of file Cluster.h.

Referenced by `updateConnectivity()`.

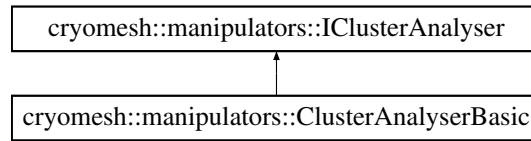
The documentation for this class was generated from the following files:

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Cluster.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Cluster.cpp`

**6.7 cryomesh::manipulators::ClusterAnalyserBasic Class Reference**

```
#include <ClusterAnalyserBasic.h>
```

Inheritance diagram for `cryomesh::manipulators::ClusterAnalyserBasic`:



## Public Types

- enum [EnergyVariation](#) { [NONE](#) = 2048, [OUT\\_OF\\_RANGE\\_POSITIVE](#) = 1024, [HIGH\\_POSITIVE](#) = 512, [MEDIUM\\_POSITIVE](#) = 256, [SMALL\\_POSITIVE](#) = 128, [STAGNANT\\_POSITIVE](#) = 64, [ZERO](#) = 32, [STAGNANT\\_NEGATIVE](#) = 16, [SMALL\\_NEGATIVE](#) = 8, [MEDIUM\\_NEGATIVE](#) = 4, [HIGH\\_NEGATIVE](#) = 2, [OUT\\_OF\\_RANGE\\_NEGATIVE](#) = 1 }

## Public Member Functions

- [ClusterAnalyserBasic](#) (const [ClusterArchitect](#) &ca)
- virtual [~ClusterAnalyserBasic](#) ()
- virtual [ClusterAnalysisData](#) [analyseCluster](#) (const [structures::Cluster](#) &cluster, const std::map< int, std::list< [ClusterAnalysisData](#) > > &histories)  
*Run an analysis on the cluster to decide what action to take on nodes and connections.*
- virtual [ClusterAnalysisData](#) [calculateRangeEnergies](#) (const std::list< [ClusterAnalysisData](#) > &history) const  
*Calculate the range energies stats.*
- int [getConnectionCreationEnabledCountdown](#) () const
- int [getConnectionDestructionEnabledCountdown](#) () const
- int [getNodeCreationEnabledCountdown](#) () const
- int [getNodeDestructionEnabledCountdown](#) () const
- void [setConnectionDestructionEnabledCountdown](#) (int connectionDestructionEnabledCountdown)
- const [RestructuringCountdown](#) & [getConnectionRestructuring](#) () const
- const [RestructuringCountdown](#) & [getNodeRestructuring](#) () const

## Protected Member Functions

- virtual [EnergyVariationWeightingMap](#) [getEnergyVariationMap](#) (const double energy\_input, double range) const

## Protected Attributes

- [RestructuringCountdown](#) [nodeRestructuring](#)
- [RestructuringCountdown](#) [connectionRestructuring](#)



## Private Attributes

- const [ClusterArchitect](#) & `clusterArchitect`

### 6.7.1 Detailed Description

Definition at line 18 of file ClusterAnalyserBasic.h.

### 6.7.2 Member Enumeration Documentation

#### 6.7.2.1 enum cryomesh::manipulators::IClusterAnalyser::EnergyVariation [inherited]

Enumerator:

***NONE***  
***OUT\_OF\_RANGE\_POSITIVE***  
***HIGH\_POSITIVE***  
***MEDIUM\_POSITIVE***  
***SMALL\_POSITIVE***  
***STAGNANT\_POSITIVE***  
***ZERO***  
***STAGNANT\_NEGATIVE***  
***SMALL\_NEGATIVE***  
***MEDIUM\_NEGATIVE***  
***HIGH\_NEGATIVE***  
***OUT\_OF\_RANGE\_NEGATIVE***

Definition at line 136 of file IClusterAnalyser.h.

### 6.7.3 Constructor & Destructor Documentation

#### 6.7.3.1 cryomesh::manipulators::ClusterAnalyserBasic::ClusterAnalyserBasic ( const ClusterArchitect & ca )

Definition at line 16 of file ClusterAnalyserBasic.cpp.

#### 6.7.3.2 cryomesh::manipulators::ClusterAnalyserBasic::~~ClusterAnalyserBasic ( ) [virtual]

Definition at line 20 of file ClusterAnalyserBasic.cpp.

## 6.7.4 Member Function Documentation

**6.7.4.1 ClusterAnalysisData cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster** ( const structures::Cluster & *cluster*, const std::map< int, std::list< ClusterAnalysisData > > & *histories* ) [virtual]

Run an analysis on the cluster to decide what action to take on nodes and connections.

### Parameters

<i>const</i>	<a href="#">structures::Cluster</a> & The cluster to analyse
<i>const</i>	std::list<ClusterAnalysisData> The historical analysis data to use

### Returns

[ClusterAnalysisData](#) The resulting analytical data

Implements [cryomesh::manipulators::IClusterAnalyser](#).

Definition at line 23 of file ClusterAnalyserBasic.cpp.

References [calculateRangeEnergies\(\)](#), [clusterArchitect](#), [cryomesh::manipulators::IClusterAnalyser::connectionRestructuring](#), [cryomesh::manipulators::ClusterAnalysisData::RangeEnergy::energyFraction](#), [cryomesh::manipulators::ClusterAnalysisData::getClusterRangeEnergy\(\)](#), [cryomesh::structures::Cluster::getConnectionMap\(\)](#), [cryomesh::structures::Cluster::getEnergy\(\)](#), [getEnergyVariationMap\(\)](#), [cryomesh::manipulators::ClusterArchitect::getHistorySteppingFactor\(\)](#), [cryomesh::structures::Cluster::getMaxEnergyFraction\(\)](#), [cryomesh::manipulators::ClusterArchitect::getMaxHistorySize\(\)](#), [cryomesh::structures::Cluster::getNodeMap\(\)](#), [cryomesh::manipulators::IClusterAnalyser::HIGH\\_NEGATIVE](#), [cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::isAnyLongRestructuringEnabled\(\)](#), [cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::isAnyMediumRestructuringEnabled\(\)](#), [cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::isAnyRestructuringEnabled\(\)](#), [cryomesh::manipulators::IClusterAnalyser::MEDIUM\\_NEGATIVE](#), [cryomesh::manipulators::IClusterAnalyser::MEDIUM\\_POSITIVE](#), [cryomesh::manipulators::IClusterAnalyser::nodeRestructuring](#), [cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::setLongCountdown\(\)](#), [cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::setMediumCountdown\(\)](#), [cryomesh::manipulators::IClusterAnalyser::SMALL\\_NEGATIVE](#), [cryomesh::manipulators::IClusterAnalyser::SMALL\\_POSITIVE](#), [cryomesh::manipulators::IClusterAnalyser::STAGNANT\\_NEGATIVE](#), [cryomesh::manipulators::IClusterAnalyser::STAGNANT\\_POSITIVE](#), and [cryomesh::manipulators::IClusterAnalyser::EnergyVariationWeightingMap::variationMap](#).

**6.7.4.2 ClusterAnalysisData cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies** ( const std::list< ClusterAnalysisData > & *history* ) const [virtual]

Calculate the range energies stats.

## Parameters

<code>const</code>	<code>std::list&lt;ClusterAnalysisData&gt;</code> & The history range to work with
--------------------	--

## Returns

[ClusterAnalysisData](#) The resulting cluster analysis data

Implements [cryomesh::manipulators::IClusterAnalyser](#).

Definition at line 252 of file ClusterAnalyserBasic.cpp.

References `cryomesh::manipulators::ClusterAnalysisData::getClusterRangeEnergy()`, `cryomesh::manipulators::ClusterAnalysisData::getConnectionCreationWeight()`, `cryomesh::manipulators::ClusterAnalysisData::getConnectionDestructionWeight()`, `cryomesh::manipulators::ClusterAnalysisData::getConnectionsToCreate()`, `cryomesh::manipulators::ClusterAnalysisData::getConnectionsToDestroy()`, `cryomesh::manipulators::ClusterAnalysisData::getNodeCreationWeight()`, `cryomesh::manipulators::ClusterAnalysisData::getNodeDestructionWeight()`, `cryomesh::manipulators::ClusterAnalysisData::getNodesToCreate()`, and `cryomesh::manipulators::ClusterAnalysisData::getNodesToDestroy()`.

Referenced by `analyseCluster()`.

**6.7.4.3** `int cryomesh::manipulators::ClusterAnalyserBasic::getConnectionCreationEnabledCountdown ( ) const`

**6.7.4.4** `int cryomesh::manipulators::ClusterAnalyserBasic::getConnectionDestructionEnabledCountdown ( ) const`

**6.7.4.5** `const RestructuringCountdown& cryomesh::manipulators::IClusterAnalyser::getConnectionRestructuring ( ) const` `[inline, inherited]`

Definition at line 198 of file IClusterAnalyser.h.

References `cryomesh::manipulators::IClusterAnalyser::connectionRestructuring`.

**6.7.4.6** `ClusterAnalyserBasic::EnergyVariationWeightingMap cryomesh::manipulators::ClusterAnalyserBasic::getEnergyVariationMap ( const double energy_input, double range ) const` `[protected, virtual]`

Implements [cryomesh::manipulators::IClusterAnalyser](#).

Definition at line 307 of file ClusterAnalyserBasic.cpp.

References `cryomesh::manipulators::IClusterAnalyser::HIGH_NEGATIVE`, `cryomesh::manipulators::IClusterAnalyser::HIGH_POSITIVE`, `cryomesh::manipulators::IClusterAnalyser::MEDIUM_NEGATIVE`, `cryomesh::manipulators::IClusterAnalyser::MEDIUM_POSITIVE`, and `cryomesh::manipulators::IClusterAnalyser::OUT_OF_RANGE_`

NEGATIVE, cryomesh::manipulators::IClusterAnalyser::OUT\_OF\_RANGE\_POSITIVE, cryomesh::manipulators::IClusterAnalyser::SMALL\_NEGATIVE, cryomesh::manipulators::IClusterAnalyser::SMALL\_POSITIVE, cryomesh::manipulators::IClusterAnalyser::STAGNANT\_NEGATIVE, cryomesh::manipulators::IClusterAnalyser::STAGNANT\_POSITIVE, cryomesh::manipulators::IClusterAnalyser::EnergyVariationWeightingMap::variationMap, and cryomesh::manipulators::IClusterAnalyser::ZERO.

Referenced by analyseCluster().

**6.7.4.7** `int cryomesh::manipulators::ClusterAnalyserBasic::getNodeCreationEnabledCountdown ( ) const`

**6.7.4.8** `int cryomesh::manipulators::ClusterAnalyserBasic::getNodeDestructionEnabledCountdown ( ) const`

**6.7.4.9** `const RestructuringCountdown& cryomesh::manipulators::IClusterAnalyser::getNodeRestructuring ( ) const` `[inline, inherited]`

Definition at line 201 of file IClusterAnalyser.h.

References cryomesh::manipulators::IClusterAnalyser::nodeRestructuring.

**6.7.4.10** `void cryomesh::manipulators::ClusterAnalyserBasic::setConnectionDestructionEnabledCountdown ( int connectionDestructionEnabledCountdown )`

## 6.7.5 Member Data Documentation

**6.7.5.1** `const ClusterArchitect& cryomesh::manipulators::ClusterAnalyserBasic::clusterArchitect` `[private]`

Definition at line 36 of file ClusterAnalyserBasic.h.

Referenced by analyseCluster().

**6.7.5.2** `RestructuringCountdown cryomesh::manipulators::IClusterAnalyser::connectionRestructuring` `[protected, inherited]`

Definition at line 206 of file IClusterAnalyser.h.

Referenced by analyseCluster(), and cryomesh::manipulators::IClusterAnalyser::getConnectionRestructuring().

### 6.7.5.3 RestructuringCountdown cryomesh::manipulators::I-ClusterAnalyser::nodeRestructuring [protected, inherited]

Definition at line 205 of file IClusterAnalyser.h.

Referenced by analyseCluster(), and cryomesh::manipulators::IClusterAnalyser::getNodeRestructuring().

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

- /home/niall/Projects/Eclipse/Cpp/cryomesh/src/manipulators/ClusterAnalyser-Basic.h
- /home/niall/Projects/Eclipse/Cpp/cryomesh/src/manipulators/ClusterAnalyser-Basic.cpp

## 6.8 cryomesh::manipulators::ClusterAnalysisData Class Reference

```
#include <ClusterAnalysisData.h>
```

### Classes

- struct [RangeEnergy](#)  
*Struct representing the value extrapolated over a history range.*

### Public Member Functions

- [ClusterAnalysisData](#) ()
- [ClusterAnalysisData](#) ([RangeEnergy](#) cre)
- [ClusterAnalysisData](#) ([RangeEnergy](#) clusterRangeEnergy, double node\_creation\_weight, double node\_destruction\_weight, double conn\_creation\_weight, double conn\_destruction\_weight, int node\_create, int nodes\_destroy, int conn\_create, int conn\_destroy)
- [ClusterAnalysisData](#) (const [ClusterAnalysisData](#) &obj)
- virtual [~ClusterAnalysisData](#) ()
- [RangeEnergy](#) getClusterRangeEnergy () const
- double getNodeCreationWeight () const
- double getNodeDestructionWeight () const
- double getConnectionCreationWeight () const
- double getConnectionDestructionWeight () const
- int getNodesToDestroy () const
- int getConnectionsToDestroy () const
- int getNodesToCreate () const
- int getConnectionsToCreate () const
- void setClusterRangeEnergy ([RangeEnergy](#) clusterEnergy)
- void setConnectionCreationWeight (double connectionCreationWeight)

- void [setConnectionDestructionWeight](#) (double [connectionDestructionWeight](#))
- void [setConnectionsToCreate](#) (int [connectionsToCreate](#))
- void [setConnectionsToDestroy](#) (int [connectionsToDestroy](#))
- void [setNodeCreationWeight](#) (double [nodeCreationWeight](#))
- void [setNodeDestructionWeight](#) (double [nodeDestructionWeight](#))
- void [setNodesToCreate](#) (int [nodesToCreate](#))
- void [setNodesToDestroy](#) (int [nodesToDestroy](#))

### Private Attributes

- [RangeEnergy](#) [clusterRangeEnergy](#)
- double [nodeCreationWeight](#)
- double [nodeDestructionWeight](#)
- double [connectionCreationWeight](#)
- double [connectionDestructionWeight](#)
- int [nodesToCreate](#)
- int [nodesToDestroy](#)
- int [connectionsToCreate](#)
- int [connectionsToDestroy](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [ClusterAnalysisData](#) &obj)  
*To stream operator.*

## 6.8.1 Detailed Description

Definition at line 18 of file ClusterAnalysisData.h.

## 6.8.2 Constructor & Destructor Documentation

### 6.8.2.1 cryomesh::manipulators::ClusterAnalysisData::ClusterAnalysisData ( )

Definition at line 13 of file ClusterAnalysisData.cpp.

### 6.8.2.2 cryomesh::manipulators::ClusterAnalysisData::ClusterAnalysisData ( [RangeEnergy](#) *cre* )

Definition at line 17 of file ClusterAnalysisData.cpp.

**6.8.2.3** cryomesh::manipulators::ClusterAnalysisData::ClusterAnalysisData  
 ( RangeEnergy clusterRangeEnergy, double node\_creation\_weight,  
 double node\_destruction\_weight, double conn\_creation\_weight, double  
 conn\_destruction\_weight, int node\_create, int nodes\_destroy, int conn\_create, int  
 conn\_destroy )

Definition at line 22 of file ClusterAnalysisData.cpp.

**6.8.2.4** cryomesh::manipulators::ClusterAnalysisData::ClusterAnalysisData (   
 const ClusterAnalysisData & obj )

Definition at line 31 of file ClusterAnalysisData.cpp.

**6.8.2.5** cryomesh::manipulators::ClusterAnalysisData::~~ClusterAnalysisData (   
 ) [virtual]

Definition at line 38 of file ClusterAnalysisData.cpp.

### 6.8.3 Member Function Documentation

**6.8.3.1** ClusterAnalysisData::RangeEnergy cryomesh::manipulators-  
 ::ClusterAnalysisData::getClusterRangeEnergy ( )  
 const

Definition at line 41 of file ClusterAnalysisData.cpp.

References clusterRangeEnergy.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster(), and  
 cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies().

**6.8.3.2** double cryomesh::manipulators::ClusterAnalysisData::getConnection-  
 CreationWeight ( ) const

Definition at line 50 of file ClusterAnalysisData.cpp.

References connectionCreationWeight.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::calculateRange-  
 Energies().

**6.8.3.3** double cryomesh::manipulators::ClusterAnalysisData::getConnection-  
 DestructionWeight ( ) const

Definition at line 53 of file ClusterAnalysisData.cpp.

References connectionDestructionWeight.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies().

#### 6.8.3.4 int cryomesh::manipulators::ClusterAnalysisData::getConnectionsToCreate ( ) const

Definition at line 66 of file ClusterAnalysisData.cpp.

References connectionsToCreate.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies(), and cryomesh::manipulators::ClusterArchitect::runAnalysis().

#### 6.8.3.5 int cryomesh::manipulators::ClusterAnalysisData::getConnectionsToDestroy ( ) const

Definition at line 60 of file ClusterAnalysisData.cpp.

References connectionsToDestroy.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies(), and cryomesh::manipulators::ClusterArchitect::runAnalysis().

#### 6.8.3.6 double cryomesh::manipulators::ClusterAnalysisData::getNodeCreationWeight ( ) const

Definition at line 44 of file ClusterAnalysisData.cpp.

References nodeCreationWeight.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies().

#### 6.8.3.7 double cryomesh::manipulators::ClusterAnalysisData::getNodeDestructionWeight ( ) const

Definition at line 47 of file ClusterAnalysisData.cpp.

References nodeDestructionWeight.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies().

#### 6.8.3.8 int cryomesh::manipulators::ClusterAnalysisData::getNodesToCreate ( ) const

Definition at line 63 of file ClusterAnalysisData.cpp.

References nodesToCreate.



Referenced by cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies(), and cryomesh::manipulators::ClusterArchitect::runAnalysis().

**6.8.3.9** `int cryomesh::manipulators::ClusterAnalysisData::getNodesToDestroy ( ) const`

Definition at line 57 of file ClusterAnalysisData.cpp.

References nodesToDestroy.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::calculateRangeEnergies(), and cryomesh::manipulators::ClusterArchitect::runAnalysis().

**6.8.3.10** `void cryomesh::manipulators::ClusterAnalysisData::setClusterRangeEnergy ( RangeEnergy clusterEnergy )`

Definition at line 70 of file ClusterAnalysisData.cpp.

References clusterRangeEnergy.

**6.8.3.11** `void cryomesh::manipulators::ClusterAnalysisData::setConnectionCreationWeight ( double connectionCreationWeight )`

Definition at line 74 of file ClusterAnalysisData.cpp.

References connectionCreationWeight.

**6.8.3.12** `void cryomesh::manipulators::ClusterAnalysisData::setConnectionDestructionWeight ( double connectionDestructionWeight )`

Definition at line 78 of file ClusterAnalysisData.cpp.

References connectionDestructionWeight.

**6.8.3.13** `void cryomesh::manipulators::ClusterAnalysisData::setConnectionsToCreate ( int connectionsToCreate )`

Definition at line 82 of file ClusterAnalysisData.cpp.

References connectionsToCreate.

6.8.3.14 **void cryomesh::manipulators::ClusterAnalysisData::setConnectionsToDestroy ( int *connectionsToDestroy* )**

Definition at line 86 of file ClusterAnalysisData.cpp.

References `connectionsToDestroy`.

6.8.3.15 **void cryomesh::manipulators::ClusterAnalysisData::setNodeCreationWeight ( double *nodeCreationWeight* )**

Definition at line 90 of file ClusterAnalysisData.cpp.

References `nodeCreationWeight`.

6.8.3.16 **void cryomesh::manipulators::ClusterAnalysisData::setNodeDestructionWeight ( double *nodeDestructionWeight* )**

Definition at line 94 of file ClusterAnalysisData.cpp.

References `nodeDestructionWeight`.

6.8.3.17 **void cryomesh::manipulators::ClusterAnalysisData::setNodesToCreate ( int *nodesToCreate* )**

Definition at line 98 of file ClusterAnalysisData.cpp.

References `nodesToCreate`.

6.8.3.18 **void cryomesh::manipulators::ClusterAnalysisData::setNodesToDestroy ( int *nodesToDestroy* )**

Definition at line 102 of file ClusterAnalysisData.cpp.

References `nodesToDestroy`.

## 6.8.4 Friends And Related Function Documentation

6.8.4.1 **std::ostream& operator<< ( std::ostream & os, const ClusterAnalysisData & obj )**  
[friend]

To stream operator.

### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">ClusterAnalysisData</a> & obj The object to stream

### Returns

std::ostream & The output stream

Definition at line 147 of file ClusterAnalysisData.h.

## 6.8.5 Member Data Documentation

### 6.8.5.1 RangeEnergy cryomesh::manipulators::ClusterAnalysisData::cluster-RangeEnergy [private]

Definition at line 157 of file ClusterAnalysisData.h.

Referenced by getClusterRangeEnergy(), and setClusterRangeEnergy().

### 6.8.5.2 double cryomesh::manipulators::ClusterAnalysisData::connection-CreationWeight [private]

Definition at line 175 of file ClusterAnalysisData.h.

Referenced by getConnectionCreationWeight(), and setConnectionCreationWeight().

### 6.8.5.3 double cryomesh::manipulators::ClusterAnalysisData::connection-DestructionWeight [private]

Definition at line 181 of file ClusterAnalysisData.h.

Referenced by getConnectionDestructionWeight(), and setConnectionDestructionWeight().

### 6.8.5.4 int cryomesh::manipulators::ClusterAnalysisData::connectionsToCreate [private]

Definition at line 185 of file ClusterAnalysisData.h.

Referenced by getConnectionsToCreate(), and setConnectionsToCreate().

### 6.8.5.5 int cryomesh::manipulators::ClusterAnalysisData::connectionsToDestroy [private]

Definition at line 186 of file ClusterAnalysisData.h.

Referenced by getConnectionsToDestroy(), and setConnectionsToDestroy().

### 6.8.5.6 double cryomesh::manipulators::ClusterAnalysisData::nodeCreation-Weight [private]

Definition at line 163 of file ClusterAnalysisData.h.

Referenced by `getNodeCreationWeight()`, and `setNodeCreationWeight()`.

#### 6.8.5.7 `double cryomesh::manipulators::ClusterAnalysisData::nodeDestructionWeight` `[private]`

Definition at line 169 of file `ClusterAnalysisData.h`.

Referenced by `getNodeDestructionWeight()`, and `setNodeDestructionWeight()`.

#### 6.8.5.8 `int cryomesh::manipulators::ClusterAnalysisData::nodesToCreate` `[private]`

Definition at line 183 of file `ClusterAnalysisData.h`.

Referenced by `getNodesToCreate()`, and `setNodesToCreate()`.

#### 6.8.5.9 `int cryomesh::manipulators::ClusterAnalysisData::nodesToDestroy` `[private]`

Definition at line 184 of file `ClusterAnalysisData.h`.

Referenced by `getNodesToDestroy()`, and `setNodesToDestroy()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterAnalysisData.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterAnalysisData.cpp`

## 6.9 `cryomesh::manipulators::ClusterArchitect` Class Reference

```
#include <ClusterArchitect.h>
```

### Public Types

- enum `ConnectionStrategy` { `ENABLE_SELF_CONNECT` = 1, `ENABLE_EVEN_DISTRIBUTION` = 2 }

### Public Member Functions

- `ClusterArchitect` (`structures::Cluster` &clus, const int max\_history\_sz=DEFAULT\_MAX\_HISTORY\_SIZE, const int history\_stepping\_factor=DEFAULT\_HISTORY\_STEPPING\_FACTOR)
- virtual `~ClusterArchitect` ()
- virtual void `runAnalysis` ()

- virtual const std::map< int, std::list< ClusterAnalysisData > > & getHistories () const  
*Get the history of analysis.*
- void createConnection (boost::shared\_ptr< components::Node > nodeStart, boost::shared\_ptr< components::Node > nodeEnd, int connectivity=1)
- boost::shared\_ptr < components::Connection > deleteConnection (boost::shared\_ptr< components::Connection > conn)
- virtual std::set < boost::shared\_ptr < cryomesh::components::Node > > createRandomNodes (int count, int connectivity=0, int strategy=0)  
*Create a number of random nodes.*
- virtual std::map < boost::uuids::uuid, boost::shared\_ptr < components::Connection > > createRandomConnections (int count)  
*Create a number of random connections.*
- virtual std::map < boost::uuids::uuid, boost::shared\_ptr < components::Node > > destroyRandomNodes (int count)  
*Destroy random nodes.*
- virtual std::map < boost::uuids::uuid, boost::shared\_ptr < components::Connection > > destroyRandomConnections (int count)  
*Destroy random connections.*
- virtual std::map < boost::uuids::uuid, boost::shared\_ptr < components::Node > > getRandomNodes (const int count, const bool allow\_primary)  
*Get a collection of random nodes from the cluster.*
- virtual std::map < boost::uuids::uuid, boost::shared\_ptr < components::Connection > > getRandomConnections (const int count, const bool allow\_primary)  
*Get a collection of random connections from the cluster.*
- int getMaxHistorySize () const
- void setMaxHistorySize (int sz)
- ClusterAnalysisData getCurrentClusterAnalysisData () const
- const std::list < ClusterAnalysisData > & getCurrentHistory () const
- int getHistorySteppingFactor () const
- void setCurrentClusterAnalysisData (ClusterAnalysisData currentClusterAnalysisData)
- void setCurrentHistory (std::list< ClusterAnalysisData > currentHistory)
- std::ostream & printAllHistory (std::ostream &os)
- const structures::Cluster & getCluster () const
- const boost::shared\_ptr < IClusterAnalyser > getClusterAnalyser () const
- void setClusterAnalyser (boost::shared\_ptr< IClusterAnalyser > clusterAnalyser)
- void setHistories (std::map< int, std::list< ClusterAnalysisData > > histories)

### Protected Attributes

- structures::Cluster & cluster

### Static Protected Attributes

- static const unsigned int `DEFAULT_HISTORY_STEPPING_FACTOR` = 2
- static const int `DEFAULT_MAX_HISTORY_SIZE` = 3
- static const double `DEFAULT_CONNECTIVITY_FRACTION` = 0.01

### Private Member Functions

- void `addHistoryEntry` (`ClusterAnalysisData` entry)
- void `splitHistoryByValue` (const std::list< `ClusterAnalysisData` > &history, double db, int countback, std::map< `common::Cycle`, `ClusterAnalysisData` > &below, std::map< `common::Cycle`, `ClusterAnalysisData` > &above) const
- std::vector< `ClusterAnalysisData` > `getHistoryEntriesInRange` (const std::list< `ClusterAnalysisData` > &history, double min\_db, double max\_db, int countback=0) const  
*Get the fraction of history entries from the past count entries that are inside of a range, default is to check all of them.*
- void `reduceContainerSize` (std::list< `ClusterAnalysisData` > &cont, const unsigned int sz)

### Private Attributes

- std::list< `ClusterAnalysisData` > `currentHistory`
- std::map< int, std::list< `ClusterAnalysisData` > > `histories`  
*Map of all histories, the int represents the cycle separation, the list is the resultant values/averages.*
- std::map< int, unsigned int > `historiesNewEntries`
- const int `historySteppingFactor`
- boost::shared\_ptr< `IClusterAnalyser` > `clusterAnalyser`
- `ClusterAnalysisData` `currentClusterAnalysisData`
- int `maxHistorySize`

### 6.9.1 Detailed Description

Definition at line 23 of file `ClusterArchitect.h`.

### 6.9.2 Member Enumeration Documentation

#### 6.9.2.1 enum `cryomesh::manipulators::ClusterArchitect::ConnectionStrategy`

Enumerator:

**`ENABLE_SELF_CONNECT`**  
**`ENABLE_EVEN_DISTRIBUTION`**

Definition at line 26 of file `ClusterArchitect.h`.

### 6.9.3 Constructor & Destructor Documentation

**6.9.3.1 cryomesh::manipulators::ClusterArchitect::ClusterArchitect**  
 ( **structures::Cluster** & *clus*, **const int** *max\_history\_sz* =  
**DEFAULT\_MAX\_HISTORY\_SIZE**, **const int** *history\_stepping\_factor* =  
**DEFAULT\_HISTORY\_STEPPING\_FACTOR** )

Definition at line 29 of file ClusterArchitect.cpp.

References historiesNewEntries, and maxHistorySize.

**6.9.3.2 cryomesh::manipulators::ClusterArchitect::~~ClusterArchitect** ( )  
 [virtual]

Definition at line 38 of file ClusterArchitect.cpp.

### 6.9.4 Member Function Documentation

**6.9.4.1 void cryomesh::manipulators::ClusterArchitect::addHistoryEntry** (  
**ClusterAnalysisData** *entry* ) [private]

Definition at line 507 of file ClusterArchitect.cpp.

References clusterAnalyser, currentClusterAnalysisData, currentHistory, getHistorySteppingFactor(), getMaxHistorySize(), histories, historiesNewEntries, printAllHistory(), and reduceContainerSize().

Referenced by runAnalysis().

**6.9.4.2 void cryomesh::manipulators::ClusterArchitect::createConnection** (  
**boost::shared\_ptr**< **components::Node** > *nodeStart*, **boost::shared\_ptr**<  
**components::Node** > *nodeEnd*, **int** *connectivity* = 1 )

Definition at line 647 of file ClusterArchitect.cpp.

References cluster, and cryomesh::structures::Cluster::getMutableConnectionMap().

Referenced by createRandomNodes().

**6.9.4.3 std::map**< **boost::uuids::uuid**, **boost::shared\_ptr**< **components::Connection** > >  
**cryomesh::manipulators::ClusterArchitect::createRandomConnections** (  
**int** *count* ) [virtual]

Create a number of random connections.

Definition at line 282 of file ClusterArchitect.cpp.

References cluster, cryomesh::structures::Cluster::getMutableConnectionMap(), and cryomesh::structures::Cluster::getMutableNodeMap().

Referenced by `runAnalysis()`.

```
6.9.4.4  std::set< boost::shared_ptr< cryomesh::components::Node > >
         cryomesh::manipulators::ClusterArchitect::createRandomNodes ( int
         count, int connectivity = 0, int strategy = 0 ) [virtual]
```

Create a number of random nodes.

Definition at line 57 of file `ClusterArchitect.cpp`.

References `cluster`, `createConnection()`, `DEFAULT_CONNECTIVITY_FRACTION`, `ENABLE_EVEN_DISTRIBUTION`, `ENABLE_SELF_CONNECT`, `cryomesh::structures::Cluster::getMutableNodeMap()`, `cryomesh::structures::Cluster::getNodeMap()`, and `cryomesh::components::Node::getRandom()`.

Referenced by `runAnalysis()`.

```
6.9.4.5  boost::shared_ptr< components::Connection > cryomesh-
         ::manipulators::ClusterArchitect::deleteConnection ( boost::shared_ptr<
         components::Connection > conn )
```

Definition at line 665 of file `ClusterArchitect.cpp`.

References `cluster`, and `cryomesh::structures::Cluster::getMutableConnectionMap()`.

```
6.9.4.6  std::map< boost::uuids::uuid, boost::shared_ptr< components::Connection > >
         cryomesh::manipulators::ClusterArchitect::destroyRandomConnections
         ( int count ) [virtual]
```

Destroy random connections.

Definition at line 361 of file `ClusterArchitect.cpp`.

References `cluster`, `cryomesh::structures::Cluster::getMutableConnectionMap()`, and `getRandomConnections()`.

Referenced by `runAnalysis()`.

```
6.9.4.7  std::map< boost::uuids::uuid, boost::shared_ptr< components::Node > >
         cryomesh::manipulators::ClusterArchitect::destroyRandomNodes ( int
         count ) [virtual]
```

Destroy random nodes.

Definition at line 333 of file `ClusterArchitect.cpp`.

References `cluster`, `cryomesh::structures::Cluster::getMutableConnectionMap()`, `cryomesh::structures::Cluster::getMutableNodeMap()`, and `getRandomNodes()`.

Referenced by `runAnalysis()`.



**6.9.4.8** `const structures::Cluster & cryomesh::manipulators::ClusterArchitect::getCluster ( ) const`

Definition at line 604 of file ClusterArchitect.cpp.

References cluster.

**6.9.4.9** `const boost::shared_ptr< IClusterAnalyser > cryomesh::manipulators::ClusterArchitect::getClusterAnalyser ( ) const`

Definition at line 608 of file ClusterArchitect.cpp.

References clusterAnalyser.

**6.9.4.10** `ClusterAnalysisData cryomesh::manipulators::ClusterArchitect::getCurrentClusterAnalysisData ( ) const`

Definition at line 584 of file ClusterArchitect.cpp.

References currentClusterAnalysisData.

**6.9.4.11** `const std::list< ClusterAnalysisData > & cryomesh::manipulators::ClusterArchitect::getCurrentHistory ( ) const`

Definition at line 588 of file ClusterArchitect.cpp.

References currentHistory.

**6.9.4.12** `const std::map< int, std::list< ClusterAnalysisData > > & cryomesh::manipulators::ClusterArchitect::getHistories ( ) const [virtual]`

Get the history of analysis.

#### Returns

`const common::SimpleCollection<ClusterAnalysisData>` & The container with the history of analysis

Definition at line 491 of file ClusterArchitect.cpp.

References histories.

6.9.4.13 `std::vector< ClusterAnalysisData > cryomesh::manipulators::ClusterArchitect::getHistoryEntriesInRange ( const std::list< ClusterAnalysisData > & history, double min_db, double max_db, int countback = 0 ) const`  
`[private]`

Get the fraction of history entries from the past count entries that are inside of a range, default is to check all of them.

#### Parameters

<i>double</i>	Value for entries to be above
<i>double</i>	Value for entries to be below
<i>int</i>	Number of past entries to go back

#### Returns

`std::vector<ClusterAnalysisData>` Entries within range

Definition at line 620 of file ClusterArchitect.cpp.

References `splitHistoryByValue()`.

6.9.4.14 `int cryomesh::manipulators::ClusterArchitect::getHistorySteppingFactor ( ) const`

Definition at line 592 of file ClusterArchitect.cpp.

References `historySteppingFactor`.

Referenced by `addHistoryEntry()`, and `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`.

6.9.4.15 `int cryomesh::manipulators::ClusterArchitect::getMaxHistorySize ( ) const`

Definition at line 495 of file ClusterArchitect.cpp.

References `maxHistorySize`.

Referenced by `addHistoryEntry()`, and `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`.

6.9.4.16 `std::map< boost::uuids::uuid, boost::shared_ptr< components::Connection > > cryomesh::manipulators::ClusterArchitect::getRandomConnections ( const int count, const bool allow_primary ) [virtual]`

Get a collection of random connections from the cluster.

## Parameters

<i>const</i>	int The number of random connections to return
<i>const</i>	bool Allow the random connections to be attached primaries, default false

## Returns

std::list<boost::shared\_ptr<components::Connection> > List of random connections

Definition at line 440 of file ClusterArchitect.cpp.

References cluster, cryomesh::components::ConnectionMap::getAllPrimaryInputConnections(), cryomesh::components::ConnectionMap::getAllPrimaryOutputConnections(), cryomesh::structures::Cluster::getConnectionMap(), and cryomesh::structures::Cluster::getMutableConnectionMap().

Referenced by destroyRandomConnections().

**6.9.4.17** std::map< boost::uuids::uuid, boost::shared\_ptr< components::Node > >  
**cryomesh::manipulators::ClusterArchitect::getRandomNodes ( const int**  
*count, const bool allow\_primary ) [virtual]*

Get a collection of random nodes from the cluster.

## Parameters

<i>const</i>	int The number of random nodes to return
<i>const</i>	bool Allow the random nodes to be attached primaries, default false

## Returns

std::list<boost::shared\_ptr<components::Node> > List of random nodes

Definition at line 388 of file ClusterArchitect.cpp.

References cluster, cryomesh::components::NodeMap::getAllPrimaryInputNodes(), cryomesh::components::NodeMap::getAllPrimaryOutputNodes(), cryomesh::structures::Cluster::getMutableNodeMap(), and cryomesh::structures::Cluster::getNodeMap().

Referenced by destroyRandomNodes().

**6.9.4.18** std::ostream & cryomesh::manipulators::ClusterArchitect::printAllHistory (  
std::ostream & os )

Definition at line 672 of file ClusterArchitect.cpp.

References currentHistory, and histories.

Referenced by addHistoryEntry().

**6.9.4.19** `void cryomesh::manipulators::ClusterArchitect::reduceContainerSize ( std::list< ClusterAnalysisData > & cont, const unsigned int sz ) [private]`

Definition at line 691 of file ClusterArchitect.cpp.

Referenced by addHistoryEntry().

**6.9.4.20** `void cryomesh::manipulators::ClusterArchitect::runAnalysis ( ) [virtual]`

Definition at line 41 of file ClusterArchitect.cpp.

References addHistoryEntry(), cluster, clusterAnalyser, createRandomConnections(), createRandomNodes(), destroyRandomConnections(), destroyRandomNodes(), cryomesh::manipulators::ClusterAnalysisData::getConnectionsToCreate(), cryomesh::manipulators::ClusterAnalysisData::getConnectionsToDestroy(), cryomesh::manipulators::ClusterAnalysisData::getNodesToCreate(), cryomesh::manipulators::ClusterAnalysisData::getNodesToDestroy(), and histories.

**6.9.4.21** `void cryomesh::manipulators::ClusterArchitect::setClusterAnalyser ( boost::shared_ptr< IClusterAnalyser > clusterAnalyser )`

Definition at line 612 of file ClusterArchitect.cpp.

References clusterAnalyser.

**6.9.4.22** `void cryomesh::manipulators::ClusterArchitect::setCurrentClusterAnalysisData ( ClusterAnalysisData currentClusterAnalysisData )`

Definition at line 596 of file ClusterArchitect.cpp.

References currentClusterAnalysisData.

**6.9.4.23** `void cryomesh::manipulators::ClusterArchitect::setCurrentHistory ( std::list< ClusterAnalysisData > currentHistory )`

Definition at line 600 of file ClusterArchitect.cpp.

References currentHistory.

**6.9.4.24** `void cryomesh::manipulators::ClusterArchitect::setHistories ( std::map< int, std::list< ClusterAnalysisData > > histories )`

Definition at line 616 of file ClusterArchitect.cpp.

References histories.

6.9.4.25 `void cryomesh::manipulators::ClusterArchitect::setMaxHistorySize ( int sz )`

Definition at line 499 of file ClusterArchitect.cpp.

References `maxHistorySize`.

6.9.4.26 `void cryomesh::manipulators::ClusterArchitect::splitHistoryByValue ( const std::list< ClusterAnalysisData > & history, double db, int countback, std::map< common::Cycle, ClusterAnalysisData > & below, std::map< common::Cycle, ClusterAnalysisData > & above ) const [private]`

Definition at line 560 of file ClusterArchitect.cpp.

Referenced by `getHistoryEntriesInRange()`.

## 6.9.5 Member Data Documentation

6.9.5.1 `structures::Cluster& cryomesh::manipulators::ClusterArchitect::cluster [protected]`

Definition at line 98 of file ClusterArchitect.h.

Referenced by `createConnection()`, `createRandomConnections()`, `createRandomNodes()`, `deleteConnection()`, `destroyRandomConnections()`, `destroyRandomNodes()`, `getCluster()`, `getRandomConnections()`, `getRandomNodes()`, and `runAnalysis()`.

6.9.5.2 `boost::shared_ptr< IClusterAnalyser > cryomesh::manipulators::ClusterArchitect::clusterAnalyser [private]`

Definition at line 131 of file ClusterArchitect.h.

Referenced by `addHistoryEntry()`, `getClusterAnalyser()`, `runAnalysis()`, and `setClusterAnalyser()`.

6.9.5.3 `ClusterAnalysisData cryomesh::manipulators::ClusterArchitect::currentClusterAnalysisData [private]`

Definition at line 132 of file ClusterArchitect.h.

Referenced by `addHistoryEntry()`, `getCurrentClusterAnalysisData()`, and `setCurrentClusterAnalysisData()`.

6.9.5.4 `std::list<ClusterAnalysisData > cryomesh::manipulators::ClusterArchitect::currentHistory [private]`

Definition at line 115 of file ClusterArchitect.h.

Referenced by `addHistoryEntry()`, `getCurrentHistory()`, `printAllHistory()`, and `setCurrentHistory()`.

**6.9.5.5** `const double cryomesh::manipulators::ClusterArchitect::  
::DEFAULT_CONNECTIVITY_FRACTION = 0.01` `[static,  
protected]`

Definition at line 107 of file `ClusterArchitect.h`.

Referenced by `createRandomNodes()`.

**6.9.5.6** `const unsigned int cryomesh::manipulators::ClusterArchitect-  
::DEFAULT_HISTORY_STEPPING_FACTOR = 2` `[static,  
protected]`

Definition at line 105 of file `ClusterArchitect.h`.

**6.9.5.7** `const int cryomesh::manipulators::ClusterArchitect-  
::DEFAULT_MAX_HISTORY_SIZE = 3` `[static,  
protected]`

Definition at line 106 of file `ClusterArchitect.h`.

**6.9.5.8** `std::map<int, std::list<ClusterAnalysisData> >  
cryomesh::manipulators::ClusterArchitect::histories` `[private]`

Map of all histories, the int represents the cycle separation, the list is the resultant values/averages.

eg

- mapping of 1 to a list of [ClusterAnalysisData](#) is the standard save of every cycles history (up to cutoff)
- mapping of 10 means that every 10 cycles are averaged and the result added to the mapped list

Note that mapping steps are recursive, we can only have int keys as multiples of each other, eg {1, 2, 4, 8, etc} or {1, 10, 100, ... }, so { 1, a, a<sup>2</sup>, a<sup>3</sup>, ... }

Definition at line 127 of file `ClusterArchitect.h`.

Referenced by `addHistoryEntry()`, `getHistories()`, `printAllHistory()`, `runAnalysis()`, and `setHistories()`.

**6.9.5.9** `std::map<int, unsigned int > cryomesh::manipulators::ClusterArchitect-  
::historiesNewEntries` `[private]`

Definition at line 128 of file `ClusterArchitect.h`.

Referenced by `addHistoryEntry()`, and `ClusterArchitect()`.

#### 6.9.5.10 `const int cryomesh::manipulators::ClusterArchitect::historySteppingFactor` `[private]`

Definition at line 129 of file `ClusterArchitect.h`.

Referenced by `getHistorySteppingFactor()`.

#### 6.9.5.11 `int cryomesh::manipulators::ClusterArchitect::maxHistorySize` `[private]`

Definition at line 134 of file `ClusterArchitect.h`.

Referenced by `ClusterArchitect()`, `getMaxHistorySize()`, and `setMaxHistorySize()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterArchitect.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterArchitect.cpp`

## 6.10 cryomesh::components::Connection Class Reference

`Connection` class to manage the transfer of Impulses between Nodes.

```
#include <Connection.h>
```

### Public Member Functions

- `Connection ()`  
*Constructor for `Connection`.*
- `virtual ~Connection ()`  
*Destructor for `Connection`.*
- `virtual void update ()`  
*Update the `Connection`.*
- `virtual const common::Connector< Connection, Node > & getConnector () const`  
*Get the Connector object for this `Connection` input.*
- `virtual common::Connector< Connection, Node > & getMutableConnector ()`  
*Get the mutable Connector object for this `Connection`.*
- `boost::shared_ptr< Impulse > add (boost::shared_ptr< Impulse > impulse)`  
*Add an `Impulse` to this connection.*
- `boost::shared_ptr< Impulse > remove (boost::shared_ptr< Impulse > impulse)`

Add an [Impulse](#) to this connection.

- `boost::shared_ptr< Impulse > remove (Impulse &impulse)`

Remove an [Impulse](#) from this connection.

- `const ImpulseCollection & getImpulses () const`

Get the impulse collection.

- `ImpulseCollection & getMutableImpulses ()`

Get the mutable impulse collection.

- `const boost::shared_ptr < components::ActivityTimerDistance > getActivityTimer () const`

The get activity timer for this object.

- `boost::shared_ptr < components::ActivityTimerDistance > getMutableActivityTimer ()`

The get mutable activity timer for this object.

- `boost::shared_ptr < manager::DatabaseObject > getDatabaseObject () const`

Return a database object for this object.

- `void updatePosition ()`

Update position.

- `void connectInput (boost::shared_ptr< Node > node)`
- `void connectOutput (boost::shared_ptr< Node > node)`
- `void disconnect ()`
- `void disconnectInput ()`
- `void disconnectOutput ()`
- `bool isPrimaryInputConnection () const`
- `bool isPrimaryOutputConnection () const`
- `virtual void enableDebug (bool b)`

## Protected Attributes

- `boost::shared_ptr < common::Connector < Connection, Node > > connector`
- `ImpulseCollection impulses`
- `boost::shared_ptr < components::ActivityTimerDistance > activityTimer`

## Friends

- `std::ostream & operator<< (std::ostream &os, const Connection &obj)`

To stream operator.

### 6.10.1 Detailed Description

[Connection](#) class to manage the transfer of Impulses between Nodes.

A [Connection](#) represents the 'journey' made by Impulses as they travel between a start [Node](#) and an end [Node](#). They can be spatially based or more abstract representations of this journey. As each clock moment passes Impulses are propagated in some way



'closer' to their end [Node](#). Impulses can be altered by the Mesh on each cycle. When Impulses reach the 'end' of their journey they are passed to the end [Node](#) for accumulation. Connections can also be bi-directional, an [Impulse](#) from a start [Node](#) could be in some way 'reflected' back to that node. Perhaps once in a 'weighted' reflection to be accumulated by the start [Node](#), or even in a constant per-cycle reflection as the original [Impulse](#) 'moves along' the [Connection](#)

Definition at line 37 of file Connection.h.

## 6.10.2 Constructor & Destructor Documentation

### 6.10.2.1 cryomesh::components::Connection::Connection ( )

Constructor for [Connection](#).

Constructor

Definition at line 16 of file Connection.cpp.

### 6.10.2.2 cryomesh::components::Connection::~~Connection ( ) [virtual]

Destructor for [Connection](#).

Destructor

Definition at line 22 of file Connection.cpp.

## 6.10.3 Member Function Documentation

### 6.10.3.1 boost::shared\_ptr< Impulse > cryomesh::components::Connection::add ( boost::shared\_ptr< Impulse > impulse )

Add an [Impulse](#) to this connection.

Parameters

<i>boost- ::shared_ _ptr&lt;- Impulse&gt;</i>	impulse The <a href="#">Impulse</a> to add
---	--

Returns

boost::shared\_ptr<Impulse> The [Impulse](#) added, null if none added

Definition at line 86 of file Connection.cpp.

References [activityTimer](#), and [impulses](#).

Referenced by [cryomesh::components::Node::emitImpulse\(\)](#).

**6.10.3.2 void cryomesh::components::Connection::connectInput ( boost::shared\_ptr< Node > node )**

Definition at line 177 of file Connection.cpp.

References getMutableConnector().

**6.10.3.3 void cryomesh::components::Connection::connectOutput ( boost::shared\_ptr< Node > node )**

Definition at line 180 of file Connection.cpp.

References getMutableConnector().

**6.10.3.4 void cryomesh::components::Connection::disconnect ( )**

Definition at line 183 of file Connection.cpp.

References disconnectInput(), and disconnectOutput().

**6.10.3.5 void cryomesh::components::Connection::disconnectInput ( )**

Definition at line 188 of file Connection.cpp.

References getConnector(), and getMutableConnector().

Referenced by disconnect().

**6.10.3.6 void cryomesh::components::Connection::disconnectOutput ( )**

Definition at line 207 of file Connection.cpp.

References getConnector(), and getMutableConnector().

Referenced by disconnect().

**6.10.3.7 void cryomesh::components::Connection::enableDebug ( bool b )**  
[virtual]

Definition at line 242 of file Connection.cpp.

**6.10.3.8 const boost::shared\_ptr< components::ActivityTimerDistance > cryomesh::components::Connection::getActivityTimer ( ) const**

The get activity timer for this object.

```
const ActiveImpulseCollection & Connection::getActiveImpulses() const { return this->activeImpulses; }
```

**Returns**

boost::shared\_ptr<ActivityTimer>

```
ActiveImpulseCollection & Connection::getMutableActiveImpulses() { return this->activeImpulses; }
```

Definition at line 132 of file Connection.cpp.

References activityTimer.

**6.10.3.9** `const common::Connector< Connection, Node > & cryomesh::components::Connection::getConnector ( ) const`  
[virtual]

Get the Connector object for this [Connection](#) input.

**Returns**

common::Connector<Connection, Node> The connector for this [Connection](#)

Definition at line 78 of file Connection.cpp.

References connector.

Referenced by disconnectInput(), disconnectOutput(), getDatabaseObject(), isPrimaryInputConnection(), isPrimaryOutputConnection(), and cryomesh::components::operator<<().

**6.10.3.10** `boost::shared_ptr< manager::DatabaseObject > cryomesh::components::Connection::getDatabaseObject ( ) const`

Return a database object for this object.

**Returns**

DatabaseObject

Definition at line 140 of file Connection.cpp.

References getConnector(), cryomesh::common::TimeKeeper::getCycle(), getImpulses(), and cryomesh::common::TimeKeeper::getTimeKeeper().

**6.10.3.11** `const ImpulseCollection & cryomesh::components::Connection::getImpulses ( ) const`

Get the impulse collection.

**Returns**

const [ImpulseCollection](#) & The impulse collection

Definition at line 114 of file Connection.cpp.

References impulses.

Referenced by cryomesh::components::Node::emitImpulse(), getDatabaseObject(), and cryomesh::components::operator<<().

**6.10.3.12** `boost::shared_ptr< components::ActivityTimerDistance >  
cryomesh::components::Connection::getMutableActivityTimer ( )`

The get mutable activity timer for this object.

**Returns**

boost::shared\_ptr<ActivityTimer>

Definition at line 136 of file Connection.cpp.

References activityTimer.

**6.10.3.13** `common::Connector< Connection, Node > &  
cryomesh::components::Connection::getMutableConnector ( )  
[virtual]`

Get the mutable Connector object for this [Connection](#).

**Returns**

common::Connector<Connection, Node> The connector for this [Connection](#)

Definition at line 82 of file Connection.cpp.

References connector.

Referenced by connectInput(), connectOutput(), disconnectInput(), and disconnectOutput().

**6.10.3.14** `ImpulseCollection & cryomesh::components::Connection::getMutable-  
Impulses ( )`

Get the mutable impulse collection.

**Returns**

[ImpulseCollection](#) & The mutable impulse collection

Definition at line 118 of file Connection.cpp.

References impulses.

6.10.3.15 `bool cryomesh::components::Connection::isPrimaryInputConnection ( ) const`

Definition at line 227 of file Connection.cpp.

References `getConnector()`.

Referenced by `cryomesh::components::operator<<()`.

6.10.3.16 `bool cryomesh::components::Connection::isPrimaryOutputConnection ( ) const`

Definition at line 235 of file Connection.cpp.

References `getConnector()`.

Referenced by `cryomesh::components::operator<<()`.

6.10.3.17 `boost::shared_ptr< Impulse > cryomesh::components::Connection::remove ( boost::shared_ptr< Impulse > impulse )`

Add an [Impulse](#) to this connection.

#### Parameters

<a href="#">Impulse</a>	& impulse The <a href="#">Impulse</a> to add
-------------------------	--

#### Returns

`boost::shared_ptr<Impulse>` The [Impulse](#) added, null if none added  
`boost::shared_ptr<Impulse>` `add(Impulse & impulse)`; Remove an [Impulse](#) from this connection

#### Parameters

<code>boost::shared_ptr&lt; Impulse &gt;</code>	impulse The impulse to remove
---	-------------------------------

#### Returns

`boost::shared_ptr<Impulse>` The [Impulse](#) removed, null if none removed

Definition at line 106 of file Connection.cpp.

References `impulses`.

### 6.10.3.18 `boost::shared_ptr< Impulse > cryomesh::components::Connection::remove ( Impulse & impulse )`

Remove an [Impulse](#) from this connection.

#### Parameters

<a href="#">Impulse</a>	& impulse The impulse to remove
-------------------------	---------------------------------

#### Returns

`boost::shared_ptr<Impulse>` The [Impulse](#) removed, null if none removed

Definition at line 110 of file Connection.cpp.

References impulses.

### 6.10.3.19 `void cryomesh::components::Connection::update ( ) [virtual]`

Update the [Connection](#).

Update our collection of impulses. If any reach the 'endpoint' of the connection then pass them on to our end Nodes

Definition at line 25 of file Connection.cpp.

References connector, `cryomesh::components::ImpulseCollection::decrementActivityTimers()`, impulses, `cryomesh::components::ImpulseCollection::LessThanOrEqualTo`, and `cryomesh::components::ImpulseCollection::removeByActivityTimerValue()`.

### 6.10.3.20 `void cryomesh::components::Connection::updatePosition ( )`

Update position.

Definition at line 160 of file Connection.cpp.

References `activityTimer`, connector, and `cryomesh::components::ActivityTimerDistance::MIN_DISTANCE`.

## 6.10.4 Friends And Related Function Documentation

### 6.10.4.1 `std::ostream& operator<< ( std::ostream & os, const Connection & obj )` [friend]

To stream operator.

#### Parameters

<code>std::ostream</code>	& os The output stream
<code>const</code>	<a href="#">Connection</a> & obj The object to stream

**Returns**

std::ostream & The output stream

Definition at line 245 of file Connection.cpp.

**6.10.5 Member Data Documentation**

**6.10.5.1** boost::shared\_ptr<components::ActivityTimerDistance>  
cryomesh::components::Connection::activityTimer [protected]

Definition at line 211 of file Connection.h.

Referenced by add(), getActivityTimer(), getMutableActivityTimer(), and updatePosition().

**6.10.5.2** boost::shared\_ptr<common::Connector<Connection, Node> >  
cryomesh::components::Connection::connector [protected]

Definition at line 197 of file Connection.h.

Referenced by getConnector(), getMutableConnector(), update(), and updatePosition().

**6.10.5.3** ImpulseCollection cryomesh::components::Connection::impulses  
[protected]

Definition at line 204 of file Connection.h.

Referenced by add(), getImpulses(), getMutableImpulses(), remove(), and update().

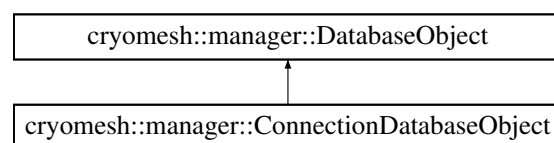
The documentation for this class was generated from the following files:

- /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection.h
- /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection.cpp

**6.11 cryomesh::manager::ConnectionDatabaseObject Class - Reference**

```
#include <ConnectionDatabaseObject.h>
```

Inheritance diagram for cryomesh::manager::ConnectionDatabaseObject:



## Public Member Functions

- [ConnectionDatabaseObject](#) (const std::string &uuid\_str, const std::string &innode\_uuid\_str, const std::string &outnode\_uuid\_str, const [common::Cycle](#) &cycle, const int impulse\_count)  
*Create database object from connection information.*
- [ConnectionDatabaseObject](#) (const std::string &connection\_table\_entry)  
*Create database object from a string database entry for a connection.*
- virtual [~ConnectionDatabaseObject](#) ()  
*Default Destructor.*
- virtual std::string [getInsert](#) (const std::string &table) const  
*Get the string that can be used to insert the sql data.*
- const std::string & [getUUID](#) () const  
*Get uuid variable of this object.*
- const std::string & [getInputNodeUUID](#) () const  
*Get inputNodeUUID variable of this object.*
- const std::string & [getOutputNodeUUID](#) () const  
*Get outputNodeUUID variable of this object.*
- const [common::Cycle](#) & [getCycle](#) () const  
*Get cycle variable of this object.*
- const int & [getImpulseCount](#) () const  
*Get impulseCount variable of this object.*
- std::string [getKey](#) (const std::string &key) const  
*Return the string object associated with a key.*

## Static Public Member Functions

- static std::string [findValue](#) (const std::string &entry, const std::map< std::string, std::string > &map)  
*Find entries value in map or return null.*
- static std::map< std::string, std::string > [getColumnMapFromEntry](#) (const std::string &entry)  
*Parse a string database entry, extract columns and values and return a map.*
- template<class T >  
static std::string [toString](#) (T obj)  
*Convert an templated object that can be piped to a stream to a string.*

## Static Public Attributes

- static const std::string [ID\\_TAG](#) = "id"
- static const std::string [INPUT\\_ID\\_TAG](#) = "inputid"
- static const std::string [OUTPUT\\_ID\\_TAG](#) = "outputid"
- static const std::string [IMPULSE\\_COUNT\\_TAG](#) = "impulses"
- static const std::string [CYCLE\\_TAG](#) = "cycle"



## Protected Attributes

- `std::map< std::string, std::string >` [columns](#)

## Private Attributes

- `std::string` [uuid](#)
- `std::string` [inputNodeUUID](#)
- `std::string` [outputNodeUUID](#)
- [common::Cycle](#) [cycle](#)
- `int` [impulseCount](#)

### 6.11.1 Detailed Description

Definition at line 21 of file `ConnectionDatabaseObject.h`.

### 6.11.2 Constructor & Destructor Documentation

**6.11.2.1** `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject ( const std::string & uuid_str, const std::string & innode_uuid_str, const std::string & outnode_uuid_str, const common::Cycle & cycle, const int impulse_count )`

Create database object from connection information.

#### Parameters

<code>std::string</code>	The uuid of the connection
<code>std::string</code>	The uuid of the input node
<code>std::string</code>	The uuid of the output node
<a href="#">common::Cycle</a>	The cycle of the entry
<code>int</code>	The impulse count of the connection on this entry

Definition at line 20 of file `ConnectionDatabaseObject.cpp`.

References `cryomesh::manager::DatabaseObject::columns`, `cycle`, `CYCLE_TAG`, `ID_TAG`, `IMPULSE_COUNT_TAG`, `INPUT_ID_TAG`, `OUTPUT_ID_TAG`, and `cryomesh::common::Cycle::toInt()`.

**6.11.2.2** `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject ( const std::string & connection_table_entry )`

Create database object from a string database entry for a connection.

