

2.0.1

Generated by Doxygen 1.7.1

Fri Oct 29 2010 18:19:08

Contents

1	Class Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	PolylibNS::BBox Class Reference	5
3.1.1	Detailed Description	6
3.1.2	Member Function Documentation	6
3.1.2.1	contain	6
3.1.2.2	crossed	6
3.1.2.3	getCrossedRegion	6
3.1.2.4	getFace	6
3.1.2.5	getSide	7
3.1.2.6	vec3to2	7
3.2	PolylibNS::CalcAreaInfo Struct Reference	7
3.2.1	Detailed Description	8
3.3	PolylibNS::MPIPolylib Class Reference	8
3.3.1	Detailed Description	9
3.3.2	Constructor & Destructor Documentation	9
3.3.2.1	MPIPolylib	9
3.3.2.2	MPIPolylib	9
3.3.3	Member Function Documentation	10
3.3.3.1	broadcast_config	10
3.3.3.2	broadcast_config_from_rank0	10
3.3.3.3	erase_outbounded_polygons	10
3.3.3.4	gather_polygons	10
3.3.3.5	get_instance	10

3.3.3.6	<code>get_myproc</code>	11
3.3.3.7	<code>get_proc</code>	11
3.3.3.8	<code>init_parallel_info</code>	11
3.3.3.9	<code>load</code>	11
3.3.3.10	<code>load_parallel</code>	12
3.3.3.11	<code>load_rank0</code>	12
3.3.3.12	<code>migrate</code>	12
3.3.3.13	<code>move</code>	12
3.3.3.14	<code>pack_num_trias</code>	13
3.3.3.15	<code>pack_tria_ids</code>	13
3.3.3.16	<code>pack_trias</code>	13
3.3.3.17	<code>receive_polygons_from_rank0</code>	13
3.3.3.18	<code>save</code>	14
3.3.3.19	<code>save_parallel</code>	14
3.3.3.20	<code>save_rank0</code>	14
3.3.3.21	<code>select_excluded_trias</code>	15
3.3.3.22	<code>send_polygons_to_all</code>	15
3.3.3.23	<code>send_polygons_to_rank0</code>	15
3.3.3.24	<code>show_group_name</code>	15
3.3.3.25	<code>used_memory_size</code>	15
3.4	PolylibNS::ParallelInfo Struct Reference	15
3.4.1	Detailed Description	16
3.5	PolylibNS::PolygonGroup Class Reference	16
3.5.1	Detailed Description	18
3.5.2	Constructor & Destructor Documentation	18
3.5.2.1	PolygonGroup	18
3.5.2.2	PolygonGroup	18
3.5.3	Member Function Documentation	19
3.5.3.1	<code>acq_file_name</code>	19
3.5.3.2	<code>acq_fullpath</code>	19
3.5.3.3	<code>add_children</code>	19
3.5.3.4	<code>add_triangles</code>	19
3.5.3.5	<code>build_group_tree</code>	19
3.5.3.6	<code>build_polygon_tree</code>	20
3.5.3.7	<code>check_leaped</code>	20
3.5.3.8	<code>get_children</code>	20

3.5.3.9	get_class_name	21
3.5.3.10	get_file_name	21
3.5.3.11	get_id	21
3.5.3.12	get_internal_id	21
3.5.3.13	get_movable	21
3.5.3.14	get_name	21
3.5.3.15	get_num_oftrias_before_move	22
3.5.3.16	get_parent	22
3.5.3.17	get_parent_path	22
3.5.3.18	get_triangles	22
3.5.3.19	get_vtree	22
3.5.3.20	init	22
3.5.3.21	init_check_leaped	23
3.5.3.22	is_far	23
3.5.3.23	linear_search	23
3.5.3.24	linear_search	24
3.5.3.25	load_id_file	24
3.5.3.26	load_stl_file	24
3.5.3.27	mk_basic_tag	25
3.5.3.28	mk_param_tag	25
3.5.3.29	move	25
3.5.3.30	rebuild_polygons	26
3.5.3.31	save_id_file	26
3.5.3.32	save_stl_file	26
3.5.3.33	search	26
3.5.3.34	search	27
3.5.3.35	search_outbounded	27
3.5.3.36	set_children	27
3.5.3.37	set_file_name	28
3.5.3.38	set_name	28
3.5.3.39	set_parent	28
3.5.3.40	set_parent_path	28
3.5.3.41	setup_attribute	28
3.5.3.42	show_group_info	29
3.5.4	Member Data Documentation	29
3.5.4.1	ATT_NAME_CLASS	29

3.6	PolylibNS::PolygonGroupFactory Class Reference	29
3.6.1	Detailed Description	29
3.6.2	Constructor & Destructor Documentation	29
3.6.2.1	PolygonGroupFactory	29
3.6.2.2	PolygonGroupFactory	29
3.6.3	Member Function Documentation	30
3.6.3.1	create_instance	30
3.7	PolylibNS::Polygons Class Reference	30
3.7.1	Detailed Description	31
3.7.2	Constructor & Destructor Documentation	31
3.7.2.1	Polygons	31
3.7.2.2	Polygons	31
3.7.3	Member Function Documentation	31
3.7.3.1	add	31
3.7.3.2	build	31
3.7.3.3	get_tri_list	31
3.7.3.4	get_vtree	32
3.7.3.5	import	32
3.7.3.6	init	32
3.7.3.7	linear_search	32
3.7.3.8	linear_search	33
3.7.3.9	search	33
3.7.3.10	search	34
3.7.3.11	triangles_num	34
3.8	PolylibNS::Polylib Class Reference	34
3.8.1	Detailed Description	36
3.8.2	Constructor & Destructor Documentation	36
3.8.2.1	Polylib	36
3.8.2.2	Polylib	36
3.8.3	Member Function Documentation	36
3.8.3.1	add_pg_list	36
3.8.3.2	check_group_name	36
3.8.3.3	create_polygon_group	37
3.8.3.4	get_group	37
3.8.3.5	get_group	37
3.8.3.6	get_instance	38

3.8.3.7	get_root_groups	38
3.8.3.8	load	38
3.8.3.9	load_config_file	38
3.8.3.10	load_polygons	39
3.8.3.11	load_with_idfile	39
3.8.3.12	make_group_tree	39
3.8.3.13	make_group_tree	40
3.8.3.14	move	40
3.8.3.15	save	40
3.8.3.16	save_config_file	41
3.8.3.17	save_polygons	41
3.8.3.18	save_with_rankno	41
3.8.3.19	search_polygons	42
3.8.3.20	set_factory	42
3.8.3.21	show_group_hierarchy	43
3.8.3.22	show_group_info	43
3.8.3.23	show_group_name	43
3.8.3.24	used_memory_size	43
3.9	PolylibNS::PolylibCfgElem Class Reference	44
3.9.1	Detailed Description	44
3.9.2	Constructor & Destructor Documentation	44
3.9.2.1	PolylibCfgElem	44
3.9.2.2	PolylibCfgElem	44
3.9.3	Member Function Documentation	44
3.9.3.1	first_element	44
3.9.3.2	first_param	45
3.9.3.3	get_name	45
3.9.3.4	next_element	45
3.9.3.5	next_param	45
3.9.3.6	set_elem	46
3.9.3.7	set_param	46
3.10	PolylibNS::PolylibCfgParam Class Reference	46
3.10.1	Detailed Description	46
3.10.2	Constructor & Destructor Documentation	46
3.10.2.1	PolylibCfgParam	46
3.10.3	Member Function Documentation	47

3.10.3.1	get_data_type	47
3.10.3.2	get_int_data	47
3.10.3.3	get_name	47
3.10.3.4	get_real_data	47
3.10.3.5	get_string_data	47
3.11	PolylibNS::PolylibConfig Class Reference	48
3.11.1	Detailed Description	48
3.11.2	Constructor & Destructor Documentation	48
3.11.2.1	PolylibConfig	48
3.11.2.2	PolylibConfig	48
3.11.2.3	PolylibConfig	49
3.11.3	Member Function Documentation	49
3.11.3.1	get_root_elem	49
3.11.3.2	load_config_file	49
3.11.3.3	mk_elem_tag	49
3.11.3.4	mk_param_tag	50
3.11.3.5	mk_param_tag	50
3.11.3.6	mk_param_tag	50
3.11.3.7	mk_parameter_tag	51
3.11.3.8	parse_xml_on_memory	51
3.11.3.9	save_file	51
3.12	PolylibNS::PolylibMoveParams Class Reference	51
3.12.1	Detailed Description	52
3.13	PolylibNS::PolylibStat2 Class Reference	52
3.14	PolylibNS::PrivateTriangle Class Reference	52
3.14.1	Detailed Description	53
3.14.2	Constructor & Destructor Documentation	53
3.14.2.1	PrivateTriangle	53
3.14.2.2	PrivateTriangle	53
3.14.2.3	PrivateTriangle	53
3.14.2.4	PrivateTriangle	54
3.14.2.5	PrivateTriangle	54
3.14.2.6	PrivateTriangle	54
3.14.3	Member Function Documentation	54
3.14.3.1	get_id	54
3.14.3.2	set_id	54

3.14.4	Member Data Documentation	55
3.14.4.1	m_id	55
3.15	PolylibNS::Triangle Class Reference	55
3.15.1	Detailed Description	56
3.15.2	Constructor & Destructor Documentation	56
3.15.2.1	Triangle	56
3.15.2.2	Triangle	56
3.15.2.3	Triangle	56
3.15.2.4	Triangle	56
3.15.3	Member Function Documentation	56
3.15.3.1	calc_area	56
3.15.3.2	calc_normal	57
3.15.3.3	get_area	57
3.15.3.4	get_normal	57
3.15.3.5	get_vertex	57
3.15.3.6	set_vertexes	57
3.16	TriangleStruct Struct Reference	57
3.16.1	Detailed Description	58
3.17	PolylibNS::TriMesh Class Reference	58
3.17.1	Detailed Description	58
3.17.2	Constructor & Destructor Documentation	59
3.17.2.1	TriMesh	59
3.17.2.2	TriMesh	59
3.17.3	Member Function Documentation	59
3.17.3.1	add	59
3.17.3.2	build	59
3.17.3.3	get_bbox	59
3.17.3.4	get_vtree	59
3.17.3.5	import	60
3.17.3.6	init	60
3.17.3.7	linear_search	60
3.17.3.8	linear_search	61
3.17.3.9	search	61
3.17.3.10	search	61
3.17.3.11	triangles_num	62
3.18	PolylibNS::TriMeshIO Class Reference	62

3.18.1 Detailed Description	62
3.18.2 Member Function Documentation	63
3.18.2.1 input_file_format	63
3.18.2.2 load	63
3.18.2.3 save	63
3.18.3 Member Data Documentation	64
3.18.3.1 FMT_STL_A	64
3.19 PolylibNS::Vec2< T > Class Template Reference	64
3.19.1 Detailed Description	65
3.20 PolylibNS::Vec3< T > Class Template Reference	65
3.20.1 Detailed Description	66
3.21 PolylibNS::VElement Class Reference	66
3.21.1 Detailed Description	66
3.21.2 Constructor & Destructor Documentation	67
3.21.2.1 VElement	67
3.21.3 Member Function Documentation	67
3.21.3.1 get_bbox	67
3.21.3.2 get_pos	67
3.21.3.3 get_triangle	67
3.22 PolylibNS::VNode Class Reference	67
3.22.1 Detailed Description	68
3.22.2 Constructor & Destructor Documentation	68
3.22.2.1 VNode	68
3.22.2.2 VNode	68
3.22.3 Member Function Documentation	68
3.22.3.1 get_axis	68
3.22.3.2 get_bbox	68
3.22.3.3 get_bbox_search	68
3.22.3.4 get_elements_num	69
3.22.3.5 get_left	69
3.22.3.6 get_right	69
3.22.3.7 get_vlist	69
3.22.3.8 is_leaf	69
3.22.3.9 set_axis	69
3.22.3.10 set_bbox	70
3.22.3.11 set_bbox_search	70

3.22.3.12	set_element	70
3.22.3.13	split	70
3.23	PolylibNS::VTree Class Reference	70
3.23.1	Detailed Description	71
3.23.2	Constructor & Destructor Documentation	71
3.23.2.1	VTree	71
3.23.2.2	VTree	71
3.23.3	Member Function Documentation	71
3.23.3.1	destroy	71
3.23.3.2	memory_size	71
3.23.3.3	search	71
3.23.3.4	search	72

Chapter 1

Class Index

1.1 Class Hierarchy

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

PolylibNS::BBox	5
PolylibNS::CalcAreaInfo	7
PolylibNS::ParallelInfo	15
PolylibNS::PolygonGroup	16
PolylibNS::PolygonGroupFactory	29
PolylibNS::Polygons	30
PolylibNS::TriMesh	58
PolylibNS::Polylib	34
PolylibNS::MPIPolylib	8
PolylibNS::PolylibCfgElem	44
PolylibNS::PolylibCfgParam	46
PolylibNS::PolylibConfig	48
PolylibNS::PolylibMoveParams	51
PolylibNS::PolylibStat2	52
PolylibNS::Triangle	55
PolylibNS::PrivateTriangle	52
TriangleStruct	57
PolylibNS::TriMeshIO	62
PolylibNS::Vec2 T	64
PolylibNS::Vec3 T	65
PolylibNS::VElement	66
PolylibNS::VNode	67
PolylibNS::VTree	70

Chapter 2

Class Index

2.1 Class List

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

PolylibNS::BBox	5
PolylibNS::CalcAreaInfo	7
PolylibNS::MPIPolylib	8
PolylibNS::ParallelInfo	15
PolylibNS::PolygonGroup	16
PolylibNS::PolygonGroupFactory	29
PolylibNS::Polygons	30
PolylibNS::Polylib	34
PolylibNS::PolylibCfgElem	44
PolylibNS::PolylibCfgParam	46
PolylibNS::PolylibConfig	48
PolylibNS::PolylibMoveParams	51
PolylibNS::PolylibStat2	52
PolylibNS::PrivateTriangle	52
PolylibNS::Triangle	55
TriangleStruct	57
PolylibNS::TriMesh	58
PolylibNS::TriMeshIO	62
PolylibNS::Vec2 T	64
PolylibNS::Vec3 T	65
PolylibNS::VElement	66
PolylibNS::VNode	67
PolylibNS::VTree	70

Chapter 3

Class Documentation

3.1 PolylibNS::BBox Class Reference

```
#include <include/common/BBox.h>
```

Public Member Functions

- **BBox** (float _minx, float _miny, float _minz, float _maxx, float _maxy, float _maxz)
- **BBox** (float _min[3], float _max[3])
- **BBox** (const [Vec3f](#) &_min, const [Vec3f](#) &_max)
- void **init** ()
- void **setMinMax** (const [Vec3f](#) &_min, const [Vec3f](#) &_max)
- void **add** (const [Vec3f](#) &v)
- [Vec3f](#) **getPoint** (int idx) const
- [Vec3f](#) **center** () const
- [Vec3f](#) **size** () const
- float **xsize** () const
- float **ysize** () const
- float **zsize** () const
- float **length** (const AxisEnum &axis) const
- float **diameter** () const
- AxisEnum **getMaxAxis** (float &length) const
- bool **contain** (const [Vec3f](#) &pos) const
- bool **crossed** (const **BBox** &bbox) const
- **BBox** **getCrossedRegion** (**BBox** &other_bbox) const
- [Vec2f](#) **vec3to2** (int axis_id, [Vec3f](#) &v3) const
- void **getFace** (int axis_id, [Vec3f](#) face[2][2]) const
- void **getSide** (int axis_id, [Vec3f](#) side[4][2]) const

Public Attributes

- [Vec3f](#) **min**
- [Vec3f](#) **max**

3.1.1 Detailed Description

クラス: [BBox](#) Bounding Box を管理するクラス

3.1.2 Member Function Documentation

3.1.2.1 `bool PolylibNS::BBox::contain (const Vec3f & pos) const [inline]`

引数で与えられた点が、この BBox に含まれるかを判定する。

Parameters

[in] *pos* 試行する点

Returns

含まれる場合は true。他は false。

3.1.2.2 `bool PolylibNS::BBox::crossed (const BBox & bbox) const [inline]`

BBox と BBox の交差判定を行う。 KD-Tree の交差判定と同じ。

Parameters

[in] *bbox* 試行する BBox

Returns

交差する場合は true。他は false。

3.1.2.3 `BBox PolylibNS::BBox::getCrossedRegion (BBox & other_bbox) const [inline]`

