

2.0.2

Generated by Doxygen 1.7.1

Wed Nov 10 2010 19:50:45



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class Hierarchy . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
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>	14
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>	15
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>	16
3.4	PolylibNS::ParallelInfo Struct Reference	16
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	19
3.5.2.1	PolygonGroup	19
3.5.2.2	PolygonGroup	19
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>	20
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>	21

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

3.5.4.1	ATT_NAME_CLASS	29
3.6	PolylibNS::PolygonGroupFactory Class Reference	29
3.6.1	Detailed Description	30
3.6.2	Constructor & Destructor Documentation	30
3.6.2.1	PolygonGroupFactory	30
3.6.2.2	PolygonGroupFactory	30
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	32
3.7.3.3	get_tri_list	32
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	33
3.7.3.8	linear_search	33
3.7.3.9	search	34
3.7.3.10	search	34
3.7.3.11	triangles_num	34
3.8	PolylibNS::Polylib Class Reference	35
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	37
3.8.3.3	create_polygon_group	37
3.8.3.4	get_group	37
3.8.3.5	get_group	38

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

3.10.3.1	<a href="#">get_data_type</a>	47
3.10.3.2	<a href="#">get_int_data</a>	47
3.10.3.3	<a href="#">get_name</a>	47
3.10.3.4	<a href="#">get_real_data</a>	47
3.10.3.5	<a href="#">get_string_data</a>	47
3.11	<a href="#">PolylibNS::PolylibConfig Class Reference</a>	48
3.11.1	<a href="#">Detailed Description</a>	48
3.11.2	<a href="#">Constructor &amp; Destructor Documentation</a>	48
3.11.2.1	<a href="#">PolylibConfig</a>	48
3.11.2.2	<a href="#">PolylibConfig</a>	49
3.11.2.3	<a href="#">PolylibConfig</a>	49
3.11.3	<a href="#">Member Function Documentation</a>	49
3.11.3.1	<a href="#">get_root_elem</a>	49
3.11.3.2	<a href="#">load_config_file</a>	49
3.11.3.3	<a href="#">mk_elem_tag</a>	49
3.11.3.4	<a href="#">mk_param_tag</a>	50
3.11.3.5	<a href="#">mk_param_tag</a>	50
3.11.3.6	<a href="#">mk_param_tag</a>	50
3.11.3.7	<a href="#">mk_parameter_tag</a>	51
3.11.3.8	<a href="#">parse_xml_on_memory</a>	51
3.11.3.9	<a href="#">save_file</a>	51
3.12	<a href="#">PolylibNS::PolylibMoveParams Class Reference</a>	51
3.12.1	<a href="#">Detailed Description</a>	52
3.13	<a href="#">PolylibNS::PolylibStat2 Class Reference</a>	52
3.13.1	<a href="#">Detailed Description</a>	52
3.13.2	<a href="#">Member Function Documentation</a>	52
3.13.2.1	<a href="#">String</a>	52
3.14	<a href="#">PolylibNS::PrivateTriangle Class Reference</a>	53
3.14.1	<a href="#">Detailed Description</a>	53
3.14.2	<a href="#">Constructor &amp; Destructor Documentation</a>	53
3.14.2.1	<a href="#">PrivateTriangle</a>	53
3.14.2.2	<a href="#">PrivateTriangle</a>	53
3.14.2.3	<a href="#">PrivateTriangle</a>	54
3.14.2.4	<a href="#">PrivateTriangle</a>	54
3.14.2.5	<a href="#">PrivateTriangle</a>	54
3.14.2.6	<a href="#">PrivateTriangle</a>	54



3.14.3	Member Function Documentation	55
3.14.3.1	get_id	55
3.14.3.2	set_id	55
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	57
3.15.3	Member Function Documentation	57
3.15.3.1	calc_area	57
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	58
3.16	TriangleStruct Struct Reference	58
3.16.1	Detailed Description	58
3.17	PolylibNS::TriMesh Class Reference	58
3.17.1	Detailed Description	59
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	60
3.17.3.3	get_bbox	60
3.17.3.4	get_vtree	60
3.17.3.5	import	60
3.17.3.6	init	60
3.17.3.7	linear_search	61
3.17.3.8	linear_search	61
3.17.3.9	search	61

