# SharpCAD

SharpCAD是一个用脚本创建三维对象的免费软件，像OpenSCAD一样，不过它使用的脚本是C#，且目前仅支持windows。现在还仅仅是免费软件，还未开源，不过以后我会将其开源。

和OpenSCAD一样，SharpCAD提供了两种主要的建模技术:

1、构造实体几何=>CSG。SharpCAD定义了Circle、Square二维对象以及Box、Sphere等三维基础对象。不过和OpenSCAD不一样的是，你需要用+、-、\*来进行布尔并、差、交运算。

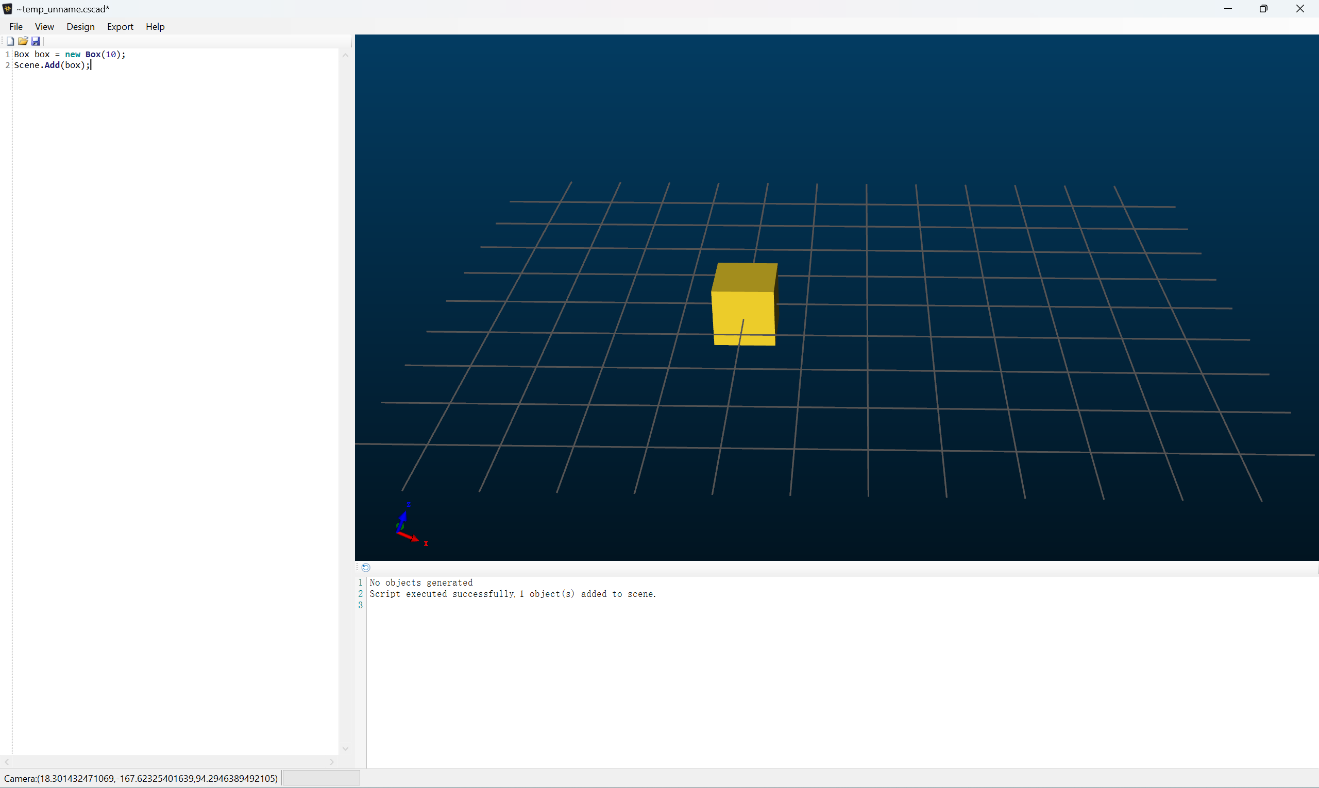
2、基于二维轮廓的拉伸旋转等操作。

目前SharpCAD可将建模成果导出为OFF以及OBJ文件，当然你也可以把你的成果保存为脚本供其他脚本调用。

# 编写一个简单的脚本吧

你可像在Visual Studio里编写C#代码一样编写CSCAD脚本。 比如创建一个Box然后添加到场景中你就会在场景看到一个立方体:

