Scalable PaleoGeomorphology Model (SPGM)

Generated by Doxygen 1.7.6.1

Mon Jan 25 2016 10:38:12

Contents

1 Namespace Index **1**

1.1 Namespace List 1

2 Class Index **3**

2.1 Class Hierarchy 3

3 Class Index **5**

3.1 Class List 5

4 File Index **7**

4.1 File List 7

5 Namespace Documentation **9**

5.1 src Namespace Reference 9

5.2 src::geometry Namespace Reference 9

5.2.1 Function Documentation 10

5.2.1.1 FABS 10

5.3 src::math Namespace Reference 10

5.4 src::mem Namespace Reference 10

5.4.1 Function Documentation 11

5.4.1.1 FinalisePool 11

5.4.1.2 InitPool 11

5.4.1.3 operator!= 11

5.4.1.4 operator== 11

5.4.2 Variable Documentation 11

5.4.2.1 CHUNK_ARRAY_DELTA 11

5.4.2.2 FAILURE 11

5.4.2.3 INVALID . . . 11
5.4.2.4 mutex . . . 11
5.4.2.5 pools . . . 11
5.4.2.6 SUCCESS . . . 12
5.4.2.7 TWO_EXP16 . . . 12
5.5 src::mesh Namespace Reference . . . 12
5.6 src::model Namespace Reference . . . 12
5.7 src::parser Namespace Reference . . . 12
5.7.1 Enumeration Type Documentation . . . 13
5.7.1.1 LogLevel . . . 13
5.7.2 Function Documentation . . . 13
5.7.2.1 debugBreak . . . 13
5.7.2.2 split . . . 13
5.7.3 Variable Documentation . . . 13
5.7.3.1 logLevel . . . 13
5.8 src::util Namespace Reference . . . 13

6 Class Documentation **15**

6.1 src::mem::Chunk Struct Reference . . . 15
6.1.1 Detailed Description . . . 15
6.1.2 Member Data Documentation . . . 15
6.1.2.1 chunkId . . . 15
6.1.2.2 freeList . . . 15
6.1.2.3 memory . . . 15
6.1.2.4 numFree . . . 16
6.2 src::parser::Config Class Reference . . . 16
6.2.1 Detailed Description . . . 16
6.2.2 Constructor & Destructor Documentation . . . 16
6.2.2.1 Config . . . 16
6.2.2.2 ~Config . . . 16
6.2.3 Member Function Documentation . . . 16
6.2.3.1 GetConfigName . . . 16
6.2.3.2 GetGroups . . . 17
6.2.3.3 GetSymbols . . . 17

6.2.3.4 Group . . . 17
6.2.3.5 PBool . . . 17
6.2.3.6 PDouble . . . 17
6.2.3.7 PInt . . . 17
6.2.3.8 PString . . . 17
6.3 src::math::Diffusion::Coord_t Struct Reference . . . 17
6.3.1 Detailed Description . . . 18
6.3.2 Constructor & Destructor Documentation . . . 18
6.3.2.1 Coord_t . . . 18
6.3.3 Member Data Documentation . . . 18
6.3.3.1 x . . . 18
6.3.3.2 y . . . 18
6.4 src::math::Diffusion Class Reference . . . 18
6.4.1 Detailed Description . . . 19
6.4.2 Member Typedef Documentation . . . 19
6.4.2.1 Coord . . . 19
6.4.2.2 ForcingFunc . . . 19
6.4.2.3 NeumannFunc . . . 19
6.4.3 Constructor & Destructor Documentation . . . 19
6.4.3.1 Diffusion . . . 19
6.4.3.2 ∼Diffusion . . . 19
6.4.4 Member Function Documentation . . . 19
6.4.4.1 GetSolution . . . 19
6.4.4.2 SetCoefficient . . . 19
6.4.4.3 SetDirichlet . . . 20
6.4.4.4 SetIC . . . 20
6.4.4.5 Step . . . 20
6.4.5 Member Data Documentation . . . 20
6.4.5.1 m_solutions . . . 20
6.5 src::geometry::Edge Class Reference . . . 20
6.5.1 Detailed Description . . . 21
6.5.2 Constructor & Destructor Documentation . . . 21
6.5.2.1 Edge . . . 21
6.5.3 Member Function Documentation . . . 21

6.5.3.1 Dest . . . 21
6.5.3.2 Dest_Set . . . 21
6.5.3.3 Dnext . . . 21
6.5.3.4 Dprev . . . 22
6.5.3.5 Lnext . . . 22
6.5.3.6 Lprev . . . 23
6.5.3.7 Onext . . . 23
6.5.3.8 Onext_Set . . . 24
6.5.3.9 Oprev . . . 24
6.5.3.10 Org . . . 25
6.5.3.11 Org_Set . . . 25
6.5.3.12 Qedge . . . 25
6.5.3.13 Rnext . . . 25
6.5.3.14 Rot . . . 26
6.5.3.15 Rprev . . . 26
6.5.3.16 Sym . . . 27
6.5.3.17 Tor . . . 27
6.5.3.18 VDest . . . 28
6.5.3.19 VDest_Set . . . 28
6.5.3.20 VOrg . . . 28
6.5.3.21 VOrg_Set . . . 28
6.5.4 Friends And Related Function Documentation . . . 29
6.5.4.1 QuadEdge . . . 29
6.6 src::util::Field Class Reference . . . 29
6.6.1 Detailed Description . . . 30
6.6.2 Constructor & Destructor Documentation . . . 30
6.6.2.1 Field . . . 30
6.6.2.2 ∼Field . . . 30
6.6.3 Member Function Documentation . . . 30
6.6.3.1 GetLength . . . 30
6.6.3.2 GetName . . . 30
6.6.4 Member Data Documentation . . . 30
6.6.4.1 m_length . . . 30
6.6.4.2 m_name . . . 30

6.7 src::model::FluvialErosion Class Reference 30
6.7.1 Detailed Description 33
6.7.2 Constructor & Destructor Documentation 33
6.7.2.1 FluvialErosion 33
6.7.2.2 ∼FluvialErosion 33
6.7.3 Member Function Documentation 33
6.7.3.1 Execute 33
6.8 src::model::FluvialErosionDeposition Class Reference 33
6.8.1 Detailed Description 36
6.8.2 Constructor & Destructor Documentation 36
6.8.2.1 FluvialErosionDeposition 36
6.8.2.2 ∼FluvialErosionDeposition 36
6.8.3 Member Function Documentation 36
6.8.3.1 Execute 36
6.9 src::model::HillSlope Class Reference 36
6.9.1 Detailed Description 39
6.9.2 Constructor & Destructor Documentation 39
6.9.2.1 HillSlope 39
6.9.2.2 ∼HillSlope 39
6.9.3 Member Function Documentation 39
6.9.3.1 CoefficientFunction 39
6.9.3.2 DirichletFunction 39
6.9.3.3 Execute 39
6.10 src::mesh::KdItem Class Reference 39
6.10.1 Detailed Description 40
6.10.2 Constructor & Destructor Documentation 40
6.10.2.1 KdItem 40
6.10.2.2 ∼KdItem 40
6.10.3 Member Function Documentation 40
6.10.3.1 Print 40
6.10.4 Friends And Related Function Documentation 40
6.10.4.1 KdTree 40
6.10.5 Member Data Documentation 40
6.10.5.1 m_coord 40

6.10.5.2 m_id . . . 40
6.11 src::mesh::KdNode Class Reference . . . 41
6.11.1 Detailed Description . . . 41
6.11.2 Constructor & Destructor Documentation . . . 41
6.11.2.1 KdNode . . . 41
6.11.2.2 KdNode . . . 41
6.11.2.3 ∼KdNode . . . 41
6.11.3 Member Function Documentation . . . 41
6.11.3.1 Add . . . 41
6.11.3.2 Contains . . . 41
6.11.3.3 DeleteHelper . . . 41
6.11.3.4 Expand . . . 41
6.11.3.5 Intersects . . . 42
6.11.3.6 Print . . . 42
6.11.3.7 Range . . . 42
6.11.3.8 Split . . . 42
6.12 src::mesh::KdTree Class Reference . . . 42
6.12.1 Detailed Description . . . 42
6.12.2 Constructor & Destructor Documentation . . . 42
6.12.2.1 KdTree . . . 42
6.12.2.2 ∼KdTree . . . 42
6.12.3 Member Function Documentation . . . 43
6.12.3.1 Add . . . 43
6.12.3.2 DeleteAll . . . 43
6.12.3.3 Print . . . 43
6.12.3.4 QueryBallPoint . . . 43
6.12.3.5 Size . . . 43
6.12.4 Friends And Related Function Documentation . . . 43
6.12.4.1 src::mesh::KdNode . . . 43
6.13 src::mem::MemoryPool Class Reference . . . 43
6.13.1 Detailed Description . . . 43
6.13.2 Member Typedef Documentation . . . 44
6.13.2.1 MemoryPoolMode . . . 44
6.13.3 Member Enumeration Documentation . . . 44

6.13.3.1 MemoryPoolMode_t 44
6.13.4 Constructor & Destructor Documentation 44
6.13.4.1 MemoryPool 44
6.13.4.2 ∼MemoryPool 45
6.13.5 Member Function Documentation 45
6.13.5.1 DeleteObject 45
6.13.5.2 GetDataPointer 46
6.13.5.3 GetNumFree 46
6.13.5.4 NewObject 47
6.14 src::model::Model Class Reference 48
6.14.1 Detailed Description 48
6.14.2 Constructor & Destructor Documentation 49
6.14.2.1 Model 49
6.14.2.2 ∼Model 49
6.14.3 Member Function Documentation 49
6.14.3.1 AddField 49
6.14.3.2 AddProcess 49
6.14.3.3 GetDt 49
6.14.3.4 GetField 49
6.14.3.5 GetNParallelCores 49
6.14.3.6 GetNumTimeSteps 49
6.14.3.7 GetNumTimeSteps 49
6.14.3.8 GetProcess 49
6.14.3.9 GetProcessCount 49
6.14.3.10 GetSurfaceTopology 49
6.14.3.11 GetSurfaceTopologyOutput 49
6.14.3.12 GetTime 49
6.14.3.13 GetTimeStep 49
6.14.3.14 NextTimeStep 49
6.14.3.15 RegisterSurfaceTopologyOutput 49
6.15 src::model::ModelBuilder Class Reference 50
6.15.1 Detailed Description 50
6.15.2 Constructor & Destructor Documentation 50
6.15.2.1 ModelBuilder 50

6.15.2.2 ~ModelBuilder 50
6.15.3 Member Function Documentation 50
6.15.3.1 GetModel 50
6.15.3.2 GetSurfaceTopology 50
6.15.3.3 GetSurfaceTopologyOutput 50
6.16 src::mem::PoolAllocator< T > Class Template Reference 51
6.16.1 Detailed Description 52
6.16.2 Member Typedef Documentation 52
6.16.2.1 const_pointer 52
6.16.2.2 const_reference 53
6.16.2.3 difference_type 53
6.16.2.4 pointer 53
6.16.2.5 reference 53
6.16.2.6 size_type 53
6.16.2.7 value_type 53
6.16.3 Constructor & Destructor Documentation 53
6.16.3.1 PoolAllocator 53
6.16.3.2 PoolAllocator 53
6.16.3.3 PoolAllocator 54
6.16.3.4 ~PoolAllocator 54
6.16.4 Member Function Documentation 54
6.16.4.1 address 54
6.16.4.2 address 54
6.16.4.3 allocate 54
6.16.4.4 construct 55
6.16.4.5 deallocate 55
6.16.4.6 destroy 56
6.16.4.7 max_size 56
6.16.5 Member Data Documentation 56
6.16.5.1 pool 56
6.17 src::mem::PoolAllocator< void > Class Template Reference 56
6.17.1 Detailed Description 56
6.18 src::model::Precipitation Class Reference 57
6.18.1 Detailed Description 59

6.18.2 Constructor & Destructor Documentation 59
6.18.2.1 Precipitation 59
6.18.2.2 ∼Precipitation 59
6.18.3 Member Function Documentation 59
6.18.3.1 Execute 59
6.19 src::model::Process Class Reference 59
6.19.1 Detailed Description 61
6.19.2 Constructor & Destructor Documentation 61
6.19.2.1 Process 61
6.19.2.2 ∼Process 61
6.19.3 Member Function Documentation 61
6.19.3.1 Execute 61
6.19.4 Member Data Documentation 61
6.19.4.1 m_config 61
6.19.4.2 m_frequency 61
6.19.4.3 m_model 61
6.20 src::geometry::QuadEdge Class Reference 61
6.20.1 Detailed Description 62
6.20.2 Constructor & Destructor Documentation 62
6.20.2.1 QuadEdge 62
6.20.3 Member Function Documentation 62
6.20.3.1 DecrementVisited 62
6.20.3.2 IncrementVisited 63
6.20.3.3 IsFree 63
6.20.3.4 ResetVisited 63
6.20.3.5 SetFree 63
6.20.3.6 SetInUse 63
6.20.3.7 Visited 64
6.20.4 Friends And Related Function Documentation 64
6.20.4.1 Triangulator 64
6.21 src::mem::PoolAllocator< T >::rebind< U > Struct Template Reference 64
6.21.1 Detailed Description 64
6.21.2 Member Typedef Documentation 64
6.21.2.1 other 64

6.22 src::mesh::RegularMesh Class Reference 65
6.22.1 Detailed Description 65
6.22.2 Member Typedef Documentation 65
6.22.2.1 MatrixRM 65
6.22.3 Constructor & Destructor Documentation 65
6.22.3.1 RegularMesh 65
6.22.3.2 ∼RegularMesh 65
6.22.4 Member Function Documentation 65
6.22.4.1 GetFunctionValuesAt 66
6.22.4.2 Print 66
6.22.4.3 UpdateInterpolator 66
6.22.4.4 V 66
6.22.4.5 X 66
6.22.4.6 Y 66
6.22.5 Friends And Related Function Documentation 66
6.22.5.1 SurfaceTopology 66
6.23 src::util::ScalarField< T > Class Template Reference 66
6.23.1 Detailed Description 68
6.23.2 Constructor & Destructor Documentation 68
6.23.2.1 ScalarField 68
6.23.2.2 ∼ScalarField 69
6.23.3 Member Function Documentation 69
6.23.3.1 operator() 69
6.24 src::geometry::Site Class Reference 69
6.24.1 Detailed Description 69
6.24.2 Member Function Documentation 70
6.24.2.1 CCW 70
6.24.2.2 CCW 70
6.24.2.3 Circumcenter 70
6.24.2.4 InCircle 71
6.24.3 Friends And Related Function Documentation 71
6.24.3.1 operator<< 72
6.24.4 Member Data Documentation 72
6.24.4.1 m_coord 72

6.24.4.2 m_id 72
6.25 src::geometry::SiteComparator Class Reference 72
6.25.1 Detailed Description 72
6.25.2 Member Function Documentation 72
6.25.2.1 operator() 72
6.26 src::mesh::SurfaceTopology Class Reference 73
6.26.1 Detailed Description 76
6.26.2 Member Typedef Documentation 76
6.26.2.1 CatchmentIterator 76
6.26.3 Constructor & Destructor Documentation 76
6.26.3.1 SurfaceTopology 76
6.26.3.2 ∼SurfaceTopology 76
6.26.4 Member Function Documentation 76
6.26.4.1 B 76
6.26.4.2 C 76
6.26.4.3 CatchmentsBegin 77
6.26.4.4 CatchmentsEnd 77
6.26.4.5 D 77
6.26.4.6 Dn 77
6.26.4.7 GetAverageCellArea 77
6.26.4.8 GetBounds 77
6.26.4.9 GetHull 77
6.26.4.10 GetLowerBound 78
6.26.4.11 GetNeighbours 78
6.26.4.12 GetNMeshPoints 78
6.26.4.13 GetNumFaces 78
6.26.4.14 GetNumNeighbours 78
6.26.4.15 GetNumTriangles 78
6.26.4.16 GetNumVoronoiVertices 79
6.26.4.17 GetTriangleIndices 79
6.26.4.18 GetVoronoiCellAreas 79
6.26.4.19 GetVoronoiSides 79
6.26.4.20 GetVoronoiVertices 79
6.26.4.21 InterpolateToRegularmesh 79

6.26.4.22 O . . . 79
6.26.4.23 PrintNode . . . 80
6.26.4.24 R . . . 80
6.26.4.25 S . . . 80
6.26.4.26 SavePreviousTimestep . . . 80
6.26.4.27 SR . . . 80
6.26.4.28 UpdateNetwork . . . 80
6.26.4.29 UpdateZ . . . 80
6.26.4.30 X . . . 80
6.26.4.31 Y . . . 80
6.26.4.32 Z . . . 80
6.26.4.33 Z0 . . . 81
6.26.4.34 Zp . . . 81
6.26.5 Friends And Related Function Documentation . . . 81
6.26.5.1 SurfaceTopologyOutput . . . 81
6.26.5.2 Triangulator . . . 81
6.26.6 Member Data Documentation . . . 81
6.26.6.1 CYCLIC . . . 81
6.26.6.2 DIRICHLET . . . 81
6.26.6.3 m_kdTree . . . 81
6.26.6.4 NEUMANN . . . 81
6.27 src::mesh::SurfaceTopologyOutput Class Reference . . . 82
6.27.1 Detailed Description . . . 82
6.27.2 Member Typedef Documentation . . . 82
6.27.2.1 Attributes . . . 82
6.27.3 Member Enumeration Documentation . . . 82
6.27.3.1 Attributes_t . . . 82
6.27.4 Constructor & Destructor Documentation . . . 83
6.27.4.1 SurfaceTopologyOutput . . . 83
6.27.4.2 ~SurfaceTopologyOutput . . . 83
6.27.5 Member Function Documentation . . . 83
6.27.5.1 RegisterScalarField . . . 83
6.27.5.2 Write . . . 83
6.27.6 Friends And Related Function Documentation . . . 83

6.27.6.1 SurfaceTopology 83
6.28 src::util::Timer Class Reference 83
6.28.1 Detailed Description 83
6.28.2 Constructor & Destructor Documentation 83
6.28.2.1 Timer 84
6.28.2.2 ∼Timer 84
6.28.3 Member Function Documentation 84
6.28.3.1 Elapsed 84
6.29 src::util::TimeSeries Class Reference 84
6.29.1 Detailed Description 84
6.29.2 Constructor & Destructor Documentation 84
6.29.2.1 TimeSeries 84
6.29.2.2 ∼TimeSeries 84
6.29.3 Member Function Documentation 84
6.29.3.1 GetCurrentFieldValue 84
6.30 src::geometry::Triangulator Class Reference 85
6.30.1 Detailed Description 85
6.30.2 Member Typedef Documentation 85
6.30.2.1 Attributes 85
6.30.3 Member Enumeration Documentation 85
6.30.3.1 Attributes_t 86
6.30.4 Constructor & Destructor Documentation 86
6.30.4.1 Triangulator 86
6.30.4.2 ∼Triangulator 86
6.30.5 Member Function Documentation 86
6.30.5.1 ComputeBound 86
6.30.5.2 GetHull 86
6.30.5.3 GetNeighbours 86
6.30.5.4 GetNumFaces 86
6.30.5.5 GetNumNeighbours 86
6.30.5.6 GetNumTriangles 86
6.30.5.7 GetNumVoronoiVertices 86
6.30.5.8 GetTriangleIndices 87
6.30.5.9 GetVoronoiCellAreas 87

6.30.5.10 GetVoronoiSides . . . 87
6.30.5.11 GetVoronoiVertices . . . 87
6.30.6 Friends And Related Function Documentation . . . 87
6.30.6.1 operator<< . . . 87
6.31 src::model::Uplift Class Reference . . . 87
6.31.1 Detailed Description . . . 90
6.31.2 Constructor & Destructor Documentation . . . 90
6.31.2.1 Uplift . . . 90
6.31.2.2 ~Uplift . . . 90
6.31.3 Member Function Documentation . . . 90
6.31.3.1 Execute . . . 90
6.32 src::geometry::VSite Class Reference . . . 90
6.32.1 Detailed Description . . . 90
6.32.2 Friends And Related Function Documentation . . . 91
6.32.2.1 operator<< . . . 91
6.32.3 Member Data Documentation . . . 91
6.32.3.1 m_coord . . . 91

7 File Documentation **93**

7.1 src/geometry/Topology.hh File Reference . . . 93
7.2 src/geometry/Triangulator.hh File Reference . . . 95
7.3 src/math/Diffusion.hh File Reference . . . 96
7.4 src/mem/Allocator.hh File Reference . . . 97
7.5 src/mem/MemoryPool.hh File Reference . . . 98
7.6 src/mesh/KdItem.hh File Reference . . . 100
7.7 src/mesh/KdNode.hh File Reference . . . 101
7.8 src/mesh/KdTree.hh File Reference . . . 103
7.9 src/mesh/RegularMesh.hh File Reference . . . 104
7.10 src/mesh/SurfaceTopology.hh File Reference . . . 106
7.11 src/mesh/SurfaceTopologyOutput.hh File Reference . . . 107
7.12 src/model/FluvialErosion.hh File Reference . . . 108
7.13 src/model/FluvialErosionDeposition.hh File Reference . . . 109
7.14 src/model/HillSlope.hh File Reference . . . 110
7.15 src/model/Model.hh File Reference . . . 111

