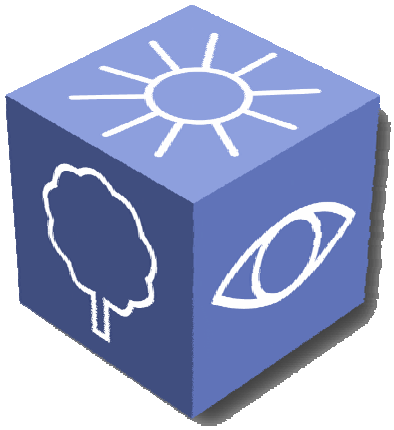


GAMETOOLS

ADVANCED TOOLS FOR DEVELOPING HIGHLY REALISTIC COMPUTER GAMES

DUMMY MODULES FOR GEOMETRY

Document identifier:	**GameTools-4-D4.1-01-1-1-Dummy Modules for Geometry**
Date: (use "update field" Word function, right mouse button)	**28/04/2005**
Work package:	**WP04: Geometry**
Partner(s):	**UJI, UPV**
Leading Partner:	**UJI**
Document status:	**APPROVED**
Deliverable identifier:	**D4.1**

Abstract: Dummy Modules Report for Geometry

Delivery Slip

	Name	Partner	Date	Signature
From	Miguel Chover	UJI	19-04-05	
Reviewed by	Moderator and reviewers	All	21-04-05	
Approved by	Moderator and reviewers	All	28-04-05	

Document Log

Issue	Date	Comment	Author
1-0	19-04-05	First draft	Miguel Chover (UJI)
1-1	21-04-05	Final Version	Miguel Chover (UJI)

Document Change Record

Issue	Item	Reason for Change

Files

Software Products	User files / URL
Word	gametools-ist-2-004363-4-d4.1-01-1-1-dummy modules for geometry.doc (use "update field" Word function)

Contents

1 Overview **1**

2 GameTools Geometry Modules Namespace Index **5**

2.1 GameTools Geometry Modules Namespace List 5

3 GameTools Geometry Modules Hierarchical Index **7**

3.1 GameTools Geometry Modules Class Hierarchy 7

4 GameTools Geometry Modules Class Index **9**

4.1 GameTools Geometry Modules Class List 9

5 GameTools Geometry Modules File Index **11**

5.1 GameTools Geometry Modules File List 11

6 GameTools Geometry Modules Namespace Documentation **13**

6.1 Geometry Namespace Reference 13

7 GameTools Geometry Modules Class Documentation **17**

7.1 Geometry::CustomStripifier Class Reference 17

7.2 Geometry::GeometryBasedSimplifier Class Reference 19

7.3 Geometry::ImageBasedSimplifier Class Reference 21

7.4 Geometry::LodStripsConstructor Class Reference 23

7.5 Geometry::LodStripsLibrary Class Reference 25

7.6 Geometry::LodTreeConstructor Class Reference 28

7.7 Geometry::LodTreeLibrary Class Reference 30

7.8 Geometry::Mesh Class Reference 34

7.9 Geometry::MeshSimplificationSequence Class Reference 37

7.10 Geometry::MeshSimplificationSequence::Step Struct Reference 39

7.11 Geometry::MeshSimplifier Class Reference 40

7.12 Geometry::MeshStripifier Class Reference 43

7.13 Geometry::Serializable Class Reference 45
7.14 Geometry::Serializer Class Reference 47
7.15 Geometry::SubMesh Class Reference 50
7.16 Geometry::TreeSimplificationSequence Class Reference 53
7.17 Geometry::TreeSimplificationSequence::Step Struct Reference 56
7.18 Geometry::TreeSimplifier Class Reference 57
7.19 Geometry::VertexBuffer Class Reference 59

8 GameTools Geometry Modules File Documentation **63**
8.1 GeoBase.h File Reference 63
8.2 GeoLodStripsConstructor.h File Reference 65
8.3 GeoLodStripsLibrary.h File Reference 66
8.4 GeoLodTreeConstructor.h File Reference 67
8.5 GeoLodTreeLibrary.h File Reference 68
8.6 GeoMesh.h File Reference 69
8.7 GeoMeshSimplifier.h File Reference 70
8.8 GeoMeshSimpSequence.h File Reference 71
8.9 GeoMeshStripifier.h File Reference 72
8.10 GeoSerializable.h File Reference 73
8.11 GeoSerializer.h File Reference 74
8.12 GeoSubMesh.h File Reference 75
8.13 GeoTreeSimplifier.h File Reference 76
8.14 GeoTreeSimpSequence.h File Reference 77
8.15 GeoVector2.h File Reference 78
8.16 GeoVector3.h File Reference 79
8.17 GeoVertexBuffer.h File Reference 80

Chapter 1

Overview

This work package consists in developing a continuous multiresolution model for polygonal objects that includes connectivity information and uses basic primitives like triangle strips to reduce the amount of information stored and the time required for rendering. Our results will be specifically applied to allow the efficient rendering of complex outdoor scenes, including plants and trees.

The work consists of the following main tasks:

1. Developing a new multiresolution model with connectivity information for general meshes; we call this model LODStrips.
2. Developing a new multiresolution model to represent plants and trees; we call this model LODTrees.

To achieve this goal we are developing two modules, one for each multiresolution model:

- **Module for General Meshes.**

This module contains functions that handle the levels of detail of objects made of polygonal meshes. With these functions, the user will be able to get information about the geometry and the current level of detail. Besides, one has the possibility of changing the complexity level of the model interactively. The meshes the module works with use triangle strips to reduce storage usage and to speed up realistic rendering.

This module is implemented in the *Geometry::LodStripsLibrary* class, which inherits from the *Ogre::Renderable* and *Ogre::Movable* classes for its integration with the engine.

- **Module for Plants and Trees.**

This module contains functions that handle the levels of detail of plant and tree models. These models support separate processing of leaves and branches. Branches, including the trunk, are handled by the general mesh module. Leaves are represented using their own specific variable multiresolution model. In order to get the proper foliage geometry, the user can interact with this module according to his necessities.

All this functionality is integrated into the *Geometry::LodTreeLibrary* class, which has as parent the *Ogre::Renderable* and *Ogre::Movable* classes for its correct behaviour inside the engine.

These modules are accompanied by four other support modules that will be independent:

- **Simplification Module.**

This module is used by both models, general mesh and the plant and tree models. It contains functions that take a triangle mesh as input and generate simplified versions of 3D objects made out of triangles, while preserving its appearance.

It receives a pointer to a *Geometry::Mesh* object containing the model to be simplified. For general meshes, this input is a triangle mesh, while for leaf sets (foliage) is a set of leaves, each made of two texture-mapped

triangles.

As result of the simplification process, it returns a simplification sequence represented by an object that contains:

- For general meshes: for each simplification step the module returns the edge to be collapsed, the two triangles begin removed, and the new triangles added to the representation. All the information of the steps is included in a *Geometry::MeshSimplificationSequence* object.
- For leaf sets (foliage): for each simplification step the module returns the collapsed leaf, the two leaves being removed, and the new leaf added to the representation. This information is included in a *Geometry::TreeSimplificationSequence* object.

This module consists of the *Geometry::MeshSimplifier* for general meshes and the *Geometry::TreeSimplifier* for plants and trees. The *Geometry::MeshSimplifier* is a virtual class and must be derived in a class that implements the simplification algorithm. This is done by means of the classes *Geometry::ImageBasedSimplifier* and *Geometry::GeometryBasedSimplifier*.

- **Stripification (Strip Search) Module.**

This module implements methods that extract triangle strips from triangle meshes. It receives a pointer to a *Geometry::Mesh* object containing the model to be stripified and returns the stripified mesh, contained in another *Geometry::Mesh* object. It consists of the virtual class *Geometry::Stripifier* and the class *Geometry::CustomStripifier* that implements the stripifying algorithm.

- **Construction module for general multiresolution models.**

This module stores general multiresolution mesh representations according to the LODStrip format. Its functionality is supported by the *LodStrips Constructor* class.

It takes as inputs the strips provided by the stripification module (*Geometry::Mesh* class) and the edge collapse sequence computed by the simplification module (*Geometry::MeshSimplificationSequence* class). It returns a file containing the mesh generated according to the *Geometry::Mesh* object structure. Besides, it outputs a LodStrips file with the multiresolution information.

- **Construction module for tree multiresolution models.**

This module stores tree multiresolution representations using our own specific file format. It consists of the *Geometry::TreeConstructor* class.

It receives a file including the geometry information of the foliage (*Geometry::Mesh* object) and also a file describing the simplification steps of the leaves (*Geometry::TreeSimplificationSequence* object). The output this module returns is a file describing the multiresolution tree object. In addition, it references the file with the original set of leaves, the one with the original geometry of the trunk and branches and the LodStrips file obtained after the simplification of the trunk and branches mesh.

Besides, this modules structure requires the auxiliary *Geometry::Serializer*, *Geometry::Serializable* and *Geometry::Mesh* classes. The *Mesh* class represents the structure that will store the information about the mesh that we will work with. The *Geometry::Serializer* and *Geometry::Serializable* classes implement a method that reads information from a file and stores it in our mesh structure. It also provides a method to save this information to a file.

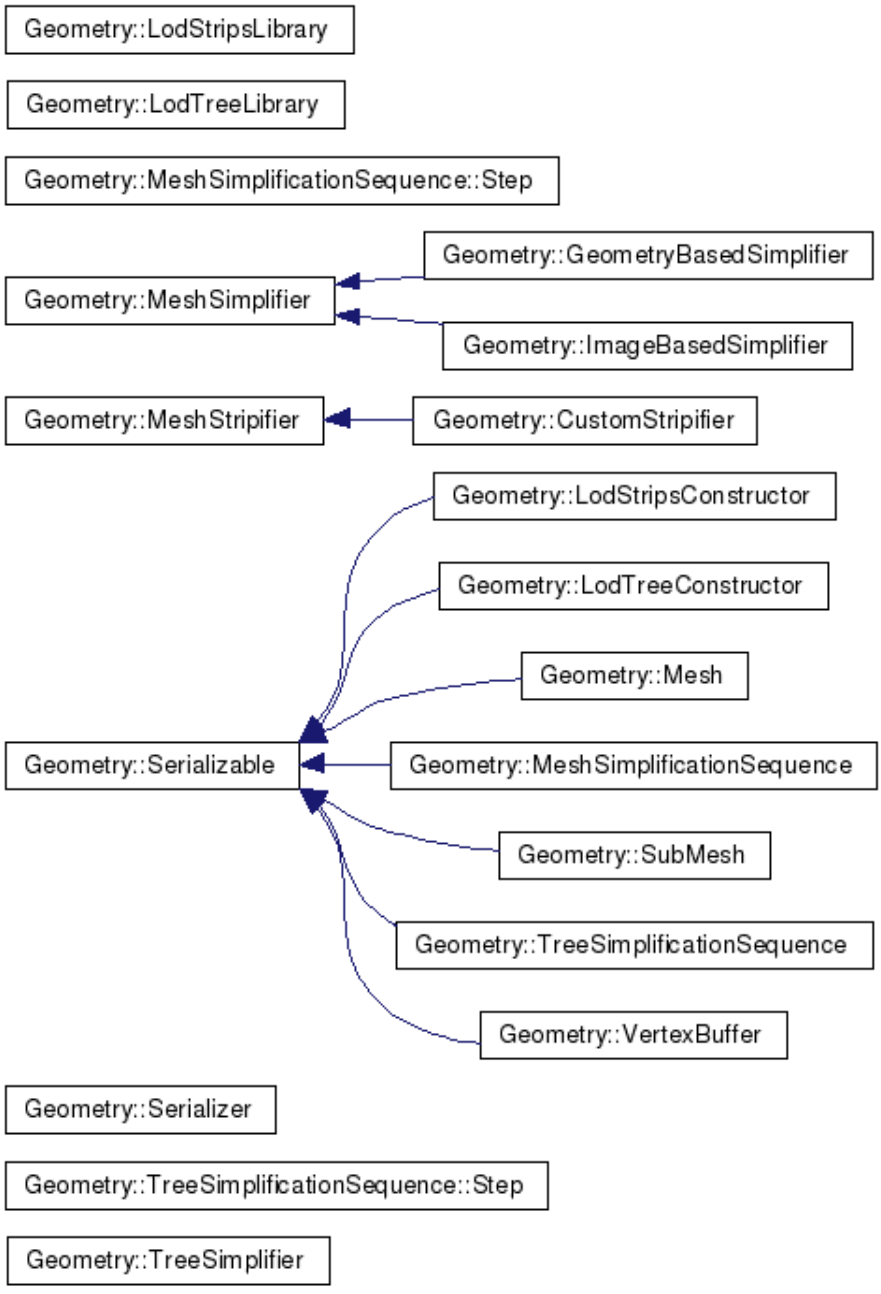


Figure 1.1: Workpackage general class diagram .