Box box = new Box(10);  
Scene.Add(box);



然后保存脚本到本地或导出为OFF/OBJ文件。

# 目录

* [用到的三方库](#用到的三方库)
* [现有示例脚本](#现有示例脚本)
* [已支持及后续计划](#已支持及后续计划)
* [函数速查表](#函数速查表)

# 用到的三方库

* [FastColoredTextBox](https://github.com/PavelTorgashov/FastColoredTextBox)
* [Newtonsoft.Json](https://github.com/JamesNK/Newtonsoft.Json/)
* [scriptcs-engines](https://github.com/scriptcs/scriptcs-engines)
* [Open Cascade](https://www.Opencascade.com/)
* [AvalonEdit](https://github.com/icsharpcode/AvalonEdit)
* [Autofac](https://github.com/autofac/Autofac)
* [CShell](https://github.com/lukebuehler/CShell)
* [MPFR](https://www.mpfr.org/)
* [CGAL](https://www.cgal.org/)
* [GMP](https://gmplib.org/)
* [TBB](https://github.com/oneapi-src/oneTBBhttps://gmplib.org/)

# 现有示例脚本

目前只提供了8个示例，所有示例都在examples目录下 ## Cube示例 Box box = new Box(22,16,14); Scene.Add(box); avatar

## CSG示例

Sphere sphere = new Sphere(10);  
Box box = new Box(15);  
  
//Union  
ShapeBase union = box + sphere;  
union.Translate(-24,0,0);  
  
//Intersect  
ShapeBase intersection = box \* sphere;  
  
//Diffrence  
ShapeBase difference = box - sphere;  
difference.Translate(24,0,0);  
  
//Scene.Add(box);  
Scene.Add(union);  
Scene.Add(intersection);  
Scene.Add(difference);

|  |
| --- |
|  |

## Linear\_Extrude

//Simple Linear\_Extrude  
Square square = new Square(20, 10);  
var linear0 = ACOCCTTools.Linear\_Extrude(square,20);  
linear0.Translate(0, -30, 0);  
linear0.Color = Color.Red;  
  
Scene.Add(linear0);

|  |
| --- |
|  |

## rotate\_extrude

//Square Rotate\_Extrude  
Square square = new Square(5);  
square.Translate(10,0,0);  
var re0 = ACOCCTTools.Rotate\_Extrude(square,360);  
re0.Color = Color.Red;  
Scene.Add(re0);  
  
//PolyFace Rotate\_Extrude  
List<Point3D> pnts = new List<Point3D>()  
{  
 new Point3D (0,0,0),  
 new Point3D (8,4,0),  
 new Point3D (4,8,0),  
 new Point3D (4,12,0),  
 new Point3D (12,16,0),  
 new Point3D (0,20,0)  
};  
PolyFace face = new PolyFace(pnts);  
  
var re1 = ACOCCTTools.Rotate\_Extrude(face ,360);  
re1.Translate(0,30,0);  
re1.Color = Color.Green;  
Scene.Add(re1);  
  
//complex Rotate\_Extrude  
{  
 Square square0 = new Square(5);  
 square0 .Color = Color.Magenta;  
 var square1 = square0.Clone();  
 square1.Translate(12.5,0,0);  
 var re2 = ACOCCTTools.Rotate\_Extrude(square1,180);  
 re2.Translate(40,40,0);  
  
 var square2 = square0.Clone();  
 square2.Translate(5,0,0);  
 var re3 = ACOCCTTools.Rotate\_Extrude(square2,180);  
 re3.Translate(7.5,0,0);  
 re3.Translate(40,40,0);  
  
  
 var square3 = square0.Clone();  
 square3.Translate(5,0,0);  
 var re4 = ACOCCTTools.Rotate\_Extrude(square3,-180);  
 re4.Translate(-7.5,0,0);  
 re4.Translate(40,40,0);  
  
