itklib

AUTHOR 9SQ
Version 1.0
CREATEDATE 2004-05-11

Table of Contents

Table of contents

libgraphics Hierarchical Index

libgraphics Class Hierarchy

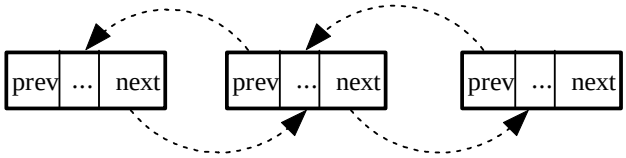
This inheritance list is sorted roughly, but not completely, alphabetically:

libgraphics Data Structure Index

libgraphics Data Structures

Here are the data structures with brief descriptions:

PRIMITIVE의 구조




```
  typedef enum{
            PRIMARY_CRV_T = 1,
            COMPLEX_CRV_T = 2,
            RULE_CRV_T       = 3
}PRIMITIVE_E;
typedef struct tagPrimtiveAttr{
            unsigned short u:1;                 // exist user data?
            unsigned short f:1;                 // is supportting format?
            unsigned short d:2;                 // dimension(2 - 2D , 3 - 3D)
            unsigned short h:1;                 // hole or solid
            unsigned short reversed:1;
            unsigned short continuous:1;        // for SQ_POINTS.
            unsigned short closed:1;            // for SQ_POLYLINE
            unsigned short del:1;               // is deleted?
            unsigned short r:3;
            unsigned short type:4;              //
} PRIMITIVEATTR_T,* PPRIMITIVEATTR_T;
```

primitive의 속성을 나타낸다.

```
typedef struct tagSQHeader
            char                id;         // id of element
            char                layer;      // level or layer element is on
            PRIMITIVEATTR_T attr;           //
            unsigned long   handle;
            SQVOLUME        volume;         /// primitive's volume
            struct{
                        short   ltype;          // line type
                        short   lweight;        // line weight
                        unsigned char lcolor[3];// line color(r,g,b)
                        char    r;              // reserved
                        long    lscale;         // arrow
            }display;
            char            desc[16];       // description
            unsigned long   tsize,esize;
}SQHEADER,* PSQHEADER;
```

primitive의header부분이다.

```
typedef struct tagSQPrimitive
  SQHEADER hdr;
  union{
            POINT_T         point;
            SQVERTEX        vertex;
```

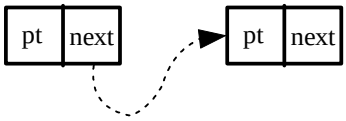
```
            LINE_T          line;
            PPOINT_TNODE    polyline;
            SQSHAPE         shape;
            TEXT_T          text;
            SQREFERENCE     reference;
            list<SQVERTEX>* plstVertices;
            vector<POINT_T>*pvtPoints;

            //--> curve
            CIRCLE_T        circle;
            ARC_T           arc;
            ELLIPSE_T       ellipse;
            SQCURVE         curve;
            SQINTCURVE      intcurve;
            //<--

            CONE_T          cone;

            //--> surface
            PSQSURFACE       pSurface;
            PSQPLANESURFACE  pPlaneSurface;
            PSQCONESURFACE   pConeSurface;
            PSQSPHERESURFACE pSphereSurface;
            PSQTORUSSURFACE  pTorusSurface;
            //<--

            PLAYER_T        player;
            struct tagSQPrimitive*  pPrimitive;
            char*                   pData;
        }body;
        void* pLinkageData;
        struct tagSQPrimitive *prev,*next; /// pointer to prev and next
    }SQPRIMITIVE,* PSQPRIMITIVE;
```

primitive의body 부분은 union으로 묶여져 있다.

tagSQVertex Struct Reference

```
typedef struct tagSQVertex{
        struct tagCode{
                unsigned char visit   : 1;  /// already visited?
                unsigned char moveto: 1;  /// moveto or lineto
                unsigned char r       : 6;  /// reserved
        }code;
        double x,y,z;       /// value
}SQVERTEX,* PSQVERTEX;
```

tagSQPointNode Struct Reference




```
typedef struct tagSQPointNode{
        POINT_T pt;
        struct tagSQPointNode* next; /// next pointnode
}SQPOINTNODE,* PPOINT_TNODE;
```

tagSQShape Struct Reference

```
typedef struct tagSQShape{
        unsigned short    fcolor; /// fill color
        struct _list{
                POINT_T point;
                struct _list* next;
        }* list;
}SQSHAPE,* PSQSHAPE;
```

