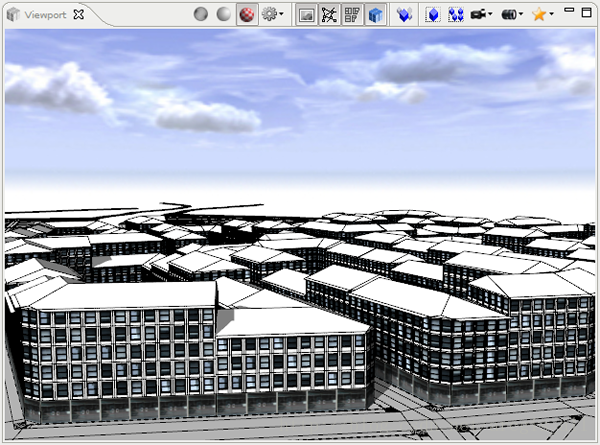
**001--教程简介**

本教程讲解如何利用CityEngine快速地从无到有创建一个城市，通过对CityEngine各部分功能的讲解使读者了解完整的工作流程。您将学习到怎样建立一个新的工程，创建道路网以及设置“形状创建参数”。最后，您将学习使用规则编辑器并生成城市建筑模型。



**第一部分：创建一个新的工程**

**创建新工程和场景**

第一步，创建一个新的CityEngine工程。

选择【File】—>【New】—>【CityEngine】—>【CityEngineproject】

点击【Next】，将工程命名为“FirstCity”并单击【[Finish](https://www.baidu.com/s?wd=Finish&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)】

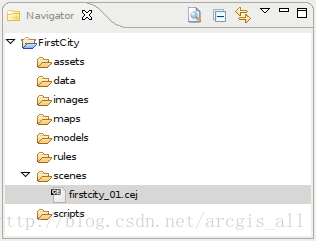
完成以上操作，一个新的工程便建立好了并出现在在文件导航窗口中（默认布局下文件导航窗口位于CityEngine界面的左上角）。这些自动生成的文件夹用来存储你的工程数据，如assets文件夹（用以存储贴图和引用的模型部件数据），rules文件夹（用以存储规则文件）等。

第二步，创建一个新的场景：

选择【File】—>【New】—>【CityEngine】—>【CityEngine*scene*】

确认新场景所保存的文件夹位置(/FirstCity/scenes)，将该场景命名为“firstcity\_01.cej”并点击【Finish】

我们的工作空间中现在包含了一个新的空工程以及一个场景文件：



**拷贝规则和资源**

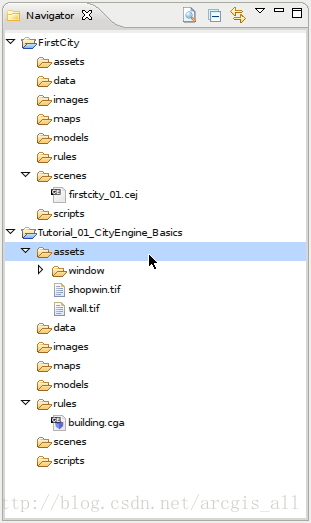
在本教程的后面部分，我们需要使用规则文件和资源（assets）用以生成建筑物模型。因此我们现在需要从教程配套的示例工程中拷贝这些文件。

首先，我们将教程1的示例工程导入到当前的工作空间

下载并导入教程工程（教程1）到我们的工作空间中

接下来，我们将所需要的文件从已下载的教程1工程中拷贝到我们的新建工程中。

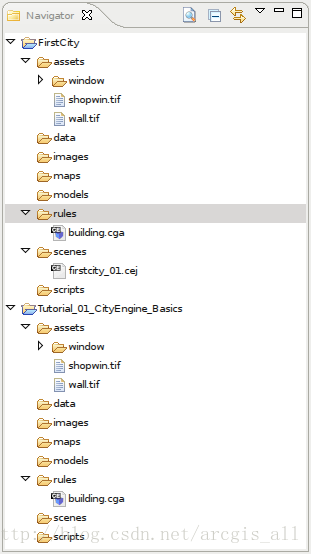
定位到教程1的资源（asset）文件夹。



使用复制和粘贴(Ctrl+C和Ctrl+V)将资源（asset）文件夹中所有的文件与文件夹拷贝到我们新建工程的资源（asset）文件夹中。

使用同样的方法，将规则文件夹中的“building.cga”规则文件拷贝到新工程的规则文件夹中。

现在，你的导航视窗应如下图所示：



现在，我们已经为第二部分的创建道路网和建筑模型工作做好了准备。

**第二部分：街道和建筑物**

**创建道路网**

第一步，创建道路网

单击viewport视窗将其激活

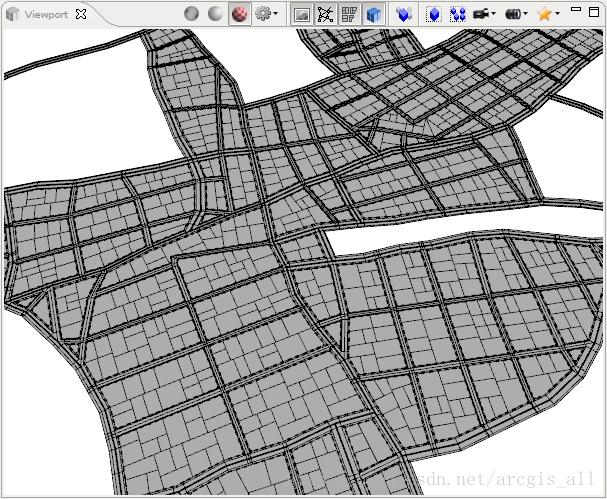
选择【Graph】—>【GrowStreets...】

选择默认设置来建立我们的城市。

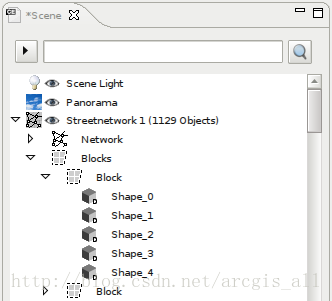
单击【Apply】，并关闭对话框。

单击ViewPort视察上方工具栏中的“FrameAll”按钮（或单击键盘上的“a”键）在Viewport视窗中全图显示你新建的道路网。

现在，你的Viewport视窗中应显示如下图所示的道路网以及默认的建筑物形状。



现在我们来看一下场景编辑器视窗“*SceneEditor*”。可以发现一个名为“Streetnetwork1”的新图层已经生成。

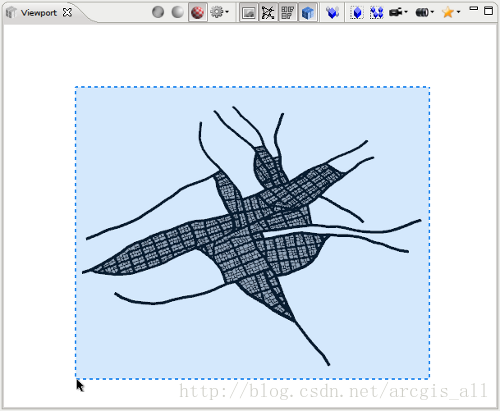


**修改“形状创建参数”**

现在我们改变一些形状创建参数使场景看起来更加有趣。

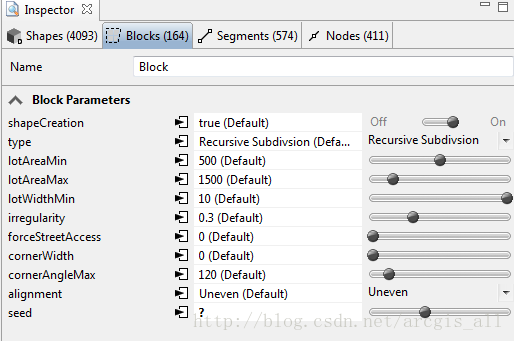
点击“a”键使当前场景全图层显示。

按住鼠标左键，从右向左拖拽一个矩形框将场景中的全部元素都包含在内。



找到位于CityEngine窗口右上侧的Inspector面板，单击"[Blocks](https://www.baidu.com/s?wd=Blocks&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)"选项卡