7.16 src/model/ModelBuilder.hh File Reference 112
7.17 src/model/Precipitation.hh File Reference 113
7.18 src/model/Process.hh File Reference 115
7.19 src/model/Uplift.hh File Reference 116
7.20 src/parser/Config.hh File Reference 116
7.21 src/parser/Log.hh File Reference 117
7.21.1 Define Documentation 119
7.21.1.1 LogDebug 119
7.21.1.2 LogError 119
7.21.1.3 LogInfo 119
7.22 src/util/Field.hh File Reference 120
7.23 src/util/ScalarField.hh File Reference 121
7.24 src/util/Timer.hh File Reference 123
7.25 src/util/TimeSeries.hh File Reference 125

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

src 9
src::geometry 9
src::math 10
src::mem 10
src::mesh 12
src::model 12
src::parser 12
src::util 13

Chapter 2

Class Index

2.1 Class Hierarchy

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

src::mem::Chunk . . . 15
src::parser::Config . . . 16
src::math::Diffusion::Coord_t . . . 17
src::math::Diffusion . . . 18
src::geometry::Edge . . . 20
src::util::Field . . . 29
    src::util::ScalarField< T > . . . 66
src::mesh::KdItem . . . 39
src::mesh::KdNode . . . 41
src::mesh::KdTree . . . 42
src::mem::MemoryPool . . . 43
src::model::Model . . . 48
src::model::ModelBuilder . . . 50
src::mem::PoolAllocator< T > . . . 51
src::mem::PoolAllocator< void > . . . 56
src::model::Process . . . 59
    src::model::FluvialErosion . . . 30
    src::model::FluvialErosionDeposition . . . 33
    src::model::HillSlope . . . 36
    src::model::Precipitation . . . 57
    src::model::Uplift . . . 87
src::geometry::QuadEdge . . . 61
src::mem::PoolAllocator< T >::rebind< U > . . . 64
src::mesh::RegularMesh . . . 65
src::geometry::Site . . . 69
src::geometry::SiteComparator . . . 72
src::mesh::SurfaceTopology . . . 73
src::mesh::SurfaceTopologyOutput . . . 82
src::util::Timer . . . 83

src::util::TimeSeries 84
src::geometry::Triangulator 85
src::geometry::VSite 90

Chapter 3

Class Index

3.1 Class List

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

src::mem::Chunk 15
src::parser::Config 16
src::math::Diffusion::Coord_t 17
src::math::Diffusion 18
src::geometry::Edge 20
src::util::Field 29
src::model::FluvialErosion 30
src::model::FluvialErosionDeposition 33
src::model::HillSlope 36
src::mesh::KdItem 39
src::mesh::KdNode 41
src::mesh::KdTree 42
src::mem::MemoryPool 43
src::model::Model 48
src::model::ModelBuilder 50
src::mem::PoolAllocator< T > 51
src::mem::PoolAllocator< void > 56
src::model::Precipitation 57
src::model::Process 59
src::geometry::QuadEdge 61
src::mem::PoolAllocator< T >::rebind< U > 64
src::mesh::RegularMesh 65
src::util::ScalarField< T > 66
src::geometry::Site 69
src::geometry::SiteComparator 72
src::mesh::SurfaceTopology 73
src::mesh::SurfaceTopologyOutput 82
src::util::Timer 83
src::util::TimeSeries 84

src::geometry::Triangulator 85
src::model::Uplift 87
src::geometry::VSite 90

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

src/geometry/Topology.hh . . . 93
src/geometry/Triangulator.hh . . . 95
src/math/Diffusion.hh . . . 96
src/mem/Allocator.hh . . . 97
src/mem/MemoryPool.hh . . . 98
src/mesh/KdItem.hh . . . 100
src/mesh/KdNode.hh . . . 101
src/mesh/KdTree.hh . . . 103
src/mesh/RegularMesh.hh . . . 104
src/mesh/SurfaceTopology.hh . . . 106
src/mesh/SurfaceTopologyOutput.hh . . . 107
src/model/FluvialErosion.hh . . . 108
src/model/FluvialErosionDeposition.hh . . . 109
src/model/HillSlope.hh . . . 110
src/model/Model.hh . . . 111
src/model/ModelBuilder.hh . . . 112
src/model/Precipitation.hh . . . 113
src/model/Process.hh . . . 115
src/model/Uplift.hh . . . 116
src/parser/Config.hh . . . 116
src/parser/Log.hh . . . 117
src/util/Field.hh . . . 120
src/util/ScalarField.hh . . . 121
src/util/Timer.hh . . . 123
src/util/TimeSeries.hh . . . 125

Chapter 5

Namespace Documentation

5.1 src Namespace Reference

Namespaces

- namespace geometry
- namespace math
- namespace mem
- namespace mesh
- namespace model
- namespace parser
- namespace util

5.2 src::geometry Namespace Reference

Classes

- class VSite
- class Site
- class SiteComparator
- class Edge
- class QuadEdge
- class Triangulator

Functions

- float FABS (float a)

5.2.1 Function Documentation

5.2.1.1 float src::geometry::FABS (float *a*) [inline]

Definition at line 57 of file Topology.hh.

```
{return ((a) >= 0.0 ? (a) : -(a));}
```

5.3 src::math Namespace Reference

Classes

- class Diffusion

5.4 src::mem Namespace Reference

Classes

- class PoolAllocator
- class PoolAllocator< void >
- struct Chunk
- class MemoryPool

Functions

- template<class T >
  bool operator== (const PoolAllocator< T > &, const PoolAllocator< T > &)
- template<class T >
  bool operator!= (const PoolAllocator< T > &, const PoolAllocator< T > &)
- void InitPool ()
- void FinalisePool ()

Variables

- pthread_mutex_t mutex
- static std::vector< MemoryPool * > pools
- const int CHUNK_ARRAY_DELTA = 10
- const int INVALID = -1
- const int SUCCESS = 1
- const int FAILURE = 0
- const int TWO_EXP16 = 65535

5.4.1 Function Documentation

5.4.1.1 void src::mem::FinalisePool ( )

5.4.1.2 void src::mem::InitPool ( )

5.4.1.3 template<class T > bool src::mem::operator!= (const PoolAllocator< T > & , const PoolAllocator< T > &) [inline]

Definition at line 167 of file Allocator.hh.

```
{
    return false;
}
```

5.4.1.4 template<class T > bool src::mem::operator== (const PoolAllocator< T > & , const PoolAllocator< T > &) [inline]

Definition at line 160 of file Allocator.hh.

```
{
    return true;
}
```

5.4.2 Variable Documentation

5.4.2.1 const int src::mem::CHUNK_ARRAY_DELTA = 10

Definition at line 49 of file MemoryPool.hh.

5.4.2.2 const int src::mem::FAILURE = 0

Definition at line 52 of file MemoryPool.hh.

5.4.2.3 const int src::mem::INVALID = -1

Definition at line 50 of file MemoryPool.hh.

5.4.2.4 pthread_mutex_t src::mem::mutex

5.4.2.5 std::vector<MemoryPool*> src::mem::pools [static]

Definition at line 49 of file Allocator.hh.

5.4.2.6 const int src::mem::SUCCESS = 1

Definition at line 51 of file MemoryPool.hh.

5.4.2.7 const int src::mem::TWO_EXP16 = 65535

Definition at line 53 of file MemoryPool.hh.

5.5 src::mesh Namespace Reference

Classes

- class KdItem
- class KdNode
- class KdTree
- class RegularMesh
- class SurfaceTopology
- class SurfaceTopologyOutput

5.6 src::model Namespace Reference

Classes

- class FluvialErosion
- class FluvialErosionDeposition
- class HillSlope
- class Model
- class ModelBuilder
- class Precipitation
- class Process
- class Uplift

5.7 src::parser Namespace Reference

Classes

- class Config

Enumerations

- enum LogLevel { LOG_QUIET, LOG_ERROR, LOG_INFO, LOG_DEBUG }

Functions

- void debugBreak ()
- vector< string > split (const string &s, char delim)

Variables

- LogLevel logLevel

5.7.1 Enumeration Type Documentation

5.7.1.1 enum src::parser::LogLevel

Enumerator:

LOG_QUIET

LOG_ERROR

LOG_INFO

LOG_DEBUG

Definition at line 48 of file Log.hh.

```
{ LOG_QUIET, LOG_ERROR, LOG_INFO, LOG_DEBUG };
```

5.7.2 Function Documentation

5.7.2.1 void src::parser::debugBreak ( )

5.7.2.2 vector<string> src::parser::split (const string & *s*, char *delim*)

5.7.3 Variable Documentation

5.7.3.1 LogLevel src::parser::logLevel

5.8 src::util Namespace Reference

Classes

- class Field
- class ScalarField
- class Timer
- class TimeSeries

Chapter 6

Class Documentation

6.1 src::mem::Chunk Struct Reference

`#include <MemoryPool.hh>`

Public Attributes

- char * memory
- int numFree
- int chunkId
- char ** freeList

6.1.1 Detailed Description

Definition at line 61 of file MemoryPool.hh.

6.1.2 Member Data Documentation

6.1.2.1 int src::mem::Chunk::chunkId

Definition at line 65 of file MemoryPool.hh.

6.1.2.2 char** src::mem::Chunk::freeList

Definition at line 66 of file MemoryPool.hh.

6.1.2.3 char* src::mem::Chunk::memory

Definition at line 63 of file MemoryPool.hh.

6.1.2.4 int src::mem::Chunk::numFree

Definition at line 64 of file MemoryPool.hh.

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

- src/mem/MemoryPool.hh

6.2 src::parser::Config Class Reference

`#include <Config.hh>`

Public Member Functions

- Config (string configFile)
- ~Config ()
- string PString (string name)
- bool PBool (string name)
- double PDouble (string name)
- int PInt (string name)
- map< string, string > & GetSymbols ()
- Config * Group (string name)
- map< string, Config * > & GetGroups ()
- string GetConfigName ()

6.2.1 Detailed Description

Definition at line 72 of file Config.hh.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 src::parser::Config::Config (string *configFile*)

6.2.2.2 src::parser::Config::~Config ( )

6.2.3 Member Function Documentation

6.2.3.1 string src::parser::Config::GetConfigName ( ) [inline]

Definition at line 116 of file Config.hh.

```
{return m_debugInfo;}
```

6.2.3.2 map<string, Config*>& src::parser::Config::GetGroups ( ) [inline]

Definition at line 111 of file Config.hh.

```
{
    return m_groups;
}
```

6.2.3.3 map<string, string>& src::parser::Config::GetSymbols ( ) [inline]

Definition at line 98 of file Config.hh.

```
{
    return m_symbols;
}
```

6.2.3.4 Config* src::parser::Config::Group (string *name*) [inline]

Definition at line 104 of file Config.hh.

```
{
    if(m_groups.find(name) == m_groups.end()) return NULL;
    return m_groups[name];
}
```

6.2.3.5 bool src::parser::Config::PBool (string *name*)

6.2.3.6 double src::parser::Config::PDouble (string *name*)

6.2.3.7 int src::parser::Config::PInt (string *name*)

6.2.3.8 string src::parser::Config::PString (string *name*)

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

- src/parser/Config.hh

6.3 src::math::Diffusion::Coord_t Struct Reference

`#include <Diffusion.hh>`

Public Member Functions

- Coord_t (float _x, float _y)

Public Attributes

- float x
- float y

6.3.1 Detailed Description

Definition at line 64 of file Diffusion.hh.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 src::math::Diffusion::Coord_t::Coord_t (float _x, float _y) [inline]

Definition at line 65 of file Diffusion.hh.

```
:x(_x),y(_y){} }Coord;
```

6.3.3 Member Data Documentation

6.3.3.1 float src::math::Diffusion::Coord_t::x

Definition at line 64 of file Diffusion.hh.

6.3.3.2 float src::math::Diffusion::Coord_t::y

Definition at line 64 of file Diffusion.hh.

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

- src/math/Diffusion.hh

6.4 src::math::Diffusion Class Reference

```
#include <Diffusion.hh>
```

Classes

- struct Coord_t

Public Types

- typedef struct src::math::Diffusion::Coord_t Coord
- typedef float(* ForcingFunc)(float, float, float)
- typedef float(* NeumannFunc)(float, float, float)

Public Member Functions

- Diffusion (SurfaceTopology *st, ForcingFunc f, NeumannFunc n, int nt, float dt, double tolerance, int maxIterations)
- ~Diffusion ()
- void SetIC (vector< float > *vals)
- void SetDirichlet (vector< float > *dirichlet)
- void SetCoefficient (vector< float > *coefficient)
- void GetSolution (vector< float > *result)
- void Step ()

Public Attributes

- MatrixXf m_solutions

6.4.1 Detailed Description

Definition at line 60 of file Diffusion.hh.

6.4.2 Member Typedef Documentation

6.4.2.1 typedef struct src::math::Diffusion::Coord_t src::math::Diffusion::Coord

6.4.2.2 typedef float(* src::math::Diffusion::ForcingFunc)(float, float, float)

Definition at line 67 of file Diffusion.hh.

6.4.2.3 typedef float(* src::math::Diffusion::NeumannFunc)(float, float, float)

Definition at line 68 of file Diffusion.hh.

6.4.3 Constructor & Destructor Documentation

6.4.3.1 src::math::Diffusion::Diffusion (SurfaceTopology * *st*, ForcingFunc *f*, NeumannFunc *n*, int *nt*, float *dt*, double *tolerance*, int *maxIterations*)

6.4.3.2 src::math::Diffusion::~Diffusion ( )

6.4.4 Member Function Documentation

6.4.4.1 void src::math::Diffusion::GetSolution (vector< float > * *result*)

6.4.4.2 void src::math::Diffusion::SetCoefficient (vector< float > * *coefficient*)

6.4.4.3 void src::math::Diffusion::SetDirichlet (vector< float > * *dirichlet*)

6.4.4.4 void src::math::Diffusion::SetIC (vector< float > * *vals*)

6.4.4.5 void src::math::Diffusion::Step ( )

6.4.5 Member Data Documentation

6.4.5.1 MatrixXf src::math::Diffusion::m_solutions

Definition at line 82 of file Diffusion.hh.

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

- src/math/Diffusion.hh

6.5 src::geometry::Edge Class Reference

`#include <Topology.hh>`

Public Member Functions

- Edge ()
- Edge * Rot ()
- Edge * Tor ()
- Edge * Sym ()
- Edge * Onext ()
- Edge * Oprev ()
- Edge * Dnext ()
- Edge * Dprev ()
- Edge * Lnext ()
- Edge * Lprev ()
- Edge * Rnext ()
- Edge * Rprev ()
- Site * Org ()
- Site * Dest ()
- VSite * VOrg ()
- VSite * VDest ()
- QuadEdge * Qedge ()
- void Onext_Set (Edge *e)
- void Org_Set (Site *s)
- void Dest_Set (Site *s)
- void VOrg_Set (VSite *vs)
- void VDest_Set (VSite *vs)

Friends

- class QuadEdge

6.5.1 Detailed Description

Definition at line 258 of file Topology.hh.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 src::geometry::Edge::Edge ( ) [inline]

Definition at line 267 of file Topology.hh.

```
{ m_data[0] = m_data[1] = 0; }
```

6.5.3 Member Function Documentation

6.5.3.1 Site* src::geometry::Edge::Dest ( ) [inline]

Definition at line 297 of file Topology.hh.

```
{ return Sym()->Org(); }
```

6.5.3.2 void src::geometry::Edge::Dest_Set (Site * *s*) [inline]

Definition at line 310 of file Topology.hh.

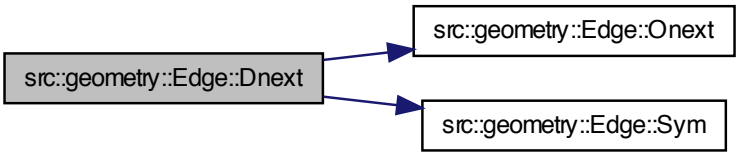
```
{ Sym()->Org_Set(s); }
```

6.5.3.3 Edge* src::geometry::Edge::Dnext ( ) [inline]

Definition at line 283 of file Topology.hh.

```
{ return Sym()->Onext()->Sym(); }
```

Here is the call graph for this function:

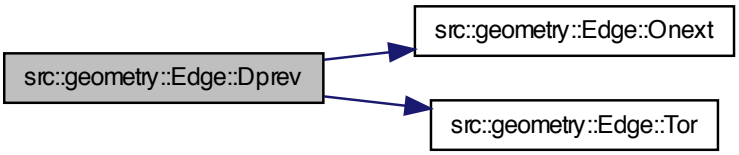


6.5.3.4 Edge* src::geometry::Edge::Dprev ( ) [inline]

Definition at line 285 of file Topology.hh.

```
{ return Tor()->Onext()->Tor(); }
```

Here is the call graph for this function:

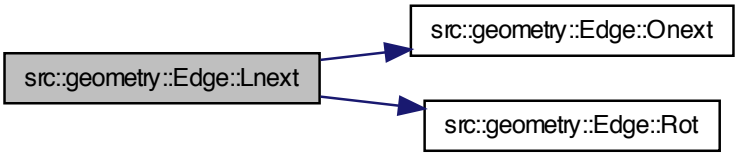


6.5.3.5 Edge* src::geometry::Edge::Lnext ( ) [inline]

Definition at line 287 of file Topology.hh.

```
{ return Tor()->Onext()->Rot(); }
```

Here is the call graph for this function:



6.5.3.6 Edge* src::geometry::Edge::Lprev ( ) [inline]

Definition at line 289 of file Topology.hh.

```
{ return Onext()->Sym(); }
```

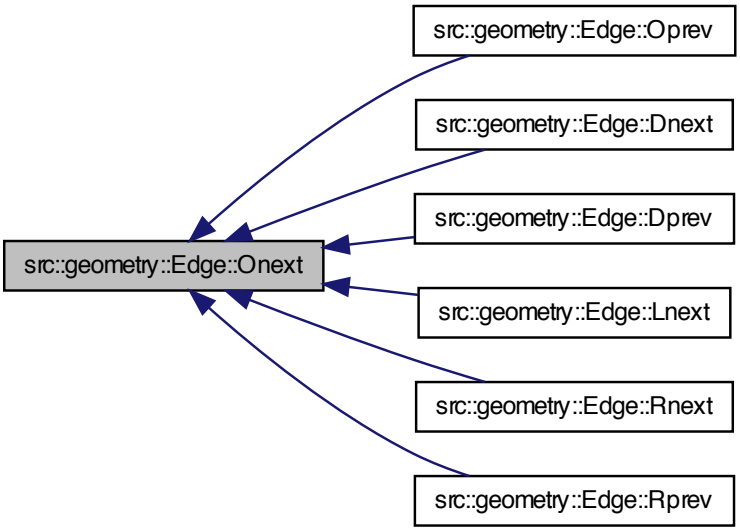
Here is the call graph for this function:

6.5.3.7 Edge* src::geometry::Edge::Onext ( ) [inline]

Definition at line 279 of file Topology.hh.

```
{ return m_next; }
```

Here is the caller graph for this function:



6.5.3.8 void src::geometry::Edge::Onext_Set (Edge * *e*) [inline]

Definition at line 308 of file Topology.hh.

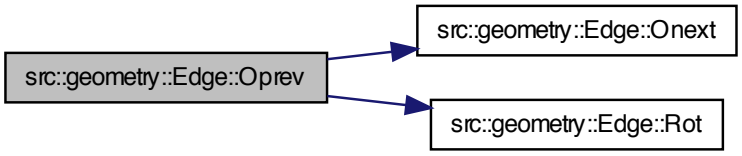
```
{ m_next = e; }
```

6.5.3.9 Edge* src::geometry::Edge::Oprev ( ) [inline]

Definition at line 281 of file Topology.hh.

```
{ return Rot()->Onext()->Rot(); }
```

Here is the call graph for this function:



6.5.3.10 Site* src::geometry::Edge::Org ( ) [inline]

Definition at line 295 of file Topology.hh.

```
{ return (Site*)m_data[0]; }
```

6.5.3.11 void src::geometry::Edge::Org_Set (Site * *s*) [inline]

Definition at line 309 of file Topology.hh.

```
{ m_data[0] = s; }
```

6.5.3.12 QuadEdge* src::geometry::Edge::Qedge ( ) [inline]

Definition at line 303 of file Topology.hh.