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

- sqstruct.h

tagCone Struct Reference

```
typedef struct tagCone{
        VECTOR_T vecAxis;       /// axis of cone
        POINT_T    ptOrigin[2];
        double     nRadius[2];
}CONE_T,* PCONE_T;
```

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

- cone.h

tagLayer Struct Reference

```
typedef struct tagLayer{
        char szName[32]; /// layer name buffer
}LAYER_T,* PLAYER_T;
```

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

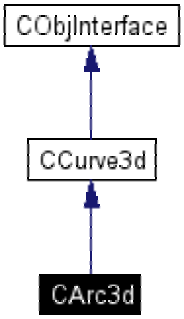
- layer.h

libgraphics Data Structure Documentation

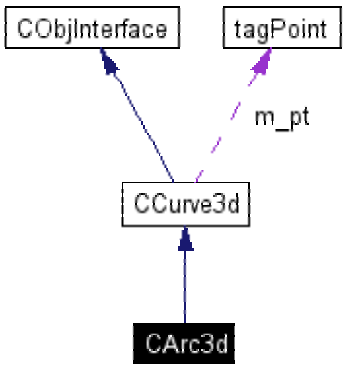
CArc3d Class Reference

002CArc3d#include <arc3d.h>

Inheritance diagram for CArc3d:



Collaboration diagram for CArc3d:



Public Member Functions

- CArc3d ()
- CArc3d (const ARC_T &arc)
- CArc3d (const CArc3d &arc)
- CArc3d (POINT_T ptOrigin, POINT_T ptStart, const double &nSweepAng)
- virtual ~CArc3d ()

- void CreateSegments ()
  The CArc3d::CreateSegments function

user must have a copy of points of segments.

because of another curve will overwrite the point value.

- PPOINT_T Revolve (VECTOR_T vecAxis, double nAngle, int nParts)
  주어진 회전축(vecAxis)에 대하여 nAngle만큼 회전한다.

Static Public Member Functions

- void segments (POINT_T ptEDGE[], const POINT_T &ptOrigin, const POINT_T &ptStart, const VECTOR_T &vecNorm, const double nSweep, const int &nSegments)
- POINT_T on (const POINT_T &ptOrigin, const VECTOR_T vecNorm, const double nRadius, const double &nAngle)
- POINT_T OnPoint (const POINT_T ptOrigin, const double nRadius, const VECTOR_T vecAxis, const double &nAngle)
  ptOrigin,nRadius,vecAxis를 가진 arc에서 nAngle에 위치한 점을 구한다.

Data Fields

- int m_nSegments

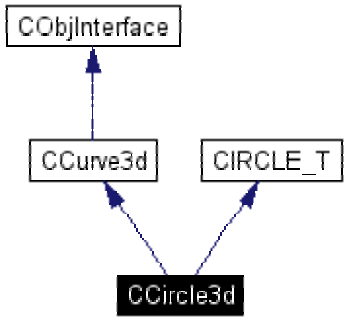
The documentation for this class was generated from the following files:

- arc3d.h
- arc3d.cpp

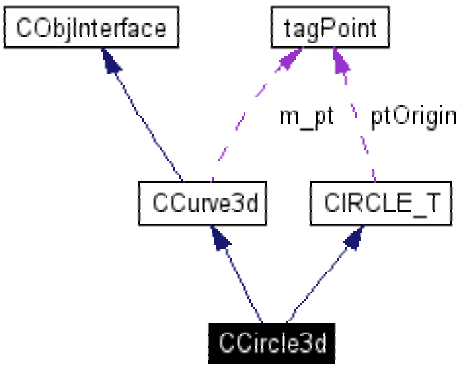
CCircle3d Class Reference

002CCircle3d#include <circle3d.h>

Inheritance diagram for CCircle3d:



Collaboration diagram for CCircle3d:



Public Member Functions

- CCircle3d ()
- CCircle3d (const CIRCLE_T &circle)
- CCircle3d (const CCircle3d &circle)
- CCircle3d (double x, double y, double z, double radius)
  x,y,z는 원의 중점 , radius = 원의 반경

- virtual ~CCircle3d ()

- POINT_T GetOrigin ()
  원의 중점을 구한다.