3.17.3.10	search	62
3.17.3.11	triangles_num	62
3.18	PolylibNS::TriMeshIO Class Reference	62
3.18.1	Detailed Description	63
3.18.2	Member Function Documentation	63
3.18.2.1	input_file_format	63
3.18.2.2	load	64
3.18.2.3	save	64
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	66
3.20.1	Detailed Description	67
3.21	PolylibNS::VElement Class Reference	67
3.21.1	Detailed Description	67
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	68
3.22	PolylibNS::VNode Class Reference	68
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	69
3.22.3.1	get_axis	69
3.22.3.2	get_bbox	69
3.22.3.3	get_bbox_search	69
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	70
3.22.3.8	is_leaf	70

---

3.22.3.9	set_axis	70
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	71
3.23	PolylibNS::VTree Class Reference	71
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	72
3.23.3.3	search	72
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 . . . . .	16
PolylibNS::PolygonGroup . . . . .	16
PolylibNS::PolygonGroupFactory . . . . .	29
PolylibNS::Polygons . . . . .	30
PolylibNS::TriMesh . . . . .	58
PolylibNS::Polylib . . . . .	35
PolylibNS::MPIPolylib . . . . .	8
PolylibNS::PolylibCfgElem . . . . .	44
PolylibNS::PolylibCfgParam . . . . .	46
PolylibNS::PolylibConfig . . . . .	48
PolylibNS::PolylibMoveParams . . . . .	51
PolylibNS::PolylibStat2 . . . . .	52
PolylibNS::Triangle . . . . .	55
PolylibNS::PrivateTriangle . . . . .	53
TriangleStruct . . . . .	58
PolylibNS::TriMeshIO . . . . .	62
PolylibNS::Vec2 T . . . . .	64
PolylibNS::Vec3 T . . . . .	66
PolylibNS::VElement . . . . .	67
PolylibNS::VNode . . . . .	68
PolylibNS::VTree . . . . .	71



# Chapter 2

## Class Index

### 2.1 Class List

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

<a href="#">PolylibNS::BBox</a>	5
<a href="#">PolylibNS::CalcAreaInfo</a>	7
<a href="#">PolylibNS::MPIPolylib</a>	8
<a href="#">PolylibNS::ParallelInfo</a>	16
<a href="#">PolylibNS::PolygonGroup</a>	16
<a href="#">PolylibNS::PolygonGroupFactory</a>	29
<a href="#">PolylibNS::Polygons</a>	30
<a href="#">PolylibNS::Polylib</a>	35
<a href="#">PolylibNS::PolylibCfgElem</a>	44
<a href="#">PolylibNS::PolylibCfgParam</a>	46
<a href="#">PolylibNS::PolylibConfig</a>	48
<a href="#">PolylibNS::PolylibMoveParams</a>	51
<a href="#">PolylibNS::PolylibStat2</a>	52
<a href="#">PolylibNS::PrivateTriangle</a>	53
<a href="#">PolylibNS::Triangle</a>	55
<a href="#">TriangleStruct</a>	58
<a href="#">PolylibNS::TriMesh</a>	58
<a href="#">PolylibNS::TriMeshIO</a>	62
<a href="#">PolylibNS::Vec2 T</a>	64
<a href="#">PolylibNS::Vec3 T</a>	66
<a href="#">PolylibNS::VElement</a>	67
<a href="#">PolylibNS::VNode</a>	68
<a href="#">PolylibNS::VTree</a>	71





# 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。

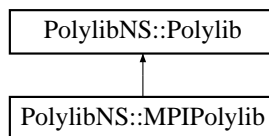
The documentation for this struct was generated from the following file:

- `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="", ID\_FORMAT id\_format=ID\_BIN)
- 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="", ID\_FORMAT id\_format=ID\_BIN)
- POLYLIB\_STAT [move](#) ([PolylibMoveParams](#) &params)
- 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 *gcsiz*e[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] *gcsiz*e 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。

**Attention**

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* = , ID\_FORMAT *id\_format* = )

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。

#### Attention

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。
- [in] *id\_format* 三角形 ID ファイルの入力形式。

#### 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。

#### Attention

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 = , ID\_FORMAT id\_format = )