 Scene.Add(re2);  
 Scene.Add(re3);  
 Scene.Add(re4);  
}

|  |
| --- |
|  |

## import\_off\_and\_projection

//import off and project triangleMesh  
String strFile = @"examples\data\elephant.off";  
TriangleMesh triMesh = ACOCCTTools.Import(strFile);  
  
triMesh.Translate(0, 0, 1);  
triMesh.Scale(10,10,10);  
Scene.Add(triMesh);  
triMesh.Color = Color.Blue;  
var faces = ACOCCTTools.Projection(triMesh);  
  
//set color  
foreach(var item in faces)  
{  
 item .Color=Color.Red;  
}  
Scene.Add(faces);  
  
//ShapeBase  
Box b = new Box(10);  
b.Translate(0, 20, 10);  
var faces1 = ACOCCTTools.Projection(b);  
  
faces1[0].Color=Color.Green;  
Scene.Add(faces1);  
Scene.Add(b);

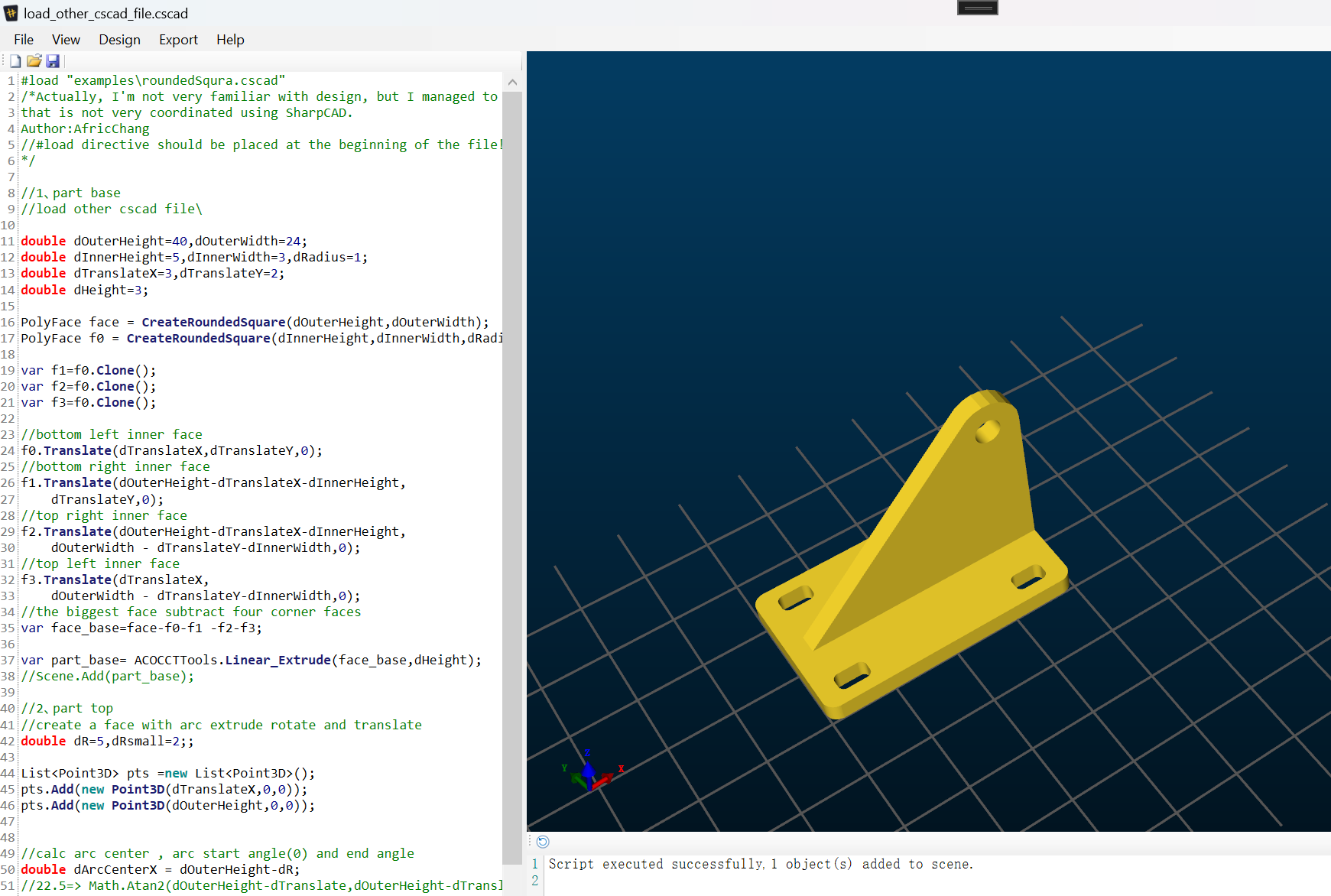
|  |
| --- |
|  |

## load\_other\_cscad\_file

#load "examples\roundedSqura.cscad"  
/\*Actually, I'm not very familiar with design, but I managed to design an unknown three-dimensional part   
that is not very coordinated using SharpCAD.  
Author:AfricChang  
//#load directive should be placed at the beginning of the file!!!!  
\*/  
  
//1、part base  
//load other cscad file\  
#load "examples\roundedSqura.cscad"  
double dOuterHeight=40,dOuterWidth=24;  
double dInnerHeight=5,dInnerWidth=3,dRadius=1;  
double dTranslateX=3,dTranslateY=2;  
double dHeight=3;  
  
PolyFace face = CreateRoundedSquare(dOuterHeight,dOuterWidth);  
PolyFace f0 = CreateRoundedSquare(dInnerHeight,dInnerWidth,dRadius);  
  
var f1=f0.Clone();  
var f2=f0.Clone();  
var f3=f0.Clone();  
  