- double GetRadius ()
  원의 반경을 구한다.
- void SetOrigin (double x, double y, double z)
  원의 중점을 정한다.
- void SetOrigin (POINT_T ptCenter)
  원의 중점을 정한다.
- void SetRadius (const double &radius)
  원의 반경을 정한다.
- void CreateSegments ()
  원을 구성하는 점들을 구한다. 구한 점들은 Ccurve3d::m_pt에 저장된다.
- PPOINT_T Sweep (VECTOR_T vecAxis, double nAngle, int nParts)
  현재 사용되지 않음.

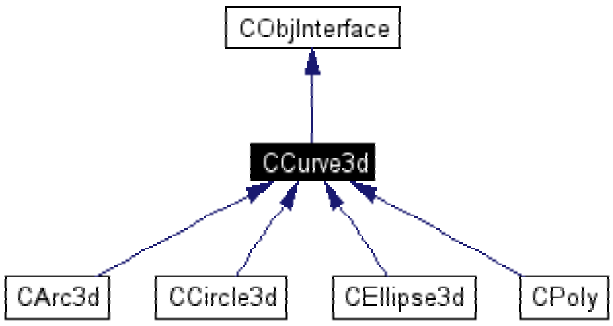
The documentation for this class was generated from the following files:

- circle3d.h
- circle3d.cpp

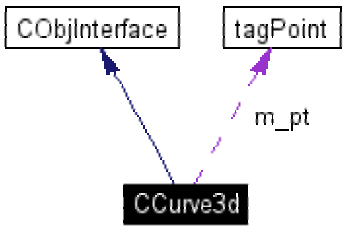
CCurve3d Class Reference

002CCurve3d#include <curve3d.h>

Inheritance diagram for CCurve3d:



Collaboration diagram for CCurve3d:



Public Member Functions

- CCurve3d ()
- virtual ~CCurve3d ()

- void SetSegments (const int nSegments)
  curve의 segment 개수를 정한다.
- int GetNumOfPoints ()
  curve를 이루는 점들의 개수를 구한다.
- virtual PPOINT_T GetFacets ()
  가상함수로써 상속받는 클래스에서 구현하여야 한다.
- virtual void CreateSegments ()=0
- virtual PPOINT_T Sweep (VECTOR_T vecAxis, double nAngle, int nParts)

현재 사용되지 않음.

Static Public Member Functions

- PPOINT_T Revolve (const VECTOR_T &vecAxis, int nSize, PPOINT_T pptOriginal, const double &nAngle, const int &nParts)
  주어진 회전축(vecAxis)에 대하여 점들을 nAngle만큼 회전하여 이루는 면을 구한다.
- const int GetNumOfSegments ()
  curve를 이루는 curve의 segment 개수를 구한다.

- POINT_T & pt (const int &nIndex)
  Return Ccuve3d::m_pt[nIndex];

Static Protected Attributes

- int NUM_OF_SEGMENTS /// *curve를 이루는 segment의 개수*
- POINT_T m_pt [MAX_CURVE_POINTS]

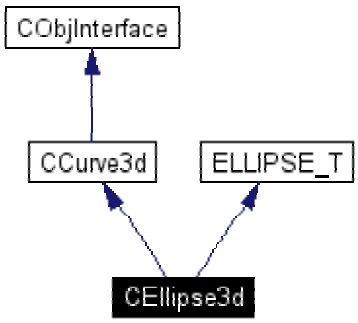
The documentation for this class was generated from the following files:

- curve3d.h
- curve3d.cpp

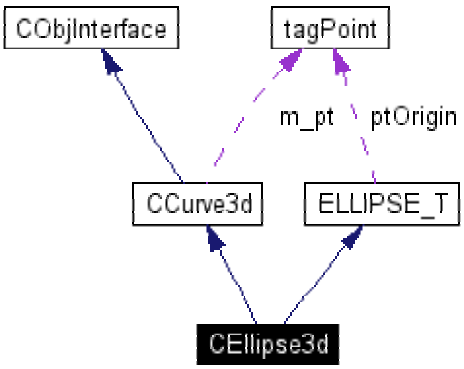
CEllipse3d Class Reference

002CEllipse3d#include <ellipse3d.h>

Inheritance diagram for CEllipse3d:



Collaboration diagram for CEllipse3d:



Public Member Functions

- CEllipse3d ()
- CEllipse3d (const ELLIPSE_T &ellipse)
- ~CEllipse3d ()

- void CreateSegments ()
  타원을 구성하는 점들을 구한다. 구한 점들은 Ccurve3d::m_pt에 저장된다.

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

- ellipse3d.h
- ellipse3d.cpp

CGeometry Class Reference

002CGeometry`#include <geometry.h>`

기하에 관련된 클래스.

Public Member Functions

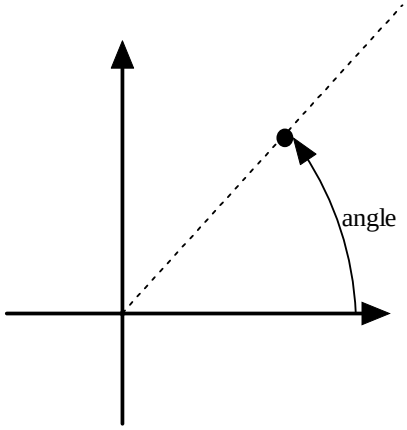
- CGeometry ()
- ~CGeometry ()

Static Public Member Functions

- bool IsSameValue (double val1, double val2, double tol)
  두 값의 차가 tol 이내인지를 검사.

- bool IsSamePoint (const POINT_T &pt1, const POINT_T &pt2, double tol)
  check whether pt1 and pt2 are same points or not.

- double GetRotatedAngleInXYPlane (const POINT_T &pt)



  calculate angle in 2D.

- double GetRotatedAngleInXYPlane (const VECTOR_T &vec)
  calculate rotated angle of pt in 2D.

- void GetRotatedAngleOfAxis (const VECTOR_T &vecAxis, double &alpha, double &beta)
  The CGeometry::GetRotatedAngleOfAxis function
  get rotated angle of x-axis,y-axis about z-axis(<0,0,1>).

- VECTOR_T RotateOnXYPlane (const VECTOR_T vec, double nDeg)
  x-y 평면에서 vec를 nDeg만큼 회전시킨다.

- POINT_T RotateAboutXYAxis (POINT_T &pt, const double &xangle, const double &yangle)
  first, rotate about x axis by alpha angle,
  then rotate about y axis by beta angle.
  rotating matrix=
  | 1 0 0 | | cos(beta) 0 -sin(beta)|
  |x y z|| 0 cos(alpha) sin(alpha)|*| 0 1 0 |*
  | 0 -sin(alpha) cos(alpha)| | sin(beta) 0 cos(beta) |.

- VECTOR_T RotateAboutXYAxis (const VECTOR_T &pt, double xangle, double yangle)
  The CGeometry::RotateAboutXYAxis function
  first, rotate about x axis by alpha angle,
  then rotate about y axis by beta angle.
  rotating matrix=
  | 1 0 0 | | cos(beta) 0 -sin(beta)|
  |x y z|| 0 cos(alpha) sin(alpha)|*| 0 1 0 |*
  | 0 -sin(alpha) cos(alpha)| | sin(beta) 0 cos(beta) |.

- POINT_T RotateAboutYXAxis (POINT_T &pt, double beta, double alpha)
  The CGeometry::RotateAboutYXAxis function
  first rotate about y axis by beta angle,
  rotate about x axis by alpha angle.
  | cos(beta) 0 -sin(beta)| | 1 0 0 |
  |x y z|| 0 1 0 |*| 0 cos(alpha) sin(alpha)|*
  | sin(beta) 0 cos(beta) | | 0 -sin(alpha) cos(alpah)|.