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 (YYYYMMDD24hmmss) を用いる。

[in] *id\_format* 三角形 ID ファイルの出力形式。

#### 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。

#### Attention

出力引数 `p_config_filename` の返却値は rank0 でのみ有効

### 3.3.3.21 POLYLIB\_STAT PolylibNS::MPIPolylib::select\_excluded\_triangles ( 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 [PolylibCfElem](#) elem)
- POLYLIB\_STAT [build\\_polygon\\_tree](#) ()
- POLYLIB\_STAT [load\\_stl\\_file](#) ()
- POLYLIB\_STAT [load\\_id\\_file](#) (ID\_FORMAT id\_format)
- 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, ID\_FORMAT id\_format)
- virtual POLYLIB\_STAT [mk\\_param\\_tag](#) (xmlNodePtr elem, std::string rank\_no, std::string extend, std::string format)
- virtual POLYLIB\_STAT [move](#) ([PolylibMoveParams](#) &params)
- 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](#) [m\\_trias\\_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] *p* 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。

#### Attention

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_oftrias_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。

#### Attention

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 ( ID_FORMAT id_format )`

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。

#### Parameters

[in] *id\_format* 三角形 ID ファイルの入力形式。

#### 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] *params* [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*, ID\_FORMAT *id\_format* )

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。

[in] *id\_format* 三角形 ID ファイルの出力形式。

#### 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] *p* 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] *ppath* 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。

### 3.5.3.43 virtual std::string PolylibNS::PolygonGroup::whoami ( ) [inline, virtual]

クラス名を取得。

**Returns**

クラス名。

**Attention**

継承するクラスのクラス名取得関数 get\_class\_name() を呼び出す。

## 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。

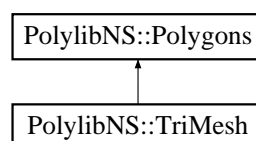
The documentation for this class was generated from the following file:

- 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](#).

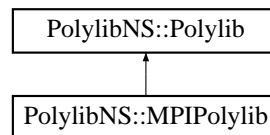
The documentation for this class was generated from the following file:

- 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\_config\_name, std::string stl\_format, std::string extend="")
- POLYLIB\_STAT [move](#) ([PolylibMoveParams](#) &params)
- std::vector [PolygonGroup](#) [get\\_root\\_groups](#) () const
- 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, ID\_FORMAT id\_format)
- POLYLIB\_STAT [load\\_polygons](#) (bool with\_id\_file, ID\_FORMAT id\_format)
- char [save\\_config\\_file](#) (std::string rank\_no, std::string extend, std::string format)
- POLYLIB\_STAT [save\\_with\\_rankno](#) (std::string p\_config\_name, int myrank, int maxrank, std::string extend, std::string stl\_format, ID\_FORMAT id\_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] *pg\_name* グループ名
- [in] *parent\_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 ( 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。

### 3.8.3.5 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。

### 3.8.3.6 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 ( ) const

PolygonGroup 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 = )

PolygonGroup、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] *config\_name* 設定ファイル名。

**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, ID\_FORMAT id\_format ) [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。

[in] *id\_format* 三角形 ID ファイルの入力形式。

**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, ID\_FORMAT id\_format ) [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] *config\_name* 設定ファイル名。

[in] *id\_format* 三角形 ID ファイルの入力形式。

**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。

**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] *config\_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\_config\_name, std::string stl\_format, std::string extend = )

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。

### Parameters

[out] *p\_config\_name* 保存した設定ファイル名の返却用。

[in] *stl\_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。

### 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。

#### 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\_with\_rankno ( std::string p\_config\_name, int myrank, int maxrank, std::string extend, std::string stl\_format, ID\_FORMAT id\_format ) [protected]

PolygoGroup ツリー、三角形ポリゴン情報の保存。 グループツリー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。ID 情報を ID ファイルへ出力。ファイル名にランク番号を付加する。

**Parameters**

- [out] *p\_config\_name* 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。
- [in] *id\_format* 三角形 ID ファイルの出力形式。