## Parameters

<i>std::string</i>	The database entry of the connection
--------------------	--------------------------------------

Definition at line 31 of file ConnectionDatabaseObject.cpp.

References `cycle`, `cryomesh::manager::DatabaseObject::findValue()`, `cryomesh::manager::DatabaseObject::getColumnMapFromEntry()`, `impulseCount`, `inputNodeUUID`, `outputNodeUUID`, and `uuid`.

### 6.11.2.3 cryomesh::manager::ConnectionDatabaseObject::~~ConnectionDatabaseObject ( ) [virtual]

Default Destructor.

Definition at line 66 of file ConnectionDatabaseObject.cpp.

## 6.11.3 Member Function Documentation

### 6.11.3.1 static std::string cryomesh::manager::DatabaseObject::findValue ( const std::string & entry, const std::map< std::string, std::string > & map ) [inline, static, inherited]

Find entries value in map or return null.

## Parameters

<i>std::string</i>	Entry to find
<i>std::map&lt; std::string, std::string &gt;</i>	map to search

## Returns

Value of entry

Definition at line 59 of file DatabaseObject.h.

Referenced by `ConnectionDatabaseObject()`, `cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject()`, and `cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject()`.

### 6.11.3.2 static std::map<std::string, std::string> cryomesh::manager::DatabaseObject::getColumnMapFromEntry ( const std::string & entry ) [inline, static, inherited]

Parse a string database entry, extract columns and values and return a map.

Definition at line 72 of file DatabaseObject.h.

Referenced by ConnectionDatabaseObject(), cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject(), and cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject().

#### 6.11.3.3 const common::Cycle & cryomesh::manager::ConnectionDatabaseObject::getCycle ( ) const

Get cycle variable of this object.

##### Returns

std::string The cycle variable

Definition at line 92 of file ConnectionDatabaseObject.cpp.

References cycle.

#### 6.11.3.4 const int & cryomesh::manager::ConnectionDatabaseObject::getImpulseCount ( ) const

Get impulseCount variable of this object.

##### Returns

std::string The impulseCount variable

Definition at line 96 of file ConnectionDatabaseObject.cpp.

References impulseCount.

#### 6.11.3.5 const std::string & cryomesh::manager::ConnectionDatabaseObject::getInputNodeUUID ( ) const

Get inputNodeUUID variable of this object.

##### Returns

std::string The inputNodeUUID variable

Definition at line 84 of file ConnectionDatabaseObject.cpp.

References inputNodeUUID.

#### 6.11.3.6 std::string cryomesh::manager::ConnectionDatabaseObject::getInsert ( const std::string & table ) const [virtual]

Get the string that can be used to insert the sql data.

**Returns**

the sql command string to insert into this table

Implements [cryomesh::manager::DatabaseObject](#).

Definition at line 69 of file ConnectionDatabaseObject.cpp.

References CYCLE\_TAG, cryomesh::manager::DatabaseObject::getKey(), ID\_TAG, IMPULSE\_COUNT\_TAG, INPUT\_ID\_TAG, and OUTPUT\_ID\_TAG.

**6.11.3.7** `std::string cryomesh::manager::DatabaseObject::getKey ( const std::string & key ) const` *[inline, inherited]*

Return the string object associated with a key.

`::string` The key to search for

**Returns**

`std::string` The object associated with the search key, "" if not found

Definition at line 37 of file DatabaseObject.h.

References cryomesh::manager::DatabaseObject::columns.

Referenced by cryomesh::manager::PatternDatabaseObject::getInsert(), cryomesh::manager::NodeDatabaseObject::getInsert(), and getInsert().

**6.11.3.8** `const std::string & cryomesh::manager::ConnectionDatabaseObject::getOutputNodeUUID ( ) const`

Get outputNodeUUID variable of this object.

**Returns**

`std::string` The outputNodeUUID variable

Definition at line 88 of file ConnectionDatabaseObject.cpp.

References outputNodeUUID.

**6.11.3.9** `const std::string & cryomesh::manager::ConnectionDatabaseObject::getUUID ( ) const`

Get uuid variable of this object.

**Returns**

`std::string` The uuid variable

Definition at line 80 of file ConnectionDatabaseObject.cpp.

References uuid.

6.11.3.10 `template<class T> static std::string cryomesh::manager::DatabaseObject::toString ( T obj )` `[inline, static, inherited]`

Convert an templated object that can be piped to a stream to a string.

#### Parameters

<i>T</i>	The object to get a string for
----------	--------------------------------

Definition at line 108 of file DatabaseObject.h.

### 6.11.4 Member Data Documentation

6.11.4.1 `std::map<std::string, std::string> cryomesh::manager::DatabaseObject::columns` `[protected, inherited]`

Definition at line 119 of file DatabaseObject.h.

Referenced by ConnectionDatabaseObject(), cryomesh::manager::DatabaseObject::getKey(), cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject(), and cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject().

6.11.4.2 `common::Cycle cryomesh::manager::ConnectionDatabaseObject::cycle` `[private]`

Definition at line 162 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getCycle().

6.11.4.3 `const std::string cryomesh::manager::ConnectionDatabaseObject::CYCLE_TAG = "cycle"` `[static]`

Definition at line 133 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getInsert().

6.11.4.4 `const std::string cryomesh::manager::ConnectionDatabaseObject::ID_TAG = "id"` `[static]`

Definition at line 106 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getInsert().

**6.11.4.5** `const std::string cryomesh::manager::ConnectionDatabaseObject::IMPULSE_COUNT_TAG = "impulses"`  
[static]

Definition at line 127 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getInsert().

**6.11.4.6** `int cryomesh::manager::ConnectionDatabaseObject::impulseCount`  
[private]

Definition at line 169 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getImpulseCount().

**6.11.4.7** `const std::string cryomesh::manager::ConnectionDatabaseObject::INPUT_ID_TAG = "inputid"` [static]

Definition at line 113 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getInsert().

**6.11.4.8** `std::string cryomesh::manager::ConnectionDatabaseObject::inputNodeUUID` [private]

Definition at line 148 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getInputNodeUUID().

**6.11.4.9** `const std::string cryomesh::manager::ConnectionDatabaseObject::OUTPUT_ID_TAG = "outputid"` [static]

Definition at line 120 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getInsert().

**6.11.4.10** `std::string cryomesh::manager::ConnectionDatabaseObject::outputNodeUUID` [private]

Definition at line 155 of file ConnectionDatabaseObject.h.

Referenced by ConnectionDatabaseObject(), and getOutputNodeUUID().

**6.11.4.11** `std::string cryomesh::manager::ConnectionDatabaseObject::uuid`  
[private]

Definition at line 141 of file ConnectionDatabaseObject.h.

Referenced by `ConnectionDatabaseObject()`, and `getUUID()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/ConnectionDatabaseObject.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/ConnectionDatabaseObject.cpp`

## 6.12 cryomesh::components::ConnectionMap Class Reference

Helper class for `ConnectionMap` to `KeyMappedCollection` mapping.

```
#include <ConnectionMap.h>
```

### Public Member Functions

- `ConnectionMap ()`  
*Default constructor.*
- `virtual ~ConnectionMap ()`  
*Default destructor.*
- `virtual void update ()`  
*Update all entries in the map.*
- `boost::shared_ptr < state::ActivityPattern > getActivityPattern () const`  
*Get activity pattern on current cycle.*
- `boost::shared_ptr < state::ActivityPattern > getActivityPattern (const common::Cycle &cycle) const`  
*Get activity pattern on cycle.*
- `const std::map < boost::uuids::uuid, boost::shared_ptr< Connection > > getAllPrimaryInputConnections () const`
- `const std::map < boost::uuids::uuid, boost::shared_ptr< Connection > > getAllPrimaryOutputConnections () const`

### Friends

- `std::ostream & operator<< (std::ostream &os, const ConnectionMap &obj)`  
*To stream operator.*

#### 6.12.1 Detailed Description

Helper class for `ConnectionMap` to `KeyMappedCollection` mapping.

Definition at line 26 of file `ConnectionMap.h`.

## 6.12.2 Constructor & Destructor Documentation

**6.12.2.1** `cryomesh::components::ConnectionMap::ConnectionMap ( )`  
`[inline]`

Default constructor.

Definition at line 31 of file ConnectionMap.h.

**6.12.2.2** `virtual cryomesh::components::ConnectionMap::~~ConnectionMap ( )`  
`[inline, virtual]`

Default destructor.

Definition at line 37 of file ConnectionMap.h.

## 6.12.3 Member Function Documentation

**6.12.3.1** `boost::shared_ptr<state::ActivityPattern> cryomesh-  
::components::ConnectionMap::getActivityPattern ( ) const`  
`[inline]`

Get activity pattern on current cycle.

@ param const Cycle & cycle Cycle to get activity on

Definition at line 65 of file ConnectionMap.h.

References cryomesh::common::TimeKeeper::getTimeKeeper().

**6.12.3.2** `boost::shared_ptr<state::ActivityPattern> cryomesh::components::-  
ConnectionMap::getActivityPattern ( const common::Cycle & cycle ) const`  
`[inline]`

Get activity pattern on cycle.

@ param const Cycle & cycle Cycle to get activity on

Definition at line 75 of file ConnectionMap.h.

**6.12.3.3** `const std::map<boost::uuids::uuid, boost::shared_ptr<Connection> >  
cryomesh::components::ConnectionMap::getAllPrimaryInput-  
Connections ( ) const` `[inline]`

Definition at line 97 of file ConnectionMap.h.

Referenced by cryomesh::manipulators::ClusterArchitect::getRandomConnections().



6.12.3.4 `const std::map<boost::uuids::uuid, boost::shared_ptr<Connection> > cryomesh::components::ConnectionMap::getAllPrimaryOutputConnections ( ) const` `[inline]`

Definition at line 119 of file ConnectionMap.h.

Referenced by `cryomesh::manipulators::ClusterArchitect::getRandomConnections()`.

6.12.3.5 `virtual void cryomesh::components::ConnectionMap::update ( )` `[inline, virtual]`

Update all entries in the map.

Definition at line 43 of file ConnectionMap.h.

Referenced by `cryomesh::structures::Cluster::update()`, and `cryomesh::structures::Fibre::update()`.

## 6.12.4 Friends And Related Function Documentation

6.12.4.1 `std::ostream& operator<< ( std::ostream & os, const ConnectionMap & obj )` `[friend]`

To stream operator.

### Parameters

<code>std::ostream</code>	& os The output stream
<code>const</code>	<a href="#">ConnectionMap</a> & obj The object to stream

### Returns

`std::ostream &` The output stream

Definition at line 152 of file ConnectionMap.h.

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

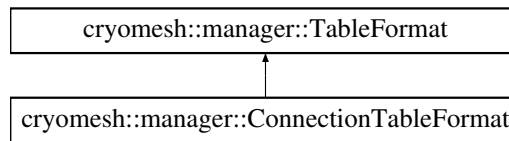
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ConnectionMap.h`

## 6.13 cryomesh::manager::ConnectionTableFormat Struct Reference

Struct representing a connections table structure.

```
#include <TableFormats.h>
```

Inheritance diagram for `cryomesh::manager::ConnectionTableFormat`:



## Public Member Functions

- [ConnectionTableFormat](#) ()  
*Default constructor will construct all the names and columns associated with a connections table.*
- std::string [getName](#) () const  
*Return the name of the table.*
- std::string [getKey](#) (const std::string &key)  
*Return the string object associated with a key.*
- std::string [getCreateTable](#) () const  
*Get the string that can be used to create the sql table.*

## Protected Attributes

- std::string [name](#)
- std::map< std::string, std::string > [columns](#)

### 6.13.1 Detailed Description

Struct representing a connections table structure.

Definition at line 118 of file TableFormats.h.

### 6.13.2 Constructor & Destructor Documentation

#### 6.13.2.1 cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat ( ) [inline]

Default constructor will construct all the names and columns associated with a connections table.

Definition at line 123 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`, and `cryomesh::manager::TableFormat::name`.

### 6.13.3 Member Function Documentation

**6.13.3.1** `std::string cryomesh::manager::TableFormat::getCreateTable ( ) const`  
[inline, inherited]

Get the string that can be used to create the sql table.

#### Returns

the sql command string to create this table

Definition at line 60 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`, and `cryomesh::manager::TableFormat::getName()`.

**6.13.3.2** `std::string cryomesh::manager::TableFormat::getKey ( const std::string & key )` [inline, inherited]

Return the string object associated with a key.

`::string` The key to search for

#### Returns

`std::string` The object associated with the search key, "" if not found

Definition at line 45 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`.

**6.13.3.3** `std::string cryomesh::manager::TableFormat::getName ( ) const`  
[inline, inherited]

Return the name of the table.

#### Returns

`std::string` The name of the table

Definition at line 32 of file TableFormats.h.

References `cryomesh::manager::TableFormat::name`.

Referenced by `cryomesh::manager::TableFormat::getCreateTable()`, `cryomesh::manager::DatabaseManager::insertConnection()`, `cryomesh::manager::DatabaseManager::insertNode()`, and `cryomesh::manager::DatabaseManager::insertOutputPattern()`.

### 6.13.4 Member Data Documentation

6.13.4.1 `std::map<std::string, std::string> cryomesh::manager::TableFormat::columns` [protected, inherited]

Definition at line 93 of file TableFormats.h.

Referenced by ConnectionTableFormat(), cryomesh::manager::TableFormat::getCreateTable(), cryomesh::manager::TableFormat::getKey(), cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat(), cryomesh::manager::NodeTableFormat::NodeTableFormat(), and cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat().

6.13.4.2 `std::string cryomesh::manager::TableFormat::name` [protected, inherited]

Definition at line 86 of file TableFormats.h.

Referenced by ConnectionTableFormat(), cryomesh::manager::TableFormat::getName(), cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat(), cryomesh::manager::NodeTableFormat::NodeTableFormat(), and cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat().

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/TableFormats.h](#)

## 6.14 cryomesh::common::Connector< U, T > Class Template - Reference

[Connector](#) is a template to add connectable functionality between two classes.

```
#include <Connector.h>
```

### Public Member Functions

- [Connector](#) (const unsigned int max\_inputs=0, const unsigned int max\_outputs=0)
- virtual [~Connector](#) ()
- bool [connectInput](#) (const boost::shared\_ptr< T > obj)  
*Connect a unit to this one as an input.*
- bool [connectInputs](#) (const std::vector< boost::shared\_ptr< T > > &list)  
*Connect a list of units to this one as inputs.*
- bool [connectInputs](#) (const std::initializer\_list< boost::shared\_ptr< T > > &list)  
*Connect an initialiser list of units to this one as inputs.*
- bool [connectOutput](#) (const boost::shared\_ptr< T > &obj)  
*Connect a unit to this one as an output.*

- bool [connectOutputs](#) (const std::vector< boost::shared\_ptr< T > > &list)  
*Connect a list of units to this one as outputs.*
- bool [connectOutputs](#) (const std::initializer\_list< boost::shared\_ptr< T > > &list)  
*Connect an initialiser list of units to this one as outputs.*
- bool [disconnectInput](#) (const boost::shared\_ptr< T > &obj)  
*Disconnect an input to this unit.*
- bool [disconnectInput](#) (const boost::uuids::uuid &id)  
*Disconnect an input to this unit.*
- bool [disconnectInputs](#) (const std::vector< boost::shared\_ptr< T > > &list)  
*Disconnect a list of input units from this one.*
- bool [disconnectInputs](#) (const std::initializer\_list< boost::shared\_ptr< T > > &list)  
*Disconnect an initialiser list of input units from this one.*
- bool [disconnectInputs](#) (const std::vector< boost::uuids::uuid > &list)  
*Disconnect a list of input units from this one.*
- bool [disconnectInputs](#) (const std::initializer\_list< boost::uuids::uuid > &list)  
*Disconnect an initialiser list of input units from this one.*
- bool [disconnectAllInputs](#) ()  
*Disconnect all input units from this one.*
- bool [disconnectOutput](#) (const boost::shared\_ptr< T > &obj)  
*Disconnect an output to this unit.*
- bool [disconnectOutput](#) (const boost::uuids::uuid &id)  
*Disconnect an output to this unit.*
- bool [disconnectOutputs](#) (const std::vector< boost::shared\_ptr< T > > &list)  
*Disconnect a list of Output units from this one.*
- bool [disconnectOutputs](#) (const std::initializer\_list< boost::shared\_ptr< T > > &list)  
*Disconnect an initialiser list of Output units from this one.*
- bool [disconnectOutputs](#) (const std::vector< boost::uuids::uuid > &list)  
*Disconnect a uuid list of Output units from this one.*
- bool [disconnectOutputs](#) (const std::initializer\_list< boost::uuids::uuid > &list)  
*Disconnect an uuid initialiser list of Output units from this one.*
- bool [disconnectAllOutputs](#) ()  
*Disconnect all Output units from this one.*
- const std::map < boost::uuids::uuid, boost::shared\_ptr< T > > & [getInputs](#) () const  
*Get all inputs.*
- const std::list < boost::uuids::uuid > [getInputsUUID](#) () const
- const std::list < boost::uuids::uuid > [getOutputsUUID](#) () const
- std::map< boost::uuids::uuid, boost::shared\_ptr< T > > & [getMutableInputs](#) ()  
*Get all inputs as mutable object.*

- `const std::map< boost::uuids::uuid, boost::shared_ptr< T > > & getOutputs ()`  
const  
*Get all outputs.*
- `std::map< boost::uuids::uuid, boost::shared_ptr< T > > & getMutableOutputs ()`  
()  
*Get all outputs as mutable object.*

## Protected Member Functions

- `boost::shared_ptr< T > connect` (const boost::shared\_ptr< T > obj, std::map< boost::uuids::uuid, boost::shared\_ptr< T > > &objs, unsigned int max\_connections)  
*Connect up an object using the supplied map.*
- `boost::shared_ptr< T > disconnect` (const boost::shared\_ptr< T > obj, std::map< boost::uuids::uuid, boost::shared\_ptr< T > > &objs)  
*Disconnect an object using the supplied map.*
- `boost::shared_ptr< T > disconnect` (const boost::uuids::uuid &id, std::map< boost::uuids::uuid, boost::shared\_ptr< T > > &objs)  
*Disconnect an object using the supplied map.*

## Protected Attributes

- `std::map< boost::uuids::uuid, boost::shared_ptr< T > > mininputs`
- `std::map< boost::uuids::uuid, boost::shared_ptr< T > > moutputs`
- unsigned int `maxInputs`
- unsigned int `maxOutputs`

## Friends

- `std::ostream & operator<<` (std::ostream &os, const `Connector< U, T >` &obj)  
*To stream operator.*

### 6.14.1 Detailed Description

`template<class U, class T>class cryomesh::common::Connector< U, T >`

`Connector` is a template to add connectable functionality between two classes.

Represents a template to add connectable functionality between class U as the central connectable and class T as those objects it can be connected to

Definition at line 33 of file `Connector.h`.

## 6.14.2 Constructor & Destructor Documentation

6.14.2.1 `template<class U, class T> cryomesh::common::Connector< U, T >::Connector ( const unsigned int max_inputs = 0, const unsigned int max_outputs = 0 ) [inline]`

Definition at line 35 of file Connector.h.

6.14.2.2 `template<class U, class T> virtual cryomesh::common::Connector< U, T >::~~Connector ( ) [inline, virtual]`

Definition at line 39 of file Connector.h.

## 6.14.3 Member Function Documentation

6.14.3.1 `template<class U, class T> boost::shared_ptr<T> cryomesh::common::Connector< U, T >::connect ( const boost::shared_ptr< T > obj, std::map< boost::uuids::uuid, boost::shared_ptr< T > > & objs, unsigned int max_connections ) [inline, protected]`

Connect up an object using the supplied map.

### Parameters

<code>boost::shared_ptr&lt;T&gt;</code>	<code>obj</code> <a href="#">Pointer</a> to the object that is to be connected
<code>std::map&lt;boost::uuids::uuid, boost::shared_ptr&lt;T&gt;&gt;</code>	<code>&gt; objs</code> The map that will be used to connect the object
<code>unsigned</code>	int Maximum connections to allow

### Returns

[Pointer](#) to the connected object

Definition at line 655 of file Connector.h.

Referenced by `cryomesh::common::Connector< Fibre, Cluster >::connectInput()`, and `cryomesh::common::Connector< Fibre, Cluster >::connectOutput()`.

6.14.3.2 `template<class U, class T> bool cryomesh::common::Connector< U, T >::connectInput ( const boost::shared_ptr< T > obj ) [inline]`

Connect a unit to this one as an input.

## Parameters

<i>boost- ::shared_ _ptr&lt;T&gt;</i>	obj <a href="#">Pointer</a> to the object to be connected as input
---	--

## Returns

true if connection succeeds, false otherwise

Definition at line 53 of file Connector.h.

Referenced by `cryomesh::structures::Fibre::connectAllConnections()`, and `cryomesh::common::Connector< Fibre, Cluster >::connectInputs()`.

```
6.14.3.3  template<class U, class T> bool cryomesh::common::Connector< U, T
>::connectInputs ( const std::vector< boost::shared_ptr< T > > & list )
[inline]
```

Connect a list of units to this one as inputs.

## Parameters

<i>std- ::vector&lt;boost- ::shared_ _ptr&lt;T&gt;</i>	> list List of pointers to objects to be connected as inputs
--	--

## Returns

true if all connections succeed, false otherwise

Definition at line 73 of file Connector.h.

```
6.14.3.4  template<class U, class T> bool cryomesh::common::Connector< U, T
>::connectInputs ( const std::initializer_list< boost::shared_ptr< T > > & list )
[inline]
```

Connect an initialiser list of units to this one as inputs.

## Parameters

<i>std- ::initializer_ _list&lt;boost- ::shared_ _ptr&lt;</i>	T > > list Initialiser list of pointers to objects to be connected as input
---	---



#### Returns

true if all connections succeed, false otherwise

Definition at line 97 of file Connector.h.

**6.14.3.5** `template<class U, class T> bool cryomesh::common::Connector< U, T >::connectOutput ( const boost::shared_ptr< T > & obj ) [inline]`

Connect a unit to this one as an output.

#### Parameters

<code>boost- ::shared_ _ - ptr&lt; T &gt;</code>	obj <a href="#">Pointer</a> to the object to be connected as output
--	---

#### Returns

true if connection succeeds, false otherwise

Definition at line 122 of file Connector.h.

Referenced by cryomesh::structures::Fibre::connectAllConnections(), and cryomesh::common::Connector< Fibre, Cluster >::connectOutputs().

**6.14.3.6** `template<class U, class T> bool cryomesh::common::Connector< U, T >::connectOutputs ( const std::vector< boost::shared_ptr< T > > & list ) [inline]`

Connect a list of units to this one as outputs.

#### Parameters

<code>std- ::vector&lt;boost- ::shared_ _ - ptr&lt; T &gt;</code>	> list List of pointers to objects to be connected as outputs
---	---

**Returns**

true if all connections succeed, false otherwise

Definition at line 142 of file Connector.h.

```
6.14.3.7 template<class U, class T> bool cryomesh::common::Connector< U, T
>::connectOutputs ( const std::initializer_list< boost::shared_ptr< T > > & list )
[inline]
```

Connect an initialiser list of units to this one as outputs.

**Parameters**

<code>std- ::initializer_ _list&lt;boost- ::shared_ _ptr&lt;</code>	<code>T &gt; &gt;</code> list Initialiser list of pointers to objects to be connected as outputs
---	--

**Returns**

true if all connections succeed, false otherwise

Definition at line 167 of file Connector.h.

```
6.14.3.8 template<class U, class T> boost::shared_ptr<T> cryomesh::common-
::Connector< U, T >::disconnect ( const boost::shared_ptr< T > & obj,
std::map< boost::uuids::uuid, boost::shared_ptr< T > > & objs ) [inline,
protected]
```

Disconnect an object using the supplied map.

**Parameters**

<code>boost- ::shared_ _ptr&lt;T&gt;</code>	obj <a href="#">Pointer</a> to the object that is to be disconnected
<code>std- ::map&lt;boost- ::uuids- ::uuid,boost- ::shared_ _ptr&lt;T&gt;</code>	> objs The map that will be used to disconnect the object

**Returns**

[Pointer](#) to the disconnected object, pointer is 0 is object was not found

Definition at line 685 of file Connector.h.

Referenced by cryomesh::common::Connector< Fibre, Cluster >::disconnectInput(), and cryomesh::common::Connector< Fibre, Cluster >::disconnectOutput().

**6.14.3.9** `template<class U, class T> boost::shared_ptr<T> cryomesh::common::Connector< U, T >::disconnect ( const boost::uuids::uuid & id, std::map< boost::uuids::uuid, boost::shared_ptr< T > > & objs ) [inline, protected]`

Disconnect an object using the supplied map.

**Parameters**

<code>boost::uuids::uuid</code>	id The uuid of the object that is to be disconnected
<code>std::map&lt; boost::uuids::uuid, boost::shared_ptr&lt; T &gt; &gt;</code>	& objs The map that will be used to disconnect the object

**Returns**

[Pointer](#) to the disconnected object, pointer is null is object was not found

Definition at line 724 of file Connector.h.

**6.14.3.10** `template<class U, class T> bool cryomesh::common::Connector< U, T >::disconnectAllInputs ( ) [inline]`

Disconnect all input units from this one.

**Returns**

true if all disconnections succeed, false otherwise

Definition at line 332 of file Connector.h.

Referenced by cryomesh::structures::Fibre::disconnectAllConnections().

**6.14.3.11** `template<class U, class T> bool cryomesh::common::Connector< U, T >::disconnectAllOutputs ( ) [inline]`

Disconnect all Output units from this one.

**Returns**

true if all disconnections succeed, false otherwise

Definition at line 500 of file Connector.h.

Referenced by cryomesh::structures::Fibre::disconnectAllConnections().

**6.14.3.12** `template<class U, class T> bool cryomesh::common::Connector< U, T >::disconnectInput ( const boost::shared_ptr< T > & obj ) [inline]`

Disconnect an input to this unit.

**Parameters**

<code>boost- ::shared_ _ptr&lt; T &gt;</code>	obj <a href="#">Pointer</a> to the object to be disconnected from input
---	---

**Returns**

true if disconnection succeeds, false otherwise

Definition at line 193 of file Connector.h.

Referenced by cryomesh::common::Connector< Fibre, Cluster >::disconnectAllInputs(), and cryomesh::common::Connector< Fibre, Cluster >::disconnectInputs().

**6.14.3.13** `template<class U, class T> bool cryomesh::common::Connector< U, T >::disconnectInput ( const boost::uuids::uuid & id ) [inline]`

Disconnect an input to this unit.

**Parameters**

<code>boost::uuids- ::uuid</code>	id The unique identifier of the object to be disconnected
---------------------------------------	---

**Returns**

true if disconnection succeeds, false otherwise

Definition at line 212 of file Connector.h.

**6.14.3.14** `template<class U, class T> bool cryomesh::common::Connector< U, T >::disconnectInputs ( const std::vector< boost::shared_ptr< T > > & list ) [inline]`

Disconnect a list of input units from this one.

## Parameters

<i>std- ::vector&lt;boost- ::shared_ _ptr&lt;T&gt;</i>	> list List of pointers to objects to be disconnected
--	---

## Returns

true if all disconnections succeed, false otherwise

Definition at line 234 of file Connector.h.

**6.14.3.15** `template<class U, class T> bool cryomesh::common::Connector< U, T  
>::disconnectInputs ( const std::initializer_list< boost::shared_ptr< T > > & list  
) [inline]`

Disconnect an initialiser list of input units from this one.

## Parameters

<i>std- ::initializer_ _list&lt;boost- ::shared_ _ptr&lt;</i>	<i>T &gt; &gt;</i> list Initialiser list of pointers to objects to be disconnected as inputs
---	--

## Returns

true if all disconnections succeed, false otherwise

Definition at line 259 of file Connector.h.

**6.14.3.16** `template<class U, class T> bool cryomesh::common::Connector< U, T  
>::disconnectInputs ( const std::vector< boost::uuids::uuid > & list )  
[inline]`

Disconnect a list of input units from this one.

## Parameters

<i>boost::uuids- ::uuid</i>	list List of uuids to objects to be disconnected
---------------------------------	--

## Returns

true if all disconnections succeed, false otherwise

Definition at line 284 of file Connector.h.

6.14.3.17 `template<class U, class T> bool cryomesh::common::Connector< U, T >::disconnectInputs ( const std::initializer_list< boost::uuids::uuid > & list ) [inline]`

Disconnect an initialiser list of input units from this one.

#### Parameters

<code>boost::uuids::uuid</code>	list Initialiser list of uuids to objects to be disconnected as inputs
---------------------------------	--

#### Returns

true if all disconnections succeed, false otherwise

Definition at line 309 of file Connector.h.

6.14.3.18 `template<class U, class T> bool cryomesh::common::Connector< U, T >::disconnectOutput ( const boost::shared_ptr< T > & obj ) [inline]`

Disconnect an output to this unit.

#### Parameters

<code>boost::shared_ptr&lt; T &gt;</code>	obj <a href="#">Pointer</a> to the object to be disconnected from output
---	--

#### Returns

true if disconnection succeeds, false otherwise

Definition at line 358 of file Connector.h.

Referenced by `cryomesh::common::Connector< Fibre, Cluster >::disconnectAll-Outputs()`, and `cryomesh::common::Connector< Fibre, Cluster >::disconnectOutputs()`.

6.14.3.19 `template<class U, class T> bool cryomesh::common::Connector< U, T >::disconnectOutput ( const boost::uuids::uuid & id ) [inline]`

Disconnect an output to this unit.

#### Parameters

<code>boost::uuids::uuid</code>	id The unique identifier of the object to be disconnected
---------------------------------	---

**Returns**

true if disconnection succeeds, false otherwise

Definition at line 379 of file Connector.h.

**6.14.3.20** `template<class U, class T> bool cryomesh::common::Connector< U, T  
>::disconnectOutputs ( const std::vector< boost::shared_ptr< T > > & list )  
[inline]`

Disconnect a list of Output units from this one.

**Parameters**

<code>std- ::vector&lt;boost- ::shared_ ptr&lt;T&gt;</code>	<code>&gt;</code> list List of pointers to objects to be disconnected
---	---

**Returns**

true if all disconnections succeed, false otherwise

Definition at line 401 of file Connector.h.

**6.14.3.21** `template<class U, class T> bool cryomesh::common::Connector< U, T  
>::disconnectOutputs ( const std::initializer_list< boost::shared_ptr< T > > &  
list ) [inline]`

Disconnect an initialiser list of Output units from this one.

**Parameters**

<code>std- ::initializer_ list&lt;boost- ::shared_ ptr&lt;</code>	<code>T &gt; &gt;</code> list Initialiser list of pointers to objects to be disconnected as Outputs
---	---

**Returns**

true if all disconnections succeed, false otherwise

Definition at line 426 of file Connector.h.

```
6.14.3.22  template<class U, class T> bool cryomesh::common::Connector< U, T
           >::disconnectOutputs ( const std::vector< boost::uuids::uuid > & list )
           [inline]
```

Disconnect a uuid list of Output units from this one.

**Parameters**

<i>boost::uuids- ::uuid</i>	list List of uuids to objects to be disconnected
---------------------------------	--

**Returns**

true if all disconnections succeed, false otherwise

Definition at line 452 of file Connector.h.

```
6.14.3.23  template<class U, class T> bool cryomesh::common::Connector< U, T
           >::disconnectOutputs ( const std::initializer_list< boost::uuids::uuid > & list )
           [inline]
```

Disconnect an uuid initialiser list of Output units from this one.

**Parameters**

<i>boost::uuids- ::uuid</i>	list Initialiser list of uuids to objects to be disconnected as Outputs
---------------------------------	---

**Returns**

true if all disconnections succeed, false otherwise

Definition at line 477 of file Connector.h.

```
6.14.3.24  template<class U, class T> const std::map<boost::uuids::uuid,
           boost::shared_ptr<T> >& cryomesh::common::Connector< U, T
           >::getInputs ( ) const [inline]
```

Get all inputs.



**Returns**

std::map<boost::uuids::uuid, boost::shared\_ptr<T> The map of inputs

Definition at line 524 of file Connector.h.

Referenced by cryomesh::structures::Fibre::countConnections(), and cryomesh::structures::Fibre::isConnected().

```
6.14.3.25 template<class U, class T> const std::list<boost::uuids::uuid>
cryomesh::common::Connector< U, T >::getInputsUUID ( ) const
[inline]
```

Definition at line 528 of file Connector.h.

```
6.14.3.26 template<class U, class T> std::map<boost::uuids::uuid, boost::shared_ptr<T>
>& cryomesh::common::Connector< U, T >::getMutableInputs ( )
[inline]
```

Get all inputs as mutable object.

**Returns**

std::map<boost::uuids::uuid, boost::shared\_ptr<T> The map of inputs

Definition at line 566 of file Connector.h.

```
6.14.3.27 template<class U, class T> std::map<boost::uuids::uuid, boost::shared_ptr<T>
>& cryomesh::common::Connector< U, T >::getMutableOutputs ( )
[inline]
```

Get all outputs as mutable object.

**Returns**

std::map<boost::uuids::uuid, boost::shared\_ptr<T> The map of outputs

Definition at line 588 of file Connector.h.

```
6.14.3.28 template<class U, class T> const std::map<boost::uuids::uuid,
boost::shared_ptr<T> >& cryomesh::common::Connector< U, T
>::getOutputs ( ) const [inline]
```

Get all outputs.

**Returns**

`std::map<boost::uuids::uuid, boost::shared_ptr<T>` The map of outputs

Definition at line 577 of file Connector.h.

Referenced by `cryomesh::structures::Fibre::countConnections()`, and `cryomesh::structures::Fibre::isConnected()`.

6.14.3.29 `template<class U, class T> const std::list<boost::uuids::uuid>  
cryomesh::common::Connector< U, T >::getOutputsUUID ( ) const  
[inline]`

Definition at line 543 of file Connector.h.

**6.14.4 Friends And Related Function Documentation**

6.14.4.1 `template<class U, class T> std::ostream& operator<< ( std::ostream & os, const  
Connector< U, T > & obj ) [friend]`

To stream operator.

**Parameters**

<code>std::ostream</code>	& os The output stream
<code>const</code>	Connector<U,T> & obj The object to stream

**Returns**

`std::ostream &` The output stream

Definition at line 603 of file Connector.h.

**6.14.5 Member Data Documentation**

6.14.5.1 `template<class U, class T> unsigned int cryomesh::common::Connector< U, T  
>::maxInputs [protected]`

Definition at line 770 of file Connector.h.

Referenced by `cryomesh::common::Connector< Fibre, Cluster >::connectInput()`.

6.14.5.2 `template<class U, class T> unsigned int cryomesh::common::Connector< U, T  
>::maxOutputs [protected]`

Definition at line 777 of file Connector.h.

Referenced by `cryomesh::common::Connector< Fibre, Cluster >::connectOutput()`.

6.14.5.3 `template<class U, class T> std::map<boost::uuids::uuid, boost::shared_ptr<T> >  
cryomesh::common::Connector< U, T >::minputs [protected]`

Definition at line 756 of file Connector.h.

Referenced by `cryomesh::common::Connector< Fibre, Cluster >::connectInput()`, `cryomesh::common::Connector< Fibre, Cluster >::disconnectAllInputs()`, `cryomesh::common::Connector< Fibre, Cluster >::disconnectInput()`, `cryomesh::common::Connector< Fibre, Cluster >::getInputs()`, `cryomesh::common::Connector< Fibre, Cluster >::getInputsUUID()`, and `cryomesh::common::Connector< Fibre, Cluster >::getMutableInputs()`.

6.14.5.4 `template<class U, class T> std::map<boost::uuids::uuid, boost::shared_ptr<T> >  
cryomesh::common::Connector< U, T >::moutputs [protected]`

Definition at line 763 of file Connector.h.

Referenced by `cryomesh::common::Connector< Fibre, Cluster >::connectOutput()`, `cryomesh::common::Connector< Fibre, Cluster >::disconnectAllOutputs()`, `cryomesh::common::Connector< Fibre, Cluster >::disconnectOutput()`, `cryomesh::common::Connector< Fibre, Cluster >::getMutableOutputs()`, `cryomesh::common::Connector< Fibre, Cluster >::getOutputs()`, and `cryomesh::common::Connector< Fibre, Cluster >::getOutputsUUID()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Connector.h`

## 6.15 cryomesh::manager::Creator Class Reference

Class to take in a config file of ConfigTranslator form and parse the commands to create a full cryomesh object.

```
#include <Creator.h>
```

### Public Member Functions

- `Creator` (const std::string &config\_filename, const std::string &database\_filename=DEFAULT\_DATABASE\_FILENAME)  
*Constructor to create a bundle from a config file name with option to specify the database file name that will be used.*
- `Creator` (std::istream &config\_stream, const std::string &database\_filename=DEFAULT\_DATABASE\_FILENAME)
- virtual `~Creator` ()  
*Default destructor.*
- const boost::shared\_ptr < `structures::Bundle` > `getBundle` () const  
*Get the create bundle.*

- `boost::shared_ptr < structures::Bundle > getMutableBundle ()`  
*Get the mutable create bundle.*
- `const std::map< int, boost::uuids::uuid > & getClusterIDMap () const`  
*get the clusterIDMap*
- `const std::map< int, boost::uuids::uuid > & getFibreIDMap () const`  
*get the fibreIDMap*
- `const std::map< int, boost::uuids::uuid > & getPatternChannelIDMap () const`  
*get the patternChannelIDMap*
- `void createCluster (int id, int size, int connectivity)`  
*Translator from config command to actual command.*
- `void connectCluster (int input_cluster_id, int output_cluster_id, int width)`  
*Translator from config command to actual command.*
- `void loadData (std::string datafile)`  
*Translator from config command to actual command.*
- `bool runCommand (const config::ConfigEntry &conf_entry)`
- `void connectPrimaryInputChannel (int channel_id, int outputid)`  
*Translator from config command to actual command.*
- `void connectPrimaryOutputChannel (int channel_id, int inputid)`  
*Translator from config command to actual command.*
- `void autoConnectPrimaryInputs (const std::vector< int > &cluster_ids)`  
*auto connect all the primary input channels to the list of clusters*
- `void autoConnectPrimaryOutputs (const std::vector< int > &cluster_ids)`  
*auto connect all the primary output channels to the list of clusters*

### Static Public Member Functions

- `static bool analyseConfig (const config::ConfigTranslator &conf_trans)`  
*Analyse the config translator for coherence.*
- `static bool checkConfigEntry (const config::ConfigEntry &conf_entry)`  
*Analyse the config entry for coherence.*
- `static bool checkConfigStructure (const std::list< config::ConfigEntry > &conf_entries)`  
*Analyse the config for structural coherence.*
- `static std::map< std::string, std::list< std::string > > getAcceptedCommandList ()`  
*Generate and return the accepted command list.*

### Static Public Attributes

- `static const std::string DEFAULT_DATABASE_FILENAME = "cryomesh_default.db"`
- `static std::map< std::string, std::list< std::string > > acceptedCommandList = Creator::getAcceptedCommandList()`

## Protected Member Functions

- void [initialise](#) ()  
*Helper to initialise the creator.*
- bool [createFromConfigFile](#) (const std::string config\_filename)  
*Run through the config file generating all commands.*
- bool [createFromConfigStream](#) (std::istream &is)

## Private Member Functions

- boost::uuids::uuid [getRealID](#) (const int id, const std::map< int, boost::uuids::uuid > &idmap) const  
*Retreive a uuid from a fake int id inside a map.*
- boost::uuids::uuid [getClusterRealID](#) (const int id) const  
*Helper to retrieve a uuid from a fake int id for clusters.*
- boost::uuids::uuid [getFibreRealID](#) (const int id) const  
*Helper to retrieve a uuid from a fake int id for fibres.*
- boost::uuids::uuid [getPatternChannelRealID](#) (const int id) const  
*Helper to retrieve a uuid from a fake int id for pattern channels.*

## Private Attributes

- std::string [databaseFilename](#)
- boost::shared\_ptr < [structures::Bundle](#) > [bundle](#)
- std::map< int, boost::uuids::uuid > [clusterIDMap](#)
- std::map< int, boost::uuids::uuid > [fibreIDMap](#)
- std::map< int, boost::uuids::uuid > [patternChannelIDMap](#)

### 6.15.1 Detailed Description

Class to take in a config file of ConfigTranslator form and parse the commands to create a full cryomesh object.

- needs to hold a fakeid to realid mapping to facillitate using fake ids in config files before the objects ids are actually known
- holds a unique instance of any create cryomeshes
- holds other data from the config file or defaults such as database name, filename, data filename, etc

Definition at line 31 of file Creator.h.

## 6.15.2 Constructor & Destructor Documentation

### 6.15.2.1 `cryomesh::manager::Creator::Creator ( const std::string & config_filename, const std::string & database_filename = DEFAULT_DATABASE_FILENAME )`

Constructor to create a bundle from a config file name with option to specify the database file name that will be used.

#### Parameters

<code>std::string</code>	The name with full path of the config file
<code>std::string</code>	The name with full path of the database file

Definition at line 47 of file Creator.cpp.

References `createFromConfigFile()`, and `initialise()`.

### 6.15.2.2 `cryomesh::manager::Creator::Creator ( std::istream & config_stream, const std::string & database_filename = DEFAULT_DATABASE_FILENAME )`

Definition at line 61 of file Creator.cpp.

References `createFromConfigStream()`, and `initialise()`.

### 6.15.2.3 `cryomesh::manager::Creator::~Creator ( )` [virtual]

Default destructor.

Definition at line 71 of file Creator.cpp.

## 6.15.3 Member Function Documentation

### 6.15.3.1 `bool cryomesh::manager::Creator::analyseConfig ( const config::ConfigTranslator & conf.trans )` [static]

Analyse the config translator for coherence.

#### Parameters

<i>Config-Translator</i>	The config translator to analyse
--------------------------	----------------------------------

#### Returns

bool True if the config translator passed all the tests for coherence, false otherwise.

Definition at line 195 of file Creator.cpp.

References `checkConfigEntry()`, and `checkConfigStructure()`.

Referenced by `createFromConfigStream()`.

**6.15.3.2** `void cryomesh::manager::Creator::autoConnectPrimaryInputs ( const  
std::vector< int > & cluster_ids )`

auto connect all the primary input channels to the list of clusters

#### Parameters

<code>std- ::vector&lt;int&gt;</code>	The fake ids of the clusters to connect
---	---

Definition at line 356 of file Creator.cpp.

References `bundle`, and `getClusterRealID()`.

Referenced by `runCommand()`.

**6.15.3.3** `void cryomesh::manager::Creator::autoConnectPrimaryOutputs ( const  
std::vector< int > & cluster_ids )`

auto connect all the primary output channels to the list of clusters

#### Parameters

<code>std- ::vector&lt;int&gt;</code>	The fake ids of the clusters to connect
---	---

Definition at line 374 of file Creator.cpp.

References `bundle`, and `getClusterRealID()`.

Referenced by `runCommand()`.

**6.15.3.4** `bool cryomesh::manager::Creator::checkConfigEntry ( const  
config::ConfigEntry & conf_entry ) [static]`

Analyse the config entry for coherence.

#### Parameters

<code>ConfigEntry</code>	The config entry to analyse
--------------------------	-----------------------------

#### Returns

`bool` True if the config entry passed all the tests for coherence, false otherwise.

Definition at line 215 of file Creator.cpp.

References `acceptedCommandList`.

Referenced by `analyseConfig()`.

**6.15.3.5** `bool cryomesh::manager::Creator::checkConfigStructure ( const std::list< config::ConfigEntry > & conf_entries ) [static]`

Analyse the config for structural coherence.

#### Parameters

<i>std- ::list&lt;config- ::Config- Entry&gt;</i>	The list of config entries to analyse for structure
---	---

#### Returns

bool True if the config entries passed all the tests for structural coherence, false otherwise.

Definition at line 245 of file Creator.cpp.

Referenced by analyseConfig().

**6.15.3.6** `void cryomesh::manager::Creator::connectCluster ( int input_cluster_id, int output_cluster_id, int width )`

Translator from config command to actual command.

Connect two clusters using there fake ids

#### Parameters

<i>int</i>	The fake id of the input cluster
<i>int</i>	The fake id of the output cluster
<i>int</i>	The width of the new fibre connection

Definition at line 284 of file Creator.cpp.

References bundle, and getClusterRealID().

Referenced by runCommand().

**6.15.3.7** `void cryomesh::manager::Creator::connectPrimaryInputChannel ( int channel_id, int outputid )`

Translator from config command to actual command.

Create a fibre to connect a primary input pattern channel to a cluster output

#### Parameters

<i>int</i>	The fake id of the pattern channel
<i>The</i>	fake id of the output cluster



Definition at line 342 of file Creator.cpp.

References bundle, getClusterRealID(), and getPatternChannelRealID().

Referenced by runCommand().

**6.15.3.8** void cryomesh::manager::Creator::connectPrimaryOutputChannel ( int *channel\_id*, int *inputid* )

Translator from config command to actual command.

Create a fibre to connect a primary output pattern channel to a cluster output

#### Parameters

<i>int</i>	The fake id of the pattern channel
<i>The</i>	fake id of the input cluster

Definition at line 349 of file Creator.cpp.

References bundle, getClusterRealID(), and getPatternChannelRealID().

Referenced by runCommand().

**6.15.3.9** void cryomesh::manager::Creator::createCluster ( int *id*, int *size*, int *connectivity* )

Translator from config command to actual command.

Create a cluster using a fake id to map to a real one

#### Parameters

<i>int</i>	The fake id of the cluster
<i>int</i>	The size of the cluster
<i>int</i>	The connetivity of the cluster

Definition at line 280 of file Creator.cpp.

References bundle, and clusterIDMap.

Referenced by runCommand().

**6.15.3.10** bool cryomesh::manager::Creator::createFromConfigFile ( const std::string *config\_filename* ) [protected]

Run through the config file generating all commands.

**Returns**

bool True if running the config file was successful, false otherwise

Definition at line 181 of file Creator.cpp.

References createFromConfigStream().

Referenced by Creator().

**6.15.3.11** `bool cryomesh::manager::Creator::createFromConfigStream ( std::istream & is )` [protected]

Definition at line 156 of file Creator.cpp.

References analyseConfig(), and runCommand().

Referenced by createFromConfigFile(), and Creator().

**6.15.3.12** `std::map< std::string, std::list< std::string > > cryomesh::manager::Creator::getAcceptedCommandList ( )` [static]

Generate and return the accepted command list.

**Returns**

std::map<std::string, std::list<std::string> > The accepted commands mapping

Definition at line 22 of file Creator.cpp.

**6.15.3.13** `const boost::shared_ptr< structures::Bundle > cryomesh::manager::Creator::getBundle ( ) const`

Get the create bundle.

**Returns**

boost::shared\_ptr<structures::Bundle> The created bundle

Definition at line 74 of file Creator.cpp.

References bundle.

**6.15.3.14** `const std::map< int, boost::uuids::uuid > & cryomesh::manager::Creator::getClusterIDMap ( ) const`

get the clusterIDMap

**Returns**

const std::map<int, boost::uuids::uuid> the clusterIDMap

Definition at line 82 of file Creator.cpp.

References clusterIDMap.

**6.15.3.15** boost::uuids::uuid cryomesh::manager::Creator::getClusterRealID ( const int *id* ) const [private]

Helper to retrieve a uuid from a fake int id for clusters.

**Parameters**

<i>int</i>	The fake id to translate
------------	--------------------------

**Returns**

boost::uuids::uuid The corresponding real uuid to the fake one, null if it doesnt exist

Definition at line 419 of file Creator.cpp.

References clusterIDMap, and getRealID().

Referenced by autoConnectPrimaryInputs(), autoConnectPrimaryOutputs(), connectCluster(), connectPrimaryInputChannel(), and connectPrimaryOutputChannel().

**6.15.3.16** const std::map< int, boost::uuids::uuid > & cryomesh::manager::Creator::getFibreIDMap ( ) const

get the fibreIDMap

**Returns**

const std::map<int, boost::uuids::uuid> the fibreIDMap

Definition at line 86 of file Creator.cpp.

References fibreIDMap.

**6.15.3.17** boost::uuids::uuid cryomesh::manager::Creator::getFibreRealID ( const int *id* ) const [private]

Helper to retrieve a uuid from a fake int id for fibres.

**Parameters**

<i>int</i>	The fake id to translate
------------	--------------------------

**Returns**

`boost::uuids::uuid` The corresponding real uuid to the fake one, null if it doesnt exist

Definition at line 426 of file `Creator.cpp`.

References `fibreIDMap`, and `getRealID()`.

### 6.15.3.18 `boost::shared_ptr< structures::Bundle > cryomesh::manager::Creator::getMutableBundle ( )`

Get the mutable create bundle.

**Returns**

`boost::shared_ptr<structures::Bundle>` The created bundle

Definition at line 78 of file `Creator.cpp`.

References `bundle`.

### 6.15.3.19 `const std::map< int, boost::uuids::uuid > & cryomesh::manager::Creator::getPatternChannelIDMap ( )` `const`

get the patternChannelIDMap

**Returns**

`const std::map<int, boost::uuids::uuid>` the patternChannelIDMap

Definition at line 90 of file `Creator.cpp`.

References `patternChannelIDMap`.

### 6.15.3.20 `boost::uuids::uuid cryomesh::manager::Creator::getPatternChannelRealID ( const int id ) const` `[private]`

Helper to retrieve a uuid from a fake int id for pattern channels.

**Parameters**

<i>int</i>	The fake id to translate
------------	--------------------------

**Returns**

`boost::uuids::uuid` The corresponding real uuid to the fake one, null if it doesnt exist

Definition at line 433 of file `Creator.cpp`.

References `getRealID()`, and `patternChannelIDMap`.

Referenced by connectPrimaryInputChannel(), and connectPrimaryOutputChannel().

**6.15.3.21** `boost::uuids::uuid cryomesh::manager::Creator::getRealID ( const int id,  
const std::map< int, boost::uuids::uuid > & idmap ) const` [private]

Retreive a uuid from a fake int id inside a map.

#### Parameters

<i>int</i>	The fake id to translate
<i>std::map&lt;int, boost::uuids::uuid&gt;</i>	The map to use for translation

#### Returns

`boost::uuids::uuid` The corresponding real uuid to the fake one, null if it doesnt exist

Definition at line 392 of file Creator.cpp.

Referenced by getClusterRealID(), getFibreRealID(), and getPatternChannelRealID().

**6.15.3.22** `void cryomesh::manager::Creator::initialise ( )` [protected]

Helper to initialise the creator.

Definition at line 94 of file Creator.cpp.

References bundle.

Referenced by Creator().

**6.15.3.23** `void cryomesh::manager::Creator::loadData ( std::string datafile )`

Translator from config command to actual command.

Load the pattern data in from a file

#### Parameters

<i>std::string</i>	The full file path name of the pattern data file
--------------------	--

Definition at line 315 of file Creator.cpp.

References bundle, and patternChannelIDMap.

Referenced by runCommand().

**6.15.3.24** `bool cryomesh::manager::Creator::runCommand ( const config::ConfigEntry & conf_entry )`

Definition at line 98 of file Creator.cpp.

References `autoConnectPrimaryInputs()`, `autoConnectPrimaryOutputs()`, `connectCluster()`, `connectPrimaryInputChannel()`, `connectPrimaryOutputChannel()`, `createCluster()`, and `loadData()`.

Referenced by `createFromConfigStream()`.

## 6.15.4 Member Data Documentation

**6.15.4.1** `std::map< std::string, std::list< std::string > > cryomesh::manager::Creator::acceptedCommandList = Creator::getAcceptedCommandList()`  
[static]

Definition at line 212 of file Creator.h.

Referenced by `checkConfigEntry()`.

**6.15.4.2** `boost::shared_ptr<structures::Bundle> cryomesh::manager::Creator::bundle` [private]

Definition at line 266 of file Creator.h.

Referenced by `autoConnectPrimaryInputs()`, `autoConnectPrimaryOutputs()`, `connectCluster()`, `connectPrimaryInputChannel()`, `connectPrimaryOutputChannel()`, `createCluster()`, `getBundle()`, `getMutableBundle()`, `initialise()`, and `loadData()`.

**6.15.4.3** `std::map<int, boost::uuids::uuid> cryomesh::manager::Creator::clusterIDMap` [private]

Definition at line 273 of file Creator.h.

Referenced by `createCluster()`, `getClusterIDMap()`, and `getClusterRealID()`.

**6.15.4.4** `std::string cryomesh::manager::Creator::databaseFilename` [private]

Definition at line 259 of file Creator.h.

**6.15.4.5** `const std::string cryomesh::manager::Creator::DEFAULT_DATABASE_FILENAME = "cryomesh_default.db"`  
[static]

Definition at line 205 of file Creator.h.

#### 6.15.4.6 `std::map<int, boost::uuids::uuid> cryomesh::manager::Creator::fibreIDMap` [private]

Definition at line 280 of file Creator.h.

Referenced by `getFibreIDMap()`, and `getFibreRealID()`.

#### 6.15.4.7 `std::map<int, boost::uuids::uuid> cryomesh::manager::Creator::pattern-ChannelIDMap` [private]

Definition at line 287 of file Creator.h.

Referenced by `getPatternChannelIDMap()`, `getPatternChannelRealID()`, and `load-Data()`.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Creator.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/Creator.cpp](#)

## 6.16 cryomesh::common::Cycle Class Reference

```
#include <Cycle.h>
```

### Public Member Functions

- [Cycle](#) ()  
*Default Constructor.*
- [Cycle](#) (const long int it)  
*Construct from unsigned long int.*
- [Cycle](#) (mpz\_class mp)  
*Construct from mpz.*
- [Cycle](#) & [operator=](#) (const [Cycle](#) &obj)  
*Assignment operator.*
- const [Cycle](#) [operator+](#) (const [Cycle](#) &obj) const  
*Non-destructive addition operator.*
- bool [operator>](#) (const [Cycle](#) &obj) const  
*Greater than operator.*
- bool [operator<](#) (const [Cycle](#) &obj) const  
*Less than operator.*
- bool [operator>=](#) (const [Cycle](#) &obj) const  
*>= than operator*
- bool [operator<=](#) (const [Cycle](#) &obj) const  
*<= than operator*
- [Cycle](#) & [operator+=](#) (const [Cycle](#) &obj)

*Destructive addition and assignment operator.*

- const [Cycle operator-](#) (const [Cycle](#) &obj) const

*Non-destructive addition operator.*

- [Cycle](#) & [operator+=](#) (const [Cycle](#) &obj)

*Destructive addition and assignment operator.*

- bool [operator==](#) (const [Cycle](#) &obj) const

*Comparator operator.*

- bool [operator!=](#) (const [Cycle](#) &obj) const

*Not comparator operator.*

- [Cycle](#) & [operator++](#) ()

*Prefix increment operator.*

- [Cycle operator++](#) (int)

*Postfix increment operator.*

- [Cycle operator--](#) (int)

*Postfix decrement operator.*

- [Cycle](#) & [operator--](#) ()

*Prefix increment operator.*

- unsigned long int [toULInt](#) () const

*Return as an unsigned integer.*

- long int [toLInt](#) () const

*Return as a signed integer.*

- const mpz\_class & [getMP](#) () const

*Get multiprecision value.*

## Private Attributes

- mpz\_class [cycle](#)

## Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [Cycle](#) &obj)

*To stream operator.*

## 6.16.1 Detailed Description

Definition at line 20 of file Cycle.h.

## 6.16.2 Constructor & Destructor Documentation

### 6.16.2.1 cryomesh::common::Cycle::Cycle ( )

Default Constructor.

Definition at line 15 of file Cycle.cpp.



### 6.16.2.2 cryomesh::common::Cycle::Cycle ( const long int *it* )

Construct from unsigned long int.

#### Parameters

<i>const</i>	unsigned long int The ulong int to construct from
--------------	---

Definition at line 17 of file Cycle.cpp.

### 6.16.2.3 cryomesh::common::Cycle::Cycle ( mpz\_class *mp* )

Construct from mpz.

#### Parameters

<i>mpz_class</i>	mp The multi-precision integer to construct from
------------------	--

Definition at line 21 of file Cycle.cpp.

## 6.16.3 Member Function Documentation

### 6.16.3.1 const mpz\_class & cryomesh::common::Cycle::getMP ( ) const

Get multiprecision value.

#### Returns

mpz\_class

Definition at line 115 of file Cycle.cpp.

References cycle.

Referenced by operator+=( ), operator-=( ), operator<( ), operator<=( ), operator=( ), operator==( ), operator>( ), and operator>=( ).

### 6.16.3.2 bool cryomesh::common::Cycle::operator!=( const Cycle & *obj* ) const

Not comparator operator.

#### Parameters

<i>const</i>	<a href="#">Cycle</a> & obj RHS object
--------------	--

**Returns**

bool True if not equal, false otherwise

Definition at line 68 of file Cycle.cpp.

**6.16.3.3 const Cycle cryomesh::common::Cycle::operator+ ( const Cycle & obj ) const**

Non-destructive addition operator.

**Parameters**

<i>const</i>	<a href="#">Cycle</a> & obj RHS addition
--------------	--

**Returns**

[Cycle](#) New object after addition

Definition at line 30 of file Cycle.cpp.

**6.16.3.4 Cycle & cryomesh::common::Cycle::operator++ ( )**

Prefix increment operator.

**Returns**

[Cycle](#) & Return this

Definition at line 88 of file Cycle.cpp.

References cycle.

**6.16.3.5 Cycle cryomesh::common::Cycle::operator++ ( int )**

Postfix increment operator.

**Returns**

[Cycle](#) & Return this

Definition at line 93 of file Cycle.cpp.

**6.16.3.6 Cycle & cryomesh::common::Cycle::operator+= ( const Cycle & obj )**

Destructive addition and assignment operator.

**Parameters**

<i>const</i>	<a href="#">Cycle</a> & obj RHS addition
--------------	--

**Returns**

[Cycle](#) & This object after addition and assignment

Definition at line 36 of file Cycle.cpp.

References [cycle](#), and [getMP\(\)](#).

**6.16.3.7 const Cycle cryomesh::common::Cycle::operator- ( const Cycle & obj ) const**

Non-destructive addition operator.

**Parameters**

<i>const</i>	<a href="#">Cycle</a> & obj RHS addition
--------------	--

**Returns**

[Cycle](#) New object after addition

Definition at line 41 of file Cycle.cpp.