- VECTOR_T RotateAboutYXAxis (VECTOR_T &pt, double beta, double alpha)
  The CGeometry::RotateAboutYXAxis function
  first rotate about y axis by beta angle,
  rotate about x axis by alpha angle.
  | cos(beta) 0 sin(beta)| | 1 0 0 |
  |x y z|| 0 1 0 |*| 0 cos(alpha) sin(alpha)|*
  | -sin(beta) 0 cos(beta)| | 0 -sin(alpha) cos(alpah)|.

- POINT_T RotateAboutZAxis (const POINT_T &pt, const double theta)
  The CGeometry::RotateAboutZAxis function
  rotating matrix=
  | cos(theta) -sin(theta) 0 | |x|
  | sin(theta) cos(theta) 0 ||y|*
  | 0 0 1 | |z|.

- VECTOR_T RotateAboutZAxis (const VECTOR_T &pt, const double theta)
  The CGeometry::RotateAboutZAxis function

  rotate about z axis by theta.

  rotating matrix=

  | cos(theta) -sin(theta) 0 | |x|

  | sin(theta) cos(theta) 0 ||y|*

  | 0 0 1 | |z|.

- POINT_T Rotate (POINT_T pt, QUAT_T quat)
  pt를 quat로 회전시킨다.
- VECTOR_T Rotate (VECTOR_T vec, QUAT_T quat)
  vec를 quat로 회전시킨다.
- bool IsPointOnLine (double x, double y, double px, double py, double qx, double qy)
  x,y가 px,py,qx,qy로 이루어지는 선상에 존재하는가?
- INTERSECTION_E IntersectLine2D (double *x, double *y, double x1, double y1, double x2, double y2, double x3, double y3, double x4, double y4, const double tol)
  check line is intersect. if so get intersection point.

  Otherwise ,line is collinear or no intersecton.

- INTERSECTION_E IntersectLine2D (POINT_T &pt, const LINE_T &line1, const LINE_T &line2)
- bool IntersectLineWithPlane (POINT_T *pt, LINE_T line, PLANE_T plane)
  line이 plane과 만나는 교점을 구한다.
- bool IntersectLineToVolume (PLINE_T pLine, PSQVOLUME pVolume, PPOINT_T pRet)
  line이 volume과 만나는 교점들을 구한다.
- bool IsLeftSidePoint (const POINT_T &pt, const LINE_T &line)
  test which pt is located on left side of line or not.

- bool IsRightSidePoint (const POINT_T &pt, const LINE_T &line)
  test which the point is located on right side of line or not.

- bool IsEqualPoint2D (POINT_T &x, POINT_T &y, double tol=0.)
  두개의 점이 서로 같은가?

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

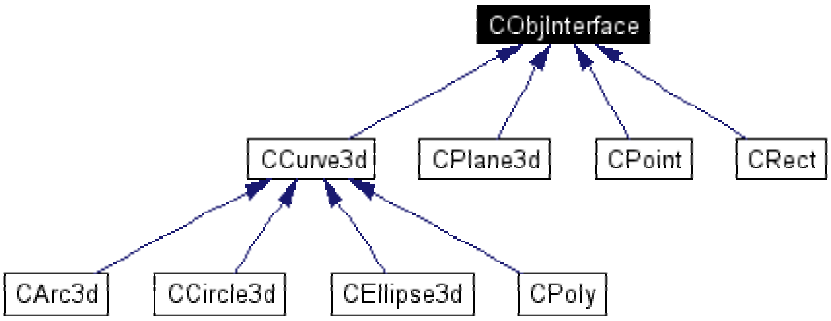
- geometry.h
- geometry.cpp

CObjInterface Class Reference

002CObjInterface#`include <ObjInterface.h>`

Inheritance diagram for CObjInterface:

Root 클래스가 된다. itklib의 모든 클래스가 CobjInterface 클래스를 상속받는다.



Public Member Functions

- virtual ~CObjInterface ()
- bool IsKindOf (const long nId)
  nId가 m_nId와 같은가?