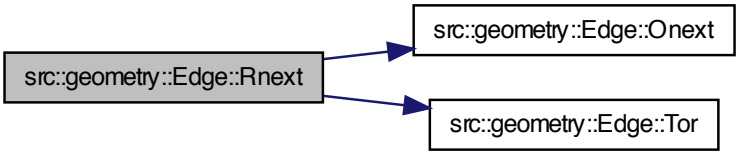
```
{ return (QuadEdge *)(this - m_num); }
```

6.5.3.13 Edge* src::geometry::Edge::Rnext ( ) [inline]

Definition at line 291 of file Topology.hh.

```
{ return Rot()->Onext()->Tor(); }
```

Here is the call graph for this function:

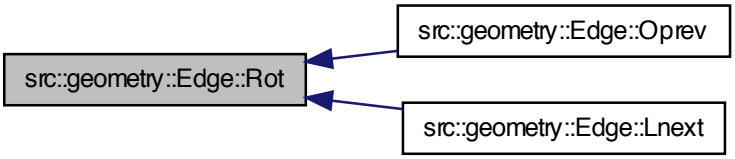


6.5.3.14 Edge* src::geometry::Edge::Rot ( ) [inline]

Definition at line 273 of file Topology.hh.

```
{ return (m_num < 3) ? this + 1 : this - 3; }
```

Here is the caller graph for this function:



6.5.3.15 Edge* src::geometry::Edge::Rprev ( ) [inline]

Definition at line 293 of file Topology.hh.

```
{ return Sym()->Onext(); }
```

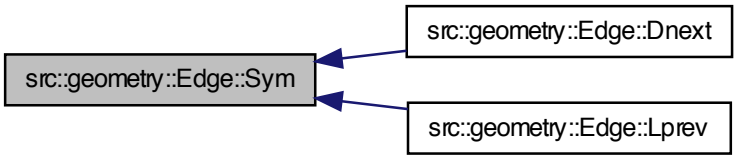
Here is the call graph for this function:

6.5.3.16 Edge* src::geometry::Edge::Sym ( ) [inline]

Definition at line 277 of file Topology.hh.

```
{ return (m_num < 2) ? this + 2 : this - 2; }
```

Here is the caller graph for this function:

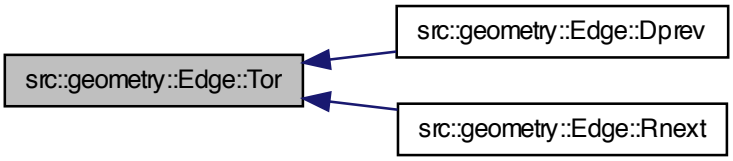


6.5.3.17 Edge* src::geometry::Edge::Tor ( ) [inline]

Definition at line 275 of file Topology.hh.

```
{ return (m_num > 0) ? this - 1 : this + 3; }
```

Here is the caller graph for this function:



6.5.3.18 VSite* src::geometry::Edge::VDest ( ) [inline]

Definition at line 301 of file Topology.hh.

```
{ return Sym()->VOrg(); }
```

6.5.3.19 void src::geometry::Edge::VDest_Set (VSite * *vs*) [inline]

Definition at line 312 of file Topology.hh.

```
{ Sym()->VOrg_Set(vs); }
```

6.5.3.20 VSite* src::geometry::Edge::VOrg ( ) [inline]

Definition at line 299 of file Topology.hh.

```
{ return (VSite*)m_data[1]; }
```

6.5.3.21 void src::geometry::Edge::VOrg_Set (VSite * *vs*) [inline]

Definition at line 311 of file Topology.hh.

```
{ m_data[1] = vs; }
```

6.5.4 Friends And Related Function Documentation

6.5.4.1 friend class QuadEdge [friend]

Definition at line 260 of file Topology.hh.

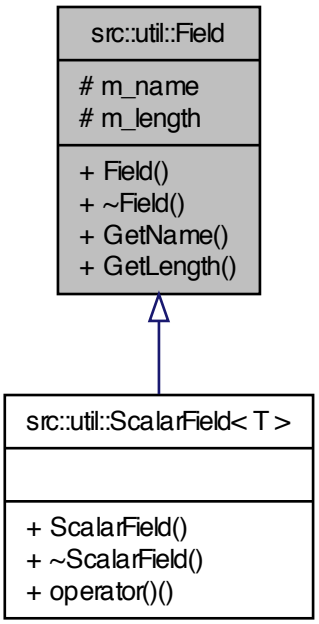
The documentation for this class was generated from the following file:

- src/geometry/Topology.hh

6.6 src::util::Field Class Reference

`#include <Field.hh>`

Inheritance diagram for src::util::Field:



Public Member Functions

- Field (string name, unsigned int length)
- virtual ~Field ()

- string GetName ()
- unsigned int GetLength ()

Protected Attributes

- string m_name
- unsigned int m_length

6.6.1 Detailed Description

Definition at line 53 of file Field.hh.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 src::util::Field::Field (string *name,* unsigned int *length*)

6.6.2.2 virtual src::util::Field::~Field ( ) [virtual]

6.6.3 Member Function Documentation

6.6.3.1 unsigned int src::util::Field::GetLength ( )

6.6.3.2 string src::util::Field::GetName ( )

6.6.4 Member Data Documentation

6.6.4.1 unsigned int src::util::Field::m_length [protected]

Definition at line 64 of file Field.hh.

6.6.4.2 string src::util::Field::m_name [protected]

Definition at line 63 of file Field.hh.

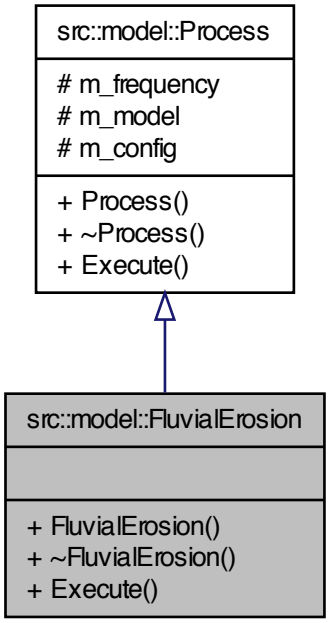
The documentation for this class was generated from the following file:

- src/util/Field.hh

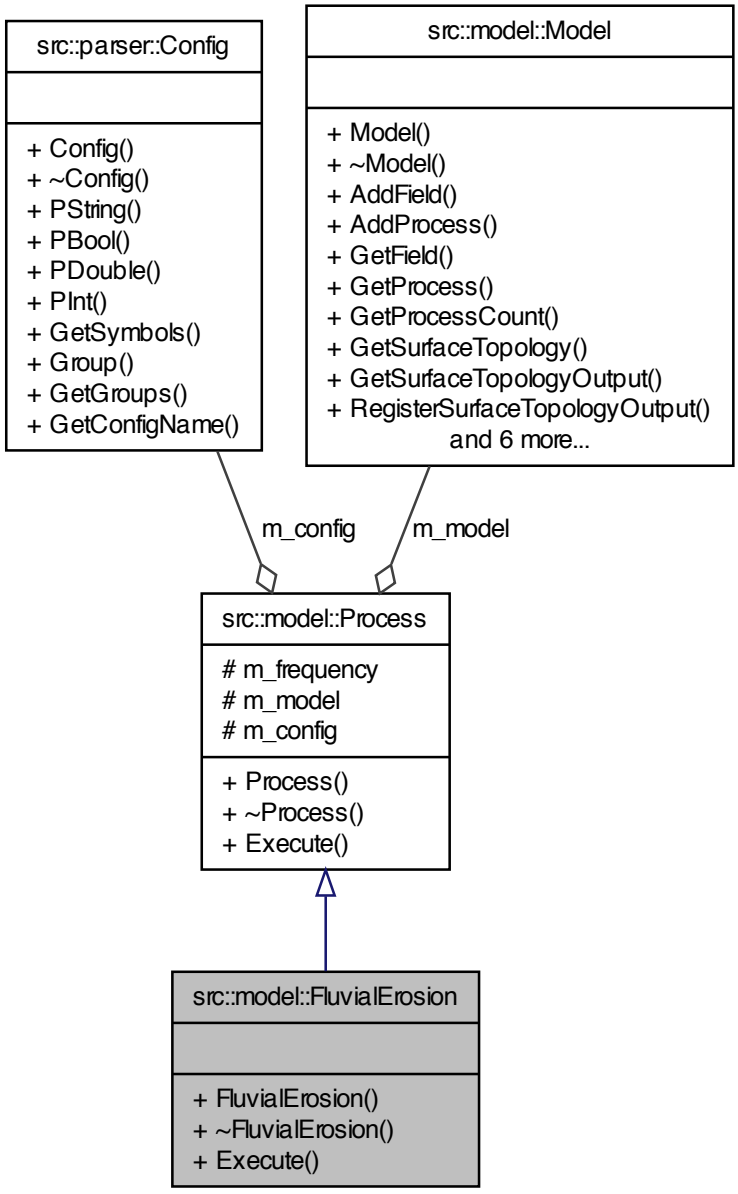
6.7 src::model::FluvialErosion Class Reference

```
#include <FluvialErosion.hh>
```

Inheritance diagram for src::model::FluvialErosion:



Collaboration diagram for src::model::FluvialErosion:



Public Member Functions

- FluvialErosion (const Model ∗m, Config ∗c)
- ∼FluvialErosion ()
- void Execute ()

6.7.1 Detailed Description

Definition at line 60 of file FluvialErosion.hh.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 src::model::FluvialErosion::FluvialErosion (const Model ∗ *m,* Config ∗ *c*)

6.7.2.2 src::model::FluvialErosion::∼FluvialErosion ( )

6.7.3 Member Function Documentation

6.7.3.1 void src::model::FluvialErosion::Execute ( ) [virtual]

Implements src::model::Process.

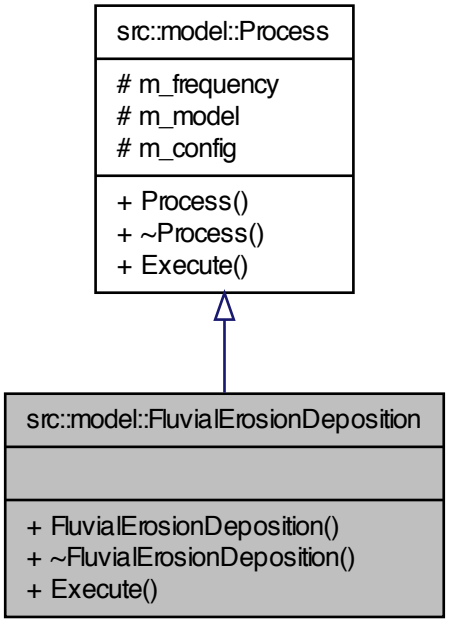
The documentation for this class was generated from the following file:

- src/model/FluvialErosion.hh

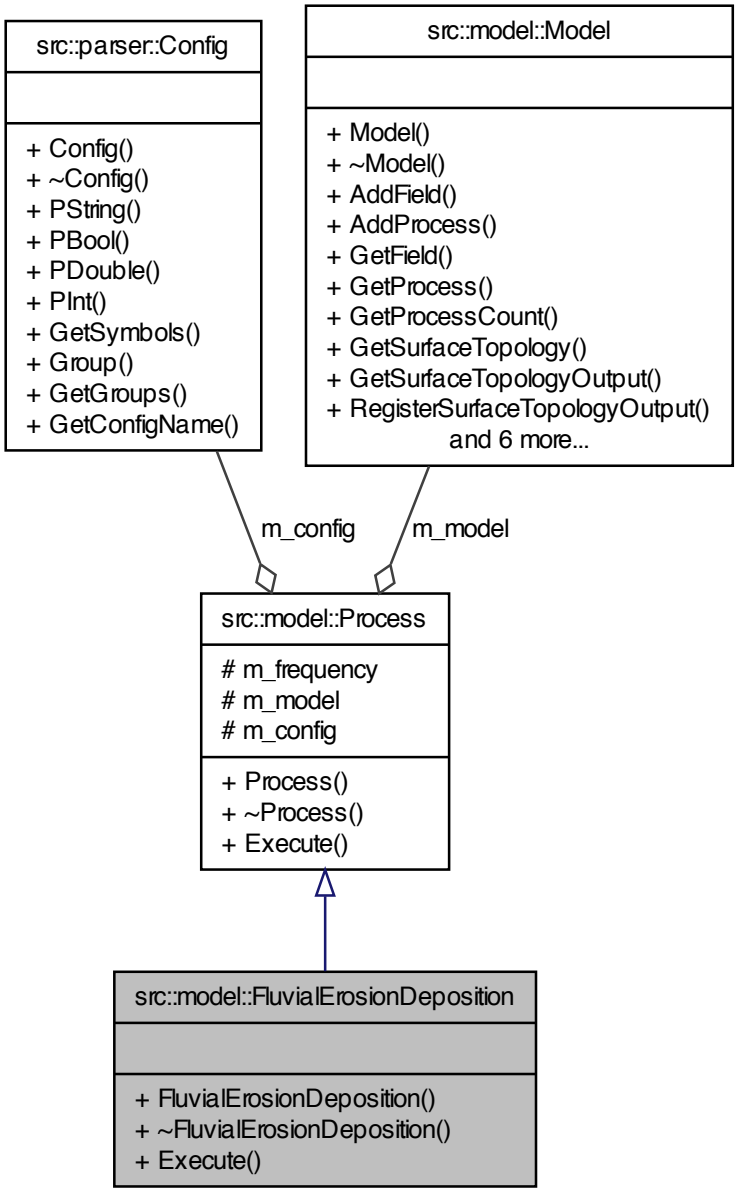
6.8 src::model::FluvialErosionDeposition Class Reference

```
#include <FluvialErosionDeposition.hh>
```

Inheritance diagram for src::model::FluvialErosionDeposition:



Collaboration diagram for src::model::FluvialErosionDeposition:



Public Member Functions

- FluvialErosionDeposition (const Model ∗m, Config ∗c)
- ∼FluvialErosionDeposition ()
- void Execute ()

6.8.1 Detailed Description

Definition at line 60 of file FluvialErosionDeposition.hh.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 src::model::FluvialErosionDeposition::FluvialErosionDeposition (const Model ∗ *m,* Config ∗ *c*)

6.8.2.2 src::model::FluvialErosionDeposition::∼FluvialErosionDeposition ( )

6.8.3 Member Function Documentation

6.8.3.1 void src::model::FluvialErosionDeposition::Execute ( ) [virtual]

Implements src::model::Process.

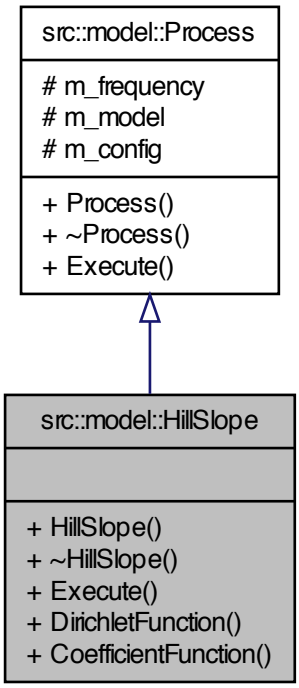
The documentation for this class was generated from the following file:

- src/model/FluvialErosionDeposition.hh

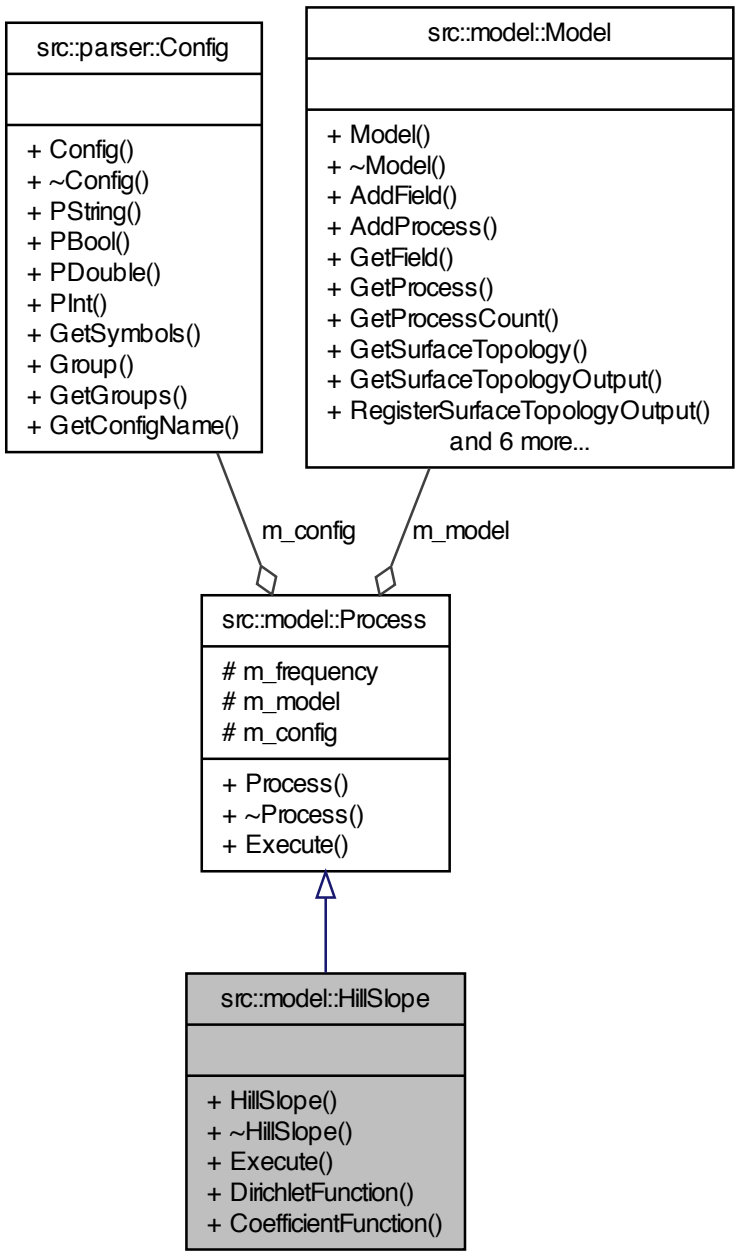
6.9 src::model::HillSlope Class Reference

```
#include <HillSlope.hh>
```

Inheritance diagram for src::model::HillSlope:



Collaboration diagram for src::model::HillSlope:



Public Member Functions

- HillSlope (const Model ∗m, Config ∗c)
- ∼HillSlope ()
- void Execute ()

Static Public Member Functions

- static float DirichletFunction (int idx)
- static float CoefficientFunction (int idx)

6.9.1 Detailed Description

Definition at line 62 of file HillSlope.hh.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 src::model::HillSlope::HillSlope (const Model ∗ *m*, Config ∗ *c*)

6.9.2.2 src::model::HillSlope::∼HillSlope ( )

6.9.3 Member Function Documentation

6.9.3.1 static float src::model::HillSlope::CoefficientFunction (int *idx*) [static]

6.9.3.2 static float src::model::HillSlope::DirichletFunction (int *idx*) [static]

6.9.3.3 void src::model::HillSlope::Execute ( ) [virtual]

Implements src::model::Process.

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

- src/model/HillSlope.hh

6.10 src::mesh::KdItem Class Reference

`#include <KdItem.hh>`

Public Member Functions

- KdItem ()
- void Print ()
- ∼KdItem ()

Public Attributes

- double m_coord [2]
- int m_id

Friends

- class KdTree

6.10.1 Detailed Description

Definition at line 52 of file KdItem.hh.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 src::mesh::KdItem::KdItem ( )

6.10.2.2 src::mesh::KdItem::~KdItem ( )

6.10.3 Member Function Documentation

6.10.3.1 void src::mesh::KdItem::Print ( )

6.10.4 Friends And Related Function Documentation

6.10.4.1 friend class KdTree [friend]

Definition at line 55 of file KdItem.hh.

6.10.5 Member Data Documentation

6.10.5.1 double src::mesh::KdItem::m_coord[2]

Definition at line 61 of file KdItem.hh.

6.10.5.2 int src::mesh::KdItem::m_id

Definition at line 62 of file KdItem.hh.

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

- src/mesh/KdItem.hh

6.11 src::mesh::KdNode Class Reference

`#include <KdNode.hh>`

Classes

- struct **KdItemSort**

Public Member Functions

- KdNode (KdTree ∗tree)
- KdNode (KdNode ∗node)
- void Add (KdItem ∗m)
- void Split (KdItem ∗m)
- void Expand (double ∗newCoord)
- void Print ()
- void DeleteHelper ()
- bool Intersects (double ∗up0, double ∗low0, double ∗up1, double ∗low1)
- void Range (double ∗upper, double ∗lower, vector< KdItem ∗ > ∗result)
- bool Contains (double ∗upper, double ∗lower, double ∗pnt)
- ∼KdNode ()

6.11.1 Detailed Description

Definition at line 56 of file KdNode.hh.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 src::mesh::KdNode::KdNode (KdTree ∗ *tree*)

6.11.2.2 src::mesh::KdNode::KdNode (KdNode ∗ *node*)

6.11.2.3 src::mesh::KdNode::∼KdNode ( )

6.11.3 Member Function Documentation

6.11.3.1 void src::mesh::KdNode::Add (KdItem ∗ *m*)

6.11.3.2 bool src::mesh::KdNode::Contains (double ∗ *upper,* double ∗ *lower,* double ∗ *pnt*)

6.11.3.3 void src::mesh::KdNode::DeleteHelper ( )

6.11.3.4 void src::mesh::KdNode::Expand (double ∗ *newCoord*)

6.11.3.5 bool src::mesh::KdNode::Intersects (double * *up0*, double * *low0*, double * *up1*, double * *low1*)

6.11.3.6 void src::mesh::KdNode::Print ( )

6.11.3.7 void src::mesh::KdNode::Range (double * *upper*, double * *lower*, vector< KdItem * > * *result*)

6.11.3.8 void src::mesh::KdNode::Split (KdItem * *m*)

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

- src/mesh/KdNode.hh

6.12 src::mesh::KdTree Class Reference

`#include <KdTree.hh>`

Public Member Functions

- KdTree (int maxElems)
- void Add (float *c, int id)
- void Print ()
- void DeleteAll ()
- int Size ()
- void QueryBallPoint (float *pos, float r, vector< float > *distance, vector< int > *id)
- ~KdTree ()

Friends

- class src::mesh::KdNode

6.12.1 Detailed Description

Definition at line 56 of file KdTree.hh.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 src::mesh::KdTree::KdTree (int *maxElems*)

6.12.2.2 src::mesh::KdTree::~KdTree ( )

6.12.3 Member Function Documentation

6.12.3.1 void src::mesh::KdTree::Add (float * c, int id)

6.12.3.2 void src::mesh::KdTree::DeleteAll ( )

6.12.3.3 void src::mesh::KdTree::Print ( )

6.12.3.4 void src::mesh::KdTree::QueryBallPoint (float * pos, float r, vector< float > * distance, vector< int > * id)

6.12.3.5 int src::mesh::KdTree::Size ( )

6.12.4 Friends And Related Function Documentation

6.12.4.1 friend class src::mesh::KdNode [friend]

Definition at line 60 of file KdTree.hh.