BBox と BBox の重複領域の抽出を行う。 自身の面と他方の辺との交差判定を行う。

Parameters

[in] *other_bbox* 試行する BBox

Returns

交差する場合は true。他は false。

3.1.2.4 `void PolylibNS::BBox::getFace (int axis_id, Vec3f face[2][2]) const [inline]`

引数 *axis_id*(0=x,1=y,z=2) に垂直な、この BBox の面の対角点を返す。

Parameters

[in] *axis_id* 軸番号。0=x 軸、1=y 軸、2=z 軸。

[in] *face* BBox の面の中で、軸に垂直な面の対角点。

3.1.2.5 `void PolylibNS::BBox::getSide (int axis_id, Vec3f side[4][2]) const`
`[inline]`

引数 `axis_id`(0=x,1=y,z=2) に平行な、この BBox の辺の端点を返す。

Parameters

- [in] *axis_id* 軸番号。0=x 軸、1=y 軸、2=z 軸。
- [in] *side* BBox の辺の中で、軸に平行な辺の端点。

3.1.2.6 `Vec2f PolylibNS::BBox::vec3to2 (int axis_id, Vec3f & v3) const`
`[inline]`

引数 `axis_id`(0=x,1=y,z=2) に垂直な成分を詰めて返す。

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

- `include/common/BBox.h`

3.2 PolylibNS::CalcAreaInfo Struct Reference

```
#include <include/Polylib.h>
```

Public Attributes

- [Vec3f m_bpos](#)
基点座標
- [Vec3f m_bbsize](#)
計算領域のボクセル数
- [Vec3f m_gcsiz](#)
ガイドセルのボクセル数
- [Vec3f m_dx](#)
ボクセル 1 辺の長さ
- [Vec3f m_gcell_min](#)
ガイドセルを含めた担当領域の最小位置
- [Vec3f m_gcell_max](#)
ガイドセルを含めた担当領域の最大位置
- [BBox m_gcell_bbox](#)
ガイドセルを含めた *Bounding Box*

3.2.1 Detailed Description

クラス:[CalcAreaInfo](#) 計算領域情報。

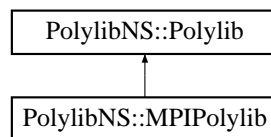
The documentation for this struct was generated from the following file:

- `include/Polylib.h`

3.3 PolylibNS::MPIPolylib Class Reference

```
#include <include/MPIPolylib.h>
```

Inheritance diagram for PolylibNS::MPIPolylib:



Public Member Functions

- POLYLIB_STAT [init_parallel_info](#) (MPI_Comm comm, float bpos[3], unsigned int bbsize[3], unsigned int gcsz[3], float dx[3])
- POLYLIB_STAT [load](#) (std::string config_filename)
- POLYLIB_STAT [load_rank0](#) (std::string config_filename="")
- POLYLIB_STAT [load_parallel](#) (std::string config_filename="")
- POLYLIB_STAT [save](#) (std::string p_config_filename)
- POLYLIB_STAT [save_rank0](#) (std::string p_config_filename, std::string stl_format, std::string extend="")
- POLYLIB_STAT [save_parallel](#) (std::string p_config_filename, std::string stl_format, std::string extend="")
- POLYLIB_STAT [move](#) (PolylibMoveParams ¶ms)
- POLYLIB_STAT [migrate](#) ()
- [ParallelInfo](#) [get_myproc](#) ()
- unsigned int [used_memory_size](#) ()

Static Public Member Functions

- static [MPIPolylib](#) [get_instance](#) ()

Protected Member Functions

- [MPIPolylib](#) ()
- [MPIPolylib](#) ()
- void [show_group_name](#) (PolygonGroup p, std::string tab)
- POLYLIB_STAT [broadcast_config](#) (std::string config_contents)
- POLYLIB_STAT [send_polygons_to_all](#) ()

- POLYLIB_STAT [pack_num_trias](#) (std::vector<int> p_vec, int group_id, const std::vector<[PrivateTriangle](#)> p_trias)
- POLYLIB_STAT [pack_trias](#) (std::vector<float> p_vec, const std::vector<[PrivateTriangle](#)> p_trias)
- POLYLIB_STAT [pack_tria_ids](#) (std::vector<int> p_vec, const std::vector<[PrivateTriangle](#)> p_trias)
- POLYLIB_STAT [erase_outbounded_polygons](#) ()
- POLYLIB_STAT [broadcast_config_from_rank0](#) ()
- POLYLIB_STAT [receive_polygons_from_rank0](#) ()
- POLYLIB_STAT [gather_polygons](#) ()
- POLYLIB_STAT [send_polygons_to_rank0](#) ()
- POLYLIB_STAT [select_excluded_trias](#) ([PolygonGroup](#) p_pg)
- [ParallelInfo](#) [get_proc](#) (int rank)

Protected Attributes

- [ParallelInfo](#) [m_myproc](#)
自 PE 担当領域情報
- std::vector<[ParallelInfo](#)> [m_other_procs](#)
自 PE を除く全 PE 担当領域情報リスト
- std::vector<[ParallelInfo](#)> [m_neighbour_procs](#)
隣接 PE 担当領域情報リスト
- int [m_myrank](#)
自プロセスのランク数
- int [m_numproc](#)
全プロセス数
- MPI_Comm [m_mycomm](#)
自プロセスが利用するコミュニケーター

3.3.1 Detailed Description

クラス:[MPIPolylib](#) ポリゴンを管理する為の並列版クラスライブラリです。

3.3.2 Constructor & Destructor Documentation

3.3.2.1 PolylibNS::MPIPolylib::MPIPolylib () [protected]

コンストラクタ。 singleton のため非公開。本クラスインスタンス取得には [get_instance\(\)](#) を利用する。

3.3.2.2 PolylibNS::MPIPolylib::~MPIPolylib () [protected]

デストラクタ。

3.3.3 Member Function Documentation

3.3.3.1 POLYLIB_STAT PolylibNS::MPIPolylib::broadcast_config (std::string *config_contents*) [protected]

設定ファイル内容を他 rank へ broadcast する。

Parameters

[in] *config_contents* 初期化ファイル内容。

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.2 POLYLIB_STAT PolylibNS::MPIPolylib::broadcast_config_from_rank0 () [protected]

ポリゴングループ定義情報を rank0 から受信し、グループ階層構造を構築。

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.3 POLYLIB_STAT PolylibNS::MPIPolylib::erase_outbounded_polygons () [protected]

自領域内ポリゴンのみ抽出してポリゴン情報を再構築。 migrate 実行後に行う。

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.4 POLYLIB_STAT PolylibNS::MPIPolylib::gather_polygons () [protected]

他 rank からポリゴン情報を rank0 で受信

3.3.3.5 static MPIPolylib PolylibNS::MPIPolylib::get_instance () [static]

インスタンス取得。本クラスは singleton クラスです。

Returns

MPIPolylib クラスのインスタンス

Reimplemented from [PolylibNS::Polylib](#).

3.3.3.6 ParallelInfo PolylibNS::MPIPolylib::get_myproc () [inline]

m_myproc の内容を get

Returns

自 PE 領域情報

3.3.3.7 ParallelInfo PolylibNS::MPIPolylib::get_proc (int rank) [protected]

プロセス担当領域クラスのポインタを返す

Parameters

[in] *rank* ランク数

Returns

プロセス担当領域クラスのポインタ

3.3.3.8 POLYLIB_STAT PolylibNS::MPIPolylib::init_parallel_info (MPI_Comm comm, float bpos[3], unsigned int bbsize[3], unsigned int gcsize[3], float dx[3])

並列計算関連情報の設定と初期化を行う。全 rank で各々設定を行い、その領域情報を全 rank へ配信する。

Parameters

- [in] *comm* MPI コミュニケーター
- [in] *bpos* 自 PE 担当領域の基点座標
- [in] *bbsize* 同、計算領域のボクセル数
- [in] *gcsize* 同、ガイドセルのボクセル数
- [in] *dx* 同、ボクセル 1 辺の長さ

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.9 POLYLIB_STAT PolylibNS::MPIPolylib::load (std::string config_filename) [inline]

[Polylib::load\(\)](#) のオーバーライドメソッド。並列環境では利用できません。

Parameters

[in] *config_filename* 初期化ファイル名。

Returns

常に PLSTAT_NG が返ります。

Reimplemented from [PolylibNS::Polylib](#).

3.3.3.10 POLYLIB_STAT PolylibNS::MPIPolylib::load_parallel (std::string *config_filename* =)

全 rank 並列でのデータ構築。指定された設定ファイルを各 rank にて読み込み、グループ階層構造の構築、およびポリゴンデータの構築を行う。各 rank が読み込むファイルに記述されたグループ階層構造が一致している必要がある。

Parameters

[in] *config_filename* 初期化ファイル名。未指定時はデフォルトファイルを読む。

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.11 POLYLIB_STAT PolylibNS::MPIPolylib::load_rank0 (std::string *config_filename* =)

rank0 によるデータ構築。指定された設定ファイルを rank0 にて読み込み、グループ階層構造の構築 およびポリゴンデータの構築を行う。グループ階層構造は全 rank に b_cast され、情報を共有する。ポリゴンデータは各 rank 領域毎のデータが分配される。

Parameters

[in] *config_filename* 初期化ファイル名。未指定時はデフォルトファイルを読む。

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.12 POLYLIB_STAT PolylibNS::MPIPolylib::migrate ()

ポリゴンデータの PE 間移動。本クラスインスタンス配下の全 PolygonGroup のポリゴンデータについて、move メソッドにより移動した三角形ポリゴン情報を隣接 PE 間でやり取りする。

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.13 POLYLIB_STAT PolylibNS::MPIPolylib::move (PolylibMoveParams & *params*)

ポリゴン座標の移動。本クラスインスタンス配下の全 PolygonGroup の move メソッドが呼び出される。

Parameters

[in] *params* 移動計算要パラメタセット。

Returns

POLYLIB_STAT で定義される値が返る。

Reimplemented from [PolylibNS::Polylib](#).

3.3.3.14 `POLYLIB_STAT PolylibNS::MPIPolylib::pack_numtrias (std::vector<int> p_vec, int group_id, const std::vector<PrivateTriangle> p_trias)` [protected]

グループ ID & グループ内三角形数の送信情報を作成。

Parameters

[in,out] *p_vec* 情報追加先ベクタ
 [in] *group_id* グループ ID
 [in] *p_trias* グループ内三角形リスト

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.15 `POLYLIB_STAT PolylibNS::MPIPolylib::pack_tria_ids (std::vector<int> p_vec, const std::vector<PrivateTriangle> p_trias)` [protected]

三角形 ID の送信情報を作成。

Parameters

[in,out] *p_vec* 情報追加先ベクタ
 [in] *p_trias* グループ内三角形リスト

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.16 `POLYLIB_STAT PolylibNS::MPIPolylib::pack_trias (std::vector<float> p_vec, const std::vector<PrivateTriangle> p_trias)` [protected]

三角形の送信情報を作成。

Parameters

[in,out] *p_vec* 情報追加先ベクタ
 [in] *p_trias* グループ内三角形リスト

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.17 `POLYLIB_STAT PolylibNS::MPIPolylib::receive_polygons_from_rank0 ()` [protected]

自領域に必要なポリゴン情報を rank0 から受信

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.18 POLYLIB_STAT PolylibNS::MPIPolylib::save (std::string *p_config_filename*) [inline]

[Polylib::save\(\)](#) のオーバーライドメソッド。 並列環境では利用できません。

Parameters

[out] *p_config_filename* 初期化ファイル名。

Returns

常に PLSTAT_NG が返ります。

3.3.3.19 POLYLIB_STAT PolylibNS::MPIPolylib::save_parallel (std::string *p_config_filename*, std::string *stl_format*, std::string *extend* =)

全 rank 並列でのデータ保存。 各 rank の本クラスインスタンスが保持するグループ階層構造を設定ファイルに各 rank 毎に書き出す。 同時にポリゴンデータも指定されたフォーマットの STL ファイルに各 rank 毎に書き出す。 設定ファイル命名規則は以下の通り *polylib_config_*ランク番号_付加文字列.xml STL ファイル命名規則は以下の通り ポリゴングループ名称_ランク番号_付加文字列.拡張子

Parameters

[out] *p_config_filename* 設定ファイル名返却用 string インスタンスへのポインタ

[in] *stl_format* STL ファイルフォーマット。 "stl_a":アスキー形式 "stl_b":バイナリ形式

[in] *extend* ファイル名に付加する文字列。省略可。省略 した場合は、付加文字列として本メソッド呼 び出し時の年月日時分秒 (YYYYMMDD24hhmmss) を用いる。

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.20 POLYLIB_STAT PolylibNS::MPIPolylib::save_rank0 (std::string *p_config_filename*, std::string *stl_format*, std::string *extend* =)

rank0 によるデータ保存。 rank0 の本クラスインスタンスが保持するグループ階層構造を設定ファイルに書き出す。 同時に各 rank に分散するポリゴンデータも rank0 に集められ、指定されたフォーマットの STL ファイルに rank0 で書き出す。 設定ファイル命名規則は以下の通り *polylib_config_*付加文字列.xml STL ファイル命名規則は以下の通り ポリゴングループ名称_付加文字列.拡張子

Parameters

[out] *p_config_filename* 設定ファイル名返却用 string インスタンスへのポインタ

[in] *stl_format* STL ファイルフォーマット。 "stl_a":アスキー形式 "stl_b":バイナリ形式

[in] *extend* ファイル名に付加する文字列。省略可。省略 した場合は、付加文字列として本メソッド呼 び出し時の年月日時分秒 (YYYYMMDD24hhmmss) を用いる。

Returns

POLYLIB_STAT で定義される値が返る。 出力引数 *p_config_filename* の返却値は rank0 のみ有効

3.3.3.21 POLYLIB_STAT PolylibNS::MPIPolylib::select_excludedtrias (PolygonGroup *p_pg*) [protected]

移動除外三角形 ID リストの作成

3.3.3.22 POLYLIB_STAT PolylibNS::MPIPolylib::send_polygons_to_all () [protected]

各 PE 領域内ポリゴン情報を全 rank に送信

Returns

POLYLIB_STAT で定義される値が返る。

3.3.3.23 POLYLIB_STAT PolylibNS::MPIPolylib::send_polygons_to_rank0 () [protected]

rank0 へポリゴン情報を送信

3.3.3.24 void PolylibNS::MPIPolylib::show_group_name (PolygonGroup *p*, std::string *tab*) [protected]

指定されたグループ以下の階層構造をツリー形式で標準出力に出力する。

Parameters

p 表示対象となるグループのポインタ。
tab 階層の深さを示すスペース。

Attention

プロセス毎に動作する。 出力にランク数加わる以外は非並列版と同じ。

3.3.3.25 unsigned int PolylibNS::MPIPolylib::used_memory_size ()

MPIPolylib が利用中の概算メモリ量を返す

Returns

利用中のメモリ量 (byte)

Reimplemented from [PolylibNS::Polylib](#).