打开"BlockParameter"，将看到“形状创建参数”。



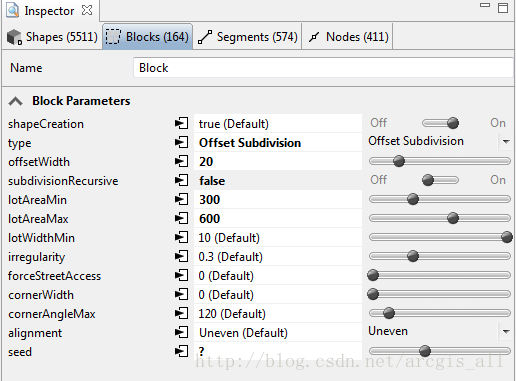
将类型（type）设置为偏移细分（**OffsetSubdivision**）

将最小面积（**lotAreaMin**）设置为300

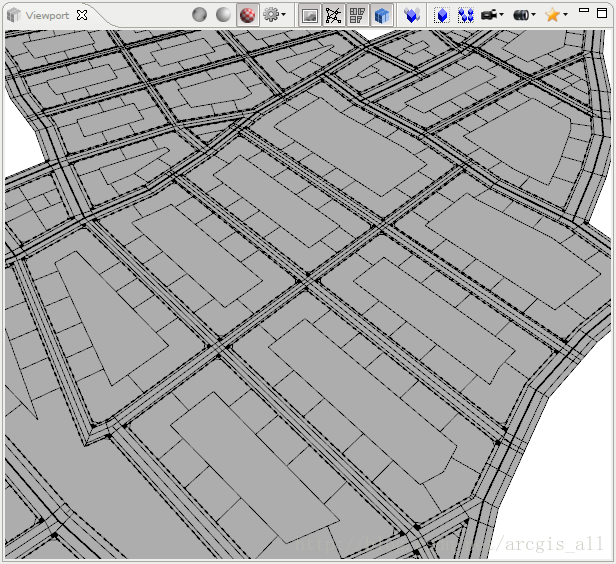
将最大面积（**lotAreaMax**）设置为600

将偏移宽度（**offsetWidth**）设置为20

将代替分割递归（**subdivisionRecursive**）设置为否（false）



仔细观察场景中建筑物形状产生的变化



接下来进入第三部分，学习怎样对建筑物形状应用CGA规则并建立建筑物模型。

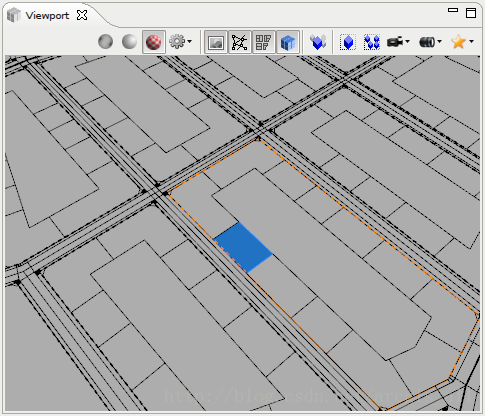
**第三部分：应用CGA规则并生成模型**

**CGA规则简介**

CityEngine中，使用CGA规则来描述三维模型。一个CGA规则文件由若干规则组成，这些规则用来定义三维模型是如何生成的。在一个CGA规则文件被指定给某个形状（如建筑物地块）时，建筑物模型便开始生成。在本部分的教程中，我们将应用和修改一个由基本规则构成的简单的CGA文件。

**应用一个CGA规则文件**

在3D视窗中，单击某个形状的内部以选中它。

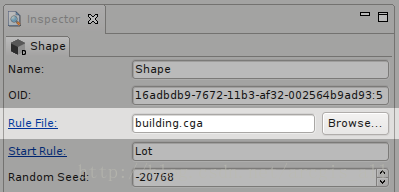


在工具条中，单击Assign按钮并选中“building.cga”规则文件

另一种应用规则的方式，是直接从导航视窗（Navigator）中将规则文件拖拽到viewport视窗中的已选中形状上。



现在，Inspector视窗应如下图所示



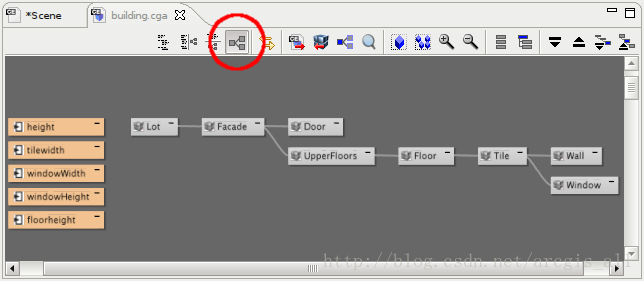
**浏览CGA规则并设置初始规则**

在生成模型前，我们需要确认在Inspector视窗中已选定形状的初始规则被设置为了“Lot”，通常这种设置为默认设置。需要注意的是初始规则的名称由CityEngine自动生成，但可以进行修改。

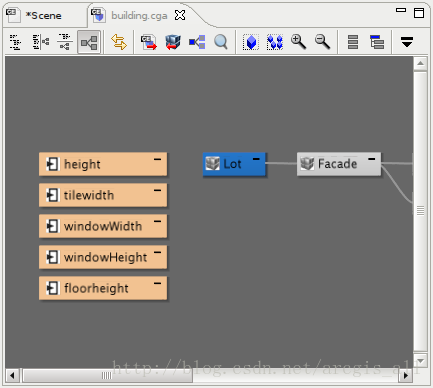
继续保持某形状在前一节中被选中的状态，找到Inspector视窗中的初始规则（StartRule）属性。

双击规则文件打开规则编辑器（RuleEditor）视窗。在左侧（黄色标签部分）显示的是规则的属性，在右侧（灰色标签部分）显示的是实际规则的图表。

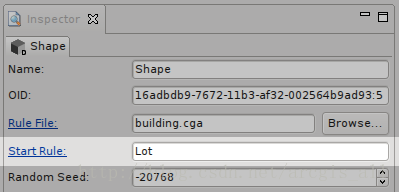
点击下图红色圆圈内所示的按钮使编辑视窗最大化。



将规则定位在规则图的根部，在这里的根部指的是“Lot”。这个规则将是已选定形状的起始规则。



再次查看inspector视窗，确认在"StartRule"处选定的是"Lot"（“Lot”为默认值），如下图所示。

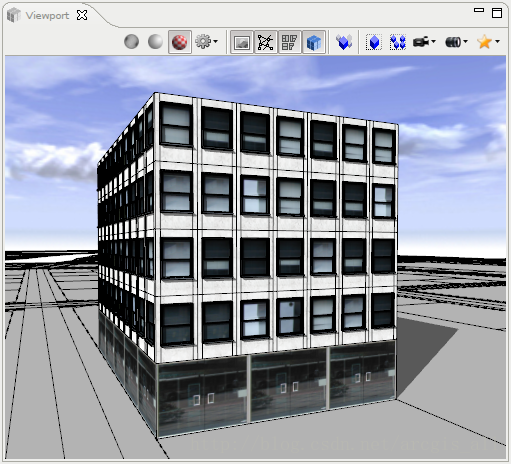


现在我们已经做好生成真实建筑模型几何体的准备了。

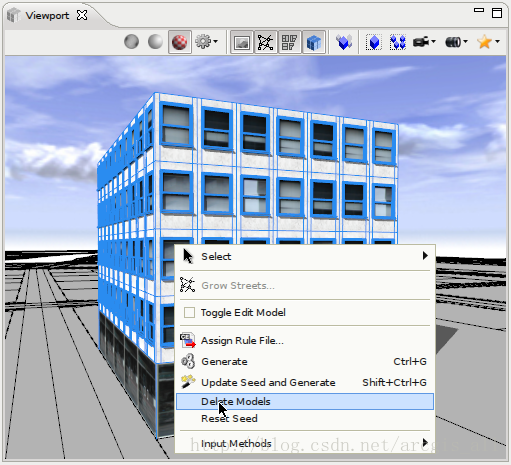
**生成第一个建筑物模型**

确认选定形状仍然在选中状态，单击生成（Generate）按钮。

建筑物模型出现在了选定形状的上方



单击选中模型，使用右键菜单中的"DeleteModel"选项来删除模型。

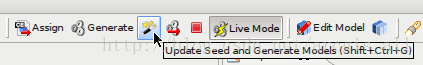


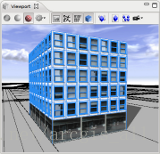
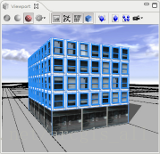
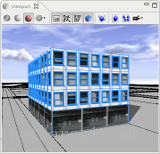
再次生成模型。