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

- src/mesh/KdTree.hh

6.13 src::mem::MemoryPool Class Reference

`#include <MemoryPool.hh>`

Public Types

- enum MemoryPoolMode_t { Fixed, Dynamic }
- typedef enum src::mem::MemoryPool::MemoryPoolMode_t MemoryPoolMode

Public Member Functions

- MemoryPool (int elemSize, int numElemsPerChunk, MemoryPoolMode m=-Dynamic)
- unsigned long GetNumFree ()
- void * GetDataPointer ()
- ~MemoryPool ()
- void * NewObject ()
- int DeleteObject (void *object)

6.13.1 Detailed Description

Definition at line 75 of file MemoryPool.hh.

6.13.2 Member Typedef Documentation

6.13.2.1 typedef enum src::mem::MemoryPool::MemoryPoolMode_t src::mem::MemoryPool::MemoryPoolMode

6.13.3 Member Enumeration Documentation

6.13.3.1 enum src::mem::MemoryPool::MemoryPoolMode_t

Enumerator:

Fixed

Dynamic

Definition at line 78 of file MemoryPool.hh.

```
{
    Fixed,
    Dynamic
}MemoryPoolMode;
```

6.13.4 Constructor & Destructor Documentation

6.13.4.1 src::mem::MemoryPool::MemoryPool (int *elemSize,* int *numElemsPerChunk,* MemoryPoolMode *m* = Dynamic) [inline]

Definition at line 91 of file MemoryPool.hh.

```
{
    int i = 0;

    assert( numElemsPerChunk > 0 );
    assert( elemSize > 0 );

    numElementsPerChunk = numElemsPerChunk;

    numChunks = 0;
    elementSize = elemSize;
    mode = m;

    if(mode == Dynamic)
    {
        if( numElementsPerChunk > TWO_EXP16 )
        {
            numElementsPerChunk = TWO_EXP16;
        }
    }

    maxChunkEntries = CHUNK_ARRAY_DELTA;
    chunkToUse = INVALID;

    chunks = ( Chunk* )NULL;
    chunks = ( Chunk* )malloc( sizeof( Chunk ) * maxChunkEntries );
    memset( chunks, 0, sizeof( Chunk )*maxChunkEntries );
```

```
        for( i=0; i<maxChunkEntries; i++ )
        {
            chunks[i].chunkId = INVALID;
            chunks[i].numFree = INVALID;
        }

        assert( chunks != NULL );
    }
```

6.13.4.2 src::mem::MemoryPool::~MemoryPool ( ) [inline]

Definition at line 179 of file MemoryPool.hh.

```
    {
        int i = 0;

        for( i=0; i<maxChunkEntries; i++ )
        {
            if( chunks[i].numFree != INVALID )
            {
                free( chunks[i].memory );
                free( chunks[i].freeList );
            }
        }

        free( chunks );
    }
```

6.13.5 Member Function Documentation

6.13.5.1 int src::mem::MemoryPool::DeleteObject (void * *object*) [inline]

Definition at line 260 of file MemoryPool.hh.

```
    {
        if( object != NULL )
        {
            int                 i           = 0;
            int                 valid       = 0;
            int                 chunkIdx    = 0;

            for( i=0; i<maxChunkEntries; i++ )
            {

                if( chunks[i].memory != NULL &&
                    (( char* )object >= chunks[i].memory ) &&
                    (( char* )object < ( chunks[i].memory+( numElementsPerChunk
      *elementSize ) ) ) )
                {
                    valid = 1;
                    chunkIdx = i;
                    break;
                }
            }
```

```
            if( valid )
            {
                memset( object, 0, elementSize );
                chunks[chunkIdx].freeList[chunks[chunkIdx].numFree++] = ( char*
    )object;

                Shrink();

                return 1;
            }
            else
            {
                return 0;
            }
        }
        else
        {
            return 0;
        }
    }
```

6.13.5.2 void* src::mem::MemoryPool::GetDataPointer ( ) [inline]

Definition at line 161 of file MemoryPool.hh.

```
    {
        if(mode==Fixed)
        {
            if(chunks[0].numFree != INVALID)
                return chunks[0].memory;
        }

        return NULL;
    }
```

6.13.5.3 unsigned long src::mem::MemoryPool::GetNumFree ( ) [inline]

Definition at line 135 of file MemoryPool.hh.

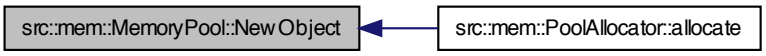
```
    {
        unsigned long numFree = 0;

        if(mode == Fixed)
        {
            int i = 0;
            for( i=0; i<maxChunkEntries; i++ )
            {
                if( chunks[i].numFree != INVALID )
                {
                    numFree += chunks[i].numFree;
                }
            }
        }

        return numFree;
    }
```

6.13.5.4 void* src::mem::MemoryPool::NewObject ( ) [inline]

Definition at line 202 of file MemoryPool.hh.

```
    {
        int             index       = 0;
        Chunk           *chunk      = NULL;

        if( chunkToUse == INVALID )
        {
            chunkToUse = CreateChunk( 0 );
        }

        chunk = &( chunks[chunkToUse] );

        assert( chunk != NULL );

        /*
       ------------------------------------------------------------------------
         * Abort if this is a fixed-pool and all slots have been allotted
         *
       ------------------------------------------------------------------------*/
        if(mode==Fixed)
        {
            if(chunk->numFree <= 0) return NULL;
        }
label:
        index = chunk->numFree - 1;
        if( index < 0 )
        {

            chunkToUse = GetChunkWithFreeSlots();

            if( chunkToUse == INVALID )
            {
                int chunkSlot = GetFreeChunkSlot();

                if( chunkSlot==INVALID )
                {
                    chunkToUse = CreateChunk( maxChunkEntries );
                    assert( chunkToUse != INVALID );
                }
                else
                {
                    chunkToUse = CreateChunk( chunkSlot );
                    assert( chunkToUse != INVALID );
                }
            }

            chunk = &( chunks[chunkToUse] );
            goto label;
        }

        return ( void* )( chunk->freeList[--chunk->numFree] );
    }
```

Here is the caller graph for this function:



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

- src/mem/MemoryPool.hh

6.14 src::model::Model Class Reference

`#include <Model.hh>`

Public Member Functions

- Model (const SurfaceTopology *st, Config *c)
- ~Model ()
- void AddField (Field *f) const
- void AddProcess (Process *p) const
- Field * GetField (string name) const
- Process * GetProcess (int index) const
- int GetProcessCount () const
- const SurfaceTopology * GetSurfaceTopology () const
- SurfaceTopologyOutput * GetSurfaceTopologyOutput () const
- void RegisterSurfaceTopologyOutput (SurfaceTopologyOutput *sfo)
- float GetDt () const
- float GetTime () const
- int GetTimeStep () const
- int GetNumTimeSteps (float t) const
- int GetNumTimeSteps () const
- bool NextTimeStep ()
- int GetNParallelCores () const

6.14.1 Detailed Description

Definition at line 77 of file Model.hh.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 src::model::Model::Model (const SurfaceTopology * *st*, Config * *c*)

6.14.2.2 src::model::Model::~Model ( )

6.14.3 Member Function Documentation

6.14.3.1 void src::model::Model::AddField (Field * *f*) const

6.14.3.2 void src::model::Model::AddProcess (Process * *p*) const

6.14.3.3 float src::model::Model::GetDt ( ) const

6.14.3.4 Field* src::model::Model::GetField (string *name*) const

6.14.3.5 int src::model::Model::GetNParallelCores ( ) const

6.14.3.6 int src::model::Model::GetNumTimeSteps (float *t*) const

6.14.3.7 int src::model::Model::GetNumTimeSteps ( ) const

6.14.3.8 Process* src::model::Model::GetProcess (int *index*) const

6.14.3.9 int src::model::Model::GetProcessCount ( ) const

6.14.3.10 const SurfaceTopology* src::model::Model::GetSurfaceTopology ( ) const

6.14.3.11 SurfaceTopologyOutput* src::model::Model::GetSurfaceTopology-Output ( ) const

6.14.3.12 float src::model::Model::GetTime ( ) const

6.14.3.13 int src::model::Model::GetTimeStep ( ) const

6.14.3.14 bool src::model::Model::NextTimeStep ( )

6.14.3.15 void src::model::Model::RegisterSurfaceTopologyOutput (SurfaceTopologyOutput * *sfo*)

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

- src/model/Model.hh

6.15 src::model::ModelBuilder Class Reference

`#include <ModelBuilder.hh>`

Public Member Functions

- ModelBuilder (Config ∗c)
- ∼ModelBuilder ()
- Model ∗ GetModel ()
- SurfaceTopology ∗ GetSurfaceTopology ()
- SurfaceTopologyOutput ∗ GetSurfaceTopologyOutput ()

6.15.1 Detailed Description

Definition at line 65 of file ModelBuilder.hh.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 src::model::ModelBuilder::ModelBuilder (Config ∗ *c*)

6.15.2.2 src::model::ModelBuilder::∼ModelBuilder ( )

6.15.3 Member Function Documentation

6.15.3.1 Model∗ src::model::ModelBuilder::GetModel ( ) [inline]

Definition at line 71 of file ModelBuilder.hh.

```
{ return m_model; }
```

6.15.3.2 SurfaceTopology∗ src::model::ModelBuilder::GetSurfaceTopology ( ) [inline]

Definition at line 72 of file ModelBuilder.hh.

```
{ return m_surfaceTopology; }
```

6.15.3.3 SurfaceTopologyOutput∗ src::model::ModelBuilder::GetSurfaceTopologyOutput ( ) [inline]

Definition at line 73 of file ModelBuilder.hh.

```
{ return m_surfaceTopologyOutput; }
```

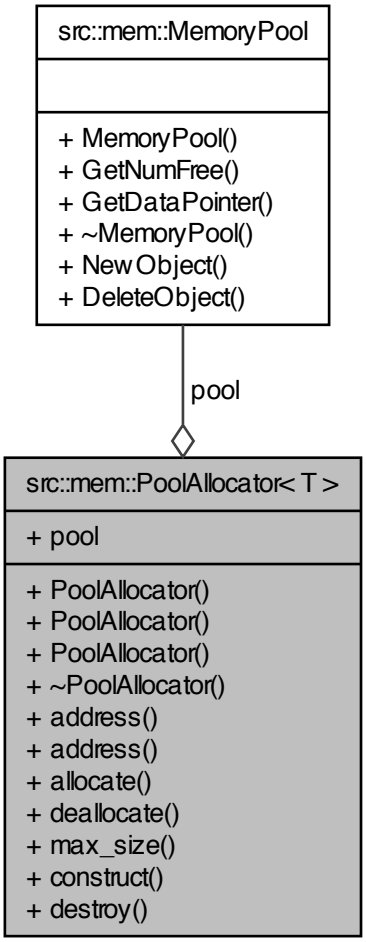
The documentation for this class was generated from the following file:

- src/model/ModelBuilder.hh

6.16 src::mem::PoolAllocator< T > Class Template Reference

`#include <Allocator.hh>`

Collaboration diagram for src::mem::PoolAllocator< T >:



Classes

- struct rebind

Public Types

- typedef T value_type
- typedef value_type * pointer
- typedef const value_type * const_pointer
- typedef value_type & reference
- typedef const value_type & const_reference
- typedef std::size_t size_type
- typedef std::ptrdiff_t difference_type

Public Member Functions

- PoolAllocator ()
- PoolAllocator (const PoolAllocator &)
- template<class U >
  PoolAllocator (const PoolAllocator< U > &)
- ~PoolAllocator ()
- pointer address (reference x) const
- const_pointer address (const_reference x) const
- pointer allocate (size_type n, const_pointer=0)
- void deallocate (pointer p, size_type)
- size_type max_size () const
- void construct (pointer p, const value_type &x)
- void destroy (pointer p)

Static Public Attributes

- static MemoryPool * pool = NULL

6.16.1 Detailed Description

template<class T>class src::mem::PoolAllocator< T >

Definition at line 57 of file Allocator.hh.

6.16.2 Member Typedef Documentation

6.16.2.1 template<class T > typedef const value_type* src::mem::PoolAllocator< T >::const_pointer

Definition at line 62 of file Allocator.hh.

6.16.2.2 template<class T > typedef const value_type& src::mem::PoolAllocator< T >::const_reference

Definition at line 64 of file Allocator.hh.

6.16.2.3 template<class T > typedef std::ptrdiff_t src::mem::PoolAllocator< T >::difference_type

Definition at line 66 of file Allocator.hh.

6.16.2.4 template<class T > typedef value_type* src::mem::PoolAllocator< T >::pointer

Definition at line 61 of file Allocator.hh.

6.16.2.5 template<class T > typedef value_type& src::mem::PoolAllocator< T >::reference

Definition at line 63 of file Allocator.hh.

6.16.2.6 template<class T > typedef std::size_t src::mem::PoolAllocator< T >::size_type

Definition at line 65 of file Allocator.hh.

6.16.2.7 template<class T > typedef T src::mem::PoolAllocator< T >::value_type

Definition at line 60 of file Allocator.hh.

6.16.3 Constructor & Destructor Documentation

6.16.3.1 template<class T > src::mem::PoolAllocator< T >::PoolAllocator ( ) [inline]

Definition at line 76 of file Allocator.hh.

```
{
}
```

6.16.3.2 template<class T > src::mem::PoolAllocator< T >::PoolAllocator (const PoolAllocator< T > &) [inline]

Definition at line 80 of file Allocator.hh.

```
{
}
```

6.16.3.3 template<class T > template<class U > src::mem::PoolAllocator< T >::PoolAllocator (const PoolAllocator< U > &) [inline]

Definition at line 85 of file Allocator.hh.

```
{
}
```

6.16.3.4 template<class T > src::mem::PoolAllocator< T >::~PoolAllocator ( ) [inline]

Definition at line 89 of file Allocator.hh.

```
{
}
```

6.16.4 Member Function Documentation

6.16.4.1 template<class T > pointer src::mem::PoolAllocator< T >::address (reference *x*) const [inline]

Definition at line 93 of file Allocator.hh.

```
{
    return &x;
}
```

6.16.4.2 template<class T > const_pointer src::mem::PoolAllocator< T >::address (const_reference *x*) const [inline]

Definition at line 97 of file Allocator.hh.

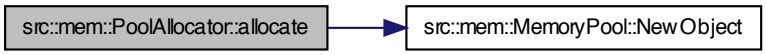
```
{
    return x;
}
```

6.16.4.3 template<class T > pointer src::mem::PoolAllocator< T >::allocate (size_type *n*, const_pointer = 0) [inline]

Definition at line 102 of file Allocator.hh.

```
{
    pthread_mutex_lock( &mutex );

    if( !PoolAllocator::pool )
    {
        PoolAllocator::pool = new MemoryPool( sizeof( T ), TWO_EXP16 );

        pools.push_back( PoolAllocator::pool );
    }

    void* p = pool->NewObject();
    if( !p )
        throw std::bad_alloc();

    pthread_mutex_unlock( &mutex );

    return static_cast<pointer>( p );
}
```

Here is the call graph for this function:



6.16.4.4 template<class T > void src::mem::PoolAllocator< T >::construct (pointer *p*, const value_type & *x*) [inline]

Definition at line 131 of file Allocator.hh.

```
{
    new( p ) value_type( x );
}
```

6.16.4.5 template<class T > void src::mem::PoolAllocator< T >::deallocate (pointer *p*, size_type) [inline]

Definition at line 122 of file Allocator.hh.

```
{
}
```

6.16.4.6 template<class T > void src::mem::PoolAllocator< T >::destroy (pointer *p*) [inline]

Definition at line 136 of file Allocator.hh.

```
{
    //p->~value_type();
}
```

6.16.4.7 template<class T > size_type src::mem::PoolAllocator< T >::max_size ( ) const [inline]

Definition at line 126 of file Allocator.hh.

```
{
    return static_cast<size_type>( -1 ) / sizeof( T );
}
```

6.16.5 Member Data Documentation

6.16.5.1 template<class T > MemoryPool * src::mem::PoolAllocator< T >::pool = NULL [static]

Definition at line 74 of file Allocator.hh.

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

- src/mem/Allocator.hh

6.17 src::mem::PoolAllocator< void > Class Template Reference

`#include <Allocator.hh>`

Classes

- struct **rebind**

6.17.1 Detailed Description

template<>class src::mem::PoolAllocator< void >

Definition at line 145 of file Allocator.hh.

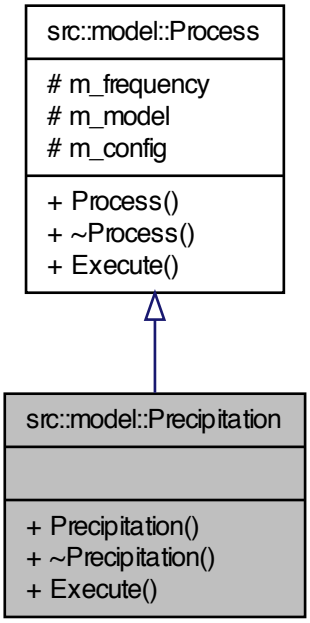
The documentation for this class was generated from the following file:

- src/mem/Allocator.hh

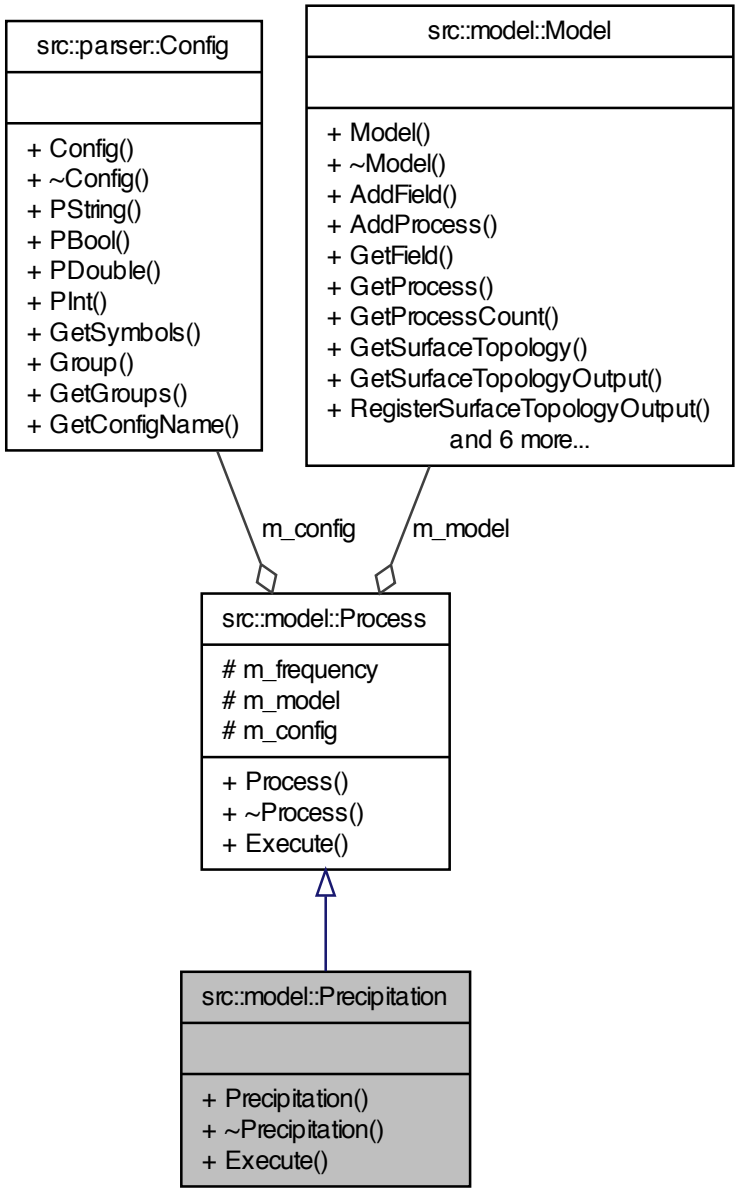
6.18 src::model::Precipitation Class Reference

`#include <Precipitation.hh>`

Inheritance diagram for src::model::Precipitation:



Collaboration diagram for src::model::Precipitation:



Public Member Functions

- Precipitation (const Model ∗m, Config ∗c)
- ∼Precipitation ()
- void Execute ()

6.18.1 Detailed Description

Definition at line 57 of file Precipitation.hh.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 src::model::Precipitation::Precipitation (const Model ∗ *m,* Config ∗ *c*)

6.18.2.2 src::model::Precipitation::∼Precipitation ( )

6.18.3 Member Function Documentation

6.18.3.1 void src::model::Precipitation::Execute ( ) [virtual]

Implements src::model::Process.