//bottom left inner face  
f0.Translate(dTranslateX,dTranslateY,0);  
//bottom right inner face  
f1.Translate(dOuterHeight-dTranslateX-dInnerHeight,  
 dTranslateY,0);  
//top right inner face  
f2.Translate(dOuterHeight-dTranslateX-dInnerHeight,  
 dOuterWidth - dTranslateY-dInnerWidth,0);  
//top left inner face  
f3.Translate(dTranslateX,  
 dOuterWidth - dTranslateY-dInnerWidth,0);  
//the biggest face subtract four corner faces  
var face\_base=face-f0-f1 -f2-f3;  
  
var part\_base= ACOCCTTools.Linear\_Extrude(face\_base,dHeight);  
//Scene.Add(part\_base);  
  
//2、part top  
//create a face with arc extrude rotate and translate  
double dR=5,dRsmall=2;;  
  
List<Point3D> pts =new List<Point3D>();  
pts.Add(new Point3D(dTranslateX,0,0));  
pts.Add(new Point3D(dOuterHeight,0,0));  
  
  
//calc arc center , arc start angle(0) and end angle  
double dArcCenterX = dOuterHeight-dR;  
//22.5=> Math.Atan2(dOuterHeight-dTranslate,dOuterHeight-dTranslate)/2;  
double dArcCenterY = dOuterHeight-dTranslateX- dR/Math.Tan(22.5\*Math.PI/180);  
  
double dArcEndAngle=157.5;//180-22.5;  
  
//Create an arc with a center point of (dArcCenterX, dArcCenterY),   
//start angle is 0, and an end angle is dArcEndAngle  
int nPtCount=10;  
double dStep=dArcEndAngle/nPtCount;  
for(int i=0;i<nPtCount;i++)  
{  
 double dAngle=(i\*dStep\*Math.PI/180);  
 double dX=dArcCenterX+dR\*Math.Cos(dAngle);  
 double dY=dArcCenterY+dR\*Math.Sin(dAngle);  
 Point3D pt= new Point3D(dX,dY,0);  
 pts.Add(pt);  
}  
  