我们现在已经做好修改模型的准备。首先，我们将修改一个规则的属性——buildingheight（建筑物高度）。接下来，我们将为建筑物添加一个不同的屋顶，这样我们就需要一个新的CGA规则。

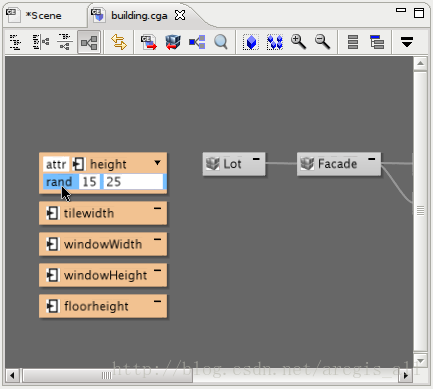
**修改规则属性“建筑物高度”**

高度在相应的规则文件中被设置为了随机值，使用更新并生成模型（UpdateseedandGenerateModels）命令多次生成建筑物，观察每一次生成的建筑物的变化。





查看规则编辑器中建筑物高程的默认定义：



在Inspector视窗中，定位到规则参数“RuleParameter”。

找到高程"height"参数将其值修改为“18”。参数值的字体变成粗体，为用户定义值。

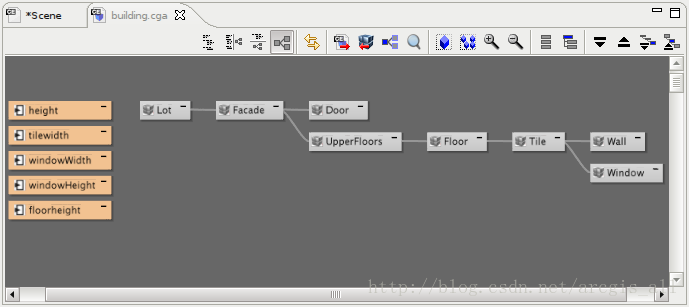


再次重复点击更新和生成模型命令（UpdateseedandGenerateModels）。可以发现建筑物模型的高度不再变化。你可以通过点击数字“18”左侧的黑色三角图标将其重置为默认行为。

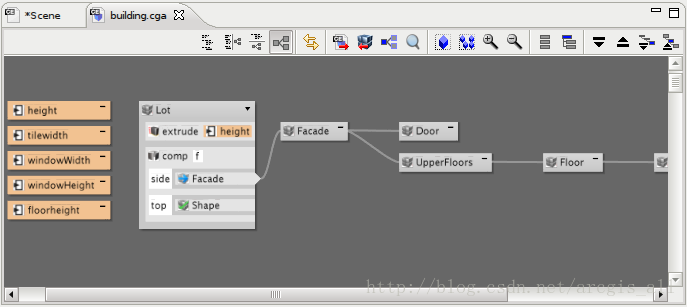
接下来，我们为屋顶添加一个新的规则。

**添加一个屋顶规则**

在inspector视窗中，通过点击规则文件打开规则编辑器



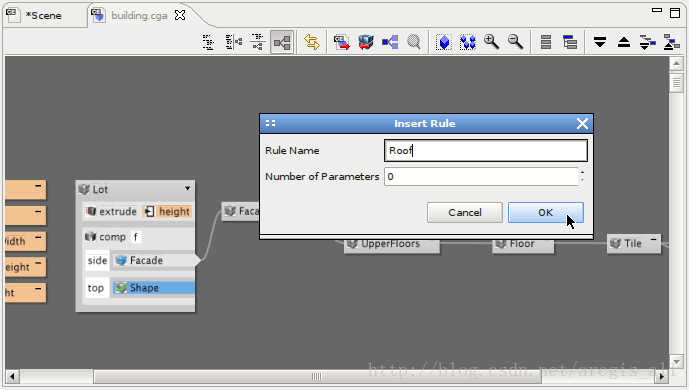
找到“Lot”规则并点击其右上角的“V”形符号将其展开。



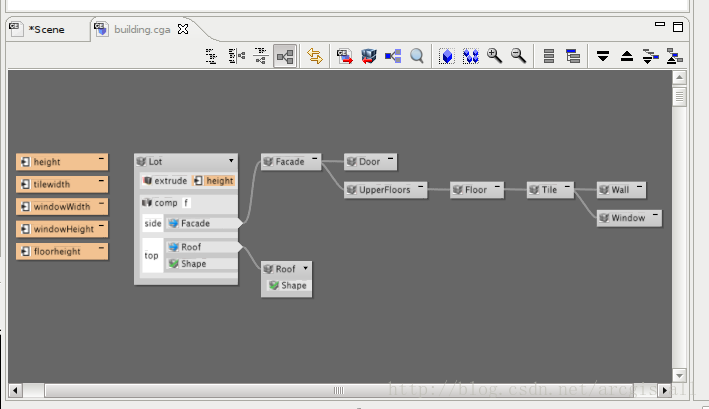
在“comp”部分，右键点击“shape”，在右键菜单中进行选择，新建一个新的规则。



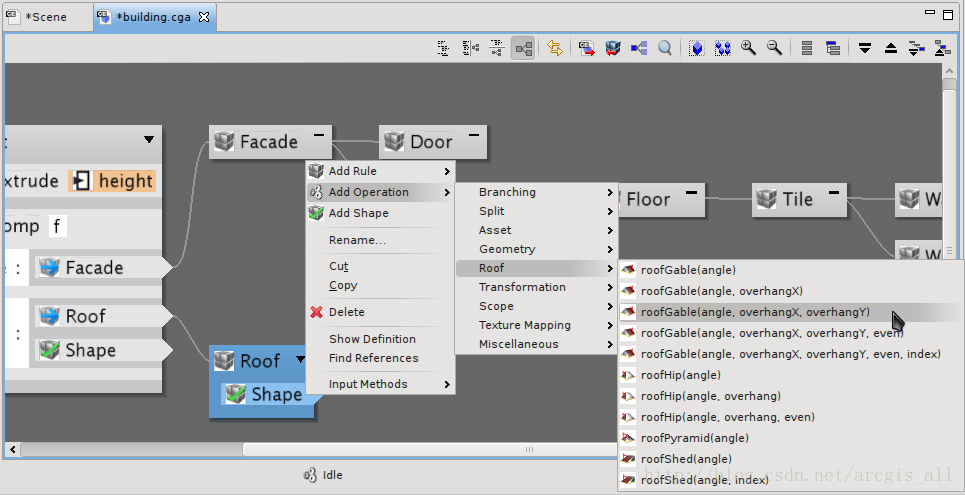
将新规则命名为"Roof"。



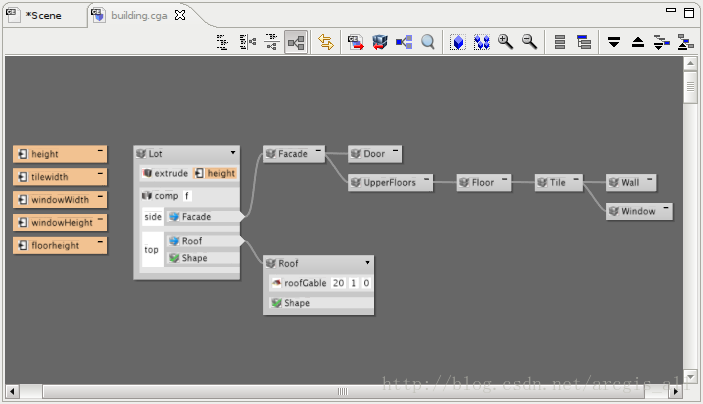
观察新规则是如何生成并被连接到"Lot"节点上的。



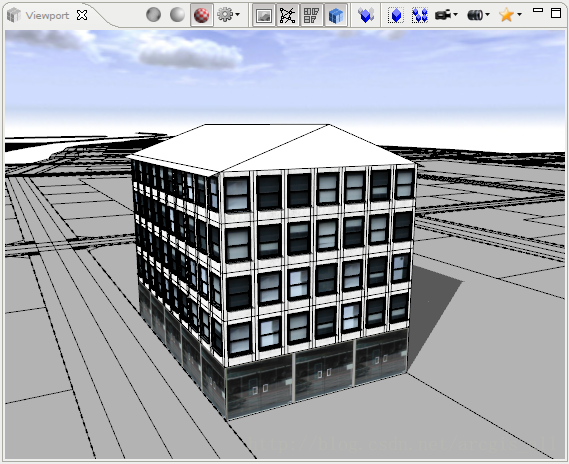
为了建立一个屋顶模型，我们现在插入一个人字形屋顶图形（即通常说的双坡顶）。右键单击“Roof”规则中的“Shape”，找到roofGable(angle,overhangX,overhangY)操作。