Chapter 2

GameTools Geometry Modules Namespace Index

2.1 GameTools Geometry Modules Namespace List

Here is a list of all namespaces with brief descriptions:

Geometry (This namespace contains all classes related to geometry operations) 13

Chapter 3

GameTools Geometry Modules Hierarchical Index

3.1 GameTools Geometry Modules Class Hierarchy

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

Geometry::LodStripsLibrary . . . 25
Geometry::LodTreeLibrary . . . 30
Geometry::MeshSimplificationSequence::Step . . . 39
Geometry::MeshSimplifier . . . 40
  Geometry::GeometryBasedSimplifier . . . 19
  Geometry::ImageBasedSimplifier . . . 21
Geometry::MeshStripifier . . . 43
  Geometry::CustomStripifier . . . 17
Geometry::Serializable . . . 45
  Geometry::LodStripsConstructor . . . 23
  Geometry::LodTreeConstructor . . . 28
  Geometry::Mesh . . . 34
  Geometry::MeshSimplificationSequence . . . 37
  Geometry::SubMesh . . . 50
  Geometry::TreeSimplificationSequence . . . 53
  Geometry::VertexBuffer . . . 59
Geometry::Serializer . . . 47
Geometry::TreeSimplificationSequence::Step . . . 56
Geometry::TreeSimplifier . . . 57

Chapter 4

GameTools Geometry Modules Class Index

4.1 GameTools Geometry Modules Class List

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

Geometry::CustomStripifier 17
Geometry::GeometryBasedSimplifier (Implementation of a simplification algorithm based on geometry information) 19
Geometry::ImageBasedSimplifier (Implementation of a simplification algorithm based on images) 21
Geometry::LodStripsConstructor (Construction module for general multiresolution models) . . 23
Geometry::LodStripsLibrary (LodStripsLibrary interface class) 25
Geometry::LodTreeConstructor (Construction module for multiresolution tree models) 28
Geometry::LodTreeLibrary (LodTreeLibrary interface class) 30
Geometry::Mesh (Mesh class interface) 34
Geometry::MeshSimplificationSequence (Represents the simplification sequence applied to a given mesh) 37
Geometry::MeshSimplificationSequence::Step (Represents a simplification step in the sequence) 39
Geometry::MeshSimplifier (Mesh simplificator interface) 40
Geometry::MeshStripifier (Stripifier interface class) 43
Geometry::Serializable (Serializable interface) 45
Geometry::Serializer (Serializer interface) 47
Geometry::SubMesh (SubMesh interface) 50
Geometry::TreeSimplificationSequence (Represents the simplification sequence applied to a given mesh) 53
Geometry::TreeSimplificationSequence::Step (Represents a Simplification Step in the sequence) 56
Geometry::TreeSimplifier (Tree Simplifier interface) 57
Geometry::VertexBuffer (VertexBuffer interface Class) 59

Chapter 5

GameTools Geometry Modules File Index

5.1 GameTools Geometry Modules File List

Here is a list of all files with brief descriptions:

GeoBase.h 63
GeoLodStripsConstructor.h 65
GeoLodStripsLibrary.h 66
GeoLodTreeConstructor.h 67
GeoLodTreeLibrary.h 68
GeoMesh.h 69
GeoMeshSimplifier.h 70
GeoMeshSimpSequence.h 71
GeoMeshStripifier.h 72
GeoSerializable.h 73
GeoSerializer.h 74
GeoSubMesh.h 75
GeoTreeSimplifier.h 76
GeoTreeSimpSequence.h 77
GeoVector2.h 78
GeoVector3.h 79
GeoVertexBuffer.h 80

Chapter 6

GameTools Geometry Modules Namespace Documentation

6.1 Geometry Namespace Reference

This namespace contains all classes related to geometry operations.

Classes

- class Geometry::LodStripsConstructor

  Construction module for general multiresolution models.

- class Geometry::LodStripsLibrary

  LodStripsLibrary interface class.

- class Geometry::LodTreeConstructor

  Construction module for multiresolution tree models.

- class Geometry::LodTreeLibrary

  LodTreeLibrary interface class.

- class Geometry::Mesh

  Mesh class interface.

- class Geometry::MeshSimplifier

  Mesh simplificator interface.

- class Geometry::ImageBasedSimplifier

  Implementation of a simplification algorithm based on images.

- class Geometry::GeometryBasedSimplifier

  Implementation of a simplification algorithm based on geometry information.

- class Geometry::MeshSimplificationSequence

Represents the simplification sequence applied to a given mesh.

- struct Geometry::MeshSimplificationSequence::Step
  Represents a simplification step in the sequence.
- class Geometry::MeshStripifier
  Stripifier interface class.
- class Geometry::CustomStripifier
- class Geometry::Serializable
  Serializable interface.
- class Geometry::Serializer
  Serializer interface.
- class Geometry::SubMesh
  SubMesh interface.
- class Geometry::TreeSimplifier
  Tree Simplifier interface.
- class Geometry::TreeSimplificationSequence
  Represents the simplification sequence applied to a given mesh.
- struct Geometry::TreeSimplificationSequence::Step
  Represents a Simplification Step in the sequence.
- class Geometry::VertexBuffer
  VertexBuffer interface Class.

Typedefs

- typedef float Real
- typedef unsigned int Index
- typedef unsigned int uint32
- typedef unsigned short uint16
- typedef std::string String

Enumerations

- enum MeshType { GEO_TRIANGLE_LIST, GEO_TRIANGLE_STRIPS }

Variables

- const unsigned short VERTEX_EMPTY = 0x00
- const unsigned short VERTEX_POSITION = 0x01
- const unsigned short VERTEX_NORMAL = 0x02
- const unsigned short VERTEX_TEXCOORDS = 0x04
- const unsigned short VERTEX_ALL = VERTEX_POSITION | VERTEX_NORMAL | VERTEX_-TEXCOORDS

6.1.1 Detailed Description

This namespace contains all classes related to geometry operations.

Geometry namespace includes classes for the following modules:

- Serialization: for file loading and saving.
- Simplification: for mesh simplification algorithms.
- Stripification: methods for finding triangle strips from a mesh.
- Construction: builds a LODStrip file from the output of simplification and stripification modules.

6.1.2 Typedef Documentation

6.1.2.1 typedef unsigned int Geometry::Index

Definition at line 18 of file GeoBase.h.

6.1.2.2 typedef float Geometry::Real

Definition at line 17 of file GeoBase.h.

6.1.2.3 typedef std::string Geometry::String

Definition at line 21 of file GeoBase.h.

6.1.2.4 typedef unsigned short Geometry::uint16

Definition at line 20 of file GeoBase.h.

6.1.2.5 typedef unsigned int Geometry::uint32

Definition at line 19 of file GeoBase.h.

6.1.3 Enumeration Type Documentation

6.1.3.1 enum Geometry::MeshType

Enumeration values:

GEO_TRIANGLE_LIST

GEO_TRIANGLE_STRIPS

Definition at line 31 of file GeoBase.h.

6.1.4 Variable Documentation

6.1.4.1 const unsigned short Geometry::VERTEX_ALL = VERTEX_POSITION | VERTEX_NORMAL | VERTEX_TEXCOORDS [static]

Definition at line 28 of file GeoBase.h.

6.1.4.2 const unsigned short Geometry::VERTEX_EMPTY = 0x00 [static]

Definition at line 24 of file GeoBase.h.

6.1.4.3 const unsigned short Geometry::VERTEX_NORMAL = 0x02 [static]

Definition at line 26 of file GeoBase.h.

6.1.4.4 const unsigned short Geometry::VERTEX_POSITION = 0x01 [static]

Definition at line 25 of file GeoBase.h.

6.1.4.5 const unsigned short Geometry::VERTEX_TEXCOORDS = 0x04 [static]

Definition at line 27 of file GeoBase.h.

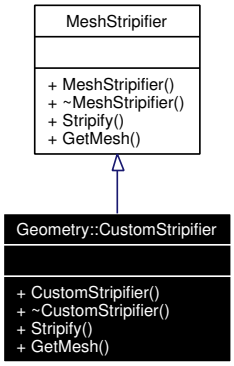
Chapter 7

GameTools Geometry Modules Class Documentation

7.1 Geometry::CustomStripifier Class Reference

`#include <GeoMeshStripifier.h>`

Inheritance diagram for Geometry::CustomStripifier:



Public Member Functions

- CustomStripifier (const Geometry::Mesh *)

  Class constructor, receives as a parameter a const pointer to the object that describes a mesh.

- virtual ~CustomStripifier (void)

  Class destructor.

- void Stripify ()

  Starts the stripification process. This is a custom stripification method.

- Mesh * GetMesh ()

  Returns the stripified mesh.

7.1.1 Constructor & Destructor Documentation

7.1.1.1 Geometry::CustomStripifier::CustomStripifier (const Geometry::Mesh *)

Class constructor, receives as a parameter a const pointer to the object that describes a mesh.

7.1.1.2 virtual Geometry::CustomStripifier::~CustomStripifier (void) [virtual]

Class destructor.

7.1.2 Member Function Documentation

7.1.2.1 Mesh* Geometry::CustomStripifier::GetMesh ()

Returns the stripified mesh.

Reimplemented from Geometry::MeshStripifier.

7.1.2.2 void Geometry::CustomStripifier::Stripify () [virtual]

Starts the stripification process. This is a custom stripification method.

Implements Geometry::MeshStripifier.

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

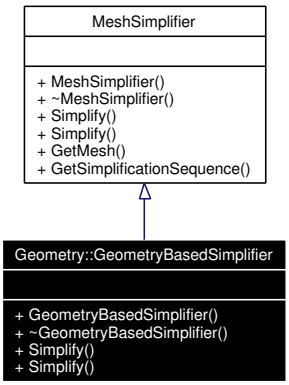
- GeoMeshStripifier.h

7.2 Geometry::GeometryBasedSimplifier Class Reference

Implementation of a simplification algorithm based on geometry information.

`#include <GeoMeshSimplifier.h>`

Inheritance diagram for Geometry::GeometryBasedSimplifier:



Public Member Functions

- GeometryBasedSimplifier (const Geometry::Mesh *)

  Class constructor. Will call Simplifier class constructor.

- virtual ~GeometryBasedSimplifier (void)

  Class destructor.

- void Simplify (Geometry::Real)

  Starts the simplification process. Receives as a parameter the LOD factor in a range of [0,1]. Implements the Simplifier::Simplify method to perform an image based simplification.

- void Simplify (Geometry::uint32)

  Starts the simplification process. Receives as a parameter the number of vertices of the resulting mesh. Implements the Simplifier::Simplify method to perform an image based simplification.

7.2.1 Detailed Description

Implementation of a simplification algorithm based on geometry information.

This class implements a simplification algorithm based on a classic geometry evaluation technique.

Definition at line 79 of file GeoMeshSimplifier.h.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 Geometry::GeometryBasedSimplifier::GeometryBasedSimplifier (const Geometry::Mesh *)

Class constructor. Will call Simplifier class constructor.

7.2.2.2 virtual Geometry::GeometryBasedSimplifier::~GeometryBasedSimplifier (void) [virtual]

Class destructor.

7.2.3 Member Function Documentation

7.2.3.1 void Geometry::GeometryBasedSimplifier::Simplify (Geometry::uint32) [virtual]

Starts the simplification process. Receives as a parameter the number of vertices of the resulting mesh. Implements the Simplifier::Simplify method to perform an image based simplification.

Implements Geometry::MeshSimplifier.

7.2.3.2 void Geometry::GeometryBasedSimplifier::Simplify (Geometry::Real) [virtual]

Starts the simplification process. Receives as a parameter the LOD factor in a range of [0,1]. Implements the Simplifier::Simplify method to perform an image based simplification.

Implements Geometry::MeshSimplifier.

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

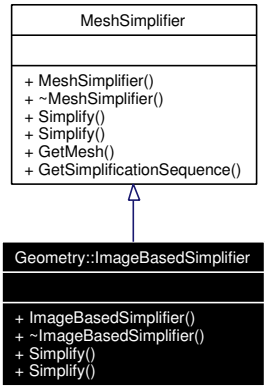
- GeoMeshSimplifier.h

7.3 Geometry::ImageBasedSimplifier Class Reference

Implementation of a simplification algorithm based on images.

`#include <GeoMeshSimplifier.h>`

Inheritance diagram for Geometry::ImageBasedSimplifier:



Public Member Functions

- ImageBasedSimplifier (const Geometry::Mesh *)

  Class constructor. Will call Simplifier class constructor.

- virtual ~ImageBasedSimplifier (void)

  Class destructor.

- void Simplify (Geometry::Real)

  Starts the simplification process. Receives as a parameter the LOD factor in a range of [0,1]. Implements the Simplifier::Simplify method to perform an image based simplification.

- void Simplify (Geometry::uint32)

  Starts the simplification process. Receives as a parameter the number of vertices of the resulting mesh. Implements the Simplifier::Simplify method to perform an image based simplification.

7.3.1 Detailed Description

Implementation of a simplification algorithm based on images.

This class implements a simplification algorithm based on an image evaluation technique.

Definition at line 54 of file GeoMeshSimplifier.h.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 Geometry::ImageBasedSimplifier::ImageBasedSimplifier (const Geometry::Mesh *)

Class constructor. Will call Simplifier class constructor.

7.3.2.2 virtual Geometry::ImageBasedSimplifier::~ImageBasedSimplifier (void) [virtual]

Class destructor.

7.3.3 Member Function Documentation

7.3.3.1 void Geometry::ImageBasedSimplifier::Simplify (Geometry::uint32) [virtual]

Starts the simplification process. Receives as a parameter the number of vertices of the resulting mesh. Implements the Simplifier::Simplify method to perform an image based simplification.

Implements Geometry::MeshSimplifier.

7.3.3.2 void Geometry::ImageBasedSimplifier::Simplify (Geometry::Real) [virtual]

Starts the simplification process. Receives as a parameter the LOD factor in a range of [0,1]. Implements the Simplifier::Simplify method to perform an image based simplification.

Implements Geometry::MeshSimplifier.

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

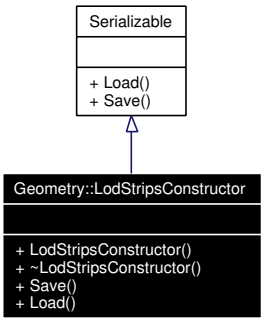
- GeoMeshSimplifier.h

7.4 Geometry::LodStripsConstructor Class Reference

Construction module for general multiresolution models.

`#include <GeoLodStripsConstructor.h>`

Inheritance diagram for Geometry::LodStripsConstructor:



Public Member Functions

- LodStripsConstructor (const Mesh *, const MeshSimplificationSequence *)

  Constructor, gets a stripified mesh and a simplification sequence, and generates the multiresolution model.

- ~LodStripsConstructor (void)

  Destructor.