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

- include/MPIPolylib.h

3.4 PolylibNS::ParallelInfo Struct Reference

```
#include include/MPIPolylib.h
```

Public Attributes

- MPI_Comm [m_comm](#)
MPI コミュニケータ.
- int [m_rank](#)
ランク数
- CalcAreaInfo [m_area](#)
計算領域情報
- std::map<int, std::vector<int>> [m_exclusion_map](#)
migrate 除外三角形 ID マップ (*k*:グループ ID, *v*:三角形 ID リスト)

3.4.1 Detailed Description

クラス:[ParallelInfo](#) 並列プロセス情報。

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

- include/MPIPolylib.h

3.5 PolylibNS::PolygonGroup Class Reference

```
#include <include/groups/PolygonGroup.h>
```

Public Member Functions

- [PolygonGroup](#) ()
- virtual [PolygonGroup](#) ()
- POLYLIB_STAT [init](#) (const std::vector< [PrivateTriangle](#) > tri_list, bool clear=true)
- virtual POLYLIB_STAT [build_group_tree](#) ([Polylib](#) polylib, [PolygonGroup](#) parent, const [PolylibCfgElem](#) elem)
- POLYLIB_STAT [build_polygon_tree](#) ()
- POLYLIB_STAT [load_stl_file](#) ()
- POLYLIB_STAT [load_id_file](#) ()
- POLYLIB_STAT [save_stl_file](#) (std::string rank_no, std::string extend, std::string format)
- POLYLIB_STAT [save_id_file](#) (std::string rank_no, std::string extend)
- virtual POLYLIB_STAT [mk_param_tag](#) (xmlNodePtr elem, std::string rank_no, std::string extend, std::string format)
- virtual POLYLIB_STAT [move](#) ([PolylibMoveParams](#) ¶ms)
- const std::vector< [PrivateTriangle](#) > [search](#) ([BBox](#) bbox, bool every) const
- POLYLIB_STAT [search](#) ([BBox](#) bbox, bool every, std::vector< [PrivateTriangle](#) > tri_list) const
- const std::vector< [PrivateTriangle](#) > [linear_search](#) ([BBox](#) bbox, bool every) const
- POLYLIB_STAT [linear_search](#) ([BBox](#) bbox, bool every, std::vector< [PrivateTriangle](#) > tri_list) const
- std::string [acq_fullpath](#) ()

- std::string [acq_file_name](#) ()
- const std::vector [PrivateTriangle](#) [search_outbounded](#) ([BBox](#) neighbour_bbox, std::vector<int> exclude_tria_ids)
- POLYLIB_STAT [add_triangles](#) (std::vector<[PrivateTriangle](#)> tri_list)
- POLYLIB_STAT [rebuild_polygons](#) ()
- POLYLIB_STAT [show_group_info](#) (int irank=-1)
- virtual std::string [whoami](#) ()
- void [set_file_name](#) (std::map<std::string, std::string> fname)
- std::map<std::string, std::string> [get_file_name](#) () const
- void [set_name](#) (std::string name)
- std::string [get_name](#) (void)
- void [set_parent_path](#) (std::string ppath)
- std::string [get_parent_path](#) (void)
- void [set_parent](#) ([PolygonGroup](#) p)
- [PolygonGroup](#) [get_parent](#) (void)
- void [set_children](#) (std::vector<[PolygonGroup](#)> &p)
- std::vector<[PolygonGroup](#)> & [get_children](#) (void)
- void [add_children](#) ([PolygonGroup](#) p)
- std::vector<[PrivateTriangle](#)> [get_triangles](#) ()
- [VTree](#) [get_vtree](#) ()
- int [get_internal_id](#) ()
- int [get_id](#) ()
- int [get_movable](#) ()
- size_t [get_num_oftrias_before_move](#) ()

Static Public Member Functions

- static std::string [get_class_name](#) ()

Static Public Attributes

- static const char [ATT_NAME_CLASS](#)

Protected Member Functions

- POLYLIB_STAT [setup_attribute](#) ([Polylib](#) polylib, [PolygonGroup](#) parent, const [PolylibCfgElem](#) elem)
- POLYLIB_STAT [init_check_leaped](#) ()
- POLYLIB_STAT [check_leaped](#) ([Vec3f](#) origin, [Vec3f](#) cell_size)
- bool [is_far](#) ([Vec3f](#) origin, [Vec3f](#) cell_size, [Vec3f](#) pos1, [Vec3f](#) pos2)
- POLYLIB_STAT [mk_basic_tag](#) (xmlNodePtr elem, std::string rank_no, std::string extend, std::string format)

Protected Attributes

- `int m_internal_id`
グループ ID。
- `std::string m_name`
自グループ名。
- `std::string m_parent_path`
親グループのパス名。
- `PolygonGroup m_parent`
親グループへのポインタ。
- `std::vector< PolygonGroup> m_children`
子グループへのポインタリスト。
- `std::map< std::string, std::string> m_file_name`
STL ファイル名とファイル形式。
- `Polygons m_polygons`
三角形 *Polygons* クラス。
- `bool m_movable`
move メソッドにより移動するグループか？
- `bool m_need_rebuild`
KD 木の再構築が必要か？
- `std::vector< PrivateTriangle> mtrias_before_move`
*move()*による移動前三角形一時保存リスト。

3.5.1 Detailed Description

クラス:`PolygonGroup` ポリゴングループを管理するクラスです。

3.5.2 Constructor & Destructor Documentation

3.5.2.1 `PolylibNS::PolygonGroup::PolygonGroup ()`

コンストラクタ

3.5.2.2 `virtual PolylibNS::PolygonGroup::~PolygonGroup () [virtual]`

デストラクタ

3.5.3 Member Function Documentation

3.5.3.1 `std::string PolylibNS::PolygonGroup::acq_file_name ()`

カンマ区切りで STL ファイル名リストを取得。

Returns

ファイル名リスト。

3.5.3.2 `std::string PolylibNS::PolygonGroup::acq_fullpath ()`

PolygonGroup のフルパス名を取得する。

Returns

フルパス名。

3.5.3.3 `void PolylibNS::PolygonGroup::add_children (PolygonGroup p)` [inline]

子グループを追加。

Parameters

[in] 子グループ。

3.5.3.4 `POLYLIB_STAT PolylibNS::PolygonGroup::add_triangles (std::vector PrivateTriangle tri_list)`

三角形リストの追加。

Parameters

[in] *tri_list* 三角形ポリゴンリストのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

三角形 ID が重複した三角形は追加しない。KD 木の再構築はしない。

3.5.3.5 `virtual POLYLIB_STAT PolylibNS::PolygonGroup::build_group_tree (Polylib polylib, PolygonGroup parent, const PolylibCfgElem elem)` [virtual]

PolygonGroup ツリーの作成。 設定ファイルの内容を再帰的に呼び出し、PolygonGroup ツリーを作成する。

Parameters

- [in] *polylib* Polygon クラスのインスタンス
- [in] *parent* 親グループ
- [in] *elem* 設定ファイルの Elem タグ

Returns

POLYLIB_STAT で定義される値が返る。

3.5.3.6 POLYLIB_STAT PolylibNS::PolygonGroup::build_polygon_tree ()

三角形ポリゴンの法線ベクトルの計算、面積の計算、KD 木の生成を行う。三角形ポリゴンは TriMesh クラスが管理している。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

TriMesh クラスの build() 参照。

3.5.3.7 POLYLIB_STAT PolylibNS::PolygonGroup::check_leaped (Vec3f origin, Vec3f cell_size) [protected]

[move\(\)](#) メソッド実行により、頂点が隣接セルよりも遠くへ移動した三角形情報を報告 (後処理)。該当する三角形について、以下の情報を cerr へ出力する。 ・ ポリゴングループ ID ・ 三角形 ID ・ 移動前/後の頂点座標

Parameters

- [in] *origin* 計算領域起点座標
- [in] *cell_size* ボクセルサイズ

Returns

POLYLIB_STAT で定義される値が返る。本メソッドはデバッグ用です。派生クラスでオーバーライドした move() メソッド内で、座標移動 処理後に呼ぶこと。

3.5.3.8 std::vector PolygonGroup & PolylibNS::PolygonGroup::get_children (void) [inline]

子グループを取得。

Returns

子グループのリスト。

3.5.3.9 `static std::string PolylibNS::PolygonGroup::get_class_name ()`
[inline, static]

クラス名を取得。

Returns

クラス名。

Attention

本クラスを継承する場合、継承後のクラス名を返すように変更すること。

3.5.3.10 `std::map<std::string, std::string> PolylibNS::PolygonGroup::get_file_name () const` [inline]

STL ファイル名とファイルフォーマットの対応マップ取得。

Returns

STL ファイル名とファイルフォーマットの対応マップ。

3.5.3.11 `int PolylibNS::PolygonGroup::get_id ()` [inline]

ユーザ定義 ID を取得。 処理追加 2010.10.20

Returns

ユーザ定義 ID。

3.5.3.12 `int PolylibNS::PolygonGroup::get_internal_id ()` [inline]

ポリゴングループ ID を取得。 メンバー名修正 (m_id - m_internal_id) 2010.10.20

Returns

ポリゴングループ ID。

3.5.3.13 `int PolylibNS::PolygonGroup::get_movable ()` [inline]

移動対象フラグを取得。

Returns

移動対象フラグ。

3.5.3.14 `std::string PolylibNS::PolygonGroup::get_name (void)` [inline]

グループ名を取得。

Returns

グループ名。

3.5.3.15 `size_t PolylibNS::PolygonGroup::get_num_of_trias_before_move ()`
[inline]

`move()`による移動前三角形一時保存リストの個数を取得。

Returns

一時保存リストサイズ。

3.5.3.16 `PolygonGroup PolylibNS::PolygonGroup::get_parent (void)` [inline]

親グループを取得。

Returns

親グループのポインタ。

3.5.3.17 `std::string PolylibNS::PolygonGroup::get_parent_path (void)`
[inline]

親グループのフルパス名を取得。

Returns

親グループのフルパス名。

3.5.3.18 `std::vector PrivateTriangle PolylibNS::PolygonGroup::get_triangles ()`
[inline]

Polygon クラスが管理する三角形ポリゴンリストを取得。

Returns

三角形ポリゴンリスト。

3.5.3.19 `VTree PolylibNS::PolygonGroup::get_vtree ()` [inline]

Polygon クラスが管理する KD 木クラスを取得。

Returns

KD 木ポリゴンリスト。

3.5.3.20 `POLYLIB_STAT PolylibNS::PolygonGroup::init (const std::vector
PrivateTriangle tri_list, bool clear =)`

引数で与えられる三角形ポリゴンリストを複製し、KD 木の生成を行う。

Parameters

- [in] *tri_list* 設定する三角形ポリゴンリスト。
- [in] *clear* true:ポリゴン複製、面積計算、KD 木生成を行う。 false:面積計算、KD 木生成だけを行う。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

TriMesh クラスの `init()` 参照。オーバーロードメソッドあり。

3.5.3.21 POLYLIB_STAT PolylibNS::PolygonGroup::init_check_leaped () [protected]

`move()` メソッド実行により、頂点が隣接セルよりも遠くへ移動した三角形情報を報告 (前処理)。

Returns

POLYLIB_STAT で定義される値が返る。本メソッドはデバッグ用です。派生クラスでオーバーライドした `move()` メソッド内で、座標移動 処理前に呼ぶこと。

3.5.3.22 bool PolylibNS::PolygonGroup::is_far (Vec3f origin, Vec3f cell_size, Vec3f pos1, Vec3f pos2) [protected]

2 点が隣接ボクセルよりも離れているか？

Parameters

- [in] *origin* 計算領域起点座標。
- [in] *cell_size* ボクセルサイズ。
- [in] *pos1* 点 (1)。
- [in] *pos2* 点 (2)。

Returns

true:2 点が隣接ボクセルよりも離れている。

3.5.3.23 const std::vector PrivateTriangle PolylibNS::PolygonGroup::linear_search (BBox bbox, bool every) const

線形探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

- [in] *bbox* 矩形領域。
- [in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

抽出したポリゴンリストのポインタ。

Attention

オーバーロードメソッドあり。

**3.5.3.24 POLYLIB_STAT PolylibNS::PolygonGroup::linear_search (BBox
bbox, bool every, std::vector PrivateTriangle tri_list) const**

線形探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

[in] *bbox* 矩形領域。

[in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

[in,out] *tri_list* 抽出した三角形ポリゴンリストのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

オーバーロードメソッドあり。

3.5.3.25 POLYLIB_STAT PolylibNS::PolygonGroup::load_id_file ()

三角形ポリゴン ID ファイルからポリゴン ID を読み込み、m_internal_id に登録する。

Returns

POLYLIB_STAT で定義される値が返る。

3.5.3.26 POLYLIB_STAT PolylibNS::PolygonGroup::load_stl_file ()

STL ファイルからポリゴン情報を読み込み、TriMesh クラスに登録する。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

TriMesh クラスの import() 参照。

3.5.3.27 `POLYLIB_STAT PolylibNS::PolygonGroup::mk_basic_tag (xmlNodePtr elem, std::string rank_no, std::string extend, std::string format)` [protected]

PolygonGroup の基本情報を設定ファイルに出力するための param タグを作成。

Parameters

- [in] *elem* XML ノード。
- [in] *rank_no* ファイル名に付加するランク番号。
- [in] *extend* ファイル名に付加する自由文字列。
- [in] *format* STL ファイルフォーマット。

Returns

POLYLIB_STAT で定義される値が返る。

3.5.3.28 `virtual POLYLIB_STAT PolylibNS::PolygonGroup::mk_param_tag (xmlNodePtr elem, std::string rank_no, std::string extend, std::string format)` [virtual]

設定ファイルに出力する param タグを作成する。

Parameters

- [in] *elem* XML ノード。
- [in] *rank_no* ファイル名に付加するランク番号。
- [in] *extend* ファイル名に付加する自由文字列。
- [in] *format* STL ファイルフォーマット。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

本クラスの `mk_basic_tag()` 参照。

3.5.3.29 `virtual POLYLIB_STAT PolylibNS::PolygonGroup::move (PolylibMoveParams & params)` [virtual]

三角形ポリゴン移動メソッド。virtual 用の関数なので処理はない。

Parameters

- [in] *param* [Polylib.h](#) で宣言しているパラメタセットクラス。

Returns

POLYLIB_STAT で定義される値が返る。

3.5.3.30 POLYLIB_STAT PolylibNS::PolygonGroup::rebuild_polygons ()

ポリゴン情報を再構築する。(KD 木の再構築をおこなう)

Returns

POLYLIB_STAT で定義される値が返る。

3.5.3.31 POLYLIB_STAT PolylibNS::PolygonGroup::save_id_file (std::string *rank_no*, std::string *extend*)

三角形ポリゴン ID ファイルにポリゴン ID を出力する。ID ファイル名は、階層化されたグループ名_ランク番号_自由文字列.id。

Parameters

- [in] *rank_no* ファイル名に付加するランク番号。
- [in] *extend* ファイル名に付加する自由文字列。

Returns

POLYLIB_STAT で定義される値が返る。

3.5.3.32 POLYLIB_STAT PolylibNS::PolygonGroup::save_stl_file (std::string *rank_no*, std::string *extend*, std::string *format*)

TriMesh クラスが管理しているポリゴン情報を STL ファイルに出力する。

Parameters

- [in] *rank_no* ファイル名に付加するランク番号。
- [in] *extend* ファイル名に付加する自由文字列。
- [in] *format* STL ファイルフォーマット。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

TriMeshIO クラスの save() 参照。オーバーロードメソッドあり。

3.5.3.33 POLYLIB_STAT PolylibNS::PolygonGroup::search (BBox *bbox*, bool *every*, std::vector<PrivateTriangle> *tri_list*) const

KD 木探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

- [in] *bbox* 矩形領域。
- [in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

[in,out] *tri_list* 抽出した三角形ポリゴンリストのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

オーバーロードメソッドあり。

3.5.3.34 const std::vector PrivateTriangle PolylibNS::PolygonGroup::search (BBox bbox, bool every) const

KD 木探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

[in] *bbox* 矩形領域。

[in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

抽出したポリゴンリストのポインタ。

Attention

オーバーロードメソッドあり。

3.5.3.35 const std::vector PrivateTriangle PolylibNS::PolygonGroup::search_outbounded (BBox neighbour_bbox, std::vector int exclude_tria_ids)

PE 領域間移動する三角形ポリゴンリストの取得。

Parameters

[in] *neighbour_bbox* 隣接 PE 領域バウンディングボックス。

[in] *exclude_tria_ids* 領域移動対象外三角形 ID リスト。

Returns

検索結果三角形リスト。

3.5.3.36 void PolylibNS::PolygonGroup::set_children (std::vector PolygonGroup & p) [inline]