**6.16.3.8 Cycle cryomesh::common::Cycle::operator-- ( int )**

Postfix decrement operator.

**Returns**

[Cycle](#) & Return this

Definition at line 99 of file Cycle.cpp.

**6.16.3.9 Cycle & cryomesh::common::Cycle::operator-- ( )**

Prefix increment operator.

**Returns**

[Cycle](#) & Return this

Definition at line 105 of file Cycle.cpp.

References [cycle](#).

**6.16.3.10 Cycle & cryomesh::common::Cycle::operator-= ( const Cycle & obj )**

Destructive addition and assignment operator.

**Parameters**

<i>const</i>	<a href="#">Cycle</a> & obj RHS addition
--------------	--

**Returns**

[Cycle](#) & This object after addition and assignment

Definition at line 51 of file Cycle.cpp.

References cycle, and getMP().

**6.16.3.11** `bool cryomesh::common::Cycle::operator< ( const Cycle & obj ) const`

Less than operator.

**Parameters**

<i>const</i>	<a href="#">Cycle</a> & obj RHS addition
--------------	--

**Returns**

bool True if less than obj, false otherwise

Definition at line 76 of file Cycle.cpp.

References cycle, and getMP().

**6.16.3.12** `bool cryomesh::common::Cycle::operator<= ( const Cycle & obj ) const`

<= than operator

**Parameters**

<i>const</i>	<a href="#">Cycle</a> & obj RHS addition
--------------	--

**Returns**

bool True if less than obj, false otherwise

Definition at line 84 of file Cycle.cpp.

References cycle, and getMP().

**6.16.3.13** `Cycle & cryomesh::common::Cycle::operator= ( const Cycle & obj )`

Assignment operator.

## Parameters

<i>const</i>	Cycle & obj RHS assignment
--------------	----------------------------

## Returns

Cycle & This object after assignment

Definition at line 25 of file Cycle.cpp.

References cycle, and getMP().

6.16.3.14 `bool cryomesh::common::Cycle::operator==( const Cycle & obj ) const`

Comparator operator.

## Parameters

<i>const</i>	Cycle & obj RHS object
--------------	------------------------

## Returns

bool True if equal, false otherwise

Definition at line 64 of file Cycle.cpp.

References cycle, and getMP().

6.16.3.15 `bool cryomesh::common::Cycle::operator> ( const Cycle & obj ) const`

Greater than operator.

## Parameters

<i>const</i>	Cycle & obj RHS addition
--------------	--------------------------

## Returns

bool True if > than obj, false otherwise

Definition at line 72 of file Cycle.cpp.

References cycle, and getMP().

6.16.3.16 `bool cryomesh::common::Cycle::operator>= ( const Cycle & obj ) const`

>= than operator

## Parameters

<i>const</i>	<a href="#">Cycle</a> & obj RHS addition
--------------	--

## Returns

bool True if > than obj, false otherwise

Definition at line 80 of file Cycle.cpp.

References cycle, and getMP().

### 6.16.3.17 long int cryomesh::common::Cycle::toLInt ( ) const

Return as an signed integer.

## Returns

signed int The cycle as an int

Definition at line 60 of file Cycle.cpp.

References cycle.

Referenced by cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject(), cryomesh::manager::DatabaseManager::deleteByCycle(), cryomesh::components::Impulse::getActivity(), cryomesh::common::operator<<(), cryomesh::components::Impulse::operator=(), cryomesh::manager::DatabaseManager::selectConnection(), cryomesh::manager::DatabaseManager::selectNode(), cryomesh::manager::DatabaseManager::selectOutputPattern(), cryomesh::manager::DatabaseManager::selectValue(), and cryomesh::manager::DatabaseManager::updateByUID().

### 6.16.3.18 unsigned long int cryomesh::common::Cycle::toULInt ( ) const

Return as an unsigned integer.

## Returns

unsigned int The cycle as an int

Definition at line 56 of file Cycle.cpp.

References cycle.

Referenced by cryomesh::state::Pattern::getDatabaseObject(), cryomesh::components::Node::getDatabaseObject(), cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject(), cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject(), and cryomesh::components::ImpulseCollection::refreshDataObject().

### 6.16.4 Friends And Related Function Documentation

6.16.4.1 `std::ostream& operator<< ( std::ostream & os, const Cycle & obj )` `[friend]`

To stream operator.

#### Parameters

<code>std::ostream</code>	& os The output stream
<code>const</code>	<a href="#">Cycle</a> & obj The object to stream

#### Returns

`std::ostream` & The output stream

Definition at line 110 of file `Cycle.cpp`.

### 6.16.5 Member Data Documentation

6.16.5.1 `mpz_class cryomesh::common::Cycle::cycle` `[private]`

Definition at line 240 of file `Cycle.h`.

Referenced by `getMP()`, `operator++()`, `operator+=()`, `operator--()`, `operator-=()`, `operator<()`, `operator<=()`, `operator=()`, `operator==()`, `operator>()`, `operator>=()`, `toInt()`, and `toULInt()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Cycle.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Cycle.cpp`

## 6.17 cryomesh::manager::DatabaseManager Class Reference

Database manager creates and maintains a database of mesh related objects and data.

```
#include <DatabaseManager.h>
```

### Public Member Functions

- [DatabaseManager](#) (const std::string &dbfile=`DEFAULT_DATABASE`)  
*Default constructor using a filename as the database or a default Opens the file or creates it and creates the tables if it doesnt exist.*
- [DatabaseManager](#) (const [DatabaseManager](#) &obj)
- [DatabaseManager](#) & `operator=` (const [DatabaseManager](#) &obj)  
*Assignment operator.*
- virtual `~DatabaseManager` ()

*Default destructor closes the database.*

- bool `isDatabaseAccessible` () const  
*Check if the database is accessible.*
- void `createTables` ()  
*Create all needed tables.*
- void `clearTables` ()  
*Clear all tables.*
- std::string `clearTable` (const std::string &table)  
*Clear all values in a table.*
- std::string `dropTable` (const std::string &table)  
*Drop a table from database.*
- std::string `insertNode` (const DatabaseObject &db\_object)  
*Insert a node data object into the table.*
- std::string `insertConnection` (const DatabaseObject &db\_object)  
*Insert a connection data object into the table.*
- std::string `insertOutputPattern` (const DatabaseObject &db\_object)
- std::string `selectNode` (const std::string &uuid, const common::Cycle &cycle)  
*Select unique node entry.*
- std::string `selectConnection` (const std::string &uuid, const common::Cycle &cycle)  
*Select unique column entry.*
- std::string `selectOutputPattern` (const std::string &uuid, const common::Cycle &cycle)
- std::string `selectNodeValue` (const std::string &uuid, const common::Cycle &cycle, const std::string &column)  
*Select unique node column entry.*
- std::string `selectConnectionValue` (const std::string &uuid, const common::Cycle &cycle, const std::string &column)  
*Select unique column column entry.*
- std::string `selectOutputPatternValue` (const std::string &uuid, const common::Cycle &cycle, const std::string &column)
- std::string `selectValue` (const std::string &table, const std::string &uuid, const common::Cycle &cycle, const std::string &column)  
*Select unique column column entry from table.*
- std::string `selectNodes` (const std::string &criteria="")  
*Select nodes by a criteria string eg, 'id=erws324 AND cycle=1'.*
- std::string `selectConnections` (const std::string &criteria="")  
*Select connections by a criteria string eg, 'id=erws324 AND cycle=1'.*
- std::string `selectOutputPatterns` (const std::string &criteria="")
- std::string `deleteNode` (const std::string &id)  
*Delete node by uuid.*
- std::string `deleteNodes` (const std::string &criteria="")  
*Delete nodes by a criteria string eg, 'id=erws324 AND cycle=1'.*
- std::string `deleteConnection` (const std::string &id)



*Delete connection by uuid.*

- std::string [deleteConnections](#) (const std::string &criteria="")

*Delete nodes by a criteria string eg, 'id=erwrs324 AND cycle=1'.*

- std::string [deleteOutputPattern](#) (const std::string &id)
- std::string [deleteOutputPatterns](#) (const std::string &criteria="")
- std::string [deleteSelected](#) (const std::string &table, const std::string &criteria="")

*Delete objects from a table by a criteria string eg, 'id=erwrs324 AND cycle=1'.*

- int [countNodes](#) (const std::string &criteria="")

*Count nodes by a criteria string eg, 'cycle=1'.*

- int [countConnections](#) (const std::string &criteria="")

*Count connections by a criteria string eg, 'cycle=1'.*

- int [countRows](#) (const std::string &table, const std::string &criteria="")

*Count objects from a table by a criteria string eg, 'cycle=1'.*

- std::string [updateNode](#) (const std::string &uuid\_str, const [common::Cycle](#) &cycle, const std::string &options)

*update node from using options list*

- std::string [updateConnection](#) (const std::string &uuid\_str, const [common::Cycle](#) &cycle, const std::string &options)

*update node from using options list*

- std::string [updateByUUID](#) (const std::string &uuid\_str, const [common::Cycle](#) &cycle, const std::string &options, const std::string &table)

*update object from a table using options list*

- std::string [select](#) (const std::string &table, const std::string &criteria="")

*Select all columns from table using criteria.*

- std::string [deleteAll](#) (const std::string &table)

*delete all data from table*

- std::string [deleteAllByCycle](#) (const [common::Cycle](#) &cycle, int comparison\_type)

*delete all by cycle*

- std::string [deleteNodesByCycle](#) (const [common::Cycle](#) &cycle, int comparison\_type)

*delete nodes by cycle*

- std::string [deleteConnectionsByCycle](#) (const [common::Cycle](#) &cycle, int comparison\_type)

*delete connections by cycle*

- std::string [deleteByCycle](#) (const std::string &table, const [common::Cycle](#) &cycle, int comparison\_type)

*delete objects from table by cycle*

- std::ostream & [printHistory](#) (std::ostream &os, const [common::Cycle](#) &cycle)

*Print sql history to output stream.*

- std::ostream & [printHistory](#) (std::ostream &os, unsigned int countback=1)

*Print sql history to output stream.*

### Static Public Member Functions

- static int [databaseCallback](#) (void \*unused, int argc, char \*\*argv, char \*\*columnName)  
*Function that is called on finishing an sql command.*

### Static Public Attributes

- static const std::string [DEFAULT\\_DATABASE](#) = "default.db"
- static const std::string [DEFAULT\\_DATABASE\\_PATH](#) = "Output"
- static const [NodeTableFormat](#) [NODES\\_TABLE\\_FORMAT](#)
- static const [ConnectionTableFormat](#) [CONNECTIONS\\_TABLE\\_FORMAT](#)  
*Default connection table format.*
- static const [OutputPatternsTableFormat](#) [OUTPUT\\_PATTERNS\\_TABLE\\_FORMAT](#)

### Protected Member Functions

- std::string [sqlCommand](#) (const std::string &command)  
*Run a provided sql command string.*
- std::string [sqlCommandBySelection](#) (const std::string &table, const std::string &command, const std::string &criteria)

### Static Protected Member Functions

- static const std::multimap < [common::Cycle](#), std::pair < std::string, std::string > > & [addHistoryEntry](#) (const std::string &command, const std::string &results, std::multimap< [common::Cycle](#), std::pair< std::string, std::string > > &map)  
*Add an entry to a historical multimap.*
- static const std::multimap < [common::Cycle](#), std::pair < std::string, std::string > > & [addHistoryEntry](#) (const std::string &command, const std::vector< std::string > &results, std::multimap< [common::Cycle](#), std::pair< std::string, std::string > > &map)  
*Add an list of entries to a historical multimap.*

### Protected Attributes

- sqlite3 \* [database](#)
- int [errorCode](#)
- char \* [errorMessage](#)
- bool [databaseAccess](#)  
*Database accessible.*
- std::multimap< [common::Cycle](#), std::pair< std::string, std::string > > [sqlResults](#)
- std::vector< std::string > [sqlResultsBuffer](#)

### Static Protected Attributes

- static const [common::Cycle](#) MAX\_COMMAND\_HISTORY = [common::Cycle](#)(100)

#### 6.17.1 Detailed Description

Database manager creates and maintains a database of mesh related objects and data.

Definition at line 30 of file DatabaseManager.h.

#### 6.17.2 Constructor & Destructor Documentation

**6.17.2.1 cryomesh::manager::DatabaseManager::DatabaseManager ( const std::string & dbfile = DEFAULT\_DATABASE )**

Default constructor using a filename as the database or a default Opens the file or creates it and creates the tables if it doesnt exist.

##### Parameters

<i>std::string</i>	The name of the database to open/create
--------------------	---

Definition at line 67 of file DatabaseManager.cpp.

References [createTables\(\)](#), [database](#), [databaseAccess](#), [DEFAULT\\_DATABASE\\_PATH](#), and [errorCode](#).

**6.17.2.2 cryomesh::manager::DatabaseManager::DatabaseManager ( const DatabaseManager & obj )**

Definition at line 91 of file DatabaseManager.cpp.

References [database](#), [databaseAccess](#), [errorCode](#), [errorMessage](#), and [sqlResults](#).

**6.17.2.3 cryomesh::manager::DatabaseManager::~DatabaseManager ( )**  
[virtual]

Default destructor closes the database.

Definition at line 99 of file DatabaseManager.cpp.

References [database](#), and [databaseAccess](#).

#### 6.17.3 Member Function Documentation

```
6.17.3.1  const std::multimap< common::Cycle, std::pair< std::string, std::string
> > & cryomesh::manager::DatabaseManager::addHistoryEntry
( const std::string & command, const std::string & results, std::multimap<
common::Cycle, std::pair< std::string, std::string > > & map ) [static,
protected]
```

Add an entry to a historical multimap.

#### Parameters

<i>std::string</i>	Entry to add
<i>std::multimap&lt; std::string, std::string &gt;</i>	Map to add entry to

#### Returns

*std::multimap<std::string, std::string>* Return the modified map

Definition at line 383 of file DatabaseManager.cpp.

References *cryomesh::common::TimeKeeper::getCycle()*, *cryomesh::common::TimeKeeper::getTimeKeeper()*, and *MAX\_COMMAND\_HISTORY*.

Referenced by *addHistoryEntry()*, and *sqlCommand()*.

```
6.17.3.2  const std::multimap< common::Cycle, std::pair< std::string, std::string > > &
cryomesh::manager::DatabaseManager::addHistoryEntry ( const
std::string & command, const std::vector< std::string > & results, std::multimap<
common::Cycle, std::pair< std::string, std::string > > & map ) [static,
protected]
```

Add an list of entries to a historical multimap.

#### Parameters

<i>std::vector&lt; std::string &gt;</i>	Entries to add
<i>std::multimap&lt; std::string, std::string &gt;</i>	Map to add entry to

#### Returns

*std::multimap<std::string, std::string>* Return the modified map

Definition at line 364 of file DatabaseManager.cpp.

References addHistoryEntry().

**6.17.3.3** `std::string cryomesh::manager::DatabaseManager::clearTable ( const std::string & table )`

Clear all values in a table.

#### Parameters

<code>std::string</code>	The table to clear
--------------------------	--------------------

#### Returns

`std::string` The result of the sql query

Definition at line 129 of file DatabaseManager.cpp.

References sqlCommand().

Referenced by clearTables().

**6.17.3.4** `void cryomesh::manager::DatabaseManager::clearTables ( )`

Clear all tables.

Definition at line 123 of file DatabaseManager.cpp.

References clearTable().

**6.17.3.5** `int cryomesh::manager::DatabaseManager::countConnections ( const std::string & criteria = " " )`

Count connections by a criteria string eg, 'cycle=1'.

#### Parameters

<code>std::string</code>	The criteria to match
--------------------------	-----------------------

#### Returns

`int` The result of the count

Definition at line 243 of file DatabaseManager.cpp.

References countRows().

**6.17.3.6** `int cryomesh::manager::DatabaseManager::countNodes ( const std::string & criteria = " " )`

Count nodes by a criteria string eg, 'cycle=1'.

## Parameters

<i>std::string</i>	The criteria to match
--------------------	-----------------------

## Returns

int The result of the count

Definition at line 240 of file DatabaseManager.cpp.

References countRows().

**6.17.3.7** `int cryomesh::manager::DatabaseManager::countRows ( const std::string & table, const std::string & criteria = " " )`

Count objects from a table by a criteria string eg, 'cycle=1'.

## Parameters

<i>std::string</i>	The table to count from
<i>std::string</i>	The criteria to match

## Returns

int The result of the count

Definition at line 247 of file DatabaseManager.cpp.

References sqlCommand().

Referenced by countConnections(), and countNodes().

**6.17.3.8** `void cryomesh::manager::DatabaseManager::createTables ( )`

Create all needed tables.

Definition at line 117 of file DatabaseManager.cpp.

References CONNECTIONS\_TABLE\_FORMAT, NODES\_TABLE\_FORMAT, OUTPUT\_PATTERNS\_TABLE\_FORMAT, and sqlCommand().

Referenced by DatabaseManager().

**6.17.3.9** `int cryomesh::manager::DatabaseManager::databaseCallback ( void * unused, int argc, char ** argv, char ** columnName ) [static]`

Function that is called on finishing an sql command.

Definition at line 32 of file DatabaseManager.cpp.

Referenced by sqlCommand().

6.17.3.10 `std::string cryomesh::manager::DatabaseManager::deleteAll ( const std::string & table )`

delete all data from table

#### Parameters

<code>std::string</code>	Name of table
--------------------------	---------------

#### Returns

`std::string` sql query results

6.17.3.11 `std::string cryomesh::manager::DatabaseManager::deleteAllByCycle ( const common::Cycle & cycle, int comparison_type )`

delete all by cycle

#### Parameters

<code>common::Cycle</code>	Cycle to compare against
<code>int</code>	The type of comparison to make, <0 for less than, ==0 for equals, and >0 for greater than

#### Returns

`std::string` sql query results

Definition at line 299 of file DatabaseManager.cpp.

References `deleteConnectionsByCycle()`, and `deleteNodesByCycle()`.

6.17.3.12 `std::string cryomesh::manager::DatabaseManager::deleteByCycle ( const std::string & table, const common::Cycle & cycle, int comparison_type )`

delete objects from table by cycle

#### Parameters

<code>std::string</code>	The table to delete from
<code>common::Cycle</code>	Cycle to compare against
<code>int</code>	The type of comparison to make, <0 for less than, ==0 for equals, and >0 for greater than

**Returns**

std::string sql query results

Definition at line 312 of file DatabaseManager.cpp.

References sqlCommand(), and cryomesh::common::Cycle::toLint().

Referenced by deleteConnectionsByCycle(), and deleteNodesByCycle().

**6.17.3.13** `std::string cryomesh::manager::DatabaseManager::deleteConnection ( const std::string & id )`

Delete connection by uuid.

**Parameters**

<i>std::string</i>	The uuid to match
--------------------	-------------------

**Returns**

std::string Result of sql query

Definition at line 214 of file DatabaseManager.cpp.

References deleteConnections().

**6.17.3.14** `std::string cryomesh::manager::DatabaseManager::deleteConnections ( const std::string & criteria = " " )`

Delete nodes by a criteria string eg, 'id=erws324 AND cycle=1'.

**Parameters**

<i>std::string</i>	The criteria to match
--------------------	-----------------------

**Returns**

std::string Result of sql query

Definition at line 220 of file DatabaseManager.cpp.

References deleteSelected().

Referenced by deleteConnection().

**6.17.3.15** `std::string cryomesh::manager::DatabaseManager::deleteConnections-ByCycle ( const common::Cycle & cycle, int comparison_type )`

delete connections by cycle



## Parameters

<i>common::Cycle</i>	Cycle to compare against
<i>int</i>	The type of comparison to make, <0 for less than, ==0 for equals, and >0 for greater than

## Returns

std::string sql query results

Definition at line 308 of file DatabaseManager.cpp.

References deleteByCycle().

Referenced by deleteAllByCycle().

**6.17.3.16** std::string cryomesh::manager::DatabaseManager::deleteNode ( const std::string & *id* )

Delete node by uuid.

## Parameters

<i>std::string</i>	The uuid to match
--------------------	-------------------

## Returns

std::string Result of sql query

Definition at line 204 of file DatabaseManager.cpp.

References deleteNodes().

**6.17.3.17** std::string cryomesh::manager::DatabaseManager::deleteNodes ( const std::string & *criteria* = " " )

Delete nodes by a criteria string eg, 'id=erws324 AND cycle=1'.

## Parameters

<i>std::string</i>	The criteria to match
--------------------	-----------------------

## Returns

std::string Result of sql query

Definition at line 210 of file DatabaseManager.cpp.

References deleteSelected().

Referenced by deleteNode().

**6.17.3.18** `std::string cryomesh::manager::DatabaseManager::deleteNodesByCycle ( const common::Cycle & cycle, int comparison_type )`

delete nodes by cycle

#### Parameters

<a href="#"><i>common::Cycle</i></a>	Cycle to compare against
<i>int</i>	The type of comparison to make, <0 for less than, ==0 for equals, and >0 for greater than

#### Returns

std::string sql query results

Definition at line 304 of file DatabaseManager.cpp.

References deleteByCycle().

Referenced by deleteAllByCycle().

**6.17.3.19** `std::string cryomesh::manager::DatabaseManager::deleteOutputPattern ( const std::string & id )`

Definition at line 224 of file DatabaseManager.cpp.

References deleteOutputPatterns().

**6.17.3.20** `std::string cryomesh::manager::DatabaseManager::deleteOutputPatterns ( const std::string & criteria = " " )`

Definition at line 230 of file DatabaseManager.cpp.

References deleteSelected().

Referenced by deleteOutputPattern().

**6.17.3.21** `std::string cryomesh::manager::DatabaseManager::deleteSelected ( const std::string & table, const std::string & criteria = " " )`

Delete objects from a table by a criteria string eg, 'id=erws324 AND cycle=1'.

#### Parameters

<i>std::string</i>	The table to delete from
<i>std::string</i>	The criteria to match

**Returns**

std::string Result of sql query

Definition at line 234 of file DatabaseManager.cpp.

References sqlCommand().

Referenced by deleteConnections(), deleteNodes(), and deleteOutputPatterns().

**6.17.3.22 std::string cryomesh::manager::DatabaseManager::dropTable ( const std::string & table )**

Drop a table from database.

**Parameters**

<i>std::string</i>	The table to drop
--------------------	-------------------

**Returns**

std::string The result of the sql query

Definition at line 292 of file DatabaseManager.cpp.

References sqlCommand().

**6.17.3.23 std::string cryomesh::manager::DatabaseManager::insertConnection ( const DatabaseObject & db\_object )**

Insert a connection data object into the table.

**Parameters**

<i>Database-Object</i>	Database object to insert as a node
------------------------	-------------------------------------

**Returns**

std::string Result of sql query

Definition at line 138 of file DatabaseManager.cpp.

References CONNECTIONS\_TABLE\_FORMAT, cryomesh::manager::DatabaseObject::getInsert(), cryomesh::manager::TableFormat::getName(), and sqlCommand().

**6.17.3.24 std::string cryomesh::manager::DatabaseManager::insertNode ( const DatabaseObject & db\_object )**

Insert a node data object into the table.

## Parameters

<a href="#">Database-Object</a>	Database object to insert as a node
---------------------------------	-------------------------------------

## Returns

std::string Result of sql query

Definition at line 135 of file DatabaseManager.cpp.

References cryomesh::manager::DatabaseObject::getInsert(), cryomesh::manager::TableFormat::getName(), NODES\_TABLE\_FORMAT, and sqlCommand().

**6.17.3.25** `std::string cryomesh::manager::DatabaseManager::insertOutputPattern ( const DatabaseObject & db_object )`

Definition at line 141 of file DatabaseManager.cpp.

References cryomesh::manager::DatabaseObject::getInsert(), cryomesh::manager::TableFormat::getName(), OUTPUT\_PATTERNS\_TABLE\_FORMAT, and sqlCommand().

**6.17.3.26** `bool cryomesh::manager::DatabaseManager::isDatabaseAccessable ( ) const`

Check if the database is accessible.

## Returns

bool True if deemed accessible, false otherwise

Definition at line 113 of file DatabaseManager.cpp.

References databaseAccess.

**6.17.3.27** `DatabaseManager & cryomesh::manager::DatabaseManager::operator= ( const DatabaseManager & obj )`

Assignment operator.

## Parameters

<i>const</i>	<a href="#">DatabaseManager</a> & obj RHS assignment
--------------	--

## Returns

[DatabaseManager](#) & This object after assignment

Definition at line 104 of file DatabaseManager.cpp.

References database, databaseAccess, errorCode, errorMessage, and sqlResults.

**6.17.3.28** `std::ostream & cryomesh::manager::DatabaseManager::printHistory (std::ostream & os, const common::Cycle & cycle )`

Print sql history to output stream.

#### Parameters

<i>std::ostream</i>	Output stream to print to
<i>Cycle</i>	The cycle to print information on

#### Returns

`std::ostream` Return the supplied output stream

Definition at line 406 of file DatabaseManager.cpp.

References sqlResults.

Referenced by printHistory().

**6.17.3.29** `std::ostream & cryomesh::manager::DatabaseManager::printHistory (std::ostream & os, unsigned int countback = 1 )`

Print sql history to output stream.

#### Parameters

<i>std::ostream</i>	Output stream to print to
<i>unsigned</i>	int Muber of cycles of previous history to print

#### Returns

`std::ostream` Return the supplied output stream

Definition at line 396 of file DatabaseManager.cpp.

References cryomesh::common::TimeKeeper::getCycle(), cryomesh::common::TimeKeeper::getTimeKeeper(), and printHistory().

**6.17.3.30** `std::string cryomesh::manager::DatabaseManager::select ( const std::string & table, const std::string & criteria = " " )`

Select all columns from table using criteria.

#### Parameters

<i>std::string</i>	Name of table
<i>std::string</i>	Selection criteria

**Returns**

std::string sql query results

Definition at line 200 of file DatabaseManager.cpp.

References sqlCommandBySelection().

Referenced by selectConnections(), selectNodes(), and selectOutputPatterns().

**6.17.3.31** `std::string cryomesh::manager::DatabaseManager::selectConnection ( const std::string & uuid, const common::Cycle & cycle )`

Select unique column entry.

**Parameters**

<i>std::string</i>	The uuid of the node
<i>Cycle</i>	The cycle to select on, to force uniqueness

**Returns**

std::string The value of the entry

Definition at line 150 of file DatabaseManager.cpp.

References sqlCommandBySelection(), and cryomesh::common::Cycle::toInt().

**6.17.3.32** `std::string cryomesh::manager::DatabaseManager::selectConnections ( const std::string & criteria = " " )`

Select connections by a criteria string eg, 'id=erwrs324 AND cycle=1'.

**Parameters**

<i>std::string</i>	The criteria to match
--------------------	-----------------------

**Returns**

std::string Result of sql query

Definition at line 194 of file DatabaseManager.cpp.

References select().

**6.17.3.33** `std::string cryomesh::manager::DatabaseManager::selectConnection-Value ( const std::string & uuid, const common::Cycle & cycle, const std::string & column )`

Select unique column column entry.

## Parameters

<i>std::string</i>	The uuid of the node
<i>Cycle</i>	The cycle to select on, to force uniqueness
<i>std::string</i>	The column to select

## Returns

std::string The value of the entry

Definition at line 169 of file DatabaseManager.cpp.

References selectValue().

**6.17.3.34** `std::string cryomesh::manager::DatabaseManager::selectNode ( const  
std::string & uuid, const common::Cycle & cycle )`

Select unique node entry.

## Parameters

<i>std::string</i>	The uuid of the node
<i>Cycle</i>	The cycle to select on, to force uniqueness

## Returns

std::string The value of the entry

Definition at line 144 of file DatabaseManager.cpp.

References sqlCommandBySelection(), and cryomesh::common::Cycle::toLInt().

**6.17.3.35** `std::string cryomesh::manager::DatabaseManager::selectNodes ( const  
std::string & criteria = " " )`

Select nodes by a criteria string eg, 'id=erwrs324 AND cycle=1'.

## Parameters

<i>std::string</i>	The criteria to match
--------------------	-----------------------

## Returns

std::string Result of sql query

Definition at line 191 of file DatabaseManager.cpp.

References select().

**6.17.3.36** `std::string cryomesh::manager::DatabaseManager::selectNodeValue ( const std::string & uuid, const common::Cycle & cycle, const std::string & column )`

Select unique node column entry.

#### Parameters

<i>std::string</i>	The uuid of the node
<i>Cycle</i>	The cycle to select on, to force uniqueness
<i>std::string</i>	The column to select

#### Returns

`std::string` The value of the entry

Definition at line 164 of file DatabaseManager.cpp.

References `selectValue()`.

**6.17.3.37** `std::string cryomesh::manager::DatabaseManager::selectOutputPattern ( const std::string & uuid, const common::Cycle & cycle )`

Definition at line 157 of file DatabaseManager.cpp.

References `sqlCommandBySelection()`, and `cryomesh::common::Cycle::toLint()`.

**6.17.3.38** `std::string cryomesh::manager::DatabaseManager::selectOutputPatterns ( const std::string & criteria = " " )`

Definition at line 197 of file DatabaseManager.cpp.

References `select()`.

**6.17.3.39** `std::string cryomesh::manager::DatabaseManager::selectOutputPattern-Value ( const std::string & uuid, const common::Cycle & cycle, const std::string & column )`

Definition at line 174 of file DatabaseManager.cpp.

References `selectValue()`.

**6.17.3.40** `std::string cryomesh::manager::DatabaseManager::selectValue ( const std::string & table, const std::string & uuid, const common::Cycle & cycle, const std::string & column )`

Select unique column column entry from table.



## Parameters

<i>std::string</i>	The table to utilise
<i>std::string</i>	The uuid of the node
<i>Cycle</i>	The cycle to select on, to force uniqueness
<i>std::string</i>	The column to select

## Returns

*std::string* The value of the entry

Definition at line 179 of file DatabaseManager.cpp.

References `sqlCommand()`, and `cryomesh::common::Cycle::toLint()`.

Referenced by `selectConnectionValue()`, `selectNodeValue()`, and `selectOutputPatternValue()`.

**6.17.3.41** `std::string cryomesh::manager::DatabaseManager::sqlCommand ( const std::string & command )` *[protected]*

Run a provided sql command string.

## Parameters

<i>std::string</i>	The command string to run
--------------------	---------------------------

## Returns

*std::vector<std::string>* vector of results

Definition at line 329 of file DatabaseManager.cpp.

References `addHistoryEntry()`, `database`, `databaseCallback()`, `errorCode`, `errorMessage`, `sqlResults`, and `sqlResultsBuffer`.

Referenced by `clearTable()`, `countRows()`, `createTables()`, `deleteByCycle()`, `deleteSelected()`, `dropTable()`, `insertConnection()`, `insertNode()`, `insertOutputPattern()`, `selectValue()`, `sqlCommandBySelection()`, and `updateByUUID()`.

**6.17.3.42** `std::string cryomesh::manager::DatabaseManager::sqlCommandBySelection ( const std::string & table, const std::string & command, const std::string & criteria )` *[protected]*

Definition at line 281 of file DatabaseManager.cpp.

References `sqlCommand()`.

Referenced by `select()`, `selectConnection()`, `selectNode()`, and `selectOutputPattern()`.

6.17.3.43 `std::string cryomesh::manager::DatabaseManager::updateByUUID ( const std::string & uuid_str, const common::Cycle & cycle, const std::string & options, const std::string & table )`

update object from a table using options list

#### Parameters

<code>std::string</code>	The id to match
<code>common::Cycle</code>	The cycle to match
<code>std::string</code>	The options to set
<code>std::string</code>	The table to use

#### Returns

int The result of the count

Definition at line 273 of file DatabaseManager.cpp.

References `sqlCommand()`, and `cryomesh::common::Cycle::toLint()`.

Referenced by `updateConnection()`, and `updateNode()`.

6.17.3.44 `std::string cryomesh::manager::DatabaseManager::updateConnection ( const std::string & uuid_str, const common::Cycle & cycle, const std::string & options )`

update node from using options list

#### Parameters

<code>std::string</code>	The id to match
<code>common::Cycle</code>	The cycle to match
<code>std::string</code>	The options to set

#### Returns

int The result of the count

Definition at line 269 of file DatabaseManager.cpp.

References `updateByUUID()`.

6.17.3.45 `std::string cryomesh::manager::DatabaseManager::updateNode ( const std::string & uuid_str, const common::Cycle & cycle, const std::string & options )`

update node from using options list

## Parameters

<i>std::string</i>	The id to match
<i>common::-</i> <i>Cycle</i>	The cycle to match
<i>std::string</i>	The options to set

## Returns

int The result of the count

Definition at line 265 of file DatabaseManager.cpp.

References `updateByUUID()`.

### 6.17.4 Member Data Documentation

**6.17.4.1** `const ConnectionTableFormat cryomesh::manager::-`  
`DatabaseManager::CONNECTIONS_TABLE_FORMAT`  
[static]

Default connection table format.

Definition at line 496 of file DatabaseManager.h.

Referenced by `createTables()`, and `insertConnection()`.

**6.17.4.2** `sqlite3* cryomesh::manager::DatabaseManager::database`  
[protected]

Definition at line 506 of file DatabaseManager.h.

Referenced by `DatabaseManager()`, `operator=()`, `sqlCommand()`, and `~DatabaseManager()`.

**6.17.4.3** `bool cryomesh::manager::DatabaseManager::databaseAccess`  
[protected]

Database accessible.

Definition at line 527 of file DatabaseManager.h.

Referenced by `DatabaseManager()`, `isDatabaseAccessable()`, `operator=()`, and `~DatabaseManager()`.

**6.17.4.4** `const std::string cryomesh::manager::DatabaseManager::DEFAULT_DAT-`  
`ABASE = "default.db"` [static]

Definition at line 478 of file DatabaseManager.h.

**6.17.4.5** `const std::string cryomesh::manager::DatabaseManager::DEFAULT_DATABASE_PATH = "Output" [static]`

Definition at line 479 of file DatabaseManager.h.

Referenced by DatabaseManager().

**6.17.4.6** `int cryomesh::manager::DatabaseManager::errorCode [protected]`

Definition at line 513 of file DatabaseManager.h.

Referenced by DatabaseManager(), operator=(), and sqlCommand().

**6.17.4.7** `char* cryomesh::manager::DatabaseManager::errorMessage [protected]`

Definition at line 520 of file DatabaseManager.h.

Referenced by DatabaseManager(), operator=(), and sqlCommand().

**6.17.4.8** `const common::Cycle cryomesh::manager::DatabaseManager::MAX_COMMAND_HISTORY = common::Cycle(100) [static, protected]`

Definition at line 548 of file DatabaseManager.h.

Referenced by addHistoryEntry().

**6.17.4.9** `const NodeTableFormat cryomesh::manager::DatabaseManager::NODES_TABLE_FORMAT [static]`

Definition at line 491 of file DatabaseManager.h.

Referenced by createTables(), and insertNode().

**6.17.4.10** `const OutputPatternsTableFormat cryomesh::manager::DatabaseManager::OUTPUT_PATTERNS_TABLE_FORMAT [static]`

Definition at line 498 of file DatabaseManager.h.

Referenced by createTables(), and insertOutputPattern().

**6.17.4.11** `std::multimap<common::Cycle, std::pair<std::string, std::string> > cryomesh::manager::DatabaseManager::sqlResults [protected]`

Definition at line 534 of file DatabaseManager.h.

Referenced by DatabaseManager(), operator=(), printHistory(), and sqlCommand().

#### 6.17.4.12 `std::vector<std::string>` `cryomesh::manager::DatabaseManager::sql-ResultsBuffer` [protected]

Definition at line 541 of file `DatabaseManager.h`.

Referenced by `sqlCommand()`.

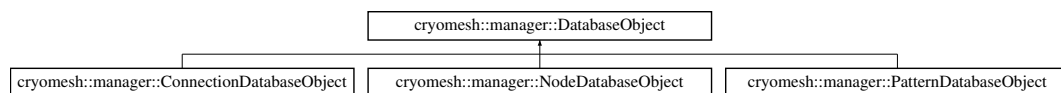
The documentation for this class was generated from the following files:

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/DatabaseManager.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/DatabaseManager.cpp](#)

## 6.18 cryomesh::manager::DatabaseObject Class Reference

```
#include <DatabaseObject.h>
```

Inheritance diagram for `cryomesh::manager::DatabaseObject`:



### Public Member Functions

- [DatabaseObject](#) ()
- virtual [~DatabaseObject](#) ()
- `std::string` [getKey](#) (const `std::string` &key) const  
*Return the string object associated with a key.*
- virtual `std::string` [getInsert](#) (const `std::string` &table) const =0  
*Get the string that can be used to insert the sql data.*

### Static Public Member Functions

- static `std::string` [findValue](#) (const `std::string` &entry, const `std::map`< `std::string`, `std::string` > &map)  
*Find entries value in map or return null.*
- static `std::map`< `std::string`, `std::string` > [getColumnMapFromEntry](#) (const `std::string` &entry)  
*Parse a string database entry, extract columns and values and return a map.*
- `template`<class T >  
static `std::string` [toString](#) (T obj)  
*Convert an templated object that can be piped to a stream to a string.*

## Protected Attributes

- `std::map< std::string, std::string >` [columns](#)

### 6.18.1 Detailed Description

Definition at line 21 of file DatabaseObject.h.

### 6.18.2 Constructor & Destructor Documentation

#### 6.18.2.1 `cryomesh::manager::DatabaseObject::DatabaseObject ( )` `[inline]`

Definition at line 23 of file DatabaseObject.h.

#### 6.18.2.2 `virtual cryomesh::manager::DatabaseObject::~~DatabaseObject ( )` `[inline, virtual]`

Definition at line 25 of file DatabaseObject.h.

### 6.18.3 Member Function Documentation

#### 6.18.3.1 `static std::string cryomesh::manager::DatabaseObject::findValue ( const std::string & entry, const std::map< std::string, std::string > & map )` `[inline, static]`

Find entries value in map or return null.

#### Parameters

<code>std::string</code>	Entry to find
<code>std::map&lt; std::string, std::string &gt;</code>	map to search

#### Returns

Value of entry

Definition at line 59 of file DatabaseObject.h.

Referenced by `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject()`, `cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject()`, and `cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject()`.

**6.18.3.2** `static std::map<std::string, std::string> cryomesh::manager::DatabaseObject::getColumnMapFromEntry ( const std::string & entry ) [inline, static]`

Parse a string database entry, extract columns and values and return a map.

Definition at line 72 of file DatabaseObject.h.

Referenced by `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject()`, `cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject()`, and `cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject()`.

**6.18.3.3** `virtual std::string cryomesh::manager::DatabaseObject::getInsert ( const std::string & table ) const [pure virtual]`

Get the string that can be used to insert the sql data.

#### Returns

the sql command string to insert into this table

Implemented in `cryomesh::manager::ConnectionDatabaseObject`, `cryomesh::manager::NodeDatabaseObject`, and `cryomesh::manager::PatternDatabaseObject`.

Referenced by `cryomesh::manager::DatabaseManager::insertConnection()`, `cryomesh::manager::DatabaseManager::insertNode()`, and `cryomesh::manager::DatabaseManager::insertOutputPattern()`.

**6.18.3.4** `std::string cryomesh::manager::DatabaseObject::getKey ( const std::string & key ) const [inline]`

Return the string object associated with a key.

`::string` The key to search for

#### Returns

`std::string` The object associated with the search key, "" if not found

Definition at line 37 of file DatabaseObject.h.

References columns.

Referenced by `cryomesh::manager::PatternDatabaseObject::getInsert()`, `cryomesh::manager::NodeDatabaseObject::getInsert()`, and `cryomesh::manager::ConnectionDatabaseObject::getInsert()`.

**6.18.3.5** `template<class T > static std::string cryomesh::manager::DatabaseObject::toString ( T obj ) [inline, static]`

Convert an templated object that can be piped to a stream to a string.

## Parameters

<code>T</code>	The object to get a string for
----------------	--------------------------------

Definition at line 108 of file DatabaseObject.h.

#### 6.18.4 Member Data Documentation

##### 6.18.4.1 `std::map<std::string, std::string> cryomesh::manager::DatabaseObject::columns` [protected]

Definition at line 119 of file DatabaseObject.h.

Referenced by `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject()`, `getKey()`, `cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject()`, and `cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject()`.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/DatabaseObject.h](#)

### 6.19 `cryomesh::dataobjects::DataObject< U, T >` Class Template Reference

Class to contain all the useful data about an object.

```
#include <DataObject.h>
```

#### Public Types

- enum `ComparisonType` { `MaximumValue`, `MinimumValue`, `EqualityValue`, `AverageValue` }
- Enum to signal the type of comparison to make.*

#### Public Member Functions

- `DataObject()`  
*Default constructor.*
- `DataObject(unsigned int sz)`  
*Constructor with max size.*
- virtual `~DataObject()`  
*Default destructor.*
- void `enableLogging` (bool enable)  
*Whether logging is enabled or not.*
- bool `isLoggingEnabled()`



*Check logging is enabled or not.*

- const std::map< U, T > & [getMap](#) () const  
*Get all cycle values.*
- std::map< U, T > & [getMutableMap](#) ()  
*Get all mutable cycle values.*
- const std::map< U, T > [getMap](#) (U start, U end) const  
*Get all values within a range [start, end].*
- const std::map< U, T > [getMap](#) (U start) const  
*Get all values within a range [start, ].*
- const T [getByKey](#) (U key) const  
*Get value from key.*
- void [clear](#) ()  
*Clear all data.*
- void [insert](#) (U key, T object)  
*Add entry.*
- unsigned int [getDatasetMaximumSize](#) () const
- void [setDatasetMaximumSize](#) (unsigned int sz)
- T [getValueComparison](#) (ComparisonType type) const  
*Get comparison values.*
- T [getMaximumValue](#) () const  
*Get maximum value.*
- T [getMinimumValue](#) () const  
*Get minimum value.*
- T [getAverageValue](#) () const  
*Get average value value.*

### Protected Attributes

- std::map< U, T > [valueMap](#)
- bool [loggingEnabled](#)
- unsigned int [datasetMaximumSize](#)

### Static Protected Attributes

- static const unsigned int [DEFAULT\\_DATASET\\_SIZE](#) = 100

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [DataObject](#) &obj)  
*To stream operator.*

### 6.19.1 Detailed Description

```
template<class U, class T>class cryomesh::dataobjects::DataObject< U, T >
```

Class to contain all the useful data about an object.

Useful for output and plotting

Definition at line 22 of file DataObject.h.

### 6.19.2 Member Enumeration Documentation

```
6.19.2.1 template<class U, class T> enum cryomesh::dataobjects::DataObject< U, T
>::ComparisonType
```

Enum to signal the type of comparison to make.

Enumerator:

***MaximumValue***

***MinimumValue***

***EqualityValue***

***AverageValue***

Definition at line 29 of file DataObject.h.

### 6.19.3 Constructor & Destructor Documentation

```
6.19.3.1 template<class U, class T> cryomesh::dataobjects::DataObject< U, T
>::DataObject ( ) [inline]
```

Default constructor.

Definition at line 36 of file DataObject.h.

```
6.19.3.2 template<class U, class T> cryomesh::dataobjects::DataObject< U, T
>::DataObject ( unsigned int sz ) [inline]
```

Constructor with max size.

Parameters

<i>unsigned</i>	int The maximum size of the data set
-----------------	--------------------------------------

Definition at line 46 of file DataObject.h.

## 6.19 cryomesh::dataobjects::DataObject< U, T > Class Template Reference 183

6.19.3.3 `template<class U, class T> virtual cryomesh::dataobjects::DataObject< U, T >::~DataObject ( ) [inline, virtual]`

Default destructor.

Definition at line 53 of file DataObject.h.

### 6.19.4 Member Function Documentation

6.19.4.1 `template<class U, class T> void cryomesh::dataobjects::DataObject< U, T >::clear ( ) [inline]`

Clear all data.

Definition at line 178 of file DataObject.h.

Referenced by `cryomesh::components::ImpulseCollection::refreshDataObject()`.

6.19.4.2 `template<class U, class T> void cryomesh::dataobjects::DataObject< U, T >::enableLogging ( bool enable ) [inline]`

Whether logging is enabled or not.

#### Parameters

<i>bool</i>	enable True to enable logging, false otherwise
-------------	--

Definition at line 62 of file DataObject.h.

6.19.4.3 `template<class U, class T> T cryomesh::dataobjects::DataObject< U, T >::getAverageValue ( ) const [inline]`

Get average value value.

#### Returns

T The resultant value

Definition at line 283 of file DataObject.h.

6.19.4.4 `template<class U, class T> const T cryomesh::dataobjects::DataObject< U, T >::getByKey ( U key ) const [inline]`

Get value from key.

#### Parameters

<i>U</i>	key The key to find
----------	---------------------

**Returns**

T The value found

Definition at line 138 of file DataObject.h.

**6.19.4.5** `template<class U, class T> unsigned int cryomesh::dataobjects::DataObject< U, T >::getDatasetMaximumSize ( ) const [inline]`

Definition at line 212 of file DataObject.h.

Referenced by `cryomesh::dataobjects::DataObject< common::Cycle, double >::insert()`, and `cryomesh::components::ImpulseCollection::refreshDataObject()`.

**6.19.4.6** `template<class U, class T> const std::map<U, T>& cryomesh::dataobjects::DataObject< U, T >::getMap ( ) const [inline]`

Get all cycle values.

**Returns**

`std::map<unsigned long int, double>` & The cycle values

Definition at line 81 of file DataObject.h.

Referenced by `cryomesh::components::Node::getActivities()`.

**6.19.4.7** `template<class U, class T> const std::map<U, T> cryomesh::dataobjects::DataObject< U, T >::getMap ( U start, U end ) const [inline]`

Get all values within a range [start, end].

**Parameters**

<i>U</i>	start The start cycle of the range
<i>U</i>	end The end cycle of the range

## 6.19 cryomesh::dataobjects::DataObject< U, T > Class Template Reference 185

### Returns

std::map<unsigned long int, double> The cycle values

Definition at line 106 of file DataObject.h.

```
6.19.4.8  template<class U, class T> const std::map<U, T> cryomesh-  
::dataobjects::DataObject< U, T >::getMap ( U start ) const  
[inline]
```

Get all values within a range [start, ].

### Parameters

<i>U</i>	start The start cycle of the range
----------	------------------------------------

### Returns

std::map<unsigned long int, double> The cycle values

Definition at line 122 of file DataObject.h.

```
6.19.4.9  template<class U, class T> T cryomesh::dataobjects::DataObject< U, T  
>::getMaximumValue ( ) const [inline]
```

Get maximum value.

### Returns

T The resultant value

Definition at line 263 of file DataObject.h.

```
6.19.4.10 template<class U, class T> T cryomesh::dataobjects::DataObject< U, T  
>::getMinimumValue ( ) const [inline]
```

Get minimum value.

### Returns

T The resultant value

Definition at line 273 of file DataObject.h.

```
6.19.4.11 template<class U, class T> std::map<U, T>& cryomesh-  
::dataobjects::DataObject< U, T >::getMutableMap ( )  
[inline]
```

Get all mutable cycle values.

**Returns**

std::map<U, T> & The mutable cycle values

Definition at line 91 of file DataObject.h.

**6.19.4.12** `template<class U, class T> T cryomesh::dataobjects::DataObject< U, T >::getValueComparison ( ComparisonType type ) const [inline]`

Get comparison values.

**Parameters**

<i>Comparison-Type</i>	type The type of comparison to make
------------------------	-------------------------------------

**Returns**

T The result of the comparison

Definition at line 229 of file DataObject.h.

Referenced by cryomesh::dataobjects::DataObject< common::Cycle, double >::getAverageValue(), cryomesh::dataobjects::DataObject< common::Cycle, double >::getMaximumValue(), and cryomesh::dataobjects::DataObject< common::Cycle, double >::getMinimumValue().

**6.19.4.13** `template<class U, class T> void cryomesh::dataobjects::DataObject< U, T >::insert ( U key, T object ) [inline]`

Add entry.

**Parameters**

<i>unsigned</i>	int cycle The cycle the value is on
<i>double</i>	The value

Definition at line 191 of file DataObject.h.

Referenced by cryomesh::components::Node::addActivity(), cryomesh::components::ImpulseCollection::refreshDataObject(), and cryomesh::components::Node::update().

**6.19.4.14** `template<class U, class T> bool cryomesh::dataobjects::DataObject< U, T >::isLoggingEnabled ( ) [inline]`

Check logging is enabled or not.

## 6.19 cryomesh::dataobjects::DataObject< U, T > Class Template Reference 187

### Returns

bool enable True if logging enabled, false otherwise

Definition at line 71 of file DataObject.h.

Referenced by cryomesh::components::ImpulseCollection::refreshDataObject(), and cryomesh::components::Node::update().

**6.19.4.15** `template<class U, class T> void cryomesh::dataobjects::DataObject< U, T >::setDatasetMaximumSize ( unsigned int sz ) [inline]`

Definition at line 216 of file DataObject.h.

Referenced by cryomesh::components::Node::Node().

## 6.19.5 Friends And Related Function Documentation

**6.19.5.1** `template<class U, class T> std::ostream& operator<< ( std::ostream & os, const DataObject< U, T > & obj ) [friend]`

To stream operator.

### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">DataObject</a> & obj The object to stream

### Returns

std::ostream & The output stream

Definition at line 159 of file DataObject.h.

## 6.19.6 Member Data Documentation

**6.19.6.1** `template<class U, class T> unsigned int cryomesh::dataobjects::DataObject< U, T >::datasetMaximumSize [protected]`

Definition at line 306 of file DataObject.h.

Referenced by cryomesh::dataobjects::DataObject< common::Cycle, double >::getDatasetMaximumSize(), and cryomesh::dataobjects::DataObject< common::Cycle, double >::setDatasetMaximumSize().

```
6.19.6.2 template<class U, class T> const unsigned int cryomesh::dataobjects::-
        DataObject< U, T >::DEFAULT_DATASET_SIZE = 100 [static,
        protected]
```

Definition at line 313 of file DataObject.h.

```
6.19.6.3 template<class U, class T> bool cryomesh::dataobjects::DataObject< U, T
        >::loggingEnabled [protected]
```

Definition at line 299 of file DataObject.h.

Referenced by cryomesh::dataobjects::DataObject< common::Cycle, double >::enableLogging(), and cryomesh::dataobjects::DataObject< common::Cycle, double >::isLoggingEnabled().

```
6.19.6.4 template<class U, class T> std::map<U, T> cryomesh::dataobjects::Data-
        Object< U, T >::valueMap [protected]
```

Definition at line 292 of file DataObject.h.

Referenced by cryomesh::dataobjects::DataObject< common::Cycle, double >::clear(), cryomesh::dataobjects::DataObject< common::Cycle, double >::getByKey(), cryomesh::dataobjects::DataObject< common::Cycle, double >::getMap(), cryomesh::dataobjects::DataObject< common::Cycle, double >::getMutableMap(), cryomesh::dataobjects::DataObject< common::Cycle, double >::getValueComparison(), and cryomesh::dataobjects::DataObject< common::Cycle, double >::insert().

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

- </home/niall/Projects/Eclipse/CPP/cryomesh/src/dataobjects/DataObject.h>

## 6.20 cryomesh::dataobjects::DataObjectController< U, T > Class Template Reference

Class used to interface with data objects.

```
#include <DataObjectController.h>
```

### Public Member Functions

- [DataObjectController](#) ()  
*Default constructor.*
- [DataObjectController](#) (unsigned int sz)  
*Constructor with size.*
- virtual [~DataObjectController](#) ()
- virtual void [enableLogging](#) (bool enable)



*Whether logging is enabled or not.*

- virtual const std::map< U, T > & [getMap](#) ()

*Get all cycle values.*

- virtual const [dataobjects::DataObject](#)< U, T > & [getDataObject](#) ()

*Get data object.*

- virtual void [refreshDataObject](#) ()

*Function to allow refreshing implementation if required by subclasses.*

## Protected Attributes

- [dataobjects::DataObject](#)< U, T > [dataObject](#)

### 6.20.1 Detailed Description

```
template<class U, class T>class cryomesh::dataobjects::DataObjectController< U, T >
```

Class used to interface with data objects.

Definition at line 21 of file DataObjectController.h.

### 6.20.2 Constructor & Destructor Documentation

6.20.2.1 `template<class U, class T> cryomesh::dataobjects::DataObjectController< U, T >::DataObjectController ( )` `[inline]`

Default constructor.

Definition at line 27 of file DataObjectController.h.

6.20.2.2 `template<class U, class T> cryomesh::dataobjects::DataObjectController< U, T >::DataObjectController ( unsigned int sz )` `[inline]`

Constructor with size.

#### Parameters

<i>unsigned</i>	int The maximum size of the data set
-----------------	--------------------------------------

Definition at line 36 of file DataObjectController.h.

6.20.2.3 `template<class U, class T> virtual cryomesh::dataobjects::DataObjectController< U, T >::~DataObjectController ( )` `[inline, virtual]`

Definition at line 39 of file DataObjectController.h.

### 6.20.3 Member Function Documentation

6.20.3.1 `template<class U, class T> virtual void cryomesh::dataobjects::DataObjectController< U, T >::enableLogging ( bool enable ) [inline, virtual]`

Whether logging is enabled or not.

#### Parameters

<i>bool</i>	enable True to enable logging, false otherwise
-------------	--

Definition at line 47 of file DataObjectController.h.

6.20.3.2 `template<class U, class T> virtual const dataobjects::DataObject<U, T>& cryomesh::dataobjects::DataObjectController< U, T >::getDataObject ( ) [inline, virtual]`

Get data object.

#### Returns

dataobjects::DataObject<U,T> & The data object

Definition at line 68 of file DataObjectController.h.

6.20.3.3 `template<class U, class T> virtual const std::map<U, T>& cryomesh::dataobjects::DataObjectController< U, T >::getMap ( ) [inline, virtual]`

Get all cycle values.

#### Returns

std::map<unsigned long int, double> & The cycle values

Definition at line 57 of file DataObjectController.h.

6.20.3.4 `template<class U, class T> virtual void cryomesh::dataobjects::DataObjectController< U, T >::refreshDataObject ( ) [inline, virtual]`

Function to allow refreshing implementation if required by subclasses.

Reimplemented in [cryomesh::components::ImpulseCollection](#).

Definition at line 76 of file DataObjectController.h.

Referenced by `cryomesh::dataobjects::DataObjectController< unsigned long int, double >::getDataObject()`, and `cryomesh::dataobjects::DataObjectController< unsigned long int, double >::getMap()`.

#### 6.20.4 Member Data Documentation

6.20.4.1 `template<class U, class T> dataobjects::DataObject<U, T>`  
`cryomesh::dataobjects::DataObjectController< U, T >::dataObject`  
`[protected]`

Definition at line 85 of file DataObjectController.h.

Referenced by `cryomesh::dataobjects::DataObjectController< unsigned long int, double >::DataObjectController()`, `cryomesh::dataobjects::DataObjectController< unsigned long int, double >::enableLogging()`, `cryomesh::dataobjects::DataObjectController< unsigned long int, double >::getDataObject()`, and `cryomesh::dataobjects::DataObjectController< unsigned long int, double >::getMap()`.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/dataobjects/DataObjectController.h](#)

## 6.21 cryomesh::manipulators::IClusterAnalyser::EnergyVariation-WeightingMap Struct Reference

```
#include <IClusterAnalyser.h>
```

### Public Member Functions

- [EnergyVariationWeightingMap \(\)](#)

### Public Attributes

- `std::map< EnergyVariation, double > variationMap`

#### 6.21.1 Detailed Description

Definition at line 150 of file IClusterAnalyser.h.

#### 6.21.2 Constructor & Destructor Documentation

6.21.2.1 `cryomesh::manipulators::IClusterAnalyser::Energy-VariationWeightingMap::EnergyVariationWeightingMap ( )`  
`[inline]`

Definition at line 151 of file IClusterAnalyser.h.

References cryomesh::manipulators::IClusterAnalyser::HIGH\_NEGATIVE, cryomesh::manipulators::IClusterAnalyser::HIGH\_POSITIVE, cryomesh::manipulators::IClusterAnalyser::MEDIUM\_NEGATIVE, cryomesh::manipulators::IClusterAnalyser::MEDIUM\_POSITIVE, cryomesh::manipulators::IClusterAnalyser::OUT\_OF\_RANGE\_NEGATIVE, cryomesh::manipulators::IClusterAnalyser::OUT\_OF\_RANGE\_POSITIVE, cryomesh::manipulators::IClusterAnalyser::SMALL\_NEGATIVE, cryomesh::manipulators::IClusterAnalyser::SMALL\_POSITIVE, cryomesh::manipulators::IClusterAnalyser::STAGNANT\_NEGATIVE, cryomesh::manipulators::IClusterAnalyser::STAGNANT\_POSITIVE, variationMap, and cryomesh::manipulators::IClusterAnalyser::ZERO.

### 6.21.3 Member Data Documentation

#### 6.21.3.1 std::map<EnergyVariation, double> cryomesh::manipulators::IClusterAnalyser::EnergyVariationWeightingMap::variationMap

Definition at line 165 of file IClusterAnalyser.h.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster(), - EnergyVariationWeightingMap(), and cryomesh::manipulators::ClusterAnalyserBasic::getEnergyVariationMap().