- void Save (Serializer &s)

  Saves the multiresolution model into a LODStrip file and the LODStrips mesh using the serializer given as a parameter.

- void Load (Serializer &s)

  Load.

7.4.1 Detailed Description

Construction module for general multiresolution models.

This module stores general mesh multiresolution representations using our own specific file format. It takes as inputs the strips provided by the stripification module and the edge collapse sequence computed by the simplification module. It builds a multiresolution representation and generates a LODStrips file.

The LodStrips file begins with a line stating the name of the mesh file it refers to. This filetype has three different kinds of records storing the information required for the multiresolution model. All the records of the same kind are output following the right order.

Those lines starting with a 'd' include:

1. Strip to modify.
2. Number of collapses.

3. Number of vertex repetitions.
4. Number of edge repetitions.

Those beginning with a 'p' include the number of strips affected by a LOD change.

And finally, the ones starting with a 'b' include all the information needed for collapses and repetitions.

Inputs:

- A strip set provided by the stripification module (Geometry::Mesh class).
- An edge collapse sequence computed by the simplification module (Geometry::MeshSimplification-Sequence class).

Outputs:

- The module writes a file with the LODStrips information.
- It also writes the LODStrips mesh into a file.

Definition at line 37 of file GeoLodStripsConstructor.h.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 Geometry::LodStripsConstructor::LodStripsConstructor (const Mesh *, const MeshSimplificationSequence *)

Constructor, gets a stripified mesh and a simplification sequence, and generates the multiresolution model.

7.4.2.2 Geometry::LodStripsConstructor::~LodStripsConstructor (void)

Destructor.

7.4.3 Member Function Documentation

7.4.3.1 void Geometry::LodStripsConstructor::Load (Serializer & *s*) [virtual]

Load.

Implements Geometry::Serializable.

7.4.3.2 void Geometry::LodStripsConstructor::Save (Serializer & *s*) [virtual]

Saves the multiresolution model into a LODStrip file and the LODStrips mesh using the serializer given as a parameter.

Implements Geometry::Serializable.

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

- GeoLodStripsConstructor.h

7.5 Geometry::LodStripsLibrary Class Reference

LodStripsLibrary interface class.

`#include <GeoLodStripsLibrary.h>`

Public Member Functions

- LodStripsLibrary (std::string)

  Constructor, receives as a parameter the name of the file including the multiresolution object.

- ~LodStripsLibrary (void)

  Destructor.

- uint32 MaxLod ()

  Returns the highest LOD.

- uint32 MinLod ()

  Returns the lowest LOD.

- uint32 GoToLod (uint32)

  Returns de current LOD and changes to the specified LOD.

- void TrimByLod (uint32, uint32)
- uint32 MaxFaces ()

  Returns the number of triangles of the highest LOD.

- uint32 MinFaces ()

  Returns the number of triangles of the lowest LOD.

- uint32 MaxVertices ()

  Returns the number of vertices of the highest LOD.

- uint32 MinVertices ()

  Returns the number of vertices of the lowest LOD.

7.5.1 Detailed Description

LodStripsLibrary interface class.

This module contains functions that handle the levels of detail of the input multiresolution objects made of polygonal meshes. For any given resolution and object, this module returns a set of triangle strips representing the object at that resolution, that is, at the level of detail requested. These models use triangle strips to reduce storage usage and to speed up realistic rendering.

Inputs:

- The module receives a file describing a multiresolution object.

Outputs:

- The module returns a strip set that represents the level of detail requested.

Definition at line 19 of file GeoLodStripsLibrary.h.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 Geometry::LodStripsLibrary::LodStripsLibrary (std::string)

Constructor, receives as a parameter the name of the file including the multiresolution object.

7.5.2.2 Geometry::LodStripsLibrary::~LodStripsLibrary (void)

Destructor.

7.5.3 Member Function Documentation

7.5.3.1 uint32 Geometry::LodStripsLibrary::GoToLod (uint32)

Returns de current LOD and changes to the specified LOD.

7.5.3.2 uint32 Geometry::LodStripsLibrary::MaxFaces ()

Returns the number of triangles of the highest LOD.

7.5.3.3 uint32 Geometry::LodStripsLibrary::MaxLod ()

Returns the highest LOD.

7.5.3.4 uint32 Geometry::LodStripsLibrary::MaxVertices ()

Returns the number of vertices of the highest LOD.

7.5.3.5 uint32 Geometry::LodStripsLibrary::MinFaces ()

Returns the number of triangles of the lowest LOD.

7.5.3.6 uint32 Geometry::LodStripsLibrary::MinLod ()

Returns the lowest LOD.

7.5.3.7 uint32 Geometry::LodStripsLibrary::MinVertices ()

Returns the number of vertices of the lowest LOD.

7.5.3.8 void Geometry::LodStripsLibrary::TrimByLod (uint32, uint32)

Establishes the new LOD range. Only the LODs in that range are stored and used.

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

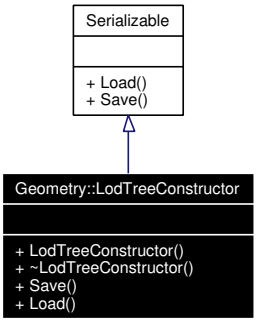
- GeoLodStripsLibrary.h

7.6 Geometry::LodTreeConstructor Class Reference

Construction module for multiresolution tree models.

`#include <GeoLodTreeConstructor.h>`

Inheritance diagram for Geometry::LodTreeConstructor:



Public Member Functions

- LodTreeConstructor (const Mesh *, const TreeSimplificationSequence *)

  Constructor, gets a mesh and a simplification sequence, and generates a multiresolution model.

- ~LodTreeConstructor (void)

  Class Destructor.

- void Save (Serializer &s)

  Assignment operator.

- void Load (Serializer &s)

  Load.

7.6.1 Detailed Description

Construction module for multiresolution tree models.

This module stores general tree multiresolution representations using our own specific file format. It takes as inputs the TreeSimplificationSequence computed by the TreeSimplification module and writes a file with the simplification sequence.

The LodTree file begins with a line stating the name of the mesh with leaves it refers to and also the name of the mesh with the trunk and the name of the LodStrips file refered to this mesh.

File format:

1. Tree reference.
2. Number of collapses.
3. Number of triangles added to the model (new leaves).
4. Triangle data.

5. Collapses: the two leaves that collapses and the new leaf.

Inputs:

- The leaves collapse sequence computed by the simplification module (Geometry::TreeSimplificationSequence).

Outputs:

- The module writes a file with the simplification sequence.

Definition at line 31 of file GeoLodTreeConstructor.h.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 Geometry::LodTreeConstructor::LodTreeConstructor (const Mesh *, const TreeSimplificationSequence *)

Constructor, gets a mesh and a simplification sequence, and generates a multiresolution model.

7.6.2.2 Geometry::LodTreeConstructor::~LodTreeConstructor (void)

Class Destructor.

7.6.3 Member Function Documentation

7.6.3.1 void Geometry::LodTreeConstructor::Load (Serializer & *s*) [virtual]

Load.

Implements Geometry::Serializable.

7.6.3.2 void Geometry::LodTreeConstructor::Save (Serializer & *s*) [virtual]

Assignment operator.

Saves the multiresolution model into a file. It writes the change sequence using the serializer as a parameter.

Implements Geometry::Serializable.

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

- GeoLodTreeConstructor.h

7.7 Geometry::LodTreeLibrary Class Reference

LodTreeLibrary interface class.

`#include <GeoLodTreeLibrary.h>`

Public Member Functions

- LodTreeLibrary (std::string)
  Constructor, receives as a parameter the name of the file including the multiresolution object.
- ~LodTreeLibrary (void)
  Destructor.
- uint32 MaxFoliageLod ()
  Returns the highest LOD of the foliage.
- uint32 MaxTrunkLod ()
  Returns the highest LOD of the trunk.
- uint32 MinFoliageLod ()
  Returns the lowest LOD of the foliage.
- uint32 MinTrunkLod ()
  Returns the lowest LOD of the trunk.
- uint32 GoToFoliageLod (uint32)
  Returns de current foliage LOD and changes to the specified LOD.
- uint32 GoToTrunkLod (uint32)
  Returns de current trunk LOD and changes to the specified LOD.
- void TrimFoliageByLod (uint32, uint32)
- void TrimTrunkByLod (uint32, uint32)
- uint32 MaxFoliageFaces ()
  Returns the number of triangles of the foliage at the highest LOD.
- uint32 MaxTrunkFaces ()
  Returns the number of triangles of the trunk at the highest LOD.
- uint32 MinFoliageFaces ()
  Returns the number of triangles of the foliage at the lowest LOD.
- uint32 MinTrunkFaces ()
  Returns the number of triangles of the trunk at the lowest LOD.
- uint32 MaxFoliageVertices ()
  Returns the number of vertices of the foliage at the highest LOD.
- uint32 MaxTrunkVertices ()

Returns the number of vertices of the trunk at the highest LOD.

- uint32 MinFoliageVertices ()

  Returns the number of vertices of the foliage at the lowest LOD.

- uint32 MinTrunkVertices ()

  Returns the number of vertices of the trunk at the lowest LOD.

7.7.1 Detailed Description

LodTreeLibrary interface class.

This module contains functions that handle the levels of detail of the input multiresolution trees. For Any given resolution and object this module returns two thins: a set of triangle strips representing the trunk and the branches at that resolution, and a triangle list representing the leaves at the same resolution.

Inputs:

- The module receives a file describing a multiresolution tree object.

Outputs:

- The module returns a strip set that represents the level of deatil demanded for the trunk and a triangle list that representes the level of detail for leaves.

Definition at line 22 of file GeoLodTreeLibrary.h.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 Geometry::LodTreeLibrary::LodTreeLibrary (std::string)

Constructor, receives as a parameter the name of the file including the multiresolution object.

7.7.2.2 Geometry::LodTreeLibrary::~LodTreeLibrary (void)

Destructor.

7.7.3 Member Function Documentation

7.7.3.1 uint32 Geometry::LodTreeLibrary::GoToFoliageLod (uint32)

Returns de current foliage LOD and changes to the specified LOD.

7.7.3.2 uint32 Geometry::LodTreeLibrary::GoToTrunkLod (uint32)

Returns de current trunk LOD and changes to the specified LOD.

7.7.3.3 uint32 Geometry::LodTreeLibrary::MaxFoliageFaces ()

Returns the number of triangles of the foliage at the highest LOD.

7.7.3.4 uint32 Geometry::LodTreeLibrary::MaxFoliageLod ()

Returns the highest LOD of the foliage.

7.7.3.5 uint32 Geometry::LodTreeLibrary::MaxFoliageVertices ()

Returns the number of vertices of the foliage at the highest LOD.

7.7.3.6 uint32 Geometry::LodTreeLibrary::MaxTrunkFaces ()

Returns the number of triangles of the trunk at the highest LOD.

7.7.3.7 uint32 Geometry::LodTreeLibrary::MaxTrunkLod ()

Returns the highest LOD of the trunk.

7.7.3.8 uint32 Geometry::LodTreeLibrary::MaxTrunkVertices ()

Returns the number of vertices of the trunk at the highest LOD.

7.7.3.9 uint32 Geometry::LodTreeLibrary::MinFoliageFaces ()

Returns the number of triangles of the foliage at the lowest LOD.

7.7.3.10 uint32 Geometry::LodTreeLibrary::MinFoliageLod ()

Returns the lowest LOD of the foliage.

7.7.3.11 uint32 Geometry::LodTreeLibrary::MinFoliageVertices ()

Returns the number of vertices of the foliage at the lowest LOD.

7.7.3.12 uint32 Geometry::LodTreeLibrary::MinTrunkFaces ()

Returns the number of triangles of the trunk at the lowest LOD.

7.7.3.13 uint32 Geometry::LodTreeLibrary::MinTrunkLod ()

Returns the lowest LOD of the trunk.

7.7.3.14 uint32 Geometry::LodTreeLibrary::MinTrunkVertices ()

Returns the number of vertices of the trunk at the lowest LOD.

7.7.3.15 void Geometry::LodTreeLibrary::TrimFoliageByLod (uint32, uint32)

Establishes the new LOD range. Only the LODs in that range are stored and used.

7.7.3.16 void Geometry::LodTreeLibrary::TrimTrunkByLod (uint32, uint32)

Establishes the new LOD range. Only the LODs in that range are stored and used.

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

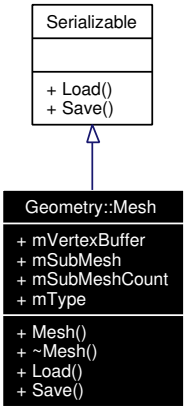
- GeoLodTreeLibrary.h

7.8 Geometry::Mesh Class Reference

Mesh class interface.

`#include <GeoMesh.h>`

Inheritance diagram for Geometry::Mesh:



Public Member Functions

- Mesh ()
  Constructor.
- ~Mesh ()
  Destructor.
- void Load (Serializer &s)
  Loads data from a Serializer.
- void Save (Serializer &s)
  Saves data to a Serializer.

Public Attributes

- VertexBuffer * mVertexBuffer
  Shared VertexBuffer.
- SubMesh * mSubMesh
  Array of subMesehs.
- size_t mSubMeshCount
  Total count of subMeshes.
- MeshType mType
  Type of mesh.

7.8.1 Detailed Description

Mesh class interface.

Definition at line 15 of file GeoMesh.h.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 Geometry::Mesh::Mesh ()

Constructor.

7.8.2.2 Geometry::Mesh::~Mesh ()

Destructor.

7.8.3 Member Function Documentation

7.8.3.1 void Geometry::Mesh::Load (Serializer & *s*) [virtual]

Loads data from a Serializer.

Implements Geometry::Serializable.

7.8.3.2 void Geometry::Mesh::Save (Serializer & *s*) [virtual]

Saves data to a Serializer.

Implements Geometry::Serializable.