子グループを設定。

Parameters

[in] 子グループのリスト。

3.5.3.37 `void PolylibNS::PolygonGroup::set_file_name (std::map<std::string, std::string> fname) [inline]`

STL ファイル名とファイルフォーマットを設定。

Parameters

[in] *fname* STL ファイル名とファイルフォーマットの対応マップ。

3.5.3.38 `void PolylibNS::PolygonGroup::set_name (std::string name) [inline]`

グループ名を設定。

Parameters

[in] *name* グループ名。

3.5.3.39 `void PolylibNS::PolygonGroup::set_parent (PolygonGroup p) [inline]`

親グループを設定。

Parameters

[in] *p* 親グループのポインタ。

3.5.3.40 `void PolylibNS::PolygonGroup::set_parent_path (std::string ppath) [inline]`

親グループのフルパス名を設定。

Parameters

[in] *pname* 親グループのフルパス名。

3.5.3.41 `POLYLIB_STAT PolylibNS::PolygonGroup::setup_attribute (Polylib polylib, PolygonGroup parent, const PolylibCfgElem elem) [protected]`

設定ファイルから取得した PolygonGroup の情報をインスタンスにセットする。

Parameters

[in] *polylib* Polygon クラスのインスタンス。

[in] *parent* 親グループ。

[in] *elem* 設定ファイルの Elem タグ。

Returns

POLYLIB_STAT で定義される値が返る。

3.5.3.42 POLYLIB_STAT PolylibNS::PolygonGroup::show_group_info (int *irank* =)

グループ情報（ランク番号、親グループ名、自分のグループ名、ファイル名、頂点数、各頂点の XYZ 座標値、法線ベクトルの XYZ 座標値、面積）を出力する。

Parameters

[in] *irank* ランク数。

Returns

POLYLIB_STAT で定義される値が返る。

3.5.4 Member Data Documentation

3.5.4.1 const char PolylibNS::PolygonGroup::ATT_NAME_CLASS [static]

config ファイルに記述する Param タグのクラス名 (value="...")。

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

- include/groups/PolygonGroup.h

3.6 PolylibNS::PolygonGroupFactory Class Reference

```
#include include/groups/PolygonGroupFactory.h
```

Public Member Functions

- [PolygonGroupFactory](#) ()
- virtual [PolygonGroupFactory](#) ()
- virtual [PolygonGroup](#) [create_instance](#) (std::string class_name)

3.6.1 Detailed Description

クラス:[PolygonGroupFactory](#)

3.6.2 Constructor & Destructor Documentation

3.6.2.1 PolylibNS::PolygonGroupFactory::PolygonGroupFactory () [inline]

コンストラクタ。

3.6.2.2 virtual PolylibNS::PolygonGroupFactory:: ~PolygonGroupFactory () [inline, virtual]

デストラクタ。

3.6.3 Member Function Documentation

3.6.3.1 virtual PolygonGroup PolylibNS::PolygonGroupFactory::create_instance (std::string class_name) [inline, virtual]

インスタンス作成。

Parameters

[in] *class_name* 作成するクラス名。

Returns

作成に失敗した場合は NULL が返る。

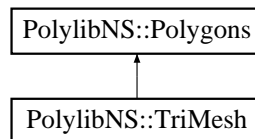
The documentation for this class was generated from the following file:

- include/groups/PolygonGroupFactory.h

3.7 PolylibNS::Polygons Class Reference

```
#include include/polygons/Polygons.h
```

Inheritance diagram for PolylibNS::Polygons:



Public Member Functions

- [Polygons](#) ()
- virtual [Polygons](#) ()=0
- virtual void [init](#) (const std::vector [PrivateTriangle](#) trias)=0
- virtual void [add](#) (const std::vector [PrivateTriangle](#) trias)=0
- virtual POLYLIB_STAT [import](#) (const std::map std::string, std::string fname)=0
- virtual POLYLIB_STAT [build](#) ()=0
- virtual int [triangles_num](#) ()=0
- virtual const std::vector [PrivateTriangle](#) [search](#) ([BBox](#) bbox, bool every) const =0
- virtual POLYLIB_STAT [search](#) ([BBox](#) bbox, bool every, std::vector [PrivateTriangle](#) tri_list) const =0
- virtual const std::vector [PrivateTriangle](#) [linear_search](#) ([BBox](#) bbox, bool every) const =0
- virtual POLYLIB_STAT [linear_search](#) ([BBox](#) bbox, bool every, std::vector [PrivateTriangle](#) tri_list) const =0
- std::vector [PrivateTriangle](#) [get_tri_list](#) () const
- virtual [VTree](#) [get_vtree](#) () const =0

Protected Attributes

- `std::vector` [PrivateTriangle](#) `m_tri_list`
三角形ポリゴンのリスト。

3.7.1 Detailed Description

クラス[Polygons](#) 三角形ポリゴン集合を管理する純粋仮想クラスです。

3.7.2 Constructor & Destructor Documentation

3.7.2.1 PolylibNS::Polygons::Polygons () [inline]

コンストラクタ。

3.7.2.2 virtual PolylibNS::Polygons::~Polygons () [pure virtual]

デストラクタ。

3.7.3 Member Function Documentation

3.7.3.1 virtual void PolylibNS::Polygons::add (const std::vector PrivateTriangle *trias*) [pure virtual]

三角形ポリゴンリストに引数で与えられる三角形を追加する。

Parameters

[in] *trias* 設定する三角形ポリゴンリスト。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.2 virtual POLYLIB_STAT PolylibNS::Polygons::build () [pure virtual]

Polygons クラスに含まれる全ポリゴン情報から KD 木を作成する。

Returns

POLYLIB_STAT で定義される値が返る。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.3 std::vector PrivateTriangle PolylibNS::Polygons::get_tri_list () const [inline]

三角形ポリゴンのリストを取得。

Returns

三角形ポリゴンのリスト。

3.7.3.4 virtual VTree PolylibNS::Polygons::get_vtree () const [pure virtual]

KD 木クラスを取得。

Returns

KD 木クラス。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.5 virtual POLYLIB_STAT PolylibNS::Polygons::import (const std::map<std::string, std::string> fname) [pure virtual]

STL ファイルを読み込みデータの初期化。

Parameters

[in] *fname* ファイル名とファイルフォーマットの map。

Returns

POLYLIB_STAT で定義される値が返る。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.6 virtual void PolylibNS::Polygons::init (const std::vector<PrivateTriangle> trias) [pure virtual]

引数で与えられる三角形ポリゴンリストの複製を設定する。

Parameters

[in] *trias* 設定する三角形ポリゴンリスト。

Attention

オーバーロードメソッドあり。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.7 virtual POLYLIB_STAT PolylibNS::Polygons::linear_search (BBox bbox, bool every, std::vector<PrivateTriangle> tri_list) const [pure virtual]

線形探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

[in] *bbox* 検索範囲を示す矩形領域。

[in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

[in,out] *tri_list* 抽出した三角形ポリゴンリストのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

オーバーロードメソッドあり。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.8 `virtual const std::vector PrivateTriangle PolylibNS::Polygons::linear_search (BBox bbox, bool every) const [pure virtual]`

線形探索により、指定矩形領域に含まれる三角形ポリゴンを抽出する。

Parameters

[in] *bbox* 検索範囲を示す矩形領域。

[in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

抽出したポリゴンリストのポインタ。

Attention

MPIPolylib 内でのみ利用するため、ユーザは使用しないで下さい。
オーバーロードメソッドあり。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.9 `virtual POLYLIB_STAT PolylibNS::Polygons::search (BBox bbox, bool every, std::vector PrivateTriangle tri_list) const [pure virtual]`

KD 木探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

[in] *bbox* 検索範囲を示す矩形領域。

[in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

[in,out] *tri_list* 抽出した三角形ポリゴンリストへのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

オーバーロードメソッドあり。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.10 `virtual const std::vector PrivateTriangle PolylibNS::Polygons::search (BBox bbox, bool every) const` [pure virtual]

KD 木探索により、指定矩形領域に含まれる三角形ポリゴンを抽出する。

Parameters

- [in] *bbox* 検索範囲を示す矩形領域。
- [in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

抽出したポリゴンリストのポインタ。

Attention

MPIPolylib 内でのみ利用するため、ユーザは使用しないで下さい。
オーバーロードメソッドあり。

Implemented in [PolylibNS::TriMesh](#).

3.7.3.11 `virtual int PolylibNS::Polygons::triangles_num ()` [pure virtual]

Polygons クラスで保持する三角形ポリゴンの総数を返す。

Returns

三角形ポリゴンの総数。

Implemented in [PolylibNS::TriMesh](#).

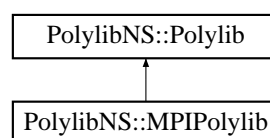
The documentation for this class was generated from the following file:

- include/polygons/Polygons.h

3.8 PolylibNS::Polylib Class Reference

```
#include include/Polylib.h
```

Inheritance diagram for PolylibNS::Polylib:



Public Member Functions

- void [set_factory](#) ([PolygonGroupFactory](#) factory=NULL)
- POLYLIB_STAT [load](#) (std::string config_name="polylib_config.xml")

- POLYLIB_STAT [save](#) (std::string p_fname, std::string format, std::string extend="", std::string rank_no="")
- POLYLIB_STAT [move](#) ([PolylibMoveParams](#) ¶ms)
- std::vector [PolygonGroup](#) [get_root_groups](#) ()
- std::vector [Triangle](#) [search_polygons](#) (std::string group_name, [Vec3f](#) min_pos, [Vec3f](#) max_pos, bool every) const
- POLYLIB_STAT [check_group_name](#) (const std::string &pg_name, const std::string &parent_path)
- [PolygonGroup](#) [create_polygon_group](#) (std::string class_name)
- void [add_pg_list](#) ([PolygonGroup](#) pg)
- void [show_group_hierarchy](#) (FILE fp=NULL)
- POLYLIB_STAT [show_group_info](#) (std::string group_name)
- unsigned int [used_memory_size](#) ()
- [PolygonGroup](#) [get_group](#) (std::string name) const

Static Public Member Functions

- static [Polylib](#) [get_instance](#) ()

Protected Member Functions

- [Polylib](#) ()
- [Polylib](#) ()
- POLYLIB_STAT [make_group_tree](#) ([PolylibConfig](#) config)
- POLYLIB_STAT [make_group_tree](#) (std::string config_contents)
- POLYLIB_STAT [load_config_file](#) (std::string contents, std::string fname="")
- POLYLIB_STAT [load_with_idfile](#) (std::string config_name)
- POLYLIB_STAT [load_polygons](#) (bool with_id_file=false)
- char [save_config_file](#) (std::string rank_no, std::string extend, std::string format)
- POLYLIB_STAT [save_polygons](#) (std::string rank_no, std::string extend, std::string format)
- POLYLIB_STAT [save_with_rankno](#) (std::string p_config_filename, int myrank, int maxrank, std::string extend, std::string stl_format)
- void [show_group_name](#) ([PolygonGroup](#) p, std::string tab, FILE fp)
- [PolygonGroup](#) [get_group](#) (int internal_id) const

Protected Attributes

- [PolygonGroupFactory](#) m_factory
PolygonGroup のファクトリークラス.
- std::vector [PolygonGroup](#) m_pg_list
ポリゴングループリスト

Static Protected Attributes

- static [Polylib](#) m_instance
自クラスのインスタンス (*singleton*)

3.8.1 Detailed Description

クラス: [Polylib](#) ポリゴンを管理する為のクラスライブラリです。

3.8.2 Constructor & Destructor Documentation

3.8.2.1 PolylibNS::Polylib::Polylib () [protected]

コンストラクタ

Attention

singleton のため、子クラス以外からの呼び出し不可とする

3.8.2.2 PolylibNS::Polylib:: ~Polylib () [protected]

デストラクタ

3.8.3 Member Function Documentation

3.8.3.1 void PolylibNS::Polylib::add_pg_list (PolygonGroup pg)

PolygonGroup の追加。 本クラスが管理している PolygonGroup のリストに PolygonGroup を追加する。

Parameters

[in] *pg* [PolygonGroup](#)

Attention

Polylib 内部で使用する関数であり、通常は利用者が用いるものではない。

3.8.3.2 POLYLIB_STAT PolylibNS::Polylib::check_group_name (const std::string & pg_name, const std::string & parent_path)

引数のグループ名が既存グループと重複しないかチェック。

Parameters

[in] *name* グループ名

[in] *path* 親グループまでのフルパス

Returns

POLYLIB_STAT で定義される値が返る。

Attention

Polylib 内部で使用する関数であり、通常は利用者が用いるものではない。

3.8.3.3 PolygonGroup PolylibNS::Polylib::create_polygon_group (std::string *class_name*)

PolygonGroup のインスタンスの生成。本クラスが管理している Factory クラスを利用して、引数で渡されたクラス名 に応じた PolygonGroup のインスタンスを生成する。

Parameters

[in] *class_name* クラス名

Returns

生成した PolygonGroup

Attention

Polylib 内部で使用する関数であり、通常は利用者が用いるもの ではない。

3.8.3.4 PolygonGroup PolylibNS::Polylib::get_group (std::string *name*) const

グループの取得。 *name* で与えられた名前の PolygonGroup を返す。

Parameters

[in] *name* グループ名

Returns

ポリゴングループクラスのポインタ。エラー時は NULL が返る。

Attention

オーバーロードメソッドあり。

3.8.3.5 PolygonGroup PolylibNS::Polylib::get_group (int *internal_id*) const [protected]

グループの取得。 *internal_id* で与えられた *m_internal_id* を持つ PolygonGroup を返す。

Parameters

[in] *internal_id* ポリゴングループ ID

Returns

ポリゴングループクラスのポインタ。エラー時は NULL が返る。

Attention

オーバーロードメソッドあり。

3.8.3.6 static Polylib PolylibNS::Polylib::get_instance () [static]

singleton の Polylib インスタンス取得。デフォルトの Factory クラスである PolygonGroupFactory を使用してインスタンス を生成する。

Returns

Polylib クラスのインスタンス。

Attention

呼び出し側で delete はできません。

Reimplemented in [PolylibNS::MPIPolylib](#).

3.8.3.7 std::vector PolygonGroup PolylibNS::Polylib::get_root_groups ()

PolygoGroup ツリーの最上位ノードの取得。

Returns

最上位ノードの vector。

Attention

返却した PolygonGroup は、削除不可。vector は要削除。

3.8.3.8 POLYLIB_STAT PolylibNS::Polylib::load (std::string config_name =)

PolygoGroup、三角形ポリゴン情報の読み込み。引数で指定された設定ファイルを読み込み、グループツリーを作成する。続いて設定ファイルで指定された STL ファイルを読み込み、KD 木を作成する。

Parameters

[in] *fname* 設定ファイル名。デフォルト値は、polylib_config.xml。

Returns

POLYLIB_STAT で定義される値が返る。

Reimplemented in [PolylibNS::MPIPolylib](#).