**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。  
 MPIPolylib クラスが MPI 環境で利用することを想定している。

### 3.8.3.18 `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。

### 3.8.3.19 `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。

### 3.8.3.20 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。

### 3.8.3.21 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。

### 3.8.3.22 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。

### 3.8.3.23 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](#)

#### Attention

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] *elem* 追加する Elem 要素。

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

子要素の Param を追加。

#### Parameters

[in] *param* 追加する Param 要素。

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](#)

#### Attention

定義ファイルの Param 要素の管理。 Param 要素の形式は " Param name="" dtype="" value="" "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。

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す。

#### 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

```
#include include/common/PolylibStat.h
```

### Static Public Member Functions

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

### 3.13.1 Detailed Description

`PolylibStat` 文字列出力用クラス

### 3.13.2 Member Function Documentation

**3.13.2.1** static std::string `PolylibNS::PolylibStat2::String` ( `POLYLIB_STAT stat`  
 ) [`inline`, `static`]

`PolylibStat` 文字列出力。

#### Parameters

[in] *stat* `PolylibStat` 値。

#### Returns

`PolylibStat` 値を文字列化したもの。

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

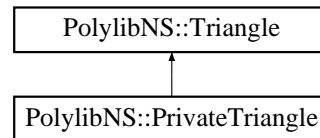
- `include/common/PolylibStat.h`



## 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] *id* 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。

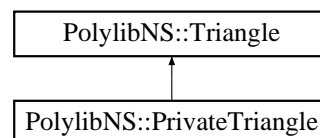
The documentation for this class was generated from the following file:

- 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] *vertex* 三角形の3頂点。
- [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]  
3頂点座標
- float [m\\_normal](#) [3]  
法線ベクトル
- float [m\\_area](#)  
面積

### 3.16.1 Detailed Description

三角形ポリゴン情報構造体

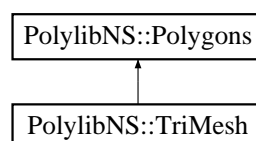
The documentation for this struct was generated from the following file:

- 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](#)  
アスキーファイル
- static const std::string [FMT\\_STL\\_AA](#)  
アスキーファイル
- static const std::string [FMT\\_STL\\_B](#)  
バイナリファイル
- static const std::string [FMT\\_STL\\_BB](#)  
バイナリファイル
- static const std::string [DEFAULT\\_FMT](#)  
*TrimeshIO.cpp* で定義している値.