7.8.4 Member Data Documentation

7.8.4.1 SubMesh* Geometry::Mesh::mSubMesh

Array of subMesehs.

Definition at line 37 of file GeoMesh.h.

7.8.4.2 size_t Geometry::Mesh::mSubMeshCount

Total count of subMeshes.

Definition at line 38 of file GeoMesh.h.

7.8.4.3 MeshType Geometry::Mesh::mType

Type of mesh.

Definition at line 39 of file GeoMesh.h.

7.8.4.4 VertexBuffer* Geometry::Mesh::mVertexBuffer

Shared VertexBuffer.

Definition at line 36 of file GeoMesh.h.

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

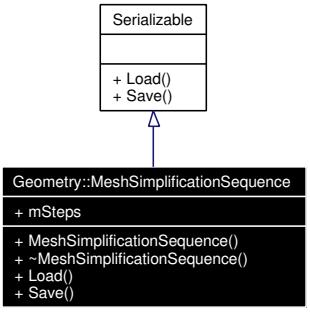
- GeoMesh.h

7.9 Geometry::MeshSimplificationSequence Class Reference

Represents the simplification sequence applied to a given mesh.

`#include <GeoMeshSimpSequence.h>`

Inheritance diagram for Geometry::MeshSimplificationSequence:



Public Member Functions

- MeshSimplificationSequence (void)

  Class constructor.

- ~MeshSimplificationSequence (void)

  Class destructor.

- void Load (Serializer &s)

  Loads a simplification sequence from a Serializer.

- void Save (Serializer &s)

  Saves the contents of the data structures.

Public Attributes

- std::vector< Step > mSteps

  Stores all the simplification steps.

7.9.1 Detailed Description

Represents the simplification sequence applied to a given mesh.

This class stores information about the simplification process, giving for each step the vertex that collapses, the vertex that remains unchanged, the two triangles that disappear and the new triangles that come out. It also offers a method to generate a mesh simplification sequence format file.

This file begins with a line stating the name of the mesh file it refers to.

Every line gives the following information:

1. Vertex that collapses.
2. Vertex that remains unchanged.
3. The two triangles that disappear.
4. & as a separator.
5. The list of new triangles.

Definition at line 29 of file GeoMeshSimpSequence.h.

7.9.2 Constructor & Destructor Documentation

7.9.2.1 Geometry::MeshSimplificationSequence::MeshSimplificationSequence (void)

Class constructor.

7.9.2.2 Geometry::MeshSimplificationSequence::~MeshSimplificationSequence (void)

Class destructor.

7.9.3 Member Function Documentation

7.9.3.1 void Geometry::MeshSimplificationSequence::Load (Serializer & *s*) [virtual]

Loads a simplification sequence from a Serializer.

Implements Geometry::Serializable.

7.9.3.2 void Geometry::MeshSimplificationSequence::Save (Serializer & *s*) [virtual]

Saves the contents of the data structures.

Implements Geometry::Serializable.

7.9.4 Member Data Documentation

7.9.4.1 std::vector<Step> Geometry::MeshSimplificationSequence::mSteps

Stores all the simplification steps.

Definition at line 53 of file GeoMeshSimpSequence.h.

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

- GeoMeshSimpSequence.h

7.10 Geometry::MeshSimplificationSequence::Step Struct Reference

Represents a simplification step in the sequence.

`#include <GeoMeshSimpSequence.h>`

Public Attributes

- Index mV0
- Index mV1
- Index mT0
- Index mT1
- std::vector< Index > mModfaces

7.10.1 Detailed Description

Represents a simplification step in the sequence.

Definition at line 45 of file GeoMeshSimpSequence.h.

7.10.2 Member Data Documentation

7.10.2.1 std::vector<Index> Geometry::MeshSimplificationSequence::Step::mModfaces

Definition at line 49 of file GeoMeshSimpSequence.h.

7.10.2.2 Index Geometry::MeshSimplificationSequence::Step::mT0

Definition at line 48 of file GeoMeshSimpSequence.h.

7.10.2.3 Index Geometry::MeshSimplificationSequence::Step::mT1

Definition at line 48 of file GeoMeshSimpSequence.h.

7.10.2.4 Index Geometry::MeshSimplificationSequence::Step::mV0

Definition at line 47 of file GeoMeshSimpSequence.h.

7.10.2.5 Index Geometry::MeshSimplificationSequence::Step::mV1

Definition at line 47 of file GeoMeshSimpSequence.h.

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

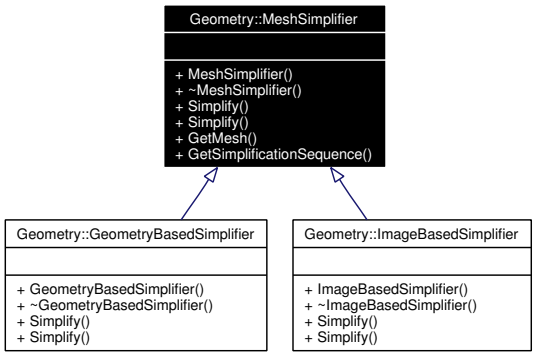
- GeoMeshSimpSequence.h

7.11 Geometry::MeshSimplifier Class Reference

Mesh simplificator interface.

`#include <GeoMeshSimplifier.h>`

Inheritance diagram for Geometry::MeshSimplifier:



Public Member Functions

- MeshSimplifier (const Geometry::Mesh *)

  Class constructor. Retrieves a pointer to a valid Geometry::Mesh object to simplify.

- virtual ~MeshSimplifier (void)

  Virtual class destructor.

- virtual void Simplify (Geometry::Real)=0

  Starts the simplification process. Receives as a parameter the LOD factor in a range of [0,1]. This is a pure virtual method and must be overloaded in a derived class that implements a simplification algorithm.

- virtual void Simplify (Geometry::uint32)=0

  Starts the simplification process. Receives as a parameter the number of vertices of the resulting mesh. This is a pure virtual method and must be overloaded in a derived class that implements a simplification algorithm.

- Mesh * GetMesh ()

  Returns the simplified mesh.

- MeshSimplificationSequence * GetSimplificationSequence ()

  Returns the simplification sequence for general meshes.

7.11.1 Detailed Description

Mesh simplificator interface.

This module is used by both models, general mesh models and the plant and tree models for the trunk and the branches. It contains functions that generate simplified versions of 3D objects made out of triangles. Given a 3D object, this module computes a sequence of geometric transformations that reduce the

object146s geometric detail while preserving its appearance. For each simplification step, returns a simplification sequence containing the edge to be collapse, the two triangles being removed and the new triangles remapped to the model.

Inputs:

- A pointer to the Geometry::Mesh object containing the 3D model to be simplified.

Outputs:

1. The simplified mesh, contained in a Geometry::Mesh object.
2. Simplification sequence, represented by a Geometry::MeshSimplificationSequence object.

Definition at line 23 of file GeoMeshSimplifier.h.

7.11.2 Constructor & Destructor Documentation

7.11.2.1 Geometry::MeshSimplifier::MeshSimplifier (const Geometry::Mesh *)

Class constructor. Retrieves a pointer to a valid Geometry::Mesh object to simplify.

7.11.2.2 virtual Geometry::MeshSimplifier::~MeshSimplifier (void) [virtual]

Virtual class destructor.

7.11.3 Member Function Documentation

7.11.3.1 Mesh* Geometry::MeshSimplifier::GetMesh ()

Returns the simplified mesh.

7.11.3.2 MeshSimplificationSequence* Geometry::MeshSimplifier::GetSimplificationSequence ()

Returns the simplification sequence for general meshes.

7.11.3.3 virtual void Geometry::MeshSimplifier::Simplify (Geometry::uint32) [pure virtual]

Starts the simplification process. Receives as a parameter the number of vertices of the resulting mesh. This is a pure virtual method and must be overloaded in a derived class that implements a simplification algorithm.

Implemented in Geometry::ImageBasedSimplifier, and Geometry::GeometryBasedSimplifier.

7.11.3.4 virtual void Geometry::MeshSimplifier::Simplify (Geometry::Real) [pure virtual]

Starts the simplification process. Receives as a parameter the LOD factor in a range of [0,1]. This is a pure virtual method and must be overloaded in a derived class that implements a simplification algorithm.

Implemented in Geometry::ImageBasedSimplifier, and Geometry::GeometryBasedSimplifier.

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

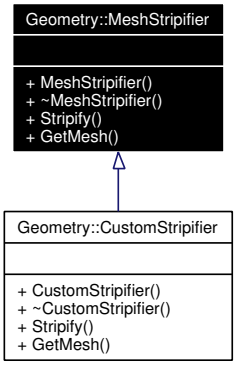
- GeoMeshSimplifier.h

7.12 Geometry::MeshStripifier Class Reference

Stripifier interface class.

`#include <GeoMeshStripifier.h>`

Inheritance diagram for Geometry::MeshStripifier:



Public Member Functions

- MeshStripifier (const Geometry::Mesh *)

  Class constructor, receives as a parameter a const pointer to the object that describes a mesh.

- virtual ~MeshStripifier (void)

  virtual class destructor.

- virtual void Stripify ()=0

  Starts the stripification process. This is a pure virtual method and must be overloaded in a derived class that implements a stripification algorithm.

- Mesh * GetMesh ()

  Returns the stripified mesh.

7.12.1 Detailed Description

Stripifier interface class.

This module implements methods that extract triangle strips from triangle meshes.

Inputs:

- This module receives a pointer to a Geometry::Mesh object containing the model to be stripified.

Outputs:

- The stripified mesh, contained also in a Geometry::Mesh object.

Definition at line 26 of file GeoMeshStripifier.h.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 Geometry::MeshStripifier::MeshStripifier (const Geometry::Mesh *)

Class constructor, receives as a parameter a const pointer to the object that describes a mesh.

7.12.2.2 virtual Geometry::MeshStripifier::~MeshStripifier (void) [virtual]

virtual class destructor.

7.12.3 Member Function Documentation

7.12.3.1 Mesh* Geometry::MeshStripifier::GetMesh ()

Returns the stripified mesh.

Reimplemented in Geometry::CustomStripifier.

7.12.3.2 virtual void Geometry::MeshStripifier::Stripify () [pure virtual]

Starts the stripification process. This is a pure virtual method and must be overloaded in a derived class that implements a stripification algorithm.

Implemented in Geometry::CustomStripifier.

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

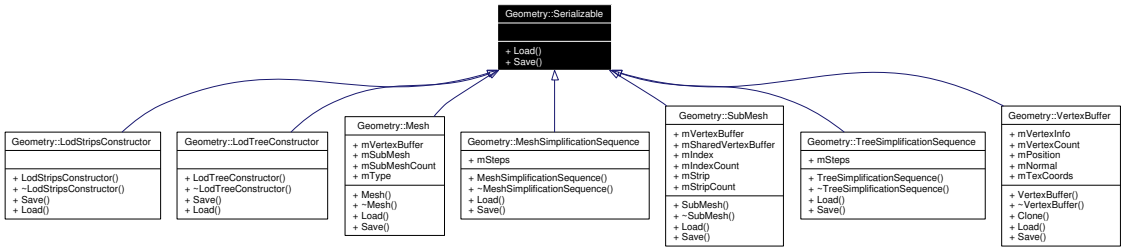
- GeoMeshStripifier.h

7.13 Geometry::Serializable Class Reference

Serializable interface.

`#include <GeoSerializable.h>`

Inheritance diagram for Geometry::Serializable:



Public Member Functions

- virtual void Load (Serializer &s)=0

  Loads data from a Serializer.

- virtual void Save (Serializer &s)=0

  Saves data to a Serializer.

7.13.1 Detailed Description

Serializable interface.

Base class of objects to simplify input/output.

Definition at line 10 of file GeoSerializable.h.

7.13.2 Member Function Documentation

7.13.2.1 virtual void Geometry::Serializable::Load (Serializer & *s*) [pure virtual]

Loads data from a Serializer.

Implemented in Geometry::LodStripsConstructor, Geometry::LodTreeConstructor, Geometry::Mesh, Geometry::MeshSimplificationSequence, Geometry::SubMesh, Geometry::TreeSimplificationSequence, and Geometry::VertexBuffer.

7.13.2.2 virtual void Geometry::Serializable::Save (Serializer & *s*) [pure virtual]

Saves data to a Serializer.

Implemented in Geometry::LodStripsConstructor, Geometry::LodTreeConstructor, Geometry::Mesh, Geometry::MeshSimplificationSequence, Geometry::SubMesh, Geometry::TreeSimplificationSequence, and Geometry::VertexBuffer.

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

- GeoSerializable.h

7.14 Geometry::Serializer Class Reference

Serializer interface.

`#include <GeoSerializer.h>`

Public Types

- enum Mode { READ, WRITE, APPEND }

Public Member Functions

- Serializer (String name, Mode mode)

  Constructor.

- virtual ~Serializer ()

  Destructor.

- void WriteData (const void *const buf, size_t size, size_t count)

  Assignment operator.

- void WriteArray (const float *const pfloat, size_t count)
- void WriteArray (const uint16 *const pShort, size_t count)
- void WriteArray (const uint32 *const pInt, size_t count)
- void WriteArray (const bool *const pLong, size_t count)
- void WriteArray (const Vector3 *const pvector3, size_t count)
- void WriteArray (const Vector2 *const pvector3, size_t count)
- void ReadData (void *buf, size_t size, size_t count)
- void ReadArray (bool *pDest, size_t count)
- void ReadArray (float *pDest, size_t count)
- void ReadArray (uint16 *pDest, size_t count)
- void ReadArray (uint32 *pDest, size_t count)
- void ReadArray (Vector3 *pDest, size_t count)
- void ReadArray (Vector2 *pDest, size_t count)
- virtual size_t GetSize ()

Protected Member Functions

- void FlipToLittleEndian (void *pData, size_t size, size_t count=1)
- void FlipFromLittleEndian (void *pData, size_t size, size_t count=1)
- void FlipEndian (void *pData, size_t size, size_t count)
- void FlipEndian (void *pData, size_t size)

Protected Attributes

- size_t mSize
- FILE * mFile
- Mode mMode

7.14.1 Detailed Description

Serializer interface.