3.8.3.9 POLYLIB_STAT PolylibNS::Polylib::load_config_file (std::string contents, std::string fname =) [protected]

設定ファイルを読み込み、内容を contents に設定。

Parameters

[out] *contents* 設定ファイルの内容 (XML 形式)。

[in] *fname* 設定ファイル名。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

MPIPolylib クラスが MPI 環境で利用することを想定している。

3.8.3.10 POLYLIB_STAT PolylibNS::Polylib::load_polygons (bool *with_id_file* =) [protected]

STL ファイルの読み込み。 グループツリーの全リーフについて、設定されている STL ファイルから ポリゴン情報を読み込む。読み込んだ後、KD 木の生成、法線の計算、面積の 計算を行う。

Parameters

[in] *with_id_file* true ならば、三角形ポリゴン ID ファイルを読み 込んで m_id を設定する。 false ならば、STL 読み込み時に m_id を自動生成。

Returns

POLYLIB_STAT で定義される値が返る。

3.8.3.11 POLYLIB_STAT PolylibNS::Polylib::load_with_idfile (std::string *config_name*) [protected]

三角形 ID ファイルの存在が必須な load 関数。 load と同様の動作を行う。但し読み込み時には、三角形 ID ファイルが必要であり、このファイルに記述されている ID を用いて m_id を設定する。

Parameters

[in] *fname* 設定ファイル名。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

MPIPolylib クラスが MPI 環境で利用することを想定している。

3.8.3.12 POLYLIB_STAT PolylibNS::Polylib::make_group_tree (std::string *config_contents*) [protected]

引数の内容でグループ階層構造を構築。

Parameters

[in] *contents* 設定ファイルの内容 (XML 形式)。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

MPIPolylib クラスが MPI 環境で利用することを想定している。
オーバーロードメソッドあり。

3.8.3.13 POLYLIB_STAT PolylibNS::Polylib::make_group_tree (PolylibConfig *config*) [protected]

グループツリー作成。設定ファイルを管理する PolylibConfig クラスから XML タグを得て、適切な PolygonGroup を作成し、グループツリーに登録する。

Parameters

[in] *config* 設定ファイル管理クラス

Returns

POLYLIB_STAT で定義される値が返る。

Attention

オーバーロードメソッドあり。

以下のコメントは Doxygen には出力したくないのだが... config に実体を渡すと PolylibConfig のデストラクタが 2 回 (1 回目は本関数を抜けるとき、2 回目は本関数を呼び出した関数 (load_config、make_group_tree) から抜けるとき) 呼ばれてしまい、結果的に Segmentation Fault で落ちてしまう。

3.8.3.14 POLYLIB_STAT PolylibNS::Polylib::move (PolylibMoveParams & *params*)

三角形ポリゴン座標の移動。本クラスインスタンス配下の全 PolygonGroup の move メソッドが呼び出される。move メソッドは、PolygonGroup クラスを拡張したクラスに利用者が記述する。

Parameters

[in] *params* [Polylib.h](#) で宣言された移動計算パラメータセット。

Returns

POLYLIB_STAT で定義される値が返る。

Reimplemented in [PolylibNS::MPIPolylib](#).

3.8.3.15 POLYLIB_STAT PolylibNS::Polylib::save (std::string *p_fname*, std::string *format*, std::string *extend* = , std::string *rank_no* =)

PolygoGroup ツリー、三角形ポリゴン情報の保存。グループツリーの情報を設定ファイルへ出力。三角形ポリゴン情報を STL ファイルへ出力。三角形ポリゴン ID を ID ファイルへ出力。

Parameters

[out] *p_fname* 設定ファイル名。

[in] *format* TriMeshIO クラスで定義されている STL ファイルのフォーマット。

[in] *extend* ファイル名に付加する文字列。省略可。省略した場合は、付加文字列として本メソッド呼び出し時の年月日時分秒 (YYYYMMDD24hhmmss) を用いる。

[in] *rank_no* ランク番号。省略可。省略した場合は、ファイル名にランク番号は付加されない。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ファイル名命名規約は次の通り。 定義ファイル：polylib_config_ランク番号_付加文字.xml。
STL ファイル：ポリゴングループ名_ランク番号_付加文字.拡張子。 ID ファイル：ポリゴン
グループ名_ランク番号_付加文字.ID。
extend と rank_no を同時に省略すること、rank_no のみを省略する ことはできるが、extend
のみを省略することはできない。

**3.8.3.16 char PolylibNS::Polylib::save_config_file (std::string rank_no,
std::string extend, std::string format) [protected]**

設定ファイルの保存。 メモリに展開しているグループツリー情報から設定ファイルを生成する。

Parameters

- [in] *rank_no* ランク番号
- [in] *extend* ファイル名に付加する文字列
- [in] *format* TriMeshIO クラスで定義されている STL ファイルのフォー マット。

Returns

作成した設定ファイルの名称。エラー時は NULL が返る。

**3.8.3.17 POLYLIB_STAT PolylibNS::Polylib::save_polygons (std::string
rank_no, std::string extend, std::string format) [protected]**

STL ファイルと三角形ポリゴン ID の保存。 KD 木の内容を STL ファイルと ID ファイルに出力す
る。単一のリーフが複数の STL ファイルで構成されていた場合、それらはすべて一つのファイルと
して出力される。

Parameters

- [in] *rank_no* ランク番号
- [in] *extend* ファイル名に付加する文字列
- [in] *format* TriMeshIO クラスで定義されている STL ファイルのフォー マット。

Returns

POLYLIB_STAT で定義される値が返る。

**3.8.3.18 POLYLIB_STAT PolylibNS::Polylib::save_with_rankno (std::string
p_config_filename, int myrank, int maxrank, std::string extend,
std::string stl_format) [protected]**

ファイル名にランク番号をつけて保存する。ファイル名にランク番号をつけること以外は Polylibsave()
と同じ処理。

Parameters

- [in,out] *p_config_filename* 保存した設定ファイル名の返却用。
- [in] *myrank* 自ランク番号。
- [in] *maxrank* 最大ランク番号。
- [in] *extend* ファイ名に付加される文字列。
- [in] *stl_format* STL ファイルフォーマット指定。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

MPIPolylib クラスが MPI 環境で利用することを想定している。

3.8.3.19 `std::vector<Triangle> PolylibNS::Polylib::search_polygons (std::string group_name, Vec3f min_pos, Vec3f max_pos, bool every) const`

三角形ポリゴンの検索。位置ベクトル `min_pos` と `max_pos` により特定される矩形領域に含まれる、三角形ポリゴンを `group_name` で指定されたグループの下から探索する。

Parameters

- [in] *group_name* 抽出グループ名。
- [in] *min_pos* 抽出する矩形領域の最小値。
- [in] *max_pos* 抽出する矩形領域の最大値。
- [in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。false:3 頂点の一部でも検索領域と重なるものを抽出。

Returns

抽出した三角形ポリゴンの `vector`。

Attention

返却した三角形ポリゴンは、削除不可。`vector` は要削除。

3.8.3.20 `void PolylibNS::Polylib::set_factory (PolygonGroupFactory factory =)`

`PolygonGroup` クラスを生成するための `Factory` クラスを登録。本メソッドは、独自の `Factory` クラスを登録しない限り、呼び出し不要である。コンストラクタで生成した `Factory` クラスを破棄し、代わりに引数で指定された `Factory` クラスを登録する。

Parameters

- [in] *factory* `Factory` クラス。

Attention

`PolygonGroup` を拡張した場合、拡張後の `PolygonGroup` の `Factory` クラスを登録する。

3.8.3.21 void PolylibNS::Polylib::show_group_hierarchy (FILE *fp* =)

グループ階層構造を標準出力に出力。 2010.10.20 引数 FILE 追加。

Parameters

[in] *fp* 出力先ファイル。指定されて行ければ、標準出力へ出力する。

Attention

テスト用の関数であり、通常は利用者が用いるものではない。

3.8.3.22 POLYLIB_STAT PolylibNS::Polylib::show_group_info (std::string *group_name*)

グループの情報と配下の三角形ポリゴン情報を標準出力に出力。親グループ名、自身の名前、STL ファイル名、登録三角形数、3 頂点ベクトルの座標、法線ベクトルの座標、面積。

Parameters

[in] *group_name* グループ名。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

テスト用の関数であり、通常は利用者が用いるものではない。

3.8.3.23 void PolylibNS::Polylib::show_group_name (PolygonGroup *p*, std::string *tab*, FILE *fp*) [protected]

グループ名の表示。指定されたグループ以下の階層構造をツリー形式で標準出力に出力する。 2010.10.20 引数 FILE 追加。

Parameters

[in] *p* 検索の基点となる PolygonGroup のポインタ

[in] *tab* 階層の深さを示すスペース

[in] *fp* 出力先ファイル。指定されて行ければ、標準出力へ出力する。

3.8.3.24 unsigned int PolylibNS::Polylib::used_memory_size ()

Polylib が利用中の概算メモリ量を返す

Returns

利用中のメモリ量 (byte)

Reimplemented in [PolylibNS::MPIPolylib](#).

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

- include/Polylib.h

3.9 PolylibNS::PolylibCfgElem Class Reference

```
#include include/file_io/PolylibConfig.h
```

Public Member Functions

- [PolylibCfgElem](#) (const std::string name)
- [PolylibCfgElem](#) ()
- const [PolylibCfgElem](#) [first_element](#) (const std::string name="") const
- const [PolylibCfgElem](#) [next_element](#) (const [PolylibCfgElem](#) param, const std::string name="") const
- const [PolylibCfgParam](#) [first_param](#) (const std::string name="") const
- const [PolylibCfgParam](#) [next_param](#) (const [PolylibCfgParam](#) param, const std::string name="") const
- const std::string [get_name](#) () const
- void [set_elem](#) ([PolylibCfgElem](#) elem)
- void [set_param](#) ([PolylibCfgParam](#) param)

3.9.1 Detailed Description

クラス:[PolylibCfgElem](#) config ファイルの Elem 要素の管理。Elem 要素の形式は Elem name="" である。

3.9.2 Constructor & Destructor Documentation

3.9.2.1 PolylibNS::PolylibCfgElem::PolylibCfgElem (const std::string name)

コンストラクタ。

3.9.2.2 PolylibNS::PolylibCfgElem::PolylibCfgElem ()

デストラクタ。

3.9.3 Member Function Documentation

3.9.3.1 const PolylibCfgElem PolylibNS::PolylibCfgElem::first_element (const std::string name =) const

name で指定される最初の Elem 要素を返す。

Parameters

[in] *name* 要素名 (指定しない場合最初の要素)。

Returns

Elem 要素 (存在しない場合 NULL が返る)。

3.9.3.2 `const PolylibCfgParam PolylibNS::PolylibCfgElem::first_param (const std::string name =) const`

name で指定される次の Elem 要素を返す。

Parameters

[in] *name* 要素名 (指定しない場合最初の要素)。

Returns

Param 要素 (存在しない場合 NULL が返る)。

3.9.3.3 `const std::string PolylibNS::PolylibCfgElem::get_name () const [inline]`

Elem 名を返す。

Returns

Elem 名。

3.9.3.4 `const PolylibCfgElem PolylibNS::PolylibCfgElem::next_element (const PolylibCfgElem param, const std::string name =) const`

name で指定される次の Elem 要素を返す。

Parameters

[in] *param* 前の要素。

[in] *name* 要素名 (指定しない場合すぐ次の要素)。

Returns

Elem 要素 (存在しない場合 NULL が返る)。

3.9.3.5 `const PolylibCfgParam PolylibNS::PolylibCfgElem::next_param (const PolylibCfgParam param, const std::string name =) const`

name で指定される次の Param 要素を返す。

Parameters

[in] *param* 前の要素。

[in] *name* 要素名 (指定しない場合すぐ次の要素)。

Returns

Param 要素 (存在しない場合 NULL が返る)

3.9.3.6 `void PolylibNS::PolylibCfgElem::set_elem (PolylibCfgElem elem)`
`[inline]`

子要素の Elem を追加。

Parameters

[in] 子要素の *Elem*。

3.9.3.7 `void PolylibNS::PolylibCfgElem::set_param (PolylibCfgParam param)`
`[inline]`

子要素の Param を追加。

Parameters

[in] 子要素の *Param*。

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

- include/file_io/PolylibConfig.h

3.10 PolylibNS::PolylibCfgParam Class Reference

```
#include include/file_io/PolylibConfig.h
```

Public Member Functions

- [PolylibCfgParam](#) (const std::string name, const std::string type, const std::string value)
- const PolylibCfgParamType [get_data_type](#) () const
- const std::string [get_string_data](#) () const
- const int [get_int_data](#) () const
- const float [get_real_data](#) () const
- const std::string [get_name](#) () const

3.10.1 Detailed Description

クラス:[PolylibCfgParam](#) config ファイルの Param 要素の管理。 Param 要素の形式は である。

3.10.2 Constructor & Destructor Documentation

3.10.2.1 `PolylibNS::PolylibCfgParam::PolylibCfgParam (const std::string name,
const std::string type, const std::string value)`

コンストラクタ

Parameters

[in] *name* Param 名。

[in] *type* Param データ型 (STRING/INT/REAL のいずれか)。

[in] *value* Param の value 値。

3.10.3 Member Function Documentation

3.10.3.1 `const PolylibCfgParamType PolylibNS::PolylibCfgParam::get_data_type () const [inline]`

Param のデータ型。

Returns

データ型。

3.10.3.2 `const int PolylibNS::PolylibCfgParam::get_int_data () const [inline]`

INTEGER 型データ取得。

Returns

INTEGER 型データ。

3.10.3.3 `const std::string PolylibNS::PolylibCfgParam::get_name () const [inline]`

Param 名。

Returns

パラメータ名。

3.10.3.4 `const float PolylibNS::PolylibCfgParam::get_real_data () const [inline]`

FLOAT 型データ取得。

Returns

float 型のデータ。

3.10.3.5 `const std::string PolylibNS::PolylibCfgParam::get_string_data () const [inline]`

文字列データ取得。

Returns

文字列型データ。

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

- include/file_io/PolylibConfig.h

3.11 PolylibNS::PolylibConfig Class Reference

```
#include include/file_io/PolylibConfig.h
```

Public Member Functions

- [PolylibConfig](#) (std::string fname)
- [PolylibConfig](#) ()
- [PolylibConfig](#) ()
- POLYLIB_STAT [parse_xml_on_memory](#) (std::string contents)
- const [PolylibCfgElem](#) [get_root_elem](#) () const

Static Public Member Functions

- static POLYLIB_STAT [load_config_file](#) (std::string contents, std::string fname)
- static xmlNodePtr [mk_parameter_tag](#) (xmlDocPtr doc)
- static xmlNodePtr [mk_elem_tag](#) (xmlNodePtr elem)
- static POLYLIB_STAT [mk_param_tag](#) (xmlNodePtr elem, std::string name, std::string value)
- static POLYLIB_STAT [mk_param_tag](#) (xmlNodePtr elem, std::string name, int value)
- static POLYLIB_STAT [mk_param_tag](#) (xmlNodePtr elem, std::string name, double value)
- static char [save_file](#) (xmlDocPtr doc, std::string rank_no, std::string extend)

3.11.1 Detailed Description

クラス:[PolylibConfig](#) config ファイルの管理

Attention

XML の書式は V-Sphere に準拠する。

3.11.2 Constructor & Destructor Documentation

3.11.2.1 PolylibNS::PolylibConfig::PolylibConfig (std::string *fname*)

コンストラクタ。

Attention

オーバーロードメソッドあり。

3.11.2.2 PolylibNS::PolylibConfig::PolylibConfig ()

コンストラクタ。

Parameters

[in] *fname* 設定ファイル名。 設定ファイルに不備があった場合は例外 PLSTAT_NG を投げる。

Attention

オーバーロードメソッドあり。

3.11.2.3 PolylibNS::PolylibConfig:: PolylibConfig ()

デストラクタ。

3.11.3 Member Function Documentation**3.11.3.1 const PolylibCfgElem PolylibNS::PolylibConfig::get_root_elem ()
const [inline]**

ルートノード取得。

Returns

Elem タグ構造体。

**3.11.3.2 static POLYLIB_STAT PolylibNS::PolylibConfig::load_config_file (
std::string contents, std::string fname) [static]**

設定ファイル読み込み。設定ファイルを読み込み、libxml2 ライブラリを用いて構文解析した後、引数 *contents* に代入して上位に戻る。

Parameters

[out] *contents* XML 形式の文字列。

[in] *fname* 設定ファイル名。

Returns

POLYLIB_STAT で定義される値が返る。

**3.11.3.3 static xmlNodePtr PolylibNS::PolylibConfig::mk_elem_tag (
xmlNodePtr elem) [static]**

設定ファイルに出力する Elem タグを作成する。

Parameters

[in] *elem* Parameter タグ構造体、または Elem タグ構造体。

Returns

作成し Elem タグ。エラー時には NULL が返る。

3.11.3.4 static POLYLIB_STAT PolylibNS::PolylibConfig::mk_param_tag (xmlNodePtr *elem*, std::string *name*, double *value*) [static]