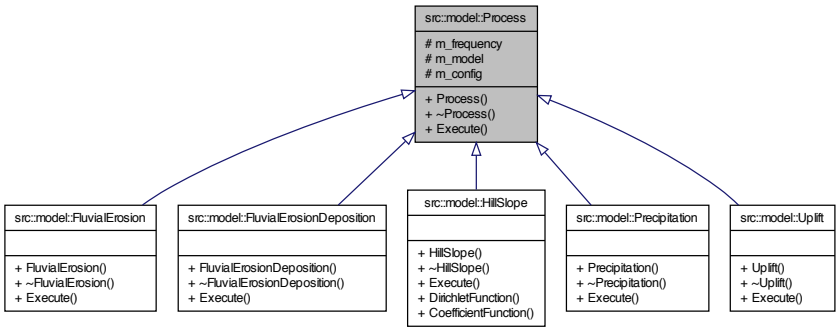
The documentation for this class was generated from the following file:

- src/model/Precipitation.hh

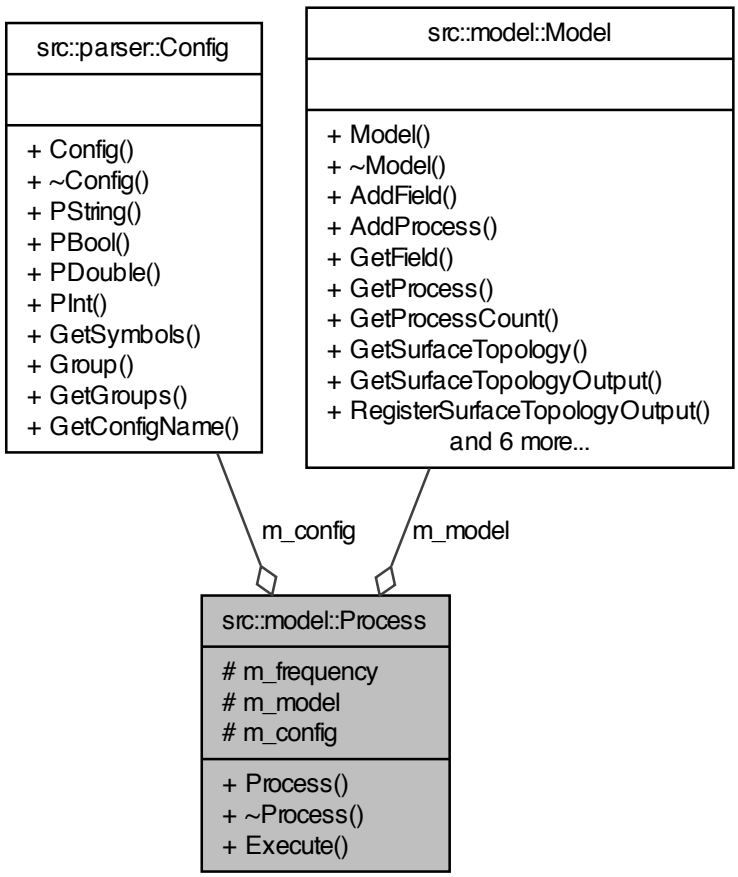
6.19 src::model::Process Class Reference

`#include <Process.hh>`

Inheritance diagram for src::model::Process:



Collaboration diagram for src::model::Process:



Public Member Functions

- Process (const Model ∗m, Config ∗c)
- virtual ∼Process ()
- virtual void Execute ()=0

Protected Attributes

- int m_frequency
- const Model ∗ m_model

- Config * m_config

6.19.1 Detailed Description

Definition at line 55 of file Process.hh.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 src::model::Process::Process (const Model * *m*, Config * *c*)

6.19.2.2 virtual src::model::Process::~Process ( ) [virtual]

6.19.3 Member Function Documentation

6.19.3.1 virtual void src::model::Process::Execute ( ) [pure virtual]

Implemented in src::model::HillSlope, src::model::FluvialErosion, src::model::FluvialErosionDeposition, src::model::Precipitation, and src::model::Uplift.

6.19.4 Member Data Documentation

6.19.4.1 Config* src::model::Process::m_config [protected]

Definition at line 65 of file Process.hh.

6.19.4.2 int src::model::Process::m_frequency [protected]

Definition at line 63 of file Process.hh.

6.19.4.3 const Model* src::model::Process::m_model [protected]

Definition at line 64 of file Process.hh.

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

- src/model/Process.hh

6.20 src::geometry::QuadEdge Class Reference

```
#include <Topology.hh>
```

Public Member Functions

- QuadEdge ()
- int Visited ()
- void ResetVisited ()
- void IncrementVisited ()
- void DecrementVisited ()
- bool IsFree ()
- void SetFree ()
- void SetInUse ()

Friends

- class Triangulator

6.20.1 Detailed Description

Definition at line 325 of file Topology.hh.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 src::geometry::QuadEdge::QuadEdge ( ) [inline]

Definition at line 340 of file Topology.hh.

```
{
    m_e[0].m_num = 0, m_e[1].m_num = 1, m_e[2].m_num = 2, m_e[3].m_num = 3;
    m_e[0].m_next = &(m_e[0]); m_e[1].m_next = &(m_e[3]);
    m_e[2].m_next = &(m_e[2]); m_e[3].m_next = &(m_e[1]);

    m_attributes = 0;
}
```

6.20.3 Member Function Documentation

6.20.3.1 void src::geometry::QuadEdge::DecrementVisited ( ) [inline]

Definition at line 404 of file Topology.hh.

```
{
    int val = (m_attributes & 0xffff0000) >> 16;
    m_attributes &= ~0xffff0000;
    val--;
    m_attributes |= (val<<16);
}
```

6.20.3.2 void src::geometry::QuadEdge::IncrementVisited ( ) [inline]

Definition at line 390 of file Topology.hh.

```
{
    int val = (m_attributes & 0xffff0000) >> 16;
    m_attributes &= ~0xffff0000;
    val++;
    m_attributes |= (val<<16);
}
```

6.20.3.3 bool src::geometry::QuadEdge::IsFree ( ) [inline]

Definition at line 418 of file Topology.hh.

```
{
    return (m_attributes & 0x0000ffff)==0;
}
```

6.20.3.4 void src::geometry::QuadEdge::ResetVisited ( ) [inline]

Definition at line 378 of file Topology.hh.

```
{
    m_attributes &= ~0xffff0000;
}
```

6.20.3.5 void src::geometry::QuadEdge::SetFree ( ) [inline]

Definition at line 429 of file Topology.hh.

```
{
    m_attributes &= ~0x0000ffff;
}
```

6.20.3.6 void src::geometry::QuadEdge::SetInUse ( ) [inline]

Definition at line 440 of file Topology.hh.

```
{
    m_attributes |= 1;
}
```

6.20.3.7 int src::geometry::QuadEdge::Visited ( ) [inline]

Definition at line 367 of file Topology.hh.

```
{
    return (m_attributes & 0xffff0000) >> 16;
}
```

6.20.4 Friends And Related Function Documentation

6.20.4.1 friend class Triangulator [friend]

Definition at line 327 of file Topology.hh.

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

- src/geometry/Topology.hh

6.21 src::mem::PoolAllocator< T >::rebind< U > Struct - Template Reference

`#include <Allocator.hh>`

Public Types

- typedef PoolAllocator< U > other

6.21.1 Detailed Description

template<class T>template<class U>struct src::mem::PoolAllocator< T >::rebind< U >

Definition at line 69 of file Allocator.hh.

6.21.2 Member Typedef Documentation

6.21.2.1 template<class T > template<class U > typedef PoolAllocator<U> src::mem::PoolAllocator< T >::rebind< U >::other

Definition at line 71 of file Allocator.hh.

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

- src/mem/Allocator.hh

6.22 src::mesh::RegularMesh Class Reference

`#include <RegularMesh.hh>`

Public Types

- typedef Matrix< double, Dynamic, Dynamic, RowMajor > MatrixRM

Public Member Functions

- RegularMesh (int nx, int ny, const float *upper, const float *lower)
- ~RegularMesh ()
- double X (int i, int j)
- double Y (int i, int j)
- double & V (int i, int j)
- void UpdateInterpolator ()
- void GetFunctionValuesAt (int nCoor, float **coor, vector< float > *result)
- void Print ()

Friends

- class SurfaceTopology

6.22.1 Detailed Description

Definition at line 56 of file RegularMesh.hh.

6.22.2 Member Typedef Documentation

6.22.2.1 typedef Matrix<double, Dynamic, Dynamic, RowMajor> src::mesh::RegularMesh::MatrixRM

Definition at line 60 of file RegularMesh.hh.

6.22.3 Constructor & Destructor Documentation

6.22.3.1 src::mesh::RegularMesh::RegularMesh (int *nx*, int *ny*, const float * *upper*, const float * *lower*)

6.22.3.2 src::mesh::RegularMesh::~RegularMesh ( )

6.22.4 Member Function Documentation

6.22.4.1 void src::mesh::RegularMesh::GetFunctionValuesAt (int *nCoor,* float ** *coor,* vector< float > * *result*)

6.22.4.2 void src::mesh::RegularMesh::Print ( )

6.22.4.3 void src::mesh::RegularMesh::UpdateInterpolator ( )

6.22.4.4 double& src::mesh::RegularMesh::V (int *i,* int *j*) [inline]

Definition at line 69 of file RegularMesh.hh.

```
{ return m_values(i,j); }
```

6.22.4.5 double src::mesh::RegularMesh::X (int *i,* int *j*) [inline]

Definition at line 67 of file RegularMesh.hh.

```
{ return m_coords[i*m_nx + j][0]; }
```

6.22.4.6 double src::mesh::RegularMesh::Y (int *i,* int *j*) [inline]

Definition at line 68 of file RegularMesh.hh.

```
{ return m_coords[i*m_nx + j][1]; }
```

6.22.5 Friends And Related Function Documentation

6.22.5.1 friend class SurfaceTopology [friend]

Definition at line 62 of file RegularMesh.hh.

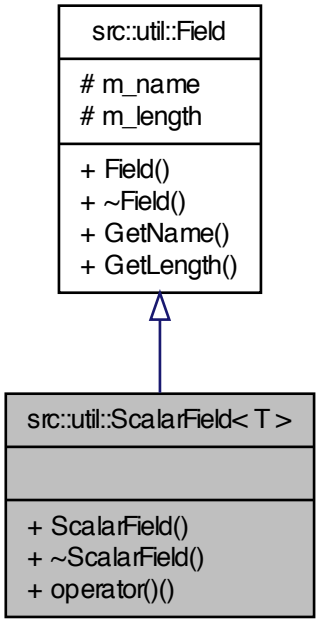
The documentation for this class was generated from the following file:

- src/mesh/RegularMesh.hh

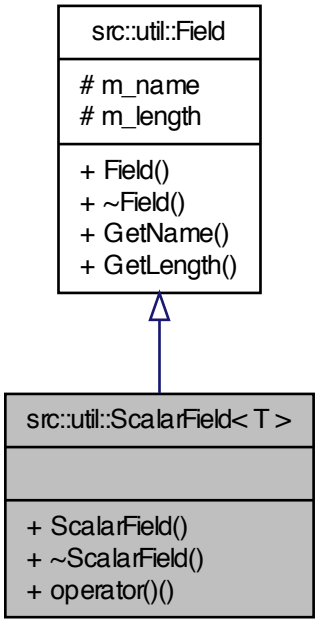
6.23 src::util::ScalarField< T > Class Template Reference

```
#include <ScalarField.hh>
```

Inheritance diagram for src::util::ScalarField< T >:



Collaboration diagram for src::util::ScalarField< T >:



Public Member Functions

- ScalarField (string name, unsigned int length)
- ~ScalarField ()
- T & operator() (unsigned int index)

6.23.1 Detailed Description

template<class T>class src::util::ScalarField< T >

Definition at line 54 of file ScalarField.hh.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 template<class T> src::util::ScalarField< T >::ScalarField (string *name*, unsigned int *length*)

6.23.2.2 template<class T> src::util::ScalarField< T >::~ScalarField ( )

6.23.3 Member Function Documentation

6.23.3.1 template<class T> T& src::util::ScalarField< T >::operator() (unsigned int *index*) [inline]

Definition at line 60 of file ScalarField.hh.

```
{
    return m_data[index];
}
```

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

- src/util/ScalarField.hh

6.24 src::geometry::Site Class Reference

`#include <Topology.hh>`

Static Public Member Functions

- static int InCircle (Site *_a, Site *_b, Site *_c, Site *_d)
- static void CCW (Site *_a, Site *_b, Site *_c, double *result)
- static void CCW (Site *_a, VSite *_b, VSite *_c, double *result)
- static bool Circumcenter (const Site *s1, const Site *s2, const Site *s3, VSite *vs)

Public Attributes

- float ** m_coord
- unsigned int m_id

Friends

- std::ostream & operator<< (std::ostream &os, const Site &mp)

6.24.1 Detailed Description

Definition at line 91 of file Topology.hh.

6.24.2 Member Function Documentation

6.24.2.1 static void src::geometry::Site::CCW (Site * _a, Site * _b, Site * _c, double * result) [inline, static]

Definition at line 128 of file Topology.hh.

```
{
    float *a = *(_a->m_coord);
    float *b = *(_b->m_coord);
    float *c = *(_c->m_coord);

    double ax = a[0];  double ay = a[1];
           double  bx = b[0]; double by = b[1];
           double  cx = c[0]; double cy = c[1];
           *(result) = ((bx*cy-by*cx) - (ax*cy-ay*cx) + (ax*by-ay*bx));
}
```

6.24.2.2 static void src::geometry::Site::CCW (Site * _a, VSite * _b, VSite * _c, double * result) [inline, static]

Definition at line 147 of file Topology.hh.

```
{
    float *a = *(_a->m_coord);
    float *b = _b->m_coord;
    float *c = _c->m_coord;

    double ax = a[0];  double ay = a[1];
           double  bx = b[0]; double by = b[1];
           double  cx = c[0]; double cy = c[1];
           *(result) = ((bx*cy-by*cx) - (ax*cy-ay*cx) + (ax*by-ay*bx));
}
```

6.24.2.3 static bool src::geometry::Site::Circumcenter (const Site * s1, const Site * s2, const Site * s3, VSite * vs) [inline, static]

Definition at line 166 of file Topology.hh.

```
{
    float *n1 = *(s1->m_coord);
    float *n2 = *(s2->m_coord);
    float *n3 = *(s3->m_coord);

    double x1 = n1[0];
    double y1 = n1[1];

    double x2 = n2[0];
    double y2 = n2[1];

    double x3 = n3[0];
    double y3 = n3[1];
```

```
    // Distances relative to point n1 of the triangle
    double x21 = x2 - x1;
    double y21 = y2 - y1;
    double x31 = x3 - x1;
    double y31 = y3 - y1;

    double denominator = 0.5 / (x21 * y31 - y21 * x31);

    if (denominator == 0)
    {
        // Degenerate triangle
        cerr << "Error: Degenerate triangle encountered.." << endl;
        return false;
    }

    // Squares of the lengths of the edges incident to n1
    double length21 = x21 * x21 + y21 * y21;
    double length31 = x31 * x31 + y31 * y31;

    // Calculate offset from n1 of circumcenter
    double x = (y31 * length21 - y21 * length31) * denominator;
    double y = (x21 * length31 - x31 * length21) * denominator;

    // Create a node at the circumcenter
    vs->m_coord[0] = x + x1;
    vs->m_coord[1] = y + y1;

    // Circumcenter calculated
    return true;
}
```

6.24.2.4 static int src::geometry::Site::InCircle (Site * _a, Site * _b, Site * _c, Site * _d) [inline, static]

Definition at line 104 of file Topology.hh.

```
{
    float *a = *(_a->m_coord);
    float *b = *(_b->m_coord);
    float *c = *(_c->m_coord);
    float *d = *(_d->m_coord);

    double x1 = a[0], y1 = a[1];
    double x2 = b[0], y2 = b[1];
    double x3 = c[0], y3 = c[1];
    double x4 = d[0], y4 = d[1];

    return ((y4-y1)*(x2-x3)+(x4-x1)*(y2-y3))*((x4-x3)*(x2-x1)-(y4-y3)*(y2-
  y1)) >
                ((y4-y3)*(x2-x1)+(x4-x3)*(y2-y1))*((x4-x1)*(x2-x3)-(y4-y1)*(
  y2-y3));
}
```

6.24.3 Friends And Related Function Documentation

6.24.3.1 std::ostream& operator<< (std::ostream & *os,* const Site & *mp*) [friend]

Definition at line 218 of file Topology.hh.

```
{
    os << "Site: id(" << mp.m_id << "): " << "(" <<
          (*mp.m_coord)[0] << ", " << (*mp.m_coord)[1] << ")" << endl;
    return os;
}
```

6.24.4 Member Data Documentation

6.24.4.1 float** src::geometry::Site::m_coord

Definition at line 94 of file Topology.hh.

6.24.4.2 unsigned int src::geometry::Site::m_id

Definition at line 95 of file Topology.hh.

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

- src/geometry/Topology.hh

6.25 src::geometry::SiteComparator Class Reference

`#include <Topology.hh>`

Public Member Functions

- bool operator() (const Site &a, const Site &b)

6.25.1 Detailed Description

Definition at line 233 of file Topology.hh.

6.25.2 Member Function Documentation

6.25.2.1 bool src::geometry::SiteComparator::operator() (const Site & *a,* const Site & *b*) [inline]

Definition at line 236 of file Topology.hh.

```
{
    float *coordA = (*(a.m_coord));
    float *coordB = (*(b.m_coord));

    if(coordA[0] != coordB[0])
        return coordA[0] < coordB[0];
    else
        return coordA[1] < coordB[1];
}
```

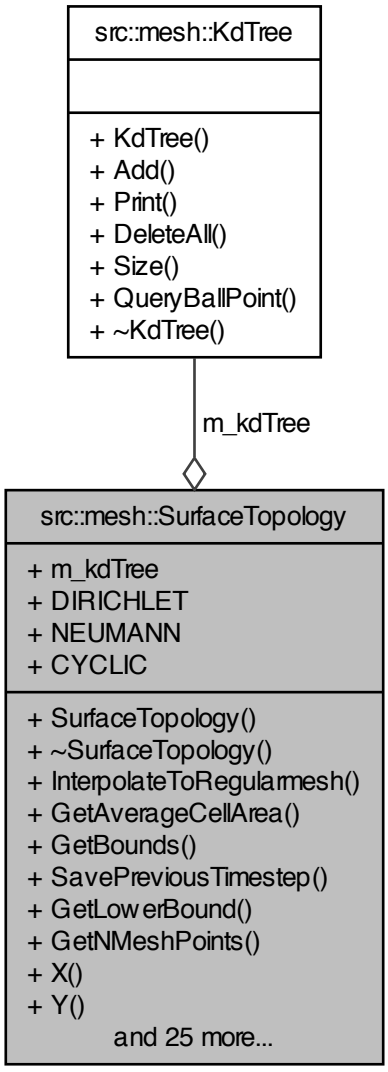
The documentation for this class was generated from the following file:

- src/geometry/Topology.hh

6.26 src::mesh::SurfaceTopology Class Reference

```
#include <SurfaceTopology.hh>
```

Collaboration diagram for src::mesh::SurfaceTopology:



Public Types

- typedef set< int >::const_iterator CatchmentIterator

Public Member Functions

- SurfaceTopology (Config *c)
- ~SurfaceTopology ()
- void InterpolateToRegularmesh (RegularMesh *rm, vector< float > &field) const
- float GetAverageCellArea () const
- void GetBounds (vector< float > &upper, vector< float > &lower) const
- void SavePreviousTimestep () const
- const float * GetLowerBound ()
- unsigned int GetNMeshPoints () const
- float X (int index) const
- float Y (int index) const
- float Z (int index) const
- float Z0 (int index) const
- float Zp (int index) const
- float B (int index) const
- int R (int index) const
- int Dn (int index) const
- const int * D (int index) const
- int C (int index) const
- int S (int index) const
- int O (int index) const
- int SR (int index) const
- const unsigned int ** GetTriangleIndices () const
- const float ** GetVoronoiSides () const
- const float * GetVoronoiCellAreas () const
- const unsigned int * GetNumNeighbours () const
- const unsigned int ** GetNeighbours () const
- const int * GetHull () const
- long int GetNumTriangles () const
- long int GetNumFaces () const
- long int GetNumVoronoiVertices () const
- const VSite * GetVoronoiVertices () const
- void UpdateZ (ScalarField< float > *z) const
- void UpdateNetwork ()
- void PrintNode (int index) const
- CatchmentIterator CatchmentsBegin () const
- CatchmentIterator CatchmentsEnd () const

Public Attributes

- KdTree * m_kdTree

Static Public Attributes

- static const int DIRICHLET
- static const int NEUMANN
- static const int CYCLIC

Friends

- class Triangulator
- class SurfaceTopologyOutput

6.26.1 Detailed Description

Definition at line 76 of file SurfaceTopology.hh.

6.26.2 Member Typedef Documentation

6.26.2.1 typedef set<int>::const_iterator src::mesh::SurfaceTopology::Catchment-Iterator

Definition at line 145 of file SurfaceTopology.hh.