Serializer is a helper class than manages file input/output.

Definition at line 13 of file GeoSerializer.h.

7.14.2 Member Enumeration Documentation

7.14.2.1 enum Geometry::Serializer::Mode

Enumeration values:

READ

WRITE

APPEND

Definition at line 17 of file GeoSerializer.h.

7.14.3 Constructor & Destructor Documentation

7.14.3.1 Geometry::Serializer::Serializer (String *name*, Mode *mode*)

Constructor.

7.14.3.2 virtual Geometry::Serializer::~Serializer () [virtual]

Destructor.

7.14.4 Member Function Documentation

7.14.4.1 void Geometry::Serializer::FlipEndian (void * *pData*, size_t *size*) [protected]

7.14.4.2 void Geometry::Serializer::FlipEndian (void * *pData*, size_t *size*, size_t *count*) [protected]

7.14.4.3 void Geometry::Serializer::FlipFromLittleEndian (void * *pData*, size_t *size*, size_t *count* = 1) [protected]

7.14.4.4 void Geometry::Serializer::FlipToLittleEndian (void * *pData*, size_t *size*, size_t *count* = 1) [protected]

7.14.4.5 virtual size_t Geometry::Serializer::GetSize () [inline, virtual]

Definition at line 54 of file GeoSerializer.h.

References mSize.

7.14.4.6 void Geometry::Serializer::ReadArray (Vector2 * *pDest*, size_t *count*)

7.14.4.7 void Geometry::Serializer::ReadArray (Vector3 * *pDest*, size_t *count*)

7.14.4.8 void Geometry::Serializer::ReadArray (uint32 * *pDest*, size_t *count*)

7.14.4.9 void Geometry::Serializer::ReadArray (uint16 * *pDest*, size_t *count*)

7.14.4.10 void Geometry::Serializer::ReadArray (float * *pDest*, size_t *count*)

7.14.4.11 void Geometry::Serializer::ReadArray (bool * *pDest*, size_t *count*)

7.14.4.12 void Geometry::Serializer::ReadData (void * *buf*, size_t *size*, size_t *count*)

7.14.4.13 void Geometry::Serializer::WriteArray (const Vector2 *const *pvector3*, size_t *count*)

7.14.4.14 void Geometry::Serializer::WriteArray (const Vector3 *const *pvector3*, size_t *count*)

7.14.4.15 void Geometry::Serializer::WriteArray (const bool *const *pLong*, size_t *count*)

7.14.4.16 void Geometry::Serializer::WriteArray (const uint32 *const *pInt*, size_t *count*)

7.14.4.17 void Geometry::Serializer::WriteArray (const uint16 *const *pShort*, size_t *count*)

7.14.4.18 void Geometry::Serializer::WriteArray (const float *const *pfloat*, size_t *count*)

7.14.4.19 void Geometry::Serializer::WriteData (const void *const *buf*, size_t *size*, size_t *count*)

Assignment operator.

7.14.5 Member Data Documentation

7.14.5.1 FILE* Geometry::Serializer::mFile [protected]

Definition at line 62 of file GeoSerializer.h.

7.14.5.2 Mode Geometry::Serializer::mMode [protected]

Definition at line 63 of file GeoSerializer.h.

7.14.5.3 size_t Geometry::Serializer::mSize [protected]

Definition at line 61 of file GeoSerializer.h.

Referenced by GetSize().

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

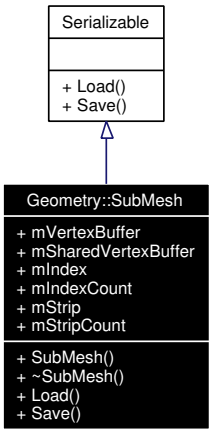
- GeoSerializer.h

7.15 Geometry::SubMesh Class Reference

SubMesh interface.

`#include <GeoSubMesh.h>`

Inheritance diagram for Geometry::SubMesh:



Public Member Functions

- SubMesh ()

  Default constructor.

- ~SubMesh ()

  Default destructor.

- void Load (Serializer &s)

  Loads data from a Serializer.

- void Save (Serializer &s)

  Saves data to a Serializer.

Public Attributes

- VertexBuffer * mVertexBuffer
- bool mSharedVertexBuffer

  true if VertexBuffer it's shared with Mesh and other SubMeshes

- Index * mIndex

  Array of Index.

- size_t mIndexCount

  Index count.

- Index ** mStrip

Array of pointers to mIndex that represents each strip.

- size_t mStripCount
  number of Strips

7.15.1 Detailed Description

SubMesh interface.

SubMesh is part of a Mesh, and stores vertex and index geometric information.

Definition at line 15 of file GeoSubMesh.h.

7.15.2 Constructor & Destructor Documentation

7.15.2.1 Geometry::SubMesh::SubMesh ()

Default constructor.

7.15.2.2 Geometry::SubMesh::~SubMesh ()

Default destructor.

7.15.3 Member Function Documentation

7.15.3.1 void Geometry::SubMesh::Load (Serializer & *s*) [virtual]

Loads data from a Serializer.

Implements Geometry::Serializable.

7.15.3.2 void Geometry::SubMesh::Save (Serializer & *s*) [virtual]

Saves data to a Serializer.

Implements Geometry::Serializable.

7.15.4 Member Data Documentation

7.15.4.1 Index* Geometry::SubMesh::mIndex

Array of Index.

Definition at line 42 of file GeoSubMesh.h.

7.15.4.2 size_t Geometry::SubMesh::mIndexCount

Index count.

Definition at line 43 of file GeoSubMesh.h.

7.15.4.3 bool Geometry::SubMesh::mSharedVertexBuffer

true if VertexBuffer it's shared with Mesh and other SubMeshes

Definition at line 40 of file GeoSubMesh.h.

7.15.4.4 Index** Geometry::SubMesh::mStrip

Array of pointers to mIndex that represents each strip.

Definition at line 45 of file GeoSubMesh.h.

7.15.4.5 size_t Geometry::SubMesh::mStripCount

number of Strips

Definition at line 46 of file GeoSubMesh.h.

7.15.4.6 VertexBuffer* Geometry::SubMesh::mVertexBuffer

VertexBuffer used to store vertex Data. Is a reference to a shared VertexData if mSharedVertexBuffer == true, and must be not deallocated in that case.

Definition at line 38 of file GeoSubMesh.h.

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

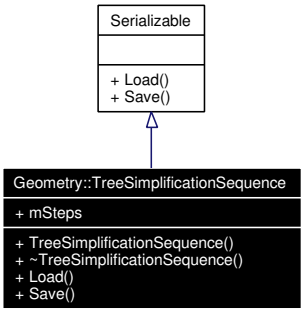
- GeoSubMesh.h

7.16 Geometry::TreeSimplificationSequence Class Reference

Represents the simplification sequence applied to a given mesh.

`#include <GeoTreeSimpSequence.h>`

Inheritance diagram for Geometry::TreeSimplificationSequence:



Public Member Functions

- TreeSimplificationSequence (void)
  Class constructor.
- ~TreeSimplificationSequence (void)
  Class destructor.
- void Load (Serializer &s)
  Loads a Simplification Sequence from a Serializer.
- void Save (Serializer &s)
  Saves the contents of the data structures.

Public Attributes

- std::vector< Step > mSteps
  Stores all the simplification steps.

7.16.1 Detailed Description

Represents the simplification sequence applied to a given mesh.

This class stores information about the simplification process, giving for each step the four triangles that compose the two leaves that will be collapsed and the two triangles of the new leaf. It also offers a method to generate a tree simplification sequence format file.

This file begins with a line stating the name of the mesh file it refers to.

This file stores the leaves geometry and at the end of the file one line for each leaf collapse operation.

begin

v / vertex sequence v

v 133

f // faces sequence (two faces compose one leaf) including new faces

f // added due to the simplification process

f 133

end

one collapse record per line

Those records gives the following information:

1. The four triangles that compose the two leaves that will be collapsed.
2. & as a separator.
3. The two triangles for the new leaf.

Definition at line 38 of file GeoTreeSimpSequence.h.

7.16.2 Constructor & Destructor Documentation

7.16.2.1 Geometry::TreeSimplificationSequence::TreeSimplificationSequence (void)

Class constructor.

7.16.2.2 Geometry::TreeSimplificationSequence::~TreeSimplificationSequence (void)

Class destructor.

7.16.3 Member Function Documentation

7.16.3.1 void Geometry::TreeSimplificationSequence::Load (Serializer & *s*) [virtual]

Loads a Simplification Sequence from a Serializer.

Implements Geometry::Serializable.

7.16.3.2 void Geometry::TreeSimplificationSequence::Save (Serializer & *s*) [virtual]

Saves the contents of the data structures.

Implements Geometry::Serializable.

7.16.4 Member Data Documentation

7.16.4.1 std::vector<Step> Geometry::TreeSimplificationSequence::mSteps

Stores all the simplification steps.

Definition at line 62 of file GeoTreeSimpSequence.h.

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

- GeoTreeSimpSequence.h

7.17 Geometry::TreeSimplificationSequence::Step Struct Reference

Represents a Simplification Step in the sequence.

`#include <GeoTreeSimpSequence.h>`

Public Attributes

- Index mV0
- Index mV1
- Index mT0
- Index mT1
- Index mNewQuad [4]

7.17.1 Detailed Description

Represents a Simplification Step in the sequence.

Definition at line 54 of file GeoTreeSimpSequence.h.

7.17.2 Member Data Documentation

7.17.2.1 Index Geometry::TreeSimplificationSequence::Step::mNewQuad[4]

Definition at line 58 of file GeoTreeSimpSequence.h.

7.17.2.2 Index Geometry::TreeSimplificationSequence::Step::mT0

Definition at line 57 of file GeoTreeSimpSequence.h.

7.17.2.3 Index Geometry::TreeSimplificationSequence::Step::mT1

Definition at line 57 of file GeoTreeSimpSequence.h.

7.17.2.4 Index Geometry::TreeSimplificationSequence::Step::mV0

Definition at line 56 of file GeoTreeSimpSequence.h.

7.17.2.5 Index Geometry::TreeSimplificationSequence::Step::mV1

Definition at line 56 of file GeoTreeSimpSequence.h.

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

- GeoTreeSimpSequence.h

7.18 Geometry::TreeSimplifier Class Reference

Tree Simplifier interface.

`#include <GeoTreeSimplifier.h>`

Public Member Functions

- TreeSimplifier (const Geometry::Mesh *)
  Class constructor. Retrieves a pointer to a valid Mesh object to simplify.
- ~TreeSimplifier (void)
  Class destructor.
- void Simplify (Geometry::Real)
  Starts the simplification process. Receives as a parameter the LOD factor in a range of [0,1].
- Mesh * GetMesh ()
  Returns the simplified mesh.
- TreeSimplificationSequence * GetSimplificationSequence ()
  Returns the simplification sequence for leaves.

7.18.1 Detailed Description

Tree Simplifier interface.

Definition at line 29 of file GeoTreeSimplifier.h.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 Geometry::TreeSimplifier::TreeSimplifier (const Geometry::Mesh *)

Class constructor. Retrieves a pointer to a valid Mesh object to simplify.

7.18.2.2 Geometry::TreeSimplifier::~TreeSimplifier (void)

Class destructor.

7.18.3 Member Function Documentation

7.18.3.1 Mesh* Geometry::TreeSimplifier::GetMesh ()

Returns the simplified mesh.

7.18.3.2 TreeSimplificationSequence* Geometry::TreeSimplifier::GetSimplificationSequence ()

Returns the simplification sequence for leaves.

7.18.3.3 void Geometry::TreeSimplifier::Simplify (Geometry::Real)

Starts the simplification process. Receives as a parameter the LOD factor in a range of [0,1].

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

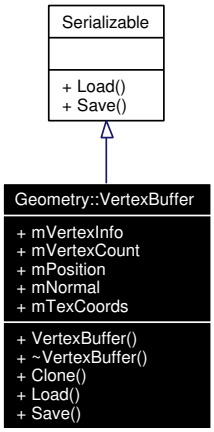
- GeoTreeSimplifier.h

7.19 Geometry::VertexBuffer Class Reference

VertexBuffer interface Class.

`#include <GeoVertexBuffer.h>`

Inheritance diagram for Geometry::VertexBuffer:



Public Member Functions

- VertexBuffer ()

  Default Constructor.

- ~VertexBuffer ()

  Default destructor, releases allocated memory.

- VertexBuffer * Clone () const

  Returns a new VertexBuffer with the same data.

- void Load (Serializer &s)

  Fills this VertexBuffer from a Serializer.

- void Save (Serializer &s)

  Stores data.

Public Attributes

- unsigned int mVertexInfo

  Type of info stored by vertex.

- size_t mVertexCount

  Number of vertices.

- Vector3 * mPosition

  Position array of each Vertex, only valid if (vertexInfo & VERTEX_POSITON) == true.

- Vector3 * mNormal

  Normal array of each Vertex, only valid if (vertexInfo & VERTEX_NORMAL) == true.

- Vector2 * mTexCoords

  Texture Coordinates array of each Vertex, only valid if (vertexInfo & VERTEX_TEXCOORDS) == true.

7.19.1 Detailed Description

VertexBuffer interface Class.

This Structure holds the vertex information used by Meshes and SubMesehs.

Definition at line 16 of file GeoVertexBuffer.h.

7.19.2 Constructor & Destructor Documentation

7.19.2.1 Geometry::VertexBuffer::VertexBuffer () [inline]

Default Constructor.

Definition at line 21 of file GeoVertexBuffer.h.

References mNormal, mPosition, mTexCoords, mVertexCount, and mVertexInfo.

7.19.2.2 Geometry::VertexBuffer::~VertexBuffer ()

Default destructor, releases allocated memory.