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

- /home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/[IClusterAnalyser.h](#)

## 6.22 cryomesh::structures::Fibre Class Reference

A [Fibre](#) is a collection of connections that connect one structure to another.

```
#include <Fibre.h>
```

### Public Types

- enum [ClusterConnectionType](#) { [NullCluster](#) = 1, [InputCluster](#) = 2, [OutputCluster](#) = 4, [LoopbackCluster](#) = 8 }
- *Enum representing the relation of a cluster to this fibre.*
- enum [FibreType](#) { [NullFibre](#), [IntermediateFibre](#), [PrimaryInputFibre](#), [PrimaryOutputFibre](#), [LoopbackFibre](#) }
- *Enum representing the type of this [Fibre](#) connection.*

### Public Member Functions

- [Fibre](#) (boost::shared\_ptr< [Cluster](#) > inputCluster, boost::shared\_ptr< [Cluster](#) > outputCluster, int width)
- *Construct a fibre between two clusters with width.*

- **Fibre** (boost::shared\_ptr< **Cluster** > cluster, const **FibreType** &type, int width)  
*Construct a primary fibre with width.*
- virtual **~Fibre** ()  
*Default destructor.*
- virtual void **update** ()  
*Update all elements.*
- void **trigger** (double percentage)  
*Trigger a random pattern with percentage of the connections to fire.*
- virtual void **trigger** ()  
*Send impulses to the connections based on a complete trigger of all of them.*
- virtual void **trigger** (const **state::Pattern** &pattern)  
*Send impulses to the connections based on a pattern.*
- virtual void **trigger** (std::vector< boost::shared\_ptr< **components::Impulse** > > &triggerImpulses)  
*Send impulses to the connections.*
- const **common::Connector**< **Fibre**, **Cluster** > & **getConnector** () const  
*Get the connector.*
- **common::Connector**< **Fibre**, **Cluster** > & **getMutableConnector** ()  
*Get the mutable connector.*
- const **components::ConnectionMap** & **getConnections** () const  
*Get the connections.*
- **components::ConnectionMap** & **getMutableConnections** ()  
*Get the mutable connections.*
- const **FibreType** & **getType** () const  
*Get the type of the fibre.*
- const std::pair< int, int > **countConnections** (const std::map< boost::uuids::uuid, boost::shared\_ptr< **Cluster** > > &all\_clusters) const  
*Count all connections of this fibre to a group of clusters.*
- int **isConnected** (const boost::shared\_ptr< **Cluster** > &cluster) const  
*Return type of connection to cluster specified, null if none.*
- unsigned int **getWidth** () const  
*Get width (number of connections) of fibre.*
- boost::shared\_ptr< **state::Pattern** > **getNodesPattern** (const std::map< boost::uuids::uuid, boost::shared\_ptr< **components::Node** > > all\_nodes) const  
*Get current pattern for firing state of nodes.*
- boost::shared\_ptr< **state::Pattern** > **getInputNodesPattern** () const  
*Get current pattern for firing state of input nodes to the fibre.*
- boost::shared\_ptr< **state::Pattern** > **getOutputNodesPattern** () const  
*Get current pattern for firing state of output nodes to the fibre.*
- const std::map< boost::uuids::uuid, boost::shared\_ptr< **components::Node** > > **getNodes** (const **ClusterConnectionType** type) const  
*Get the map of nodes to this fibre.*
- const std::map< boost::uuids::uuid, boost::shared\_ptr< **components::Node** > > **getInputNodes** () const

*Get the map of input nodes to this fibre.*

- const std::map < boost::uuids::uuid, boost::shared\_ptr < components::Node > > [getOutputNodes](#) () const

*Get the map of output nodes to this fibre.*

- void [forceFireInputNodes](#) (const state::Pattern &pattern)

*Force fire input nodes using a pattern.*

- void [forceFireOutputNodes](#) (const state::Pattern &pattern)

*Force fire output nodes using a pattern.*

- void [forceFireNodes](#) (const state::Pattern &pattern, std::map< boost::uuids::uuid, boost::shared\_ptr< components::Node > > > nodes)

*Force fire nodes using a pattern.*

- virtual void [enableDebug](#) (bool b)

## Protected Member Functions

- virtual void [setType](#) (const FibreType &tp)

*Set the type of this fibre.*

- virtual void [createConnections](#) (int number)

*Reset and create a number of connections in this Fibre.*

- virtual void [disconnectAllConnections](#) ()

*Disconnect all the connections from clusters.*

- virtual void [connectAllConnections](#) (boost::shared\_ptr< Cluster > cluster, - ClusterConnectionType type)

*Connect all the connections in this Fibre to a cluster of a specified type.*

## Private Attributes

- common::Connector< Fibre, Cluster > connector
- components::ConnectionMap connections
- FibreType fibreType

## Friends

- std::ostream & [operator<<](#) (std::ostream &os, const Fibre &obj)

*Get the activity pattern of the Fibre, 0 for no activity, 1 otherwise.*

### 6.22.1 Detailed Description

A [Fibre](#) is a collection of connections that connect one structure to another.

For example, two clusters.

Definition at line 27 of file Fibre.h.

6.22.2 Member Enumeration Documentation

6.22.2.1 enum cryomesh::structures::Fibre::ClusterConnectionType

Enum representing the relation of a cluster to this fibre.

Enumerator:

- NullCluster*
- InputCluster*
- OutputCluster*
- LoopbackCluster*

Definition at line 33 of file Fibre.h.

6.22.2.2 enum cryomesh::structures::Fibre::FibreType

Enum representing the type of this [Fibre](#) connection.

The type of this [Fibre](#).

Enumerator:

- NullFibre*
- IntermediateFibre*
- PrimaryInputFibre*
- PrimaryOutputFibre*
- LoopbackFibre*

Definition at line 42 of file Fibre.h.

6.22.3 Constructor & Destructor Documentation

6.22.3.1 cryomesh::structures::Fibre::Fibre ( boost::shared\_ptr< Cluster > inputCluster, boost::shared\_ptr< Cluster > outputCluster, int width )

Construct a fibre between two clusters with width.

Parameters

<i>boost::shared_ptr&lt; Cluster &gt;</i>	The input cluster to this <a href="#">Fibre</a>
<i>boost::shared_ptr&lt; Cluster &gt;</i>	The output cluster to this <a href="#">Fibre</a>
<i>int</i>	The width of the fibre connection to create

Definition at line 20 of file Fibre.cpp.

References `connectAllConnections()`, `createConnections()`, `InputCluster`, `IntermediateFibre`, `LoopbackFibre`, `OutputCluster`, and `setType()`.

**6.22.3.2** `cryomesh::structures::Fibre::Fibre ( boost::shared_ptr< Cluster > cluster, const FibreType & type, int width )`

Construct a primary fibre with width.

#### Parameters

<code>boost::shared_ptr&lt; Cluster&gt;</code>	cluster <a href="#">Cluster</a> to connect to fibre
<code>const FibreType &amp; type</code>	Type of fibre connection to make
<code>int width</code>	Width of fibre to create

#### Returns

The new fibre created, possible null

Definition at line 32 of file Fibre.cpp.

References `connectAllConnections()`, `createConnections()`, `getType()`, `InputCluster`, `OutputCluster`, `PrimaryInputFibre`, `PrimaryOutputFibre`, and `setType()`.

**6.22.3.3** `cryomesh::structures::Fibre::~Fibre ( ) [virtual]`

Default destructor.

Definition at line 42 of file Fibre.cpp.

References `disconnectAllConnections()`.

## 6.22.4 Member Function Documentation

**6.22.4.1** `void cryomesh::structures::Fibre::connectAllConnections ( boost::shared_ptr< Cluster > cluster, ClusterConnectionType type ) [protected, virtual]`

Connect all the connections in this [Fibre](#) to a cluster of a specified type.

#### Parameters

<code>boost::shared_ptr&lt; Cluster&gt;</code>	cluster The cluster to connect to
--	-----------------------------------



<i>Cluster- Connection- Type</i>	type The type of cluster we're connecting to
--	--

Definition at line 260 of file Fibre.cpp.

References `cryomesh::common::Connector< U, T >::connectInput()`, `connections`, `connector`, `cryomesh::common::Connector< U, T >::connectOutput()`, `InputCluster`, and `OutputCluster`.

Referenced by `Fibre()`.

**6.22.4.2** `const std::pair< int, int > cryomesh::structures::Fibre::countConnections (`  
`const std::map< boost::uuids::uuid, boost::shared_ptr< Cluster > > & all_clusters`  
`) const`

Count all connections of this fibre to a group of clusters.

#### Parameters

<code>std- ::map&lt;boost- ::uuids- ::uuid,boost- ::shared_- ptr&lt;- Cluster&gt;</code>	<code>&gt; Cluster</code> collection to search for connections to this fibre
--	--

#### Returns

`std::pair<int, int>` Pair of input/output connection count to this fibre within the supplied cluster collection

Definition at line 148 of file Fibre.cpp.

References `getConnector()`, `cryomesh::common::Connector< U, T >::getInputs()`, and `cryomesh::common::Connector< U, T >::getOutputs()`.

**6.22.4.3** `void cryomesh::structures::Fibre::createConnections ( int number )`  
`[protected, virtual]`

Reset and create a number of connections in this [Fibre](#).

#### Parameters

<code>int</code>	number Number of connections to create
------------------	--

Definition at line 230 of file Fibre.cpp.

References connections, and disconnectAllConnections().

Referenced by Fibre().

**6.22.4.4** `void cryomesh::structures::Fibre::disconnectAllConnections ( )`  
`[protected, virtual]`

Disconnect all the connections from clusters.

Definition at line 242 of file Fibre.cpp.

References connections, connector, cryomesh::common::Connector< U, T >::disconnectAllInputs(), and cryomesh::common::Connector< U, T >::disconnectAllOutputs().

Referenced by createConnections(), and ~Fibre().

**6.22.4.5** `void cryomesh::structures::Fibre::enableDebug ( bool b )` `[virtual]`

Definition at line 445 of file Fibre.cpp.

**6.22.4.6** `void cryomesh::structures::Fibre::forceFireInputNodes ( const state::Pattern & pattern )`

Force fire input nodes using a pattern.

#### Parameters

<i>Pattern</i>	The pattern to fire
----------------	---------------------

Definition at line 406 of file Fibre.cpp.

References forceFireNodes(), and getInputNodes().

**6.22.4.7** `void cryomesh::structures::Fibre::forceFireNodes ( const state::Pattern & pattern, std::map< boost::uuids::uuid, boost::shared_ptr< components::Node > > nodes )`

Force fire nodes using a pattern.

#### Parameters

<i>Pattern</i>	The pattern to fire
<i>std::map&lt; boost::uuids::uuid, boost::shared_ptr&lt; components::Node &gt; &gt;</i>	> The nodes to fire the pattern on

Definition at line 414 of file Fibre.cpp.

References `cryomesh::state::Pattern::getPattern()`, and `cryomesh::state::Pattern::getSize()`.

Referenced by `forceFireInputNodes()`, and `forceFireOutputNodes()`.

**6.22.4.8** `void cryomesh::structures::Fibre::forceFireOutputNodes ( const state::Pattern & pattern )`

Force fire output nodes using a pattern.

Parameters

<i>Pattern</i>	The pattern to fire
----------------	---------------------

Definition at line 410 of file Fibre.cpp.

References `forceFireNodes()`, and `getOutputNodes()`.

**6.22.4.9** `const components::ConnectionMap & cryomesh::structures::Fibre::getConnections ( ) const`

Get the connections.

Returns

[components::ConnectionMap](#) The connection map for this [Fibre](#)

Definition at line 128 of file Fibre.cpp.

References `connections`.

Referenced by `getWidth()`, `cryomesh::structures::operator<<()`, and `trigger()`.

**6.22.4.10** `const common::Connector< Fibre, Cluster > & cryomesh::structures::Fibre::getConnector ( ) const`

Get the connector.

Returns

`common::Connector<Fibre, Cluster> &` The connector object

Definition at line 121 of file Fibre.cpp.

References `connector`.

Referenced by `countConnections()`, and `isConnected()`.

**6.22.4.11** `const std::map< boost::uuids::uuid, boost::shared_ptr< components::Node > > cryomesh::structures::Fibre::getInputNodes ( ) const`

Get the map of input nodes to this fibre.

#### Returns

`std::map<boost::uuid, boost::shared_ptr< components::Node > >` The map of input nodes

Definition at line 340 of file `Fibre.cpp`.

References `getNodes()`, and `InputCluster`.

Referenced by `forceFireInputNodes()`, and `getInputNodesPattern()`.

**6.22.4.12** `boost::shared_ptr< state::Pattern > cryomesh::structures::Fibre::getInputNodesPattern ( ) const`

Get current pattern for firing state of input nodes to the fibre.

#### Returns

`boost::shared_ptr< state::Pattern >` The current firing pattern of the input nodes to the fibre

Definition at line 331 of file `Fibre.cpp`.

References `getInputNodes()`, and `getNodesPattern()`.

**6.22.4.13** `components::ConnectionMap & cryomesh::structures::Fibre::getMutableConnections ( )`

Get the mutable connections.

#### Returns

[components::ConnectionMap](#) The mutable connection map for this [Fibre](#)

Definition at line 132 of file `Fibre.cpp`.

References `connections`.

**6.22.4.14** `common::Connector< Fibre, Cluster > & cryomesh::structures::Fibre::getMutableConnector ( )`

Get the mutable connector.

**Returns**

common::Connector<Fibre, Cluster> The connector object

Definition at line 125 of file Fibre.cpp.

References connector.

**6.22.4.15** `const std::map< boost::uuids::uuid, boost::shared_ptr< components::Node > > cryomesh::structures::Fibre::getNodes ( const ClusterConnectionType type ) const`

Get the map of nodes to this fibre.

**Parameters**

<i>Cluster-Connection-Type</i>	The cluster to get the nodes from, eg, InputCluster means get the input nodes
--------------------------------	---

**Returns**

std::map<boost::uuid, boost::shared\_ptr< components::Node > > The map of nodes

Definition at line 348 of file Fibre.cpp.

References connections, InputCluster, and OutputCluster.

Referenced by getInputNodes(), and getOutputNodes().

**6.22.4.16** `boost::shared_ptr< state::Pattern > cryomesh::structures::Fibre::getNodePattern ( const std::map< boost::uuids::uuid, boost::shared_ptr< components::Node > > all_nodes ) const`

Get current pattern for firing state of nodes.

**Parameters**

<i>const</i>	std::map<boost::uuids::uuid, boost::shared_ptr<components::Node> > The nodes to check for firing pattern
--------------	--

**Returns**

boost::shared\_ptr< state::Pattern > The current firing pattern of the input nodes to the fibre

Definition at line 302 of file Fibre.cpp.

References cryomesh::components::Node::Positive.

Referenced by getInputNodesPattern(), and getOutputNodesPattern().

**6.22.4.17** `const std::map< boost::uuids::uuid, boost::shared_ptr< components::Node > > cryomesh::structures::Fibre::getOutputNodes ( ) const`

Get the map of output nodes to this fibre.

#### Returns

`std::map<boost::uuid, boost::shared_ptr< components::Node > >` The map of output nodes

Definition at line 344 of file Fibre.cpp.

References `getNodes()`, and `OutputCluster`.

Referenced by `forceFireOutputNodes()`, and `getOutputNodesPattern()`.

**6.22.4.18** `boost::shared_ptr< state::Pattern > cryomesh::structures::Fibre::getOutputNodesPattern ( ) const`

Get current pattern for firing state of output nodes to the fibre.

#### Returns

`boost::shared_ptr< state::Pattern >` The current firing pattern of the output nodes to the fibre

Definition at line 335 of file Fibre.cpp.

References `getNodesPattern()`, and `getOutputNodes()`.

Referenced by `cryomesh::structures::operator<<()`.

**6.22.4.19** `const Fibre::FibreType & cryomesh::structures::Fibre::getType ( ) const`

Get the type of the fibre.

#### Returns

`FibreType` The type of the fibre connection

Definition at line 136 of file Fibre.cpp.

References `fibreType`.

Referenced by `Fibre()`.

**6.22.4.20** `unsigned int cryomesh::structures::Fibre::getWidth ( ) const`

Get width (number of connections) of fibre.

**Returns**

unsigned int Width of fibre

Definition at line 144 of file Fibre.cpp.

References `getConnections()`.

Referenced by `trigger()`.

**6.22.4.21** `int cryomesh::structures::Fibre::isConnected ( const boost::shared_ptr< Cluster > & cluster ) const`

Return type of connection to cluster specified, null if none.

**Parameters**

<i>boost- ::shared_ _ptr&lt; Cluster&gt;</i>	cluster Check connection to this cluster
--	--

**Returns**

const ClusterConnectionType & Connection type to cluster, Null if none

Definition at line 189 of file Fibre.cpp.

References `getConnector()`, `cryomesh::common::Connector< U, T >::getInputs()`, `cryomesh::common::Connector< U, T >::getOutputs()`, `InputCluster`, `LoopbackCluster`, and `OutputCluster`.

**6.22.4.22** `void cryomesh::structures::Fibre::setType ( const FibreType & tp )`  
[protected, virtual]

Set the type of this fibre.

**Parameters**

<i>const</i>	FibreType & tp The type of this fibre
--------------	---------------------------------------

Definition at line 140 of file Fibre.cpp.

References `fibreType`.

Referenced by `Fibre()`.

**6.22.4.23** `void cryomesh::structures::Fibre::trigger ( double percentage )`

Trigger a random pattern with percentage of the connections to fire.

## Parameters

<i>double</i>	Fraction of connection to trigger randomly
---------------	--

Definition at line 63 of file Fibre.cpp.

References `cryomesh::state::Pattern::getRandom()`, `getWidth()`, and `trigger()`.

#### 6.22.4.24 `void cryomesh::structures::Fibre::trigger ( ) [virtual]`

Send impulses to the connections based on a complete trigger of all of them.

Definition at line 54 of file Fibre.cpp.

References `getConnections()`, and `cryomesh::components::Impulse::getTriggerImpulse()`.

Referenced by `trigger()`.

#### 6.22.4.25 `void cryomesh::structures::Fibre::trigger ( const state::Pattern & pattern ) [virtual]`

Send impulses to the connections based on a pattern.

## Parameters

<a href="#"><i>state::Pattern</i></a>	& pattern The pattern to use to create impulses and send to connections
---------------------------------------	---

Definition at line 69 of file Fibre.cpp.

References `getConnections()`, `cryomesh::state::Pattern::getPattern()`, `cryomesh::components::Impulse::getTriggerImpulse()`, and `trigger()`.

#### 6.22.4.26 `void cryomesh::structures::Fibre::trigger ( std::vector< boost::shared_ptr< components::Impulse > > & triggerImpulses ) [virtual]`

Send impulses to the connections.

## Parameters

<i>const</i>	<code>std::vector&lt;boost::shared_ptr&lt; components::Impulse &gt; &gt; &amp; triggerImpulses</code> The impulses to send to connections
--------------	---

Definition at line 90 of file Fibre.cpp.

References `connections`.

#### 6.22.4.27 `void cryomesh::structures::Fibre::update ( ) [virtual]`

Update all elements.



Definition at line 46 of file Fibre.cpp.

References connections, and cryomesh::components::ConnectionMap::update().

### 6.22.5 Friends And Related Function Documentation

**6.22.5.1** `std::ostream& operator<< ( std::ostream & os, const Fibre & obj )` `[friend]`

Get the activity pattern of the [Fibre](#), 0 for no activity, 1 otherwise.

#### Returns

Pattern To stream operator

#### Parameters

<code>std::ostream</code>	& os The output stream
<code>const</code>	<a href="#">Fibre</a> & obj The object to stream

#### Returns

std::ostream & The output stream

Definition at line 449 of file Fibre.cpp.

### 6.22.6 Member Data Documentation

**6.22.6.1** `components::ConnectionMap cryomesh::structures::Fibre::connections`  
`[private]`

Definition at line 334 of file Fibre.h.

Referenced by connectAllConnections(), createConnections(), disconnectAllConnections(), getConnections(), getMutableConnections(), getNodes(), trigger(), and update().

**6.22.6.2** `common::Connector<Fibre, Cluster> cryomesh::structures::Fibre-  
::connector` `[private]`

Definition at line 327 of file Fibre.h.

Referenced by connectAllConnections(), disconnectAllConnections(), getConnector(), and getMutableConnector().

**6.22.6.3** `FibreType cryomesh::structures::Fibre::fibreType` `[private]`

Definition at line 341 of file Fibre.h.

Referenced by `getType()`, and `setType()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Fibre.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Fibre.cpp`

## 6.23 cryomesh::structures::FibreMap Class Reference

```
#include <FibreMap.h>
```

### Public Member Functions

- `FibreMap()`
- `virtual ~FibreMap()`
- `virtual void update()`

### Friends

- `std::ostream & operator<< (std::ostream &os, const FibreMap &objs)`

#### 6.23.1 Detailed Description

Definition at line 18 of file `FibreMap.h`.

#### 6.23.2 Constructor & Destructor Documentation

**6.23.2.1** `cryomesh::structures::FibreMap::FibreMap( )` [`inline`]

Definition at line 20 of file `FibreMap.h`.

**6.23.2.2** `virtual cryomesh::structures::FibreMap::~FibreMap( )` [`inline`,  
`virtual`]

Definition at line 22 of file `FibreMap.h`.

#### 6.23.3 Member Function Documentation

**6.23.3.1** `virtual void cryomesh::structures::FibreMap::update( )` [`inline`,  
`virtual`]

Definition at line 24 of file `FibreMap.h`.

Referenced by cryomesh::structures::Bundle::update(), cryomesh::structures::Bundle::updatePrimaryInputFibres(), and cryomesh::structures::Bundle::updatePrimaryOutputFibres().

### 6.23.4 Friends And Related Function Documentation

6.23.4.1 `std::ostream& operator<< ( std::ostream & os, const FibreMap & objs )`  
[friend]

Definition at line 40 of file FibreMap.h.

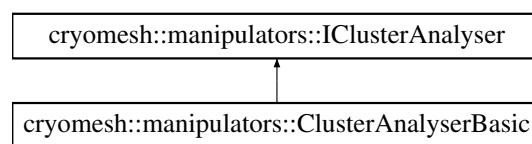
The documentation for this class was generated from the following file:

- /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/[FibreMap.h](#)

## 6.24 cryomesh::manipulators::IClusterAnalyser Class Reference

```
#include <IClusterAnalyser.h>
```

Inheritance diagram for cryomesh::manipulators::IClusterAnalyser:



### Classes

- struct [EnergyVariationWeightingMap](#)
- struct [RestructuringCountdown](#)

*Class to hold together information on whether we can act to restructure items.*

### Public Types

- enum [EnergyVariation](#) { [NONE](#) = 2048, [OUT\\_OF\\_RANGE\\_POSITIVE](#) = 1024, [HIGH\\_POSITIVE](#) = 512, [MEDIUM\\_POSITIVE](#) = 256, [SMALL\\_POSITIVE](#) = 128, [STAGNANT\\_POSITIVE](#) = 64, [ZERO](#) = 32, [STAGNANT\\_NEGATIVE](#) = 16, [SMALL\\_NEGATIVE](#) = 8, [MEDIUM\\_NEGATIVE](#) = 4, [HIGH\\_NEGATIVE](#) = 2, [OUT\\_OF\\_RANGE\\_NEGATIVE](#) = 1 }

### Public Member Functions

- [IClusterAnalyser](#) ()

- virtual `~IClusterAnalyser ()`
- virtual `ClusterAnalysisData analyseCluster (const structures::Cluster &cluster, const std::map< int, std::list< ClusterAnalysisData > > &histories)=0`  
*Run an analysis on the cluster to decide what action to take on nodes and connections.*
- virtual `ClusterAnalysisData calculateRangeEnergies (const std::list< ClusterAnalysisData > &history) const =0`  
*Calculate the range energies stats.*
- virtual `EnergyVariationWeightingMap getEnergyVariationMap (const double energy_input, double range) const =0`
- const `RestructuringCountdown & getConnectionRestructuring () const`
- const `RestructuringCountdown & getNodeRestructuring () const`

### Protected Attributes

- `RestructuringCountdown nodeRestructuring`
- `RestructuringCountdown connectionRestructuring`

### 6.24.1 Detailed Description

Definition at line 17 of file IClusterAnalyser.h.

### 6.24.2 Member Enumeration Documentation

#### 6.24.2.1 enum cryomesh::manipulators::IClusterAnalyser::EnergyVariation

Enumerator:

**NONE**  
**OUT\_OF\_RANGE\_POSITIVE**  
**HIGH\_POSITIVE**  
**MEDIUM\_POSITIVE**  
**SMALL\_POSITIVE**  
**STAGNANT\_POSITIVE**  
**ZERO**  
**STAGNANT\_NEGATIVE**  
**SMALL\_NEGATIVE**  
**MEDIUM\_NEGATIVE**  
**HIGH\_NEGATIVE**  
**OUT\_OF\_RANGE\_NEGATIVE**

Definition at line 136 of file IClusterAnalyser.h.

### 6.24.3 Constructor & Destructor Documentation

6.24.3.1 `cryomesh::manipulators::IClusterAnalyser::IClusterAnalyser ( )`  
`[inline]`

Definition at line 168 of file IClusterAnalyser.h.

6.24.3.2 `virtual cryomesh::manipulators::IClusterAnalyser::~~IClusterAnalyser ( )`  
`[inline, virtual]`

Definition at line 171 of file IClusterAnalyser.h.

### 6.24.4 Member Function Documentation

6.24.4.1 `virtual ClusterAnalysisData cryomesh::manipulators::IClusterAnalyser-  
::analyseCluster ( const structures::Cluster & cluster, const std::map< int,  
std::list< ClusterAnalysisData > > & histories ) [pure virtual]`

Run an analysis on the cluster to decide what action to take on nodes and connections.

#### Parameters

<i>const</i>	<a href="#">structures::Cluster</a> & The cluster to analyse
<i>const</i>	<code>std::list&lt;ClusterAnalysisData&gt;</code> The historical analysis data to use

#### Returns

[ClusterAnalysisData](#) The resulting analytical data

Implemented in [cryomesh::manipulators::ClusterAnalyserBasic](#).

6.24.4.2 `virtual ClusterAnalysisData cryomesh::manipulators::IClusterAnalyser-  
::calculateRangeEnergies ( const std::list< ClusterAnalysisData > & history  
) const [pure virtual]`

Calculate the range energies stats.

#### Parameters

<i>const</i>	<code>std::list&lt;ClusterAnalysisData&gt;</code> & The history range to work with
--------------	--

#### Returns

[ClusterAnalysisData](#) The resulting cluster analysis data

Implemented in [cryomesh::manipulators::ClusterAnalyserBasic](#).

**6.24.4.3** `const RestructuringCountdown& cryomesh::manipulators::IClusterAnalyser::getConnectionRestructuring ( ) const`  
`[inline]`

Definition at line 198 of file IClusterAnalyser.h.

References `connectionRestructuring`.

**6.24.4.4** `virtual EnergyVariationWeightingMap cryomesh::manipulators::IClusterAnalyser::getEnergyVariationMap ( const double energy_input, double range ) const`  
`[pure virtual]`

Implemented in [cryomesh::manipulators::ClusterAnalyserBasic](#).

**6.24.4.5** `const RestructuringCountdown& cryomesh::manipulators::IClusterAnalyser::getNodeRestructuring ( ) const`  
`[inline]`

Definition at line 201 of file IClusterAnalyser.h.

References `nodeRestructuring`.

## 6.24.5 Member Data Documentation

**6.24.5.1** `RestructuringCountdown cryomesh::manipulators::IClusterAnalyser::connectionRestructuring` `[protected]`

Definition at line 206 of file IClusterAnalyser.h.

Referenced by `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`, and `getConnectionRestructuring()`.

**6.24.5.2** `RestructuringCountdown cryomesh::manipulators::IClusterAnalyser::nodeRestructuring` `[protected]`

Definition at line 205 of file IClusterAnalyser.h.

Referenced by `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`, and `getNodeRestructuring()`.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/IClusterAnalyser.h](#)

## 6.25 cryomesh::components::Impulse Class Reference

[Impulse](#) is a mobile information packet to be passed between Nodes.

```
#include <Impulse.h>
```

## Public Member Functions

- [Impulse](#) ()  
*Constructor.*
- [Impulse](#) (const double max\_y, const int length, const int delay=0)  
*Construct a from a curve with max  $f(x)$  and length and set starting cycle to startCycle, which defaults to the present, 'now' cycle.*
- [Impulse](#) (const double max\_y, const int length, const int delay, boost::shared\_ptr< [ActivityTimerDistance](#) >)  
*Construct a from a curve with max  $f(x)$  and length and set starting cycle to startCycle, and an activity timer.*
- [Impulse](#) (const [Impulse](#) &obj)
- virtual [~Impulse](#) ()  
*Destructor.*
- void [randomise](#) (double positive\_bias=0.5)
- bool [isActive](#) () const  
*Is the [Impulse](#) active on current cycle.*
- bool [isActive](#) (const [common::Cycle](#) &cycle) const  
*Is the [Impulse](#) active on cycle.*
- bool [isActive](#) (const [common::Cycle](#) &startCycle, const [common::Cycle](#) &endCycle) const  
*Is the [Impulse](#) active at some point in cycle range.*
- double [getActivity](#) ([common::Cycle](#) cycle) const  
*Get activity at cycle.*
- double [getActivity](#) () const  
*Get activity at current cycle.*
- double [getActivityMaximum](#) () const  
*Get maximum activity.*
- double [getActivityMinimum](#) () const  
*Get minimum activity.*
- virtual [Impulse](#) & [invert](#) ()  
*Invert the impulse.*
- [common::Cycle](#) [getFirstActiveCycle](#) () const  
*Get the first active cycle.*
- void [setFirstActiveCycle](#) (const [common::Cycle](#) cycle)  
*Set the first active cycle.*
- [common::Cycle](#) [getLastActiveCycle](#) () const  
*Get the last active cycle.*
- const std::list< double > & [getActivities](#) () const  
*Get activities.*
- const boost::shared\_ptr< [ActivityTimerDistance](#) > [getActivityTimer](#) () const  
*Get activity timer.*

- `boost::shared_ptr< ActivityTimerDistance > getMutableActivityTimer ()`  
*Get mutable activity timer.*
- `void setActivityTimer (boost::shared_ptr< ActivityTimerDistance > timer)`  
*Set activity timer.*
- `int getActivityDelay () const`
- `void setActivityDelay (int delay)`
- `const Impulse operator+ (const Impulse &obj) const`  
*Non-destructive addition operator.*
- `Impulse & operator+= (const Impulse &obj)`  
*Destructive addition and assignment operator.*
- `Impulse & operator= (const Impulse &obj)`  
*Assignment operator.*
- `bool operator== (const Impulse &obj) const`  
*Comparator operator.*
- `bool operator!= (const Impulse &obj) const`  
*Not comparator operator.*
- `virtual void enableDebug (bool b)`

### Static Public Member Functions

- `static boost::shared_ptr< Impulse > getTriggerImpulse ()`  
*Get a 'trigger' impulse, a maximum impulse.*
- `static boost::shared_ptr< Impulse > getRandom (double positive_bias=0.5)`  
*Get a randomised impulse.*

### Static Public Attributes

- `static const double FORCED\_TRIGGER\_ACTIVITY = 1000`
- `static const double MAX\_ACTIVITY = 1`
- `static const double MIN\_ACTIVITY = -1`
- `static const int MAX\_ACTIVITY\_LENGTH = 20`
- `static const int MIN\_ACTIVITY\_LENGTH = 1`
- `static const double MIN\_ACTIVITY\_MAGNITUDE = 0.01`
- `static const int MIN\_ACTIVITY\_DELAY = 0`
- `static const int MAX\_ACTIVITY\_DELAY = 10`

### Protected Member Functions

- `double getActivityBoundary (bool maximal) const`  
*Get the boundary value of activity.*



## Private Attributes

- `common::Cycle` `firstActiveCycle`  
The first cycle that this *Impulse* has activity.
- `common::Cycle` `lastActiveCycle`  
The lase cycle that this *Impulse* has activity.
- `int` `activityDelay`
- `boost::shared_ptr` `< ActivityTimerDistance > activityTimer`

## Friends

- `std::ostream` & `operator<<` (`std::ostream` &`os`, `const Impulse` &`obj`)  
To stream operator.

### 6.25.1 Detailed Description

*Impulse* is a mobile information packet to be passed between Nodes.

Impulses represent information generated by a *Node* firing They are propagated along a connection Can be modified by the overlying Mesh as they propagate

Definition at line 31 of file *Impulse.h*.

### 6.25.2 Constructor & Destructor Documentation

#### 6.25.2.1 cryomesh::components::Impulse::Impulse ( )

Constructor.

Constructor for *Impulse*

#### Parameters

<i>bool</i>	random If true then randomise the impulse on creation
-------------	---

Definition at line 45 of file *Impulse.cpp*.

References `MIN_ACTIVITY_LENGTH`, `setActivityTimer()`, and `setFirstActiveCycle()`.

#### 6.25.2.2 cryomesh::components::Impulse::Impulse ( `const double max_y`, `const int length`, `const int delay = 0` )

Construct a from a curve with max  $f(x)$  and length and set starting cycle to `startCycle`, which defaults to the present, 'now' cycle.

## Parameters

<i>const</i>	int max_y Boundary value of curve
<i>const</i>	int length Length of <a href="#">Impulse</a>
<i>int</i>	Delay before starting impulse

Definition at line 54 of file `Impulse.cpp`.

References `MIN_ACTIVITY_LENGTH`, `setActivityDelay()`, `setActivityTimer()`, and `setFirstActiveCycle()`.

#### 6.25.2.3 `cryomesh::components::Impulse::Impulse ( const double max_y, const int length, const int delay, boost::shared_ptr< ActivityTimerDistance > timer )`

Construct a from a curve with max  $f(x)$  and length and set starting cycle to `startCycle`, and an activity timer.

## Parameters

<i>const</i>	int max_y Boundary value of curve
<i>const</i>	int length Length of <a href="#">Impulse</a>
<i>int</i>	Delay before starting impulse
<i>boost- ::shared_ _ptr&lt;Activity- Timer&gt;</i>	timer The activity timer associated with this

Definition at line 65 of file `Impulse.cpp`.

References `MIN_ACTIVITY_LENGTH`, `setActivityDelay()`, `setActivityTimer()`, and `setFirstActiveCycle()`.

#### 6.25.2.4 `cryomesh::components::Impulse::Impulse ( const Impulse & obj )`

Definition at line 75 of file `Impulse.cpp`.

References `firstActiveCycle`, `getActivityDelay()`, `getActivityTimer()`, `getFirstActiveCycle()`, `getLastActiveCycle()`, `lastActiveCycle`, `setActivityDelay()`, and `setActivityTimer()`.

#### 6.25.2.5 `cryomesh::components::Impulse::~Impulse ( ) [virtual]`

Destructor.

Destructor for [Impulse](#)

Definition at line 83 of file `Impulse.cpp`.

### 6.25.3 Member Function Documentation

**6.25.3.1** void cryomesh::components::Impulse::enableDebug ( bool *b* )  
[virtual]

Definition at line 336 of file Impulse.cpp.

**6.25.3.2** const std::list< double > & cryomesh::components::Impulse::getActivities ( ) const

Get activities.

#### Returns

const std::list<double> & The activities list

Definition at line 192 of file Impulse.cpp.

**6.25.3.3** double cryomesh::components::Impulse::getActivity ( common::Cycle *cycle* ) const

Get activity at cycle.

Sum all the Impulses in the collection on specified cycle and return activity

#### Parameters

<i>int</i>	<i>cycle</i> The cycle to calculate the activity on
------------	---

#### Returns

double The activity on specified cycle

Definition at line 129 of file Impulse.cpp.

References firstActiveCycle, getFirstActiveCycle(), getLastActiveCycle(), lastActiveCycle, and cryomesh::common::Cycle::toInt().

**6.25.3.4** double cryomesh::components::Impulse::getActivity ( ) const

Get activity at current cycle.

Sum all the Impulses in the collection on the current cycle and return activity

#### Returns

double The activity on specified cycle

Definition at line 125 of file Impulse.cpp.

References cryomesh::common::TimeKeeper::getTimeKeeper().

**6.25.3.5** `double cryomesh::components::Impulse::getActivityBoundary ( bool maximal ) const` [protected]

Get the boundary value of activity.

#### Parameters

<i>bool</i>	maximal True if maximal boundary, false if minimal
-------------	--

#### Returns

double The boundary value of activity

**6.25.3.6** `int cryomesh::components::Impulse::getActivityDelay ( ) const`

Definition at line 208 of file Impulse.cpp.

References `activityDelay`.

Referenced by `Impulse()`, `cryomesh::components::operator<<()`, `operator=()`, and `setFirstActiveCycle()`.

**6.25.3.7** `double cryomesh::components::Impulse::getActivityMaximum ( ) const`

Get maximum activity.

Find the maximum activity between start and end cycles

#### Returns

double The maximum activity

Definition at line 156 of file Impulse.cpp.

**6.25.3.8** `double cryomesh::components::Impulse::getActivityMinimum ( ) const`

Get minimum activity.

Find the minimum activity between start and end cycles

#### Returns

double The minimum activity

Definition at line 160 of file Impulse.cpp.

**6.25.3.9** `const boost::shared_ptr< ActivityTimerDistance > cryomesh::components::Impulse::getActivityTimer ( ) const`

Get activity timer.

**Returns**

boost::shared\_ptr< ActivityTimer > activityTimer; The activity timer

Definition at line 196 of file Impulse.cpp.

References activityTimer.

Referenced by Impulse(), cryomesh::components::operator<<(), and operator=().

**6.25.3.10 Cycle cryomesh::components::Impulse::getFirstActiveCycle ( ) const**

Get the first active cycle.

**Returns**

Cycle The first active cycle

Definition at line 169 of file Impulse.cpp.

References firstActiveCycle.

Referenced by cryomesh::components::ImpulseCollection::clearActiveImpulses(), getActivity(), Impulse(), operator+=(), cryomesh::components::operator<<(), operator=(), and operator==().

**6.25.3.11 Cycle cryomesh::components::Impulse::getLastActiveCycle ( ) const**

Get the last active cycle.

**Returns**

Cycle The last active cycle

Definition at line 188 of file Impulse.cpp.

References lastActiveCycle.

Referenced by cryomesh::components::ImpulseCollection::clearActiveImpulses(), getActivity(), Impulse(), operator+=(), cryomesh::components::operator<<(), operator=(), and operator==().

**6.25.3.12 boost::shared\_ptr< ActivityTimerDistance > cryomesh::components::Impulse::getMutableActivityTimer ( )**

Get mutable activity timer.

**Returns**

boost::shared\_ptr< ActivityTimer > activityTimer; The activity timer

Definition at line 200 of file Impulse.cpp.

References activityTimer.

**6.25.3.13** `boost::shared_ptr< Impulse > cryomesh::components::Impulse::getRandom ( double positive_bias = 0.5 )`  
`[static]`

Get a randomised impulse.

#### Parameters

<i>double</i>	the (0,1) bias of a positive impulse outcome, 0 negative, 1, positive
---------------	---

#### Returns

`boost::shared_ptr<Impulse>` The randomised impulse

Definition at line 38 of file `Impulse.cpp`.

Referenced by `cryomesh::components::NodeMap::addRandomImpulses()`, and `randomise()`.

**6.25.3.14** `boost::shared_ptr< Impulse > cryomesh::components::Impulse::getTriggerImpulse ( )` `[static]`

Get a 'trigger' impulse, a maximum impulse.

#### Returns

`boost::shared_ptr<Impulse>` The trigger impulse

Definition at line 33 of file `Impulse.cpp`.

Referenced by `cryomesh::components::Node::forceFire()`, and `cryomesh::structures::Fibre::trigger()`.

**6.25.3.15** `Impulse & cryomesh::components::Impulse::invert ( )` `[virtual]`

Invert the impulse.

@ return `Impulse` & This object inverted

Definition at line 164 of file `Impulse.cpp`.

**6.25.3.16** `bool cryomesh::components::Impulse::isActive ( ) const`

Is the `Impulse` active on current cycle.

**Returns**

bool True if active, false otherwise

Definition at line 108 of file Impulse.cpp.

References cryomesh::common::TimeKeeper::getTimeKeeper().

Referenced by cryomesh::components::ImpulseCollection::clearActiveImpulses(), and isActive().

**6.25.3.17** bool cryomesh::components::Impulse::isActive ( const common::Cycle & cycle ) const

Is the [Impulse](#) active on cycle.

**Returns**

bool True if active, false otherwise

Definition at line 112 of file Impulse.cpp.

References isActive().

**6.25.3.18** bool cryomesh::components::Impulse::isActive ( const common::Cycle & startCycle, const common::Cycle & endCycle ) const

Is the [Impulse](#) active at some point in cycle range.

**Returns**

bool True if active, false otherwise

Definition at line 116 of file Impulse.cpp.

References firstActiveCycle, and lastActiveCycle.

**6.25.3.19** bool cryomesh::components::Impulse::operator!= ( const Impulse & obj ) const

Not comparator operator.

**Parameters**

const	<a href="#">Impulse</a> & obj RHS object
-------	--

**Returns**

bool True if not equal, false otherwise

Definition at line 333 of file Impulse.cpp.

6.25.3.20 `const Impulse cryomesh::components::Impulse::operator+ ( const Impulse & obj )`  
`const`

Non-destructive addition operator.

#### Parameters

<code>const</code>	<a href="#">Impulse</a> & obj RHS addition
--------------------	--

#### Returns

[Impulse](#) New object after addition

Definition at line 215 of file `Impulse.cpp`.

6.25.3.21 `Impulse & cryomesh::components::Impulse::operator+= ( const Impulse & obj )`

Destructive addition and assignment operator.

#### Parameters

<code>const</code>	<a href="#">Impulse</a> & obj RHS addition
--------------------	--

#### Returns

[Impulse](#) & This object after addition and assignment

Definition at line 221 of file `Impulse.cpp`.

References `getFirstActiveCycle()`, `getLastActiveCycle()`, and `setFirstActiveCycle()`.

6.25.3.22 `Impulse & cryomesh::components::Impulse::operator= ( const Impulse & obj )`

Assignment operator.

#### Parameters

<code>const</code>	<a href="#">Impulse</a> & obj RHS assignment
--------------------	--

#### Returns

[Impulse](#) & This object after assignment

Definition at line 275 of file `Impulse.cpp`.

References `firstActiveCycle`, `getActivityDelay()`, `getActivityTimer()`, `getFirstActiveCycle()`, `getLastActiveCycle()`, `lastActiveCycle`, `setActivityDelay()`, `setActivityTimer()`, and `cryomesh::common::Cycle::toInt()`.



6.25.3.23 `bool cryomesh::components::Impulse::operator==( const Impulse & obj ) const`

Comparator operator.

#### Parameters

<i>const</i>	<a href="#">Impulse</a> & obj RHS object
--------------	--

#### Returns

bool True if equal, false otherwise

Definition at line 296 of file Impulse.cpp.

References `getFirstActiveCycle()`, and `getLastActiveCycle()`.

6.25.3.24 `void cryomesh::components::Impulse::randomise ( double positive_bias = 0.5 )`

Definition at line 86 of file Impulse.cpp.

References `firstActiveCycle`, `getRandom()`, `lastActiveCycle`, `MAX_ACTIVITY`, `MAX_ACTIVITY_DELAY`, `MAX_ACTIVITY_LENGTH`, `MIN_ACTIVITY`, `MIN_ACTIVITY_DELAY`, `MIN_ACTIVITY_LENGTH`, `MIN_ACTIVITY_MAGNITUDE`, `setActivityDelay()`, and `setActivityTimer()`.

6.25.3.25 `void cryomesh::components::Impulse::setActivityDelay ( int delay )`

Definition at line 212 of file Impulse.cpp.

References `activityDelay`.

Referenced by `Impulse()`, `operator=()`, and `randomise()`.

6.25.3.26 `void cryomesh::components::Impulse::setActivityTimer ( boost::shared_ptr< ActivityTimerDistance > timer )`

Set activity timer.

#### Parameters

<i>boost- ::shared_ ptr&lt;Activity- Timer&gt;</i>	The activity timer to set
--	---------------------------

Definition at line 204 of file Impulse.cpp.

References `activityTimer`.

Referenced by `Impulse()`, `operator=()`, and `randomise()`.

**6.25.3.27** `void cryomesh::components::Impulse::setFirstActiveCycle ( const common::Cycle cycle )`

Set the first active cycle.

#### Parameters

<i>const</i>	Cycle cycle The first active cycle
--------------	------------------------------------

Definition at line 173 of file Impulse.cpp.

References firstActiveCycle, getActivityDelay(), and lastActiveCycle.

Referenced by Impulse(), and operator+=().

### 6.25.4 Friends And Related Function Documentation

**6.25.4.1** `std::ostream& operator<< ( std::ostream & os, const Impulse & obj )`  
[friend]

To stream operator.

#### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Impulse</a> & obj The object to stream

#### Returns

std::ostream & The output stream

Definition at line 339 of file Impulse.cpp.

### 6.25.5 Member Data Documentation

**6.25.5.1** `int cryomesh::components::Impulse::activityDelay` [private]

Definition at line 411 of file Impulse.h.

Referenced by getActivityDelay(), and setActivityDelay().

**6.25.5.2** `boost::shared_ptr<ActivityTimerDistance> cryomesh::components::Impulse::activityTimer` [private]

Definition at line 418 of file Impulse.h.

Referenced by getActivityTimer(), getMutableActivityTimer(), and setActivityTimer().

### 6.25.5.3 `common::Cycle cryomesh::components::Impulse::firstActiveCycle` [private]

The first cycle that this [Impulse](#) has activity.

#### Returns

Cycle Return first active cycle

Definition at line 396 of file `Impulse.h`.

Referenced by `getActivity()`, `getFirstActiveCycle()`, `Impulse()`, `isActive()`, `operator=()`, `randomise()`, and `setFirstActiveCycle()`.

### 6.25.5.4 `const double cryomesh::components::Impulse::FORCED_TRIGGER_ACTIVITY = 1000` [static]

Definition at line 300 of file `Impulse.h`.

### 6.25.5.5 `common::Cycle cryomesh::components::Impulse::lastActiveCycle` [private]

The last cycle that this [Impulse](#) has activity.

#### Returns

Cycle Return last active cycle

Definition at line 404 of file `Impulse.h`.

Referenced by `getActivity()`, `getLastActiveCycle()`, `Impulse()`, `isActive()`, `operator=()`, `randomise()`, and `setFirstActiveCycle()`.

### 6.25.5.6 `const double cryomesh::components::Impulse::MAX_ACTIVITY = 1` [static]

Definition at line 307 of file `Impulse.h`.

Referenced by `randomise()`.

### 6.25.5.7 `const int cryomesh::components::Impulse::MAX_ACTIVITY_DELAY = 10` [static]

Definition at line 355 of file `Impulse.h`.

Referenced by `randomise()`.

**6.25.5.8** `const int cryomesh::components::Impulse::MAX_ACTIVITY_LENGTH = 20`  
[static]

Definition at line 323 of file Impulse.h.

Referenced by `randomise()`.

**6.25.5.9** `const double cryomesh::components::Impulse::MIN_ACTIVITY = -1`  
[static]

Definition at line 315 of file Impulse.h.

Referenced by `randomise()`.

**6.25.5.10** `const int cryomesh::components::Impulse::MIN_ACTIVITY_DELAY = 0`  
[static]

Definition at line 347 of file Impulse.h.

Referenced by `randomise()`.

**6.25.5.11** `const int cryomesh::components::Impulse::MIN_ACTIVITY_LENGTH = 1`  
[static]

Definition at line 331 of file Impulse.h.

Referenced by `Impulse()`, and `randomise()`.

**6.25.5.12** `const double cryomesh::components::Impulse::MIN_ACTIVITY_MAGNITUDE = 0.01` [static]

Definition at line 339 of file Impulse.h.

Referenced by `randomise()`.

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

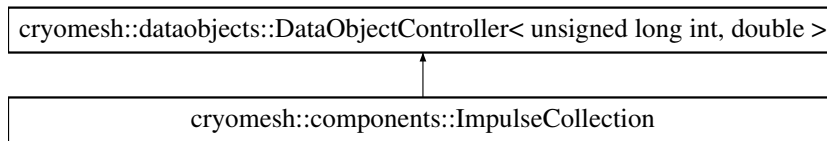
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Impulse.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Impulse.cpp](#)

## 6.26 cryomesh::components::ImpulseCollection Class Reference

[ImpulseCollection](#) represents a collection of [Impulse](#) objects.

```
#include <ImpulseCollection.h>
```

Inheritance diagram for `cryomesh::components::ImpulseCollection`:



## Public Types

- enum [Comparison](#) { [GreaterThan](#), [LessThan](#), [EqualTo](#), [NotEqualTo](#), [LessThanOrEqualTo](#), [GreaterThanOrEqualTo](#) }

## Public Member Functions

- [ImpulseCollection](#) ()  
*Constructor for [ImpulseCollection](#).*
- virtual [~ImpulseCollection](#) ()  
*Destructor for [ImpulseCollection](#).*
- double [getActivity](#) ([common::Cycle](#) cycle) const  
*Get activity at cycle.*
- double [getActivity](#) () const  
*Get activity at current cycle.*
- void [clearImpulses](#) ()  
*Clear collection up to present cycle.*
- void [clearImpulses](#) ([common::Cycle](#) cycle)  
*Clear collection up to specified cycle.*
- std::map< boost::uuids::uuid, boost::shared\_ptr< [Impulse](#) > > [clearImpulses](#) ([common::Cycle](#) cycleStart, [common::Cycle](#) cycleEnd)  
*Clear the Impulses that start on or after cycle start parameter and finish before cycle end parameter.*
- std::map< boost::uuids::uuid, boost::shared\_ptr< [Impulse](#) > > [clearActiveImpulses](#) ()  
*Clear cycles that are active on this cycle.*
- std::map< boost::uuids::uuid, boost::shared\_ptr< [Impulse](#) > > [clearActiveImpulses](#) ([common::Cycle](#) cycle)  
*Clear cycles that are active on cycle.*
- std::map< boost::uuids::uuid, boost::shared\_ptr< [Impulse](#) > > [clearActiveImpulses](#) ([common::Cycle](#) cycleStart, [common::Cycle](#) cycleEnd)  
*Clear cycles that are active on cycle range.*
- std::map< boost::uuids::uuid, boost::shared\_ptr< [Impulse](#) > > [clearActivitiesByMinimum](#) (double activity)  
*Clear the Impulses that have activities less than parameter.*
- std::map< boost::uuids::uuid, boost::shared\_ptr< [Impulse](#) > > [clearActivitiesByMaximum](#) (double activity)  
*Clear the Impulses that have activities greater than parameter.*

- void [decrementActivityTimers](#) ()  
*Decrement the activity timers of all impulses.*
- std::list< boost::shared\_ptr < [Impulse](#) > > [getByActivityTimerValue](#) (double value, [Comparison](#) comp)  
*Get impulse list by activity timer value.*
- std::list< boost::shared\_ptr < [Impulse](#) > > [removeByActivityTimerValue](#) (double value=0, [Comparison](#) comp=[LessThanOrEqualTo](#))  
*remove impulses by activity timer value*
- virtual void [refreshDataObject](#) ()  
*Inherited from DataObjectController.*
- virtual void [enableDebug](#) (bool b)
- [ImpulseCollection](#) & [operator=](#) (const [ImpulseCollection](#) &obj)  
*Assignment operator.*
- [ImpulseCollection](#) & [operator+=](#) (const [ImpulseCollection](#) &obj)  
*Destructive addition and assignment operator.*
- const [ImpulseCollection](#) [operator+](#) (const [ImpulseCollection](#) &obj) const  
*Non-destructive addition operator.*
- bool [operator==](#) (const [ImpulseCollection](#) &obj) const  
*Comparator operator.*
- bool [operator!=](#) (const [ImpulseCollection](#) &obj) const  
*Not comparator operator.*
- virtual void [enableLogging](#) (bool enable)  
*Whether logging is enabled or not.*
- virtual const std::map < unsigned long int, double > & [getMap](#) ()  
*Get all cycle values.*
- virtual const [dataobjects::DataObject](#) < unsigned long int, double > & [getDataObject](#) ()  
*Get data object.*

### Protected Member Functions

- std::map< boost::uuids::uuid, boost::shared\_ptr< [Impulse](#) > > [clearActivitiesByValue](#) (double activity, bool greater)  
*Get the associated Mesh.*

### Protected Attributes

- [dataobjects::DataObject](#) < unsigned long int, double > [dataObject](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [ImpulseCollection](#) &obj)  
*To stream operator.*

### 6.26.1 Detailed Description

[ImpulseCollection](#) represents a collection of [Impulse](#) objects.

A collection of Impulses that allows for Impulses to be held, 'moved forward' in time, and summated in some way

Definition at line 35 of file ImpulseCollection.h.

### 6.26.2 Member Enumeration Documentation

#### 6.26.2.1 enum cryomesh::components::ImpulseCollection::Comparison

Enumerator:

***GreaterThan***  
***LessThan***  
***EqualTo***  
***NotEqualTo***  
***LessThanOrEqualTo***  
***GreaterThanOrEqualTo***

Definition at line 38 of file ImpulseCollection.h.

### 6.26.3 Constructor & Destructor Documentation

#### 6.26.3.1 cryomesh::components::ImpulseCollection::ImpulseCollection ( )

Contructor for [ImpulseCollection](#).

Contruct using default Mesh

Definition at line 18 of file ImpulseCollection.cpp.

#### 6.26.3.2 cryomesh::components::ImpulseCollection::~~ImpulseCollection ( ) [virtual]

Destructor for [ImpulseCollection](#).

Destructor

Definition at line 22 of file ImpulseCollection.cpp.

### 6.26.4 Member Function Documentation

#### 6.26.4.1 std::map< boost::uuids::uuid, boost::shared\_ptr< Impulse > > cryomesh::components::ImpulseCollection::clearActiveImpulses ( )

Clear cycles that are active on this cycle.

Update the collection to by dropping all impulses that are active on this cycle

#### Returns

`std::map<boost::uuids::uuid, boost::shared_ptr<Impulse> >` The collection of deleted impulses

Definition at line 122 of file `ImpulseCollection.cpp`.

References `cryomesh::common::TimeKeeper::getTimeKeeper()`.

Referenced by `clearActiveImpulses()`, `cryomesh::components::Node::enterRecovery()`, and `cryomesh::components::Node::update()`.

**6.26.4.2** `std::map< boost::uuids::uuid, boost::shared_ptr< Impulse > >`  
**`cryomesh::components::ImpulseCollection::clearActiveImpulses (`**  
**`common::Cycle cycle )`**

Clear cycles that are active on cycle.

Update the collection to by dropping all impulses that are active on cycle

#### Parameters

<a href="#"><i>common::Cycle</i></a>	cycle The cycle to drop inclusive impulses from
--------------------------------------	---

#### Returns

`std::map<boost::uuids::uuid, boost::shared_ptr<Impulse> >` The collection of deleted impulses

Definition at line 127 of file `ImpulseCollection.cpp`.

References `clearActiveImpulses()`.

**6.26.4.3** `std::map< boost::uuids::uuid, boost::shared_ptr< Impulse > >`  
**`cryomesh::components::ImpulseCollection::clearActiveImpulses (`**  
**`common::Cycle cycleStart, common::Cycle cycleEnd )`**

Clear cycles that are active on cycle range.

Interval is [cycle\_start,cycle\_end)

Update the collection to by dropping all impulses that are active on cycle range

#### Parameters

<a href="#"><i>common::Cycle</i></a>	cycleStart The start cycle to drop inclusive impulses from
<a href="#"><i>common::Cycle</i></a>	cycleEnd The end cycle to drop inclusive impulses from excluded



**Returns**

`std::map<boost::uuids::uuid, boost::shared_ptr<Impulse> >` The collection of deleted impulses

Definition at line 131 of file `ImpulseCollection.cpp`.

References `cryomesh::common::TimeKeeper::getCycle()`, `cryomesh::components::Impulse::getFirstActiveCycle()`, `cryomesh::components::Impulse::getLastActiveCycle()`, `cryomesh::common::TimeKeeper::getTimeKeeper()`, and `cryomesh::components::Impulse::isActive()`.

**6.26.4.4** `std::map< boost::uuids::uuid, boost::shared_ptr< Impulse > >`  
**`cryomesh::components::ImpulseCollection::clearActivitiesByMaximum`**  
**`( double activity )`**

Clear the Impulses that have activities greater than parameter.

**Parameters**

<i>double</i>	activity The maximum activity impulses must have to avoid deleteion
---------------	---

**Returns**

`std::map<boost::uuids::uuid, boost::shared_ptr<Impulse> >` The deleted collection of impulses

Definition at line 188 of file `ImpulseCollection.cpp`.

References `clearActivitiesByValue()`.

**6.26.4.5** `std::map< boost::uuids::uuid, boost::shared_ptr< Impulse > >`  
**`cryomesh::components::ImpulseCollection::clearActivitiesByMinimum`**  
**`( double activity )`**

Clear the Impulses that have activities less than parameter.

**Parameters**

<i>double</i>	activity The minimum activity impulses must have to avoid deleteion
---------------	---

**Returns**

`std::map<boost::uuids::uuid, boost::shared_ptr<Impulse> >` The deleted collection of impulses

Definition at line 184 of file `ImpulseCollection.cpp`.

References `clearActivitiesByValue()`.

6.26.4.6 `std::map< boost::uuids::uuid, boost::shared_ptr< Impulse > >`  
`cryomesh::components::ImpulseCollection::clearActivitiesByValue (`  
`double activity, bool greater )` `[protected]`

Get the associated Mesh.

#### Returns

Mesh

`const boost::shared_ptr<Mesh> getMesh() const;` Clear the Impulses that have activities greater or less than parameter

#### Parameters

<i>double</i>	activity The maximum or minimum activity impulses must have to avoid deleteion
<i>bool</i>	True is first parameter is maximum allowed value, false if its the minimum

#### Returns

`std::map<boost::uuids::uuid, boost::shared_ptr<Impulse> >` The deleted collection of impulses

Definition at line 400 of file `ImpulseCollection.cpp`.

Referenced by `clearActivitiesByMaximum()`, and `clearActivitiesByMinimum()`.

6.26.4.7 `void cryomesh::components::ImpulseCollection::clearImpulses ( )`

Clear collection up to present cycle.

Update the collection to present cycle (non-inclusive) by dropping all impulses that are 'in the past' relative to that cycle. Interval is `[0,present_cycle)`

Definition at line 56 of file `ImpulseCollection.cpp`.

References `cryomesh::common::TimeKeeper::getCycle()`, and `cryomesh::common::TimeKeeper::getTimeKeeper()`.

Referenced by `clearImpulses()`, and `cryomesh::components::Node::updateImpulses()`.

6.26.4.8 `void cryomesh::components::ImpulseCollection::clearImpulses (`  
`common::Cycle cycle )`

Clear collection up to specified cycle.

Update the collection to specified cycle (non-inclusive) by dropping all impulses that are 'in the past' relative to that cycle. Interval is `[0,cycle)`

## Parameters

<i>common::Cycle</i>	cycle The cycle that is the cutoff point for the collection
----------------------	---

Definition at line 60 of file ImpulseCollection.cpp.

References `clearImpulses()`.

**6.26.4.9** `std::map< boost::uuids::uuid, boost::shared_ptr< Impulse > > cryomesh::components::ImpulseCollection::clearImpulses ( common::Cycle cycleStart, common::Cycle cycleEnd )`

Clear the Impulses that start on or after cycle start parameter and finish before cycle end parameter.