Protected Attributes

- unsigned long m_nId

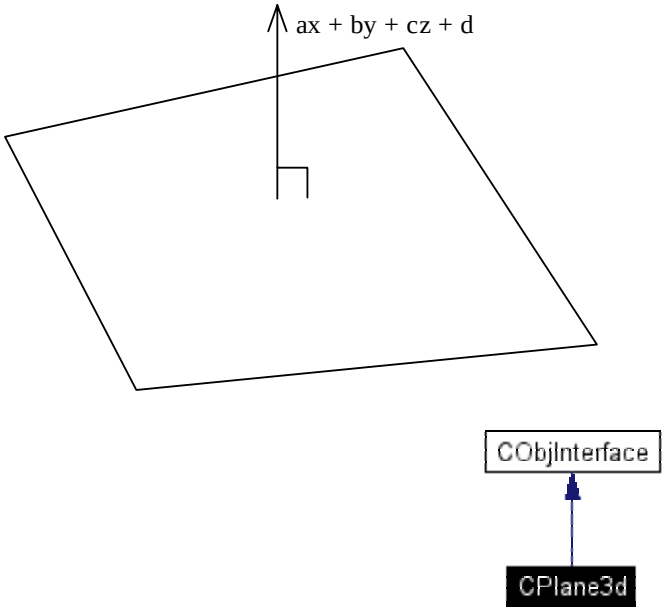
The documentation for this class was generated from the following files:

- ObjInterface.h
- ObjInterface.cpp

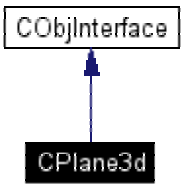
CPlane3d Class Reference

002CPlane3d#include <plane3d.h>

Inheritance diagram for CPlane3d:



Collaboration diagram for CPlane3d:



Public Member Functions

- CPlane3d ()
- CPlane3d (VECTOR_T &vec)
- CPlane3d (POINT_T &pt1, POINT_T &pt2, POINT_T &pt3)
- virtual ~CPlane3d ()

- VECTOR_T GetNormVector ()
  평면의 법선 벡터를 구한다.
- void SetNormVector (double dx, double dy, double dz)

평면의 법선 벡터를 설정한다.

- void SetNormVector (VECTOR_T vecNorm)
  평면의 법선 벡터를 설정한다.
- void GetRotatedAngleOfAxis (double &alpha, double &beta)
  평면의 법선 벡터가 x,y축에 대하여 회전된 각도를 구한다.
- POINT_T RotateAboutXYAxis (POINT_T pt, const double &alpha, const double &beta)
  pt를 x,y축에 대하여 각각 alpha,beta만큼 회전시킨다.
- VECTOR_T RotateAboutXYAxis (const VECTOR_T &vec, double alpha, double beta)
  vec를 x,y축에 대하여 각각 alpha,beta만큼 회전시킨다.
- POINT_T RotateAboutYXAxis (POINT_T pt, double beta, double alpha)
  pt를 y,x축에 대하여 각각 beta,alpha만큼 회전시킨다.
- VECTOR_T RotateAboutYXAxis (VECTOR_T &vec, double beta, double alpha)
  vec를 y,x축에 대하여 각각 beta,alpha만큼 회전시킨다.
- double GetAngle (VECTOR_T &vec)
  평면상의 vec가 회전된 각도를 구한다.
- bool ComputePlaneEquation (POINT_T &pt1, POINT_T &pt2, POINT_T &pt3)
  Computes plane equation.

- SIGN_T WhichSideOfPlane (POINT_T &pt)
  pt가 평면의 어느 면에 존재하는가?