7.19.3 Member Function Documentation

7.19.3.1 VertexBuffer* Geometry::VertexBuffer::Clone () const

Returns a new VertexBuffer with the same data.

7.19.3.2 void Geometry::VertexBuffer::Load (Serializer & *s*) [virtual]

Fills this VertexBuffer from a Serializer.

Implements Geometry::Serializable.

7.19.3.3 void Geometry::VertexBuffer::Save (Serializer & *s*) [virtual]

Stores data.

Implements Geometry::Serializable.

7.19.4 Member Data Documentation

7.19.4.1 Vector3* Geometry::VertexBuffer::mNormal

Normal array of each Vertex, only valid if (vertexInfo & VERTEX_NORMAL) == true.

Definition at line 45 of file GeoVertexBuffer.h.

Referenced by VertexBuffer().

7.19.4.2 Vector3* Geometry::VertexBuffer::mPosition

Position array of each Vertex, only valid if (vertexInfo & VERTEX_POSITON) == true.

Definition at line 44 of file GeoVertexBuffer.h.

Referenced by VertexBuffer().

7.19.4.3 Vector2* Geometry::VertexBuffer::mTexCoords

Texture Coordinates array of each Vertex, only valid if (vertexInfo & VERTEX_TEXCOORDS) == true.

Definition at line 46 of file GeoVertexBuffer.h.

Referenced by VertexBuffer().

7.19.4.4 size_t Geometry::VertexBuffer::mVertexCount

Number of vertices.

Definition at line 43 of file GeoVertexBuffer.h.

Referenced by VertexBuffer().

7.19.4.5 unsigned int Geometry::VertexBuffer::mVertexInfo

Type of info stored by vertex.

Definition at line 42 of file GeoVertexBuffer.h.

Referenced by VertexBuffer().

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

- GeoVertexBuffer.h

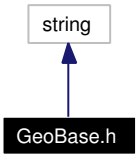
Chapter 8

GameTools Geometry Modules File Documentation

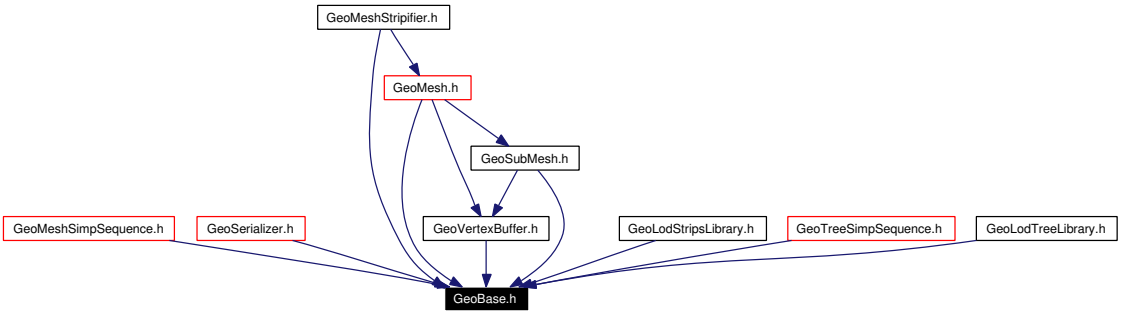
8.1 GeoBase.h File Reference

`#include <string>`

Include dependency graph for GeoBase.h:



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



Namespaces

- namespace Geometry

Defines

- #define GEO_ENDIAN_LITTLE 1

- #define GEO_ENDIAN_BIG 2
- #define GEO_ENDIAN GEO_ENDIAN_LITTLE

8.1.1 Define Documentation

8.1.1.1 #define GEO_ENDIAN GEO_ENDIAN_LITTLE

Definition at line 47 of file GeoBase.h.

8.1.1.2 #define GEO_ENDIAN_BIG 2

Definition at line 41 of file GeoBase.h.

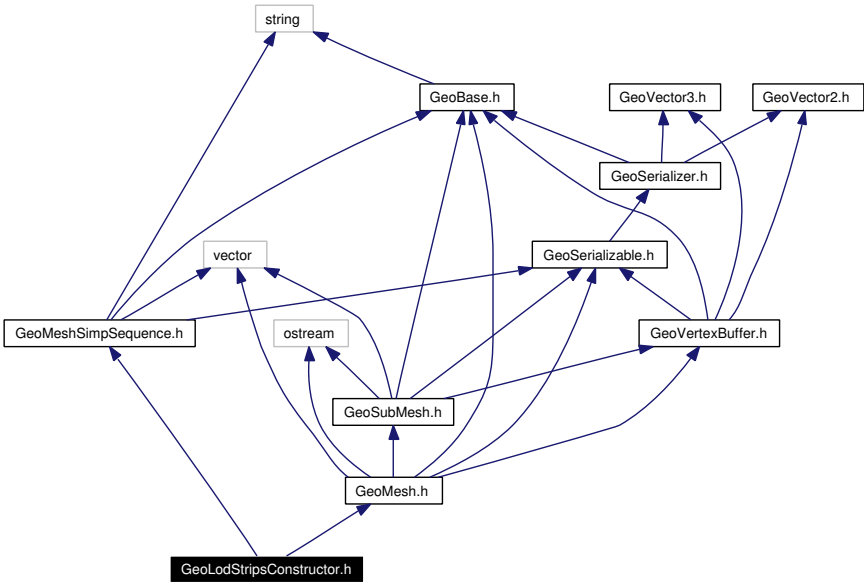
8.1.1.3 #define GEO_ENDIAN_LITTLE 1

Definition at line 40 of file GeoBase.h.

8.2 GeoLodStripsConstructor.h File Reference

```
#include "GeoMeshSimpSequence.h"
#include "GeoMesh.h"
```

Include dependency graph for GeoLodStripsConstructor.h:



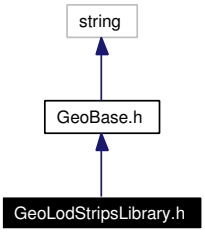
Namespaces

- namespace Geometry

8.3 GeoLodStripsLibrary.h File Reference

```
#include "GeoBase.h"
```

Include dependency graph for GeoLodStripsLibrary.h:



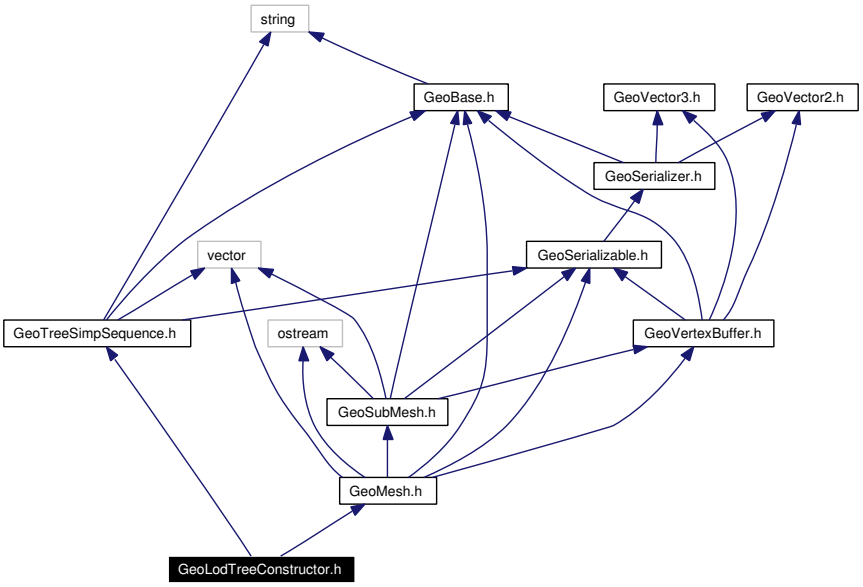
Namespaces

- namespace Geometry

8.4 GeoLodTreeConstructor.h File Reference

```
#include "GeoTreeSimpSequence.h"
#include "GeoMesh.h"
```

Include dependency graph for GeoLodTreeConstructor.h:



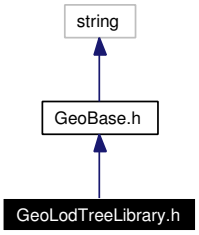
Namespaces

- namespace Geometry

8.5 GeoLodTreeLibrary.h File Reference

```
#include "GeoBase.h"
```

Include dependency graph for GeoLodTreeLibrary.h:



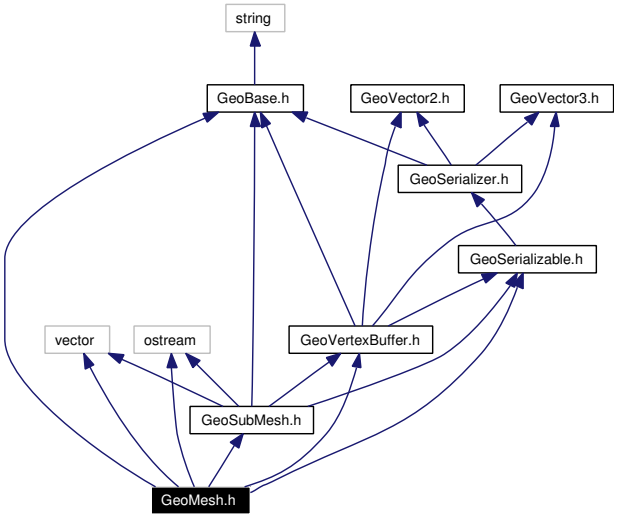
Namespaces

- namespace Geometry

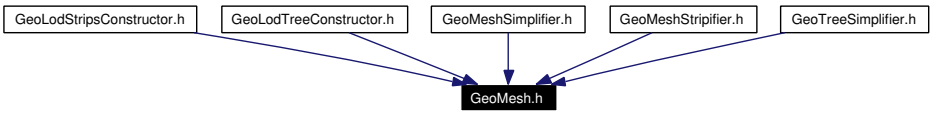
8.6 GeoMesh.h File Reference

```
#include <vector>
#include <ostream>
#include "GeoBase.h"
#include "GeoVertexBuffer.h"
#include "GeoSerializable.h"
#include "GeoSubMesh.h"
```

Include dependency graph for GeoMesh.h:



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



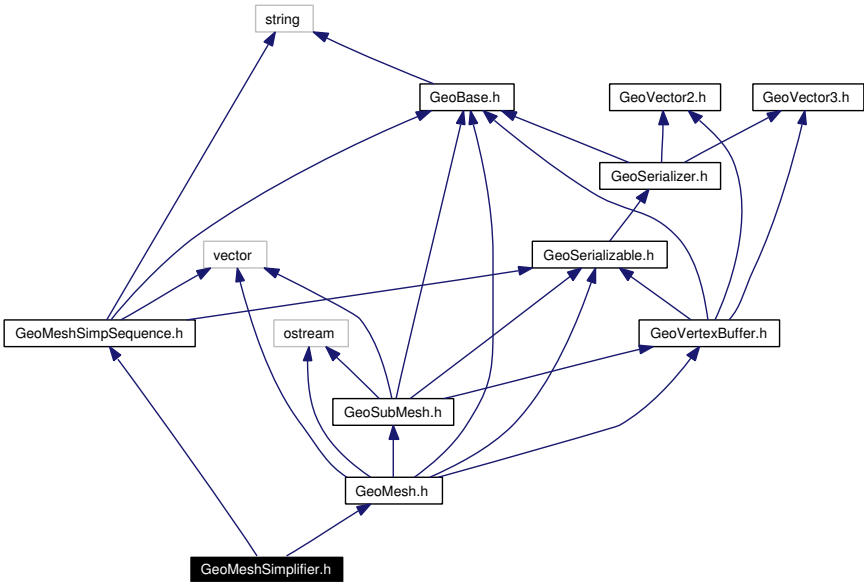
Namespaces

- namespace Geometry

8.7 GeoMeshSimplifier.h File Reference

```
#include "GeoMesh.h"
#include "GeoMeshSimpSequence.h"
```

Include dependency graph for GeoMeshSimplifier.h:



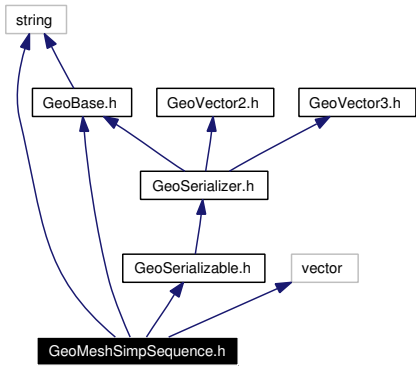
Namespaces

- namespace Geometry

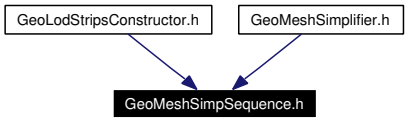
8.8 GeoMeshSimpSequence.h File Reference

```
#include <string>
#include <vector>
#include "GeoBase.h"
#include "GeoSerializable.h"
```

Include dependency graph for GeoMeshSimpSequence.h:



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



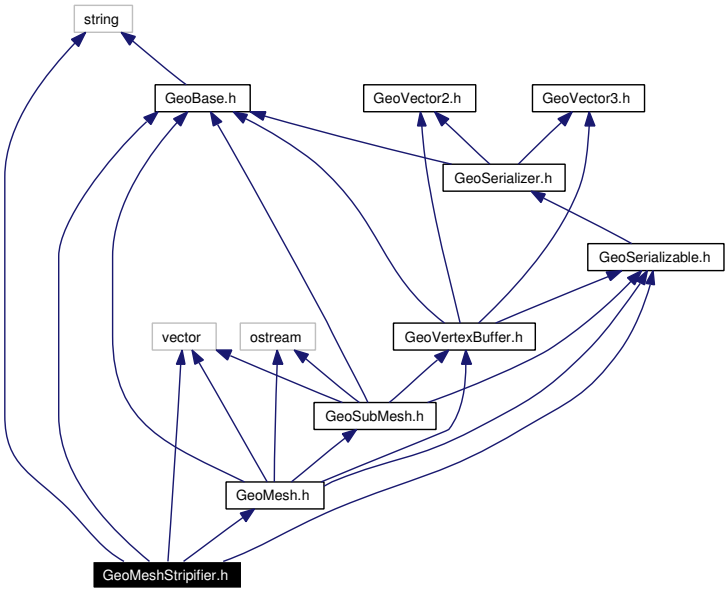
Namespaces

- namespace Geometry

8.9 GeoMeshStripifier.h File Reference

```
#include <string>
#include <vector>
#include "GeoBase.h"
#include "GeoSerializable.h"
#include "GeoMesh.h"
```