### 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]
```

アスキーファイル

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](#) &bbox)
- [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, [19](#)
- PolygonGroupFactory
  - PolylibNS::PolygonGroupFactory, [30](#)
- 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, [57](#)
- calc\_normal
  - PolylibNS::Triangle, [57](#)
- 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, [37](#)
- 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, [69](#)
- get\_bbox
  - PolylibNS::TriMesh, [60](#)
  - PolylibNS::VElement, [67](#)
  - PolylibNS::VNode, [69](#)
- get\_bbox\_search
  - PolylibNS::VNode, [69](#)

- get\_children
  - PolylibNS::PolygonGroup, 20
- get\_class\_name
  - PolylibNS::PolygonGroup, 21
- get\_data\_type
  - PolylibNS::PolylibCfgParam, 47
- get\_elements\_num
  - PolylibNS::VNode, 69
- get\_file\_name
  - PolylibNS::PolygonGroup, 21
- get\_group
  - PolylibNS::Polylib, 37
- get\_id
  - PolylibNS::PolygonGroup, 21
  - PolylibNS::PrivateTriangle, 55
- get\_instance
  - PolylibNS::MPIPolylib, 10
  - PolylibNS::Polylib, 38
- 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, 22
  - PolylibNS::PolylibCfgElem, 45
  - PolylibNS::PolylibCfgParam, 47
- get\_normal
  - PolylibNS::Triangle, 57
- get\_num\_oftrias\_before\_move
  - PolylibNS::PolygonGroup, 22
- 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, 32
- 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, 32
  - PolylibNS::TriMesh, 60
- getCrossedRegion
  - PolylibNS::BBox, 6
- getFace
  - PolylibNS::BBox, 6
- getSide
  - PolylibNS::BBox, 6
- import
  - PolylibNS::Polygons, 32
  - PolylibNS::TriMesh, 60
- init
  - PolylibNS::PolygonGroup, 23
  - 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, 70
- linear\_search
  - PolylibNS::PolygonGroup, 23, 24
  - PolylibNS::Polygons, 33
  - PolylibNS::TriMesh, 60, 61
- load
  - PolylibNS::MPIPolylib, 11
  - PolylibNS::Polylib, 38
  - PolylibNS::TriMeshIO, 63
- load\_config\_file
  - PolylibNS::Polylib, 39
  - 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, 40
- memory\_size
  - PolylibNS::VTree, 71
- migrate
  - PolylibNS::MPIPolylib, 12
- mk\_basic\_tag
  - PolylibNS::PolygonGroup, 25
- 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, 13
- pack\_tria\_ids
  - PolylibNS::MPIPolylib, 13
- pack\_trias
  - PolylibNS::MPIPolylib, 13
- parse\_xml\_on\_memory
  - PolylibNS::PolylibConfig, 51
- PolygonGroup
  - PolylibNS::PolygonGroup, 19
- PolygonGroupFactory
  - PolylibNS::PolygonGroupFactory, 30
- 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, 13
  - pack\_tria\_ids, 13
  - pack\_trias, 13
  - receive\_polygons\_from\_rank0, 13
  - save, 14
  - save\_parallel, 14
  - save\_rank0, 14
  - select\_excluded\_trias, 15
  - send\_polygons\_to\_all, 15
  - send\_polygons\_to\_rank0, 15
  - show\_group\_name, 15
  - used\_memory\_size, 16
- PolylibNS::ParallelInfo, 16
- PolylibNS::PolygonGroup, 16
  - PolygonGroup, 19
  - 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, 21
- get\_file\_name, 21
- get\_id, 21
- get\_internal\_id, 21
- get\_movable, 21
- get\_name, 22
- get\_num\_oftrias\_before\_move, 22
- get\_parent, 22
- get\_parent\_path, 22
- get\_triangles, 22
- get\_vtree, 22
- init, 23
- init\_check\_leaped, 23
- is\_far, 23
- linear\_search, 23, 24
- load\_id\_file, 24
- load\_stl\_file, 24
- mk\_basic\_tag, 25
- mk\_param\_tag, 25
- move, 25
- PolygonGroup, 19
- rebuild\_polygons, 26
- save\_id\_file, 26
- save\_stl\_file, 26
- search, 26, 27
- search\_outbounded, 27
- set\_children, 27
- set\_file\_name, 28
- set\_name, 28
- set\_parent, 28
- set\_parent\_path, 28
- setup\_attribute, 28
- show\_group\_info, 29
- whoami, 29
- PolylibNS::PolygonGroupFactory, 29
  - PolygonGroupFactory, 30
  - create\_instance, 30
  - PolygonGroupFactory, 30
- PolylibNS::Polygons, 30
  - Polygons, 31
  - add, 31
  - build, 31
  - get\_tri\_list, 32
  - get\_vtree, 32
  - import, 32
  - init, 32