6.26.3 Constructor & Destructor Documentation

6.26.3.1 src::mesh::SurfaceTopology::SurfaceTopology (Config * *c*)

6.26.3.2 src::mesh::SurfaceTopology::~SurfaceTopology ( )

6.26.4 Member Function Documentation

6.26.4.1 float src::mesh::SurfaceTopology::B (int *index*) const [inline]

Definition at line 157 of file SurfaceTopology.hh.

```
{return m_rawGeometry[index][3];}   /* BC */
```

6.26.4.2 int src::mesh::SurfaceTopology::C (int *index*) const [inline]

Definition at line 162 of file SurfaceTopology.hh.

```
{return m_catchmentIds[index];}       /* Catchment-Id */
```

6.26.4.3 CatchmentIterator src::mesh::SurfaceTopology::CatchmentsBegin ( ) const [inline]

Definition at line 189 of file SurfaceTopology.hh.

```
{ return m_catchments.begin(); };
```

6.26.4.4 CatchmentIterator src::mesh::SurfaceTopology::CatchmentsEnd ( ) const [inline]

Definition at line 190 of file SurfaceTopology.hh.

```
{ return m_catchments.end(); };
```

6.26.4.5 const int* src::mesh::SurfaceTopology::D (int *index*) const [inline]

Definition at line 161 of file SurfaceTopology.hh.

```
{return m_donors[index];}      /* Donors-list */
```

6.26.4.6 int src::mesh::SurfaceTopology::Dn (int *index*) const [inline]

Definition at line 160 of file SurfaceTopology.hh.

```
{return m_donorCounts[index];}       /* Donors-count */
```

6.26.4.7 float src::mesh::SurfaceTopology::GetAverageCellArea ( ) const [inline]

Definition at line 147 of file SurfaceTopology.hh.

```
{return m_averageCellArea;}
```

6.26.4.8 void src::mesh::SurfaceTopology::GetBounds (vector< float > & *upper*, vector< float > & *lower*) const

6.26.4.9 const int* src::mesh::SurfaceTopology::GetHull ( ) const [inline]

Definition at line 174 of file SurfaceTopology.hh.

```
{return (const int*) m_triangulator->GetHull();}
```

6.26.4.10 const float* src::mesh::SurfaceTopology::GetLowerBound ( ) [inline]

Definition at line 150 of file SurfaceTopology.hh.

```
{ return m_lower; }
```

6.26.4.11 const unsigned int** src::mesh::SurfaceTopology::GetNeighbours ( ) const [inline]

Definition at line 173 of file SurfaceTopology.hh.

```
{return (const unsigned int**) m_triangulator->GetNeighbours();}
```

6.26.4.12 unsigned int src::mesh::SurfaceTopology::GetNMeshPoints ( ) const [inline]

Definition at line 151 of file SurfaceTopology.hh.

```
{return m_nMeshPoints;}
```

6.26.4.13 long int src::mesh::SurfaceTopology::GetNumFaces ( ) const [inline]

Definition at line 176 of file SurfaceTopology.hh.

```
{return m_triangulator->GetNumFaces();}
```

6.26.4.14 const unsigned int* src::mesh::SurfaceTopology::GetNumNeighbours ( ) const [inline]

Definition at line 172 of file SurfaceTopology.hh.

```
{return (const unsigned int*) m_triangulator->GetNumNeighbours();}
```

6.26.4.15 long int src::mesh::SurfaceTopology::GetNumTriangles ( ) const [inline]

Definition at line 175 of file SurfaceTopology.hh.

```
{return m_triangulator->GetNumTriangles();}
```

6.26.4.16 long int src::mesh::SurfaceTopology::GetNumVoronoiVertices ( ) const [inline]

Definition at line 177 of file SurfaceTopology.hh.

```
{return m_triangulator->GetNumVoronoiVertices();}
```

6.26.4.17 const unsigned int** src::mesh::SurfaceTopology::GetTriangleIndices ( ) const [inline]

Definition at line 169 of file SurfaceTopology.hh.

```
{return (const unsigned int **)m_triangulator->GetTriangleIndices();}
```

6.26.4.18 const float* src::mesh::SurfaceTopology::GetVoronoiCellAreas ( ) const [inline]

Definition at line 171 of file SurfaceTopology.hh.

```
{return (const float*) m_triangulator->GetVoronoiCellAreas();}
```

6.26.4.19 const float** src::mesh::SurfaceTopology::GetVoronoiSides ( ) const [inline]

Definition at line 170 of file SurfaceTopology.hh.

```
{return (const float **) m_triangulator->GetVoronoiSides();}
```

6.26.4.20 const VSite* src::mesh::SurfaceTopology::GetVoronoiVertices ( ) const [inline]

Definition at line 178 of file SurfaceTopology.hh.

```
{return (const VSite*) m_triangulator->GetVoronoiVertices();}
```

6.26.4.21 void src::mesh::SurfaceTopology::InterpolateToRegularmesh (RegularMesh * *rm*, vector< float > & *field*) const

6.26.4.22 int src::mesh::SurfaceTopology::O (int *index*) const [inline]

Definition at line 164 of file SurfaceTopology.hh.

```
{return m_originalOrder[index];}      /* original-order */
```

6.26.4.23 void src::mesh::SurfaceTopology::PrintNode (int *index*) const

6.26.4.24 int src::mesh::SurfaceTopology::R (int *index*) const [inline]

Definition at line 159 of file SurfaceTopology.hh.

```
{return m_receivers[index];}          /* Receiver-id */
```

6.26.4.25 int src::mesh::SurfaceTopology::S (int *index*) const [inline]

Definition at line 163 of file SurfaceTopology.hh.

```
{return m_stack[index];}              /* Stack-order */
```

6.26.4.26 void src::mesh::SurfaceTopology::SavePreviousTimestep ( ) const

6.26.4.27 int src::mesh::SurfaceTopology::SR (int *index*) const [inline]

Definition at line 166 of file SurfaceTopology.hh.

```
{return m_receiversSillCorrected[index];}
```

6.26.4.28 void src::mesh::SurfaceTopology::UpdateNetwork ( )

6.26.4.29 void src::mesh::SurfaceTopology::UpdateZ (ScalarField< float > * *z*) const

6.26.4.30 float src::mesh::SurfaceTopology::X (int *index*) const [inline]

Definition at line 152 of file SurfaceTopology.hh.

```
{return m_rawGeometry[index][0];}   /* x-coordinate */
```

6.26.4.31 float src::mesh::SurfaceTopology::Y (int *index*) const [inline]

Definition at line 153 of file SurfaceTopology.hh.

```
{return m_rawGeometry[index][1];}   /* y-coordinate */
```

6.26.4.32 float src::mesh::SurfaceTopology::Z (int *index*) const [inline]

Definition at line 154 of file SurfaceTopology.hh.

```
{return m_rawGeometry[index][2];}   /* z-coordinate */
```

6.26.4.33 float src::mesh::SurfaceTopology::Z0 (int *index*) const [inline]

Definition at line 155 of file SurfaceTopology.hh.

```
{return m_z0[index];}              /* z at time-step 0 */
```

6.26.4.34 float src::mesh::SurfaceTopology::Zp (int *index*) const [inline]

Definition at line 156 of file SurfaceTopology.hh.

```
{return m_zp[index];}              /* z at previous time-step */
```

6.26.5 Friends And Related Function Documentation

6.26.5.1 friend class SurfaceTopologyOutput [friend]

Definition at line 80 of file SurfaceTopology.hh.

6.26.5.2 friend class Triangulator [friend]

Definition at line 79 of file SurfaceTopology.hh.

6.26.6 Member Data Documentation

6.26.6.1 const int src::mesh::SurfaceTopology::CYCLIC [static]

Definition at line 84 of file SurfaceTopology.hh.

6.26.6.2 const int src::mesh::SurfaceTopology::DIRICHLET [static]

Definition at line 82 of file SurfaceTopology.hh.

6.26.6.3 KdTree* src::mesh::SurfaceTopology::m_kdTree

Definition at line 186 of file SurfaceTopology.hh.

6.26.6.4 const int src::mesh::SurfaceTopology::NEUMANN [static]

Definition at line 83 of file SurfaceTopology.hh.

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

- src/mesh/SurfaceTopology.hh

6.27 src::mesh::SurfaceTopologyOutput Class Reference

`#include <SurfaceTopologyOutput.hh>`

Public Types

- enum Attributes_t { SurfaceTopologyOutput_WriteMesh = (1<<0), SurfaceTopologyOutput_WriteNetwork = (1<<1) }
- typedef enum src::mesh::SurfaceTopologyOutput::Attributes_t Attributes

Public Member Functions

- SurfaceTopologyOutput (const Model *m, Config *c)
- ~SurfaceTopologyOutput ()
- void RegisterScalarField (ScalarField< float > *sf)
- void Write ()

Friends

- class SurfaceTopology

6.27.1 Detailed Description

Definition at line 59 of file SurfaceTopologyOutput.hh.

6.27.2 Member Typedef Documentation

6.27.2.1 typedef enum src::mesh::SurfaceTopologyOutput::Attributes_t src::mesh::SurfaceTopologyOutput::Attributes

6.27.3 Member Enumeration Documentation

6.27.3.1 enum src::mesh::SurfaceTopologyOutput::Attributes_t

Enumerator:

SurfaceTopologyOutput_WriteMesh

SurfaceTopologyOutput_WriteNetwork

Definition at line 67 of file SurfaceTopologyOutput.hh.

```
                        {
    SurfaceTopologyOutput_WriteMesh      = (1<<0),
    SurfaceTopologyOutput_WriteNetwork   = (1<<1),
}Attributes;
```

6.27.4 Constructor & Destructor Documentation

6.27.4.1 src::mesh::SurfaceTopologyOutput::SurfaceTopologyOutput (const Model * *m*, Config * *c*)

6.27.4.2 src::mesh::SurfaceTopologyOutput::~SurfaceTopologyOutput ( )

6.27.5 Member Function Documentation

6.27.5.1 void src::mesh::SurfaceTopologyOutput::RegisterScalarField (ScalarField< float > * *sf*)

6.27.5.2 void src::mesh::SurfaceTopologyOutput::Write ( )

6.27.6 Friends And Related Function Documentation

6.27.6.1 friend class SurfaceTopology [friend]

Definition at line 65 of file SurfaceTopologyOutput.hh.

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

- src/mesh/SurfaceTopologyOutput.hh

6.28 src::util::Timer Class Reference

`#include <Timer.hh>`

Public Member Functions

- Timer ()
- ~Timer ()

Static Public Member Functions

- static double Elapsed (const Timer &begin, const Timer &end)

6.28.1 Detailed Description

Definition at line 53 of file Timer.hh.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 src::util::Timer::Timer ( )

6.28.2.2 src::util::Timer::~Timer ( )

6.28.3 Member Function Documentation

6.28.3.1 static double src::util::Timer::Elapsed (const Timer & *begin*, const Timer & *end*) [static]

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

- src/util/Timer.hh

6.29 src::util::TimeSeries Class Reference

`#include <TimeSeries.hh>`

Public Member Functions

- TimeSeries (const Model *m, Config *c, string paramName)
- ~TimeSeries ()
- void GetCurrentFieldValue (vector< double > *result)

6.29.1 Detailed Description

Definition at line 66 of file TimeSeries.hh.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 src::util::TimeSeries::TimeSeries (const Model * *m*, Config * *c*, string *paramName*)

6.29.2.2 src::util::TimeSeries::~TimeSeries ( )

6.29.3 Member Function Documentation

6.29.3.1 void src::util::TimeSeries::GetCurrentFieldValue (vector< double > * *result*)

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

- src/util/TimeSeries.hh

6.30 src::geometry::Triangulator Class Reference

`#include <Triangulator.hh>`

Public Types

- enum Attributes_t { Triangulator_SuperTriangle = (1<<0), Triangulator_TriangleIndices = (1<<1), Triangulator_TriangleNeighbours = (1<<2), Triangulator_VoronoiVertices = (1<<3), Triangulator_VoronoiSides = (1<<4), Triangulator_VoronoiCellAreas = (1<<5), Triangulator_NodeNeighbours = (1<<6) }
- typedef enum src::geometry::Triangulator::Attributes_t Attributes

Public Member Functions

- Triangulator (int ns, float **s, unsigned int attr)
- ~Triangulator ()
- unsigned int ** GetTriangleIndices ()
- float ** GetVoronoiSides ()
- float * GetVoronoiCellAreas ()
- unsigned int * GetNumNeighbours ()
- unsigned int ** GetNeighbours ()
- int * GetHull ()
- long int GetNumTriangles ()
- long int GetNumFaces ()
- long int GetNumVoronoiVertices ()
- VSite * GetVoronoiVertices ()
- void ComputeBound (float *minx, float *miny, float *maxx, float *maxy)

Friends

- std::ostream & operator<< (std::ostream &os, const Triangulator &t)

6.30.1 Detailed Description

Definition at line 56 of file Triangulator.hh.

6.30.2 Member Typedef Documentation

6.30.2.1 typedef enum src::geometry::Triangulator::Attributes_t src::geometry::Triangulator::Attributes

6.30.3 Member Enumeration Documentation

6.30.3.1 enum src::geometry::Triangulator::Attributes_t

Enumerator:

Triangulator_SuperTriangle

Triangulator_TriangleIndices

Triangulator_TriangleNeighbours

Triangulator_VoronoiVertices

Triangulator_VoronoiSides

Triangulator_VoronoiCellAreas

Triangulator_NodeNeighbours

Definition at line 98 of file Triangulator.hh.

```
{
    Triangulator_SuperTriangle          = (1<<0),
    Triangulator_TriangleIndices        = (1<<1),
    Triangulator_TriangleNeighbours     = (1<<2),
    Triangulator_VoronoiVertices        = (1<<3),
    Triangulator_VoronoiSides           = (1<<4),
    Triangulator_VoronoiCellAreas        = (1<<5),
    Triangulator_NodeNeighbours         = (1<<6)
}Attributes;
```

6.30.4 Constructor & Destructor Documentation

6.30.4.1 src::geometry::Triangulator::Triangulator (int *ns,* float ** *s,* unsigned int *attr*)

6.30.4.2 src::geometry::Triangulator::~Triangulator ( )

6.30.5 Member Function Documentation

6.30.5.1 void src::geometry::Triangulator::ComputeBound (float * *minx,* float * *miny,* float * *maxx,* float * *maxy*)

6.30.5.2 int* src::geometry::Triangulator::GetHull ( )

6.30.5.3 unsigned int** src::geometry::Triangulator::GetNeighbours ( )

6.30.5.4 long int src::geometry::Triangulator::GetNumFaces ( )

6.30.5.5 unsigned int* src::geometry::Triangulator::GetNumNeighbours ( )

6.30.5.6 long int src::geometry::Triangulator::GetNumTriangles ( )

6.30.5.7 long int src::geometry::Triangulator::GetNumVoronoiVertices ( )

6.30.5.8 unsigned int** src::geometry::Triangulator::GetTriangleIndices ( )

6.30.5.9 float* src::geometry::Triangulator::GetVoronoiCellAreas ( )

6.30.5.10 float** src::geometry::Triangulator::GetVoronoiSides ( )

6.30.5.11 VSite* src::geometry::Triangulator::GetVoronoiVertices ( )

6.30.6 Friends And Related Function Documentation

6.30.6.1 std::ostream& operator<< (std::ostream & *os*, const Triangulator & *t*) [friend]

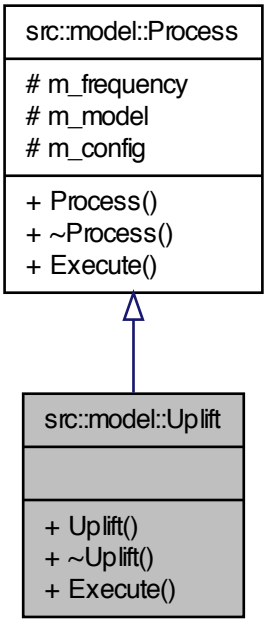
The documentation for this class was generated from the following file:

- src/geometry/Triangulator.hh

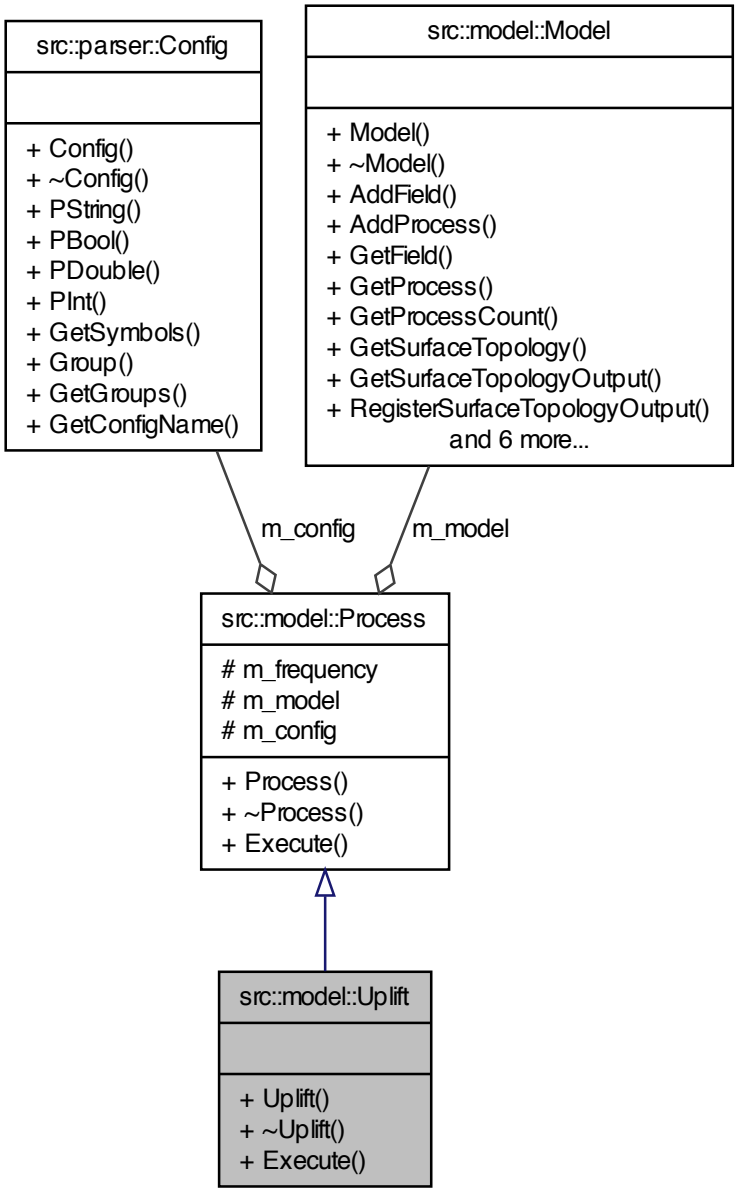
6.31 src::model::Uplift Class Reference

```
#include <Uplift.hh>
```

Inheritance diagram for src::model::Uplift:



Collaboration diagram for src::model::Uplift:



Public Member Functions

- Uplift (const Model *m, Config *c)
- ~Uplift ()
- void Execute ()

6.31.1 Detailed Description

Definition at line 57 of file Uplift.hh.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 src::model::Uplift::Uplift (const Model * *m*, Config * *c*)

6.31.2.2 src::model::Uplift::~Uplift ( )

6.31.3 Member Function Documentation

6.31.3.1 void src::model::Uplift::Execute ( ) [virtual]

Implements src::model::Process.

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

- src/model/Uplift.hh

6.32 src::geometry::VSite Class Reference

`#include <Topology.hh>`

Public Attributes

- float m_coord [2]

Friends

- std::ostream & operator<< (std::ostream &os, const VSite &mp)

6.32.1 Detailed Description

Definition at line 65 of file Topology.hh.

6.32.2 Friends And Related Function Documentation

6.32.2.1 std::ostream& operator<< (std::ostream & *os*, const VSite & *mp*) [friend]

Definition at line 75 of file Topology.hh.

```
{
    os << "VSite: (" << (mp.m_coord)[0] <<
          ", " << (mp.m_coord)[1] << ")" << endl;
    return os;
}
```

6.32.3 Member Data Documentation

6.32.3.1 float src::geometry::VSite::m_coord[2]

Definition at line 68 of file Topology.hh.