Include dependency graph for GeoMeshStripifier.h:



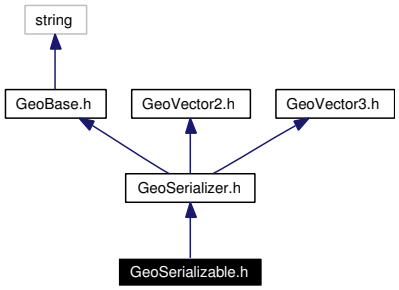
Namespaces

- namespace Geometry

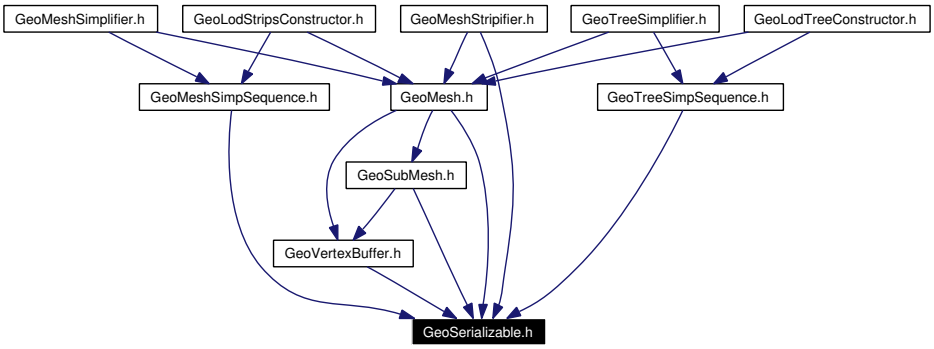
8.10 GeoSerializable.h File Reference

```
#include "GeoSerializer.h"
```

Include dependency graph for GeoSerializable.h:



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



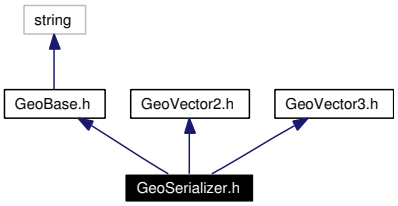
Namespaces

- namespace Geometry

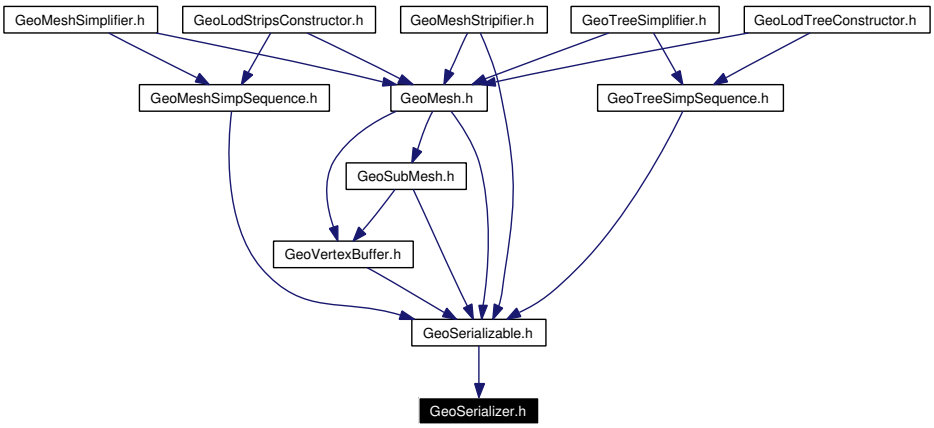
8.11 GeoSerializer.h File Reference

```
#include "GeoBase.h"
#include "GeoVector2.h"
#include "GeoVector3.h"
```

Include dependency graph for GeoSerializer.h:



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



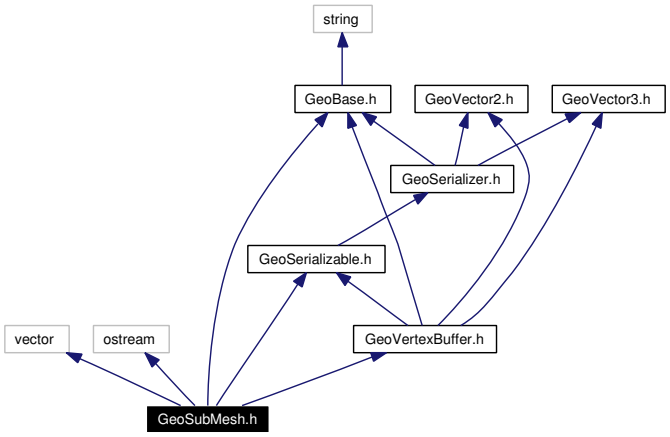
Namespaces

- namespace Geometry

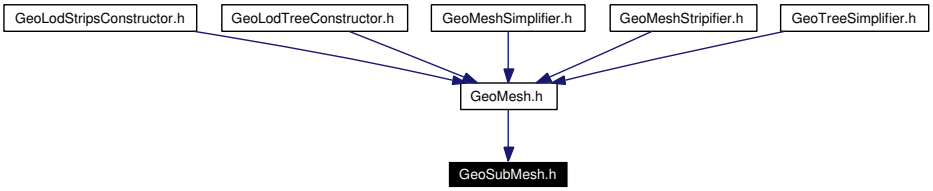
8.12 GeoSubMesh.h File Reference

```
#include <vector>
#include <ostream>
#include "GeoBase.h"
#include "GeoVertexBuffer.h"
#include "GeoSerializable.h"
```

Include dependency graph for GeoSubMesh.h:



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



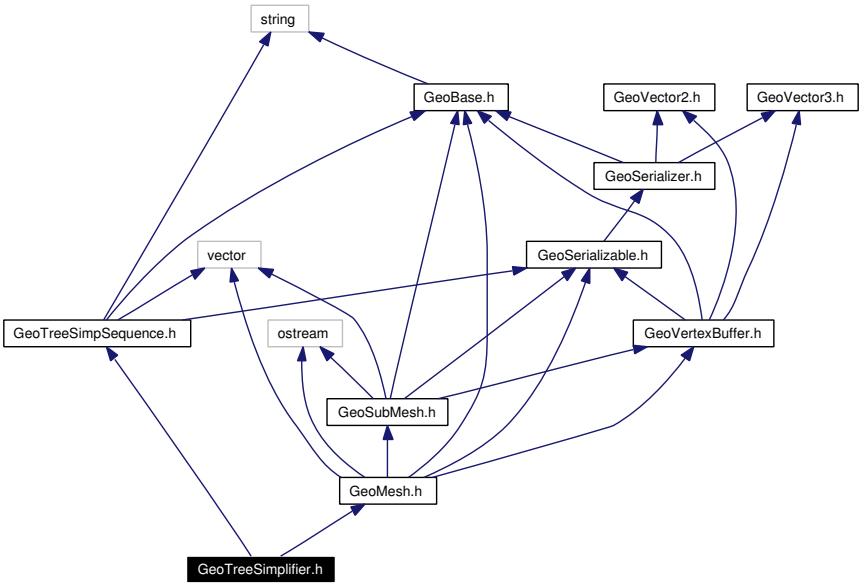
Namespaces

- namespace Geometry

8.13 GeoTreeSimplifier.h File Reference

```
#include "GeoMesh.h"
#include "GeoTreeSimpSequence.h"
```

Include dependency graph for GeoTreeSimplifier.h:



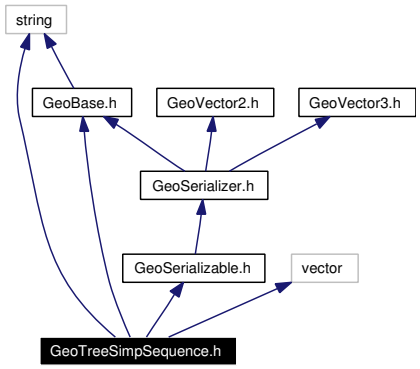
Namespaces

- namespace Geometry

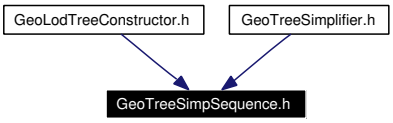
8.14 GeoTreeSimpSequence.h File Reference

```
#include <string>
#include <vector>
#include "GeoBase.h"
#include "GeoSerializable.h"
```

Include dependency graph for GeoTreeSimpSequence.h:



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

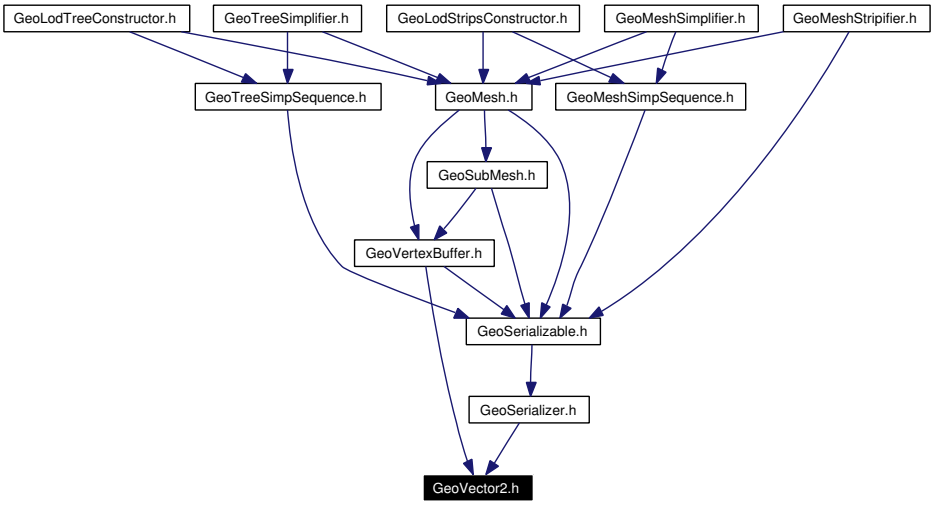


Namespaces

- namespace Geometry

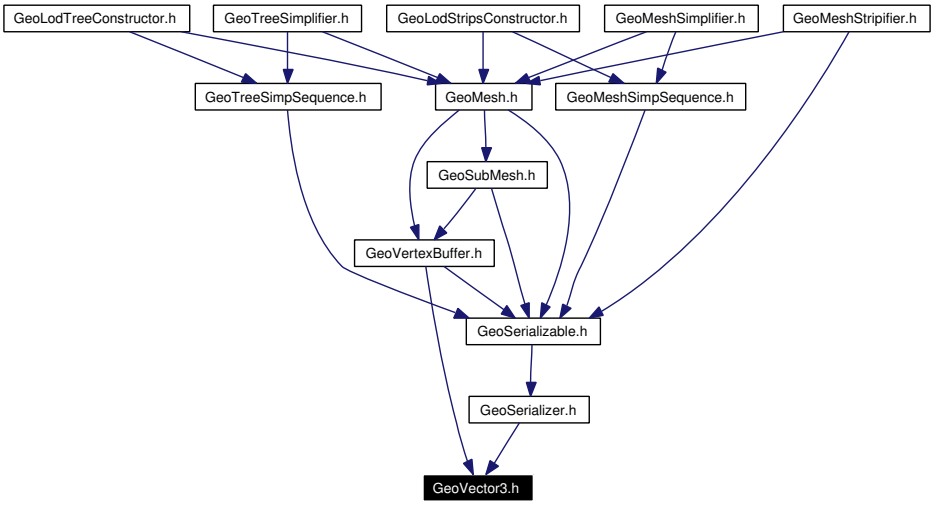
8.15 GeoVector2.h File Reference

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



8.16 GeoVector3.h File Reference

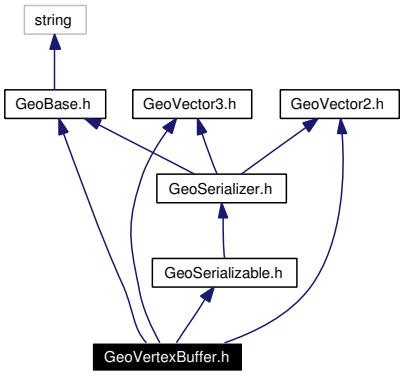
This graph shows which files directly or indirectly include this file:



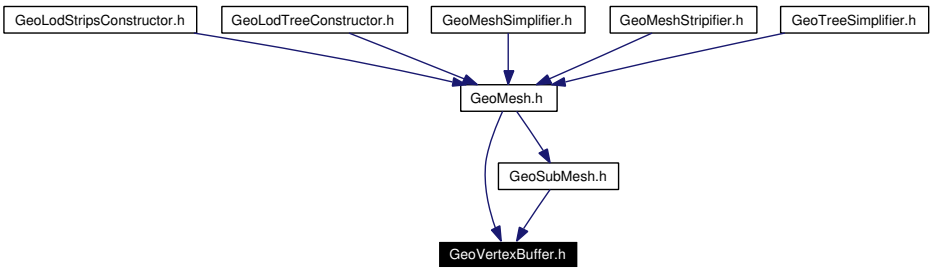
8.17 GeoVertexBuffer.h File Reference

```
#include "GeoBase.h"
#include "GeoVector3.h"
#include "GeoVector2.h"
#include "GeoSerializable.h"
```