Interval is `[cycle_start, cycle_end)`

## Parameters

<i>Cycle</i>	cycleStart Cycle parameter that marks the start of the cleared area
<i>Cycle</i>	cycleEnd Cycle parameter that marks the end of the cleared area (non-inclusive)

## Returns

`std::map< boost::uuids::uuid, boost::shared_ptr< Impulse > >` The deleted collection of impulses

Definition at line 64 of file ImpulseCollection.cpp.

References `cryomesh::common::TimeKeeper::getCycle()`, and `cryomesh::common::TimeKeeper::getTimeKeeper()`.

**6.26.4.10** `void cryomesh::components::ImpulseCollection::decrementActivityTimers ( )`

Decrement the activity timers of all impulses.

Definition at line 192 of file ImpulseCollection.cpp.

Referenced by `cryomesh::components::Connection::update()`.

**6.26.4.11** `void cryomesh::components::ImpulseCollection::enableDebug ( bool b ) [virtual]`

Definition at line 320 of file ImpulseCollection.cpp.

**6.26.4.12** `virtual void cryomesh::dataobjects::DataObjectController< unsigned long int, double >::enableLogging ( bool enable )` [inline, virtual, inherited]

Whether logging is enabled or not.

#### Parameters

<i>bool</i>	enable True to enable logging, false otherwise
-------------	--

Definition at line 47 of file DataObjectController.h.

References cryomesh::dataobjects::DataObjectController< U, T >::dataObject.

**6.26.4.13** `double cryomesh::components::ImpulseCollection::getActivity ( common::Cycle cycle ) const`

Get activity at cycle.

Sum all the Impulses in the collection on specified cycle and return activity

#### Parameters

<i>Cycle</i>	cycle The cycle to calculate the activity on
--------------	--

#### Returns

double The activity on specified cycle

Definition at line 25 of file ImpulseCollection.cpp.

Referenced by cryomesh::components::Node::getActivity().

**6.26.4.14** `double cryomesh::components::ImpulseCollection::getActivity ( ) const`

Get activity at current cycle.

Sum all the Impulses in the collection on the current cycle and return activity

#### Returns

double The activity on specified cycle

Definition at line 51 of file ImpulseCollection.cpp.

References cryomesh::common::TimeKeeper::getCycle(), and cryomesh::common::TimeKeeper::getTimeKeeper().

Referenced by refreshDataObject().

6.26.4.15 `std::list< boost::shared_ptr< Impulse > > cryomesh::components-  
::ImpulseCollection::getByActivityTimerValue ( double value,  
ImpulseCollection::Comparison comp )`

Get impulse list by activity timer value.

#### Parameters

<i>double</i>	value activity timer value
<i>Comparison</i>	comp What comparison to make with the value

#### Returns

`std::list<boost::shared_ptr< Impulse> >` The list of impulses that meet the comparison

Definition at line 208 of file ImpulseCollection.cpp.

References `EqualTo`, `GreaterThan`, `GreaterThanOrEqualTo`, `LessThan`, and `LessThanOrEqualTo`.

Referenced by `removeByActivityTimerValue()`.

6.26.4.16 `virtual const dataobjects::DataObject<unsigned long int , double >&  
cryomesh::dataobjects::DataObjectController< unsigned long int , double  
>::getDataObject ( ) [inline, virtual, inherited]`

Get data object.

#### Returns

`dataobjects::DataObject<U,T> &` The data object

Definition at line 68 of file DataObjectController.h.

References `cryomesh::dataobjects::DataObjectController< U, T >::dataObject`, and `cryomesh::dataobjects::DataObjectController< U, T >::refreshDataObject()`.

6.26.4.17 `virtual const std::map<unsigned long int , double >& cryomesh::dataobjects-  
::DataObjectController< unsigned long int , double >::getMap ( )  
[inline, virtual, inherited]`

Get all cycle values.

#### Returns

`std::map<unsigned long int, double> &` The cycle values

Definition at line 57 of file DataObjectController.h.

References `cryomesh::dataobjects::DataObjectController< U, T >::dataObject`, and `cryomesh::dataobjects::DataObjectController< U, T >::refreshDataObject()`.

6.26.4.18 **bool** cryomesh::components::ImpulseCollection::operator!= ( **const** ImpulseCollection & *obj* ) **const**

Not comparator operator.

#### Parameters

<i>const</i>	<a href="#">ImpulseCollection</a> & obj RHS object
--------------	--

#### Returns

bool True if not equal, false otherwise

Definition at line 371 of file ImpulseCollection.cpp.

6.26.4.19 **const** ImpulseCollection cryomesh::components::ImpulseCollection::operator+ ( **const** ImpulseCollection & *obj* ) **const**

Non-destructive addition operator.

#### Parameters

<i>const</i>	<a href="#">ImpulseCollection</a> & obj RHS addition
--------------	--

#### Returns

[ImpulseCollection](#) New object after addition

Definition at line 314 of file ImpulseCollection.cpp.

6.26.4.20 **ImpulseCollection** & cryomesh::components::ImpulseCollection::operator+= ( **const** ImpulseCollection & *obj* )

Destructive addition and assignment operator.

#### Parameters

<i>const</i>	<a href="#">ImpulseCollection</a> & obj RHS addition
--------------	--

#### Returns

[ImpulseCollection](#) & This object after addition and assignment

Definition at line 294 of file ImpulseCollection.cpp.

#### 6.26.4.21 ImpulseCollection & cryomesh::components::ImpulseCollection::operator= ( const ImpulseCollection & obj )

Assignment operator.

##### Parameters

<i>const</i>	<a href="#">ImpulseCollection</a> & obj RHS assignment
--------------	--

##### Returns

[ImpulseCollection](#) & This object after assignment

Definition at line 285 of file ImpulseCollection.cpp.

#### 6.26.4.22 bool cryomesh::components::ImpulseCollection::operator== ( const ImpulseCollection & obj ) const

Comparator operator.

##### Parameters

<i>const</i>	<a href="#">ImpulseCollection</a> & obj RHS object
--------------	--

##### Returns

bool True if equal, false otherwise

Definition at line 324 of file ImpulseCollection.cpp.

#### 6.26.4.23 void cryomesh::components::ImpulseCollection::refreshDataObject ( ) [virtual]

Inherited from DataObjectController.

Overriden to force refresh update on call

Reimplemented from [cryomesh::dataobjects::DataObjectController< unsigned long int, double >](#).

Definition at line 265 of file ImpulseCollection.cpp.

References [cryomesh::dataobjects::DataObject< U, T >::clear\(\)](#), [cryomesh::dataobjects::DataObjectController< unsigned long int, double >::dataObject](#), [getActivity\(\)](#), [cryomesh::common::TimeKeeper::getCycle\(\)](#), [cryomesh::dataobjects::DataObject< U, T >::getDatasetMaximumSize\(\)](#), [cryomesh::common::TimeKeeper::getTimeKeeper\(\)](#), [cryomesh::dataobjects::DataObject< U, T >::insert\(\)](#), [cryomesh::dataobjects::DataObject< U, T >::isLoggingEnabled\(\)](#), and [cryomesh::common::Cycle::toULInt\(\)](#).

6.26.4.24 `std::list< boost::shared_ptr< Impulse > > cryomesh::components-  
::ImpulseCollection::removeByActivityTimerValue ( double value = 0,  
ImpulseCollection::Comparison comp = LessThanOrEqualTo )`

remove impulses by activity timer value

#### Parameters

<i>double</i>	value activity timer value
<i>Comparison</i>	comp What comparison to make with the value

#### Returns

`std::list<boost::shared_ptr< Impulse> >` The that meet the comparison and were removed

Definition at line 258 of file ImpulseCollection.cpp.

References `getByActivityTimerValue()`.

Referenced by `cryomesh::components::Connection::update()`.

### 6.26.5 Friends And Related Function Documentation

6.26.5.1 `std::ostream& operator<< ( std::ostream & os, const ImpulseCollection & obj )`  
[friend]

To stream operator.

#### Parameters

<i>std::ostream</i>	& <i>os</i> The output stream
<i>const</i>	<a href="#">ImpulseCollection</a> & <i>obj</i> The object to stream

#### Returns

`std::ostream &` The output stream

Definition at line 375 of file ImpulseCollection.cpp.

### 6.26.6 Member Data Documentation

6.26.6.1 `dataobjects::DataObject<unsigned long int , double >`  
`cryomesh::dataobjects::DataObjectController< unsigned long int , double`  
`>::dataObject` [protected, inherited]

Definition at line 85 of file DataObjectController.h.

Referenced by `refreshDataObject()`, and `cryomesh::components::Node::update()`.



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

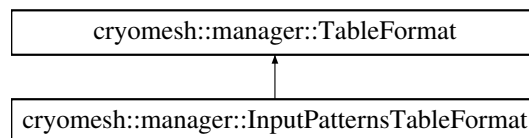
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ImpulseCollection.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ImpulseCollection.cpp](#)

## 6.27 cryomesh::manager::InputPatternsTableFormat Struct Reference

Struct representing input pattern table structure.

```
#include <TableFormats.h>
```

Inheritance diagram for cryomesh::manager::InputPatternsTableFormat:



### Public Member Functions

- [InputPatternsTableFormat \(\)](#)  
*Default constructor will construct all the names and columns associated with a pattern table.*
- `std::string` [getName \(\)](#) const  
*Return the name of the table.*
- `std::string` [getKey](#) (const `std::string` &key)  
*Return the string object associated with a key.*
- `std::string` [getCreateTable \(\)](#) const  
*Get the string that can be used to create the sql table.*

### Protected Attributes

- `std::string` [name](#)
- `std::map< std::string, std::string >` [columns](#)

#### 6.27.1 Detailed Description

Struct representing input pattern table structure.

Definition at line 136 of file TableFormats.h.

## 6.27.2 Constructor & Destructor Documentation

### 6.27.2.1 `cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat ( )` [inline]

Default constructor will construct all the names and columns associated with a pattern table.

Definition at line 141 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`, and `cryomesh::manager::TableFormat::name`.

## 6.27.3 Member Function Documentation

### 6.27.3.1 `std::string cryomesh::manager::TableFormat::getCreateTable ( )` const [inline, inherited]

Get the string that can be used to create the sql table.

#### Returns

the sql command string to create this table

Definition at line 60 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`, and `cryomesh::manager::TableFormat::getName()`.

### 6.27.3.2 `std::string cryomesh::manager::TableFormat::getKey ( const std::string & key )` [inline, inherited]

Return the string object associated with a key.

`::string` The key to search for

#### Returns

`std::string` The object associated with the search key, "" if not found

Definition at line 45 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`.

### 6.27.3.3 `std::string cryomesh::manager::TableFormat::getName ( )` const [inline, inherited]

Return the name of the table.

**Returns**

std::string The name of the table

Definition at line 32 of file TableFormats.h.

References cryomesh::manager::TableFormat::name.

Referenced by cryomesh::manager::TableFormat::getCreateTable(), cryomesh::manager::DatabaseManager::insertConnection(), cryomesh::manager::DatabaseManager::insertNode(), and cryomesh::manager::DatabaseManager::insertOutputPattern().

**6.27.4 Member Data Documentation**

**6.27.4.1** `std::map<std::string, std::string> cryomesh::manager::TableFormat::columns` [protected, inherited]

Definition at line 93 of file TableFormats.h.

Referenced by cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat(), cryomesh::manager::TableFormat::getCreateTable(), cryomesh::manager::TableFormat::getKey(), InputPatternsTableFormat(), cryomesh::manager::NodeTableFormat::NodeTableFormat(), and cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat().

**6.27.4.2** `std::string cryomesh::manager::TableFormat::name` [protected, inherited]

Definition at line 86 of file TableFormats.h.

Referenced by cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat(), cryomesh::manager::TableFormat::getName(), InputPatternsTableFormat(), cryomesh::manager::NodeTableFormat::NodeTableFormat(), and cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat().

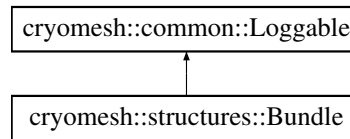
The documentation for this struct was generated from the following file:

- /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/[TableFormats.h](#)

**6.28 cryomesh::common::Loggable Class Reference**

```
#include <Loggable.h>
```

Inheritance diagram for cryomesh::common::Loggable:



## Public Types

- enum [LoggingDepth](#) { [SUMMARY](#), [MAX](#) }

*Enum representing print detail.*

## Public Member Functions

- [Loggable](#) ()
- virtual [~Loggable](#) ()
- virtual std::ostream & [print](#) (std::ostream &os, const [Loggable::LoggingDepth](#) depth=[Loggable::SUMMARY](#)) const =0

### 6.28.1 Detailed Description

Definition at line 17 of file Loggable.h.

### 6.28.2 Member Enumeration Documentation

#### 6.28.2.1 enum cryomesh::common::Loggable::LoggingDepth

Enum representing print detail.

Enumerator:

***SUMMARY***

***MAX***

Definition at line 23 of file Loggable.h.

### 6.28.3 Constructor & Destructor Documentation

#### 6.28.3.1 cryomesh::common::Loggable::Loggable ( ) [inline]

Definition at line 26 of file Loggable.h.

6.28.3.2 `virtual cryomesh::common::Loggable::~~Loggable ( ) [inline, virtual]`

Definition at line 27 of file Loggable.h.

## 6.28.4 Member Function Documentation

6.28.4.1 `virtual std::ostream& cryomesh::common::Loggable::print ( std::ostream & os, const Loggable::LoggingDepth depth = Loggable::SUMMARY ) const [pure virtual]`

Implemented in [cryomesh::structures::Bundle](#).

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/Loggable.h](#)

## 6.29 cryomesh::structures::Mesh Class Reference

[Mesh](#) is the fabric of connection space and warps and is warped by it.

```
#include <Mesh.h>
```

### Public Types

- enum [BlendingMethod](#) { [BLEND\\_LINEAR](#) }

### Public Member Functions

- [Mesh](#) ([Cluster](#) &clus)  
*Constructor.*
- [Mesh](#) (const [Mesh](#) &)  
*Copy Constructor.*
- virtual [~Mesh](#) ()  
*Destructor.*
- void [update](#) ()  
*Update mesh from cluster.*
- [components::Node](#) & [warp](#) ([components::Node](#) &node)  
*Warp the a Node using the mesh.*
- [components::ImpulseCollection](#) & [warp](#) ([components::ImpulseCollection](#) &impulseCollection)  
*Warp an ImpulseCollection using the mesh.*
- [cryomesh::components::Impulse](#) & [warp](#) ([cryomesh::components::Impulse](#) &impulse)

*Warp an Impulse using the mesh.*

- const [Cluster](#) & [getCluster](#) () const
- const boost::shared\_ptr < spacial::ActivityGrid > [getActivityGrid](#) () const

### Protected Member Functions

- double [getBlendedActivity](#) (const double first\_activity, const double second\_activity, const [BlendingMethod](#) blending\_method=[BLEND\\_LINEAR](#), double force=0.5)

### Private Attributes

- [Cluster](#) & [cluster](#)
- boost::shared\_ptr < spacial::ActivityGrid > [grid](#)
- const int [DEFAULT\\_MESH\\_GRANULARITY](#)
- const int [DEFAULT\\_BLEND\\_FORCE](#)

#### 6.29.1 Detailed Description

[Mesh](#) is the fabric of connection space and warps and is warped by it.

The [Mesh](#) is an overlying structure on top of Connections space that can be used to warp the underlying space, Impulses, Connections, Nodes, etc based on spacial criteria on any other criteria that can be applied to the underlying data objects

Definition at line 33 of file Mesh.h.

#### 6.29.2 Member Enumeration Documentation

##### 6.29.2.1 enum cryomesh::structures::Mesh::BlendingMethod

Enumerator:

**[BLEND\\_LINEAR](#)**

Definition at line 36 of file Mesh.h.

#### 6.29.3 Constructor & Destructor Documentation

##### 6.29.3.1 cryomesh::structures::Mesh::Mesh ( [Cluster](#) & *clus* )

Constructor.

Constructor for [Mesh](#). Inaccessible to force singleton class

Definition at line 32 of file Mesh.cpp.

References [update\(\)](#).

## 6.29.3.2 cryomesh::structures::Mesh::Mesh ( const Mesh &amp; )

Copy Constructor.

Overridden Copy Constructor for [Mesh](#). Inaccessible to force singleton class

## Parameters

<a href="#">Mesh</a>	Object to Copy Construct from
----------------------	-------------------------------

## 6.29.3.3 cryomesh::structures::Mesh::~~Mesh ( ) [virtual]

Destructor.

Definition at line 43 of file Mesh.cpp.

## 6.29.4 Member Function Documentation

## 6.29.4.1 const boost::shared\_ptr&lt; spacial::ActivityGrid &gt; cryomesh::structures::Mesh::getActivityGrid ( ) const

Definition at line 109 of file Mesh.cpp.

References [grid](#).

6.29.4.2 double cryomesh::structures::Mesh::getBlendedActivity ( const double *first\_activity*, const double *second\_activity*, const BlendingMethod *blending\_method* = BLEND\_LINEAR, double *force* = 0.5 ) [protected]

Definition at line 93 of file Mesh.cpp.

References [BLEND\\_LINEAR](#).

Referenced by [warp\(\)](#).

## 6.29.4.3 const Cluster &amp; cryomesh::structures::Mesh::getCluster ( ) const

Definition at line 105 of file Mesh.cpp.

References [cluster](#).

## 6.29.4.4 void cryomesh::structures::Mesh::update ( )

Update mesh from cluster.

Definition at line 46 of file Mesh.cpp.

References [cluster](#), [cryomesh::components::Node::getActivity\(\)](#), [cryomesh::structures::Cluster::getNodes\(\)](#), [cryomesh::components::Node::getPosition\(\)](#), and [grid](#).

Referenced by Mesh().

#### 6.29.4.5 Node & cryomesh::structures::Mesh::warp ( components::Node & *node* )

Warp the a Node using the mesh.

This function will use any values of the node, such as position in space for example, to apply a warp to the node. In practice this might be to suppress or increase the activity threshold, or to scale the activities at that node in some way. Note that this is a permanent change, ie, the node is warped 'in place'

##### Parameters

<i>Node</i>	& node The node to be warped
-------------	------------------------------

##### Returns

Node & The warped node

Definition at line 71 of file Mesh.cpp.

References BLEND\_LINEAR, DEFAULT\_BLEND\_FORCE, cryomesh::components::Node::getActivity(), getBlendedActivity(), cryomesh::components::Node::getPosition(), grid, and cryomesh::components::Node::setActivity().

#### 6.29.4.6 ImpulseCollection & cryomesh::structures::Mesh::warp ( components::ImpulseCollection & *impulseCollection* )

Warp an ImpulseCollection using the mesh.

##### Parameters

<i>Impulse-Collection</i>	& ImpulseCollection
---------------------------	---------------------

##### Returns

ImpulseCollection & The warped collection

Definition at line 81 of file Mesh.cpp.

#### 6.29.4.7 Impulse & cryomesh::structures::Mesh::warp ( cryomesh::components::Impulse & *impulse* )

Warp an Impulse using the mesh.

##### Parameters

<i>Impulse</i>	& Impulse
----------------	-----------



## 6.30 cryomesh::structures::NodeMesh::NeighbourhoodRanges Struct Reference 245

### Returns

Impulse & The warped Impulse

Definition at line 87 of file Mesh.cpp.

### 6.29.5 Member Data Documentation

#### 6.29.5.1 Cluster& cryomesh::structures::Mesh::cluster [private]

Definition at line 110 of file Mesh.h.

Referenced by getCluster(), and update().

#### 6.29.5.2 const int cryomesh::structures::Mesh::DEFAULT\_BLEND\_FORCE [private]

Definition at line 113 of file Mesh.h.

Referenced by warp().

#### 6.29.5.3 const int cryomesh::structures::Mesh::DEFAULT\_MESH\_GRANULARITY [private]

Definition at line 112 of file Mesh.h.

#### 6.29.5.4 boost::shared\_ptr<spacial::ActivityGrid> cryomesh::structures::Mesh::grid [private]

Definition at line 111 of file Mesh.h.

Referenced by getActivityGrid(), update(), and warp().

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Mesh.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/Mesh.cpp](#)

## 6.30 cryomesh::structures::NodeMesh::NeighbourhoodRanges - Struct Reference

Struct to capture some statistics data on a nodes neighbourhood.

```
#include <NodeMesh.h>
```

## Public Attributes

- int [minimumNeighbourCount](#)
- int [maximumNeighbourCount](#)
- double [minimumNeighbourDistance](#)
- double [maximumNeighbourDistance](#)

### 6.30.1 Detailed Description

Struct to capture some statistics data on a nodes neighbourhood.

Definition at line 40 of file NodeMesh.h.

### 6.30.2 Member Data Documentation

#### 6.30.2.1 int cryomesh::structures::NodeMesh::NeighbourhoodRanges::maximumNeighbourCount

Definition at line 42 of file NodeMesh.h.

Referenced by cryomesh::structures::NodeMesh::getNeighbourRanges().

#### 6.30.2.2 double cryomesh::structures::NodeMesh::NeighbourhoodRanges::maximumNeighbourDistance

Definition at line 44 of file NodeMesh.h.

Referenced by cryomesh::structures::NodeMesh::getNeighbourRanges().

#### 6.30.2.3 int cryomesh::structures::NodeMesh::NeighbourhoodRanges::minimumNeighbourCount

Definition at line 41 of file NodeMesh.h.

Referenced by cryomesh::structures::NodeMesh::getNeighbourRanges().

#### 6.30.2.4 double cryomesh::structures::NodeMesh::NeighbourhoodRanges::minimumNeighbourDistance

Definition at line 43 of file NodeMesh.h.

Referenced by cryomesh::structures::NodeMesh::getNeighbourRanges().

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

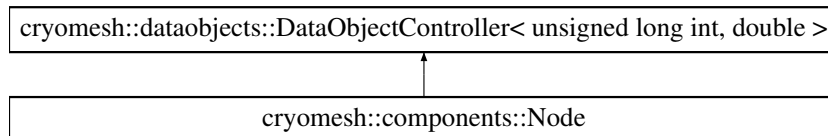
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/NodeMesh.h](#)

## 6.31 cryomesh::components::Node Class Reference

**Node** is an accumulation and computational nodal point of impulses.

```
#include <Node.h>
```

Inheritance diagram for cryomesh::components::Node:



### Public Types

- enum **ActivationState** { **Positive**, **Negative**, **None** }  
*Enum representing possible activation states.*
- enum **RecoverySetting** { **CLEAR\_ALL\_IMPULSES** = 1, **CLEAR\_ACTIVE\_IMPULSES** = 2, **DEACTIVATE\_DURING\_RECOVERY** = 4 }

### Public Member Functions

- Node** ()  
*Constructor.*
- virtual **~Node** ()  
*Destructor.*
- virtual void **update** ()
- virtual void **forceFire** ()  
*Force the node to fire.*
- virtual boost::shared\_ptr< **Impulse** > **addImpulse** (boost::shared\_ptr< **Impulse** > impulse)  
*Add incoming Impulse.*
- virtual void **addImpulses** (std::list< boost::shared\_ptr< **Impulse** > > impulses)  
*Add a list of incoming Impulses.*
- const **common::Connector**< **Node**, **Connection** > & **getConnector** () const  
*Get the Connector object for this Node.*
- common::Connector**< **Node**, **Connection** > & **getMutableConnector** ()  
*Get the mutable Connector object for this Node.*
- bool **isInputIsolated** () const
- bool **isOutputIsolated** () const
- void **connectInput** (boost::shared\_ptr< **Connection** > con)
- void **connectOutput** (boost::shared\_ptr< **Connection** > con)
- const **ImpulseCollection** & **getImpulses** () const  
*Get the collection of Impulses for this Node.*

- `const boost::shared_ptr< Impulse > getEmittedImpulse () const`  
*Get the Impulse that is emitted.*
- `boost::shared_ptr< Impulse > getMutableEmittedImpulse ()`  
*Get the mutable Impulse that is emitted.*
- `virtual void emitImpulsePositive ()`  
*Emit a positive impulse to outgoing connections.*
- `virtual void emitImpulseNegative ()`  
*Emit a negative impulse to outgoing connections.*
- `ImpulseCollection & getMutableImpulses ()`  
*Get the mutable collection of Impulses for this Node.*
- `const std::map< common::Cycle, double > & getActivities () const`  
*Get the collection of all activities.*
- `double updateActivity ()`  
*Update and get the current activity of the node.*
- `double updateActivity (const common::Cycle &cycle)`  
*Update and get the activity of the node on specific cycle.*
- `double getActivity () const`  
*Get the current activity of the node.*
- `double getActivity (const common::Cycle &cycle) const`  
*Get the activity of the node on specific cycle.*
- `double getActivityThreshold () const`
- `double setActivity (double activity)`  
*Set the current activity of the node.*
- `double setActivity (const common::Cycle &cycle, double activity)`  
*Set the activity at cycle of the node.*
- `boost::shared_ptr < manager::DatabaseObject > getDatabaseObject () const`  
*Return a database object for this node.*
- `const spacial::Point & getPosition () const`  
*get the position of the node*
- `void setPosition (const spacial::Point &new_position)`  
*Set the spacial position of the node, remembering to update connections lengths.*
- `ActivationState getLastActivationState () const`  
*Get the last activation state.*
- `void randomise ()`  
*Randomise the nodes state.*
- `virtual void enableDebug (bool b)`
- `bool isTriggered (ActivationState state=None)`  
*Check if Node is currently triggered.*
- `bool isActive (const ActivationState state=None)`  
*Check if Node is currently activated.*
- `bool isLive ()`  
*Check if Node is live, ie active at any point in now or the future.*
- `void destroyAllConnections ()`

*Destroy all connections.*

- void [destroyAllInputConnections](#) ()

*Destroy Input connections by removing them all from both their attached input and output nodes.*

- void [destroyAllOutputConnections](#) ()

*Destroy Output connections by removing them all from both their attached input and output nodes.*

- bool [isPrimaryInputAttachedNode](#) () const
- bool [isPrimaryOutputAttachedNode](#) () const
- std::vector< boost::shared\_ptr < [Connection](#) > > [getPrimaryInputConnections](#) ()
- std::vector< boost::shared\_ptr < [Connection](#) > > [getPrimaryOutputConnections](#) ()
- std::ostream & [printConnections](#) (std::ostream &os, const std::map< boost::uuids::uuid, boost::shared\_ptr< [Connection](#) > > &all\_cons, const std::string formatter="") const
- virtual void [enableLogging](#) (bool enable)

*Whether logging is enabled or not.*

- virtual const std::map < unsigned long int, double > & [getMap](#) ()

*Get all cycle values.*

- virtual const [dataobjects::DataObject](#) < unsigned long int, double > & [getDataObject](#) ()

*Get data object.*

- virtual void [refreshDataObject](#) ()

*Function to allow refreshing implementation if required by subclasses.*

## Static Public Member Functions

- static boost::shared\_ptr< [Node](#) > [getRandom](#) (const spacial::Point &max\_point=[MAX\\_BOUNDING\\_BOX\\_POINT](#))

## Static Public Attributes

- static const int [MAX\\_ACTIVITIES\\_LENGTH](#) = 10
- static const double [MAX\\_ACTIVITY\\_THRESHOLD](#) = 3 \* [Impulse::MAX\\_ACTIVITY](#)
- static const double [MIN\\_ACTIVITY\\_THRESHOLD](#) = 1 \* [Impulse::MAX\\_ACTIVITY](#)
- static const spacial::Point [MAX\\_BOUNDING\\_BOX\\_POINT](#) = spacial::Point(100, 100, 100)

### Protected Member Functions

- virtual [Node::ActivationState](#) [checkActivationState](#) ()  
*Check level of impulses and decide whether to activate the node.*
- virtual [Node::ActivationState](#) [checkFire](#) ()  
*Check if the object is ready to fire off an impulse and carry it out.*
- virtual void [updateImpulses](#) ()  
*Update the collection of impulses by one cycle.*
- virtual void [emitImpulse](#) (bool positive)  
*Emit an impulse to outgoing connections.*
- virtual double [addActivity](#) ([common::Cycle](#), double activity)  
*Add an activity to the list of activities.*
- virtual void [updatePosition](#) ()  
*Recalculate state of node and connections based on current position.*
- virtual void [enterRecovery](#) (const int recovery\_settings=[CLEAR\\_ALL\\_IMPULSES](#))

### Protected Attributes

- [dataobjects::DataObject](#) < unsigned long int, double > [dataObject](#)

### Private Attributes

- double [activityThreshold](#)
- boost::shared\_ptr < [common::Connector](#)< [Node](#), [Connection](#) > > [connector](#)
- [ImpulseCollection](#) [impulses](#)
- boost::shared\_ptr< [Impulse](#) > [emittedImpulse](#)
- [dataobjects::DataObject](#) < [common::Cycle](#), double > [activities](#)
- spacial::Point [position](#)
- [ActivationState](#) [lastActivationState](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [Node](#) &obj)  
*To stream operator.*

### 6.31.1 Detailed Description

[Node](#) is an accumulation and computational nodal point of impulses.

A [Node](#) represents the end point of one or many connections. Here, Impulses are accumulated and new Impulses generated depending on some determining criteria

Definition at line 38 of file [Node.h](#).

### 6.31.2 Member Enumeration Documentation

#### 6.31.2.1 enum cryomesh::components::Node::ActivationState

Enum representing possible activation states.

Last activation state.

Enumerator:

***Positive***

***Negative***

***None***

Definition at line 45 of file Node.h.

#### 6.31.2.2 enum cryomesh::components::Node::RecoverySetting

Enumerator:

***CLEAR\_ALL\_IMPULSES***

***CLEAR\_ACTIVE\_IMPULSES***

***DEACTIVATE\_DURING\_RECOVERY***

Definition at line 49 of file Node.h.

### 6.31.3 Constructor & Destructor Documentation

#### 6.31.3.1 cryomesh::components::Node::Node ( )

Constructor.

Constructor for [Node](#)

Definition at line 37 of file Node.cpp.

References [activities](#), [connector](#), [emittedImpulse](#), [MAX\\_ACTIVITIES\\_LENGTH](#), and [cryomesh::dataobjects::DataObject< U, T >::setDatasetMaximumSize\(\)](#).

#### 6.31.3.2 cryomesh::components::Node::~~Node ( ) [virtual]

Destructor.

Destructor for [Node](#)

Definition at line 46 of file Node.cpp.

### 6.31.4 Member Function Documentation

**6.31.4.1** `double cryomesh::components::Node::addActivity ( common::Cycle cycle,  
double activity )` `[protected, virtual]`

Add an activity to the list of activities.

#### Parameters

<i>Cycle</i>	cycle The cycle this activity is on
<i>double</i>	activity The activity to add

#### Returns

The current activity

Definition at line 291 of file Node.cpp.

References `activities`, and `cryomesh::dataobjects::DataObject< U, T >::insert()`.

Referenced by `setActivity()`.

**6.31.4.2** `boost::shared_ptr< Impulse > cryomesh::components::Node::addImpulse (`  
`boost::shared_ptr< Impulse > impulse )` `[virtual]`

Add incoming [Impulse](#).

#### Parameters

<i>boost- ::shared_ ptr&lt;- Impulse&gt;</i>	impulse The <a href="#">Impulse</a> to add
--	--

#### Returns

`boost::shared_ptr<Impulse>` The impulse added, null if none added

Definition at line 141 of file Node.cpp.

References `getMutableImpulses()`, and `cryomesh::common::TimeKeeper::getTimeKeeper()`.

Referenced by `addImpulses()`, and `forceFire()`.

**6.31.4.3** `void cryomesh::components::Node::addImpulses ( std::list<`  
`boost::shared_ptr< Impulse > > impulses )` `[virtual]`

Add a list of incoming Impulses.



## Parameters

<code>std::list&lt;boost::shared_ptr&lt;Impulse&gt;</code>	> impulses The Impulses to add
--	--------------------------------

Definition at line 151 of file Node.cpp.

References `addImpulse()`, `getImpulses()`, and `impulses`.

#### 6.31.4.4 Node::ActivationState cryomesh::components::Node::checkActivationState ( ) [protected, virtual]

Check level of impulses and decide whether to activate the node.

## Returns

[Node::ActivationState](#) Positive if activity is over threshold, negative if under - threshold, None otherwise

Definition at line 173 of file Node.cpp.

References `getActivityThreshold()`, `Negative`, `None`, `Positive`, and `updateActivity()`.

Referenced by `checkFire()`.

#### 6.31.4.5 Node::ActivationState cryomesh::components::Node::checkFire ( ) [protected, virtual]

Check if the object is ready to fire off an impulse and carry it out.

## Returns

`ActivationState` Return the action that was taken

Definition at line 92 of file Node.cpp.

References `checkActivationState()`, `emitImpulseNegative()`, `emitImpulsePositive()`, `enterRecovery()`, `Negative`, and `Positive`.

Referenced by `update()`.

#### 6.31.4.6 void cryomesh::components::Node::connectInput ( boost::shared\_ptr<Connection> con )

Definition at line 469 of file Node.cpp.

References `getMutableConnector()`.

#### 6.31.4.7 void cryomesh::components::Node::connectOutput ( boost::shared\_ptr< Connection > con )

Definition at line 472 of file Node.cpp.

References getMutableConnector().

#### 6.31.4.8 void cryomesh::components::Node::destroyAllConnections ( )

Destroy all connections.

Definition at line 476 of file Node.cpp.

References destroyAllInputConnections(), and destroyAllOutputConnections().

#### 6.31.4.9 void cryomesh::components::Node::destroyAllInputConnections ( )

Destroy Input connections by removing them all from both their attached input and output nodes.

Definition at line 481 of file Node.cpp.

References getMutableConnector().

Referenced by destroyAllConnections().

#### 6.31.4.10 void cryomesh::components::Node::destroyAllOutputConnections ( )

Destroy Output connections by removing them all from both their attached input and output nodes.

Definition at line 498 of file Node.cpp.

References getMutableConnector().

Referenced by destroyAllConnections().

#### 6.31.4.11 void cryomesh::components::Node::emitImpulse ( bool positive ) [protected, virtual]

Emit an impulse to outgoing connections.

##### Parameters

<i>bool</i>	positive Is the impulse to be emitted positive or negative
-------------	--

Definition at line 199 of file Node.cpp.

References cryomesh::components::Connection::add(), getEmittedImpulse(), cryomesh::components::Connection::getImpulses(), getMutableConnector(), and getMutableEmittedImpulse().

Referenced by emitImpulseNegative(), and emitImpulsePositive().

**6.31.4.12** void **cryomesh::components::Node::emitImpulseNegative** ( )  
[virtual]

Emit a negative impulse to outgoing connections.

Definition at line 195 of file Node.cpp.

References emitImpulse().

Referenced by checkFire().

**6.31.4.13** void **cryomesh::components::Node::emitImpulsePositive** ( )  
[virtual]

Emit a positive impulse to outgoing connections.

Definition at line 191 of file Node.cpp.

References emitImpulse().

Referenced by checkFire().

**6.31.4.14** void **cryomesh::components::Node::enableDebug** ( bool *b* )  
[virtual]

Definition at line 545 of file Node.cpp.

**6.31.4.15** virtual void **cryomesh::dataobjects::DataObjectController**< unsigned long  
int, double >::enableLogging ( bool *enable* ) [inline, virtual,  
inherited]

Whether logging is enabled or not.

#### Parameters

<i>bool</i>	enable True to enable logging, false otherwise
-------------	--

Definition at line 47 of file DataObjectController.h.

References cryomesh::dataobjects::DataObjectController< U, T >::dataObject.

**6.31.4.16** void **cryomesh::components::Node::enterRecovery** ( const int  
*recovery\_settings* = CLEAR\_ALL\_IMPULSES ) [protected,  
virtual]

Definition at line 115 of file Node.cpp.

References CLEAR\_ACTIVE\_IMPULSES, CLEAR\_ALL\_IMPULSES, cryomesh-

`::components::ImpulseCollection::clearActiveImpulses()`, `cryomesh::common::TimeKeeper::getTimeKeeper()`, and `impulses`.

Referenced by `checkFire()`.

#### 6.31.4.17 `void cryomesh::components::Node::forceFire ( ) [virtual]`

Force the node to fire.

Definition at line 80 of file `Node.cpp`.

References `addImpulse()`, and `cryomesh::components::Impulse::getTriggerImpulse()`.

#### 6.31.4.18 `const std::map< common::Cycle, double > & cryomesh::components::Node::getActivities ( ) const`

Get the collection of all activities.

##### Returns

`std::list<double>` & List of activities

Definition at line 259 of file `Node.cpp`.

References `activities`, and `cryomesh::dataobjects::DataObject< U, T >::getMap()`.

Referenced by `update()`.

#### 6.31.4.19 `double cryomesh::components::Node::getActivity ( ) const`

Get the current activity of the node.

##### Returns

`double` The current activity

Definition at line 263 of file `Node.cpp`.

References `cryomesh::common::TimeKeeper::getTimeKeeper()`.

Referenced by `getDatabaseObject()`, `isActive()`, `cryomesh::structures::Mesh::update()`, `updateActivity()`, and `cryomesh::structures::Mesh::warp()`.

#### 6.31.4.20 `double cryomesh::components::Node::getActivity ( const common::Cycle & cycle ) const`

Get the activity of the node on specific cycle.

**Returns**

double The current activity

Definition at line 267 of file Node.cpp.

References cryomesh::components::ImpulseCollection::getActivity(), and getImpulses().

**6.31.4.21 double cryomesh::components::Node::getActivityThreshold ( ) const**

Definition at line 271 of file Node.cpp.

References activityThreshold.

Referenced by checkActivationState(), and cryomesh::components::operator<<().

**6.31.4.22 const common::Connector< Node, Connection > & cryomesh::components::Node::getConnector ( ) const**

Get the Connector object for this [Node](#).

**Returns**

const common::Connector<Node, Connection> & The Connector for this object

Definition at line 84 of file Node.cpp.

References connector.

Referenced by getPrimaryInputConnections(), getPrimaryOutputConnections(), isInputIsolated(), isOutputIsolated(), isPrimaryInputAttachedNode(), isPrimaryOutputAttachedNode(), and cryomesh::components::operator<<().

**6.31.4.23 boost::shared\_ptr< manager::DatabaseObject > cryomesh::components::Node::getDatabaseObject ( ) const**

Return a database object for this node.

**Returns**

DatabaseObject

Definition at line 296 of file Node.cpp.

References getActivity(), cryomesh::common::TimeKeeper::getCycle(), getPosition(), cryomesh::common::TimeKeeper::getTimeKeeper(), and cryomesh::common::Cycle::toULInt().

**6.31.4.24** `virtual const dataobjects::DataObject<unsigned long int , double >& cryomesh::dataobjects::DataObjectController< unsigned long int , double >::getDataObject ( ) [inline, virtual, inherited]`

Get data object.

#### Returns

`dataobjects::DataObject<U,T>` & The data object

Definition at line 68 of file `DataObjectController.h`.

References `cryomesh::dataobjects::DataObjectController< U, T >::dataObject`, and `cryomesh::dataobjects::DataObjectController< U, T >::refreshDataObject()`.

**6.31.4.25** `const boost::shared_ptr< Impulse > cryomesh::components::Node::getEmittedImpulse ( ) const`

Get the [Impulse](#) that is emitted.

#### Returns

`const boost::shared_ptr< Impulse >` The emitted [Impulse](#)

Definition at line 247 of file `Node.cpp`.

References `emittedImpulse`.

Referenced by `emitImpulse()`.

**6.31.4.26** `const ImpulseCollection & cryomesh::components::Node::getImpulses ( ) const`

Get the collection of Impulses for this [Node](#).

#### Returns

`const ImpulseCollection` & The collection of Impulses for this [Node](#)

Definition at line 243 of file `Node.cpp`.

References `impulses`.

Referenced by `addImpulses()`, `getActivity()`, `isLive()`, `cryomesh::components::operator<<()`, and `updateActivity()`.

**6.31.4.27** `Node::ActivationState cryomesh::components::Node::getLastActivationState ( ) const`

Get the last activation state.

**Returns**

ActivationState Return the last activation state

Definition at line 314 of file Node.cpp.

References lastActivationState.

Referenced by isTriggered().

**6.31.4.28** `virtual const std::map<unsigned long int, double> & cryomesh::dataobjects-  
::DataObjectController< unsigned long int, double >::getMap ( )  
[inline, virtual, inherited]`

Get all cycle values.

**Returns**

std::map<unsigned long int, double> & The cycle values

Definition at line 57 of file DataObjectController.h.

References cryomesh::dataobjects::DataObjectController< U, T >::dataObject, and  
cryomesh::dataobjects::DataObjectController< U, T >::refreshDataObject().

**6.31.4.29** `common::Connector< Node, Connection > &  
cryomesh::components::Node::getMutableConnector ( )`

Get the mutable Connector object for this [Node](#).

**Returns**

common::Connector<Node, Connection> & The mutable Connector for this object

Definition at line 88 of file Node.cpp.

References connector.

Referenced by connectInput(), connectOutput(), destroyAllInputConnections(), destroy-  
AllOutputConnections(), and emitImpulse().

**6.31.4.30** `boost::shared_ptr< Impulse > cryomesh::components::Node::getMutable-  
EmittedImpulse ( )`

Get the mutable [Impulse](#) that is emitted.

**Returns**

boost::shared\_ptr< Impulse > The mutable emitted [Impulse](#)

Definition at line 251 of file Node.cpp.

References emittedImpulse.

Referenced by emitImpulse().

#### 6.31.4.31 **ImpulseCollection & cryomesh::components::Node::getMutableImpulses ( )**

Get the mutable collection of Impulses for this [Node](#).

##### Returns

[ImpulseCollection](#) & The mutable collection of Impulses for this [Node](#)

Definition at line 255 of file Node.cpp.

References impulses.

Referenced by addImpulse(), and update().

#### 6.31.4.32 **const spacial::Point & cryomesh::components::Node::getPosition ( ) const**

get the position of the node

##### Returns

spacial::Point The spacial location of the node

Definition at line 305 of file Node.cpp.

References position.

Referenced by getDatabaseObject(), cryomesh::structures::Mesh::update(), and cryomesh::structures::Mesh::warp().

#### 6.31.4.33 **std::vector< boost::shared\_ptr< Connection > > cryomesh::components::Node::getPrimaryInputConnections ( )**

Definition at line 425 of file Node.cpp.

References getConnector().

#### 6.31.4.34 **std::vector< boost::shared\_ptr< Connection > > cryomesh::components::Node::getPrimaryOutputConnections ( )**

Definition at line 447 of file Node.cpp.

References getConnector().

#### 6.31.4.35 **boost::shared\_ptr< Node > cryomesh::components::Node::getRandom ( const spacial::Point & max\_point = MAX\_BOUNDING\_BOX\_POINT ) [static]**

Definition at line 26 of file Node.cpp.



Referenced by cryomesh::manipulators::ClusterArchitect::createRandomNodes(), and randomise().

**6.31.4.36** `bool cryomesh::components::Node::isActive ( const ActivationState state = None )`

Check if [Node](#) is currently activated.

#### Parameters

<i>Activation-State</i>	Positive for positive activity test, Negative for negative activity test, None for any activity test
-------------------------	--

#### Returns

bool True if activated, false otherwise

Definition at line 333 of file Node.cpp.

References getActivity(), Negative, None, and Positive.

**6.31.4.37** `bool cryomesh::components::Node::isInputIsolated ( ) const`

Definition at line 361 of file Node.cpp.

References getConnector().

Referenced by isPrimaryInputAttachedNode().

**6.31.4.38** `bool cryomesh::components::Node::isLive ( )`

Check if [Node](#) is live, ie active at any point in now or the future.

#### Returns

bool True if live, false otherwise

Definition at line 352 of file Node.cpp.

References getImpulses().

**6.31.4.39** `bool cryomesh::components::Node::isOutputIsolated ( ) const`

Definition at line 368 of file Node.cpp.

References getConnector().

Referenced by isPrimaryOutputAttachedNode().

**6.31.4.40** `bool cryomesh::components::Node::isPrimaryInputAttachedNode ( )`  
`const`

Definition at line 375 of file Node.cpp.

References `getConnector()`, and `isInputIsolated()`.

Referenced by `cryomesh::components::operator<<()`.

**6.31.4.41** `bool cryomesh::components::Node::isPrimaryOutputAttachedNode ( )`  
`const`

Definition at line 400 of file Node.cpp.

References `getConnector()`, and `isOutputIsolated()`.

Referenced by `cryomesh::components::operator<<()`.

**6.31.4.42** `bool cryomesh::components::Node::isTriggered ( ActivationState state = None )`

Check if [Node](#) is currently triggered.

#### Parameters

<i>Activation-State</i>	Positive for positive trigger test, Negative for negative trigger test, None for any trigger test
-------------------------	---

#### Returns

`bool` True if triggered, false otherwise

Definition at line 323 of file Node.cpp.

References `getLastActivationState()`, and `None`.

**6.31.4.43** `std::ostream & cryomesh::components::Node::printConnections (`  
`std::ostream & os, const std::map< boost::uuids::uuid, boost::shared_ptr<`  
`Connection > > & all_cons, const std::string formatter = " " ) const`

Definition at line 577 of file Node.cpp.

Referenced by `cryomesh::components::operator<<()`.

**6.31.4.44** `void cryomesh::components::Node::randomise ( )`

Randomise the nodes state.

Definition at line 318 of file Node.cpp.

References `activityThreshold`, `emittedImpulse`, `getRandom()`, `MAX_ACTIVITY_THRESHOLD`, and `MIN_ACTIVITY_THRESHOLD`.

6.31.4.45 `virtual void cryomesh::dataobjects::DataObjectController< unsigned long int, double >::refreshDataObject ( ) [inline, virtual, inherited]`

Function to allow refreshing implementation if required by subclasses.

Reimplemented in [cryomesh::components::ImpulseCollection](#).

Definition at line 76 of file DataObjectController.h.

6.31.4.46 `double cryomesh::components::Node::setActivity ( double activity )`

Set the current activity of the node.

#### Parameters

<i>double</i>	The current activity value to be set
---------------	--------------------------------------

#### Returns

*double* The activity set

Definition at line 283 of file Node.cpp.

References `cryomesh::common::TimeKeeper::getTimeKeeper()`.

Referenced by `updateActivity()`, and `cryomesh::structures::Mesh::warp()`.

6.31.4.47 `double cryomesh::components::Node::setActivity ( const common::Cycle & cycle, double activity )`

Set the activity at cycle of the node.

#### Parameters

<i>const</i>	Cycle & cycle The cycle the activity is on
<i>double</i>	The current activity value to be set

#### Returns

*double* The activity set

Definition at line 287 of file Node.cpp.

References `addActivity()`.

6.31.4.48 `void cryomesh::components::Node::setPosition ( const spacial::Point & new_position )`

Set the spacial position of the node, remembering to update connections lengths.

## Parameters

<i>spacial::Point</i>	The position to place this node at
-----------------------	------------------------------------

Definition at line 309 of file Node.cpp.

References position, and updatePosition().

#### 6.31.4.49 void cryomesh::components::Node::update ( ) [virtual]

Definition at line 49 of file Node.cpp.

References checkFire(), cryomesh::components::ImpulseCollection::clearActiveImpulses(), cryomesh::dataobjects::DataObjectController< unsigned long int, double >::dataObject, getActivities(), getMutableImpulses(), cryomesh::dataobjects::DataObject< U, T >::insert(), cryomesh::dataobjects::DataObject< U, T >::isLoggingEnabled(), lastActivationState, and updateImpulses().

#### 6.31.4.50 double cryomesh::components::Node::updateActivity ( )

Update and get the current activity of the node.

## Returns

double The current activity

Definition at line 275 of file Node.cpp.

References getActivity(), getImpulses(), and setActivity().

Referenced by checkActivationState().

#### 6.31.4.51 double cryomesh::components::Node::updateActivity ( const common::Cycle & cycle )

Update and get the activity of the node on specific cycle.

## Returns

double The current activity

Definition at line 279 of file Node.cpp.

References getActivity(), getImpulses(), and setActivity().

#### 6.31.4.52 void cryomesh::components::Node::updateImpulses ( ) [protected, virtual]

Update the collection of impulses by one cycle.

Definition at line 135 of file Node.cpp.

References cryomesh::components::ImpulseCollection::clearImpulses(), and impulses.

Referenced by update().

**6.31.4.53 void cryomesh::components::Node::updatePosition ( )**  
[protected, virtual]

Recalculate state of node and connections based on current position.

Definition at line 515 of file Node.cpp.

References connector.

Referenced by setPosition().

### 6.31.5 Friends And Related Function Documentation

**6.31.5.1 std::ostream& operator<< ( std::ostream & os, const Node & obj )** [friend]

To stream operator.

#### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Node</a> & obj The object to stream

#### Returns

std::ostream & The output stream

Definition at line 548 of file Node.cpp.

### 6.31.6 Member Data Documentation

**6.31.6.1 dataobjects::DataObject<common::Cycle, double>**  
**cryomesh::components::Node::activities** [private]

Definition at line 461 of file Node.h.

Referenced by addActivity(), getActivities(), and Node().

**6.31.6.2 double cryomesh::components::Node::activityThreshold** [private]

Definition at line 433 of file Node.h.

Referenced by getActivityThreshold(), and randomise().

**6.31.6.3** `boost::shared_ptr<common::Connector<Node, Connection>>`  
`cryomesh::components::Node::connector` [private]

Definition at line 440 of file Node.h.

Referenced by `getConnector()`, `getMutableConnector()`, `Node()`, and `updatePosition()`.

**6.31.6.4** `dataobjects::DataObject<unsigned long int, double>`  
`cryomesh::dataobjects::DataObjectController< unsigned long int, double`  
`>::dataObject` [protected, inherited]

Definition at line 85 of file DataObjectController.h.

Referenced by `cryomesh::components::ImpulseCollection::refreshDataObject()`, and `update()`.

**6.31.6.5** `boost::shared_ptr<Impulse>` `cryomesh::components::Node::emitted-`  
`Impulse` [private]

Definition at line 454 of file Node.h.

Referenced by `getEmittedImpulse()`, `getMutableEmittedImpulse()`, `Node()`, and `randomise()`.

**6.31.6.6** `ImpulseCollection` `cryomesh::components::Node::impulses`  
[private]

Definition at line 447 of file Node.h.

Referenced by `addImpulses()`, `enterRecovery()`, `getImpulses()`, `getMutableImpulses()`, and `updateImpulses()`.

**6.31.6.7** `ActivationState` `cryomesh::components::Node::lastActivationState`  
[private]

Definition at line 475 of file Node.h.

Referenced by `getLastActivationState()`, and `update()`.

**6.31.6.8** `const int` `cryomesh::components::Node::MAX_ACTIVITIES_LENGTH` = 10  
[static]

Definition at line 341 of file Node.h.

Referenced by `Node()`.

6.31.6.9 `const double cryomesh::components::Node::MAX_ACTIVITY_THRESHOLD = 3 * Impulse::MAX_ACTIVITY`  
`[static]`

Definition at line 348 of file Node.h.

Referenced by `randomise()`.

6.31.6.10 `const spacial::Point cryomesh::components::Node::MAX_BOUNDING_BOX_POINT = spacial::Point(100, 100, 100)`  
`[static]`

Definition at line 362 of file Node.h.

6.31.6.11 `const double cryomesh::components::Node::MIN_ACTIVITY_THRESHOLD = 1 * Impulse::MAX_ACTIVITY`  
`[static]`

Definition at line 355 of file Node.h.

Referenced by `randomise()`.

6.31.6.12 `spacial::Point cryomesh::components::Node::position` `[private]`

Definition at line 468 of file Node.h.

Referenced by `getPosition()`, and `setPosition()`.

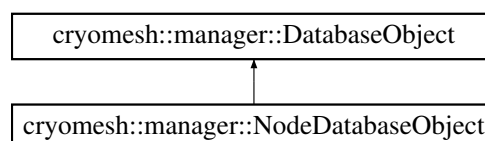
The documentation for this class was generated from the following files:

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Node.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Node.cpp](#)

## 6.32 cryomesh::manager::NodeDatabaseObject Class Reference

```
#include <NodeDatabaseObject.h>
```

Inheritance diagram for `cryomesh::manager::NodeDatabaseObject`:



## Public Member Functions

- [NodeDatabaseObject](#) (std::string uuid\_str, spacial::Point pt, [common::Cycle](#) cyc, double act)  
*Create object from node variables.*
- [NodeDatabaseObject](#) (const std::string &node\_table\_entry)  
*Create object from the string of entries in the database node table.*
- virtual [~NodeDatabaseObject](#) ()  
*Default destructor.*
- virtual std::string [getInsert](#) (const std::string &table) const  
*Get the string that can be used to insert the sql data.*
- std::string [getUUID](#) () const  
*Get uuid variable.*
- const spacial::Point & [getPoint](#) () const  
*Get point variable.*
- const [common::Cycle](#) & [getCycle](#) () const  
*Get cycle variable.*
- double [getActivity](#) () const  
*Get activity variable.*
- std::string [getKey](#) (const std::string &key) const  
*Return the string object associated with a key.*

## Static Public Member Functions

- static std::string [findValue](#) (const std::string &entry, const std::map< std::string, std::string > &map)  
*Find entries value in map or return null.*
- static std::map< std::string, std::string > [getColumnMapFromEntry](#) (const std::string &entry)  
*Parse a string database entry, extract columns and values and return a map.*
- template<class T >  
static std::string [toString](#) (T obj)  
*Convert an templated object that can be piped to a stream to a string.*

## Static Public Attributes

- static const std::string [ID\\_TAG](#) = "id"
- static const std::string [X\\_TAG](#) = "x"
- static const std::string [Y\\_TAG](#) = "y"
- static const std::string [Z\\_TAG](#) = "z"
- static const std::string [ACTIVITY\\_TAG](#) = "activity"
- static const std::string [CYCLE\\_TAG](#) = "cycle"



### Protected Attributes

- `std::map< std::string, std::string >` [columns](#)

### Private Attributes

- `std::string` [uuid](#)
- `spacial::Point` [point](#)
- `common::Cycle` [cycle](#)
- `double` [activity](#)

#### 6.32.1 Detailed Description

Definition at line 20 of file NodeDatabaseObject.h.

#### 6.32.2 Constructor & Destructor Documentation

**6.32.2.1 cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject (**  
`std::string uuid_str, spacial::Point pt, common::Cycle cyc, double act )`

Create object from node variables.

##### Parameters

<i>std::string</i>	The uuid string of the node
<i>spacial::Point</i>	Location of the node
<i>Cycle</i>	The cycle of the entry
<i>double</i>	The activity of the node on the cycle

Definition at line 13 of file NodeDatabaseObject.cpp.

References [activity](#), [ACTIVITY\\_TAG](#), [cryomesh::manager::DatabaseObject::columns](#), [cycle](#), [CYCLE\\_TAG](#), [ID\\_TAG](#), [point](#), [cryomesh::common::Cycle::toULInt\(\)](#), [X\\_TAG](#), [Y\\_TAG](#), and [Z\\_TAG](#).

**6.32.2.2 cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject (** `const std::string & node_table_entry )`

Create object from the string of entries in the database node table.

##### Parameters

<i>std::string</i>	The string of data taken from a node entry in the database node table
--------------------	---

Definition at line 23 of file NodeDatabaseObject.cpp.

References activity, cycle, cryomesh::manager::DatabaseObject::findValue(), cryomesh::manager::DatabaseObject::getColumnMapFromEntry(), point, and uuid.

#### 6.32.2.3 cryomesh::manager::NodeDatabaseObject::~~NodeDatabaseObject ( ) [virtual]

Default destructor.

Definition at line 60 of file NodeDatabaseObject.cpp.

### 6.32.3 Member Function Documentation

#### 6.32.3.1 static std::string cryomesh::manager::DatabaseObject::findValue ( const std::string & entry, const std::map< std::string, std::string > & map ) [inline, static, inherited]

Find entries value in map or return null.

##### Parameters

<i>std::string</i>	Entry to find
<i>std- ::map&lt; std- ::string, std- ::string</i>	map to search

##### Returns

Value of entry

Definition at line 59 of file DatabaseObject.h.

Referenced by cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject(), NodeDatabaseObject(), and cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject().

#### 6.32.3.2 double cryomesh::manager::NodeDatabaseObject::getActivity ( ) const

Get activity variable.

##### Returns

double The activity variable

Definition at line 83 of file NodeDatabaseObject.cpp.

References activity.

6.32.3.3 `static std::map<std::string, std::string> cryomesh::manager::DatabaseObject::getColumnMapFromEntry ( const std::string & entry ) [inline, static, inherited]`

Parse a string database entry, extract columns and values and return a map.

Definition at line 72 of file DatabaseObject.h.

Referenced by `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject()`, `NodeDatabaseObject()`, and `cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject()`.

6.32.3.4 `const common::Cycle & cryomesh::manager::NodeDatabaseObject::getCycle ( ) const`

Get cycle variable.

#### Returns

`common::Cycle` The cycle variable

Definition at line 80 of file NodeDatabaseObject.cpp.

References `cycle`.

6.32.3.5 `std::string cryomesh::manager::NodeDatabaseObject::getInsert ( const std::string & table ) const [virtual]`

Get the string that can be used to insert the sql data.

#### Returns

the sql command string to insert into this table

Implements `cryomesh::manager::DatabaseObject`.

Definition at line 63 of file NodeDatabaseObject.cpp.

References `ACTIVITY_TAG`, `CYCLE_TAG`, `cryomesh::manager::DatabaseObject::getKey()`, `ID_TAG`, `X_TAG`, `Y_TAG`, and `Z_TAG`.

6.32.3.6 `std::string cryomesh::manager::DatabaseObject::getKey ( const std::string & key ) const [inline, inherited]`

Return the string object associated with a key.

`::string` The key to search for

**Returns**

std::string The object associated with the search key, "" if not found

Definition at line 37 of file DatabaseObject.h.

References cryomesh::manager::DatabaseObject::columns.

Referenced by cryomesh::manager::PatternDatabaseObject::getInsert(), getInsert(), and cryomesh::manager::ConnectionDatabaseObject::getInsert().

### 6.32.3.7 const spacial::Point & cryomesh::manager::NodeDatabaseObject::getPoint ( ) const

Get point variable.