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

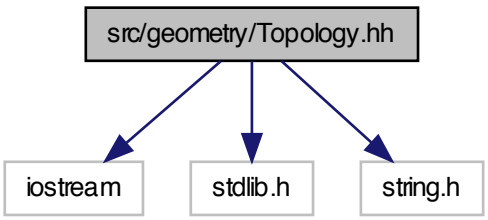
- src/geometry/Topology.hh

Chapter 7

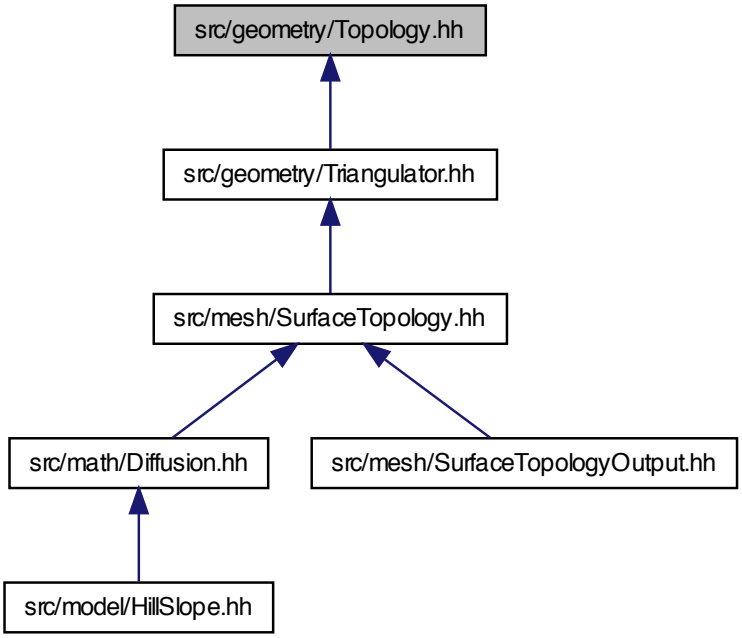
File Documentation

7.1 src/geometry/Topology.hh File Reference

`#include <iostream>` `#include <stdlib.h>` `#include <string.-h>` Include dependency graph for Topology.hh:



This graph shows which files directly or indirectly include this file:



Classes

- class src::geometry::VSite
- class src::geometry::Site
- class src::geometry::SiteComparator
- class src::geometry::Edge
- class src::geometry::QuadEdge

Namespaces

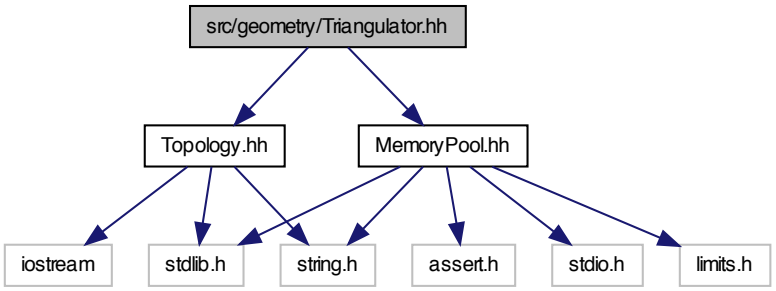
- namespace src
- namespace src::geometry

Functions

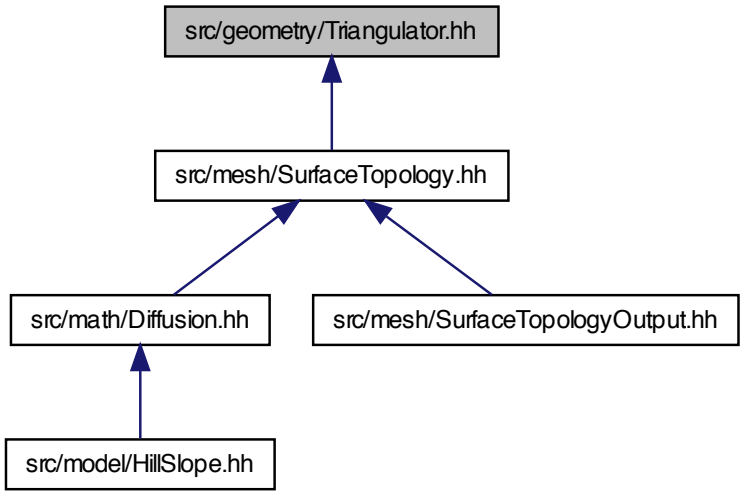
- float src::geometry::FABS (float a)

7.2 src/geometry/Triangulator.hh File Reference

`#include <Topology.hh>` `#include <MemoryPool.hh>` Include dependency graph for Triangulator.hh:



This graph shows which files directly or indirectly include this file:



Classes

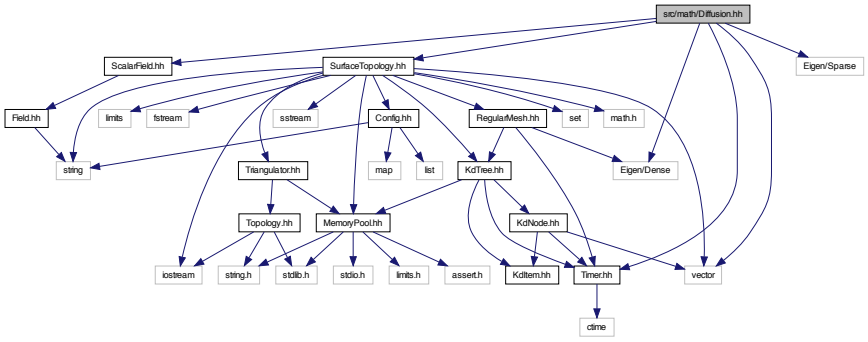
- class src::geometry::Triangulator

Namespaces

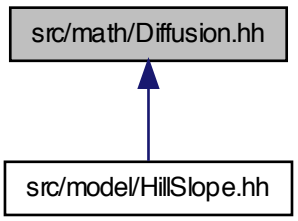
- namespace src
- namespace src::geometry

7.3 src/math/Diffusion.hh File Reference

`#include <vector> #include <SurfaceTopology.hh> #include <ScalarField.hh> #include <Timer.hh> #include <Eigen/Dense> #include <Eigen/Sparse>` Include dependency graph for Diffusion.hh:



This graph shows which files directly or indirectly include this file:



Classes

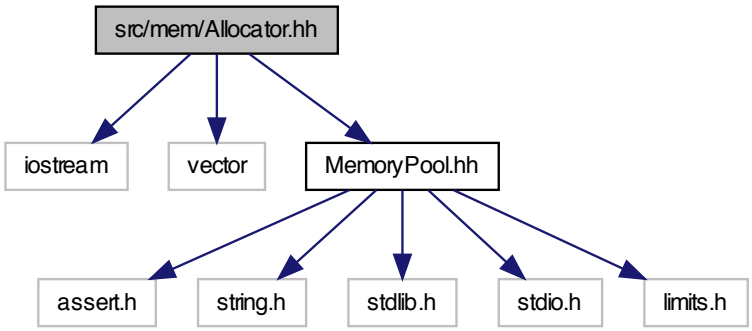
- class src::math::Diffusion
- struct src::math::Diffusion::Coord_t

Namespaces

- namespace src
- namespace src::math

7.4 src/mem/Allocator.hh File Reference

`#include <iostream> #include <vector> #include <MemoryPool.hh>` Include dependency graph for Allocator.hh:



Classes

- class src::mem::PoolAllocator< T >
- struct src::mem::PoolAllocator< T >::rebind< U >
- class src::mem::PoolAllocator< void >
- struct **src::mem::PoolAllocator**< **void** >**::rebind**< **U** >

Namespaces

- namespace src
- namespace src::mem

Functions

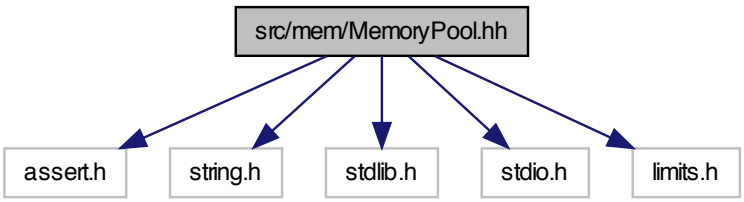
- template<class T >
  bool src::mem::operator== (const PoolAllocator< T > &, const PoolAllocator< T > &)
- template<class T >
  bool src::mem::operator!= (const PoolAllocator< T > &, const PoolAllocator< T > &)
- void src::mem::InitPool ()
- void src::mem::FinalisePool ()

Variables

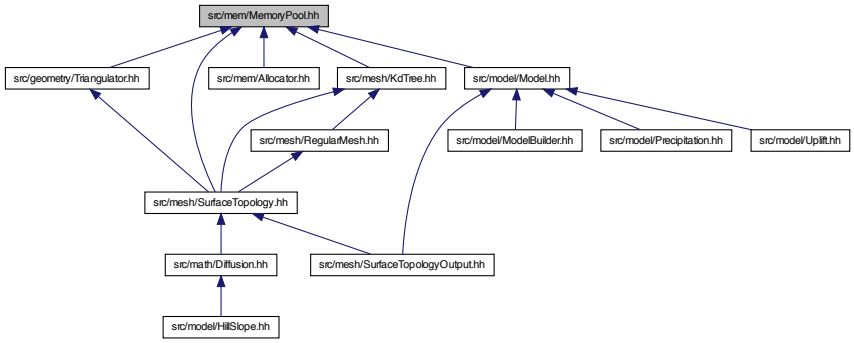
- pthread_mutex_t src::mem::mutex
- static std::vector< MemoryPool * > src::mem::pools

7.5 src/mem/MemoryPool.hh File Reference

`#include <assert.h>` `#include <string.h>` `#include <stdlib.h>` `#include <stdio.h>` `#include <limits.h>` Include dependency graph for MemoryPool.hh:



This graph shows which files directly or indirectly include this file:



Classes

- struct src::mem::Chunk
- class src::mem::MemoryPool

Namespaces

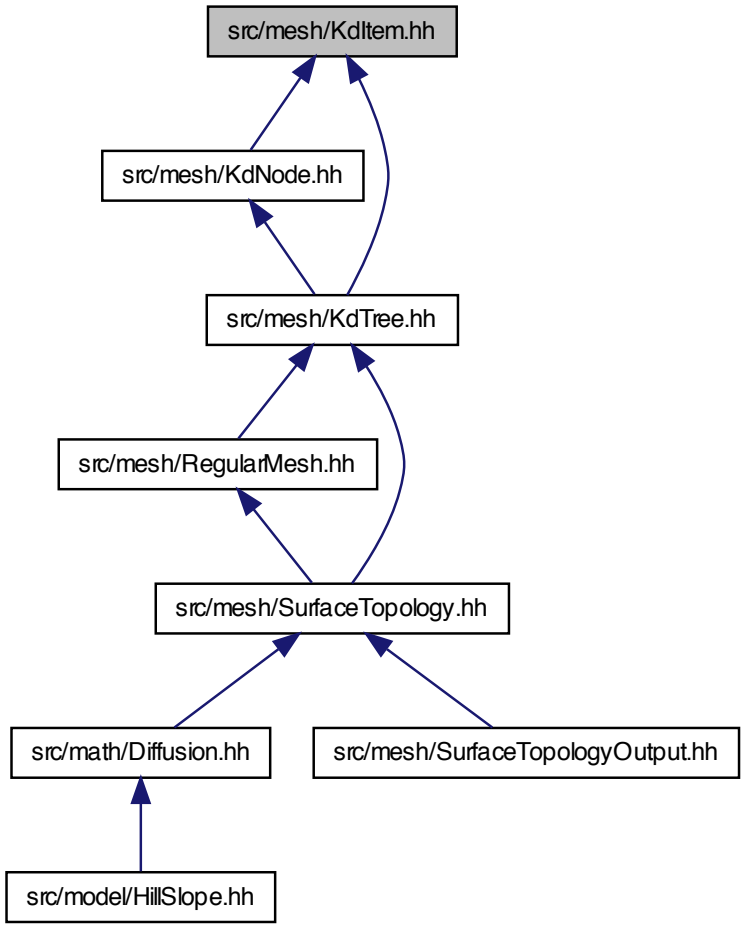
- namespace src
- namespace src::mem

Variables

- const int src::mem::CHUNK_ARRAY_DELTA = 10
- const int src::mem::INVALID = -1
- const int src::mem::SUCCESS = 1
- const int src::mem::FAILURE = 0
- const int src::mem::TWO_EXP16 = 65535

7.6 src/mesh/KdItem.hh File Reference

This graph shows which files directly or indirectly include this file:



Classes

- class src::mesh::KdItem

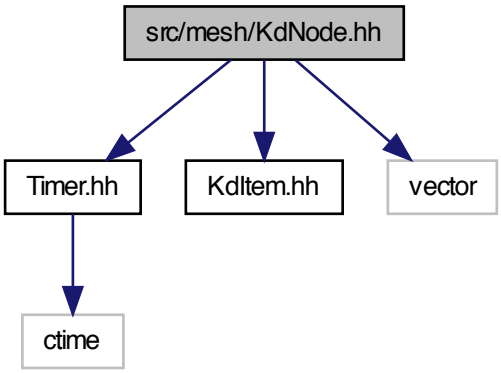
Namespaces

- namespace src

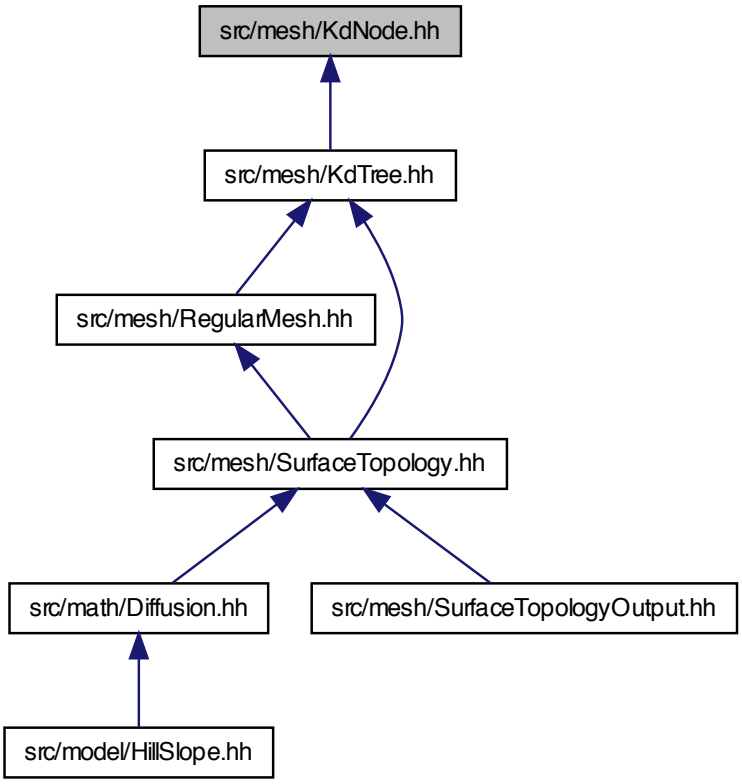
- namespace src::mesh

7.7 src/mesh/KdNode.hh File Reference

`#include <Timer.hh>` `#include <KdItem.hh>` `#include <vector>` ×
Include dependency graph for KdNode.hh:



This graph shows which files directly or indirectly include this file:



Classes

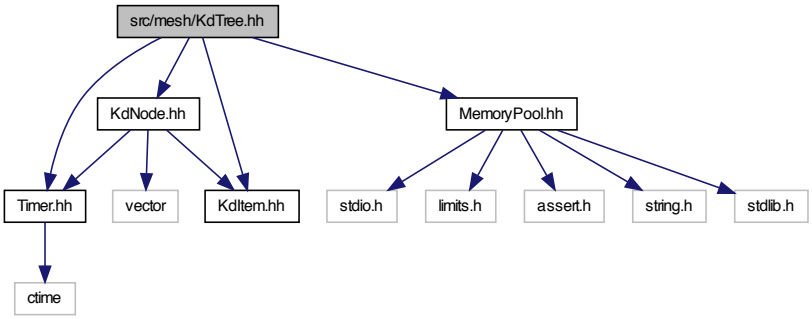
- class src::mesh::KdNode
- struct **src::mesh::KdNode::KdItemSort**

Namespaces

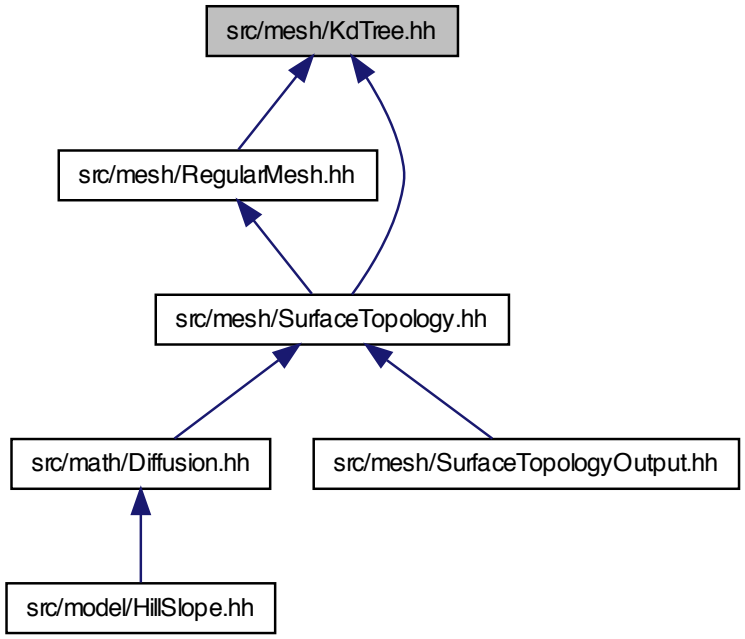
- namespace src
- namespace src::mesh

7.8 src/mesh/KdTree.hh File Reference

`#include <Timer.hh>  #include <KdItem.hh>  #include <KdNode.hh> #include <MemoryPool.hh>` Include dependency graph for KdTree.hh:



This graph shows which files directly or indirectly include this file:



Classes

- class src::mesh::KdTree

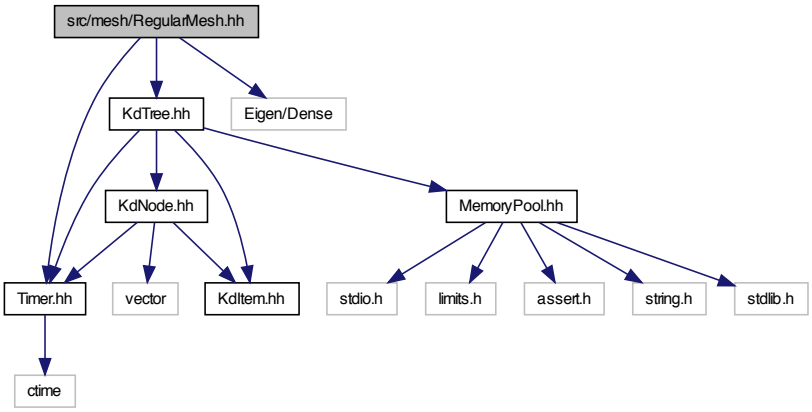
Namespaces

- namespace src
- namespace src::mesh

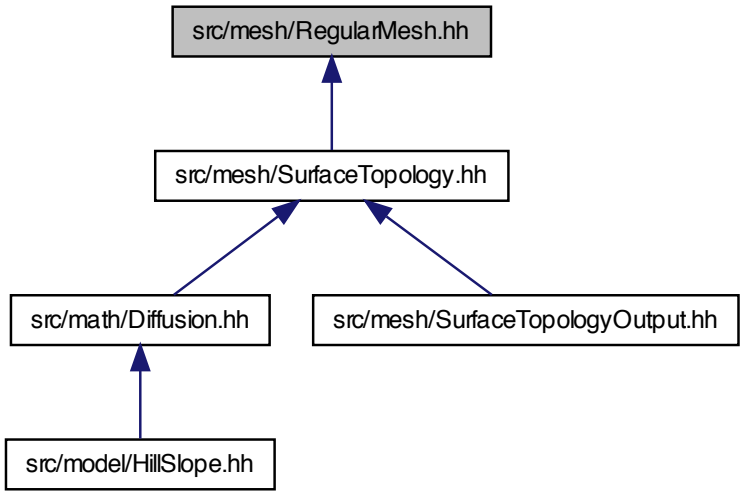
7.9 src/mesh/RegularMesh.hh File Reference

```
#include <KdTree.hh>   #include <Timer.hh>   #include <-
```

Eigen/Dense> Include dependency graph for RegularMesh.hh:



This graph shows which files directly or indirectly include this file:



Classes

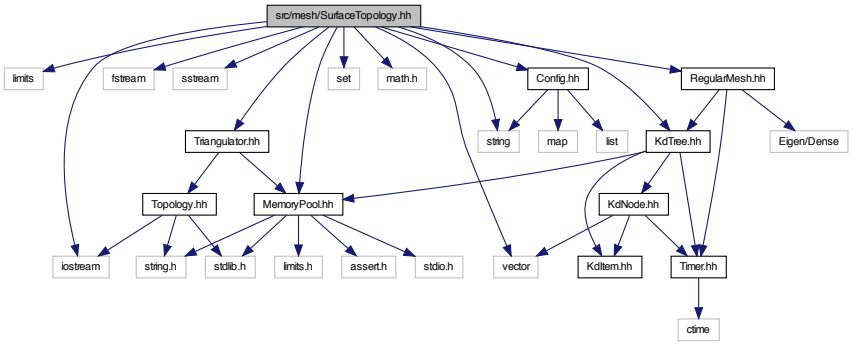
- class src::mesh::RegularMesh

Namespaces

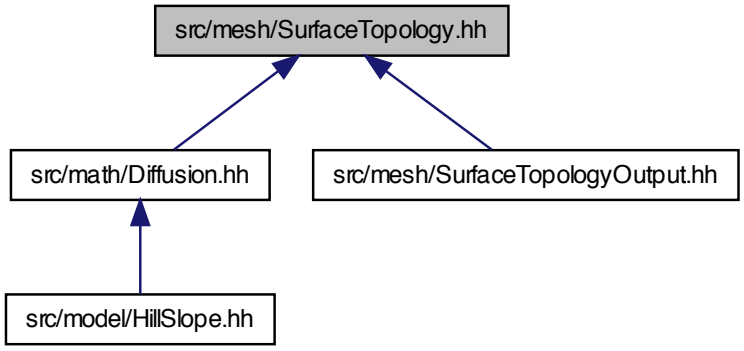
- namespace src
- namespace src::mesh

7.10 src/mesh/SurfaceTopology.hh File Reference

`#include <limits> #include <iostream> #include <fstream> #include <sstream> #include <string> #include <vector> #include <set> #include <math.h> #include <MemoryPool.hh> #include <Triangulator.hh> #include <Config.hh> #include <KdTree.hh> #include <RegularMesh.hh>` Include dependency graph for SurfaceTopology.hh:



This graph shows which files directly or indirectly include this file:



Classes

- class src::mesh::SurfaceTopology

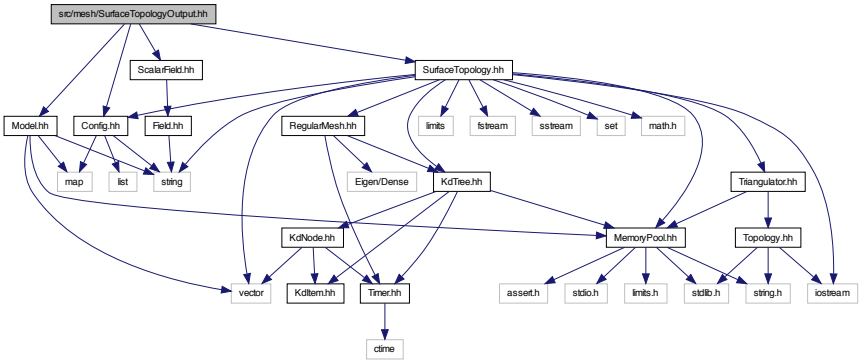
Namespaces

- namespace src
- namespace src::util
- namespace src::mesh

7.11 src/mesh/SurfaceTopologyOutput.hh File Reference

`#include <Model.hh>` `#include <SurfaceTopology.hh>` `#include <ScalarField.hh>` `#include <Config.hh>` Include dependency graph for

SurfaceTopologyOutput.hh:



Classes

- class src::mesh::SurfaceTopologyOutput

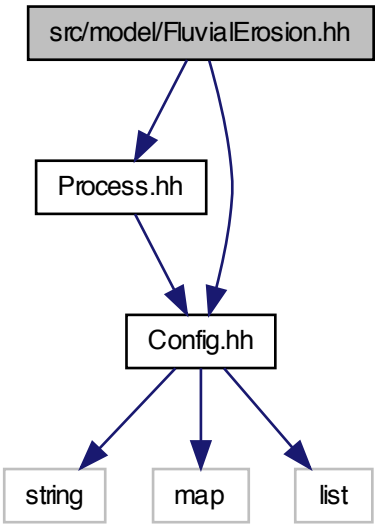
Namespaces

- namespace src
- namespace src::mesh

7.12 src/model/FluvialErosion.hh File Reference

`#include <Process.hh>` `#include <Config.hh>` Include dependency

graph for FluvialErosion.hh:



Classes

- class src::model::FluvialErosion

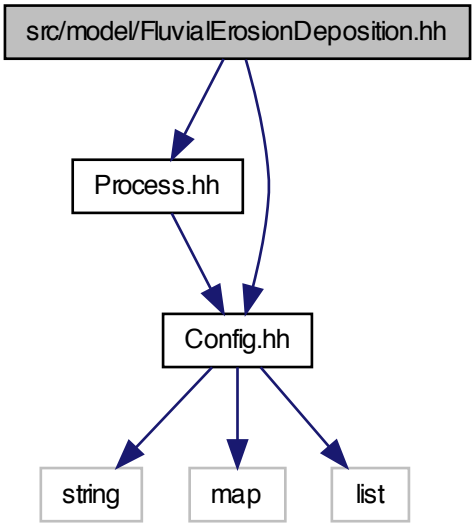
Namespaces

- namespace src
- namespace src::model

7.13 src/model/FluvialErosionDeposition.hh File Reference

`#include <Process.hh>` `#include <Config.hh>` Include dependency

graph for FluvialErosionDeposition.hh:



Classes

- class src::model::FluvialErosionDeposition

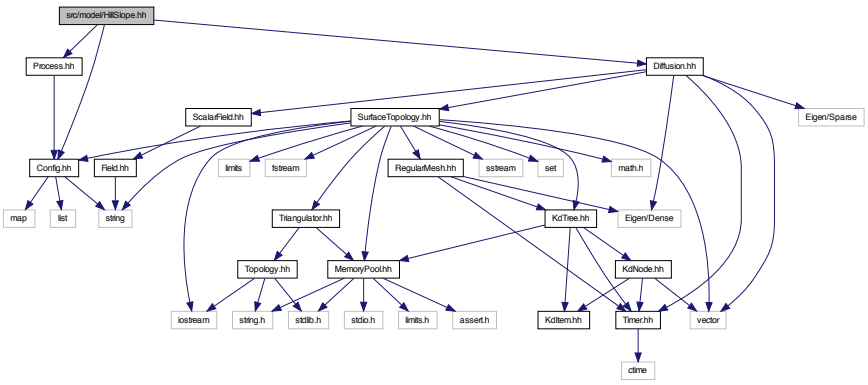
Namespaces

- namespace src
- namespace src::model

7.14 src/model/HillSlope.hh File Reference

```
#include <Process.hh>  #include <Config.hh>  #include <-
```

Diffusion.hh> Include dependency graph for HillSlope.hh:



Classes

- class src::model::HillSlope

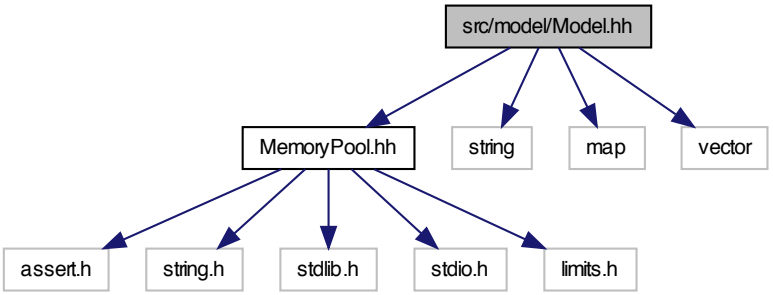
Namespaces

- namespace src
- namespace src::model

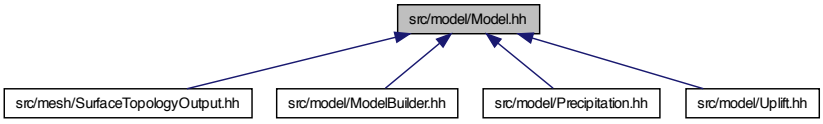
7.15 src/model/Model.hh File Reference

#include <MemoryPool.hh> #include <string> #include <map> ×

#include <vector> Include dependency graph for Model.hh:



This graph shows which files directly or indirectly include this file:



Classes

- class src::model::Model

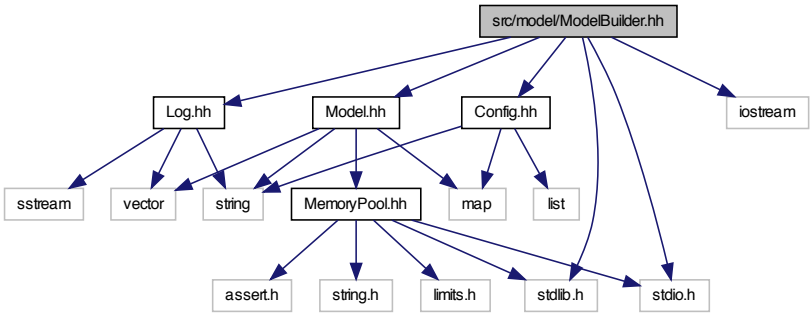
Namespaces

- namespace src
- namespace src::util
- namespace src::mesh
- namespace src::parser
- namespace src::model

7.16 src/model/ModelBuilder.hh File Reference

#include <Config.hh> #include <Log.hh> #include <Model.-
hh> #include <iostream> #include <stdio.h> #include <stdlib.-

h> Include dependency graph for ModelBuilder.hh:



Classes

- class src::model::ModelBuilder

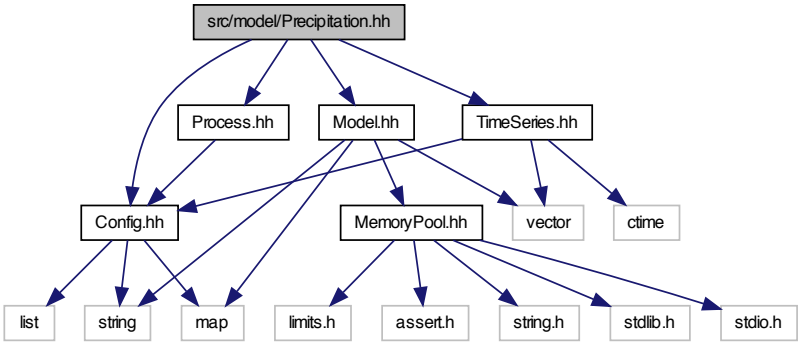
Namespaces

- namespace src
- namespace src::mesh
- namespace src::model

7.17 src/model/Precipitation.hh File Reference

#include <Process.hh> #include <Model.hh> #include <-Config.hh> #include <TimeSeries.hh> Include dependency graph for

Precipitation.hh:



Classes

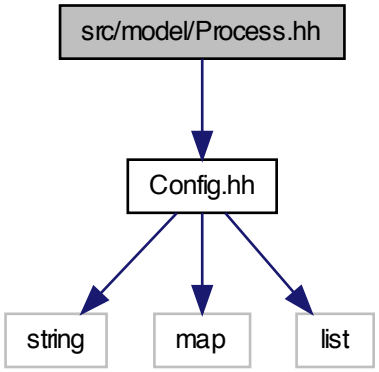
- class src::model::Precipitation

Namespaces

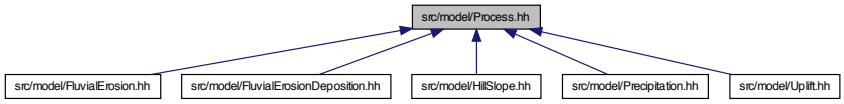
- namespace src
- namespace src::model

7.18 src/model/Process.hh File Reference

`#include <Config.hh>` Include dependency graph for Process.hh:



This graph shows which files directly or indirectly include this file:



Classes

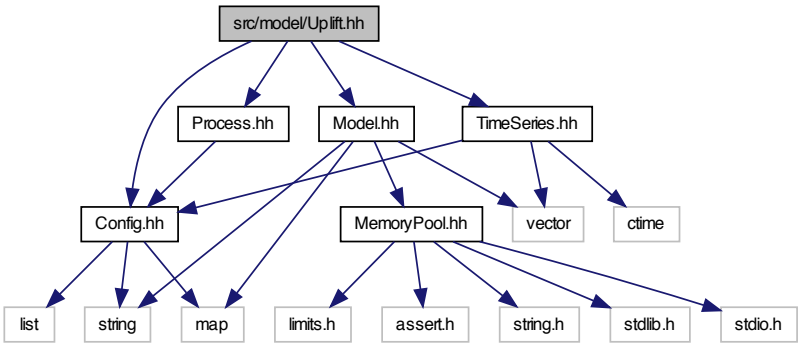
- class src::model::Process

Namespaces

- namespace src
- namespace src::model

7.19 src/model/Uplift.hh File Reference

`#include <Process.hh>  #include <Model.hh>  #include <-Config.hh> #include <TimeSeries.hh>` Include dependency graph for Uplift.hh:



Classes

- class src::model::Uplift

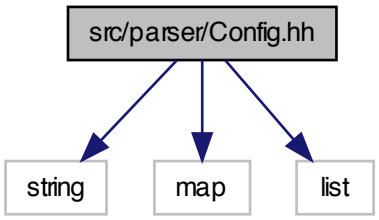
Namespaces

- namespace src
- namespace src::model

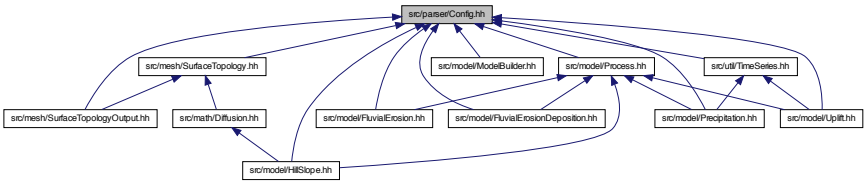
7.20 src/parser/Config.hh File Reference

`#include <string> #include <map> #include <list>` Include de-

pendency graph for Config.hh:



This graph shows which files directly or indirectly include this file:



Classes

- class src::parser::Config

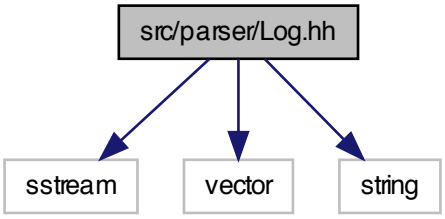
Namespaces

- namespace src
- namespace src::parser

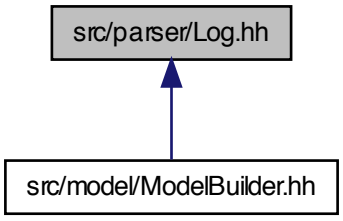
7.21 src/parser/Log.hh File Reference

```
#include <sstream> #include <vector> #include <string> x
```

Include dependency graph for Log.hh:



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace src
- namespace src::parser

Defines

- #define LogError(A) ((logLevel >= LOG_ERROR)?((A),0):(0))
- #define LogInfo(A) ((logLevel >= LOG_INFO)?((A),0):(0))
- #define LogDebug(A) ((logLevel >= LOG_DEBUG)?((A),0):(0))

Enumerations

- enum src::parser::LogLevel { src::parser::LOG_QUIET, src::parser::LOG_ERROR, src::parser::LOG_INFO, src::parser::LOG_DEBUG }

Functions

- void src::parser::debugBreak ()
- vector< string > src::parser::split (const string &s, char delim)

Variables

- LogLevel src::parser::logLevel

7.21.1 Define Documentation

7.21.1.1 #define LogDebug(A) ((logLevel >= LOG_DEBUG)?((A),0):(0))

Definition at line 54 of file Log.hh.

7.21.1.2 #define LogError(A) ((logLevel >= LOG_ERROR)?((A),0):(0))

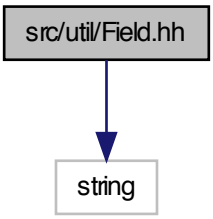
Definition at line 52 of file Log.hh.

7.21.1.3 #define LogInfo(A) ((logLevel >= LOG_INFO)?((A),0):(0))

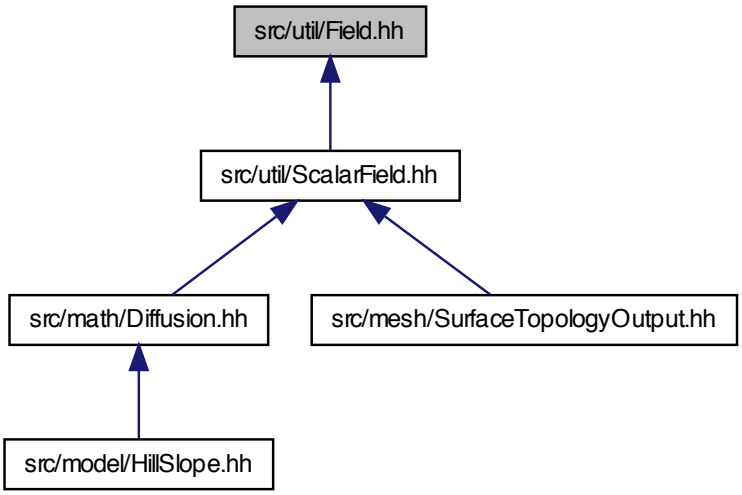
Definition at line 53 of file Log.hh.

7.22 src/util/Field.hh File Reference

`#include <string>` Include dependency graph for Field.hh:



This graph shows which files directly or indirectly include this file:



Classes

- class src::util::Field

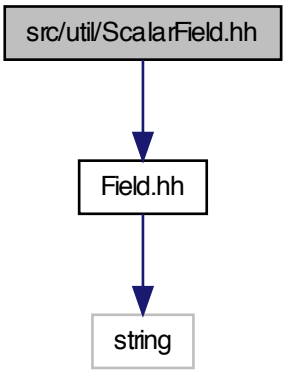
Namespaces

- namespace src

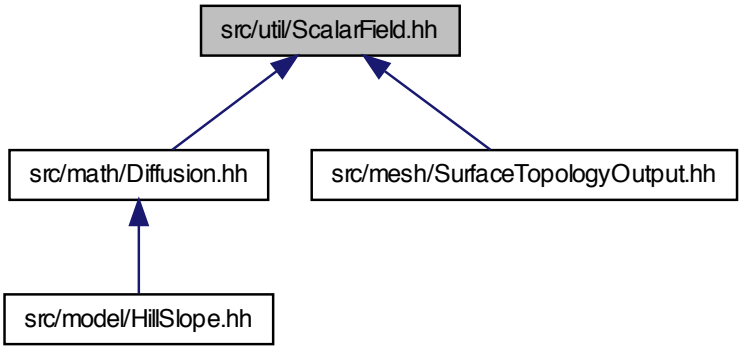
- namespace src::util

7.23 src/util/ScalarField.hh File Reference

`#include <Field.hh>` Include dependency graph for ScalarField.hh:



This graph shows which files directly or indirectly include this file:



Classes

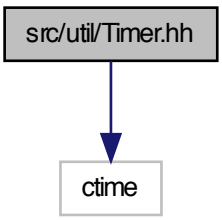
- class src::util::ScalarField< T >

Namespaces

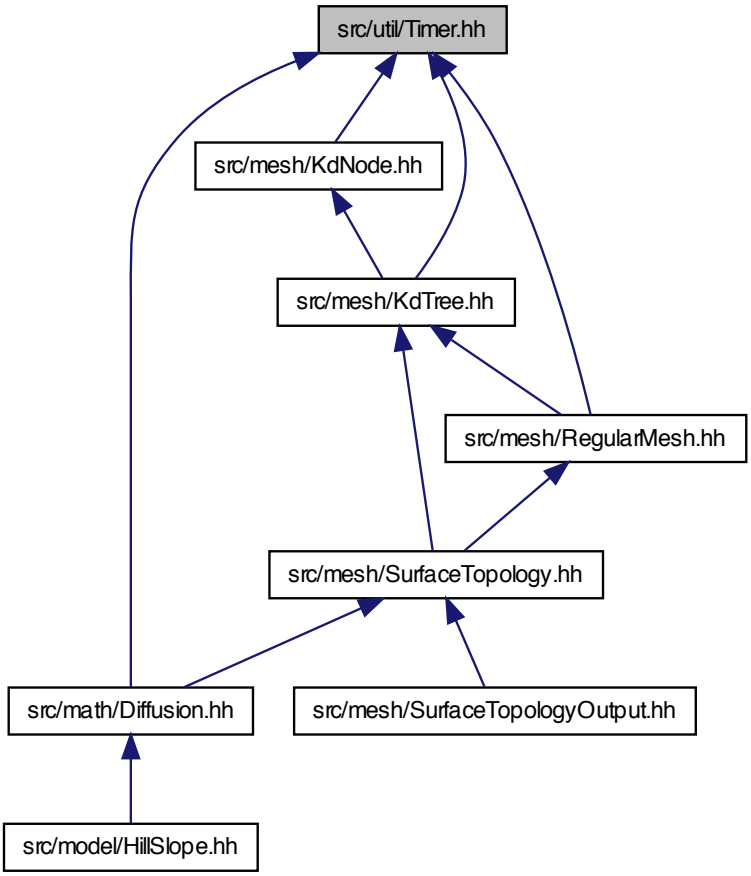
- namespace src
- namespace src::util

7.24 src/util/Timer.hh File Reference

`#include <ctime>` Include dependency graph for Timer.hh:



This graph shows which files directly or indirectly include this file:



Classes

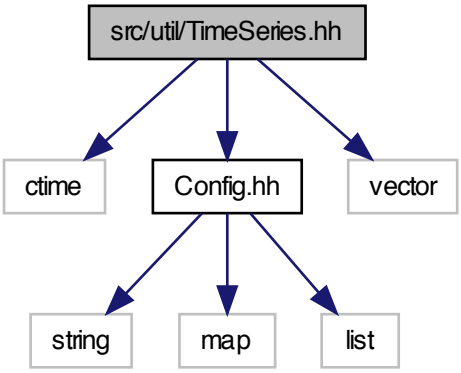
- class src::util::Timer

Namespaces

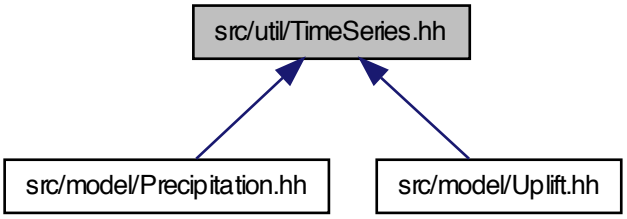
- namespace src
- namespace src::util

7.25 src/util/TimeSeries.hh File Reference

`#include <ctime> #include <Config.hh> #include <vector>` ×
Include dependency graph for TimeSeries.hh:



This graph shows which files directly or indirectly include this file:



Classes

- class src::util::TimeSeries

Namespaces

- namespace src
- namespace src::model
- namespace src::util