观察roofGable操作的生成情况。点击三个参数值分别输入20,1,0作为屋顶的参数。



单击生成（generate）按钮，查看新生成的模型。

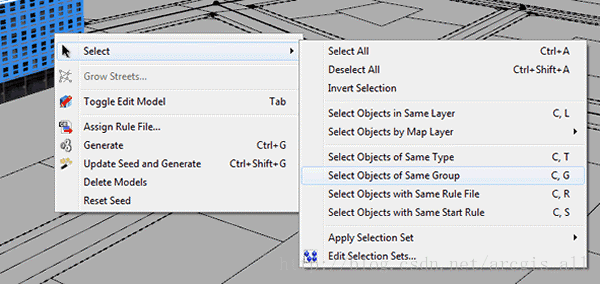


**生成一个小的城市模型**

在最后一个步骤里面，我们会将规则应用到较大数量的形状上。为了给较大数量的形状分配规则，我们将使用选择菜单。

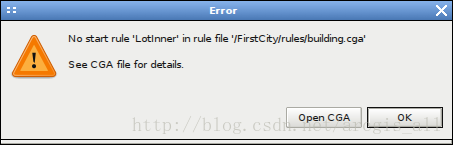
选择已生成的模型。

右键单击3D视口，并选择“Select->SelectObjectsofSameGroup”以选择大量的形状。

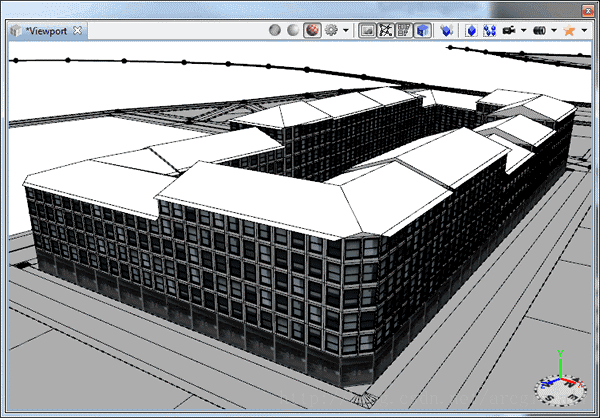


单击assign按钮，将building.cga规则分派给选中的形状。

单击“Generate”按钮生成模型。注意：根据你的选择，将会弹出一些关于起始规则“LotInner”或类似的警告对话框。你可以忽略这些对话框，因为我们对所有的形状分配规则，但我们的规则并不包含名为“LotInner”等的起始规则。



你现在应该可以看到一些高度不同的建筑物了。



**002---CityEngineCGA规则包**

* [CityEngine的目录结构](file:///C:\Users\admin\Desktop\Untitled-1.html#CityEngine_12)
* [基本规则](file:///C:\Users\admin\Desktop\Untitled-1.html#_39)
* [开始编写](file:///C:\Users\admin\Desktop\Untitled-1.html#_91)
  + [简单的拉伸建筑](file:///C:\Users\admin\Desktop\Untitled-1.html#_112)
    - [创建变量动态修改高度](file:///C:\Users\admin\Desktop\Untitled-1.html#_145)
    - [控制GUI界面中Rule的高度变化范围](file:///C:\Users\admin\Desktop\Untitled-1.html#GUIRule_162)
    - [拆分建筑六面](file:///C:\Users\admin\Desktop\Untitled-1.html#_188)
    - [建筑物贴图](file:///C:\Users\admin\Desktop\Untitled-1.html#_211)
    - [建筑分层](file:///C:\Users\admin\Desktop\Untitled-1.html#_251)
  + [shape的理解](file:///C:\Users\admin\Desktop\Untitled-1.html#shape_352)
  + [常用的函数](file:///C:\Users\admin\Desktop\Untitled-1.html#_376)
    - [数据处理函数](file:///C:\Users\admin\Desktop\Untitled-1.html#font_color2D7BCCfont_383)
    - [数据处理函数](file:///C:\Users\admin\Desktop\Untitled-1.html#font_color2D7BCCfont_435)
    - [重要常量](file:///C:\Users\admin\Desktop\Untitled-1.html#font_color2D7BCCfont_461)
* [实例](file:///C:\Users\admin\Desktop\Untitled-1.html#_482)
  + [CGA的高级用法](file:///C:\Users\admin\Desktop\Untitled-1.html#CGA__490)
  + [各样的屋顶](file:///C:\Users\admin\Desktop\Untitled-1.html#_511)
* [辅助工具](file:///C:\Users\admin\Desktop\Untitled-1.html#_550)
  + [ModelHierachy](file:///C:\Users\admin\Desktop\Untitled-1.html#Model_Hierachy_558)
* [CityEngine实例](file:///C:\Users\admin\Desktop\Untitled-1.html#CityEngine_576)
  + [导出为带属性的slpk](file:///C:\Users\admin\Desktop\Untitled-1.html#_slpk_585)

# CityEngine的目录结构

工程中的目录结构如下：

* Assets——存放模型的零件与纹理图片
* Data——存放道路或地块数据（.shp、.osm、.dxf等）
* Images——存放场景快照
* Maps——存放地图图层来源的影像，（.jpg、.img、.tif等）
* Models——导出的3D模型存放位置
* Rules——存放规则文件.cga
* Scenes——存放场景文件.cej
* Scripts——存放脚本文件

https://www.esri.com/en-us/arcgis/products/arcgis-cityengine/resources

https://doc.arcgis.com/en/cityengine/latest/help/cityengine-help-intro.htm

# 基本规则

1、在cga编辑器中，alt+/有智能提示

2、基本规则

* attrheight=250——定义属性
* extrude(height)、extrude(axisWorld,height)——拉伸
* comp(f){top:Roof|side:Facede}——面分割
* comp(e)——边分割
* comp(v)——点分割
* color(1,0,0)或color("#69431a")——颜色