設定ファイルに出力する Param タグを作成する。出力する属性値は実数型。

Parameters

- [in] *elem* 親の Elem タグ。
- [in] *name* Param タグの属性名。
- [in] *value* Param タグの属性値。

Returns

作成し Param タグ。エラー時には NULL が返る。

Attention

オーバーロードメソッドあり

3.11.3.5 static POLYLIB_STAT PolylibNS::PolylibConfig::mk_param_tag (xmlNodePtr *elem*, std::string *name*, std::string *value*) [static]

設定ファイルに出力する Param タグを作成する。出力する属性値は文字列型。

Parameters

- [in] *elem* 親の Elem タグ。
- [in] *name* Param タグの属性名。
- [in] *value* Param タグの属性値。

Returns

作成し Param タグ。エラー時には NULL が返る。

Attention

オーバーロードメソッドあり。

3.11.3.6 static POLYLIB_STAT PolylibNS::PolylibConfig::mk_param_tag (xmlNodePtr *elem*, std::string *name*, int *value*) [static]

設定ファイルに出力する Param タグを作成する。出力する属性値は整数型。

Parameters

- [in] *elem* 親の Elem タグ。
- [in] *name* Param タグの属性名。
- [in] *value* Param タグの属性値。

Returns

作成し Param タグ。エラー時には NULL が返る。

Attention

オーバーロードメソッドあり。

3.11.3.7 `static xmlNodePtr PolylibNS::PolylibConfig::mk_parameter_tag (xmlDocPtr doc) [static]`

設定ファイルに出力する Parameter タグを作成する。

Parameters

[in] *doc* libxml2 ライブラリで定義している XML 文書構造体。

Returns

作成した Parameter タグ。エラー時には NULL が返る。

3.11.3.8 `POLYLIB_STAT PolylibNS::PolylibConfig::parse_xml_on_memory (std::string contents)`

引数で渡された XML 形式のデータをメモリ展開する。

Parameters

[in] *contents* XML 形式の文字列。

Returns

POLYLIB_STAT で定義される値が返る。

3.11.3.9 `static char PolylibNS::PolylibConfig::save_file (xmlDocPtr doc, std::string rank_no, std::string extend) [static]`

設定情報を XML 形式でファイルに出力する。設定ファイルのファイル名は、polylib_config_ランク番号_自由文字列.xml。

Parameters

[in] *doc* libxml2 ライブラリで定義している XML 構造体。

[in] *rank_no* ファイル名に付加するランク番号。

[in] *extend* ファイル名に付加する自由文字列。

Returns

出力ファイル名。エラー時には NULL が返る。

Attention

戻り値の char はフリー不要。

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

- include/file_io/PolylibConfig.h

3.12 PolylibNS::PolylibMoveParams Class Reference

```
#include include/Polylib.h
```

Public Attributes

- int `m_current_step`
現在の計算ステップ番号
- int `m_next_step`
移動後の計算ステップ番号
- double `m_delta_t`
1 計算ステップあたりの時間変異

3.12.1 Detailed Description

クラス:[PolylibMoveParams](#) `Polylib::move()`の引数として利用するパラメタセットクラスです。本クラスメンバ変数ではパラメタが不足する場合は、継承クラスをユーザ定義してください。

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

- `include/Polylib.h`

3.13 PolylibNS::PolylibStat2 Class Reference

Static Public Member Functions

- static std::string **String** (POLYLIB_STAT stat)

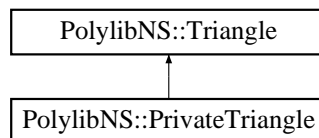
The documentation for this class was generated from the following file:

- `include/common/PolylibStat.h`

3.14 PolylibNS::PrivateTriangle Class Reference

```
#include include/polygons/Triangle.h
```

Inheritance diagram for PolylibNS::PrivateTriangle:



Public Member Functions

- `PrivateTriangle` ([Vec3f](#) vertex[3], int id)
- `PrivateTriangle` ([Vec3f](#) vertex[3], [Vec3f](#) normal, int id)
- `PrivateTriangle` ([Vec3f](#) vertex[3], [Vec3f](#) normal, float area, int id)

- [PrivateTriangle](#) ([Triangle](#) tri, int id)
- [PrivateTriangle](#) (const [PrivateTriangle](#) &tri)
- [PrivateTriangle](#) (float dim, int id)
- void [set_id](#) (int id)
- int [get_id](#) () const

Protected Attributes

- int [m_id](#)

3.14.1 Detailed Description

クラス:PrivateTriangle クラス Polylib 内のデータ保存用の基本クラスです。

3.14.2 Constructor & Destructor Documentation

3.14.2.1 PolylibNS::PrivateTriangle::PrivateTriangle ([Vec3f](#) *vertex*[3], int *id*) [inline]

コンストラクタ。

Parameters

- [in] *vertex* ポリゴンの頂点。
- [in] *id* 三角形ポリゴン ID。

3.14.2.2 PolylibNS::PrivateTriangle::PrivateTriangle ([Vec3f](#) *vertex*[3], [Vec3f](#) *normal*, int *id*) [inline]

コンストラクタ。

Parameters

- [in] *vertex* ポリゴンの頂点。
- [in] *normal* 法線。
- [in] *id* 三角形ポリゴン ID。

3.14.2.3 PolylibNS::PrivateTriangle::PrivateTriangle ([Vec3f](#) *vertex*[3], [Vec3f](#) *normal*, float *area*, int *id*) [inline]

コンストラクタ。

Parameters

- [in] *vertex* ポリゴンの頂点。
- [in] *normal* 法線。
- [in] *area* ポリゴンの面積。
- [in] *id* 三角形ポリゴン ID。

3.14.2.4 PolylibNS::PrivateTriangle::PrivateTriangle (Triangle *tri*, int *id*) [inline]

コンストラクタ。

Parameters

[in] *tri* ポリゴン。

[in] *id* 三角形ポリゴン ID。

3.14.2.5 PolylibNS::PrivateTriangle::PrivateTriangle (const PrivateTriangle & *tri*) [inline]

コンストラクタ。

Parameters

[in] *tri* ポリゴン。

3.14.2.6 PolylibNS::PrivateTriangle::PrivateTriangle (float *dim*, int *id*) [inline]

コンストラクタ。

Parameters

[in] *dim* ポリゴン頂点座標配列。

[in] *id* 三角形ポリゴン ID。

3.14.3 Member Function Documentation

3.14.3.1 int PolylibNS::PrivateTriangle::get_id () const [inline]

三角形ポリゴン ID を返す。

Returns

三角形ポリゴン ID。

3.14.3.2 void PolylibNS::PrivateTriangle::set_id (int *id*) [inline]

三角形ポリゴン ID を設定。

Parameters

[in] 三角形ポリゴン *ID*。

3.14.4 Member Data Documentation

3.14.4.1 int PolylibNS::PrivateTriangle::m_id [protected]

PolygonGroup 内で一意となる三角形ポリゴン ID。

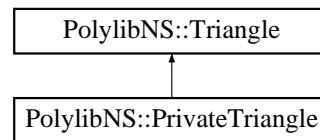
The documentation for this class was generated from the following file:

- include/polygons/Triangle.h

3.15 PolylibNS::Triangle Class Reference

```
#include include/polygons/Triangle.h
```

Inheritance diagram for PolylibNS::Triangle:



Public Member Functions

- [Triangle](#) ()
- [Triangle](#) ([Vec3f](#) vertex[3])
- [Triangle](#) ([Vec3f](#) vertex[3], [Vec3f](#) normal)
- [Triangle](#) ([Vec3f](#) vertex[3], [Vec3f](#) normal, float area)
- void [set_vertexes](#) ([Vec3f](#) vertex[3], bool calc_normal, bool calc_area)
- [Vec3f](#) [get_vertex](#) () const
- [Vec3f](#) [get_normal](#) () const
- float [get_area](#) () const

Protected Member Functions

- void [calc_normal](#) ()
- void [calc_area](#) ()

Protected Attributes

- [Vec3f](#) [m_vertex](#) [3]
三角形の頂点座標（反時計回りで並んでいる）。
- [Vec3f](#) [m_normal](#)
三角形の法線ベクトル。
- float [m_area](#)
三角形の面積。

3.15.1 Detailed Description

クラス: [Triangle](#) 入出力用インターフェースクラスであり、本ヘッダに対応する.cxx ファイルは存在しない。

3.15.2 Constructor & Destructor Documentation

3.15.2.1 PolylibNS::Triangle::Triangle () [inline]

コンストラクタ。

3.15.2.2 PolylibNS::Triangle::Triangle (Vec3f *vertex[3]*) [inline]

コンストラクタ。

Parameters

[in] *vertex* ポリゴンの頂点。

Attention

面積と法線は *vertex* を元に自動計算される。

3.15.2.3 PolylibNS::Triangle::Triangle (Vec3f *vertex[3]*, Vec3f *normal*) [inline]

コンストラクタ。

Parameters

[in] *vertex* ポリゴンの頂点。

[in] *normal* 法線。

Attention

面積は *vertex* を元に自動計算される。

3.15.2.4 PolylibNS::Triangle::Triangle (Vec3f *vertex[3]*, Vec3f *normal*, float *area*) [inline]

コンストラクタ。

Parameters

[in] *vertex* ポリゴンの頂点。

[in] *normal* 法線。

[in] *area* ポリゴンの面積。

3.15.3 Member Function Documentation

3.15.3.1 void PolylibNS::Triangle::calc_area () [inline, protected]

面積算出。

3.15.3.2 `void PolylibNS::Triangle::calc_normal () [inline, protected]`

法線ベクトル算出。

3.15.3.3 `float PolylibNS::Triangle::get_area () const [inline]`

面積を取得。

Returns

面積。

3.15.3.4 `Vec3f PolylibNS::Triangle::get_normal () const [inline]`

法線ベクトルを取得。

Returns

法線ベクトル。

3.15.3.5 `Vec3f PolylibNS::Triangle::get_vertex () const [inline]`

vertex の配列を取得。

Returns

vertex の配列。

3.15.3.6 `void PolylibNS::Triangle::set_vertexes (Vec3f vertex[3], bool calc_normal, bool calc_area) [inline]`

頂点を設定。

Parameters

[in] *calc_normal* 法線ベクトルを再計算するか？

[in] *calc_area* 面積を再計算するか？

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

- include/polygons/Triangle.h

3.16 TriangleStruct Struct Reference

```
#include <include/c_lang/CPolylib.h>
```

Public Attributes

- float **m__vertex** [9]
- float **m__normal** [3]
- float **m__area**

3.16.1 Detailed Description

三角形ポリゴン情報構造体

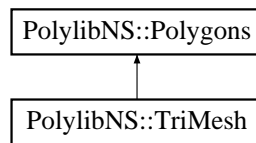
The documentation for this struct was generated from the following file:

- include/c_lang/CPolylib.h

3.17 PolylibNS::TriMesh Class Reference

```
#include include/polygons/TriMesh.h
```

Inheritance diagram for PolylibNS::TriMesh:



Public Member Functions

- [TriMesh](#) ()
- [TriMesh](#) ()
- void [init](#) (const std::vector [PrivateTriangle](#) trias)
- void [add](#) (const std::vector [PrivateTriangle](#) trias)
- POLYLIB_STAT [import](#) (const std::map std::string, std::string fmap)
- POLYLIB_STAT [build](#) ()
- int [triangles_num](#) ()
- const std::vector [PrivateTriangle](#) [search](#) ([BBox](#) bbox, bool every) const
- POLYLIB_STAT [search](#) ([BBox](#) bbox, bool every, std::vector [PrivateTriangle](#) tri_list) const
- const std::vector [PrivateTriangle](#) [linear_search](#) ([BBox](#) q_bbox, bool every) const
- POLYLIB_STAT [linear_search](#) ([BBox](#) q_bbox, bool every, std::vector [PrivateTriangle](#) tri_list) const
- [BBox](#) [get_bbox](#) () const
- [VTree](#) [get_vtree](#) () const

3.17.1 Detailed Description

クラス:[TriMesh](#) 三角形ポリゴン集合を管理するクラス (KD 木用に特化したクラス)。

3.17.2 Constructor & Destructor Documentation

3.17.2.1 PolylibNS::TriMesh::TriMesh ()

コンストラクタ。

3.17.2.2 PolylibNS::TriMesh::~TriMesh ()

デストラクタ。

3.17.3 Member Function Documentation

3.17.3.1 void PolylibNS::TriMesh::add (const std::vector PrivateTriangle *trias*) [virtual]

三角形ポリゴンリストに引数で与えられる三角形の複製を追加する。

Parameters

[in] *trias* 設定する三角形ポリゴンリスト。

Attention

`m_id` が重複するインスタンスは追加されない。
KD 木の再構築は行わない。

Implements [PolylibNS::Polygons](#).

3.17.3.2 POLYLIB_STAT PolylibNS::TriMesh::build () [virtual]

Polygons クラスに含まれる全ポリゴン情報から KD 木を作成する。

Returns

POLYLIB_STAT で定義される値が返る。

Implements [PolylibNS::Polygons](#).

3.17.3.3 BBox PolylibNS::TriMesh::get_bbox () const [inline]

TriMesh クラスが管理している BoundingBox を返す。

3.17.3.4 VTree PolylibNS::TriMesh::get_vtree () const [inline, virtual]

KD 木クラスを取得。

Returns

KD 木クラス。

Implements [PolylibNS::Polygons](#).

**3.17.3.5 POLYLIB_STAT PolylibNS::TriMesh::import (const std::map
std::string, std::string *fmap*) [virtual]**

ファイルからデータの初期化。

Parameters

[in] *fmap* ファイル名、ファイルフォーマット。

Returns

PLSTAT_OK=成功/false=失敗

Implements [PolylibNS::Polygons](#).

**3.17.3.6 void PolylibNS::TriMesh::init (const std::vector PrivateTriangle
trias) [virtual]**

TriMesh クラスで管理する三角形ポリゴンリストを初期化し、引数で与えられる三角形ポリゴンリストを設定する。 三角形ポリゴン用のメモリ領域は、Polylib 内で新たに確保される。

Parameters

[in] *trias* 設定する三角形ポリゴンリスト。

Implements [PolylibNS::Polygons](#).

**3.17.3.7 POLYLIB_STAT PolylibNS::TriMesh::linear_search (BBox *q_bbox*,
bool *every*, std::vector PrivateTriangle *tri_list*) const**
[virtual]

線形探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

[in] *q_bbox* 検索範囲を示す矩形領域。

[in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

[in,out] *tri_list* 抽出した三角形ポリゴンリストへのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

tri_list で戻される三角形ポリゴンのポインタは、Polylib 内で 保持されるアドレス値なので、ユーザは delete しないで下さい。
オーバーロードメソッドあり。

Implements [PolylibNS::Polygons](#).

3.17.3.8 `const std::vector PrivateTriangle PolylibNS::TriMesh::linear_search (BBox q_bbox, bool every) const [virtual]`

線形探索により、指定矩形領域に含まれる三角形ポリゴンを抽出する。

Parameters

- [in] *q_bbox* 検索範囲を示す矩形領域。
- [in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

抽出したポリゴンリストのポインタ。

Attention

三角形ポリゴンのメモリ領域は新たに Polylib 内で確保される。
MPIPolylib 内での利用が目的なので、ユーザは使用しないこと。

Implements [PolylibNS::Polygons](#).

3.17.3.9 `POLYLIB_STAT PolylibNS::TriMesh::search (BBox bbox, bool every, std::vector PrivateTriangle tri_list) const [virtual]`

KD 木探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

- [in] *bbox* 検索範囲を示す矩形領域
- [in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。
- [in,out] *tri_list* 抽出した三角形ポリゴンリストへのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

tri_list で戻される三角形ポリゴンのポインタは、Polylib 内で 保持されるアドレス値なので、ユーザは delete しないで下さい。
オーバーロードメソッドあり。

Implements [PolylibNS::Polygons](#).