**Returns**

spacial::Point The point variable

Definition at line 77 of file NodeDatabaseObject.cpp.

References point.

### 6.32.3.8 std::string cryomesh::manager::NodeDatabaseObject::getUUID ( ) const

Get uuid variable.

**Returns**

std::string The uuid variable

Definition at line 74 of file NodeDatabaseObject.cpp.

References uuid.

### 6.32.3.9 template<class T > static std::string cryomesh::manager::DatabaseObject::toString ( T obj ) [inline, static, inherited]

Convert an templated object that can be piped to a stream to a string.

**Parameters**

<i>T</i>	The object to get a string for
----------	--------------------------------

Definition at line 108 of file DatabaseObject.h.

## 6.32.4 Member Data Documentation

**6.32.4.1 double cryomesh::manager::NodeDatabaseObject::activity** [private]

Definition at line 158 of file NodeDatabaseObject.h.

Referenced by getActivity(), and NodeDatabaseObject().

**6.32.4.2 const std::string cryomesh::manager::NodeDatabaseObject::ACTIVITY\_TAG = "activity"** [static]

Definition at line 122 of file NodeDatabaseObject.h.

Referenced by getInsert(), and NodeDatabaseObject().

**6.32.4.3 std::map<std::string, std::string> cryomesh::manager::DatabaseObject::columns** [protected, inherited]

Definition at line 119 of file DatabaseObject.h.

Referenced by cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject(), cryomesh::manager::DatabaseObject::getKey(), NodeDatabaseObject(), and cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject().

**6.32.4.4 common::Cycle cryomesh::manager::NodeDatabaseObject::cycle** [private]

Definition at line 151 of file NodeDatabaseObject.h.

Referenced by getCycle(), and NodeDatabaseObject().

**6.32.4.5 const std::string cryomesh::manager::NodeDatabaseObject::CYCLE\_TAG = "cycle"** [static]

Definition at line 129 of file NodeDatabaseObject.h.

Referenced by getInsert(), and NodeDatabaseObject().

**6.32.4.6 const std::string cryomesh::manager::NodeDatabaseObject::ID\_TAG = "id"** [static]

Definition at line 94 of file NodeDatabaseObject.h.

Referenced by getInsert(), and NodeDatabaseObject().

**6.32.4.7 spacial::Point cryomesh::manager::NodeDatabaseObject::point** [private]

Definition at line 144 of file NodeDatabaseObject.h.

Referenced by getPoint(), and NodeDatabaseObject().

#### 6.32.4.8 `std::string cryomesh::manager::NodeDatabaseObject::uuid` [private]

Definition at line 137 of file `NodeDatabaseObject.h`.

Referenced by `getUUID()`, and `NodeDatabaseObject()`.

#### 6.32.4.9 `const std::string cryomesh::manager::NodeDatabaseObject::X_TAG = "x"` [static]

Definition at line 101 of file `NodeDatabaseObject.h`.

Referenced by `getInsert()`, and `NodeDatabaseObject()`.

#### 6.32.4.10 `const std::string cryomesh::manager::NodeDatabaseObject::Y_TAG = "y"` [static]

Definition at line 108 of file `NodeDatabaseObject.h`.

Referenced by `getInsert()`, and `NodeDatabaseObject()`.

#### 6.32.4.11 `const std::string cryomesh::manager::NodeDatabaseObject::Z_TAG = "z"` [static]

Definition at line 115 of file `NodeDatabaseObject.h`.

Referenced by `getInsert()`, and `NodeDatabaseObject()`.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/NodeDatabaseObject.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/NodeDatabaseObject.cpp](#)

### 6.33 `cryomesh::utilities::SequencerGeneric::NodeEntry` Struct - Reference

```
#include <SequencerGeneric.h>
```

#### Public Member Functions

- [NodeEntry](#) ()

#### Public Attributes

- `std::string` [name](#)

- `std::map< std::string, std::string >` [info](#)
- `boost::shared_ptr< NodeEntry >` [parentNode](#)
- `std::list< boost::shared_ptr < NodeEntry > >` [childNodes](#)

## Friends

- `std::ostream & operator<< (std::ostream &os, const NodeEntry &obj)`

### 6.33.1 Detailed Description

Definition at line 14 of file SequencerGeneric.h.

### 6.33.2 Constructor & Destructor Documentation

6.33.2.1 `cryomesh::utilities::SequencerGeneric::NodeEntry::NodeEntry ( )`  
`[inline]`

Definition at line 16 of file SequencerGeneric.h.

### 6.33.3 Friends And Related Function Documentation

6.33.3.1 `std::ostream& operator<< ( std::ostream &os, const NodeEntry &obj )`  
`[friend]`

Definition at line 23 of file SequencerGeneric.h.

### 6.33.4 Member Data Documentation

6.33.4.1 `std::list<boost::shared_ptr<NodeEntry> >` `cryomesh::utilities::SequencerGeneric::NodeEntry::childNodes`

Definition at line 21 of file SequencerGeneric.h.

Referenced by `cryomesh::utilities::SequencerChannels::readSequences()`.

6.33.4.2 `std::map<std::string, std::string>` `cryomesh::utilities::SequencerGeneric::NodeEntry::info`

Definition at line 19 of file SequencerGeneric.h.

Referenced by `cryomesh::utilities::SequencerChannels::readSequences()`.

#### 6.33.4.3 `std::string cryomesh::utilities::SequencerGeneric::NodeEntry::name`

Definition at line 17 of file SequencerGeneric.h.

Referenced by `cryomesh::utilities::SequencerChannels::readSequences()`.

#### 6.33.4.4 `boost::shared_ptr<NodeEntry> cryomesh::utilities::SequencerGeneric::NodeEntry::parentNode`

Definition at line 20 of file SequencerGeneric.h.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/SequencerGeneric.h`

## 6.34 `cryomesh::components::NodeMap` Class Reference

Helper class for `NodeMap` to `KeyMappedCollection` mapping.

```
#include <NodeMap.h>
```

### Public Member Functions

- `NodeMap ()`  
*Default constructor.*
- `virtual ~NodeMap ()`  
*Default destructor.*
- `void update ()`  
*Update all entries in the nodemap.*
- `const std::map < boost::uuids::uuid, boost::shared_ptr< Node > > getAllPrimaryInputNodes () const`
- `const std::map < boost::uuids::uuid, boost::shared_ptr< Node > > getAllPrimaryOutputNodes () const`
- `void addRandomImpulses (double positive_bias=0.5)`
- `const std::map < boost::uuids::uuid, boost::shared_ptr< Connection > > getAllInputConnections () const`
- `const std::map < boost::uuids::uuid, boost::shared_ptr< Connection > > getAllOutputConnections () const`
- `const std::map < boost::uuids::uuid, boost::shared_ptr< Connection > > getAllConnections () const`

### Friends

- `std::ostream & operator<< (std::ostream &os, const NodeMap &obj)`  
*To stream operator.*



### 6.34.1 Detailed Description

Helper class for [NodeMap](#) to KeyMappedCollection mapping.

Definition at line 23 of file NodeMap.h.

### 6.34.2 Constructor & Destructor Documentation

#### 6.34.2.1 cryomesh::components::NodeMap::NodeMap ( ) [inline]

Default constructor.

Definition at line 28 of file NodeMap.h.

#### 6.34.2.2 virtual cryomesh::components::NodeMap::~NodeMap ( ) [inline, virtual]

Default destructor.

Definition at line 35 of file NodeMap.h.

### 6.34.3 Member Function Documentation

#### 6.34.3.1 void cryomesh::components::NodeMap::addRandomImpulses ( double *positive\_bias* = 0.5 ) [inline]

Definition at line 95 of file NodeMap.h.

References cryomesh::components::Impulse::getRandom().

#### 6.34.3.2 const std::map<boost::uuids::uuid, boost::shared\_ptr<Connection> > cryomesh::components::NodeMap::getAllConnections ( ) const [inline]

Definition at line 149 of file NodeMap.h.

#### 6.34.3.3 const std::map<boost::uuids::uuid, boost::shared\_ptr<Connection> > cryomesh::components::NodeMap::getAllInputConnections ( ) const [inline]

Definition at line 109 of file NodeMap.h.

#### 6.34.3.4 const std::map<boost::uuids::uuid, boost::shared\_ptr<Connection> > cryomesh::components::NodeMap::getAllOutputConnections ( ) const [inline]

Definition at line 129 of file NodeMap.h.

```
6.34.3.5  const std::map<boost::uuids::uuid, boost::shared_ptr<Node> >
          cryomesh::components::NodeMap::getAllPrimaryInputNodes ( ) const
          [inline]
```

Definition at line 53 of file NodeMap.h.

Referenced by cryomesh::manipulators::ClusterArchitect::getRandomNodes().

```
6.34.3.6  const std::map<boost::uuids::uuid, boost::shared_ptr<Node> >
          cryomesh::components::NodeMap::getAllPrimaryOutputNodes ( ) const
          [inline]
```

Definition at line 74 of file NodeMap.h.

Referenced by cryomesh::manipulators::ClusterArchitect::getRandomNodes().

```
6.34.3.7  void cryomesh::components::NodeMap::update ( ) [inline]
```

Update all entries in the nodemap.

Definition at line 41 of file NodeMap.h.

Referenced by cryomesh::structures::Cluster::update().

## 6.34.4 Friends And Related Function Documentation

```
6.34.4.1  std::ostream& operator<< ( std::ostream & os, const NodeMap & obj )
          [friend]
```

To stream operator.

### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">NodeMap</a> & obj The object to stream

### Returns

std::ostream & The output stream

Definition at line 181 of file NodeMap.h.

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

- /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/[NodeMap.h](#)

## 6.35 cryomesh::structures::NodeMesh Class Reference

[Mesh](#) of nodes and their neighbouring nodes and distances.

```
#include <NodeMesh.h>
```

## Classes

- struct [NeighbourhoodRanges](#)  
*Struct to capture some statistics data on a nodes neighbourhood.*

## Public Types

- enum [InterpolationStyle](#) { [INVERSE\\_R](#), [INVERSE\\_R2](#) }  
*Stype to use when interpolating values.*

## Public Member Functions

- [NodeMesh](#) ([Cluster](#) &clus)  
*Constructor to create a node mesh from a cluster.*
- [NodeMesh](#) ([Cluster](#) &clus, double max\_radius)  
*Constructor to create a node mesh from at minimum, a cluster and an optional maximum neighbour radius.*
- virtual [~NodeMesh](#) ()  
*Default destructor.*
- void [update](#) ()  
*Per cycle update calls.*
- void [warpNodes](#) ()  
*Merge the interpolated activities with the actual node activities.*
- void [regenerateNeighbourhoods](#) ()  
*Regenerate the neighbourhood nodes and distances.*
- void [regenerateActivities](#) ()  
*Recalculate the applied interpolated activity at all nodes from their neighbours.*
- [NeighbourhoodRanges](#) [getNeighbourRanges](#) () const  
*Return a pair of pairs.*
- const [NeighbourhoodMap](#) & [getNodeNeighbourhoodMap](#) () const  
*Get the neighbourhood map.*
- const std::map < boost::shared\_ptr < [components::Node](#) >, double > & [getNeighbourhoodActivities](#) () const  
*Get the neighbourhood activities.*
- std::ostream & [printNeighbourhoods](#) (std::ostream &os) const  
*Print the neighbourhood map.*
- std::ostream & [printNeighbourhoodActivities](#) (std::ostream &os) const  
*Print the neighbourhood activities.*

### Protected Member Functions

- double [getInterpolatedActivity](#) (const std::map< boost::shared\_ptr< [cryomesh::components::Node](#) >, double > &all\_neighbours, const [InterpolationStyle](#) style=[INVERSE\\_R](#)) const  
*Use a list of neighbours and their distances to generated an interpolated activity value at the central node.*
- double [getDecayRate](#) () const  
*Get the spacial decay rate for activities.*

### Static Protected Attributes

- static const double [MAX\\_RADIUS\\_FRACTION\\_OF\\_BOUNDING\\_BOX](#) = (1.0 / 100.0)
- static const double [INTERPOLATED\\_ACTIVITY\\_SCALING\\_FACTOR](#) = (1.0 / 100.0)

### Private Attributes

- [NeighbourhoodMap](#) nodeNeighbourhoodMap
- std::map< boost::shared\_ptr < [components::Node](#) >, double > [neighbourhoodActivities](#)
- [Cluster](#) & cluster
- double [maximumNeighbourhoodRadius](#)
- const double [decayRate](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [NodeMesh](#) &obj)  
*To stream operator.*

## 6.35.1 Detailed Description

[Mesh](#) of nodes and their neighbouring nodes and distances.

Definition at line 34 of file NodeMesh.h.

## 6.35.2 Member Enumeration Documentation

### 6.35.2.1 enum cryomesh::structures::NodeMesh::InterpolationStyle

Stype to use when interpolating values.

Enumerator:

**[INVERSE\\_R](#)**

***INVERSE\_R2***

Definition at line 50 of file NodeMesh.h.

**6.35.3 Constructor & Destructor Documentation****6.35.3.1 cryomesh::structures::NodeMesh::NodeMesh ( Cluster & *clus* )**

Constructor to create a node mesh from a cluster.

**Parameters**

<a href="#"><i>Cluster</i></a>	The cluster associated with this node mesh
--------------------------------	--

Definition at line 22 of file NodeMesh.cpp.

References cluster, cryomesh::structures::Cluster::getNodes(), maximumNeighbourhoodRadius, and regenerateNeighbourhoods().

**6.35.3.2 cryomesh::structures::NodeMesh::NodeMesh ( Cluster & *clus*, double *max\_radius* )**

Constructor to create a node mesh from at minimum, a cluster and an optional maximum neighbour radius.

**Parameters**

<a href="#"><i>Cluster</i></a>	The cluster associated with this node mesh
<i>double</i>	Maximum radius cutoff point for neighbourhood distance

Definition at line 43 of file NodeMesh.cpp.

References regenerateNeighbourhoods().

**6.35.3.3 cryomesh::structures::NodeMesh::~~NodeMesh ( ) [virtual]**

Default destructor.

Definition at line 48 of file NodeMesh.cpp.

**6.35.4 Member Function Documentation****6.35.4.1 double cryomesh::structures::NodeMesh::getDecayRate ( ) const [protected]**

Get the spacial decay rate for activities.

**Returns**

double The spacial decay rate

Definition at line 216 of file NodeMesh.cpp.

References decayRate.

Referenced by getInterpolatedActivity().

```
6.35.4.2 double cryomesh::structures::NodeMesh::getInterpolatedActivity ( const
std::map< boost::shared_ptr< cryomesh::components::Node >, double
> & all_neighbours, const InterpolationStyle style = INVERSE_R ) const
[protected]
```

Use a list of neighbours and their distances to generated an interpolated activity value at the central node.

**Parameters**

<code>std::map&lt; boost::shared_ptr&lt; cryomesh::components::Node&gt;, double&gt;</code>	List of all the neighbour nodes and their distances to their central node
<code>InterpolationStyle</code>	Which method to use to interpolate the central activity

**Returns**

double The interpolated activity

Definition at line 164 of file NodeMesh.cpp.

References getDecayRate().

Referenced by regenerateActivities().

```
6.35.4.3 const std::map< boost::shared_ptr< components::Node >, double > &
cryomesh::structures::NodeMesh::getNeighbourhoodActivities ( ) const
```

Get the neighbourhood activities.

**Returns**

`std::map< boost::shared_ptr< components::Node>, double>` The neighbourhood activities

Definition at line 261 of file NodeMesh.cpp.

References neighbourhoodActivities.

**6.35.4.4 NodeMesh::NeighbourhoodRanges** cryomesh-  
::structures::NodeMesh::getNeighbourRanges ( )  
const

Return a pair of pairs.

the first representing the min/max of neighbour counts, the second the min/max of distances

Returns

[NeighbourhoodRanges](#) min/max of neighbour counts and min/max of distances

Definition at line 220 of file NodeMesh.cpp.

References cryomesh::structures::NodeMesh::NeighbourhoodRanges::maximumNeighbourCount, cryomesh::structures::NodeMesh::NeighbourhoodRanges::maximumNeighbourDistance, cryomesh::structures::NodeMesh::NeighbourhoodRanges::minimumNeighbourCount, cryomesh::structures::NodeMesh::NeighbourhoodRanges::minimumNeighbourDistance, and nodeNeighbourhoodMap.

**6.35.4.5 const NeighbourhoodMap & cryomesh::structures-  
::NodeMesh::getNodeNeighbourhoodMap ( )**  
const

Get the neighbourhood map.

Returns

NeighbourhoodMap The neighbourhood map

Definition at line 258 of file NodeMesh.cpp.

References nodeNeighbourhoodMap.

**6.35.4.6 std::ostream & cryomesh::structures::NodeMesh-  
::printNeighbourhoodActivities ( std::ostream & os )**  
const

Print the neighbourhood activities.

Parameters

<i>std::ostream</i>	The output stream
---------------------	-------------------

**Returns**

`std::ostream` The output stream

Definition at line 287 of file NodeMesh.cpp.

References `neighbourhoodActivities`.

Referenced by `cryomesh::structures::operator<<()`, `regenerateNeighbourhoods()`, and `update()`.

#### 6.35.4.7 `std::ostream & cryomesh::structures::NodeMesh::printNeighbourhoods ( std::ostream & os ) const`

Print the neighbourhood map.

**Parameters**

<code>std::ostream</code>	The output stream
---------------------------	-------------------

**Returns**

`std::ostream` The output stream

Definition at line 265 of file NodeMesh.cpp.

References `nodeNeighbourhoodMap`.

Referenced by `cryomesh::structures::operator<<()`, `regenerateNeighbourhoods()`, and `update()`.

#### 6.35.4.8 `void cryomesh::structures::NodeMesh::regenerateActivities ( )`

Recalculate the applied interpolated activity at all nodes from their neighbours.

Definition at line 146 of file NodeMesh.cpp.

References `getInterpolatedActivity()`, `neighbourhoodActivities`, and `nodeNeighbourhoodMap`.

Referenced by `update()`.

#### 6.35.4.9 `void cryomesh::structures::NodeMesh::regenerateNeighbourhoods ( )`

Regenerate the neighbourhood nodes and distances.

Definition at line 99 of file NodeMesh.cpp.

References `cluster`, `cryomesh::structures::Cluster::getNodes()`, `maximumNeighbourhoodRadius`, `neighbourhoodActivities`, `nodeNeighbourhoodMap`, `printNeighbourhoodActivities()`, and `printNeighbourhoods()`.

Referenced by `NodeMesh()`.



**6.35.4.10 void cryomesh::structures::NodeMesh::update ( )**

Per cycle update calls.

Definition at line 51 of file NodeMesh.cpp.

References `printNeighbourhoodActivities()`, `printNeighbourhoods()`, and `regenerateActivities()`.

**6.35.4.11 void cryomesh::structures::NodeMesh::warpNodes ( )**

Merge the interpolated activities with the actual node activities.

Definition at line 65 of file NodeMesh.cpp.

References `cluster`, `cryomesh::structures::Cluster::getEnergy()`, and `neighbourhoodActivities`.

**6.35.5 Friends And Related Function Documentation****6.35.5.1 std::ostream& operator<< ( std::ostream & os, const NodeMesh & obj )**  
[friend]

To stream operator.

**Parameters**

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">NodeMesh</a> & obj The object to stream

**Returns**

std::ostream & The output stream

Definition at line 306 of file NodeMesh.cpp.

**6.35.6 Member Data Documentation****6.35.6.1 Cluster& cryomesh::structures::NodeMesh::cluster** [private]

Definition at line 219 of file NodeMesh.h.

Referenced by `NodeMesh()`, `regenerateNeighbourhoods()`, and `warpNodes()`.

**6.35.6.2 const double cryomesh::structures::NodeMesh::decayRate** [private]

Definition at line 233 of file NodeMesh.h.

Referenced by `getDecayRate()`.

6.35.6.3 `const double cryomesh::structures::NodeMesh::INTERPOLATED_ACTIVITY_SCALING_FACTOR = (1.0 / 100.0) [static, protected]`

Definition at line 196 of file NodeMesh.h.

6.35.6.4 `const double cryomesh::structures::NodeMesh::MAX_RADIUS_FRACTION_OF_BOUNDING_BOX = (1.0 / 100.0) [static, protected]`

Definition at line 189 of file NodeMesh.h.

6.35.6.5 `double cryomesh::structures::NodeMesh::maximumNeighbourhoodRadius [private]`

Definition at line 226 of file NodeMesh.h.

Referenced by NodeMesh(), and regenerateNeighbourhoods().

6.35.6.6 `std::map<boost::shared_ptr<components::Node>, double> cryomesh::structures::NodeMesh::neighbourhoodActivities [private]`

Definition at line 212 of file NodeMesh.h.

Referenced by getNeighbourhoodActivities(), printNeighbourhoodActivities(), regenerateActivities(), regenerateNeighbourhoods(), and warpNodes().

6.35.6.7 `NeighbourhoodMap cryomesh::structures::NodeMesh::nodeNeighbourhoodMap [private]`

Definition at line 205 of file NodeMesh.h.

Referenced by getNeighbourRanges(), getNodeNeighbourhoodMap(), printNeighbourhoods(), regenerateActivities(), and regenerateNeighbourhoods().

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

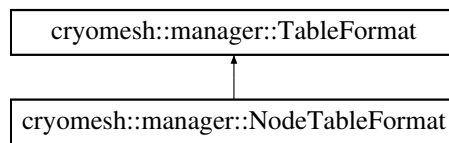
- /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/[NodeMesh.h](#)
- /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/[NodeMesh.cpp](#)

## 6.36 cryomesh::manager::NodeTableFormat Struct Reference

Struct representing a node table structure.

```
#include <TableFormats.h>
```

Inheritance diagram for cryomesh::manager::NodeTableFormat:



### Public Member Functions

- [NodeTableFormat](#) ()  
*Default constructor will construct all the names and columns associated with a node table.*
- std::string [getName](#) () const  
*Return the name of the table.*
- std::string [getKey](#) (const std::string &key)  
*Return the string object associated with a key.*
- std::string [getCreateTable](#) () const  
*Get the string that can be used to create the sql table.*

### Protected Attributes

- std::string [name](#)
- std::map< std::string, std::string > [columns](#)

#### 6.36.1 Detailed Description

Struct representing a node table structure.

Definition at line 99 of file TableFormats.h.

#### 6.36.2 Constructor & Destructor Documentation

##### 6.36.2.1 cryomesh::manager::NodeTableFormat::NodeTableFormat ( ) [inline]

Default constructor will construct all the names and columns associated with a node table.

Definition at line 104 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`, and `cryomesh::manager::TableFormat::name`.

### 6.36.3 Member Function Documentation

**6.36.3.1** `std::string cryomesh::manager::TableFormat::getCreateTable ( ) const`  
[inline, inherited]

Get the string that can be used to create the sql table.

#### Returns

the sql command string to create this table

Definition at line 60 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`, and `cryomesh::manager::TableFormat::getName()`.

**6.36.3.2** `std::string cryomesh::manager::TableFormat::getKey ( const std::string & key )`  
[inline, inherited]

Return the string object associated with a key.

`::string` The key to search for

#### Returns

`std::string` The object associated with the search key, "" if not found

Definition at line 45 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`.

**6.36.3.3** `std::string cryomesh::manager::TableFormat::getName ( ) const`  
[inline, inherited]

Return the name of the table.

#### Returns

`std::string` The name of the table

Definition at line 32 of file TableFormats.h.

References `cryomesh::manager::TableFormat::name`.

Referenced by `cryomesh::manager::TableFormat::getCreateTable()`, `cryomesh::manager::DatabaseManager::insertConnection()`, `cryomesh::manager::DatabaseManager::insertNode()`, and `cryomesh::manager::DatabaseManager::insertOutputPattern()`.

### 6.36.4 Member Data Documentation

6.36.4.1 `std::map<std::string, std::string> cryomesh::manager::TableFormat::columns` [protected, inherited]

Definition at line 93 of file TableFormats.h.

Referenced by `cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat()`, `cryomesh::manager::TableFormat::getCreateTable()`, `cryomesh::manager::TableFormat::getKey()`, `cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat()`, `NodeTableFormat()`, and `cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat()`.

6.36.4.2 `std::string cryomesh::manager::TableFormat::name` [protected, inherited]

Definition at line 86 of file TableFormats.h.

Referenced by `cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat()`, `cryomesh::manager::TableFormat::getName()`, `cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat()`, `NodeTableFormat()`, and `cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat()`.

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

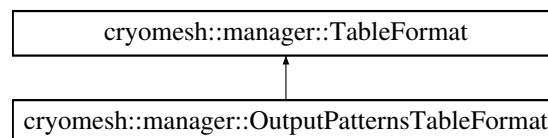
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/TableFormats.h](#)

## 6.37 cryomesh::manager::OutputPatternsTableFormat Struct - Reference

Struct representing output pattern table structure.

```
#include <TableFormats.h>
```

Inheritance diagram for `cryomesh::manager::OutputPatternsTableFormat`:



### Public Member Functions

- [OutputPatternsTableFormat \(\)](#)

*Default constructor will construct all the names and columns associated with a pattern table.*

- `std::string getName () const`  
*Return the name of the table.*
- `std::string getKey (const std::string &key)`  
*Return the string object associated with a key.*
- `std::string getCreateTable () const`  
*Get the string that can be used to create the sql table.*

### Protected Attributes

- `std::string name`
- `std::map< std::string, std::string > columns`

### 6.37.1 Detailed Description

Struct representing output pattern table structure.

Definition at line 152 of file TableFormats.h.

### 6.37.2 Constructor & Destructor Documentation

#### 6.37.2.1 `cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat ( ) [inline]`

Default constructor will construct all the names and columns associated with a pattern table.

Definition at line 157 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`, and `cryomesh::manager::TableFormat::name`.

### 6.37.3 Member Function Documentation

#### 6.37.3.1 `std::string cryomesh::manager::TableFormat::getCreateTable ( ) const [inline, inherited]`

Get the string that can be used to create the sql table.

#### Returns

the sql command string to create this table

Definition at line 60 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`, and `cryomesh::manager::TableFormat::getName()`.

**6.37.3.2** `std::string cryomesh::manager::TableFormat::getKey ( const std::string & key ) [inline, inherited]`

Return the string object associated with a key.

`::string` The key to search for

#### Returns

`std::string` The object associated with the search key, "" if not found

Definition at line 45 of file TableFormats.h.

References `cryomesh::manager::TableFormat::columns`.

**6.37.3.3** `std::string cryomesh::manager::TableFormat::getName ( ) const [inline, inherited]`

Return the name of the table.

#### Returns

`std::string` The name of the table

Definition at line 32 of file TableFormats.h.

References `cryomesh::manager::TableFormat::name`.

Referenced by `cryomesh::manager::TableFormat::getCreateTable()`, `cryomesh::manager::DatabaseManager::insertConnection()`, `cryomesh::manager::DatabaseManager::insertNode()`, and `cryomesh::manager::DatabaseManager::insertOutputPattern()`.

## 6.37.4 Member Data Documentation

**6.37.4.1** `std::map<std::string, std::string> cryomesh::manager::TableFormat::columns [protected, inherited]`

Definition at line 93 of file TableFormats.h.

Referenced by `cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat()`, `cryomesh::manager::TableFormat::getCreateTable()`, `cryomesh::manager::TableFormat::getKey()`, `cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat()`, `cryomesh::manager::NodeTableFormat::NodeTableFormat()`, and `cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat()`.

**6.37.4.2** `std::string cryomesh::manager::TableFormat::name [protected, inherited]`

Definition at line 86 of file TableFormats.h.

Referenced by `cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat()`, `cryomesh::manager::TableFormat::getName()`, `cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat()`, `cryomesh::manager::NodeTableFormat::NodeTableFormat()`, and `OutputPatternsTableFormat()`.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/TableFormats.h](#)

## 6.38 cryomesh::state::Pattern Class Reference

```
#include <Pattern.h>
```

### Public Member Functions

- [Pattern](#) ()
- [Pattern](#) (const [Pattern](#) &)
- [Pattern](#) (const std::string &)
- [Pattern](#) (const std::vector< bool > &)
- [Pattern](#) (const [BinaryString](#) &obj)
- virtual [~Pattern](#) ()
- [Pattern](#) & [operator=](#) (const [Pattern](#) &)
- double [compare](#) (const [Pattern](#) &) const
- bool [isAllZeroes](#) () const
- bool [operator==](#) (const [Pattern](#) &) const
- bool [operator<](#) (const [Pattern](#) &) const
- std::vector< bool > [getPattern](#) () const
- std::string [getString](#) () const
- void [setPattern](#) (const std::vector< bool > &)
- int [getId](#) () const
- void [setId](#) (int)
- int [getWidth](#) () const
- int [getSize](#) () const
- const [BinaryString](#) & [getBinaryString](#) () const
- [BinaryString](#) & [getMutableBinaryString](#) ()
- const boost::shared\_ptr< [PatternTag](#) > [getPatternTag](#) () const
- void [setPatternTag](#) (boost::shared\_ptr< [PatternTag](#) > pt)
- boost::shared\_ptr< [manager::DatabaseObject](#) > [getDatabaseObject](#) () const

### Static Public Member Functions

- static int [getIds](#) ()
- static int [assignIds](#) ()
- static void [setId](#) (int)
- static std::string [patternToString](#) (const std::vector< bool > &vec)



- static std::vector< bool > [stringToPattern](#) (const std::string &)
- static [Pattern getRandom](#) (unsigned int width, double fraction=0.5)  
*Generate a random pattern of width.*

### Private Member Functions

- template<class Archive >  
void [serialize](#) (Archive &ar, const unsigned int version)
- void [initialise](#) ()

### Private Attributes

- [BinaryString](#) [binaryString](#)
- int [id](#)
- boost::shared\_ptr< [PatternTag](#) > [patternTag](#)

### Static Private Attributes

- static int [ids](#) = 1

### Friends

- class [boost::serialization::access](#)
- std::ostream & [operator<<](#) (std::ostream &os, const [Pattern](#) &obj)

## 6.38.1 Detailed Description

Definition at line 27 of file Pattern.h.

## 6.38.2 Constructor & Destructor Documentation

### 6.38.2.1 cryomesh::state::Pattern::Pattern ( )

Definition at line 31 of file Pattern.cpp.

References [initialise\(\)](#).

Referenced by [getRandom\(\)](#).

### 6.38.2.2 cryomesh::state::Pattern::Pattern ( const Pattern & pat )

Definition at line 35 of file Pattern.cpp.

References [initialise\(\)](#).

**6.38.2.3 cryomesh::state::Pattern::Pattern ( const std::string & *str* )**

Definition at line 43 of file Pattern.cpp.

References initialise().

**6.38.2.4 cryomesh::state::Pattern::Pattern ( const std::vector< bool > & *pat* )**

Definition at line 39 of file Pattern.cpp.

References initialise().

**6.38.2.5 cryomesh::state::Pattern::Pattern ( const BinaryString & *obj* )****6.38.2.6 cryomesh::state::Pattern::~~Pattern ( ) [virtual]**

Definition at line 47 of file Pattern.cpp.

**6.38.3 Member Function Documentation****6.38.3.1 int cryomesh::state::Pattern::assignIds ( ) [static]**

Definition at line 212 of file Pattern.cpp.

References ids.

**6.38.3.2 double cryomesh::state::Pattern::compare ( const Pattern & *obj* ) const**

Definition at line 63 of file Pattern.cpp.

References getPattern().

**6.38.3.3 const BinaryString & cryomesh::state::Pattern::getBinaryString ( ) const**

Definition at line 159 of file Pattern.cpp.

References binaryString.

Referenced by operator=().

**6.38.3.4 boost::shared\_ptr< manager::DatabaseObject >  
cryomesh::state::Pattern::getDatabaseObject ( ) const**

Definition at line 229 of file Pattern.cpp.

References cryomesh::common::TimeKeeper::getCycle(), getString(), cryomesh::common::TimeKeeper::getTimeKeeper(), and cryomesh::common::Cycle::toULInt().

**6.38.3.5** `int cryomesh::state::Pattern::getId ( ) const`

Definition at line 142 of file Pattern.cpp.

References `id`.

Referenced by `operator<()`, `cryomesh::state::operator<<()`, and `operator=()`.

**6.38.3.6** `int cryomesh::state::Pattern::getIds ( ) [static]`

Definition at line 218 of file Pattern.cpp.

References `ids`.

Referenced by `cryomesh::state::PatternTagById::getEndTag()`.

**6.38.3.7** `BinaryString & cryomesh::state::Pattern::getMutableBinaryString ( )`

Definition at line 162 of file Pattern.cpp.

References `binaryString`.

**6.38.3.8** `std::vector< bool > cryomesh::state::Pattern::getPattern ( ) const`

Definition at line 133 of file Pattern.cpp.

References `binaryString`, and `cryomesh::state::BinaryString::getBools()`.

Referenced by `compare()`, `cryomesh::structures::Fibre::forceFireNodes()`, `getString()`, `operator==()`, and `cryomesh::structures::Fibre::trigger()`.

**6.38.3.9** `const boost::shared_ptr< PatternTag > cryomesh::state::Pattern::getPatternTag ( ) const`

Definition at line 205 of file Pattern.cpp.

References `patternTag`.

**6.38.3.10** `Pattern cryomesh::state::Pattern::getRandom ( unsigned int width, double fraction = 0.5 ) [static]`

Generate a random pattern of width.

**Parameters**

<i>unsigned</i>	int Width of pattern to generate
<i>double</i>	Fraction of pattern to activate, default to 0.5

#### Returns

[Pattern](#) The random pattern

Definition at line 21 of file Pattern.cpp.

References Pattern().

Referenced by cryomesh::structures::Fibre::trigger().

#### 6.38.3.11 int cryomesh::state::Pattern::getSize ( ) const

Definition at line 155 of file Pattern.cpp.

References binaryString, and cryomesh::state::BinaryString::getWidth().

Referenced by cryomesh::structures::Fibre::forceFireNodes(), and getWidth().

#### 6.38.3.12 std::string cryomesh::state::Pattern::getString ( ) const

Definition at line 136 of file Pattern.cpp.

References getPattern(), and patternToString().

Referenced by getDatabaseObject(), and cryomesh::state::operator<<().

#### 6.38.3.13 int cryomesh::state::Pattern::getWidth ( ) const

Definition at line 151 of file Pattern.cpp.

References getSize().

#### 6.38.3.14 void cryomesh::state::Pattern::initialise ( ) [private]

Definition at line 237 of file Pattern.cpp.

References patternTag.

Referenced by operator=(), and Pattern().

#### 6.38.3.15 bool cryomesh::state::Pattern::isAllZeroes ( ) const

Definition at line 60 of file Pattern.cpp.

References binaryString, and cryomesh::state::BinaryString::isAllZeroes().

#### 6.38.3.16 bool cryomesh::state::Pattern::operator< ( const Pattern & obj ) const

Definition at line 129 of file Pattern.cpp.

References getId().

**6.38.3.17 Pattern & cryomesh::state::Pattern::operator= ( const Pattern & obj )**

Definition at line 51 of file Pattern.cpp.

References `binaryString`, `getBinaryString()`, `getId()`, `initialise()`, and `setId()`.

**6.38.3.18 bool cryomesh::state::Pattern::operator== ( const Pattern & obj ) const**

Definition at line 115 of file Pattern.cpp.

References `getPattern()`.

**6.38.3.19 std::string cryomesh::state::Pattern::patternToString ( const std::vector< bool > & vec ) [static]**

Definition at line 166 of file Pattern.cpp.

Referenced by `getString()`.

**6.38.3.20 template<class Archive > void cryomesh::state::Pattern::serialize ( Archive & ar, const unsigned int version ) [inline, private]**

Definition at line 30 of file Pattern.h.

References `binaryString`, `id`, and `ids`.

**6.38.3.21 void cryomesh::state::Pattern::setId ( int new\_id )**

Definition at line 145 of file Pattern.cpp.

Referenced by `operator=()`.

**6.38.3.22 void cryomesh::state::Pattern::setIds ( int is ) [static]**

Definition at line 222 of file Pattern.cpp.

References `ids`.

**6.38.3.23 void cryomesh::state::Pattern::setPattern ( const std::vector< bool > & pat )**

Definition at line 139 of file Pattern.cpp.

References `binaryString`, and `cryomesh::state::BinaryString::setBinaryString()`.

**6.38.3.24 void cryomesh::state::Pattern::setPatternTag ( boost::shared\_ptr< PatternTag > pt )**

Definition at line 208 of file Pattern.cpp.

References patternTag.

**6.38.3.25** `std::vector< bool > cryomesh::state::Pattern::stringToPattern ( const std::string & str )` `[static]`

Definition at line 184 of file Pattern.cpp.

## 6.38.4 Friends And Related Function Documentation

**6.38.4.1** `friend class boost::serialization::access` `[friend]`

Definition at line 28 of file Pattern.h.

**6.38.4.2** `std::ostream& operator<< ( std::ostream & os, const Pattern & obj )` `[friend]`

Definition at line 242 of file Pattern.cpp.

## 6.38.5 Member Data Documentation

**6.38.5.1** `BinaryString cryomesh::state::Pattern::binaryString` `[private]`

Definition at line 115 of file Pattern.h.

Referenced by `getBinaryString()`, `getMutableBinaryString()`, `getPattern()`, `getSize()`, `isAllZeroes()`, `operator=()`, `serialize()`, and `setPattern()`.

**6.38.5.2** `int cryomesh::state::Pattern::id` `[private]`

Definition at line 117 of file Pattern.h.

Referenced by `getId()`, and `serialize()`.

**6.38.5.3** `int cryomesh::state::Pattern::ids = 1` `[static, private]`

Definition at line 128 of file Pattern.h.

Referenced by `assignIds()`, `getIds()`, `serialize()`, and `setIds()`.

**6.38.5.4** `boost::shared_ptr<PatternTag> cryomesh::state::Pattern::patternTag` `[private]`

Definition at line 120 of file Pattern.h.

Referenced by `getPatternTag()`, `initialise()`, and `setPatternTag()`.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern.cpp](#)

## 6.39 cryomesh::state::PatternChannel Class Reference

```
#include <PatternChannel.h>
```

### Public Types

- enum [PrintFormat](#) { [BINARY](#), [TEXT](#), [INTEGER](#) }
- enum [ChannelDataType](#) { [Input](#), [Output](#), [Transitional](#) }

### Public Member Functions

- [PatternChannel](#) ([ChannelDataType](#) dt)
- [PatternChannel](#) (const std::list< boost::shared\_ptr< [Pattern](#) > > &pats, [ChannelDataType](#) dt)
- virtual [~PatternChannel](#) ()
- [PatternChannel](#) (const [PatternChannel](#) &obj)
- [PatternChannel](#) & [operator=](#) (const [PatternChannel](#) &obj)  
*Assignment operator.*
- void [addPattern](#) (boost::shared\_ptr< [Pattern](#) > pat)
- void [addPatterns](#) (const std::list< boost::shared\_ptr< [Pattern](#) > > &pats)
- void [addPatterns](#) (const std::list< boost::shared\_ptr< [Pattern](#) > > &pats, int position)
- std::list< boost::shared\_ptr< [Pattern](#) > > [removePatterns](#) (const std::list< boost::shared\_ptr< [Pattern](#) > > &pats)
- std::list< boost::shared\_ptr< [Pattern](#) > > [removePatterns](#) (const std::list< boost::uuids::uuid > &uuids)
- void [clearPatternList](#) ()
- std::list< boost::shared\_ptr< [Pattern](#) > > [forcePatternListSize](#) (int sz)
- std::list< boost::shared\_ptr< [Pattern](#) > > [forcePatternListSize](#) ()
- const boost::shared\_ptr< [Pattern](#) > [getCurrentPattern](#) ()
- const boost::shared\_ptr< [Pattern](#) > [getPatternByCycle](#) (const [common::Cycle](#) &cycle)
- const boost::shared\_ptr< [Pattern](#) > [getPatternByTag](#) (const boost::shared\_ptr< [PatternTag](#) > tg) const
- const boost::shared\_ptr< [Pattern](#) > [nextPattern](#) ()
- const boost::shared\_ptr< [Pattern](#) > [previousPattern](#) ()
- double [matchGlobally](#) (const [PatternChannel](#) &)
- double [matchSequentially](#) (const [PatternChannel](#) &)
- virtual void [enableDebug](#) (bool b)
- [ChannelDataType](#) [getChannelDataType](#) () const
- int [getLength](#) () const

- `const std::map < boost::shared_ptr < PatternTag > , boost::uuids::uuid > & getPatternByTagMap () const`
- `const boost::shared_ptr< Pattern > getPatternByUUID (const boost::uuids::uuid &id) const`
- `boost::shared_ptr< Pattern > getMutablePatternByUUID (const boost::uuids::uuid &id)`
- `const std::list < boost::uuids::uuid > & getPatternList () const`
- `std::list< boost::uuids::uuid > ::iterator getPatternListIterator () const`
- `const std::map < boost::uuids::uuid, boost::shared_ptr< Pattern > > & getPatternMap () const`
- `int getPatternPosition () const`
- `int getWidth () const`
- `void setWidth (int w)`
- `int getRefID () const`
- `void setRefID (int r)`
- `int getMaxPatternListSize () const`
- `void setMaxPatternListSize (int sz)`
- `std::ostream & printPatternList (std::ostream &os, bool reversed=false) const`
- `std::ostream & printTextFormattedPatternList (std::ostream &os, bool reversed=false) const`
- `std::ostream & printIntegerFormattedPatternList (std::ostream &os, bool reversed) const`
- `std::ostream & printBinaryFormattedPatternList (std::ostream &os, bool reversed) const`
- `std::ostream & printFormattedPatternList (std::ostream &os, const PrintFormat &pf, bool reversed) const`

### Static Public Member Functions

- `static unsigned int getRefIDS ()`

### Static Public Attributes

- `static const int DEFAULT\_MAX\_PATTERN\_LIST\_SIZE = -1`
- `static const unsigned int REFID\_CREATE\_START = 100000`
- `static unsigned int refIDS = PatternChannel::REFID\_CREATE\_START`

### Private Attributes

- `ChannelDataType channelDataType`
- `int length`
- `int maxPatternListSize`
- `std::map< boost::shared_ptr < PatternTag > , boost::uuids::uuid > patternByTagMap`
- `std::list< boost::uuids::uuid > patternList`



- `std::list< boost::uuids::uuid > ::iterator` [patternListIterator](#)
- `std::map< boost::uuids::uuid, boost::shared_ptr< Pattern > >` [patternMap](#)
- `int` [patternPosition](#)
- `int` [refID](#)
- `int` [width](#)

## Friends

- `std::ostream &` [operator<<](#) (`std::ostream &os`, `const` [PatternChannel](#) &obj)

### 6.39.1 Detailed Description

Definition at line 25 of file `PatternChannel.h`.

### 6.39.2 Member Enumeration Documentation

#### 6.39.2.1 `enum cryomesh::state::PatternChannel::ChannelDataType`

Enumerator:

***Input***

***Output***

***Transitional***

Definition at line 30 of file `PatternChannel.h`.

#### 6.39.2.2 `enum cryomesh::state::PatternChannel::PrintFormat`

Enumerator:

***BINARY***

***TEXT***

***INTEGER***

Definition at line 27 of file `PatternChannel.h`.

### 6.39.3 Constructor & Destructor Documentation

#### 6.39.3.1 `cryomesh::state::PatternChannel::PatternChannel ( ChannelDataType dt )`

Definition at line 24 of file `PatternChannel.cpp`.

References `channelDataType`, and `Input`.

**6.39.3.2 cryomesh::state::PatternChannel::PatternChannel ( const std::list< boost::shared\_ptr< Pattern > > & pats, ChannelDataType dt )**

Definition at line 37 of file PatternChannel.cpp.

References addPatterns(), channelDataType, DEFAULT\_MAX\_PATTERN\_LIST\_SIZE, getRefIDS(), Input, maxPatternListSize, Output, patternList, patternListIterator, and refID.

**6.39.3.3 cryomesh::state::PatternChannel::~~PatternChannel ( ) [virtual]**

Definition at line 59 of file PatternChannel.cpp.

**6.39.3.4 cryomesh::state::PatternChannel::PatternChannel ( const PatternChannel & obj )**

Definition at line 65 of file PatternChannel.cpp.

References channelDataType, length, maxPatternListSize, patternByTagMap, patternList, patternListIterator, patternMap, patternPosition, refID, and width.

## 6.39.4 Member Function Documentation

**6.39.4.1 void cryomesh::state::PatternChannel::addPattern ( boost::shared\_ptr< Pattern > pat )**

Definition at line 98 of file PatternChannel.cpp.

References addPatterns(), and getLength().

**6.39.4.2 void cryomesh::state::PatternChannel::addPatterns ( const std::list< boost::shared\_ptr< Pattern > > & pats )**

Definition at line 103 of file PatternChannel.cpp.

References getLength().

Referenced by addPattern(), and PatternChannel().

**6.39.4.3 void cryomesh::state::PatternChannel::addPatterns ( const std::list< boost::shared\_ptr< Pattern > > & pats, int position )**

Definition at line 107 of file PatternChannel.cpp.

References forcePatternListSize(), getLength(), getWidth(), length, patternByTagMap, patternList, patternListIterator, patternMap, patternPosition, and setWidth().

**6.39.4.4 void cryomesh::state::PatternChannel::clearPatternList ( )**

Definition at line 200 of file PatternChannel.cpp.

References length, patternByTagMap, patternList, patternListIterator, patternMap, patternPosition, and width.

**6.39.4.5 void cryomesh::state::PatternChannel::enableDebug ( bool b )**  
[virtual]

Definition at line 536 of file PatternChannel.cpp.

**6.39.4.6 std::list< boost::shared\_ptr< Pattern > > cryomesh-  
::state::PatternChannel::forcePatternListSize ( int sz  
)**

Definition at line 209 of file PatternChannel.cpp.

References patternList, and removePatterns().

**6.39.4.7 std::list< boost::shared\_ptr< Pattern > > cryomesh::state::PatternChannel-  
::forcePatternListSize ( )**

Definition at line 233 of file PatternChannel.cpp.

References getMaxPatternListSize().

Referenced by addPatterns(), and setMaxPatternListSize().

**6.39.4.8 PatternChannel::ChannelDataType cryomesh::state::PatternChannel-  
::getChannelDataType ( ) const**

Definition at line 498 of file PatternChannel.cpp.

References channelDataType.

**6.39.4.9 const boost::shared\_ptr< Pattern > cryomesh::state::PatternChannel::get-  
CurrentPattern ( )**

Definition at line 241 of file PatternChannel.cpp.

References patternMap.

Referenced by nextPattern(), and previousPattern().

**6.39.4.10 int cryomesh::state::PatternChannel::getLength ( ) const**

Definition at line 458 of file PatternChannel.cpp.

References patternList.

Referenced by addPattern(), addPatterns(), and nextPattern().

**6.39.4.11** `int cryomesh::state::PatternChannel::getMaxPatternListSize ( ) const`

Definition at line 502 of file PatternChannel.cpp.

References maxPatternListSize.

Referenced by forcePatternListSize().

**6.39.4.12** `boost::shared_ptr< Pattern > cryomesh::state::PatternChannel::getMutablePatternByUUID ( const boost::uuids::uuid & id )`

Definition at line 524 of file PatternChannel.cpp.

References patternMap.

**6.39.4.13** `const boost::shared_ptr< Pattern > cryomesh::state::PatternChannel::getPatternByCycle ( const common::Cycle & cycle )`

Definition at line 245 of file PatternChannel.cpp.

References cryomesh::common::TimeKeeper::getCycle(), cryomesh::common::TimeKeeper::getTimeKeeper(), patternList, and patternMap.

**6.39.4.14** `const boost::shared_ptr< Pattern > cryomesh::state::PatternChannel::getPatternByTag ( const boost::shared_ptr< PatternTag > tg ) const`

Definition at line 269 of file PatternChannel.cpp.

References patternByTagMap, and patternMap.

**6.39.4.15** `const std::map< boost::shared_ptr< PatternTag >, boost::uuids::uuid > & cryomesh::state::PatternChannel::getPatternByTagMap ( ) const`

Definition at line 462 of file PatternChannel.cpp.

References patternByTagMap.

**6.39.4.16** `const boost::shared_ptr< Pattern > cryomesh::state::PatternChannel::getPatternByUUID ( const boost::uuids::uuid & id ) const`

Definition at line 512 of file PatternChannel.cpp.

References patternMap.

Referenced by printFormattedPatternList(), and printPatternList().

**6.39.4.17** `const std::list< boost::uuids::uuid > & cryomesh::state::PatternChannel::getPatternList ( ) const`

Definition at line 466 of file PatternChannel.cpp.

References patternList.

Referenced by matchGlobally(), and matchSequentially().

**6.39.4.18** `std::list< boost::uuids::uuid >::iterator cryomesh::state::PatternChannel::getPatternListIterator ( ) const`

Definition at line 470 of file PatternChannel.cpp.

References patternListIterator.

**6.39.4.19** `const std::map< boost::uuids::uuid, boost::shared_ptr< Pattern > > & cryomesh::state::PatternChannel::getPatternMap ( ) const`

Definition at line 474 of file PatternChannel.cpp.

References patternMap.

Referenced by matchGlobally(), and matchSequentially().

**6.39.4.20** `int cryomesh::state::PatternChannel::getPatternPosition ( ) const`

Definition at line 478 of file PatternChannel.cpp.

References patternPosition.

**6.39.4.21** `int cryomesh::state::PatternChannel::getRefID ( ) const`

Definition at line 490 of file PatternChannel.cpp.

References refID.

**6.39.4.22** `unsigned int cryomesh::state::PatternChannel::getRefIDS ( ) [static]`

Definition at line 20 of file PatternChannel.cpp.

References refIDS.

Referenced by PatternChannel().

**6.39.4.23 int cryomesh::state::PatternChannel::getWidth ( ) const**

Definition at line 482 of file PatternChannel.cpp.

References width.

Referenced by addPatterns().

**6.39.4.24 double cryomesh::state::PatternChannel::matchGlobally ( const PatternChannel & obj )**

Definition at line 309 of file PatternChannel.cpp.

References getPatternList(), and getPatternMap().

**6.39.4.25 double cryomesh::state::PatternChannel::matchSequentially ( const PatternChannel & obj )**

Definition at line 395 of file PatternChannel.cpp.

References getPatternList(), and getPatternMap().

**6.39.4.26 const boost::shared\_ptr< Pattern > cryomesh::state::PatternChannel::nextPattern ( )**

Definition at line 278 of file PatternChannel.cpp.

References getCurrentPattern(), getLength(), patternList, patternListIterator, and patternPosition.

**6.39.4.27 PatternChannel & cryomesh::state::PatternChannel::operator= ( const PatternChannel & obj )**

Assignment operator.

**Parameters**

<i>const</i>	<a href="#">PatternChannel</a> & obj RHS assignment
--------------	---

**Returns**

[PatternChannel](#) & This object after assignment

Definition at line 81 of file PatternChannel.cpp.

References channelDataType, length, maxPatternListSize, patternByTagMap, patternList, patternListIterator, patternMap, patternPosition, refID, and width.

6.39.4.28 `const boost::shared_ptr< Pattern > cryomesh::state::PatternChannel::previousPattern ( )`

Definition at line 298 of file PatternChannel.cpp.

References `getCurrentPattern()`, `patternList`, `patternListIterator`, and `patternPosition`.

6.39.4.29 `std::ostream & cryomesh::state::PatternChannel::printBinaryFormattedPatternList ( std::ostream & os, bool reversed ) const`

Definition at line 576 of file PatternChannel.cpp.

References `BINARY`, and `printFormattedPatternList()`.

Referenced by `cryomesh::state::operator<<()`.

6.39.4.30 `std::ostream & cryomesh::state::PatternChannel::printFormattedPatternList ( std::ostream & os, const PrintFormat & pf, bool reversed ) const`

Definition at line 579 of file PatternChannel.cpp.

References `BINARY`, `getPatternByUUID()`, `INTEGER`, `patternList`, and `TEXT`.

Referenced by `printBinaryFormattedPatternList()`, `printIntegerFormattedPatternList()`, and `printTextFormattedPatternList()`.

6.39.4.31 `std::ostream & cryomesh::state::PatternChannel::printIntegerFormattedPatternList ( std::ostream & os, bool reversed ) const`

Definition at line 573 of file PatternChannel.cpp.

References `INTEGER`, and `printFormattedPatternList()`.

6.39.4.32 `std::ostream & cryomesh::state::PatternChannel::printPatternList ( std::ostream & os, bool reversed = false ) const`

Definition at line 540 of file PatternChannel.cpp.

References `getPatternByUUID()`, and `patternList`.

6.39.4.33 `std::ostream & cryomesh::state::PatternChannel::printTextFormattedPatternList ( std::ostream & os, bool reversed = false ) const`

Definition at line 569 of file PatternChannel.cpp.

References `printFormattedPatternList()`, and `TEXT`.

**6.39.4.34** `std::list< boost::shared_ptr< Pattern > > cryomesh::state::PatternChannel::removePatterns ( const std::list< boost::shared_ptr< Pattern > > & pats )`

Definition at line 156 of file PatternChannel.cpp.

Referenced by forcePatternListSize().

**6.39.4.35** `std::list< boost::shared_ptr< Pattern > > cryomesh::state::PatternChannel::removePatterns ( const std::list< boost::uuids::uuid > & uuids )`

Definition at line 175 of file PatternChannel.cpp.

References length, patternByTagMap, patternList, patternListIterator, patternMap, and patternPosition.

**6.39.4.36** `void cryomesh::state::PatternChannel::setMaxPatternListSize ( int sz )`

Definition at line 505 of file PatternChannel.cpp.

References forcePatternListSize(), and maxPatternListSize.

**6.39.4.37** `void cryomesh::state::PatternChannel::setRefID ( int r )`

Definition at line 493 of file PatternChannel.cpp.

References refID.

**6.39.4.38** `void cryomesh::state::PatternChannel::setWidth ( int w )`

Definition at line 485 of file PatternChannel.cpp.

References width.

Referenced by addPatterns().

## 6.39.5 Friends And Related Function Documentation

**6.39.5.1** `std::ostream& operator<< ( std::ostream & os, const PatternChannel & obj )`  
[friend]

Definition at line 639 of file PatternChannel.cpp.

## 6.39.6 Member Data Documentation



#### 6.39.6.1 ChannelDataType cryomesh::state::PatternChannel::channelDataType [private]

Definition at line 169 of file PatternChannel.h.

Referenced by getChannelDataType(), operator=(), and PatternChannel().

#### 6.39.6.2 const int cryomesh::state::PatternChannel::DEFAULT\_MAX\_PATTERN\_LIST\_SIZE = -1 [static]

Definition at line 161 of file PatternChannel.h.

Referenced by PatternChannel().

#### 6.39.6.3 int cryomesh::state::PatternChannel::length [private]

Definition at line 171 of file PatternChannel.h.

Referenced by addPatterns(), clearPatternList(), cryomesh::state::operator<<(), operator=(), PatternChannel(), and removePatterns().

#### 6.39.6.4 int cryomesh::state::PatternChannel::maxPatternListSize [private]

Definition at line 172 of file PatternChannel.h.

Referenced by getMaxPatternListSize(), cryomesh::state::operator<<(), operator=(), PatternChannel(), and setMaxPatternListSize().

#### 6.39.6.5 std::map<boost::shared\_ptr<PatternTag>, boost::uuids::uuid> cryomesh::state::PatternChannel::patternByTagMap [private]

Definition at line 174 of file PatternChannel.h.

Referenced by addPatterns(), clearPatternList(), getPatternByTag(), getPatternByTagMap(), operator=(), PatternChannel(), and removePatterns().

#### 6.39.6.6 std::list<boost::uuids::uuid> cryomesh::state::PatternChannel::patternList [private]

Definition at line 176 of file PatternChannel.h.

Referenced by addPatterns(), clearPatternList(), forcePatternListSize(), getLength(), getPatternByCycle(), getPatternList(), nextPattern(), operator=(), PatternChannel(), previousPattern(), printFormattedPatternList(), printPatternList(), and removePatterns().

**6.39.6.7** `std::list<boost::uuids::uuid>::iterator cryomesh::state::PatternChannel::patternListIterator` [private]

Definition at line 178 of file PatternChannel.h.

Referenced by `addPatterns()`, `clearPatternList()`, `getPatternListIterator()`, `nextPattern()`, `operator=()`, `PatternChannel()`, `previousPattern()`, and `removePatterns()`.

**6.39.6.8** `std::map<boost::uuids::uuid, boost::shared_ptr<Pattern>> > cryomesh::state::PatternChannel::patternMap` [private]

Definition at line 180 of file PatternChannel.h.

Referenced by `addPatterns()`, `clearPatternList()`, `getCurrentPattern()`, `getMutablePatternByUUID()`, `getPatternByCycle()`, `getPatternByTag()`, `getPatternByUUID()`, `getPatternMap()`, `cryomesh::state::operator<<()`, `operator=()`, `PatternChannel()`, and `removePatterns()`.

**6.39.6.9** `int cryomesh::state::PatternChannel::patternPosition` [private]

Definition at line 182 of file PatternChannel.h.

Referenced by `addPatterns()`, `clearPatternList()`, `getPatternPosition()`, `nextPattern()`, `cryomesh::state::operator<<()`, `operator=()`, `PatternChannel()`, `previousPattern()`, and `removePatterns()`.

**6.39.6.10** `int cryomesh::state::PatternChannel::refID` [private]

Definition at line 184 of file PatternChannel.h.

Referenced by `getRefID()`, `cryomesh::state::operator<<()`, `operator=()`, `PatternChannel()`, and `setRefID()`.

**6.39.6.11** `const unsigned int cryomesh::state::PatternChannel::REFID_CREATE_START = 100000` [static]

Definition at line 163 of file PatternChannel.h.

**6.39.6.12** `unsigned int cryomesh::state::PatternChannel::refIDS = PatternChannel::REFID_CREATE_START` [static]

Definition at line 164 of file PatternChannel.h.

Referenced by `getRefIDS()`.

**6.39.6.13** `int cryomesh::state::PatternChannel::width` [private]

Definition at line 186 of file PatternChannel.h.