3、返回值，返回值在cga中不以等号形式返回。而是在语句之后添加新的变量名用以承接

4、CityEngine中只支持投影坐标系

5、CityEngine的规则包区分大小写

6、CityEngine2019中坐标方向使用左手坐标系，Y轴向上（垂直方向为Y）

7、CGA的注释使用#、//、/\*\*/

8、CGA的变量定义用attr、常量定义用const

# 开始编写

第一行使用：

Lot-->

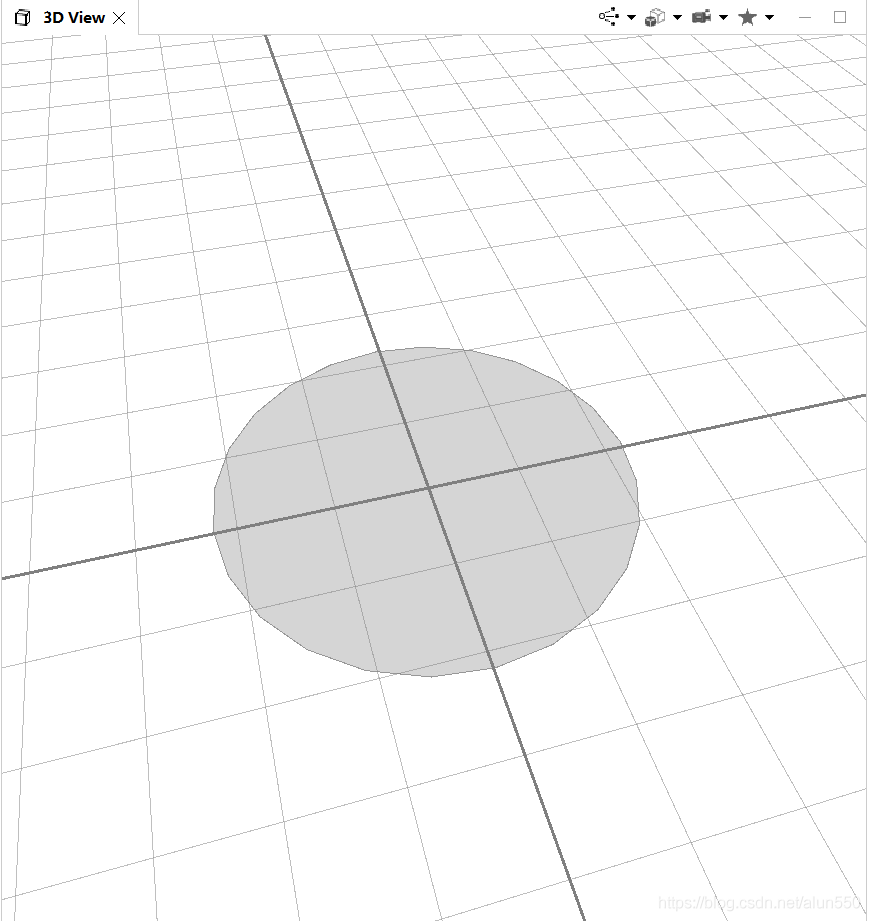


Lot并不具有特殊含义，只是一个比较通用的表达，表示当前的规则文件被拖动到的那个形状上。也可用别的名称代替。

Lot也可被理解为initialshape，即初始化图形，代表图形要被接下来的一系列规则所修改

## 简单的拉伸建筑

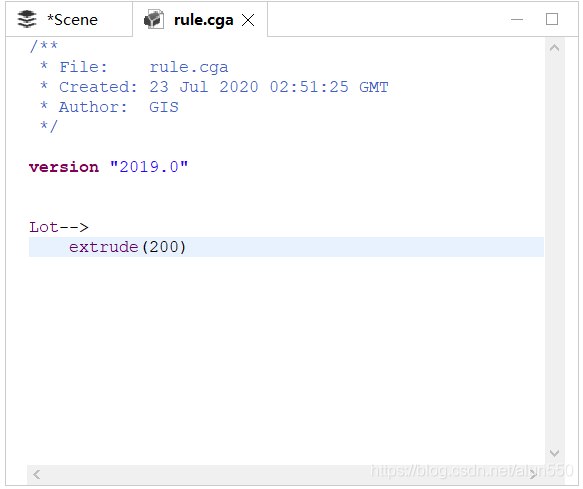
1、画一个简单的面



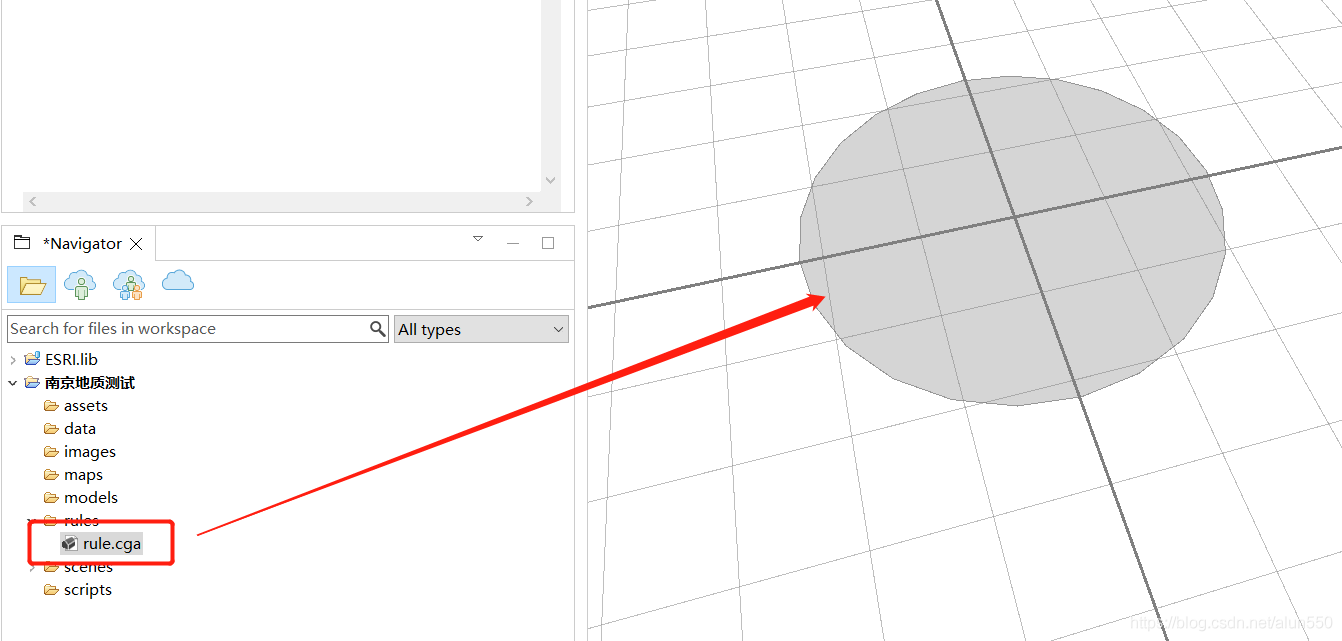
2、编写规则，把面拉起来

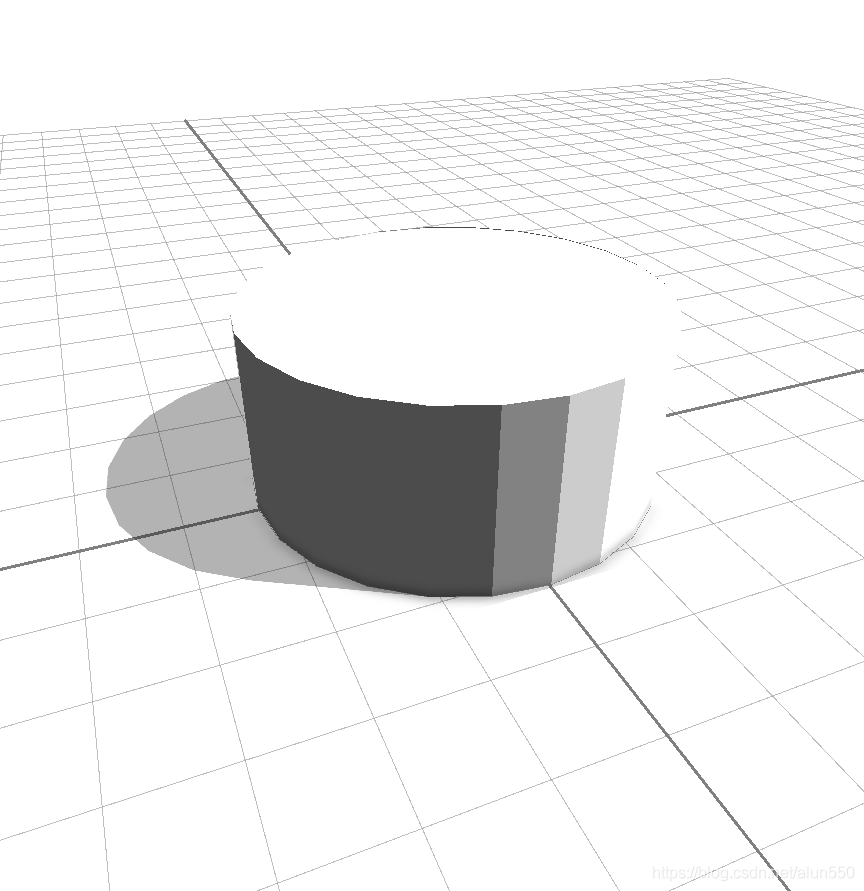
Lot-->

extrude(200)