Include dependency graph for GeoVertexBuffer.h:



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



Namespaces

- namespace Geometry

Index

~CustomStripifier
    Geometry::CustomStripifier, 18
~GeometryBasedSimplifier
    Geometry::GeometryBasedSimplifier, 19
~ImageBasedSimplifier
    Geometry::ImageBasedSimplifier, 21
~LodStripsConstructor
    Geometry::LodStripsConstructor, 24
~LodStripsLibrary
    Geometry::LodStripsLibrary, 26
~LodTreeConstructor
    Geometry::LodTreeConstructor, 29
~LodTreeLibrary
    Geometry::LodTreeLibrary, 31
~Mesh
    Geometry::Mesh, 35
~MeshSimplificationSequence
    Geometry::MeshSimplificationSequence, 38
~MeshSimplifier
    Geometry::MeshSimplifier, 41
~MeshStripifier
    Geometry::MeshStripifier, 44
~Serializer
    Geometry::Serializer, 48
~SubMesh
    Geometry::SubMesh, 51
~TreeSimplificationSequence
    Geometry::TreeSimplificationSequence, 54
~TreeSimplifier
    Geometry::TreeSimplifier, 57
~VertexBuffer
    Geometry::VertexBuffer, 60

APPEND
    Geometry::Serializer, 48

Clone
    Geometry::VertexBuffer, 60
CustomStripifier
    Geometry::CustomStripifier, 18

FlipEndian
    Geometry::Serializer, 48
FlipFromLittleEndian
    Geometry::Serializer, 48
FlipToLittleEndian
    Geometry::Serializer, 48

GEO_ENDIAN
    GeoBase.h, 64
GEO_ENDIAN_BIG
    GeoBase.h, 64
GEO_ENDIAN_LITTLE
    GeoBase.h, 64
GEO_TRIANGLE_LIST
    Geometry, 15
GEO_TRIANGLE_STRIPS
    Geometry, 15
GeoBase.h, 63
GeoBase.h
    GEO_ENDIAN, 64
    GEO_ENDIAN_BIG, 64
    GEO_ENDIAN_LITTLE, 64
GeoLodStripsConstructor.h, 65
GeoLodStripsLibrary.h, 66
GeoLodTreeConstructor.h, 67
GeoLodTreeLibrary.h, 68
GeoMesh.h, 69
GeoMeshSimplifier.h, 70
GeoMeshSimpSequence.h, 71
GeoMeshStripifier.h, 72
Geometry, 13
    GEO_TRIANGLE_LIST, 15
    GEO_TRIANGLE_STRIPS, 15
    Index, 15
    MeshType, 15
    Real, 15
    String, 15
    uint16, 15
    uint32, 15
    VERTEX_ALL, 16
    VERTEX_EMPTY, 16
    VERTEX_NORMAL, 16
    VERTEX_POSITION, 16
    VERTEX_TEXCOORDS, 16
Geometry::CustomStripifier, 17
Geometry::CustomStripifier
    ~CustomStripifier, 18
    CustomStripifier, 18

GetMesh, 18
Stripify, 18
Geometry::GeometryBasedSimplifier, 19
Geometry::GeometryBasedSimplifier
~GeometryBasedSimplifier, 19
GeometryBasedSimplifier, 19
Simplify, 20
Geometry::ImageBasedSimplifier, 21
Geometry::ImageBasedSimplifier
~ImageBasedSimplifier, 21
ImageBasedSimplifier, 21
Simplify, 22
Geometry::LodStripsConstructor, 23
Geometry::LodStripsConstructor
~LodStripsConstructor, 24
Load, 24
LodStripsConstructor, 24
Save, 24
Geometry::LodStripsLibrary, 25
Geometry::LodStripsLibrary
~LodStripsLibrary, 26
GoToLod, 26
LodStripsLibrary, 26
MaxFaces, 26
MaxLod, 26
MaxVertices, 26
MinFaces, 26
MinLod, 26
MinVertices, 26
TrimByLod, 26
Geometry::LodTreeConstructor, 28
Geometry::LodTreeConstructor
~LodTreeConstructor, 29
Load, 29
LodTreeConstructor, 29
Save, 29
Geometry::LodTreeLibrary, 30
Geometry::LodTreeLibrary
~LodTreeLibrary, 31
GoToFoliageLod, 31
GoToTrunkLod, 31
LodTreeLibrary, 31
MaxFoliageFaces, 31
MaxFoliageLod, 32
MaxFoliageVertices, 32
MaxTrunkFaces, 32
MaxTrunkLod, 32
MaxTrunkVertices, 32
MinFoliageFaces, 32
MinFoliageLod, 32
MinFoliageVertices, 32
MinTrunkFaces, 32
MinTrunkLod, 32
MinTrunkVertices, 32
TrimFoliageByLod, 33
TrimTrunkByLod, 33
Geometry::Mesh, 34
~Mesh, 35
Load, 35
Mesh, 35
mSubMesh, 35
mSubMeshCount, 35
mType, 35
mVertexBuffer, 35
Save, 35
Geometry::MeshSimplificationSequence, 37
Geometry::MeshSimplificationSequence
~MeshSimplificationSequence, 38
Load, 38
MeshSimplificationSequence, 38
mSteps, 38
Save, 38
Geometry::MeshSimplificationSequence::Step, 39
Geometry::MeshSimplificationSequence::Step
mModfaces, 39
mT0, 39
mT1, 39
mV0, 39
mV1, 39
Geometry::MeshSimplifier, 40
Geometry::MeshSimplifier
~MeshSimplifier, 41
GetMesh, 41
GetSimplificationSequence, 41
MeshSimplifier, 41
Simplify, 41
Geometry::MeshStripifier, 43
Geometry::MeshStripifier
~MeshStripifier, 44
GetMesh, 44
MeshStripifier, 44
Stripify, 44
Geometry::Serializable, 45
Load, 45
Save, 45
Geometry::Serializer, 47
~Serializer, 48
APPEND, 48
FlipEndian, 48
FlipFromLittleEndian, 48
FlipToLittleEndian, 48
GetSize, 48
mFile, 49
mMode, 49
Mode, 48
mSize, 49
READ, 48

ReadArray, 48, 49
ReadData, 49
Serializer, 48
WRITE, 48
WriteArray, 49
WriteData, 49
Geometry::SubMesh, 50
Geometry::SubMesh
~SubMesh, 51
Load, 51
mIndex, 51
mIndexCount, 51
mSharedVertexBuffer, 51
mStrip, 52
mStripCount, 52
mVertexBuffer, 52
Save, 51
SubMesh, 51
Geometry::TreeSimplificationSequence, 53
Geometry::TreeSimplificationSequence
~TreeSimplificationSequence, 54
Load, 54
mSteps, 54
Save, 54
TreeSimplificationSequence, 54
Geometry::TreeSimplificationSequence::Step, 56
Geometry::TreeSimplificationSequence::Step
mNewQuad, 56
mT0, 56
mT1, 56
mV0, 56
mV1, 56
Geometry::TreeSimplifier, 57
Geometry::TreeSimplifier
~TreeSimplifier, 57
GetMesh, 57
GetSimplificationSequence, 57
Simplify, 57
TreeSimplifier, 57
Geometry::VertexBuffer, 59
Geometry::VertexBuffer
~VertexBuffer, 60
Clone, 60
Load, 60
mNormal, 61
mPosition, 61
mTexCoords, 61
mVertexCount, 61
mVertexInfo, 61
Save, 60
VertexBuffer, 60
GeometryBasedSimplifier
Geometry::GeometryBasedSimplifier, 19
GeoSerializable.h, 73
GeoSerializer.h, 74
GeoSubMesh.h, 75
GeoTreeSimplifier.h, 76
GeoTreeSimpSequence.h, 77
GeoVector2.h, 78
GeoVector3.h, 79
GeoVertexBuffer.h, 80
GetMesh
Geometry::CustomStripifier, 18
Geometry::MeshSimplifier, 41
Geometry::MeshStripifier, 44
Geometry::TreeSimplifier, 57
GetSimplificationSequence
Geometry::MeshSimplifier, 41
Geometry::TreeSimplifier, 57
GetSize
Geometry::Serializer, 48
GoToFoliageLod
Geometry::LodTreeLibrary, 31
GoToLod
Geometry::LodStripsLibrary, 26
GoToTrunkLod
Geometry::LodTreeLibrary, 31

ImageBasedSimplifier
Geometry::ImageBasedSimplifier, 21
Index
Geometry, 15

Load
Geometry::LodStripsConstructor, 24
Geometry::LodTreeConstructor, 29
Geometry::Mesh, 35
Geometry::MeshSimplificationSequence, 38
Geometry::Serializable, 45
Geometry::SubMesh, 51
Geometry::TreeSimplificationSequence, 54
Geometry::VertexBuffer, 60
LodStripsConstructor
Geometry::LodStripsConstructor, 24
LodStripsLibrary
Geometry::LodStripsLibrary, 26
LodTreeConstructor
Geometry::LodTreeConstructor, 29
LodTreeLibrary
Geometry::LodTreeLibrary, 31

MaxFaces
Geometry::LodStripsLibrary, 26
MaxFoliageFaces
Geometry::LodTreeLibrary, 31
MaxFoliageLod

Geometry::LodTreeLibrary, 32
MaxFoliageVertices
Geometry::LodTreeLibrary, 32
MaxLod
Geometry::LodStripsLibrary, 26
MaxTrunkFaces
Geometry::LodTreeLibrary, 32
MaxTrunkLod
Geometry::LodTreeLibrary, 32
MaxTrunkVertices
Geometry::LodTreeLibrary, 32
MaxVertices
Geometry::LodStripsLibrary, 26
Mesh
Geometry::Mesh, 35
MeshSimplificationSequence
Geometry::MeshSimplificationSequence, 38
MeshSimplifier
Geometry::MeshSimplifier, 41
MeshStripifier
Geometry::MeshStripifier, 44
MeshType
Geometry, 15
mFile
Geometry::Serializer, 49
mIndex
Geometry::SubMesh, 51
mIndexCount
Geometry::SubMesh, 51
MinFaces
Geometry::LodStripsLibrary, 26
MinFoliageFaces
Geometry::LodTreeLibrary, 32
MinFoliageLod
Geometry::LodTreeLibrary, 32
MinFoliageVertices
Geometry::LodTreeLibrary, 32
MinLod
Geometry::LodStripsLibrary, 26
MinTrunkFaces
Geometry::LodTreeLibrary, 32
MinTrunkLod
Geometry::LodTreeLibrary, 32
MinTrunkVertices
Geometry::LodTreeLibrary, 32
MinVertices
Geometry::LodStripsLibrary, 26
mMode
Geometry::Serializer, 49
mModfaces
Geometry::MeshSimplification-Sequence::Step, 39
mNewQuad
Geometry::TreeSimplification-Sequence::Step, 56
mNormal
Geometry::VertexBuffer, 61
Mode
Geometry::Serializer, 48
mPosition
Geometry::VertexBuffer, 61
mSharedVertexBuffer
Geometry::SubMesh, 51
mSize
Geometry::Serializer, 49
mSteps
Geometry::MeshSimplificationSequence, 38
Geometry::TreeSimplificationSequence, 54
mStrip
Geometry::SubMesh, 52
mStripCount
Geometry::SubMesh, 52
mSubMesh
Geometry::Mesh, 35
mSubMeshCount
Geometry::Mesh, 35
mT0
Geometry::MeshSimplification-Sequence::Step, 39
Geometry::TreeSimplification-Sequence::Step, 56
mT1
Geometry::MeshSimplification-Sequence::Step, 39
Geometry::TreeSimplification-Sequence::Step, 56
mTexCoords
Geometry::VertexBuffer, 61
mType
Geometry::Mesh, 35
mV0
Geometry::MeshSimplification-Sequence::Step, 39
Geometry::TreeSimplification-Sequence::Step, 56
mV1
Geometry::MeshSimplification-Sequence::Step, 39
Geometry::TreeSimplification-Sequence::Step, 56
mVertexBuffer
Geometry::Mesh, 35
Geometry::SubMesh, 52
mVertexCount
Geometry::VertexBuffer, 61
mVertexInfo

Geometry::VertexBuffer, 61

READ
Geometry::Serializer, 48
ReadArray
Geometry::Serializer, 48, 49
ReadData
Geometry::Serializer, 49
Real
Geometry, 15

Save
Geometry::LodStripsConstructor, 24
Geometry::LodTreeConstructor, 29
Geometry::Mesh, 35
Geometry::MeshSimplificationSequence, 38
Geometry::Serializable, 45
Geometry::SubMesh, 51
Geometry::TreeSimplificationSequence, 54
Geometry::VertexBuffer, 60
Serializer
Geometry::Serializer, 48
Simplify
Geometry::GeometryBasedSimplifier, 20
Geometry::ImageBasedSimplifier, 22
Geometry::MeshSimplifier, 41
Geometry::TreeSimplifier, 57
String
Geometry, 15
Stripify
Geometry::CustomStripifier, 18
Geometry::MeshStripifier, 44
SubMesh
Geometry::SubMesh, 51

TreeSimplificationSequence
Geometry::TreeSimplificationSequence, 54
TreeSimplifier
Geometry::TreeSimplifier, 57
TrimByLod
Geometry::LodStripsLibrary, 26
TrimFoliageByLod
Geometry::LodTreeLibrary, 33
TrimTrunkByLod
Geometry::LodTreeLibrary, 33

uint16
Geometry, 15
uint32
Geometry, 15

VERTEX_ALL
Geometry, 16
VERTEX_EMPTY
Geometry, 16
VERTEX_NORMAL
Geometry, 16
VERTEX_POSITION
Geometry, 16
VERTEX_TEXCOORDS
Geometry, 16
VertexBuffer
Geometry::VertexBuffer, 60

WRITE
Geometry::Serializer, 48
WriteArray
Geometry::Serializer, 49
WriteData
Geometry::Serializer, 49