Referenced by `clearPatternList()`, `getWidth()`, `cryomesh::state::operator<<()`, `operator=()`, `PatternChannel()`, and `setWidth()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannel.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannel.cpp`

## 6.40 cryomesh::state::PatternChannelMap Class Reference

```
#include <PatternChannelMap.h>
```

### Public Member Functions

- [PatternChannelMap](#) ()
- virtual [~PatternChannelMap](#) ()
- const std::map < boost::uuids::uuid, boost::shared\_ptr < [state::Pattern](#) > > [getPatterns](#) (const [common::Cycle](#) &cycle=[common::TimeKeeper::getTimeKeeper\(\)](#).[getTimeKeeper\(\)](#).[getCycle\(\)](#)) const

*Get a mapping of pattern channel uuids to their patterns on a specific cycle.*

### 6.40.1 Detailed Description

Definition at line 19 of file `PatternChannelMap.h`.

### 6.40.2 Constructor & Destructor Documentation

6.40.2.1 `cryomesh::state::PatternChannelMap::PatternChannelMap ( )`  
[inline]

Definition at line 21 of file `PatternChannelMap.h`.

6.40.2.2 `virtual cryomesh::state::PatternChannelMap::~~PatternChannelMap ( )`  
[inline, virtual]

Definition at line 23 of file `PatternChannelMap.h`.

### 6.40.3 Member Function Documentation

```

6.40.3.1  const std::map<boost::uuids::uuid, boost::shared_ptr<state::Pattern>
> cryomesh::state::PatternChannelMap::getPatterns ( const
common::Cycle & cycle = common::TimeKeeper::getTime-
Keeper().getTimeKeeper().getCycle() ) const
[inline]

```

Get a mapping of pattern channel uuids to their patterns on a specific cycle.

Definition at line 30 of file PatternChannelMap.h.

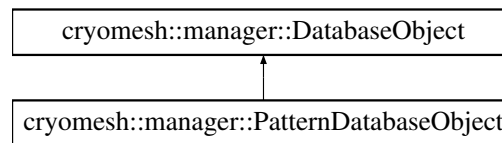
The documentation for this class was generated from the following file:

- /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternChannelMap.h

## 6.41 cryomesh::manager::PatternDatabaseObject Class Reference

```
#include <PatternDatabaseObject.h>
```

Inheritance diagram for cryomesh::manager::PatternDatabaseObject:



### Public Member Functions

- [PatternDatabaseObject](#) (const std::string uuid\_str, const [common::Cycle](#) &cyc, const [state::Pattern](#) &pat)  
*Create database object using pattern and a cycle.*
- [PatternDatabaseObject](#) (const std::string &node\_table\_entry)  
*Create object from the string of entries in the database output table.*
- virtual [~PatternDatabaseObject](#) ()  
*Default destructor.*
- virtual std::string [getInsert](#) (const std::string &table) const  
*Get the string that can be used to insert the sql data.*
- std::string [getUUID](#) () const  
*Get uuid variable.*
- const [common::Cycle](#) & [getCycle](#) () const  
*Get cycle variable.*
- const [state::Pattern](#) & [getPattern](#) () const  
*Get pattern variable.*
- std::string [getKey](#) (const std::string &key) const  
*Return the string object associated with a key.*

### Static Public Member Functions

- static std::string [findValue](#) (const std::string &entry, const std::map< std::string, std::string > &map)  
*Find entries value in map or return null.*
- static std::map< std::string, std::string > [getColumnMapFromEntry](#) (const std::string &entry)  
*Parse a string database entry, extract columns and values and return a map.*
- template<class T >  
static std::string [toString](#) (T obj)  
*Convert an templated object that can be piped to a stream to a string.*

### Static Public Attributes

- static const std::string [ID\\_TAG](#) = "id"
- static const std::string [CYCLE\\_TAG](#) = "cycle"
- static const std::string [PATTERN\\_TAG](#) = "pattern"

### Protected Attributes

- std::map< std::string, std::string > [columns](#)

### Private Attributes

- std::string [uuid](#)
- [common::Cycle](#) [cycle](#)
- boost::shared\_ptr< [state::Pattern](#) > [pattern](#)

#### 6.41.1 Detailed Description

Definition at line 22 of file PatternDatabaseObject.h.

#### 6.41.2 Constructor & Destructor Documentation

- 6.41.2.1 **cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject** (  
const std::string *uuid\_str*, const [common::Cycle](#) & *cyc*, const [state::Pattern](#) &  
*pat* )

Create database object using pattern and a cycle.

Definition at line 19 of file PatternDatabaseObject.cpp.

References [cryomesh::manager::DatabaseObject::columns](#), [cycle](#), [CYCLE\\_TAG](#), [ID\\_TAG](#), [pattern](#), [PATTERN\\_TAG](#), [cryomesh::common::Cycle::toULInt\(\)](#), and [uuid](#).

#### 6.41.2.2 `cryomesh::manager::PatternDatabaseObject::PatternDatabaseObject ( const std::string & node_table_entry )`

Create object from the string of entries in the database output table.

##### Parameters

<code>std::string</code>	The string of data taken from a output entry in the database node table
--------------------------	---

Definition at line 26 of file `PatternDatabaseObject.cpp`.

References `cycle`, `CYCLE_TAG`, `cryomesh::manager::DatabaseObject::findValue()`, `cryomesh::manager::DatabaseObject::getColumnMapFromEntry()`, `ID_TAG`, `pattern`, `PATTERN_TAG`, and `uuid`.

#### 6.41.2.3 `cryomesh::manager::PatternDatabaseObject::~~PatternDatabaseObject ( ) [virtual]`

Default destructor.

Definition at line 38 of file `PatternDatabaseObject.cpp`.

### 6.41.3 Member Function Documentation

#### 6.41.3.1 `static std::string cryomesh::manager::DatabaseObject::findValue ( const std::string & entry, const std::map< std::string, std::string > & map ) [inline, static, inherited]`

Find entries value in map or return null.

##### Parameters

<code>std::string</code>	Entry to find
<code>std::map&lt; std::string, std::string &gt;</code>	map to search

##### Returns

Value of entry

Definition at line 59 of file `DatabaseObject.h`.

Referenced by `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject()`, `cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject()`, and `PatternDatabaseObject()`.

6.41.3.2 `static std::map<std::string, std::string> cryomesh::manager::DatabaseObject::getColumnMapFromEntry ( const std::string & entry ) [inline, static, inherited]`

Parse a string database entry, extract columns and values and return a map.

Definition at line 72 of file DatabaseObject.h.

Referenced by `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject()`, `cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject()`, and `PatternDatabaseObject()`.

6.41.3.3 `const common::Cycle & cryomesh::manager::PatternDatabaseObject::getCycle ( ) const`

Get cycle variable.

#### Returns

`common::Cycle` The cycle variable

Definition at line 55 of file PatternDatabaseObject.cpp.

References `cycle`.

6.41.3.4 `std::string cryomesh::manager::PatternDatabaseObject::getInsert ( const std::string & table ) const [virtual]`

Get the string that can be used to insert the sql data.

#### Returns

the sql command string to insert into this table

Implements `cryomesh::manager::DatabaseObject`.

Definition at line 42 of file PatternDatabaseObject.cpp.

References `CYCLE_TAG`, `cryomesh::manager::DatabaseObject::getKey()`, `ID_TAG`, and `PATTERN_TAG`.

6.41.3.5 `std::string cryomesh::manager::DatabaseObject::getKey ( const std::string & key ) const [inline, inherited]`

Return the string object associated with a key.

`::string` The key to search for

**Returns**

std::string The object associated with the search key, "" if not found

Definition at line 37 of file DatabaseObject.h.

References cryomesh::manager::DatabaseObject::columns.

Referenced by getInsert(), cryomesh::manager::NodeDatabaseObject::getInsert(), and cryomesh::manager::ConnectionDatabaseObject::getInsert().

#### 6.41.3.6 **const state::Pattern & cryomesh::manager::PatternDatabaseObject::getPattern ( ) const**

Get pattern variable.

**Returns**

Pattern The pattern variable

Definition at line 59 of file PatternDatabaseObject.cpp.

References pattern.

#### 6.41.3.7 **std::string cryomesh::manager::PatternDatabaseObject::getUUID ( ) const**

Get uuid variable.

**Returns**

std::string The uuid variable

Definition at line 52 of file PatternDatabaseObject.cpp.

References uuid.

#### 6.41.3.8 **template<class T > static std::string cryomesh::manager::DatabaseObject::toString ( T obj ) [inline, static, inherited]**

Convert an templated object that can be piped to a stream to a string.

**Parameters**

<b>T</b>	The object to get a string for
----------	--------------------------------

Definition at line 108 of file DatabaseObject.h.

### 6.41.4 Member Data Documentation



**6.41.4.1** `std::map<std::string, std::string> cryomesh::manager::DatabaseObject::columns` `[protected, inherited]`

Definition at line 119 of file DatabaseObject.h.

Referenced by `cryomesh::manager::ConnectionDatabaseObject::ConnectionDatabaseObject()`, `cryomesh::manager::DatabaseObject::getKey()`, `cryomesh::manager::NodeDatabaseObject::NodeDatabaseObject()`, and `PatternDatabaseObject()`.

**6.41.4.2** `common::Cycle cryomesh::manager::PatternDatabaseObject::cycle` `[private]`

Definition at line 106 of file PatternDatabaseObject.h.

Referenced by `getCycle()`, and `PatternDatabaseObject()`.

**6.41.4.3** `const std::string cryomesh::manager::PatternDatabaseObject::CYCLE_TAG = "cycle"` `[static]`

Definition at line 85 of file PatternDatabaseObject.h.

Referenced by `getInsert()`, and `PatternDatabaseObject()`.

**6.41.4.4** `const std::string cryomesh::manager::PatternDatabaseObject::ID_TAG = "id"` `[static]`

Definition at line 79 of file PatternDatabaseObject.h.

Referenced by `getInsert()`, and `PatternDatabaseObject()`.

**6.41.4.5** `boost::shared_ptr<state::Pattern> cryomesh::manager::PatternDatabaseObject::pattern` `[private]`

Definition at line 113 of file PatternDatabaseObject.h.

Referenced by `getPattern()`, and `PatternDatabaseObject()`.

**6.41.4.6** `const std::string cryomesh::manager::PatternDatabaseObject::PATTERN_TAG = "pattern"` `[static]`

Definition at line 92 of file PatternDatabaseObject.h.

Referenced by `getInsert()`, and `PatternDatabaseObject()`.

**6.41.4.7** `std::string cryomesh::manager::PatternDatabaseObject::uuid` `[private]`

Definition at line 99 of file PatternDatabaseObject.h.

Referenced by `getUUID()`, and `PatternDatabaseObject()`.

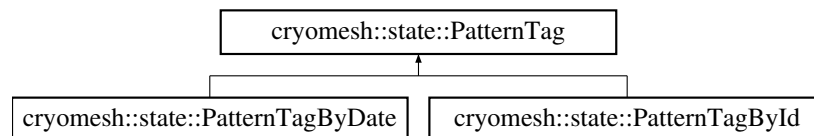
The documentation for this class was generated from the following files:

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/PatternDatabase-Object.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/PatternDatabase-Object.cpp`

## 6.42 cryomesh::state::PatternTag Class Reference

```
#include <PatternTag.h>
```

Inheritance diagram for `cryomesh::state::PatternTag`:



### Public Member Functions

- `PatternTag ()`
- `virtual ~PatternTag ()`
- `virtual std::string getTag () const =0`
- `virtual void setTag (std::string tg)=0`
- `virtual std::string moveTag ()=0`
- `virtual std::string moveTag (int i)=0`
- `virtual std::string getStartTag () const =0`
- `virtual std::string getEndTag () const =0`
- `virtual void setStartTag (std::string tg)=0`
- `virtual void setEndTag (std::string tg)=0`
- `virtual boost::shared_ptr < PatternTag > getGlobalTag ()=0`

### 6.42.1 Detailed Description

Definition at line 17 of file `PatternTag.h`.

### 6.42.2 Constructor & Destructor Documentation

#### 6.42.2.1 cryomesh::state::PatternTag::PatternTag ( ) [inline]

Definition at line 19 of file `PatternTag.h`.

6.42.2.2 `virtual cryomesh::state::PatternTag::~~PatternTag ( ) [inline, virtual]`

Definition at line 20 of file PatternTag.h.

### 6.42.3 Member Function Documentation

6.42.3.1 `virtual std::string cryomesh::state::PatternTag::getEndTag ( ) const [pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

6.42.3.2 `virtual boost::shared_ptr<PatternTag> cryomesh::state::PatternTag::get-GlobalTag ( ) [pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

6.42.3.3 `virtual std::string cryomesh::state::PatternTag::getStartTag ( ) const [pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

6.42.3.4 `virtual std::string cryomesh::state::PatternTag::getTag ( ) const [pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

6.42.3.5 `virtual std::string cryomesh::state::PatternTag::moveTag ( ) [pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

6.42.3.6 `virtual std::string cryomesh::state::PatternTag::moveTag ( int i ) [pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

6.42.3.7 `virtual void cryomesh::state::PatternTag::setEndTag ( std::string tg )`  
`[pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

6.42.3.8 `virtual void cryomesh::state::PatternTag::setStartTag ( std::string tg )`  
`[pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

6.42.3.9 `virtual void cryomesh::state::PatternTag::setTag ( std::string tg )` `[pure virtual]`

Implemented in [cryomesh::state::PatternTagByDate](#), and [cryomesh::state::PatternTag-ById](#).

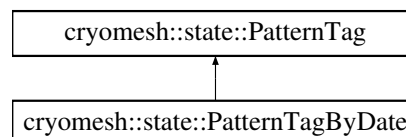
The documentation for this class was generated from the following file:

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTag.h](#)

## 6.43 cryomesh::state::PatternTagByDate Class Reference

```
#include <PatternTagByDate.h>
```

Inheritance diagram for cryomesh::state::PatternTagByDate:



### Public Types

- enum [DateType](#) { [ByHour](#), [ByDay](#), [ByWeek](#), [ByMonth](#), [ByYear](#) }

### Public Member Functions

- `template<class Archive >`  
`void serialize (Archive &ar, const unsigned int version)`
- `PatternTagByDate (DateType dt)`
- `PatternTagByDate (DateType dt, std::tm start_time, std::tm end_time, std::tm current_time)`

- virtual [~PatternTagByDate](#) ()
- virtual std::string [getTag](#) () const
- virtual void [setTag](#) (std::string tg)
- virtual std::string [moveTag](#) ()
- virtual std::string [moveTag](#) (int i)
- virtual std::string [getStartTag](#) () const
- virtual void [setStartTag](#) (std::string tg)
- virtual std::string [getEndTag](#) () const
- virtual void [setEndTag](#) (std::string tg)
- virtual boost::shared\_ptr < [PatternTag](#) > [getGlobalTag](#) ()

### Static Public Member Functions

- static std::tm [tagToTm](#) (const std::string &tg)
- static std::string [tmToTag](#) (const std::tm &t)
- static std::tm [moveHour](#) (std::tm &ttime, int i)
- static std::tm [moveDay](#) (std::tm &ttime, int i)
- static std::tm [moveWeek](#) (std::tm &ttime, int i)
- static std::tm [moveMonth](#) (std::tm &ttime, int i)
- static std::tm [moveYear](#) (std::tm &ttime, int i)
- static bool [isLeapYear](#) (const std::tm &tmtime)

### Static Protected Attributes

- static std::string [GlobalStartTag](#) = ""
- static std::string [GlobalEndTag](#) = ""
- static std::string [GlobalCurrentTag](#) = ""
- static std::string [DateFormat](#) = "%S %M %H %d %m %Y %w %j "

### Private Attributes

- [DateType](#) [dateType](#)
- std::tm [startTime](#)
- std::tm [endTime](#)
- std::tm [currentTime](#)

### Static Private Attributes

- static boost::shared\_ptr < [PatternTagByDate](#) > [globalTag](#)

### Friends

- class [boost::serialization::access](#)

### 6.43.1 Detailed Description

Definition at line 20 of file PatternTagByDate.h.

### 6.43.2 Member Enumeration Documentation

#### 6.43.2.1 enum cryomesh::state::PatternTagByDate::DateType

Enumerator:

***ByHour***  
***ByDay***  
***ByWeek***  
***ByMonth***  
***ByYear***

Definition at line 22 of file PatternTagByDate.h.

### 6.43.3 Constructor & Destructor Documentation

#### 6.43.3.1 cryomesh::state::PatternTagByDate::PatternTagByDate ( DateType dt )

Definition at line 22 of file PatternTagByDate.cpp.

#### 6.43.3.2 cryomesh::state::PatternTagByDate::PatternTagByDate ( DateType dt, std::tm start\_time, std::tm end\_time, std::tm current\_time )

Definition at line 26 of file PatternTagByDate.cpp.

References GlobalEndTag, GlobalStartTag, globalTag, and startTime.

#### 6.43.3.3 cryomesh::state::PatternTagByDate::~~PatternTagByDate ( ) [virtual]

Definition at line 34 of file PatternTagByDate.cpp.

### 6.43.4 Member Function Documentation

#### 6.43.4.1 std::string cryomesh::state::PatternTagByDate::getEndTag ( ) const [virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 70 of file PatternTagByDate.cpp.

References endTime, and tmToTag().

**6.43.4.2** `boost::shared_ptr< PatternTag > cryomesh::state::PatternTagByDate::get-GlobalTag ( )` [virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 77 of file PatternTagByDate.cpp.

References `globalTag`.

**6.43.4.3** `std::string cryomesh::state::PatternTagByDate::getStartTag ( )` const [virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 64 of file PatternTagByDate.cpp.

References `startTime`, and `tmToTag()`.

**6.43.4.4** `std::string cryomesh::state::PatternTagByDate::getTag ( )` const [virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 37 of file PatternTagByDate.cpp.

References `currentTime`, and `tmToTag()`.

**6.43.4.5** `bool cryomesh::state::PatternTagByDate::isLeapYear ( const std::tm & tmtime )` [static]

Definition at line 206 of file PatternTagByDate.cpp.

Referenced by `moveMonth()`, and `moveYear()`.

**6.43.4.6** `std::tm cryomesh::state::PatternTagByDate::moveDay ( std::tm & ttime, int i )` [static]

Definition at line 151 of file PatternTagByDate.cpp.

References `moveHour()`.

Referenced by `moveMonth()`, `moveTag()`, `moveWeek()`, and `moveYear()`.

**6.43.4.7** `std::tm cryomesh::state::PatternTagByDate::moveHour ( std::tm & ttime, int i )` [static]

Definition at line 139 of file PatternTagByDate.cpp.

Referenced by `moveDay()`, and `moveTag()`.

**6.43.4.8** `std::tm cryomesh::state::PatternTagByDate::moveMonth ( std::tm & time,  
int i ) [static]`

Definition at line 161 of file PatternTagByDate.cpp.

References `isLeapYear()`, `moveDay()`, and `moveYear()`.

Referenced by `moveTag()`.

**6.43.4.9** `std::string cryomesh::state::PatternTagByDate::moveTag ( )  
[virtual]`

Implements [cryomesh::state::PatternTag](#).

Definition at line 43 of file PatternTagByDate.cpp.

References `ByDay`, `ByHour`, `ByMonth`, `ByWeek`, `ByYear`, `currentTime`, `dateType`, `moveDay()`, `moveHour()`, `moveMonth()`, `moveWeek()`, `moveYear()`, and `tmToTag()`.

Referenced by `moveTag()`.

**6.43.4.10** `std::string cryomesh::state::PatternTagByDate::moveTag ( int i )  
[virtual]`

Implements [cryomesh::state::PatternTag](#).

Definition at line 57 of file PatternTagByDate.cpp.

References `currentTime`, `moveTag()`, and `tmToTag()`.

**6.43.4.11** `std::tm cryomesh::state::PatternTagByDate::moveWeek ( std::tm & time,  
int i ) [static]`

Definition at line 156 of file PatternTagByDate.cpp.

References `moveDay()`.

Referenced by `moveTag()`.

**6.43.4.12** `std::tm cryomesh::state::PatternTagByDate::moveYear ( std::tm & time, int  
i ) [static]`

Definition at line 193 of file PatternTagByDate.cpp.

References `isLeapYear()`, and `moveDay()`.

Referenced by `moveMonth()`, and `moveTag()`.



6.43.4.13 `template<class Archive > void cryomesh::state::PatternTagByDate::serialize ( Archive & ar, const unsigned int version )`  
[inline]

Definition at line 28 of file PatternTagByDate.h.

References `currentTime`, `dateType`, `endTime`, `globalTag`, and `startTime`.

6.43.4.14 `void cryomesh::state::PatternTagByDate::setEndTag ( std::string tg )`  
[virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 73 of file PatternTagByDate.cpp.

References `endTime`, and `tagToTm()`.

6.43.4.15 `void cryomesh::state::PatternTagByDate::setStartTag ( std::string tg )`  
[virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 67 of file PatternTagByDate.cpp.

References `startTime`, and `tagToTm()`.

6.43.4.16 `void cryomesh::state::PatternTagByDate::setTag ( std::string tg )`  
[virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 40 of file PatternTagByDate.cpp.

References `currentTime`, and `tagToTm()`.

6.43.4.17 `std::tm cryomesh::state::PatternTagByDate::tagToTm ( const std::string & tg )`  
[static]

Definition at line 81 of file PatternTagByDate.cpp.

Referenced by `setEndTag()`, `setStartTag()`, and `setTag()`.

6.43.4.18 `std::string cryomesh::state::PatternTagByDate::tmToTag ( const std::tm & t )`  
[static]

Definition at line 132 of file PatternTagByDate.cpp.

References `DateFormat`.

Referenced by `getEndTag()`, `getStartTag()`, `getTag()`, and `moveTag()`.

### 6.43.5 Friends And Related Function Documentation

#### 6.43.5.1 friend class boost::serialization::access [friend]

Definition at line 26 of file PatternTagByDate.h.

### 6.43.6 Member Data Documentation

#### 6.43.6.1 std::tm cryomesh::state::PatternTagByDate::currentTime [private]

Definition at line 73 of file PatternTagByDate.h.

Referenced by getTag(), moveTag(), serialize(), and setTag().

#### 6.43.6.2 std::string cryomesh::state::PatternTagByDate::DateFormat = "%S %M %H %d %m %Y %w %j" [static, protected]

Definition at line 66 of file PatternTagByDate.h.

Referenced by tmToTag().

#### 6.43.6.3 DateType cryomesh::state::PatternTagByDate::dateType [private]

Definition at line 70 of file PatternTagByDate.h.

Referenced by moveTag(), and serialize().

#### 6.43.6.4 std::tm cryomesh::state::PatternTagByDate::endTime [private]

Definition at line 72 of file PatternTagByDate.h.

Referenced by getEndTag(), serialize(), and setEndTag().

#### 6.43.6.5 std::string cryomesh::state::PatternTagByDate::GlobalCurrentTag = "" [static, protected]

Definition at line 65 of file PatternTagByDate.h.

#### 6.43.6.6 std::string cryomesh::state::PatternTagByDate::GlobalEndTag = "" [static, protected]

Definition at line 64 of file PatternTagByDate.h.

Referenced by PatternTagByDate().

6.43.6.7 `std::string cryomesh::state::PatternTagByDate::GlobalStartTag = ""`  
`[static, protected]`

Definition at line 63 of file PatternTagByDate.h.

Referenced by PatternTagByDate().

6.43.6.8 `boost::shared_ptr< PatternTagByDate > cryomesh-`  
`::state::PatternTagByDate::globalTag [static,`  
`private]`

Definition at line 69 of file PatternTagByDate.h.

Referenced by getGlobalTag(), PatternTagByDate(), and serialize().

6.43.6.9 `std::tm cryomesh::state::PatternTagByDate::startTime [private]`

Definition at line 71 of file PatternTagByDate.h.

Referenced by getStartTag(), PatternTagByDate(), serialize(), and setStartTag().

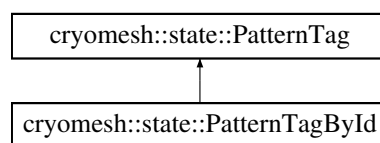
The documentation for this class was generated from the following files:

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagByDate.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagByDate.cpp](#)

## 6.44 cryomesh::state::PatternTagById Class Reference

```
#include <PatternTagById.h>
```

Inheritance diagram for cryomesh::state::PatternTagById:



### Public Member Functions

- [PatternTagById](#) (int d)
- virtual [~PatternTagById](#) ()
- virtual std::string [getTag](#) () const
- virtual void [setTag](#) (std::string tg)
- virtual std::string [moveTag](#) ()
- virtual std::string [moveTag](#) (int i)
- virtual std::string [getStartTag](#) () const

- virtual std::string [getEndTag](#) () const
- virtual void [setStartTag](#) (std::string tg)
- virtual void [setEndTag](#) (std::string tg)
- virtual boost::shared\_ptr < [PatternTag](#) > [getGlobalTag](#) ()

### Private Attributes

- std::string [id](#)

### Static Private Attributes

- static boost::shared\_ptr < [PatternTagByld](#) > [globalTag](#) = boost::shared\_ptr<[PatternTagByld](#)>(new [PatternTagByld](#)(1))

## 6.44.1 Detailed Description

Definition at line 14 of file [PatternTagByld.h](#).

## 6.44.2 Constructor & Destructor Documentation

### 6.44.2.1 cryomesh::state::PatternTagByld::PatternTagByld ( int *d* )

Definition at line 15 of file [PatternTagByld.cpp](#).

### 6.44.2.2 cryomesh::state::PatternTagByld::~PatternTagByld ( ) [virtual]

Definition at line 21 of file [PatternTagByld.cpp](#).

## 6.44.3 Member Function Documentation

### 6.44.3.1 std::string cryomesh::state::PatternTagByld::getEndTag ( ) const [virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 43 of file [PatternTagByld.cpp](#).

References [cryomesh::state::Pattern::getIds](#)().

Referenced by [setEndTag](#)().

### 6.44.3.2 boost::shared\_ptr< PatternTag > cryomesh::state::PatternTagByld::getGlobalTag ( ) [virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 56 of file PatternTagByld.cpp.

References `globalTag`.

**6.44.3.3** `std::string cryomesh::state::PatternTagByld::getStartTag ( ) const`  
[virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 40 of file PatternTagByld.cpp.

Referenced by `setStartTag()`.

**6.44.3.4** `std::string cryomesh::state::PatternTagByld::getTag ( ) const`  
[virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 23 of file PatternTagByld.cpp.

References `id`.

Referenced by `moveTag()`.

**6.44.3.5** `std::string cryomesh::state::PatternTagByld::moveTag ( )` [virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 29 of file PatternTagByld.cpp.

**6.44.3.6** `std::string cryomesh::state::PatternTagByld::moveTag ( int i )`  
[virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 32 of file PatternTagByld.cpp.

References `getTag()`.

**6.44.3.7** `void cryomesh::state::PatternTagByld::setEndTag ( std::string tg )`  
[virtual]

Implements [cryomesh::state::PatternTag](#).

Definition at line 53 of file PatternTagByld.cpp.

References `getEndTag()`.

**6.44.3.8** `void cryomesh::state::PatternTagByld::setStartTag ( std::string tg )`  
`[virtual]`

Implements [cryomesh::state::PatternTag](#).

Definition at line 50 of file PatternTagByld.cpp.

References [getStartTag\(\)](#).

**6.44.3.9** `void cryomesh::state::PatternTagByld::setTag ( std::string tg )`  
`[virtual]`

Implements [cryomesh::state::PatternTag](#).

Definition at line 26 of file PatternTagByld.cpp.

## 6.44.4 Member Data Documentation

**6.44.4.1** `boost::shared_ptr< PatternTagByld > cryomesh::state::PatternTagByld-  
::globalTag = boost::shared_ptr<PatternTagByld>(new PatternTagByld(1))`  
`[static, private]`

Definition at line 29 of file PatternTagByld.h.

Referenced by [getGlobalTag\(\)](#).

**6.44.4.2** `std::string cryomesh::state::PatternTagByld::id` `[private]`

Definition at line 30 of file PatternTagByld.h.

Referenced by [getTag\(\)](#).

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagByld.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/state/PatternTagByld.cpp](#)

## 6.45 cryomesh::Pointer< T > Struct Template Reference

[Pointer](#) struct to allow typedef of templated smart pointers.

```
#include <Defs.h>
```

### Public Types

- typedef `boost::scoped_ptr< T >` [scoped\\_ptr](#)
- typedef `boost::shared_ptr< T >` [shared\\_ptr](#)

### 6.45.1 Detailed Description

template<class T>struct cryomesh::Pointer< T >

[Pointer](#) struct to allow typedef of templated smart pointers.

Definition at line 23 of file Defs.h.

### 6.45.2 Member Typedef Documentation

6.45.2.1 template<class T > typedef boost::scoped\_ptr<T> cryomesh::Pointer< T >::scoped\_ptr

Definition at line 28 of file Defs.h.

6.45.2.2 template<class T > typedef boost::shared\_ptr<T> cryomesh::Pointer< T >::shared\_ptr

Definition at line 29 of file Defs.h.

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

- /home/niall/Projects/Eclipse/Cpp/cryomesh/src/[Defs.h](#)

## 6.46 cryomesh::manipulators::ClusterAnalysisData::RangeEnergy Struct Reference

Struct representing the value extrapolated over a history range.

```
#include <ClusterAnalysisData.h>
```

### Public Member Functions

- [RangeEnergy](#) ()
- [RangeEnergy](#) (double en, double energy\_fraction)
- [RangeEnergy](#) (double en, double energy\_fraction, [common::Cycle](#) st, [common::Cycle](#) ed, double min, double max)
- [RangeEnergy](#) (const [RangeEnergy](#) &obj)
- virtual [~RangeEnergy](#) ()
- const [RangeEnergy operator+](#) (const [RangeEnergy](#) &obj) const  
*Non-destructive addition operator.*
- [RangeEnergy & operator+=](#) (const [RangeEnergy](#) &obj)  
*Destructive addition and assignment operator.*
- const [RangeEnergy operator/](#) (double div) const
- [RangeEnergy & operator/=](#) (double div)

## Public Attributes

- double [energy](#)
- double [energyFraction](#)
- [common::Cycle](#) [startCycle](#)
- [common::Cycle](#) [endCycle](#)
- double [energyMin](#)
- double [energyMax](#)

## Friends

- `std::ostream & operator<< (std::ostream &os, const RangeEnergy &obj)`  
*To stream operator.*

### 6.46.1 Detailed Description

Struct representing the value extrapolated over a history range.

Definition at line 24 of file ClusterAnalysisData.h.

### 6.46.2 Constructor & Destructor Documentation

**6.46.2.1** `cryomesh::manipulators::ClusterAnalysisData::RangeEnergy::RangeEnergy ( ) \[inline\]`

Definition at line 25 of file ClusterAnalysisData.h.

**6.46.2.2** `cryomesh::manipulators::ClusterAnalysisData::RangeEnergy::RangeEnergy ( double en, double energy_fraction ) \[inline\]`

Definition at line 28 of file ClusterAnalysisData.h.

**6.46.2.3** `cryomesh::manipulators::ClusterAnalysisData::RangeEnergy::RangeEnergy ( double en, double energy_fraction, common::Cycle st, common::Cycle ed, double min, double max ) \[inline\]`

Definition at line 32 of file ClusterAnalysisData.h.

**6.46.2.4** `cryomesh::manipulators::ClusterAnalysisData::RangeEnergy::RangeEnergy ( const RangeEnergy & obj ) \[inline\]`

Definition at line 36 of file ClusterAnalysisData.h.



6.46.2.5 `virtual cryomesh::manipulators::ClusterAnalysisData-  
::RangeEnergy::~~RangeEnergy ( ) [inline,  
virtual]`

Definition at line 40 of file ClusterAnalysisData.h.

### 6.46.3 Member Function Documentation

6.46.3.1 `const RangeEnergy cryomesh::manipulators::ClusterAnalysisData-  
::RangeEnergy::operator+ ( const RangeEnergy & obj ) const  
[inline]`

Non-destructive addition operator.

#### Parameters

<i>const</i>	<a href="#">RangeEnergy</a> & obj RHS addition
--------------	--

#### Returns

[RangeEnergy](#) New object after addition

Definition at line 51 of file ClusterAnalysisData.h.

6.46.3.2 `RangeEnergy& cryomesh::manipulators::ClusterAnalysisData-  
::RangeEnergy::operator+= ( const RangeEnergy & obj )  
[inline]`

Destructive addition and assignment operator.

#### Parameters

<i>const</i>	<a href="#">RangeEnergy</a> & obj RHS addition
--------------	--

#### Returns

[RangeEnergy](#) & This object after addition and assignment

Definition at line 65 of file ClusterAnalysisData.h.

References `endCycle`, `energy`, `energyMax`, `energyMin`, and `startCycle`.

6.46.3.3 `const RangeEnergy cryomesh::manipulators::Cluster-  
AnalysisData::RangeEnergy::operator/ ( double div ) const  
[inline]`

Definition at line 75 of file ClusterAnalysisData.h.

#### 6.46.3.4 **RangeEnergy**& cryomesh::manipulators::ClusterAnalysisData::RangeEnergy- ::operator/= ( double *div* ) [inline]

Definition at line 80 of file ClusterAnalysisData.h.

References energy.

### 6.46.4 Friends And Related Function Documentation

#### 6.46.4.1 **std::ostream& operator<<** ( **std::ostream & os**, **const RangeEnergy & obj** ) [friend]

To stream operator.

##### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">RangeEnergy</a> & obj The object to stream

##### Returns

std::ostream & The output stream

Definition at line 96 of file ClusterAnalysisData.h.

### 6.46.5 Member Data Documentation

#### 6.46.5.1 **common::Cycle** cryomesh::manipulators::ClusterAnalysisData::Range- Energy::endCycle

Definition at line 104 of file ClusterAnalysisData.h.

Referenced by operator+=().

#### 6.46.5.2 **double** cryomesh::manipulators::ClusterAnalysisData::RangeEnergy- ::energy

Definition at line 101 of file ClusterAnalysisData.h.

Referenced by operator+=(), and operator/=().

#### 6.46.5.3 **double** cryomesh::manipulators::ClusterAnalysisData::RangeEnergy- ::energyFraction

Definition at line 102 of file ClusterAnalysisData.h.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster().

## 6.47 cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown Struct Reference 335

---

### 6.46.5.4 double cryomesh::manipulators::ClusterAnalysisData::RangeEnergy::energyMax

Definition at line 106 of file ClusterAnalysisData.h.

Referenced by operator+=().

### 6.46.5.5 double cryomesh::manipulators::ClusterAnalysisData::RangeEnergy::energyMin

Definition at line 105 of file ClusterAnalysisData.h.

Referenced by operator+=().

### 6.46.5.6 common::Cycle cryomesh::manipulators::ClusterAnalysisData::RangeEnergy::startCycle

Definition at line 103 of file ClusterAnalysisData.h.

Referenced by operator+=().

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/ClusterAnalysisData.h](#)

## 6.47 cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown Struct Reference

Class to hold together information on whether we can act to restructure items.

```
#include <IClusterAnalyser.h>
```

### Public Member Functions

- [RestructuringCountdown](#) (int ct=0)
- bool [isRestructuringEnabled](#) (int var) const
- bool [isAnyShortRestructuringEnabled](#) () const
- bool [isAnyMediumRestructuringEnabled](#) () const
- bool [isAnyLongRestructuringEnabled](#) () const
- bool [isAnyRestructuringEnabled](#) () const
- bool [isAllShortRestructuringEnabled](#) () const
- bool [isAllMediumRestructuringEnabled](#) () const
- bool [isAllLongRestructuringEnabled](#) () const
- bool [isAllRestructuringEnabled](#) () const
- [RestructuringCountdown](#) & [operator--](#) ()

*Prefix decrement operator.*

- void [setShortCountdown](#) (int ct)
- void [setMediumCountdown](#) (int ct)
- void [setLongCountdown](#) (int ct)

### Public Attributes

- int [shortCreation](#)
- int [shortDestruction](#)
- int [mediumCreation](#)
- int [mediumDestruction](#)
- int [longCreation](#)
- int [longDestruction](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [RestructuringCountdown](#) &obj)

*To stream operator.*

## 6.47.1 Detailed Description

Class to hold together information on whether we can act to restructure items.

Definition at line 23 of file IClusterAnalyser.h.

## 6.47.2 Constructor & Destructor Documentation

**6.47.2.1** `cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::RestructuringCountdown ( int ct = 0 )`  
`[inline]`

Definition at line 24 of file IClusterAnalyser.h.

## 6.47.3 Member Function Documentation

**6.47.3.1** `bool cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::isAllLongRestructuringEnabled ( ) const`  
`[inline]`

Definition at line 60 of file IClusterAnalyser.h.

References `isRestructuringEnabled()`, `longCreation`, and `longDestruction`.

**6.47.3.2** `bool cryomesh::manipulators::IClusterAnalyser::Restructuring-  
Countdown::isAllMediumRestructuringEnabled ( ) const`  
`[inline]`

Definition at line 56 of file IClusterAnalyser.h.

References `isRestructuringEnabled()`, `mediumCreation`, and `mediumDestruction`.

**6.47.3.3** `bool cryomesh::manipulators::IClusterAnalyser::Restructuring-  
Countdown::isAllRestructuringEnabled ( ) const`  
`[inline]`

Definition at line 64 of file IClusterAnalyser.h.

References `isAnyLongRestructuringEnabled()`, `isAnyMediumRestructuringEnabled()`, and `isAnyShortRestructuringEnabled()`.

**6.47.3.4** `bool cryomesh::manipulators::IClusterAnalyser::Restructuring-  
Countdown::isAllShortRestructuringEnabled ( ) const`  
`[inline]`

Definition at line 52 of file IClusterAnalyser.h.

References `isRestructuringEnabled()`, `shortCreation`, and `shortDestruction`.

**6.47.3.5** `bool cryomesh::manipulators::IClusterAnalyser::Restructuring-  
Countdown::isAnyLongRestructuringEnabled ( ) const`  
`[inline]`

Definition at line 43 of file IClusterAnalyser.h.

References `isRestructuringEnabled()`, `longCreation`, and `longDestruction`.

Referenced by `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`, `isAllRestructuringEnabled()`, and `isAnyRestructuringEnabled()`.

**6.47.3.6** `bool cryomesh::manipulators::IClusterAnalyser::Restructuring-  
Countdown::isAnyMediumRestructuringEnabled ( ) const`  
`[inline]`

Definition at line 39 of file IClusterAnalyser.h.

References `isRestructuringEnabled()`, `mediumCreation`, and `mediumDestruction`.

Referenced by `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`, `isAllRestructuringEnabled()`, and `isAnyRestructuringEnabled()`.

**6.47.3.7** `bool cryomesh::manipulators::IClusterAnalyser::Restructuring-  
Countdown::isAnyRestructuringEnabled ( ) const`  
[inline]

Definition at line 47 of file IClusterAnalyser.h.

References `isAnyLongRestructuringEnabled()`, `isAnyMediumRestructuringEnabled()`, and `isAnyShortRestructuringEnabled()`.

Referenced by `cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster()`.

**6.47.3.8** `bool cryomesh::manipulators::IClusterAnalyser::Restructuring-  
Countdown::isAnyShortRestructuringEnabled ( ) const`  
[inline]

Definition at line 35 of file IClusterAnalyser.h.

References `isRestructuringEnabled()`, `shortCreation`, and `shortDestruction`.

Referenced by `isAllRestructuringEnabled()`, and `isAnyRestructuringEnabled()`.

**6.47.3.9** `bool cryomesh::manipulators::IClusterAnalyser::Restructuring-  
Countdown::isRestructuringEnabled ( int var ) const`  
[inline]

Definition at line 28 of file IClusterAnalyser.h.

Referenced by `isAllLongRestructuringEnabled()`, `isAllMediumRestructuringEnabled()`, `isAllShortRestructuringEnabled()`, `isAnyLongRestructuringEnabled()`, `isAnyMediumRestructuringEnabled()`, and `isAnyShortRestructuringEnabled()`.

**6.47.3.10** `RestructuringCountdown& cryomesh::manipulators::-  
IClusterAnalyser::RestructuringCountdown::operator-- ( )`  
[inline]

Prefix decrement operator.

#### Returns

[RestructuringCountdown](#) & Return this

Definition at line 75 of file IClusterAnalyser.h.

References `longCreation`, `longDestruction`, `mediumCreation`, `mediumDestruction`, `shortCreation`, and `shortDestruction`.

**6.47.3.11** `void cryomesh::manipulators::IClusterAnalyser::-  
RestructuringCountdown::setLongCountdown ( int ct )`  
[inline]

Definition at line 105 of file IClusterAnalyser.h.

## 6.47 cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown Struct Reference 339

---

References longCreation, and longDestruction.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster().

**6.47.3.12** void cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::setMediumCountdown ( int *ct* )  
[inline]

Definition at line 101 of file IClusterAnalyser.h.

References mediumCreation, and mediumDestruction.

Referenced by cryomesh::manipulators::ClusterAnalyserBasic::analyseCluster().

**6.47.3.13** void cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::setShortCountdown ( int *ct* )  
[inline]

Definition at line 97 of file IClusterAnalyser.h.

References shortCreation, and shortDestruction.

## 6.47.4 Friends And Related Function Documentation

**6.47.4.1** std::ostream& operator<< ( std::ostream & *os*, const RestructuringCountdown & *obj* ) [friend]

To stream operator.

### Parameters

<i>std::ostream</i>	& <i>os</i> The output stream
<i>const</i>	<a href="#">RestructuringCountdown</a> & <i>obj</i> The object to stream

### Returns

std::ostream & The output stream

Definition at line 121 of file IClusterAnalyser.h.

## 6.47.5 Member Data Documentation

**6.47.5.1** int cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown::longCreation

Definition at line 132 of file IClusterAnalyser.h.

Referenced by isAllLongRestructuringEnabled(), isAnyLongRestructuringEnabled(), operator--(), and setLongCountdown().

#### 6.47.5.2 `int cryomesh::manipulators::IClusterAnalyser::Restructuring-Countdown::longDestruction`

Definition at line 133 of file `IClusterAnalyser.h`.

Referenced by `isAllLongRestructuringEnabled()`, `isAnyLongRestructuringEnabled()`, `operator--()`, and `setLongCountdown()`.

#### 6.47.5.3 `int cryomesh::manipulators::IClusterAnalyser::Restructuring-Countdown::mediumCreation`

Definition at line 130 of file `IClusterAnalyser.h`.

Referenced by `isAllMediumRestructuringEnabled()`, `isAnyMediumRestructuringEnabled()`, `operator--()`, and `setMediumCountdown()`.

#### 6.47.5.4 `int cryomesh::manipulators::IClusterAnalyser::Restructuring-Countdown::mediumDestruction`

Definition at line 131 of file `IClusterAnalyser.h`.

Referenced by `isAllMediumRestructuringEnabled()`, `isAnyMediumRestructuringEnabled()`, `operator--()`, and `setMediumCountdown()`.

#### 6.47.5.5 `int cryomesh::manipulators::IClusterAnalyser::Restructuring-Countdown::shortCreation`

Definition at line 128 of file `IClusterAnalyser.h`.

Referenced by `isAllShortRestructuringEnabled()`, `isAnyShortRestructuringEnabled()`, `operator--()`, and `setShortCountdown()`.

#### 6.47.5.6 `int cryomesh::manipulators::IClusterAnalyser::Restructuring-Countdown::shortDestruction`

Definition at line 129 of file `IClusterAnalyser.h`.

Referenced by `isAllShortRestructuringEnabled()`, `isAnyShortRestructuringEnabled()`, `operator--()`, and `setShortCountdown()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/IClusterAnalyser.h`

## 6.48 `cryomesh::state::Sequence` Class Reference

```
#include <Sequence.h>
```



## Public Member Functions

- [Sequence](#) ()
- virtual [~Sequence](#) ()
- [Sequence](#) (const [Sequence](#) &seq)
- [Sequence](#) (const std::vector< [Pattern](#) > &input\_pats, const std::vector< [Pattern](#) > &output\_pats)
- [Sequence](#) (std::ifstream &ifs)
- [Sequence](#) & [operator=](#) (const [Sequence](#) &seq)
- const [Pattern](#) & [getCurrentInputPattern](#) () const
- const [Pattern](#) & [getCurrentOutputPattern](#) () const
- const [Pattern](#) & [getAndAdvanceCurrentInputPattern](#) ()
- const [Pattern](#) & [getAndAdvanceCurrentOutputPattern](#) ()
- int [getCurrentInputPatternId](#) () const
- int [getCurrentOutputPatternId](#) () const
- std::list< std::pair< [Pattern](#), [Pattern](#) > >::const\_iterator [getCurrentIterator](#) () const
- void [setCurrentIterator](#) (std::list< std::pair< [Pattern](#), [Pattern](#) > >::const\_iterator)
- const [Pattern](#) & [getNextInputPattern](#) ()
- const std::list< std::pair< [Pattern](#), [Pattern](#) > > & [getPatterns](#) () const
- void [setPatterns](#) (const std::list< std::pair< [Pattern](#), [Pattern](#) > > &pats)
- void [saveToFile](#) (std::ofstream &ofs) const
- void [loadFromFile](#) (std::ifstream &ifs)
- void [addEntry](#) ([Pattern](#) new\_in\_pat, [Pattern](#) new\_out\_pat)
- void [clear](#) ()
- double [compareOutput](#) (const [Sequence](#) &seq) const
- double [compareInput](#) (const [Sequence](#) &seq) const
- double [compare](#) (const [Sequence](#) &seq) const
- double [compare](#) (int id, const [Pattern](#) &pat) const
- bool [operator==](#) (const [Sequence](#) &obj) const
- bool [isAllZeroes](#) () const
- bool [isInputAllZeroes](#) () const
- bool [isOutputAllZeroes](#) () const

## Static Public Attributes

- static const std::string [INPUT\\_TAG](#) = "<input>"
- static const std::string [OUTPUT\\_TAG](#) = "<output>"

## Private Member Functions

- template<class Archive >  
void [serialize](#) (Archive &ar, const unsigned int version)
- void [initialise](#) ()

### Private Attributes

- `std::list< std::pair< Pattern, Pattern > > patterns`
- `std::list< std::pair< Pattern, Pattern > >::const_iterator it\_patterns`

### Friends

- class [boost::serialization::access](#)
- `std::ostream & operator<< (std::ostream &os, const Sequence &obj)`

### 6.48.1 Detailed Description

Definition at line 26 of file `Sequence.h`.

### 6.48.2 Constructor & Destructor Documentation

#### 6.48.2.1 `cryomesh::state::Sequence::Sequence ( )`

Definition at line 17 of file `Sequence.cpp`.

References `initialise()`.

#### 6.48.2.2 `cryomesh::state::Sequence::~~Sequence ( )` `[virtual]`

Definition at line 20 of file `Sequence.cpp`.

#### 6.48.2.3 `cryomesh::state::Sequence::Sequence ( const Sequence & seq )`

Definition at line 22 of file `Sequence.cpp`.

References `initialise()`.

#### 6.48.2.4 `cryomesh::state::Sequence::Sequence ( const std::vector< Pattern > & input_pats, const std::vector< Pattern > & output_pats )`

Definition at line 25 of file `Sequence.cpp`.

References `initialise()`, and `patterns`.

#### 6.48.2.5 `cryomesh::state::Sequence::Sequence ( std::ifstream & ifs )`

Definition at line 44 of file `Sequence.cpp`.

References `initialise()`, and `loadFromFile()`.

### 6.48.3 Member Function Documentation

**6.48.3.1** void cryomesh::state::Sequence::addEntry ( Pattern *new\_in\_pat*, Pattern *new\_out\_pat* )

Definition at line 150 of file Sequence.cpp.

References patterns.

**6.48.3.2** void cryomesh::state::Sequence::clear ( )

Definition at line 156 of file Sequence.cpp.

References it\_patterns, and patterns.

**6.48.3.3** double cryomesh::state::Sequence::compare ( const Sequence & *seq* ) const

Definition at line 253 of file Sequence.cpp.

References getPatterns().

**6.48.3.4** double cryomesh::state::Sequence::compare ( int *id*, const Pattern & *pat* ) const

Definition at line 331 of file Sequence.cpp.

References getPatterns().

**6.48.3.5** double cryomesh::state::Sequence::compareInput ( const Sequence & *seq* ) const

Definition at line 209 of file Sequence.cpp.

References getPatterns().

**6.48.3.6** double cryomesh::state::Sequence::compareOutput ( const Sequence & *seq* ) const

Definition at line 163 of file Sequence.cpp.

References getPatterns().

**6.48.3.7** const Pattern & cryomesh::state::Sequence::getAndAdvanceCurrent-InputPattern ( )

Definition at line 62 of file Sequence.cpp.

References getCurrentIterator(), and it\_patterns.

#### 6.48.3.8 `const Pattern & cryomesh::state::Sequence::getAndAdvanceCurrentOutputPattern ( )`

Definition at line 70 of file Sequence.cpp.

References `getCurrentIterator()`, and `it_patterns`.

#### 6.48.3.9 `const Pattern & cryomesh::state::Sequence::getCurrentInputPattern ( ) const`

Definition at line 59 of file Sequence.cpp.

References `getCurrentIterator()`.

Referenced by `getNextInputPattern()`.

#### 6.48.3.10 `int cryomesh::state::Sequence::getCurrentInputPatternId ( ) const`

Definition at line 75 of file Sequence.cpp.

References `getCurrentIterator()`.

#### 6.48.3.11 `std::list< std::pair< Pattern, Pattern > >::const_iterator cryomesh::state::Sequence::getCurrentIterator ( ) const`

Definition at line 83 of file Sequence.cpp.

References `it_patterns`.

Referenced by `getAndAdvanceCurrentInputPattern()`, `getAndAdvanceCurrentOutputPattern()`, `getCurrentInputPattern()`, `getCurrentInputPatternId()`, `getCurrentOutputPattern()`, `getCurrentOutputPatternId()`, and `getNextInputPattern()`.

#### 6.48.3.12 `const Pattern & cryomesh::state::Sequence::getCurrentOutputPattern ( ) const`

Definition at line 67 of file Sequence.cpp.

References `getCurrentIterator()`.

#### 6.48.3.13 `int cryomesh::state::Sequence::getCurrentOutputPatternId ( ) const`

Definition at line 79 of file Sequence.cpp.

References `getCurrentIterator()`.

#### 6.48.3.14 `const Pattern & cryomesh::state::Sequence::getNextInputPattern ( )`

Definition at line 86 of file Sequence.cpp.

References `getCurrentInputPattern()`, `getCurrentIterator()`, `getPatterns()`, `it_patterns`, `patterns`, and `setCurrentIterator()`.

**6.48.3.15** `const std::list< std::pair< Pattern, Pattern > > & cryomesh::state::Sequence::getPatterns ( ) const`

Definition at line 93 of file `Sequence.cpp`.

References `patterns`.

Referenced by `compare()`, `compareInput()`, `compareOutput()`, `getNextInputPattern()`, `cryomesh::state::operator<<()`, `operator=()`, `operator==()`, and `saveToFile()`.

**6.48.3.16** `void cryomesh::state::Sequence::initialise ( ) [private]`

Definition at line 437 of file `Sequence.cpp`.

References `it_patterns`, and `patterns`.

Referenced by `loadFromFile()`, `operator=()`, `Sequence()`, and `setPatterns()`.

**6.48.3.17** `bool cryomesh::state::Sequence::isAllZeroes ( ) const`

Definition at line 392 of file `Sequence.cpp`.

References `it_patterns`, and `patterns`.

**6.48.3.18** `bool cryomesh::state::Sequence::isInputAllZeroes ( ) const`

Definition at line 407 of file `Sequence.cpp`.

References `it_patterns`, and `patterns`.

**6.48.3.19** `bool cryomesh::state::Sequence::isOutputAllZeroes ( ) const`

Definition at line 422 of file `Sequence.cpp`.

References `it_patterns`, and `patterns`.

**6.48.3.20** `void cryomesh::state::Sequence::loadFromFile ( std::ifstream & ifs )`

Definition at line 116 of file `Sequence.cpp`.

References `initialise()`, and `patterns`.

Referenced by `Sequence()`.

**6.48.3.21 Sequence & cryomesh::state::Sequence::operator= ( const Sequence & seq )**

Definition at line 50 of file Sequence.cpp.

References `getPatterns()`, `initialise()`, and `patterns`.

**6.48.3.22 bool cryomesh::state::Sequence::operator== ( const Sequence & obj ) const**

Definition at line 349 of file Sequence.cpp.

References `getPatterns()`.

**6.48.3.23 void cryomesh::state::Sequence::saveToFile ( std::ofstream & ofs ) const**

Definition at line 100 of file Sequence.cpp.

References `getPatterns()`.

**6.48.3.24 template<class Archive > void cryomesh::state::Sequence::serialize ( Archive & ar, const unsigned int version ) [inline, private]**

Definition at line 29 of file Sequence.h.

References `patterns`.

**6.48.3.25 void cryomesh::state::Sequence::setCurrentIterator ( std::list< std::pair< Pattern, Pattern > >::const\_iterator it )**

Definition at line 447 of file Sequence.cpp.

References `it_patterns`.

Referenced by `getNextInputPattern()`.

**6.48.3.26 void cryomesh::state::Sequence::setPatterns ( const std::list< std::pair< Pattern, Pattern > > & pats )**

Definition at line 96 of file Sequence.cpp.

References `initialise()`, and `patterns`.

**6.48.4 Friends And Related Function Documentation****6.48.4.1 friend class boost::serialization::access [friend]**

Definition at line 27 of file Sequence.h.

6.48.4.2 `std::ostream& operator<< ( std::ostream & os, const Sequence & obj )`  
`[friend]`

Definition at line 380 of file Sequence.cpp.

### 6.48.5 Member Data Documentation

6.48.5.1 `const std::string cryomesh::state::Sequence::INPUT_TAG = "<input>"`  
`[static]`

Definition at line 122 of file Sequence.h.

6.48.5.2 `std::list<std::pair<Pattern, Pattern> >::const_iterator`  
`cryomesh::state::Sequence::it_patterns [private]`

Definition at line 134 of file Sequence.h.

Referenced by `clear()`, `getAndAdvanceCurrentInputPattern()`, `getAndAdvanceCurrentOutputPattern()`, `getCurrentIterator()`, `getNextInputPattern()`, `initialise()`, `isAllZeroes()`, `isInputAllZeroes()`, `isOutputAllZeroes()`, and `setCurrentIterator()`.

6.48.5.3 `const std::string cryomesh::state::Sequence::OUTPUT_TAG = "<output>"`  
`[static]`

Definition at line 123 of file Sequence.h.

6.48.5.4 `std::list<std::pair<Pattern, Pattern> > cryomesh::state::Sequence::patterns [private]`

Definition at line 130 of file Sequence.h.

Referenced by `addEntry()`, `clear()`, `getNextInputPattern()`, `getPatterns()`, `initialise()`, `isAllZeroes()`, `isInputAllZeroes()`, `isOutputAllZeroes()`, `loadFromFile()`, `operator=()`, `Sequence()`, `serialize()`, and `setPatterns()`.

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

- `/home/niall/Projects/Eclipse/Cpp/cryomesh/src/state/Sequence.h`
- `/home/niall/Projects/Eclipse/Cpp/cryomesh/src/state/Sequence.cpp`

## 6.49 cryomesh::utilities::SequencerChannels Class Reference

```
#include <SequencerChannels.h>
```

## Public Member Functions

- [SequencerChannels](#) ()
- virtual [~SequencerChannels](#) ()
- void [readSequences](#) (const std::string &ifstr, [state::PatternChannelMap](#) &in\_channels, [state::PatternChannelMap](#) &out\_channels)
- void [writeSequences](#) (const std::string &ofstr, std::map< boost::uuids::uuid, boost::shared\_ptr< [state::PatternChannel](#) > > &in\_channels, std::map< boost::uuids::uuid, boost::shared\_ptr< [state::PatternChannel](#) > > &out\_channels) const

## Static Public Attributes

- static const std::string [PATTERN\\_CHANNEL\\_STRING](#) = "PatternChannel"  

```
std::map<boost::uuids::uuid, boost::shared_ptr<state::PatternChannel> > getInputChannels() const; std::map<boost::uuids::uuid, boost::shared_ptr<state::PatternChannel> > getOutputChannels() const;
```
- static const std::string [PATTERN\\_CHANNEL\\_TYPE\\_STRING](#) = "Type"
- static const std::string [PATTERN\\_CHANNEL\\_INPUT\\_STRING](#) = "Input"
- static const std::string [PATTERN\\_CHANNEL\\_OUTPUT\\_STRING](#) = "Output"
- static const std::string [PATTERN\\_CHANNEL\\_REFID\\_STRING](#) = "ReferenceID"
- static const std::string [PATTERN\\_STRING](#) = "Pattern"
- static const std::string [PATTERN\\_BINARY\\_STRING](#) = "Binary"
- static const std::string [PATTERN\\_TAG\\_STRING](#) = "Tag"
- static const std::string [VERSION\\_STRING](#) = "Version"
- static const std::string [DESCRIPTION\\_STRING](#) = "Description"
- static const std::string [PATTERN\\_CHANNEL\\_WIDTH\\_STRING](#) = "Width"
- static const std::string [PATTERN\\_CHANNEL\\_DEPTH\\_STRING](#) = "Depth"
- static const std::string [PATTERN\\_CHANNEL\\_NOTE\\_STRING](#) = "Note"

## Private Attributes

- std::map< boost::uuids::uuid, boost::shared\_ptr < [state::PatternChannel](#) > > [in\\_channels\\_filtered](#)  
*To stream operator.*
- std::map< boost::uuids::uuid, boost::shared\_ptr < [state::PatternChannel](#) > > [out\\_channels\\_filtered](#)

### 6.49.1 Detailed Description

Definition at line 21 of file SequencerChannels.h.



## 6.49.2 Constructor & Destructor Documentation

### 6.49.2.1 cryomesh::utilities::SequencerChannels::SequencerChannels ( )

Definition at line 32 of file SequencerChannels.cpp.

### 6.49.2.2 cryomesh::utilities::SequencerChannels::~SequencerChannels ( ) [virtual]

Definition at line 36 of file SequencerChannels.cpp.

## 6.49.3 Member Function Documentation

### 6.49.3.1 void cryomesh::utilities::SequencerChannels::readSequences ( const std::string & *ifstr*, state::PatternChannelMap & *in\_channels*, state::PatternChannelMap & *out\_channels* )

Definition at line 39 of file SequencerChannels.cpp.