  - linear\_search, 33
  - Polygons, 31
  - search, 33, 34
  - triangles\_num, 34
- PolylibNS::Polylib, 35
  - Polylib, 36
  - add\_pg\_list, 36
  - check\_group\_name, 36
  - create\_polygon\_group, 37
  - get\_group, 37
  - get\_instance, 38
  - get\_root\_groups, 38
  - load, 38
  - load\_config\_file, 39
  - load\_polygons, 39
  - load\_with\_idfile, 39
  - make\_group\_tree, 40
  - move, 40
  - Polylib, 36
  - save, 41
  - save\_config\_file, 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
  - PolylibConfig, 48
  - save\_file, 51
- PolylibNS::PolylibMoveParams, 51
- PolylibNS::PolylibStat2, 52
  - String, 52
- PolylibNS::PrivateTriangle, 53
  - get\_id, 55
  - m\_id, 55
  - PrivateTriangle, 53, 54



- set\_id, [55](#)
- PolylibNS::Triangle, [55](#)
  - calc\_area, [57](#)
  - calc\_normal, [57](#)
  - get\_area, [57](#)
  - get\_normal, [57](#)
  - get\_vertex, [57](#)
  - set\_vertexes, [57](#)
  - Triangle, [56](#), [57](#)
- PolylibNS::TriMesh, [58](#)
  - TriMesh, [59](#)
  - add, [59](#)
  - build, [59](#)
  - get\_bbox, [60](#)
  - get\_vtree, [60](#)
  - import, [60](#)
  - init, [60](#)
  - linear\_search, [60](#), [61](#)
  - search, [61](#), [62](#)
  - triangles\_num, [62](#)
  - TriMesh, [59](#)
- PolylibNS::TriMeshIO, [62](#)
  - FMT\_STL\_A, [64](#)
  - input\_file\_format, [63](#)
  - load, [63](#)
  - save, [64](#)
- PolylibNS::Vec2, [64](#)
- PolylibNS::Vec3, [66](#)
- PolylibNS::VElement, [67](#)
  - get\_bbox, [67](#)
  - get\_pos, [67](#)
  - get\_triangle, [67](#)
  - VElement, [67](#)
- PolylibNS::VNode, [68](#)
  - VNode, [68](#)
  - get\_axis, [69](#)
  - get\_bbox, [69](#)
  - get\_bbox\_search, [69](#)
  - get\_elements\_num, [69](#)
  - get\_left, [69](#)
  - get\_right, [69](#)
  - get\_vlist, [69](#)
  - is\_leaf, [70](#)
  - set\_axis, [70](#)
  - set\_bbox, [70](#)
  - set\_bbox\_search, [70](#)
  - set\_element, [70](#)
  - split, [70](#)
  - VNode, [68](#)
- PolylibNS::VTree, [71](#)
  - VTree, [71](#)
  - destroy, [71](#)
  - memory\_size, [71](#)
  - search, [72](#)
- VTree, [71](#)
- PrivateTriangle
  - PolylibNS::PrivateTriangle, [53](#), [54](#)
- rebuild\_polygons
  - PolylibNS::PolygonGroup, [26](#)
- receive\_polygons\_from\_rank0
  - PolylibNS::MPIPolylib, [13](#)
- save
  - PolylibNS::MPIPolylib, [14](#)
  - PolylibNS::Polylib, [41](#)
  - PolylibNS::TriMeshIO, [64](#)
- save\_config\_file
  - PolylibNS::Polylib, [41](#)
- save\_file
  - PolylibNS::PolylibConfig, [51](#)
- save\_id\_file
  - PolylibNS::PolygonGroup, [26](#)
- save\_parallel
  - PolylibNS::MPIPolylib, [14](#)
- 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](#), [34](#)
  - PolylibNS::TriMesh, [61](#), [62](#)
  - PolylibNS::VTree, [72](#)
- search\_outbounded
  - PolylibNS::PolygonGroup, [27](#)
- search\_polygons
  - PolylibNS::Polylib, [42](#)
- select\_excluded\_tris
  - PolylibNS::MPIPolylib, [15](#)
- send\_polygons\_to\_all
  - PolylibNS::MPIPolylib, [15](#)
- send\_polygons\_to\_rank0
  - PolylibNS::MPIPolylib, [15](#)
- set\_axis
  - PolylibNS::VNode, [70](#)
- set\_bbox
  - PolylibNS::VNode, [70](#)
- 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, [28](#)
- set\_id
  - PolylibNS::PrivateTriangle, [55](#)
- 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, [29](#)
  - PolylibNS::Polylib, [43](#)
- show\_group\_name
  - PolylibNS::MPIIPolylib, [15](#)
  - PolylibNS::Polylib, [43](#)
- split
  - PolylibNS::VNode, [70](#)
- String
  - PolylibNS::PolylibStat2, [52](#)
- Triangle
  - PolylibNS::Triangle, [56](#), [57](#)
- triangles\_num
  - PolylibNS::Polygons, [34](#)
  - PolylibNS::TriMesh, [62](#)
- TriangleStruct, [58](#)
- TriMesh
  - PolylibNS::TriMesh, [59](#)
- used\_memory\_size
  - PolylibNS::MPIIPolylib, [16](#)
  - PolylibNS::Polylib, [43](#)
- vec3to2
  - PolylibNS::BBox, [7](#)
- VElement
  - PolylibNS::VElement, [67](#)
- VNode
  - PolylibNS::VNode, [68](#)
- VTree
  - PolylibNS::VTree, [71](#)
- whoami
  - PolylibNS::PolygonGroup, [29](#)