  
PolyFace face\_top\_base = new PolyFace(pts);  
  
Circle innerCircle= new Circle(dRsmall);  
innerCircle=(Circle)innerCircle.Translate(dArcCenterX,dArcCenterY,0);  
var fff\_top=face\_top\_base-innerCircle;  
  
//extrude fff\_top  
var part\_top = ACOCCTTools.Linear\_Extrude(fff\_top,dHeight);  
  
//rotate and translate part\_top  
part\_top.Rotate(90,0,0);  
part\_top.Translate(0,(dOuterWidth+dHeight)/2,dHeight);  
  
//union part\_top part\_base  
var part= part\_top+part\_base;  
Scene.Add(part);

avatar ## triangleMesh Boolean //import off and boolean String strFile = @“examples.off”; TriangleMesh triMesh0 = ACOCCTTools.Import(strFile); triMesh0.Scale(10,10,10);

String strFile1 = @"examples\data\eight.off";  
TriangleMesh triMesh1 = ACOCCTTools.Import(strFile1);  
triMesh1.Scale(10,10,10);  
  
var triBoolean = triMesh1-triMesh0;  
triBoolean.Color = Color.Green;  
Scene.Add(triBoolean );  
  
triMesh0.Translate(-15,0, 0);  
triMesh1.Translate(-15, 0, 0);  
  
triMesh0.Color = Color.Red;  
Scene.Add(triMesh0);  
  
triMesh1.Color = Color.Green;  
Scene.Add(triMesh1);



## create\_TriangleMesh

Point3D[] pts = new Point3D[3] {  
new Point3D(0,0,0),  
new Point3D(10,0,0),  
new Point3D(10,10,0),  
};  
  
Indeces id0 = new Indeces(0,-1,-1);  
Indeces id1 = new Indeces(1,-1,-1);  
Indeces id2 = new Indeces(2,-1,-1);  
TriangleIndex[] ids = new TriangleIndex[1]{new TriangleIndex(id0,id1,id2)};  
TriangleMesh tri = new TriangleMesh(pts,ids,null,null);  
Scene.Add(tri);

|  |
| --- |
|  |

# 已支持及后续计划

## 相对OpenSCAD已支持

* 二维图元已支持，Text未支持，后续考虑支持
* projection cut参数未支持，后续考虑支持
* linear\_extrude部分支持，twist , slices , scale 参数未支持，后续可能会支持，其余参数不考虑支持
* rotate\_extrude,$fn没支持，但在三角化时可知设置dAngleDelta，后续考虑在导出时或渲染时支持该参数
* import，目前仅支持OFF，后续会考虑支持OBJ、STL、DXF等其他格式
* 布尔运算已支持 ## 后续计划
* 上述OpenSCAD参数完善
* Text支持
* 镜像
* 扫掠
* 放样
* 平面分割对象
* 平面与对象相交
* 更多的示例
* 自定义插件
* ……

# 函数速查表

目前所有建模操作都在ACC命名空间(OCCTModeling.dll). ACC命名空间提供了基础的Point2D、Point3D、几何图元变换及相关运算。 ## 几何图元 包括二维及三维几何图元，所有的二三维图元(除了TriangleMesh)都继承自ShapeBase，都可以进行旋转(Rotate方法)、平移(Translate方法)、缩放(Scale方法)、投影以及布尔运算(通过符号+、-、\*实现)，其中二维图元还可以进行拉伸、旋转拉伸。非三角网的图元可以三角化后导出。

### Circle

二维圆  
Circle(double radius)//创建指定半径的圆

### Square

方形  
Square(double height, double width, bool bCenter)//指定长宽及中心点是否在原点  
Square(double height, double width)//指定长宽(中心点在原点)  
Square(double size, bool bCenter)//长宽均等于size中心点是否在原点  
Square(double size)//长宽均等于size(中心点在原点)