3.17.3.10 `const std::vector PrivateTriangle PolylibNS::TriMesh::search (BBox bbox, bool every) const [virtual]`

KD 木探索により、指定矩形領域に含まれる三角形ポリゴンを抽出する。

Parameters

- [in] *bbox* 検索範囲を示す矩形領域。

[in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

抽出したポリゴンリストのポインタ。

Attention

三角形ポリゴンのメモリ領域は新たに Polylib 内で確保される。
MPIPolylib 内での利用が目的なので、ユーザは使用しないこと。
オーバーロードメソッドあり。

Implements [PolylibNS::Polygons](#).

3.17.3.11 int PolylibNS::TriMesh::triangles_num () [virtual]

TriMesh クラスが管理している三角形ポリゴン数を返す。

Implements [PolylibNS::Polygons](#).

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

- include/polygons/TriMesh.h

3.18 PolylibNS::TriMeshIO Class Reference

```
#include include/file_io/TriMeshIO.h
```

Static Public Member Functions

- static POLYLIB_STAT [load](#) (std::vector [PrivateTriangle](#) tri_list, const std::map<std::string, std::string> &fmap)
- static POLYLIB_STAT [save](#) (std::vector [PrivateTriangle](#) tri_list, std::string fname, std::string fmt="")
- static std::string [input_file_format](#) (const std::string &filename)

Static Public Attributes

- static const std::string [FMT_STL_A](#)
- static const std::string [FMT_STL_AA](#)
- static const std::string [FMT_STL_B](#)
- static const std::string [FMT_STL_BB](#)
- static const std::string [DEFAULT_FMT](#)

3.18.1 Detailed Description

クラス:[TriMeshIO](#) 三角形ポリゴン入出力管理。

3.18.2 Member Function Documentation

3.18.2.1 static std::string PolylibNS::TriMeshIO::input_file_format (const std::string & *filename*) [static]

ファイル名を元に入力ファイルのフォーマットを取得する。

Parameters

[in] *filename* 入力ファイル名。

Returns

判定したファイルフォーマット。

Attention

ファイル拡張子が"stl"の場合、ファイルを読み込んで判定する。

3.18.2.2 static POLYLIB_STAT PolylibNS::TriMeshIO::load (std::vector PrivateTriangle *tri_list*, const std::map std::string, std::string & *fmap*) [static]

STL ファイルを読み込み、*tri_list* にセットする。

Parameters

[in,out] *tri_list* 三角形ポリゴンリストの領域。

[in] *fmap* ファイル名、ファイルフォーマットのセット。

Returns

POLYLIB_STAT で定義される値が返る。

3.18.2.3 static POLYLIB_STAT PolylibNS::TriMeshIO::save (std::vector PrivateTriangle *tri_list*, std::string *fname*, std::string *fmt* =) [static]

tri_list の内容を STL 形式でファイルへ保存。

Parameters

[in] *tri_list* 三角形ポリゴンのリスト (出力内容)。

[in] *fname* ファイル名。

[in] *fmt* ファイルフォーマット。

Returns

POLYLIB_STAT で定義される値が返る。

3.18.3 Member Data Documentation

3.18.3.1 `const std::string PolylibNS::TriMeshIO::FMT_STL_A` [static]

STL ファイルのフォーマット種別

Attention

STL ファイルの拡張子とは異なるので注意すること。

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

- `include/file_io/TriMeshIO.h`

3.19 PolylibNS::Vec2 T Class Template Reference

```
#include <include/common/Vec2.h>
```

Public Member Functions

- **Vec2** (T v=0)
- **Vec2** (T _x, T _y)
- **Vec2** (const T v[2])
- **Vec2** T & **assign** (T _x, T _y)
- **operator T** ()
- **operator const T** () const
- T **ptr** ()
- const T **ptr** () const
- T & **operator[]** (int i)
- const T & **operator[]** (int i) const
- **Vec2** T & **operator** += (const **Vec2** T &v)
- **Vec2** T & **operator** -= (const **Vec2** T &v)
- **Vec2** T & **operator** = (const **Vec2** T &v)
- **Vec2** T & **operator** /= (const **Vec2** T &v)
- **Vec2** T & **operator** = (T s)
- **Vec2** T & **operator** /= (T s)
- **Vec2** T **operator** + (const **Vec2** T &v) const
- **Vec2** T **operator** - (const **Vec2** T &v) const
- **Vec2** T **operator** * (const **Vec2** T &v) const
- **Vec2** T **operator** / (const **Vec2** T &v) const
- **Vec2** T **operator** (T s) const
- **Vec2** T **operator** / (T s) const
- **Vec2** T **operator** - () const
- bool **operator** == (const **Vec2** T &v) const
- bool **operator** != (const **Vec2** T &v) const
- float **lengthSquared** () const
- float **length** () const
- **Vec2** T & **normalize** ()
- **Vec2** T & **normalize** (float len)
- float **average** () const

Static Public Member Functions

- static [Vec2](#) T **xaxis** ()
- static [Vec2](#) T **yaxis** ()

Public Attributes

- T **x**
- T **y**

3.19.1 Detailed Description

template **typename** T **class** PolylibNS::Vec2 T

クラス:Vec2 T

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

- include/common/Vec2.h

3.20 PolylibNS::Vec3 T Class Template Reference

```
#include <include/common/Vec3.h>
```

Public Member Functions

- **Vec3** (T v=0)
- **Vec3** (T _x, T _y, T _z)
- **Vec3** (const T v[3])
- [Vec3](#) T & **assign** (T _x, T _y, T _z)
- **operator** T ()
- **operator** const T () const
- T **ptr** ()
- const T **ptr** () const
- T & **operator**[] (const AxisEnum &axis)
- const T & **operator**[] (const AxisEnum &axis) const
- [Vec3](#) T & **operator**+= (const [Vec3](#) T &v)
- [Vec3](#) T & **operator**-= (const [Vec3](#) T &v)
- [Vec3](#) T & **operator** = (const [Vec3](#) T &v)
- [Vec3](#) T & **operator**/= (const [Vec3](#) T &v)
- [Vec3](#) T & **operator** = (T s)
- [Vec3](#) T & **operator**/= (T s)
- [Vec3](#) T **operator**+ (const [Vec3](#) T &v) const
- [Vec3](#) T **operator**- (const [Vec3](#) T &v) const
- [Vec3](#) T **operator** (const [Vec3](#) T &v) const
- [Vec3](#) T **operator**/ (const [Vec3](#) T &v) const
- [Vec3](#) T **operator** (T s) const
- [Vec3](#) T **operator**/ (T s) const
- [Vec3](#) T **operator**- () const

- bool **operator==** (const [Vec3](#) T &v) const
- bool **operator!=** (const [Vec3](#) T &v) const
- float **lengthSquared** () const
- float **length** () const
- [Vec3](#) T & **normalize** ()
- [Vec3](#) T & **normalize** (float len)
- float **average** () const

Static Public Member Functions

- static [Vec3](#) T **xaxis** ()
- static [Vec3](#) T **yaxis** ()
- static [Vec3](#) T **zaxis** ()

Public Attributes

- T t [3]

3.20.1 Detailed Description

template typename T class PolylibNS::Vec3 T

クラス:Vec3 T

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

- include/common/Vec3.h

3.21 PolylibNS::VElement Class Reference

#include include/polygons/VTree.h

Public Member Functions

- [VElement](#) ([PrivateTriangle](#) tri)
- [PrivateTriangle](#) get_triangle ()
- [Vec3f](#) get_pos () const
- [BBox](#) get_bbox () const

3.21.1 Detailed Description

クラス:[VElement](#) KD 木構造の要素クラスです。

3.21.2 Constructor & Destructor Documentation

3.21.2.1 PolylibNS::VElement::VElement (PrivateTriangle *tri*)

コンストラクタ。

Parameters

[in] *tri* ポリゴン情報のポインタ。

Attention

ポインタを格納するが、参照のみ。delete は行わない。

3.21.3 Member Function Documentation

3.21.3.1 BBox PolylibNS::VElement::get_bbox () const [inline]

Bounding box of this triangle

3.21.3.2 Vec3f PolylibNS::VElement::get_pos () const [inline]

Center position of bbox on triangle.

3.21.3.3 PrivateTriangle PolylibNS::VElement::get_triangle () [inline]

triangle.

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

- include/polygons/VTree.h

3.22 PolylibNS::VNode Class Reference

```
#include include/polygons/VTree.h
```

Public Member Functions

- [VNode](#) ()
- [VNode](#) ()
- void [split](#) (const int &max_elem)
- bool [is_leaf](#) () const
- [BBox](#) [get_bbox](#) () const
- void [set_bbox](#) (const [BBox](#) &bbbox)
- [BBox](#) [get_bbox_search](#) () const
- void [set_bbox_search](#) (const [VElement](#) p)
- [VNode](#) [get_left](#) ()
- [VNode](#) [get_right](#) ()
- [AxisEnum](#) [get_axis](#) () const
- void [set_axis](#) (const [AxisEnum](#) axis)

- `std::vector<VElement> & get_vlist ()`
- `void set_element (VElement elm)`
- `int get_elements_num () const`

3.22.1 Detailed Description

VNode クラス KD 木構造のノードクラスです。

3.22.2 Constructor & Destructor Documentation

3.22.2.1 PolylibNS::VNode::VNode ()

コンストラクタ。

3.22.2.2 PolylibNS::VNode::~VNode ()

デストラクタ。

3.22.3 Member Function Documentation

3.22.3.1 AxisEnum PolylibNS::VNode::get_axis () const [inline]

Axis を取得。

Returns

axis。

3.22.3.2 BBox PolylibNS::VNode::get_bbox () const [inline]

BBox の値を取得。

Returns

bbox。

3.22.3.3 BBox PolylibNS::VNode::get_bbox_search () const [inline]

検索用 BBox を取得。

Returns

検索用 bbox。

3.22.3.4 `int PolylibNS::VNode::get_elements_num () const [inline]`

ノードが所持する要素の数を取得。

Returns

要素数。

3.22.3.5 `VNode PolylibNS::VNode::get_left () [inline]`

左の Node を取得。

Returns

左の Node。

3.22.3.6 `VNode PolylibNS::VNode::get_right () [inline]`

右の Node を取得。

Returns

右の Node。

3.22.3.7 `std::vector VElement & PolylibNS::VNode::get_vlist () [inline]`

要素のリストを取得。

Returns

要素のリスト。

3.22.3.8 `bool PolylibNS::VNode::is_leaf () const [inline]`

ノードがリーフかどうかの判定結果。

Returns

true=リーフ/false=リーフでない。

3.22.3.9 `void PolylibNS::VNode::set_axis (const AxisEnum axis) [inline]`

Axis を設定。

Parameters

[in] *axis*。

3.22.3.10 `void PolylibNS::VNode::set_bbox (const BBox & bbox) [inline]`

BBox の値を設定。

Parameters

[in] *bbox*。

3.22.3.11 `void PolylibNS::VNode::set_bbox_search (const VElement p) [inline]`

このノードの Bounding Box を引数で与えられる要素を含めた大きさに変更する。

Parameters

[in] *p* 要素。

3.22.3.12 `void PolylibNS::VNode::set_element (VElement elm) [inline]`

木の要素を設定。

Parameters

[in] *elm*。

3.22.3.13 `void PolylibNS::VNode::split (const int & max_elem)`

ノードを 2 つの子供ノードに分割する。

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

- include/polygons/VTree.h

3.23 PolylibNS::VTree Class Reference

```
#include include/polygons/VTree.h
```

Public Member Functions

- [VTree](#) (int *max_elem*, const [BBox](#) *bbox*, std::vector [PrivateTriangle](#) *tri_list*)
- [VTree](#) ()
- void [destroy](#) ()
- std::vector [PrivateTriangle](#) [search](#) ([BBox](#) *bbox*, bool *every*) const
- POLYLIB_STAT [search](#) ([BBox](#) *bbox*, bool *every*, std::vector [PrivateTriangle](#) *tri_list*) const
- unsigned int [memory_size](#) ()

3.23.1 Detailed Description

クラス: [VTree](#) リーフを三角形ポリゴンとする KD 木クラスです。

3.23.2 Constructor & Destructor Documentation

3.23.2.1 PolylibNS::VTree::VTree (int *max_elem*, const BBox *bbox*, std::vector PrivateTriangle *tri_list*)

コンストラクタ。

Parameters

- [in] *max_elem* 最大要素数。
- [in] *bbox* VTree の box 範囲。
- [in] *tri_list* 木構造の元になるポリゴンのリスト。

3.23.2.2 PolylibNS::VTree:: VTree ()

デストラクタ。

3.23.3 Member Function Documentation

3.23.3.1 void PolylibNS::VTree::destroy ()

木構造を消去する。

3.23.3.2 unsigned int PolylibNS::VTree::memory_size ()

KD 木クラスが利用しているメモリ量を返す。

Returns

利用中のメモリ量 (byte)

3.23.3.3 POLYLIB_STAT PolylibNS::VTree::search (BBox *bbox*, bool *every*, std::vector PrivateTriangle *tri_list*) const

KD 木探索により、指定矩形領域に含まれるポリゴンを抽出する。

Parameters

- [in] *bbox* 検索範囲を示す矩形領域。
- [in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。
- [in,out] *tri_list* 抽出した三角形ポリゴンリストへのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

オーバーロードメソッドあり。

3.23.3.4 `std::vector PrivateTriangle PolylibNS::VTree::search (BBox bbox, bool every) const`

KD 木探索により、指定矩形領域に含まれる三角形ポリゴンを抽出する。

Parameters

[in] *bbox* 検索範囲を示す矩形領域。

[in] *every* true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

抽出したポリゴンリストのポインタ。

Attention

MPIPolylib 用のメソッドなので、ユーザは利用しないで下さい。
オーバーロードメソッドあり。

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

- include/polygons/VTree.h

Index

- MPIPolylib
 - PolylibNS::MPIPolylib, [9](#)
- PolygonGroup
 - PolylibNS::PolygonGroup, [18](#)
- PolygonGroupFactory
 - PolylibNS::PolygonGroupFactory, [29](#)
- Polygons
 - PolylibNS::Polygons, [31](#)
- Polylib
 - PolylibNS::Polylib, [36](#)
- PolylibCfgElem
 - PolylibNS::PolylibCfgElem, [44](#)
- PolylibConfig
 - PolylibNS::PolylibConfig, [49](#)
- TriMesh
 - PolylibNS::TriMesh, [59](#)
- VNode
 - PolylibNS::VNode, [68](#)
- VTree
 - PolylibNS::VTree, [71](#)
- acq_file_name
 - PolylibNS::PolygonGroup, [19](#)
- acq_fullpath
 - PolylibNS::PolygonGroup, [19](#)
- add
 - PolylibNS::Polygons, [31](#)
 - PolylibNS::TriMesh, [59](#)
- add_children
 - PolylibNS::PolygonGroup, [19](#)
- add_pg_list
 - PolylibNS::Polylib, [36](#)
- add_triangles
 - PolylibNS::PolygonGroup, [19](#)
- ATT_NAME_CLASS
 - PolylibNS::PolygonGroup, [29](#)
- broadcast_config
 - PolylibNS::MPIPolylib, [10](#)
- broadcast_config_from_rank0
 - PolylibNS::MPIPolylib, [10](#)
- build
 - PolylibNS::Polygons, [31](#)
 - PolylibNS::TriMesh, [59](#)
- build_group_tree
 - PolylibNS::PolygonGroup, [19](#)
- build_polygon_tree
 - PolylibNS::PolygonGroup, [20](#)
- calc_area
 - PolylibNS::Triangle, [56](#)
- calc_normal
 - PolylibNS::Triangle, [56](#)
- check_group_name
 - PolylibNS::Polylib, [36](#)
- check_leaped
 - PolylibNS::PolygonGroup, [20](#)
- contain
 - PolylibNS::BBox, [6](#)
- create_instance
 - PolylibNS::PolygonGroupFactory, [30](#)
- create_polygon_group
 - PolylibNS::Polylib, [36](#)
- crossed
 - PolylibNS::BBox, [6](#)
- destroy
 - PolylibNS::VTree, [71](#)
- erase_outbounded_polygons
 - PolylibNS::MPIPolylib, [10](#)
- first_element
 - PolylibNS::PolylibCfgElem, [44](#)
- first_param
 - PolylibNS::PolylibCfgElem, [44](#)
- FMT_STL_A
 - PolylibNS::TriMeshIO, [64](#)
- gather_polygons
 - PolylibNS::MPIPolylib, [10](#)
- get_area
 - PolylibNS::Triangle, [57](#)
- get_axis
 - PolylibNS::VNode, [68](#)
- get_bbox
 - PolylibNS::TriMesh, [59](#)
 - PolylibNS::VElement, [67](#)
 - PolylibNS::VNode, [68](#)
- get_bbox_search
 - PolylibNS::VNode, [68](#)