Data Fields

- VECTOR_T m_vecNorm */// 평면의 법선 벡터(a,b,c)*
- double m_nD

Static Public Attributes

- const VECTOR_T XAXIS = {1.,0.,0.}
- const VECTOR_T YAXIS = {0.,1.,0.}
- const VECTOR_T ZAXIS = {0.,0.,1.}

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

- plane3d.h
- plane3d.cpp

PLANE_T Struct Reference

002PLANE_T#include <plane3d.h>

Data Fields

- double a
- double b
- double c
- double d

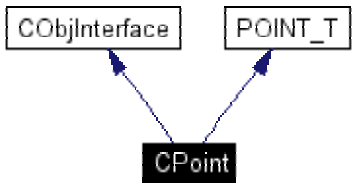
The documentation for this struct was generated from the following file:

- plane3d.h

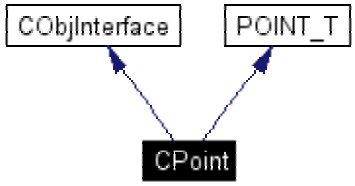
CPoint Class Reference

002CPoint#`include <point.h>`

Inheritance diagram for CPoint:



Collaboration diagram for CPoint:



Public Member Functions

- CPoint ()
- virtual ~CPoint ()

Static Public Member Functions

- const double Distance (const POINT_T &pt1, const POINT_T &pt2)
  calculate distance between pt1 and pt2.

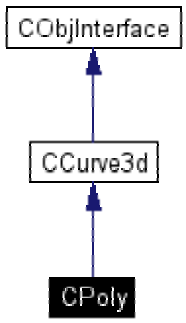
The documentation for this class was generated from the following files:

- point.h
- point.cpp

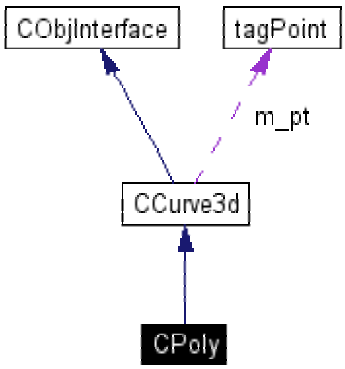
CPoly Class Reference

002CPoly#include <poly.h>

Inheritance diagram for CPoly:



Collaboration diagram for CPoly:



Public Member Functions

- CPoly ()
- virtual ~CPoly ()

Protected Attributes

- bool m_bClosed /// *시작점과 끝점이 서로 연결되었는가?*
- list< CPoint * > * m_plst

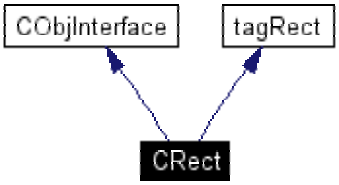
The documentation for this class was generated from the following files:

- poly.h
- poly.cpp

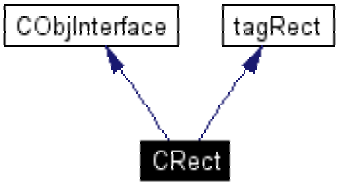
CRect Class Reference

002CRect#include <rect.h>

Inheritance diagram for CRect:



Collaboration diagram for CRect:



Public Member Functions

- CRect ()
- virtual ~CRect ()
- bool IsInnerPoint (const POINT_T &pt)
  pt가 내부의 점인가?
- bool IsOverlapped (const CRect &rect)
  rect와 서로 겹쳐지는가?

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

- rect.h
- rect.cpp

ITKGeo Class Reference

002ITKGeo#include <ITKGeo.h>

Static Public Member Functions

- double GetRotatedAngleInXYPlane (POINT_T pt)
  pt의 회전 각을 구한다.
- bool IsOverlapRect (PRECT_T pRect1, PRECT_T pRect2)
  두개의 pRect1,pRect2,가 서로 겹쳐지는가?
- bool IsPointInRect (const POINT_T &point, const RECT_T &rect)
  point가 rect내부의 점인가?
- bool IntersectLineToVolume (PLINE_T pLine, PSQVOLUME pVolume, PPOINT_T pRet)
- bool ComputePlaneEquation (PPLANE_T pPlaneEquation, POINT_T pt1, POINT_T pt2, POINT_T pt3)
  pt1,pt2,pt3로 이루어지는 평면의 식을 구한다.
- POINT_T GetMinDistPoint (const POINT_T &pt, int &nLoc, const LINE_T &line)
  pt와 가장 가까운 line상의 점을 구한다.