References cryomesh::utilities::SequencerGeneric::NodeEntry::childNodes, cryomesh::utilities::SequencerGeneric::getNodeEntries(), cryomesh::utilities::SequencerGeneric::NodeEntry::info, cryomesh::state::PatternChannel::Input, cryomesh::utilities::SequencerGeneric::NodeEntry::name, cryomesh::state::PatternChannel::Output, PATTERN\_BINARY\_STRING, PATTERN\_CHANNEL\_INPUT\_STRING, PATTERN\_CHANNEL\_OUTPUT\_STRING, PATTERN\_CHANNEL\_REFID\_STRING, PATTERN\_CHANNEL\_STRING, and PATTERN\_CHANNEL\_TYPE\_STRING.

Referenced by cryomesh::structures::Bundle::loadChannels().

### 6.49.3.2 void cryomesh::utilities::SequencerChannels::writeSequences ( const std::string & *ofstr*, std::map< boost::uuids::uuid, boost::shared\_ptr< state::PatternChannel > > & *in\_channels*, std::map< boost::uuids::uuid, boost::shared\_ptr< state::PatternChannel > > & *out\_channels* ) const

## 6.49.4 Member Data Documentation

### 6.49.4.1 const std::string cryomesh::utilities::SequencerChannels::DESCRIPTION\_ - STRING = "Description" [static]

Definition at line 56 of file SequencerChannels.h.

### 6.49.4.2 std::map< boost::uuids::uuid, boost::shared\_ptr< state::PatternChannel > > > cryomesh::utilities::SequencerChannels::in\_channels\_filtered [private]

To stream operator.

## Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">SequencerChannels</a> & obj The object to stream

## Returns

`std::ostream` & The output stream

Definition at line 86 of file `SequencerChannels.h`.

```
6.49.4.3  std::map<boost::uuids::uuid, boost::shared_ptr<state::PatternChannel>
          > cryomesh::utilities::SequencerChannels::out_channels_filtered
          [private]
```

Definition at line 87 of file `SequencerChannels.h`.

```
6.49.4.4  const std::string cryomesh::utilities::SequencerChannels::PATTERN_BIN-
          ARY_STRING = "Binary"  [static]
```

Definition at line 53 of file `SequencerChannels.h`.

Referenced by `readSequences()`.

```
6.49.4.5  const std::string cryomesh::utilities::SequencerChannels-
          ::PATTERN_CHANNEL_DEPTH_STRING = "Depth"
          [static]
```

Definition at line 58 of file `SequencerChannels.h`.

```
6.49.4.6  const std::string cryomesh::utilities::SequencerChannels-
          ::PATTERN_CHANNEL_INPUT_STRING = "Input"
          [static]
```

Definition at line 49 of file `SequencerChannels.h`.

Referenced by `readSequences()`.

```
6.49.4.7  const std::string cryomesh::utilities::SequencerChannels-
          ::PATTERN_CHANNEL_NOTE_STRING = "Note"
          [static]
```

Definition at line 59 of file `SequencerChannels.h`.

**6.49.4.8** `const std::string cryomesh::utilities::SequencerChannels-  
::PATTERN_CHANNEL_OUTPUT_STRING = "Output"  
[static]`

Definition at line 50 of file SequencerChannels.h.

Referenced by readSequences().

**6.49.4.9** `const std::string cryomesh::utilities::SequencerChannels-  
::PATTERN_CHANNEL_REFID_STRING = "ReferenceID"  
[static]`

Definition at line 51 of file SequencerChannels.h.

Referenced by readSequences().

**6.49.4.10** `const std::string cryomesh::utilities::SequencerChannels-  
::PATTERN_CHANNEL_STRING = "PatternChannel"  
[static]`

`std::map<boost::uuids::uuid, boost::shared_ptr<state::PatternChannel> > getInput-  
Channels() const; std::map<boost::uuids::uuid, boost::shared_ptr<state::Pattern-  
Channel> > getOutputChannels() const;`

`state::PatternChannelMap getInputChannelsMap() const; state::PatternChannelMap  
getOutputChannelsMap() const;`

Definition at line 47 of file SequencerChannels.h.

Referenced by readSequences().

**6.49.4.11** `const std::string cryomesh::utilities::SequencerChannels-  
::PATTERN_CHANNEL_TYPE_STRING = "Type"  
[static]`

Definition at line 48 of file SequencerChannels.h.

Referenced by readSequences().

**6.49.4.12** `const std::string cryomesh::utilities::SequencerChannels-  
::PATTERN_CHANNEL_WIDTH_STRING = "Width"  
[static]`

Definition at line 57 of file SequencerChannels.h.

**6.49.4.13** `const std::string cryomesh::utilities::SequencerChannels::PATTERN_ST-  
RING = "Pattern" [static]`

Definition at line 52 of file SequencerChannels.h.

6.49.4.14 `const std::string cryomesh::utilities::SequencerChannels::PATTERN_TAG_STRING = "Tag" [static]`

Definition at line 54 of file SequencerChannels.h.

6.49.4.15 `const std::string cryomesh::utilities::SequencerChannels::VERSION_STRING = "Version" [static]`

Definition at line 55 of file SequencerChannels.h.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/SequencerChannels.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/SequencerChannels.cpp](#)

## 6.50 cryomesh::utilities::SequencerGeneric Class Reference

```
#include <SequencerGeneric.h>
```

### Classes

- struct [NodeEntry](#)

### Public Member Functions

- [SequencerGeneric](#) (const std::string &ifstr)
- virtual [~SequencerGeneric](#) ()
- const std::list< boost::shared\_ptr< [SequencerGeneric::NodeEntry](#) > > & [getNodeEntries](#) () const
- virtual void [on\\_start\\_document](#) ()
- virtual void [on\\_end\\_document](#) ()
- virtual void [on\\_start\\_element](#) (const Glib::ustring &name, const AttributeList &properties)
- virtual void [on\\_end\\_element](#) (const Glib::ustring &name)
- virtual void [on\\_comment](#) (const Glib::ustring &text)
- virtual void [on\\_warning](#) (const Glib::ustring &text)
- virtual void [on\\_error](#) (const Glib::ustring &text)
- virtual void [on\\_fatal\\_error](#) (const Glib::ustring &text)

### Private Attributes

- std::list< boost::shared\_ptr< [NodeEntry](#) > > [nodeEntries](#)
- std::list< boost::shared\_ptr< [NodeEntry](#) > > [nodeStack](#)
- int [elementCount](#)

## Friends

- `std::ostream & operator<< (std::ostream &os, const SequencerGeneric &obj)`

### 6.50.1 Detailed Description

Definition at line 12 of file SequencerGeneric.h.

### 6.50.2 Constructor & Destructor Documentation

**6.50.2.1** `cryomesh::utilities::SequencerGeneric::SequencerGeneric ( const std::string & ifstr )`

Definition at line 7 of file SequencerGeneric.cpp.

**6.50.2.2** `cryomesh::utilities::SequencerGeneric::~~SequencerGeneric ( )`  
[virtual]

Definition at line 19 of file SequencerGeneric.cpp.

### 6.50.3 Member Function Documentation

**6.50.3.1** `const std::list< boost::shared_ptr< SequencerGeneric::NodeEntry > > & cryomesh::utilities::SequencerGeneric::getNodeEntries ( ) const`

Definition at line 106 of file SequencerGeneric.cpp.

References `nodeEntries`.

Referenced by `cryomesh::utilities::SequencerChannels::readSequences()`.

**6.50.3.2** `void cryomesh::utilities::SequencerGeneric::on_comment ( const Glib::ustring & text )` [virtual]

Definition at line 70 of file SequencerGeneric.cpp.

**6.50.3.3** `void cryomesh::utilities::SequencerGeneric::on_end_document ( )`  
[virtual]

Definition at line 26 of file SequencerGeneric.cpp.

**6.50.3.4** `void cryomesh::utilities::SequencerGeneric::on_end_element ( const Glib::ustring & name )` [virtual]

Definition at line 65 of file SequencerGeneric.cpp.

References nodeStack.

**6.50.3.5** void cryomesh::utilities::SequencerGeneric::on\_error ( const Glib::ustring & *text* ) [virtual]

Definition at line 88 of file SequencerGeneric.cpp.

**6.50.3.6** void cryomesh::utilities::SequencerGeneric::on\_fatal\_error ( const Glib::ustring & *text* ) [virtual]

Definition at line 97 of file SequencerGeneric.cpp.

**6.50.3.7** void cryomesh::utilities::SequencerGeneric::on\_start\_document ( ) [virtual]

Definition at line 22 of file SequencerGeneric.cpp.

**6.50.3.8** void cryomesh::utilities::SequencerGeneric::on\_start\_element ( const Glib::ustring & *name*, const AttributeList & *properties* ) [virtual]

Definition at line 30 of file SequencerGeneric.cpp.

References nodeEntries, and nodeStack.

**6.50.3.9** void cryomesh::utilities::SequencerGeneric::on\_warning ( const Glib::ustring & *text* ) [virtual]

Definition at line 79 of file SequencerGeneric.cpp.

## 6.50.4 Friends And Related Function Documentation

**6.50.4.1** std::ostream& operator<< ( std::ostream & *os*, const SequencerGeneric & *obj* ) [friend]

Definition at line 109 of file SequencerGeneric.cpp.

## 6.50.5 Member Data Documentation

**6.50.5.1** int cryomesh::utilities::SequencerGeneric::elementCount [private]

Definition at line 56 of file SequencerGeneric.h.

### 6.50.5.2 `std::list< boost::shared_ptr<NodeEntry> > cryomesh::utilities::SequencerGeneric::nodeEntries` [private]

Definition at line 54 of file SequencerGeneric.h.

Referenced by `getNodeEntries()`, `on_start_element()`, and `cryomesh::utilities::operator<<()`.

### 6.50.5.3 `std::list< boost::shared_ptr<NodeEntry> > cryomesh::utilities::SequencerGeneric::nodeStack` [private]

Definition at line 55 of file SequencerGeneric.h.

Referenced by `on_end_element()`, and `on_start_element()`.

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/SequencerGeneric.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/SequencerGeneric.cpp](#)

## 6.51 cryomesh::utilities::Statistician Class Reference

Class to draw together lots of useful statistics and monitoring data for a Bundle and its components.

```
#include <Statistician.h>
```

### Public Member Functions

- [Statistician](#) (const [structures::Bundle](#) &bun)  
*Construct a statistician with a build in Bundle reference.*
- virtual [~Statistician](#) ()  
*Default destructor.*
- void [update](#) ()
- const [structures::Bundle](#) & [getBundle](#) () const
- int [getClusterCount](#) () const
- int [getInputFibresCount](#) () const
- int [getOutputFibresCount](#) () const
- int [getNormalFibresCount](#) () const
- int [getInputChannelsCount](#) () const
- int [getOutputChannelsCount](#) () const
- std::string [getBundleUUID](#) () const
- std::map< std::string, int > [getTriggeredNodesPerCluster](#) () const
- std::map< std::string, int > [getActiveNodesPerCluster](#) () const
- int [getTriggeredNodesTotal](#) () const
- int [getActiveNodesTotal](#) () const

### Private Attributes

- const `structures::Bundle` & `bundle`
- int `clusterCount`
- int `inputFibresCount`
- int `outputFibresCount`
- int `normalFibresCount`
- int `inputChannelsCount`
- int `outputChannelsCount`
- std::string `bundleuuid`
- int `nodesTotal`
- int `nodesTriggered`
- int `nodesActive`

### Friends

- std::ostream & `operator<<` (std::ostream &os, const `Statistician` &obj)  
*To stream operator.*

#### 6.51.1 Detailed Description

Class to draw together lots of useful statistics and monitoring data for a Bundle and its components.

Definition at line 28 of file Statistician.h.

#### 6.51.2 Constructor & Destructor Documentation

##### 6.51.2.1 `cryomesh::utilities::Statistician::Statistician ( const structures::Bundle & bun )`

Construct a statistician with a build in Bundle reference.

Definition at line 15 of file Statistician.cpp.

References `update()`.

##### 6.51.2.2 `cryomesh::utilities::Statistician::~Statistician ( ) [virtual]`

Default destructor.

Definition at line 21 of file Statistician.cpp.



### 6.51.3 Member Function Documentation

#### 6.51.3.1 `std::map< std::string, int > cryomesh::utilities::Statistician::getActiveNodesPerCluster ( ) const`

Definition at line 76 of file Statistician.cpp.

References `bundle`, and `cryomesh::structures::Bundle::getClusters()`.

Referenced by `getActiveNodesTotal()`, and `cryomesh::utilities::operator<<()`.

#### 6.51.3.2 `int cryomesh::utilities::Statistician::getActiveNodesTotal ( ) const`

Definition at line 110 of file Statistician.cpp.

References `getActiveNodesPerCluster()`.

Referenced by `cryomesh::utilities::operator<<()`.

#### 6.51.3.3 `const structures::Bundle & cryomesh::utilities::Statistician::getBundle ( ) const`

Definition at line 127 of file Statistician.cpp.

References `bundle`.

#### 6.51.3.4 `std::string cryomesh::utilities::Statistician::getBundleUUID ( ) const`

Definition at line 53 of file Statistician.cpp.

References `bundleuuid`.

Referenced by `cryomesh::utilities::operator<<()`.

#### 6.51.3.5 `int cryomesh::utilities::Statistician::getClusterCount ( ) const`

Definition at line 35 of file Statistician.cpp.

References `clusterCount`.

Referenced by `cryomesh::utilities::operator<<()`.

#### 6.51.3.6 `int cryomesh::utilities::Statistician::getInputChannelsCount ( ) const`

Definition at line 47 of file Statistician.cpp.

References `inputChannelsCount`.

**6.51.3.7 int cryomesh::utilities::Statistician::getInputFibresCount ( ) const**

Definition at line 38 of file Statistician.cpp.

References inputFibresCount.

**6.51.3.8 int cryomesh::utilities::Statistician::getNormalFibresCount ( ) const**

Definition at line 44 of file Statistician.cpp.

References normalFibresCount.

**6.51.3.9 int cryomesh::utilities::Statistician::getOutputChannelsCount ( ) const**

Definition at line 50 of file Statistician.cpp.

References outputChannelsCount.

**6.51.3.10 int cryomesh::utilities::Statistician::getOutputFibresCount ( ) const**

Definition at line 41 of file Statistician.cpp.

References outputFibresCount.

**6.51.3.11 std::map< std::string, int > cryomesh::utilities::Statistician::getTriggeredNodesPerCluster ( ) const**

Definition at line 57 of file Statistician.cpp.

References bundle, and cryomesh::structures::Bundle::getClusters().

Referenced by getTriggeredNodesTotal().

**6.51.3.12 int cryomesh::utilities::Statistician::getTriggeredNodesTotal ( ) const**

Definition at line 95 of file Statistician.cpp.

References getTriggeredNodesPerCluster().

Referenced by cryomesh::utilities::operator<<().

**6.51.3.13 void cryomesh::utilities::Statistician::update ( )**

Definition at line 24 of file Statistician.cpp.

References bundle, clusterCount, cryomesh::structures::Bundle::getClusters(), cryomesh::structures::Bundle::getFibres(), cryomesh::structures::Bundle::getInputFibres(), cryomesh::structures::Bundle::getOutputFibres(), cryomesh::structures::Bundle::getRealInputChannelsMap(), cryomesh::structures::Bundle::getRealOutput-

ChannelsMap(), inputChannelsCount, inputFibresCount, normalFibresCount, outputChannelsCount, and outputFibresCount.

Referenced by Statistician().

#### 6.51.4 Friends And Related Function Documentation

**6.51.4.1** `std::ostream& operator<< ( std::ostream & os, const Statistician & obj )`  
[friend]

To stream operator.

##### Parameters

<i>std::ostream</i>	& os The output stream
<i>const</i>	<a href="#">Statistician</a> & obj The object to stream

##### Returns

`std::ostream &` The output stream

Definition at line 131 of file Statistician.cpp.

#### 6.51.5 Member Data Documentation

**6.51.5.1** `const structures::Bundle& cryomesh::utilities::Statistician::bundle`  
[private]

Definition at line 74 of file Statistician.h.

Referenced by `getActiveNodesPerCluster()`, `getBundle()`, `getTriggeredNodesPerCluster()`, and `update()`.

**6.51.5.2** `std::string cryomesh::utilities::Statistician::bundleuuid` [private]

Definition at line 82 of file Statistician.h.

Referenced by `getBundleUUID()`.

**6.51.5.3** `int cryomesh::utilities::Statistician::clusterCount` [private]

Definition at line 76 of file Statistician.h.

Referenced by `getClusterCount()`, and `update()`.

**6.51.5.4** `int cryomesh::utilities::Statistician::inputChannelsCount` [private]

Definition at line 80 of file Statistician.h.

Referenced by `getInputChannelsCount()`, `cryomesh::utilities::operator<<()`, and `update()`.

#### 6.51.5.5 `int cryomesh::utilities::Statistician::inputFibresCount` [private]

Definition at line 77 of file `Statistician.h`.

Referenced by `getInputFibresCount()`, `cryomesh::utilities::operator<<()`, and `update()`.

#### 6.51.5.6 `int cryomesh::utilities::Statistician::nodesActive` [private]

Definition at line 85 of file `Statistician.h`.

#### 6.51.5.7 `int cryomesh::utilities::Statistician::nodesTotal` [private]

Definition at line 83 of file `Statistician.h`.

#### 6.51.5.8 `int cryomesh::utilities::Statistician::nodesTriggered` [private]

Definition at line 84 of file `Statistician.h`.

#### 6.51.5.9 `int cryomesh::utilities::Statistician::normalFibresCount` [private]

Definition at line 79 of file `Statistician.h`.

Referenced by `getNormalFibresCount()`, `cryomesh::utilities::operator<<()`, and `update()`.

#### 6.51.5.10 `int cryomesh::utilities::Statistician::outputChannelsCount` [private]

Definition at line 81 of file `Statistician.h`.

Referenced by `getOutputChannelsCount()`, `cryomesh::utilities::operator<<()`, and `update()`.

#### 6.51.5.11 `int cryomesh::utilities::Statistician::outputFibresCount` [private]

Definition at line 78 of file `Statistician.h`.

Referenced by `getOutputFibresCount()`, `cryomesh::utilities::operator<<()`, and `update()`.

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

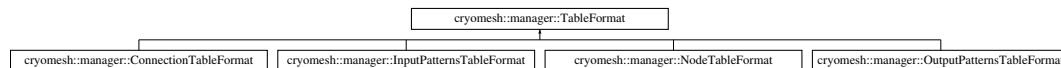
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/Statistician.h`
- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/Statistician.cpp`

## 6.52 cryomesh::manager::TableFormat Struct Reference

General structure of a table.

```
#include <TableFormats.h>
```

Inheritance diagram for cryomesh::manager::TableFormat:



### Public Member Functions

- [TableFormat](#) ()
- virtual [~TableFormat](#) ()
- std::string [getName](#) () const  
*Return the name of the table.*
- std::string [getKey](#) (const std::string &key)  
*Return the string object associated with a key.*
- std::string [getCreateTable](#) () const  
*Get the string that can be used to create the sql table.*

### Protected Attributes

- std::string [name](#)
- std::map< std::string, std::string > [columns](#)

#### 6.52.1 Detailed Description

General structure of a table.

Definition at line 22 of file TableFormats.h.

#### 6.52.2 Constructor & Destructor Documentation

##### 6.52.2.1 cryomesh::manager::TableFormat::TableFormat ( ) [inline]

Definition at line 24 of file TableFormats.h.

##### 6.52.2.2 virtual cryomesh::manager::TableFormat::~~TableFormat ( ) [inline, virtual]

Definition at line 25 of file TableFormats.h.

### 6.52.3 Member Function Documentation

**6.52.3.1** `std::string cryomesh::manager::TableFormat::getCreateTable ( ) const`  
[inline]

Get the string that can be used to create the sql table.

#### Returns

the sql command string to create this table

Definition at line 60 of file TableFormats.h.

References columns, and getName().

**6.52.3.2** `std::string cryomesh::manager::TableFormat::getKey ( const std::string & key`  
 ) [inline]

Return the string object associated with a key.

::string The key to search for

#### Returns

std::string The object associated with the search key, "" if not found

Definition at line 45 of file TableFormats.h.

References columns.

**6.52.3.3** `std::string cryomesh::manager::TableFormat::getName ( ) const`  
[inline]

Return the name of the table.

#### Returns

std::string The name of the table

Definition at line 32 of file TableFormats.h.

References name.

Referenced by getCreateTable(), cryomesh::manager::DatabaseManager::insert-Connection(), cryomesh::manager::DatabaseManager::insertNode(), and cryomesh::manager::DatabaseManager::insertOutputPattern().

### 6.52.4 Member Data Documentation

#### 6.52.4.1 `std::map<std::string, std::string> cryomesh::manager::TableFormat::columns` [protected]

Definition at line 93 of file TableFormats.h.

Referenced by `cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat()`, `getCreateTable()`, `getKey()`, `cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat()`, `cryomesh::manager::NodeTableFormat::NodeTableFormat()`, and `cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat()`.

#### 6.52.4.2 `std::string cryomesh::manager::TableFormat::name` [protected]

Definition at line 86 of file TableFormats.h.

Referenced by `cryomesh::manager::ConnectionTableFormat::ConnectionTableFormat()`, `getName()`, `cryomesh::manager::InputPatternsTableFormat::InputPatternsTableFormat()`, `cryomesh::manager::NodeTableFormat::NodeTableFormat()`, and `cryomesh::manager::OutputPatternsTableFormat::OutputPatternsTableFormat()`.

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

- `/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/`[TableFormats.h](#)

## 6.53 cryomesh::common::TimeKeeper Class Reference

[TimeKeeper](#) is a class keep track of the cycle state and timing.

```
#include <TimeKeeper.h>
```

### Public Member Functions

- void [reset](#) ()  
*Destructor.*
- virtual [~TimeKeeper](#) ()
- bool [operator==](#) (const [TimeKeeper](#) &)  
*Equality test override for object.*
- void [update](#) ()  
*Move the timing on one cycle.*
- [Cycle](#) [getCycle](#) () const  
*Get the current cycle we're on as an [Cycle](#).*
- std::time\_t [getStartTime](#) () const  
*Get the time we started cycling.*
- double [getTiming](#) () const  
*Difference between the last time and now.*
- const boost::timer & [getTimer](#) () const  
*Get the Timer.*

### Static Public Member Functions

- static [TimeKeeper](#) & [getTimeKeeper](#) ()

### Protected Member Functions

- [TimeKeeper](#) ()  
*Constructor.*
- [TimeKeeper](#) (const [TimeKeeper](#) &)  
*Copy Constructor.*
- [TimeKeeper](#) & [operator=](#) (const [TimeKeeper](#) &)  
*Assignment Operator.*

### Private Attributes

- [Cycle](#) [cycle](#)
- std::time\_t [start\\_time](#)
- boost::timer [timer](#)
- double [this\\_timing](#)
- double [last\\_timing](#)

### Static Private Attributes

- static boost::shared\_ptr < [TimeKeeper](#) > [timekeeper](#)

## 6.53.1 Detailed Description

[TimeKeeper](#) is a class keep track of the cycle state and timing.

[TimeKeeper](#) manages the 'tick' cycle and provides the means by which classes throughout the system can keep track of the cycle

Definition at line 31 of file [TimeKeeper.h](#).

## 6.53.2 Constructor & Destructor Documentation

### 6.53.2.1 [cryomesh::common::TimeKeeper::~TimeKeeper](#) ( ) [virtual]

Definition at line 37 of file [TimeKeeper.cpp](#).



**6.53.2.2 cryomesh::common::TimeKeeper::TimeKeeper ( )** [protected]

Constructor.

Constructor for [TimeKeeper](#). Inaccessible to force singleton class

Definition at line 34 of file TimeKeeper.cpp.

References [reset\(\)](#).

Referenced by [getTimeKeeper\(\)](#).

**6.53.2.3 cryomesh::common::TimeKeeper::TimeKeeper ( const TimeKeeper & )**  
[protected]

Copy Constructor.

Overridden Copy Constructor for [TimeKeeper](#). Inaccessible to force singleton class

**Parameters**

<a href="#">TimeKeeper</a>	Object to Copy Construct from
----------------------------	-------------------------------

**6.53.3 Member Function Documentation****6.53.3.1 Cycle cryomesh::common::TimeKeeper::getCycle ( ) const**

Get the current cycle we're on as an [Cycle](#).

**Returns**

[Cycle](#) The cycle we're currently on

Definition at line 76 of file TimeKeeper.cpp.

References [cycle](#).

Referenced by [cryomesh::manager::DatabaseManager::addHistoryEntry\(\)](#), [cryomesh::components::ImpulseCollection::clearActiveImpulses\(\)](#), [cryomesh::components::ImpulseCollection::clearImpulses\(\)](#), [cryomesh::components::ImpulseCollection::getActivity\(\)](#), [cryomesh::state::Pattern::getDatabaseObject\(\)](#), [cryomesh::components::Connection::getDatabaseObject\(\)](#), [cryomesh::components::Node::getDatabaseObject\(\)](#), [cryomesh::state::PatternChannel::getPatternByCycle\(\)](#), [cryomesh::utilities::operator<<\(\)](#), [cryomesh::structures::Bundle::print\(\)](#), [cryomesh::manager::DatabaseManager::printHistory\(\)](#), and [cryomesh::components::ImpulseCollection::refreshDataObject\(\)](#).

**6.53.3.2 std::time\_t cryomesh::common::TimeKeeper::getStartTime ( ) const**

Get the time we started cycling.

**Returns**

time\_t The start time

Definition at line 83 of file TimeKeeper.cpp.

References start\_time.

### 6.53.3.3 TimeKeeper & cryomesh::common::TimeKeeper::getTimeKeeper ( ) [static]

Definition at line 19 of file TimeKeeper.cpp.

References TimeKeeper(), and timekeeper.

Referenced by cryomesh::manager::DatabaseManager::addHistoryEntry(), cryomesh::components::Node::addImpulse(), cryomesh::components::ImpulseCollection::clearActiveImpulses(), cryomesh::components::ImpulseCollection::clearImpulses(), cryomesh::components::Node::enterRecovery(), cryomesh::components::ImpulseCollection::getActivity(), cryomesh::components::Impulse::getActivity(), cryomesh::components::Node::getActivity(), cryomesh::components::ConnectionMap::getActivityPattern(), cryomesh::state::Pattern::getDatabaseObject(), cryomesh::components::Connection::getDatabaseObject(), cryomesh::components::Node::getDatabaseObject(), cryomesh::state::PatternChannel::getPatternByCycle(), cryomesh::components::Impulse::isActive(), cryomesh::utilities::operator<<(), cryomesh::structures::Bundle::print(), cryomesh::manager::DatabaseManager::printHistory(), cryomesh::components::ImpulseCollection::refreshDataObject(), cryomesh::components::Node::setActivity(), and cryomesh::structures::Bundle::update().

### 6.53.3.4 const boost::timer & cryomesh::common::TimeKeeper::getTimer ( ) const

Get the Timer.

**Returns**

boost::Timer

Definition at line 87 of file TimeKeeper.cpp.

References timer.

### 6.53.3.5 double cryomesh::common::TimeKeeper::getTiming ( ) const

Difference between the last time and now.

**Returns**

double The difference between the clock now and the last clock

Definition at line 80 of file TimeKeeper.cpp.

References last\_timing, and this\_timing.

**6.53.3.6** `TimeKeeper& cryomesh::common::TimeKeeper::operator= ( const TimeKeeper & ) [protected]`

Assignment Operator.

Overridden Assignment Operator for [TimeKeeper](#). Inaccessible to force singleton class

#### Parameters

<a href="#">TimeKeeper</a>	Object to Assign this to
----------------------------	--------------------------

**6.53.3.7** `bool cryomesh::common::TimeKeeper::operator== ( const TimeKeeper & obj )`

Equality test override for object.

#### Parameters

<a href="#">TimeKeeper</a>	obj Object to compare this with
----------------------------	---------------------------------

#### Returns

bool True if equal, false otherwise

Definition at line 40 of file TimeKeeper.cpp.

References `cycle`, `last_timing`, `start_time`, and `this_timing`.

**6.53.3.8** `void cryomesh::common::TimeKeeper::reset ( )`

Destructor.

Destructor for [TimeKeeper](#) Reset the timekeeper

Definition at line 26 of file TimeKeeper.cpp.

References `cycle`, `last_timing`, `start_time`, `this_timing`, and `timer`.

Referenced by `TimeKeeper()`.

**6.53.3.9** `void cryomesh::common::TimeKeeper::update ( )`

Move the timing on one cycle.

Definition at line 66 of file TimeKeeper.cpp.

References `cycle`, `last_timing`, `this_timing`, and `timer`.

Referenced by `cryomesh::structures::Bundle::update()`.

## 6.53.4 Member Data Documentation

**6.53.4.1 Cycle cryomesh::common::TimeKeeper::cycle** [private]

Definition at line 159 of file TimeKeeper.h.

Referenced by getCycle(), operator==(), reset(), and update().

**6.53.4.2 double cryomesh::common::TimeKeeper::last\_timing** [private]

Definition at line 187 of file TimeKeeper.h.

Referenced by getTiming(), operator==(), reset(), and update().

**6.53.4.3 std::time\_t cryomesh::common::TimeKeeper::start\_time** [private]

Definition at line 166 of file TimeKeeper.h.

Referenced by getStartTime(), operator==(), and reset().

**6.53.4.4 double cryomesh::common::TimeKeeper::this\_timing** [private]

Definition at line 180 of file TimeKeeper.h.

Referenced by getTiming(), operator==(), reset(), and update().

**6.53.4.5 boost::shared\_ptr< TimeKeeper > cryomesh::common::TimeKeeper-  
::timekeeper** [static, private]

Definition at line 151 of file TimeKeeper.h.

Referenced by getTimeKeeper().

**6.53.4.6 boost::timer cryomesh::common::TimeKeeper::timer** [private]

Definition at line 173 of file TimeKeeper.h.

Referenced by getTimer(), reset(), and update().

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

- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/TimeKeeper.h](#)
- [/home/niall/Projects/Eclipse/CPP/cryomesh/src/common/TimeKeeper.cpp](#)

## Chapter 7

# File Documentation

### 7.1 /home/niall/Projects/Eclipse/CPP/cryomesh/src/common/- Connector.h File Reference

```
#include "common/Cycle.h" #include <map> #include <boost/shared-  
_ptr.hpp> #include <boost/uuid/uuid.hpp> #include <boost/uuid/uuid-  
_io.hpp> #include "common/Misc.h"
```

#### Classes

- class [cryomesh::common::Connector< U, T >](#)  
*[Connector](#) is a template to add connectable functionality between two classes.*

#### Namespaces

- namespace [cryomesh](#)  
*[Connector.h](#).*
- namespace [cryomesh::common](#)

### 7.2 /home/niall/Projects/Eclipse/CPP/cryomesh/src/common/- Cycle.cpp File Reference

```
#include "Cycle.h" #include <iostream>
```

#### Namespaces

- namespace [cryomesh](#)  
*[Connector.h](#).*

- namespace [cryomesh::common](#)

## Functions

- `std::ostream & cryomesh::common::operator<< (std::ostream &os, const Cycle &obj)`

## 7.3 /home/niall/Projects/Eclipse/CPP/cryomesh/src/common/- Cycle.h File Reference

```
#include <gmpxx.h>
```

## Classes

- class [cryomesh::common::Cycle](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::common](#)

## 7.4 /home/niall/Projects/Eclipse/CPP/cryomesh/src/common/- Loggable.h File Reference

```
#include <iostream>
```

## Classes

- class [cryomesh::common::Loggable](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::common](#)

## 7.5 /home/niall/Projects/Eclipse/CPP/cryomesh/src/common/TimeKeeper.cpp File Reference

```
#include "TimeKeeper.h" #include <iostream> #include <ctime> ×  
#include <time.h>
```

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::common](#)

## 7.6 /home/niall/Projects/Eclipse/CPP/cryomesh/src/common/TimeKeeper.h File Reference

```
#include "common/Cycle.h" #include <boost/shared_ptr.-  
hpp> #include <boost/serialization/shared_ptr.hpp> #include  
<boost/date_time/posix_time/posix_time.hpp> #include <boost/timer.-  
hpp> #include <time.h>
```

### Classes

- class [cryomesh::common::TimeKeeper](#)  
*TimeKeeper is a class keep track of the cycle state and timing.*

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::common](#)

## 7.7 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/ActivityTimer.h File Reference

```
#include <ostream>
```

### Classes

- class [cryomesh::components::ActivityTimer](#)  
*Simple interface class for activity timers.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

## 7.8 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/- ActivityTimerDistance.cpp File Reference

```
#include "ActivityTimerDistance.h"      #include "common/-  
Maths.h"
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

## Defines

- `#define` [ACTIVITYTIMERDISTANCE\\_DEBUG](#)

### 7.8.1 Define Documentation

#### 7.8.1.1 `#define` [ACTIVITYTIMERDISTANCE\\_DEBUG](#)

Definition at line 8 of file ActivityTimerDistance.cpp.

## 7.9 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/- ActivityTimerDistance.h File Reference

```
#include "ActivityTimer.h"  #include "common/Debuggable.-  
h" #include <boost/shared_ptr.hpp>
```

## Classes

- class [cryomesh::components::ActivityTimerDistance](#)



## 7.10

### /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/Connection.cpp File Reference

373

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

## 7.10 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/- Connection.cpp File Reference

```
#include "Connection.h"      #include "manager/Connection-  
DatabaseObject.h"
```

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

#### Functions

- `std::ostream & cryomesh::components::operator<< (std::ostream &os, const -  
Connection &obj)`

## 7.11 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/- Connection.h File Reference

```
#include "Node.h"  #include "common/Connector.h"  #include  
"common/Tagged.h"      #include "manager/DatabaseObject.h" x  
#include "common/Debuggable.h"
```

#### Classes

- class [cryomesh::components::Connection](#)  
[Connection](#) class to manage the transfer of Impulses between Nodes.

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

## 7.12 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/- ConnectionMap.h File Reference

```
#include "common/KeyMappedCollection.h" #include "common/-  
TimeKeeper.h" #include "state/ActivityPattern.h" #include  
"common/Cycle.h" #include <boost/uuid/uuid.hpp> #include  
<boost/shared_ptr.hpp> #include "Connection.h"
```

### Classes

- class [cryomesh::components::ConnectionMap](#)  
*Helper class for [ConnectionMap](#) to [KeyMappedCollection](#) mapping.*

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

## 7.13 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/- Impulse.cpp File Reference

```
#include "Impulse.h" #include "ActivityTimerDistance.h" ×  
#include "common/Maths.h" #include "common/TimeKeeper.h"  
#include <algorithm>
```

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

### Functions

- `std::ostream & cryomesh::components::operator<< (std::ostream &os, const -  
Impulse &obj)`

## 7.14 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/-Impulse.h File Reference

```
#include "ActivityTimerDistance.h"      #include <common/-  
SimpleCollection.h> #include "common/Tagged.h" #include  
"common/Cycle.h" #include "common/TimeKeeper.h" #include  
<common/Debuggable.h> #include <list>
```

### Classes

- class [cryomesh::components::Impulse](#)  
*Impulse is a mobile information packet to be passed between Nodes.*

### Namespaces

- namespace [cryomesh](#)  
*Connector.h.*
- namespace [cryomesh::components](#)

## 7.15 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/-ImpulseCollection.cpp File Reference

```
#include "ImpulseCollection.h"          #include "common/Time-  
Keeper.h" #include "common/Maths.h"
```

### Namespaces

- namespace [cryomesh](#)  
*Connector.h.*
- namespace [cryomesh::components](#)

### Functions

- `std::ostream & cryomesh::components::operator<< (std::ostream &os, const -  
ImpulseCollection &obj)`

## 7.16 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/-ImpulseCollection.h File Reference

```
#include "Impulse.h" #include "common/Cycle.h" #include  
"common/KeyMappedCollection.h" #include "dataobjects/-
```

```
DataObjectController.h" #include <boost/uuid/uuid.hpp> ×
#include "common/Debuggable.h" #include <map>
```

## Classes

- class [cryomesh::components::ImpulseCollection](#)  
*ImpulseCollection* represents a collection of *Impulse* objects.

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

## 7.17 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/-Node.cpp File Reference

```
#include "Node.h" #include "structures/Mesh.h" #include
"components/Connection.h" #include "common/TimeKeeper.h"
#include "common/Maths.h"
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::components](#)

## Functions

- `std::ostream & cryomesh::components::operator<< (std::ostream &os, const Node &obj)`

## 7.18 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/-Node.h File Reference

```
#include "ImpulseCollection.h" #include "common/Cycle.h"
#include "common/Connector.h" #include "common/Tagged.-
h" #include "common/Debuggable.h" #include "common/Defs.-
h" #include "spacial/Point.h" #include "dataobjects/Data-
ObjectController.h" #include "manager/NodeDatabaseObject.-
h" #include <list> #include <map>
```

## Classes

- class [cryomesh::components::Node](#)  
*Node is an accumulation and computational nodal point of impulses.*

## Namespaces

- namespace [cryomesh](#)  
*Connector.h.*
- namespace [cryomesh::components](#)

## 7.19 /home/niall/Projects/Eclipse/CPP/cryomesh/src/components/- NodeMap.h File Reference

```
#include "common/KeyMappedCollection.h" #include <boost/uuid/uuid.-  
hpp> #include <boost/shared_ptr.hpp> #include "Node.h"
```

## Classes

- class [cryomesh::components::NodeMap](#)  
*Helper class for [NodeMap](#) to [KeyMappedCollection](#) mapping.*

## Namespaces

- namespace [cryomesh](#)  
*Connector.h.*
- namespace [cryomesh::components](#)

## 7.20 /home/niall/Projects/Eclipse/CPP/cryomesh/src/dataobjects/- DataObject.h File Reference

```
#include <map> #include <ostream>
```

## Classes

- class [cryomesh::dataobjects::DataObject< U, T >](#)  
*Class to contain all the useful data about an object.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::dataobjects](#)

## 7.21 /home/niall/Projects/Eclipse/CPP/cryomesh/src/dataobjects/-DataObjectController.h File Reference

```
#include "DataObject.h"
```

## Classes

- class [cryomesh::dataobjects::DataObjectController< U, T >](#)  
*Class used to interface with data objects.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::dataobjects](#)

## 7.22 /home/niall/Projects/Eclipse/CPP/cryomesh/src/Defs.h File - Reference

```
#include <boost/shared_ptr.hpp> #include <boost/scoped_ptr.hpp>
```

## Classes

- struct [cryomesh::Pointer< T >](#)  
*[Pointer](#) struct to allow typedef of templated smart pointers.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).

## 7.23 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/ConnectionDatabaseObject.cpp File Reference

379

### Typedefs

- typedef std::map < boost::shared\_ptr < components::Node >, std::map < boost::shared\_ptr < components::Node > > > [NeighbourhoodMap](#)  
[Defs.h](#).

### 7.22.1 Typedef Documentation

- 7.22.1.1 typedef std::map<boost::shared\_ptr< components::Node >, std::map<boost::shared\_ptr< components::Node > > > [NeighbourhoodMap](#)

[Defs.h](#).

Created on: 26 Jan 2011 Author: SevenMachines<[SevenMachines@yahoo.co.uk](mailto:SevenMachines@yahoo.co.uk)>

Definition at line 14 of file Defs.h.

## 7.23 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/-ConnectionDatabaseObject.cpp File Reference

```
#include "ConnectionDatabaseObject.h"
```

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

## 7.24 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/-ConnectionDatabaseObject.h File Reference

```
#include "DatabaseObject.h"          #include "spacial/Point.h"
#include "common/Cycle.h" #include <string> #include <sstream>
```

### Classes

- class [cryomesh::manager::ConnectionDatabaseObject](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.25 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- Creator.cpp File Reference

```
#include "Creator.h" #include <algorithm> #include "common/-  
Containers.h" #include <boost/uuid/uuid_io.hpp>
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.26 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- Creator.h File Reference

```
#include "config/ConfigTranslator.h" #include <string> ×  
#include <list> #include <boost/shared_ptr.hpp> #include  
<structures/Bundle.h>
```

## Classes

- class [cryomesh::manager::Creator](#)  
*Class to take in a config file of ConfigTranslator form and parse the commands to  
create a full cryomesh object.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.27 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- CryoManager.cpp File Reference

```
#include "CryoManager.h"
```



## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

## 7.28 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- CryoManager.h File Reference

```
#include <iostream> #include <ctime> #include "Creator.-  
h" #include "common/TimeKeeper.h" #include "PatternDatabase-  
Object.h" #include "DatabaseManager.h" #include <boost/uuid/uuid-  
_io.hpp> #include <sstream>
```

## Defines

- #define [CRYOMANAGER\\_DEBUG](#)

### 7.28.1 Define Documentation

#### 7.28.1.1 #define CRYOMANAGER\_DEBUG

Definition at line 8 of file CryoManager.h.

## 7.29 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- DatabaseManager.cpp File Reference

```
#include "DatabaseManager.h" #include "common/TimeKeeper.-  
h" #include "common/Containers.h" #include <iostream>  
#include <sstream> #include <fstream> #include <algorithm> x
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.30 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- DatabaseManager.h File Reference

```
#include "TableFormats.h"    #include "NodeDatabaseObject.-  
h" #include "ConnectionDatabaseObject.h" #include "common/-  
Cycle.h" #include "common/TimeKeeper.h" #include <sqlite3.-  
h> #include <string> #include <boost/shared_ptr.hpp> ×  
#include <map> #include <sstream>
```

#### Classes

- class [cryomesh::manager::DatabaseManager](#)  
*Database manager creates and maintains a database of mesh related objects and data.*

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.31 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- DatabaseObject.h File Reference

```
#include <map> #include <string> #include <sstream> ×  
#include <iostream> #include <boost/tokenizer.hpp>
```

#### Classes

- class [cryomesh::manager::DatabaseObject](#)

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.32 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- NodeDatabaseObject.cpp File Reference

```
#include "NodeDatabaseObject.h"
```

### 7.33

/home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/NodeDatabaseObject.h

File Reference

383

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.33 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- NodeDatabaseObject.h File Reference

```
#include "DatabaseObject.h"      #include "spacial/Point.-  
h" #include "common/Cycle.h"    #include <string> #include  
<sstream>
```

#### Classes

- class [cryomesh::manager::NodeDatabaseObject](#)

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.34 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- PatternDatabaseObject.cpp File Reference

```
#include "PatternDatabaseObject.h" #include "NodeDatabase-  
Object.h" #include "state/Pattern.h"
```

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.35 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- PatternDatabaseObject.h File Reference

```
#include "DatabaseObject.h"      #include "common/Cycle.h" ×  
#include <string> #include <boost/shared_ptr.hpp>
```

## Classes

- class [cryomesh::manager::PatternDatabaseObject](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)
- namespace [cryomesh::manager](#)

### 7.36 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manager/- TableFormats.h File Reference

```
#include <string> #include <map> #include <sstream>
```

## Classes

- struct [cryomesh::manager::TableFormat](#)  
*General structure of a table.*
- struct [cryomesh::manager::NodeTableFormat](#)  
*Struct representing a node table structure.*
- struct [cryomesh::manager::ConnectionTableFormat](#)  
*Struct representing a connections table structure.*
- struct [cryomesh::manager::InputPatternsTableFormat](#)  
*Struct representing input pattern table structure.*
- struct [cryomesh::manager::OutputPatternsTableFormat](#)  
*Struct representing output pattern table structure.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manager](#)

### 7.37 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/- ClusterAnalyserBasic.cpp File Reference

```
#include "ClusterAnalyserBasic.h" #include "ClusterArchitect.h"
```

### 7.38 /home/niall/Projects/Eclipse/\_CPP/cryomesh/src/manipulators/ClusterAnalyserBasic.h File Reference

385

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manipulators](#)

### 7.38 /home/niall/Projects/Eclipse/\_CPP/cryomesh/src/manipulators/ClusterAnalyserBasic.h File Reference

```
#include "IClusterAnalyser.h"
```

#### Classes

- class [cryomesh::manipulators::ClusterAnalyserBasic](#)

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manipulators](#)

### 7.39 /home/niall/Projects/Eclipse/\_CPP/cryomesh/src/manipulators/ClusterAnalysisData.cpp File Reference

```
#include "ClusterAnalysisData.h"
```

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manipulators](#)

### 7.40 /home/niall/Projects/Eclipse/\_CPP/cryomesh/src/manipulators/ClusterAnalysisData.h File Reference

```
#include "common/Tagged.h"          #include "common/Cycle.h" ×  
#include "common/TimeKeeper.h"
```

## Classes

- class [cryomesh::manipulators::ClusterAnalysisData](#)
- struct [cryomesh::manipulators::ClusterAnalysisData::RangeEnergy](#)

*Struct representing the value extrapolated over a history range.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manipulators](#)

### 7.41 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/- ClusterArchitect.cpp File Reference

```
#include "ClusterArchitect.h" #include "ClusterAnalyser-  
Basic.h" #include "common/Containers.h" #include <boost/shared-  
_ptr.hpp> #include <vector> #include <components/Node.-  
h> #include <components/NodeMap.h> #include "structures/-  
Fibre.h" #include <algorithm>
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manipulators](#)

### 7.42 /home/niall/Projects/Eclipse/CPP/cryomesh/src/manipulators/- ClusterArchitect.h File Reference

```
#include "ClusterAnalysisData.h" #include "IClusterAnalyser.-  
h" #include "structures/Cluster.h" #include "common/-  
Cycle.h" #include <map> #include <list> #include <set>
```

## Classes

- class [cryomesh::manipulators::ClusterArchitect](#)

### 7.43

/home/niall/Projects/Eclipse/\_CPP/cryomesh/src/manipulators/IClusterAnalyser.h

#### File Reference

387

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manipulators](#)

### 7.43 /home/niall/Projects/Eclipse/\_CPP/cryomesh/src/manipulators/- IClusterAnalyser.h File Reference

```
#include "ClusterAnalysisData.h"    #include "structures/-  
Cluster.h"
```

#### Classes

- class [cryomesh::manipulators::IClusterAnalyser](#)
- struct [cryomesh::manipulators::IClusterAnalyser::RestructuringCountdown](#)  
*Class to hold together information on whether we can act to restructure items.*
- struct [cryomesh::manipulators::IClusterAnalyser::EnergyVariationWeightingMap](#)

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manipulators](#)

### 7.44 /home/niall/Projects/Eclipse/\_CPP/cryomesh/src/state/Activity- Pattern.cpp File Reference

```
#include "ActivityPattern.h"    #include <list>    #include  
"common/Maths.h" #include "common/Misc.h" #include <sstream> ×
```

#### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## 7.45 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Activity-Pattern.h File Reference

```
#include "common/SimpleCollection.h"
```

### Classes

- class [cryomesh::state::ActivityPattern](#)  
*A simple collection of doubles representing a pattern of activities.*

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## 7.46 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Binary-String.cpp File Reference

```
#include "BinaryString.h" #include "common/Misc.h" #include  
<cstdio> #include <iostream> #include <assert.h>
```

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

### Functions

- `std::ostream & cryomesh::state::operator<< (std::ostream &os, const Binary-String &obj)`

## 7.47 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Binary-String.h File Reference

```
#include <list> #include <string> #include <sstream> ×  
#include <vector> #include <boost/serialization/string.-  
hpp> #include <boost/serialization/serialization.hpp>
```



## Classes

- class [cryomesh::state::BinaryString](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## 7.48 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern.cpp File Reference

```
#include "Pattern.h" #include "PatternTagById.h" #include  
"common/Misc.h" #include "common/Maths.h" #include "common/-  
TimeKeeper.h" #include <sstream> #include <iostream>
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## Functions

- `std::ostream & cryomesh::state::operator<< (std::ostream &os, const Pattern &obj)`

## 7.49 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern.h File Reference

```
#include "PatternTag.h" #include "BinaryString.h" #include  
"common/Tagged.h" #include <vector> #include <string>  
#include <boost/serialization/vector.hpp> #include <boost/uuid/uuid.-  
hpp> #include <boost/uuid/uuid_generators.hpp> #include  
"manager/PatternDatabaseObject.h"
```

## Classes

- class [cryomesh::state::Pattern](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

### 7.50 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern-Channel.cpp File Reference

```
#include "PatternChannel.h"      #include <boost/foreach.-  
hpp> #include <boost/uuid/uuid_generators.hpp> #include  
"common/TimeKeeper.h"
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## Functions

- `std::ostream & cryomesh::state::operator<< (std::ostream &os, const Pattern-Channel &obj)`

### 7.51 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern-Channel.h File Reference

```
#include "Pattern.h"      #include "PatternTag.h"      #include  
"common/Tagged.h" #include "common/Debuggable.h" #include  
<map> #include <list> #include <set> #include <boost/uuid/uuid.-  
hpp> #include <boost/shared_ptr.hpp>
```

## Classes

- class [cryomesh::state::PatternChannel](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## 7.52 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern-ChannelMap.h File Reference

```
#include "PatternChannel.h"    #include "common/KeyMapped-  
Collection.h" #include "common/TimeKeeper.h"
```

### Classes

- class [cryomesh::state::PatternChannelMap](#)

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## 7.53 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern-Tag.h File Reference

```
#include <string> #include <boost/shared_ptr.hpp>
```

### Classes

- class [cryomesh::state::PatternTag](#)

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## 7.54 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern-TagByDate.cpp File Reference

```
#include "PatternTagByDate.h" #include <boost/tokenizer.-  
hpp> #include <iostream> #include <sstream>
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

### 7.55 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern-TagByDate.h File Reference

```
#include "PatternTag.h" #include <ctime> #include <string> ×  
#include <boost/serialization/vector.hpp> #include <boost/serialization/ma  
hpp> #include <boost/serialization/shared_ptr.hpp>
```

## Classes

- class [cryomesh::state::PatternTagByDate](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

### 7.56 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern-TagById.cpp File Reference

```
#include "PatternTagById.h" #include "Pattern.h" #include  
<sstream>
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

### 7.57 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Pattern-TagById.h File Reference

```
#include "PatternTag.h"
```

## Classes

- class [cryomesh::state::PatternTagById](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## 7.58 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Sequence.cpp File Reference

```
#include "Sequence.h" #include <algorithm> #include <sstream> ×  
#include <boost/scoped_ptr.hpp>
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## Functions

- `std::ostream & cryomesh::state::operator<< (std::ostream &os, const Sequence &obj)`

## 7.59 /home/niall/Projects/Eclipse/CPP/cryomesh/src/state/Sequence.h File Reference

```
#include "Pattern.h" #include <vector> #include <fstream> ×  
#include <ostream> #include <string> #include <list>  
#include <boost/serialization/map.hpp> #include <boost/serialization/string.-  
hpp> #include <boost/serialization/list.hpp>
```

## Classes

- class [cryomesh::state::Sequence](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::state](#)

## 7.60 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- Bundle.cpp File Reference

```
#include "Bundle.h" #include "utilities/SequencerChannels.-  
h" #include <boost/uuid/uuid_io.hpp>
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::structures](#)

## Functions

- `std::ostream & cryomesh::structures::operator<< (std::ostream &os, const -  
Bundle &obj)`

## 7.61 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- Bundle.h File Reference

```
#include <boost/shared_ptr.hpp> #include <boost/uuid/uuid.-  
hpp> #include <iostream> #include "ClusterMap.h" #include  
"FibreMap.h" #include "state/PatternChannelMap.h" #include  
"common/Debuggable.h" #include "common/Tagged.h" #include  
"common/Loggable.h" #include "utilities/Statistician.h"
```

## Classes

- class [cryomesh::structures::Bundle](#)  
*A [Bundle](#) is the collection of clusters and fibres, it represents the system as a whole.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::structures](#)

## 7.62 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- Cluster.cpp File Reference

```
#include "Cluster.h" #include "manipulators/ClusterArchitect.-  
h" #include <list> #include <algorithm> #include "common/-  
Maths.h" #include "Fibre.h"
```

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::structures](#)

### Functions

- `std::ostream & cryomesh::structures::operator<< (std::ostream &os, const -  
Cluster &obj)`

## 7.63 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- Cluster.h File Reference

```
#include "components/NodeMap.h" #include "components/-  
ConnectionMap.h" #include "spacial/Spacial.h" #include  
"spacial/Point.h" #include <structures/NodeMesh.h> #include  
<boost/uuid/uuid.hpp> #include <boost/shared_ptr.hpp>  
#include <components/Node.h> #include <components/Connection.-  
h> #include "common/Connector.h" #include "common/Tagged.-  
h" #include "common/Debuggable.h"
```

### Classes

- class [cryomesh::structures::Cluster](#)  
*A [Cluster](#) is a collection of self-contained nodes and connections along with an asso-  
ciated [Mesh](#), that can be connected up to one another.*

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::manipulators](#)
- namespace [cryomesh::structures](#)

## 7.64 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/-ClusterMap.h File Reference

```
#include "common/KeyMappedCollection.h" #include "Cluster.-h"
```

### Defines

- #define [CLUSTERMAP\\_DEBUG](#)

#### 7.64.1 Define Documentation

##### 7.64.1.1 #define CLUSTERMAP\_DEBUG

Definition at line 8 of file ClusterMap.h.

## 7.65 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/-Fibre.cpp File Reference

```
#include "Fibre.h" #include "components/Connection.h" ×  
#include "Cluster.h"
```

### Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::structures](#)

### Functions

- std::ostream & [cryomesh::structures::operator<<](#) (std::ostream &os, const Fibre &obj)

## 7.66 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/-Fibre.h File Reference

```
#include "components/ConnectionMap.h" #include "components/-  
Node.h" #include "common/Connector.h" #include "state/-  
Pattern.h" #include <boost/shared_ptr.hpp> #include "common/-  
Tagged.h"
```



## Classes

- class [cryomesh::structures::Fibre](#)  
*A [Fibre](#) is a collection of connections that connect one structure to another.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::structures](#)

## 7.67 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- FibreMap.h File Reference

```
#include "common/KeyMappedCollection.h" #include "Fibre.-  
h"
```

## Classes

- class [cryomesh::structures::FibreMap](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::structures](#)

## 7.68 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- Mesh.cpp File Reference

```
#include "Mesh.h" #include "components/Node.h" #include  
"components/Impulse.h" #include "components/ImpulseCollection.-  
h" #include "Cluster.h"
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::structures](#)

## 7.69 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- Mesh.h File Reference

```
#include "common/Cycle.h" #include "spacial/ActivityGrid.-  
h" #include <boost/shared_ptr.hpp>
```

### Classes

- class [cryomesh::structures::Mesh](#)  
*[Mesh](#) is the fabric of connection space and warps and is warped by it.*

### Namespaces

- namespace [cryomesh](#)  
*[Connector.h](#).*
- namespace [cryomesh::components](#)
- namespace [cryomesh::structures](#)

## 7.70 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- NodeMesh.cpp File Reference

```
#include "NodeMesh.h" #include "Cluster.h" #include <cmath> ×
```

### Namespaces

- namespace [cryomesh](#)  
*[Connector.h](#).*
- namespace [cryomesh::structures](#)

### Functions

- `std::ostream & cryomesh::structures::operator<< (std::ostream &os, const -  
NodeMesh &obj)`

## 7.71 /home/niall/Projects/Eclipse/CPP/cryomesh/src/structures/- NodeMesh.h File Reference

```
#include "components/Node.h" #include <map> #include <boost/shared-  
_ptr.hpp>
```

## 7.72

/home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/SequencerChannels.cpp  
File Reference 399  
Classes

---

- class [cryomesh::structures::NodeMesh](#)  
*Mesh of nodes and their neighbouring nodes and distances.*
- struct [cryomesh::structures::NodeMesh::NeighbourhoodRanges](#)  
*Struct to capture some statistics data on a nodes neighbourhood.*

## Namespaces

- namespace [cryomesh](#)  
*Connector.h.*
- namespace [cryomesh::structures](#)

## Typedefs

- typedef `std::map < boost::shared_ptr < cryomesh::components::Node > , std::map< boost::shared_ptr < cryomesh::components::Node > , double > > cryomesh::structures::NeighbourhoodMap`  
*Typedef to simplify neighbourhood map structure.*
- typedef `std::map < boost::shared_ptr < cryomesh::components::Node > , std::map< boost::shared_ptr < cryomesh::components::Node > , double > >::const_iterator cryomesh::structures::NeighbourhoodMapConstIterator`  
*Typedef for iterator to neighbourhood map.*

## 7.72 /home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/- SequencerChannels.cpp File Reference

```
#include "SequencerChannels.h"
```

## Namespaces

- namespace [cryomesh](#)  
*Connector.h.*
- namespace [cryomesh::utilities](#)

## 7.73 /home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/- SequencerChannels.h File Reference

```
#include "state/PatternChannel.h" #include "state/Pattern.-  
h" #include "state/PatternChannelMap.h" #include "Sequencer-  
Generic.h" #include <map> #include <boost/uuid/uuid.hpp>
```

## Classes

- class [cryomesh::utilities::SequencerChannels](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::utilities](#)

### 7.74 /home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/- SequencerGeneric.cpp File Reference

```
#include "SequencerGeneric.h" #include <glibmm/convert.-  
h> #include <iostream>
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::utilities](#)

## Functions

- `std::ostream & cryomesh::utilities::operator<< (std::ostream &os, const -  
SequencerGeneric &obj)`

### 7.75 /home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/- SequencerGeneric.h File Reference

```
#include <map> #include <list> #include <libxml++-2.-  
6/libxml++/libxml++.h> #include <fstream> #include <iostream> ×  
#include <boost/shared_ptr.hpp>
```

## Classes

- class [cryomesh::utilities::SequencerGeneric](#)
- struct [cryomesh::utilities::SequencerGeneric::NodeEntry](#)

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::utilities](#)

## 7.76 /home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/- Statistician.cpp File Reference

```
#include "Statistician.h" #include "structures/Bundle.h"
```

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::utilities](#)

## Functions

- `std::ostream & cryomesh::utilities::operator<< (std::ostream &os, const Statistician &obj)`

## 7.77 /home/niall/Projects/Eclipse/CPP/cryomesh/src/utilities/- Statistician.h File Reference

```
#include <iostream> #include <map> #include <string>
```

## Classes

- class [cryomesh::utilities::Statistician](#)  
*Class to draw together lots of useful statistics and monitoring data for a Bundle and its components.*

## Namespaces

- namespace [cryomesh](#)  
[Connector.h](#).
- namespace [cryomesh::structures](#)
- namespace [cryomesh::utilities](#)