### PolyFace

多边形  
PolyFace(List<Point3D> verteces, List<List<int>> indeces)//指定顶点及索引创建带洞多边形，注意，支持带洞多边形，第一个面需要逆时针，后续面作为洞应顺时针。  
PolyFace(List<Point3D> verteces)//简单多边形

### Box

立方体  
Box(double dLength, double dWidth, double dHeight, bool bCenter)//指定长宽高以及中心点是否在原点  
Box(double dLength, double dWidth, double dHeight)//指定长宽高,中心点在原点  
Box(double size, bool bCenter)//长宽高均为size，中心点是否在原点  
Box(double size)//长宽高均为size，中心点在原点

### Sphere

球体  
Sphere(double dRadius)//指定半径创建球体

### Cylinder

圆柱体  
Cylinder(double dTopRadius, double dBottomRadius, double dHeight)//指定顶面半径、底面半径以及高创建圆柱体  
Cylinder(double dRadius, double dHeight)//创建顶底半径相等高为dHeight的圆柱体

### Torus

圆环  
Torus(double dRadius, double dRingRadius, double dSweepAngle)//指定圆半径、管半径以及扫掠角度，注意，扫掠角度是以X轴为起点，逆时针扫过的角度，以度为单位

### TriangleMesh

三角网  
导入的OFF(目前仅支持OFF,后续会支持OBJ、STL等其他格式)以及几何图元三角化之后的结果。

* public TriangleMesh(Point3D[] verteces, TriangleIndex[] triangles, Point3D[] normals, Point2D[] textureCoods)//根据顶点、索引、法线、纹理坐标构建三角网，其中法线及纹理坐标可为空。
* TriangleIndex 构造 public TriangleIndex(Indeces index0, Indeces index1, Indeces index2)//Indeces 构造 public Indeces(int nID, int nUV, int nNormal)
* 构造三角网请参照示例[create\_TriangleMesh](#X9dd2732e0967092d596b930be6a740f8382fbec)

## 运算

所有的运算都在ACC.ACOCCTTools内中,主要包含旋转、平移、缩放、投影、布尔运算等操作

### 变换操作

* public static void Translate(ShapeBase shape, double x, double y, double z)//平移图元
* public static void Translate(ShapeBase shape, Point3D translate)//平移图元
* public static void Rotate(ShapeBase shape, double x, double y, double z)//旋转图元，以度为单位
* public static void Rotate(ShapeBase shape, Point3D rotate)//旋转图元，以度为单位
* public static void Scale(ShapeBase shape, double x, double y, double z)//缩放图元
* public static void Scale(ShapeBase shape, Point3D scale)//缩放图元

### 运算操作

* public static TriangleMesh BooleanOperation(TriangleMesh triMeshA, TriangleMesh triMeshB, BooleanType eBoType)//三角网布尔运算，BooleanType包含交(Intesection)、并(Union)、差(Diffrence)三种。注意，如果ShapeBase图元和三角网图元布尔运算需要先将ShapeBase三角化为三角网图元才可运算
* public static TriangleMesh Trianguation(ShapeBase shape, double dAngleDelta, double dLengthDelta)//三角化ShapeBase图元,dAngleDelta为角度阈值，dLengthDelta为长度阈值。
* public static ShapeBase Linear\_Extrude(ShapeBase shape, double height)//拉伸图元
* public static ShapeBase Rotate\_Extrude(ShapeBase shape, double dAngle)//旋转拉伸图元
* public static TriangleMesh Import(string strFileName)//导入文件，目前仅支持OFF
* public static List Projection(ShapeBase shape)//图元投影
* public static List Projection(TriangleMesh triMesh)//三角网投影
* public static Combine(System.Collections.Generic.List<ACC.TriangleMesh>)//合并三角网

# 彩蛋

虽然目前没支持导入其他格式的文件，但可以用其他库导入然后转成TriangleMesh。可以用#r load外部dll,如下:

#r "\examples\data\geometry3.dll"  
g3.Vector3f g= new g3.Vector3f(10,10,10);  
SharpCADConsole.WriteLine(g.Length);