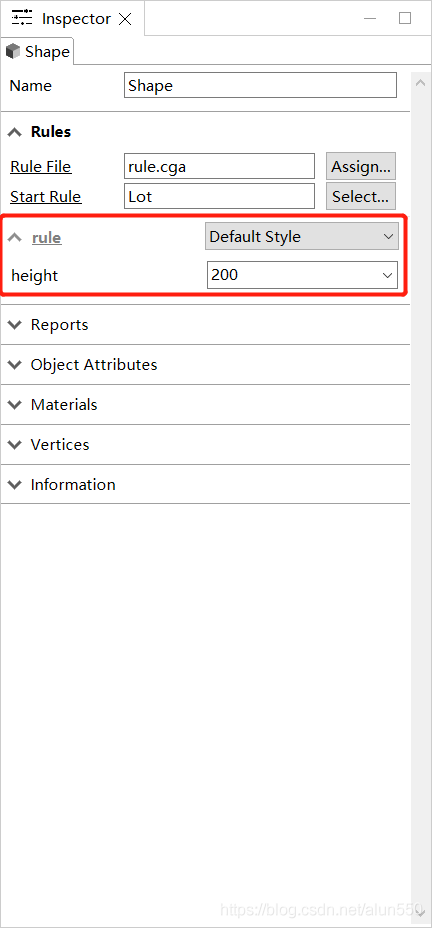


将规则文件拖到圆上





选中物体后，可以看到规则中的高度

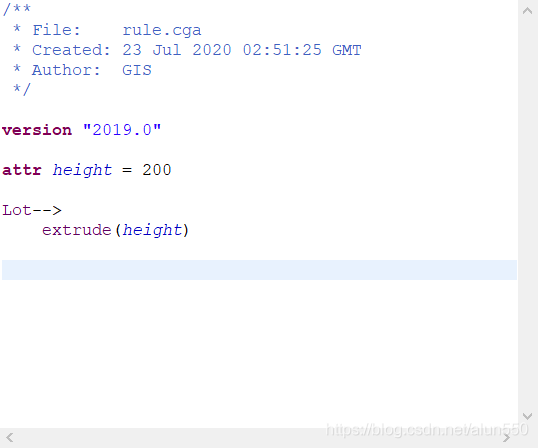


### 创建变量动态修改高度

方法

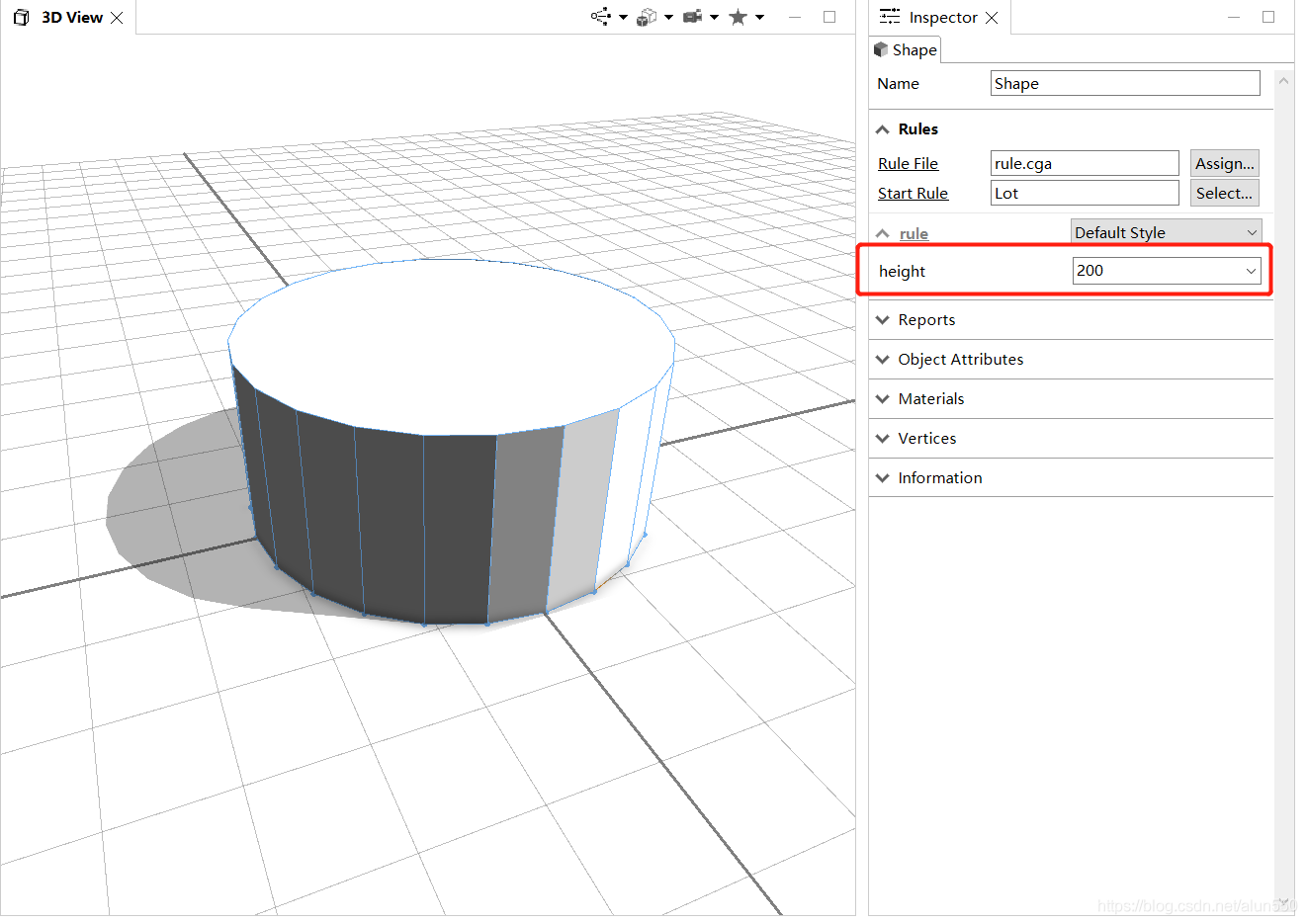
attrheight=20

写于Lot之上



### 控制GUI界面中Rule的高度变化范围

1、未控制前高度可以随意改动

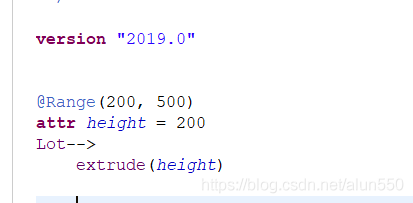


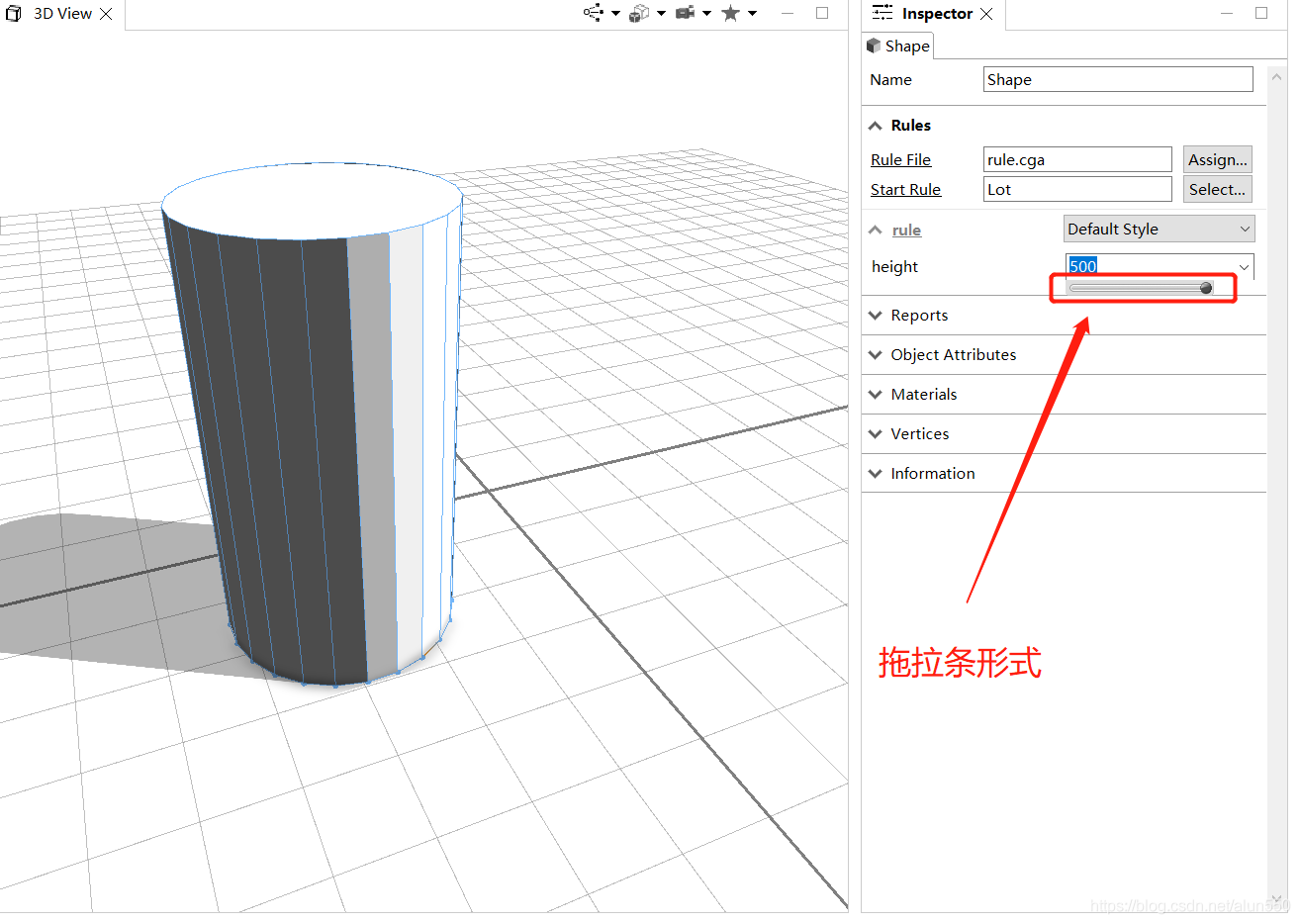
@Range(200,500)

attrheight=200

Lot-->

extrude(height)





### 拆分建筑六面

面分割使用comp(f){}，关键字top、side表示屋顶和侧面，top:roof表示将建筑物整体分割，且顶部赋值至变量roof

@Range(200,500)

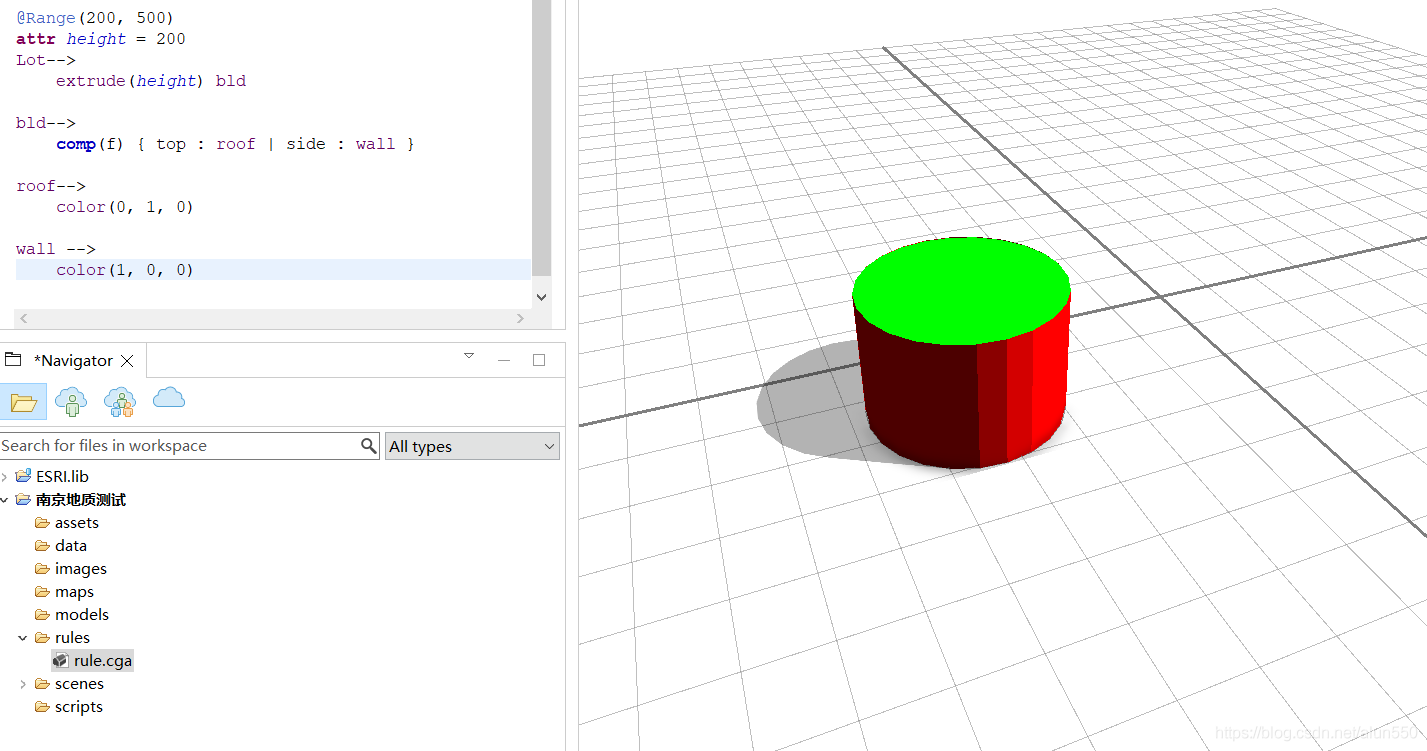
attrheight=200

Lot-->

extrude(height)bld

bld-->

comp(f){top:roof|side:wall}



### 建筑物贴图

1、设置贴图的坐标系

setupProjection(uvset,prjAxesSelector,texWidth,texHeight)

* uvset——要被贴图的元素的索引值，[0,9]的整数
* prjAxesSelector——设置有效的方向轴，有效参数有：scope.xy,scope.xz,scope.yx,scope.yz,scope.zx,scope.zy
* texWidth——宽度方向上的贴图比例，即贴图大小与建筑大小的对应。设置为'1则自适应以一张贴图贴满整个面
* texHeight——高度方向上的贴图比例，即贴图大小与建筑大小的对应。设置为'1则自适应以一张贴图贴满整个面

如：

@Range(200,500)

attrheight=200

Lot-->

extrude(height)bld

bld-->

comp(f){top:roof|side:wall}

roof-->

setupProjection(0,scope.xy,'1,'1)

projectUV(0)

texture("/ESRI.lib/assets/Roofs/Flat/flatRoof\_12.jpg")

wall-->

color(1,0,0)



### 建筑分层

1、基本步骤：

拉伸建筑物-->调用split(y)进行建筑分层

#限定数值范围是[20,50]

@Range(20,50)

#申明变量height，其值为20

attrheight=rand(20,50)