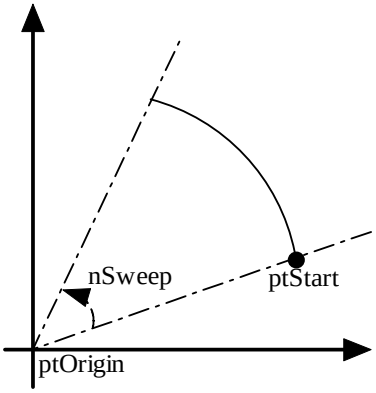
The documentation for this class was generated from the following files:

- ITKGeo.h
- ITKGeo.cpp

tagArc Struct Reference

002tagArc#include <arc3d.h>

Collaboration diagram for tagArc:



Data Fields

- POINT_T ptOrigin /// *arc의 원점*
- POINT_T ptStart /// *arc의 시작점*
- VECTOR_T vecNorm /// *arc의 법선 벡터*
- double nSweep
- double nRatio /// *장축과 단축의 비(1이면 circular arc)*

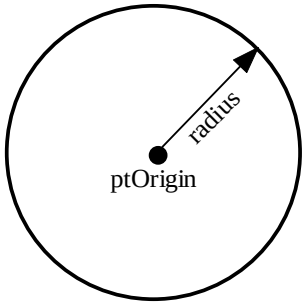
The documentation for this struct was generated from the following file:

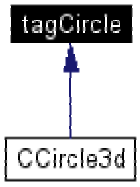
- arc3d.h

tagCircle Struct Reference

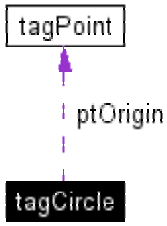
002tagCircle`#include <circle3d.h>`

Inheritance diagram for tagCircle:





Collaboration diagram for tagCircle:



Data Fields

- POINT_T ptOrigin
- double nRadius
- VECTOR_T vecNorm

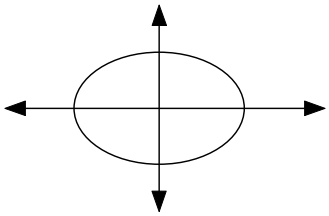
The documentation for this struct was generated from the following file:

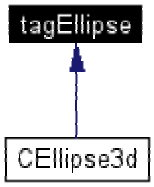
- circle3d.h

tagEllipse Struct Reference

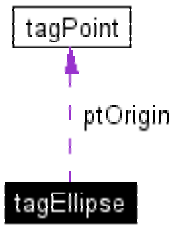
002tagEllipse#include <ellipse3d.h>

Inheritance diagram for tagEllipse:





Collaboration diagram for tagEllipse:



Data Fields

- VECTOR_T vecNorm /// *법선 벡터*
- POINT_T ptOrigin /// *원점*
- double nRotate /// *회전각도*
- double nAxis [2] /// *장축 , 단축*
- double nStartAngle /// *시작 각도*
- double nSweepAngle /// *sweep 각도*

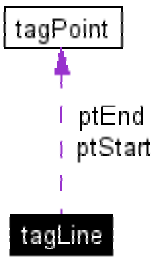
The documentation for this struct was generated from the following file:

- ellipse3d.h

tagLine Struct Reference

002tagLine#include <line.h>

Collaboration diagram for tagLine:



Data Fields

- POINT_T ptStart /// *시작점*
- POINT_T ptEnd /// *끝점*

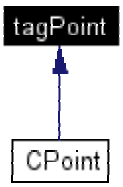
The documentation for this struct was generated from the following file:

- line.h

tagPoint Struct Reference

002tagPoint#include <point.h>

Inheritance diagram for tagPoint:



Data Fields

- double x
- double y
- double z

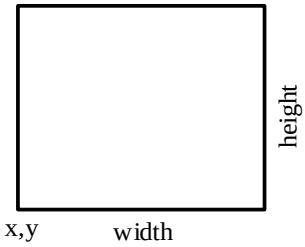
The documentation for this struct was generated from the following file:

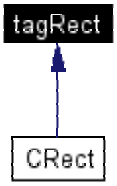
- point.h

tagRect Struct Reference

002tagRect#include <rect.h>

Inheritance diagram for tagRect:





Data Fields

- double x
- double y
- double width
- double height

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

- rect.h