- get_children
 - PolylibNS::PolygonGroup, 20
- get_class_name
 - PolylibNS::PolygonGroup, 20
- get_data_type
 - PolylibNS::PolylibCfgParam, 47
- get_elements_num
 - PolylibNS::VNode, 68
- get_file_name
 - PolylibNS::PolygonGroup, 21
- get_group
 - PolylibNS::Polylib, 37
- get_id
 - PolylibNS::PolygonGroup, 21
 - PolylibNS::PrivateTriangle, 54
- get_instance
 - PolylibNS::MPIPolylib, 10
 - PolylibNS::Polylib, 37
- get_int_data
 - PolylibNS::PolylibCfgParam, 47
- get_internal_id
 - PolylibNS::PolygonGroup, 21
- get_left
 - PolylibNS::VNode, 69
- get_movable
 - PolylibNS::PolygonGroup, 21
- get_myproc
 - PolylibNS::MPIPolylib, 10
- get_name
 - PolylibNS::PolygonGroup, 21
 - PolylibNS::PolylibCfgElem, 45
 - PolylibNS::PolylibCfgParam, 47
- get_normal
 - PolylibNS::Triangle, 57
- get_num_oftrias_before_move
 - PolylibNS::PolygonGroup, 21
- get_parent
 - PolylibNS::PolygonGroup, 22
- get_parent_path
 - PolylibNS::PolygonGroup, 22
- get_pos
 - PolylibNS::VElement, 67
- get_proc
 - PolylibNS::MPIPolylib, 11
- get_real_data
 - PolylibNS::PolylibCfgParam, 47
- get_right
 - PolylibNS::VNode, 69
- get_root_elem
 - PolylibNS::PolylibConfig, 49
- get_root_groups
 - PolylibNS::Polylib, 38
- get_string_data
 - PolylibNS::PolylibCfgParam, 47
- get_tri_list
 - PolylibNS::Polygons, 31
- get_triangle
 - PolylibNS::VElement, 67
- get_triangles
 - PolylibNS::PolygonGroup, 22
- get_vertex
 - PolylibNS::Triangle, 57
- get_vlist
 - PolylibNS::VNode, 69
- get_vtree
 - PolylibNS::PolygonGroup, 22
 - PolylibNS::Polygons, 31
 - PolylibNS::TriMesh, 59
- getCrossedRegion
 - PolylibNS::BBox, 6
- getFace
 - PolylibNS::BBox, 6
- getSide
 - PolylibNS::BBox, 6
- import
 - PolylibNS::Polygons, 32
 - PolylibNS::TriMesh, 59
- init
 - PolylibNS::PolygonGroup, 22
 - PolylibNS::Polygons, 32
 - PolylibNS::TriMesh, 60
- init_check_leaped
 - PolylibNS::PolygonGroup, 23
- init_parallel_info
 - PolylibNS::MPIPolylib, 11
- input_file_format
 - PolylibNS::TriMeshIO, 63
- is_far
 - PolylibNS::PolygonGroup, 23
- is_leaf
 - PolylibNS::VNode, 69
- linear_search
 - PolylibNS::PolygonGroup, 23, 24
 - PolylibNS::Polygons, 32, 33
 - PolylibNS::TriMesh, 60
- load
 - PolylibNS::MPIPolylib, 11
 - PolylibNS::Polylib, 38
 - PolylibNS::TriMeshIO, 63
- load_config_file
 - PolylibNS::Polylib, 38
 - PolylibNS::PolylibConfig, 49
- load_id_file
 - PolylibNS::PolygonGroup, 24
- load_parallel
 - PolylibNS::MPIPolylib, 11

- load_polygons
 - PolylibNS::Polylib, 39
- load_rank0
 - PolylibNS::MPIPolylib, 12
- load_stl_file
 - PolylibNS::PolygonGroup, 24
- load_with_idfile
 - PolylibNS::Polylib, 39
- m_id
 - PolylibNS::PrivateTriangle, 55
- make_group_tree
 - PolylibNS::Polylib, 39
- memory_size
 - PolylibNS::VTree, 71
- migrate
 - PolylibNS::MPIPolylib, 12
- mk_basic_tag
 - PolylibNS::PolygonGroup, 24
- mk_elem_tag
 - PolylibNS::PolylibConfig, 49
- mk_param_tag
 - PolylibNS::PolygonGroup, 25
 - PolylibNS::PolylibConfig, 49, 50
- mk_parameter_tag
 - PolylibNS::PolylibConfig, 50
- move
 - PolylibNS::MPIPolylib, 12
 - PolylibNS::PolygonGroup, 25
 - PolylibNS::Polylib, 40
- MPIPolylib
 - PolylibNS::MPIPolylib, 9
- next_element
 - PolylibNS::PolylibCfgElem, 45
- next_param
 - PolylibNS::PolylibCfgElem, 45
- pack_num_trias
 - PolylibNS::MPIPolylib, 12
- pack_tria_ids
 - PolylibNS::MPIPolylib, 13
- pack_trias
 - PolylibNS::MPIPolylib, 13
- parse_xml_on_memory
 - PolylibNS::PolylibConfig, 51
- PolygonGroup
 - PolylibNS::PolygonGroup, 18
- PolygonGroupFactory
 - PolylibNS::PolygonGroupFactory, 29
- Polygons
 - PolylibNS::Polygons, 31
- Polylib
 - PolylibNS::Polylib, 36
- PolylibCfgElem
 - PolylibNS::PolylibCfgElem, 44
- PolylibCfgParam
 - PolylibNS::PolylibCfgParam, 46
- PolylibConfig
 - PolylibNS::PolylibConfig, 48
- PolylibNS::BBox, 5
 - contain, 6
 - crossed, 6
 - getCrossedRegion, 6
 - getFace, 6
 - getSide, 6
 - vec3to2, 7
- PolylibNS::CalcAreaInfo, 7
- PolylibNS::MPIPolylib, 8
 - MPIPolylib, 9
 - broadcast_config, 10
 - broadcast_config_from_rank0, 10
 - erase_outbounded_polygons, 10
 - gather_polygons, 10
 - get_instance, 10
 - get_myproc, 10
 - get_proc, 11
 - init_parallel_info, 11
 - load, 11
 - load_parallel, 11
 - load_rank0, 12
 - migrate, 12
 - move, 12
 - MPIPolylib, 9
 - pack_num_trias, 12
 - pack_tria_ids, 13
 - pack_trias, 13
 - receive_polygons_from_rank0, 13
 - save, 13
 - save_parallel, 14
 - save_rank0, 14
 - select_excluded_trias, 14
 - send_polygons_to_all, 15
 - send_polygons_to_rank0, 15
 - show_group_name, 15
 - used_memory_size, 15
- PolylibNS::ParallelInfo, 15
- PolylibNS::PolygonGroup, 16
 - PolygonGroup, 18
 - acq_file_name, 19
 - acq_fullpath, 19
 - add_children, 19
 - add_triangles, 19
 - ATT_NAME_CLASS, 29
 - build_group_tree, 19
 - build_polygon_tree, 20
 - check_leaped, 20
 - get_children, 20

- get_class_name, 20
- get_file_name, 21
- get_id, 21
- get_internal_id, 21
- get_movable, 21
- get_name, 21
- get_num_oftrias_before_move, 21
- get_parent, 22
- get_parent_path, 22
- get_triangles, 22
- get_vtree, 22
- init, 22
- init_check_leaped, 23
- is_far, 23
- linear_search, 23, 24
- load_id_file, 24
- load_stl_file, 24
- mk_basic_tag, 24
- mk_param_tag, 25
- move, 25
- PolygonGroup, 18
- rebuild_polygons, 25
- save_id_file, 26
- save_stl_file, 26
- search, 26, 27
- search_outbounded, 27
- set_children, 27
- set_file_name, 27
- set_name, 28
- set_parent, 28
- set_parent_path, 28
- setup_attribute, 28
- show_group_info, 28
- PolylibNS::PolygonGroupFactory, 29
 - PolygonGroupFactory, 29
 - create_instance, 30
 - PolygonGroupFactory, 29
- PolylibNS::Polygons, 30
 - Polygons, 31
 - add, 31
 - build, 31
 - get_tri_list, 31
 - get_vtree, 31
 - import, 32
 - init, 32
 - linear_search, 32, 33
 - Polygons, 31
 - search, 33
 - triangles_num, 34
- PolylibNS::Polylib, 34
 - Polylib, 36
 - add_pg_list, 36
 - check_group_name, 36
 - create_polygon_group, 36
 - get_group, 37
 - get_instance, 37
 - get_root_groups, 38
 - load, 38
 - load_config_file, 38
 - load_polygons, 39
 - load_with_idfile, 39
 - make_group_tree, 39
 - move, 40
 - Polylib, 36
 - save, 40
 - save_config_file, 41
 - save_polygons, 41
 - save_with_rankno, 41
 - search_polygons, 42
 - set_factory, 42
 - show_group_hierarchy, 42
 - show_group_info, 43
 - show_group_name, 43
 - used_memory_size, 43
- PolylibNS::PolylibCfgElem, 44
 - PolylibCfgElem, 44
 - first_element, 44
 - first_param, 44
 - get_name, 45
 - next_element, 45
 - next_param, 45
 - PolylibCfgElem, 44
 - set_elem, 45
 - set_param, 46
- PolylibNS::PolylibCfgParam, 46
 - get_data_type, 47
 - get_int_data, 47
 - get_name, 47
 - get_real_data, 47
 - get_string_data, 47
 - PolylibCfgParam, 46
- PolylibNS::PolylibConfig, 48
 - PolylibConfig, 49
 - get_root_elem, 49
 - load_config_file, 49
 - mk_elem_tag, 49
 - mk_param_tag, 49, 50
 - mk_parameter_tag, 50
 - parse_xml_on_memory, 51
 - save_file, 51
- PolylibNS::PolylibMoveParams, 51
- PolylibNS::PolylibStat2, 52
- PolylibNS::PrivateTriangle, 52
 - get_id, 54
 - m_id, 55
 - PrivateTriangle, 53, 54
 - set_id, 54
- PolylibNS::Triangle, 55

- calc_area, [56](#)
- calc_normal, [56](#)
- get_area, [57](#)
- get_normal, [57](#)
- get_vertex, [57](#)
- set_vertexes, [57](#)
- Triangle, [56](#)
- PolylibNS::TriMesh, [58](#)
 - TriMesh, [59](#)
 - add, [59](#)
 - build, [59](#)
 - get_bbox, [59](#)
 - get_vtree, [59](#)
 - import, [59](#)
 - init, [60](#)
 - linear_search, [60](#)
 - search, [61](#)
 - triangles_num, [62](#)
 - TriMesh, [59](#)
- PolylibNS::TriMeshIO, [62](#)
 - FMT_STL_A, [64](#)
 - input_file_format, [63](#)
 - load, [63](#)
 - save, [63](#)
- PolylibNS::Vec2, [64](#)
- PolylibNS::Vec3, [65](#)
- PolylibNS::VElement, [66](#)
 - get_bbox, [67](#)
 - get_pos, [67](#)
 - get_triangle, [67](#)
 - VElement, [67](#)
- PolylibNS::VNode, [67](#)
 - VNode, [68](#)
 - get_axis, [68](#)
 - get_bbox, [68](#)
 - get_bbox_search, [68](#)
 - get_elements_num, [68](#)
 - get_left, [69](#)
 - get_right, [69](#)
 - get_vlist, [69](#)
 - is_leaf, [69](#)
 - set_axis, [69](#)
 - set_bbox, [69](#)
 - set_bbox_search, [70](#)
 - set_element, [70](#)
 - split, [70](#)
 - VNode, [68](#)
- PolylibNS::VTree, [70](#)
 - VTree, [71](#)
 - destroy, [71](#)
 - memory_size, [71](#)
 - search, [71](#), [72](#)
 - VTree, [71](#)
- PrivateTriangle
 - PolylibNS::PrivateTriangle, [53](#), [54](#)
- rebuild_polygons
 - PolylibNS::PolygonGroup, [25](#)
- receive_polygons_from_rank0
 - PolylibNS::MPIPolylib, [13](#)
- save
 - PolylibNS::MPIPolylib, [13](#)
 - PolylibNS::Polylib, [40](#)
 - PolylibNS::TriMeshIO, [63](#)
- save_config_file
 - PolylibNS::Polylib, [41](#)
- save_file
 - PolylibNS::PolylibConfig, [51](#)
- save_id_file
 - PolylibNS::PolygonGroup, [26](#)
- save_parallel
 - PolylibNS::MPIPolylib, [14](#)
- save_polygons
 - PolylibNS::Polylib, [41](#)
- save_rank0
 - PolylibNS::MPIPolylib, [14](#)
- save_stl_file
 - PolylibNS::PolygonGroup, [26](#)
- save_with_rankno
 - PolylibNS::Polylib, [41](#)
- search
 - PolylibNS::PolygonGroup, [26](#), [27](#)
 - PolylibNS::Polygons, [33](#)
 - PolylibNS::TriMesh, [61](#)
 - PolylibNS::VTree, [71](#), [72](#)
- search_outbounded
 - PolylibNS::PolygonGroup, [27](#)
- search_polygons
 - PolylibNS::Polylib, [42](#)
- select_excluded_tris
 - PolylibNS::MPIPolylib, [14](#)
- send_polygons_to_all
 - PolylibNS::MPIPolylib, [15](#)
- send_polygons_to_rank0
 - PolylibNS::MPIPolylib, [15](#)
- set_axis
 - PolylibNS::VNode, [69](#)
- set_bbox
 - PolylibNS::VNode, [69](#)
- set_bbox_search
 - PolylibNS::VNode, [70](#)
- set_children
 - PolylibNS::PolygonGroup, [27](#)
- set_elem
 - PolylibNS::PolylibCfgElem, [45](#)
- set_element
 - PolylibNS::VNode, [70](#)

- set_factory
 - PolylibNS::Polylib, [42](#)
- set_file_name
 - PolylibNS::PolygonGroup, [27](#)
- set_id
 - PolylibNS::PrivateTriangle, [54](#)
- set_name
 - PolylibNS::PolygonGroup, [28](#)
- set_param
 - PolylibNS::PolylibCfgElem, [46](#)
- set_parent
 - PolylibNS::PolygonGroup, [28](#)
- set_parent_path
 - PolylibNS::PolygonGroup, [28](#)
- set_vertexes
 - PolylibNS::Triangle, [57](#)
- setup_attribute
 - PolylibNS::PolygonGroup, [28](#)
- show_group_hierarchy
 - PolylibNS::Polylib, [42](#)
- show_group_info
 - PolylibNS::PolygonGroup, [28](#)
 - PolylibNS::Polylib, [43](#)
- show_group_name
 - PolylibNS::MPIPolylib, [15](#)
 - PolylibNS::Polylib, [43](#)
- split
 - PolylibNS::VNode, [70](#)
- Triangle
 - PolylibNS::Triangle, [56](#)
- triangles_num
 - PolylibNS::Polygons, [34](#)
 - PolylibNS::TriMesh, [62](#)
- TriangleStruct, [57](#)
- TriMesh
 - PolylibNS::TriMesh, [59](#)
- used_memory_size
 - PolylibNS::MPIPolylib, [15](#)
 - PolylibNS::Polylib, [43](#)
- vec3to2
 - PolylibNS::BBox, [7](#)
- VElement
 - PolylibNS::VElement, [67](#)
- VNode
 - PolylibNS::VNode, [68](#)
- VTree
 - PolylibNS::VTree, [71](#)