#规则开始运行

Lot-->

#以height作为值拉伸数据，且拉伸后的数据命名为bld

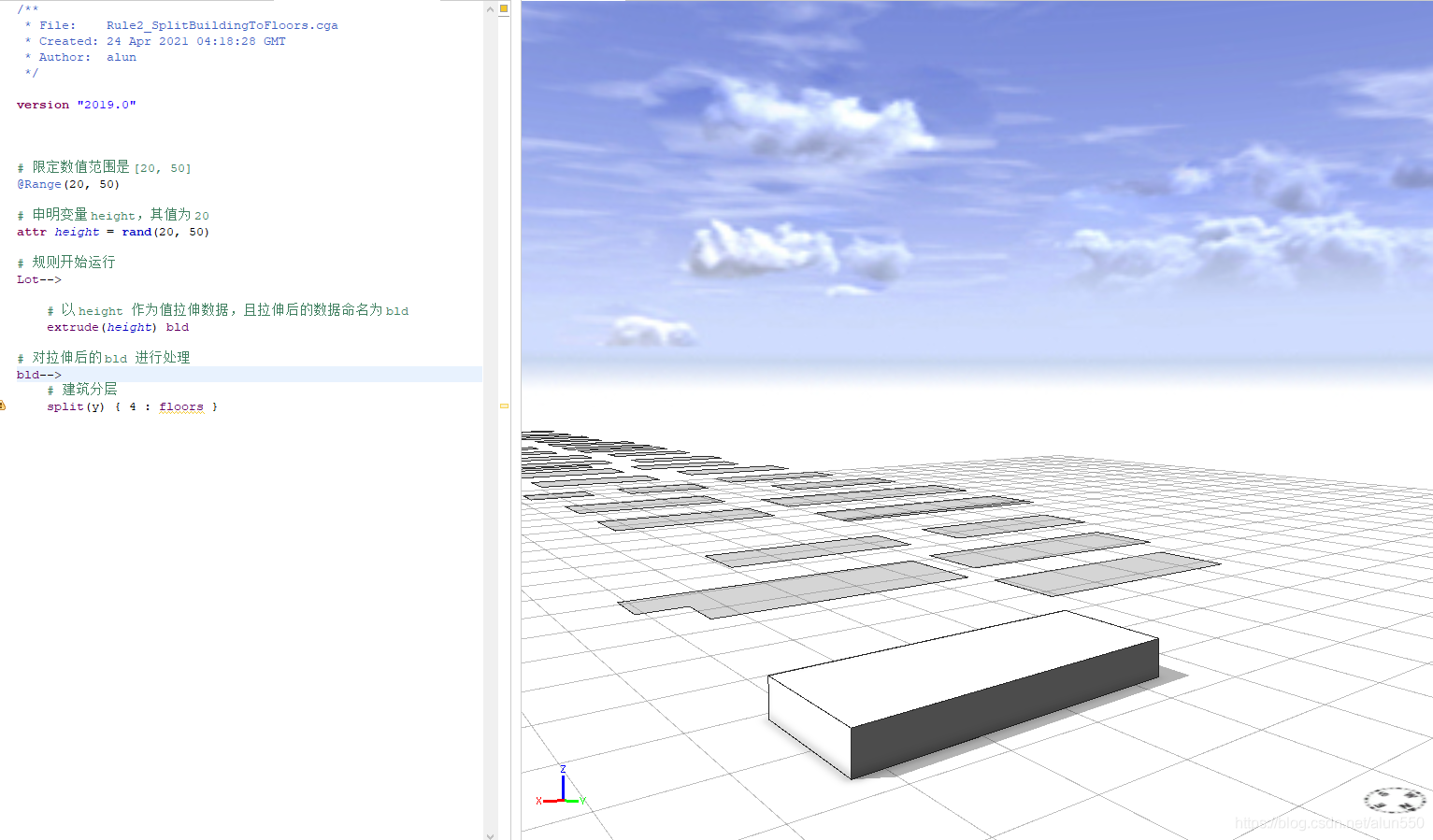
extrude(height)bld

#对拉伸后的bld进行处理

bld-->

#建筑分层

split(y){4:floors}



2、注意事项：

* CityEngine中y轴向上
* 按照上述的规则仅可获取建筑物的第一层，若要获取所有层需要在split(y){4:floors}改为split(y){4:floors}\*，此处的\*代表循环执行

#限定数值范围是[20,50]

@Range(20,50)

#申明变量height，其值为20

attrheight=rand(20,50)

#规则开始运行

Lot-->

#以height作为值拉伸数据，且拉伸后的数据命名为bld

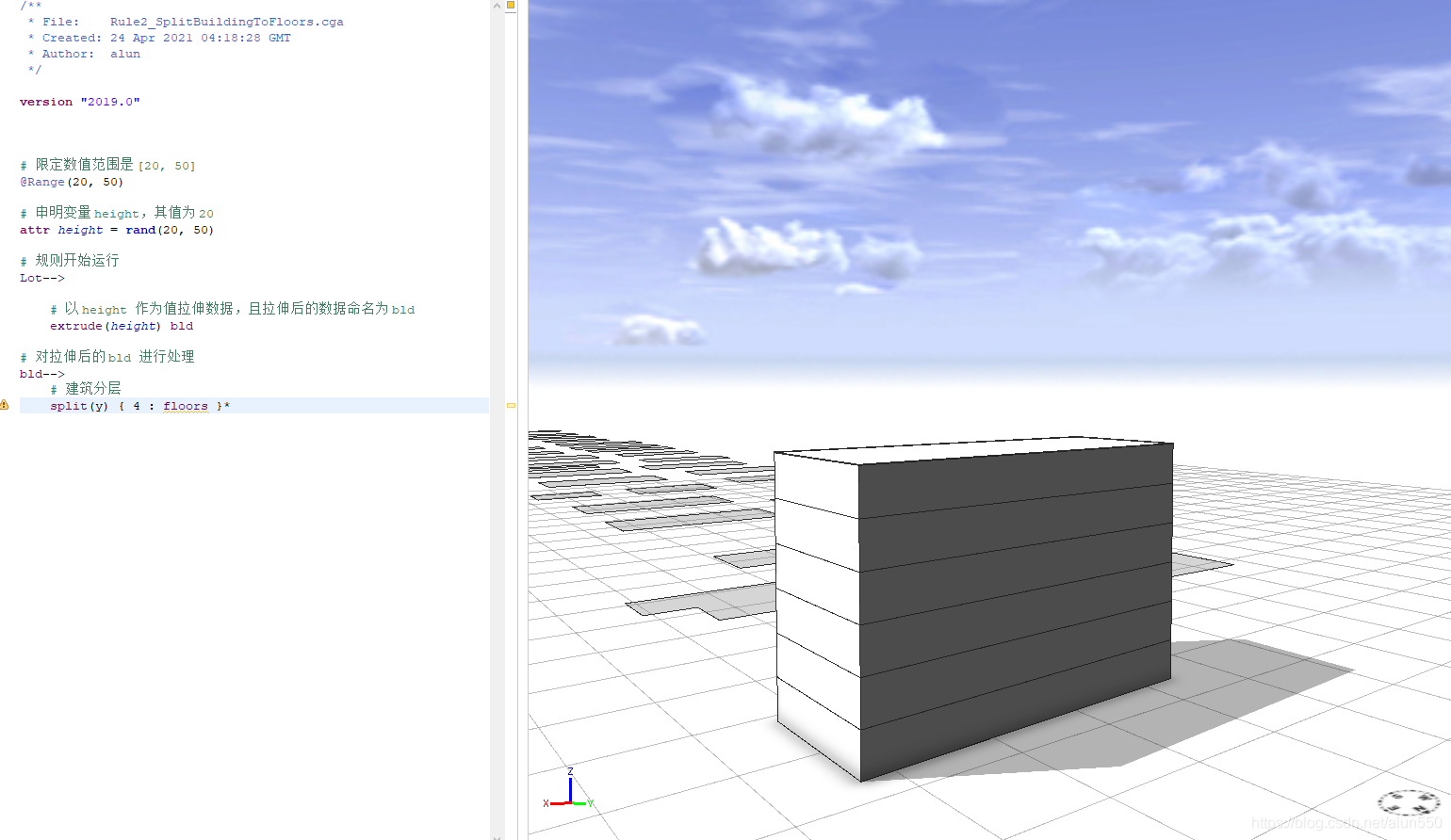
extrude(height)bld

#对拉伸后的bld进行处理

bld-->

#建筑分层

split(y){4:floors}\*



3、分层时容易出现的另外一个问题——由于建筑高度非层高的整数倍，最顶部的一层无法达到分层高度

解决：通过cga规则的求均值来更改每一层的拆分高度

#限定数值范围是[20,50]

@Range(20,50)

#申明变量height，其值为20

attrheight=rand(20,50)

#规则开始运行

Lot-->

#以height作为值拉伸数据，且拉伸后的数据命名为bld

extrude(height)bld

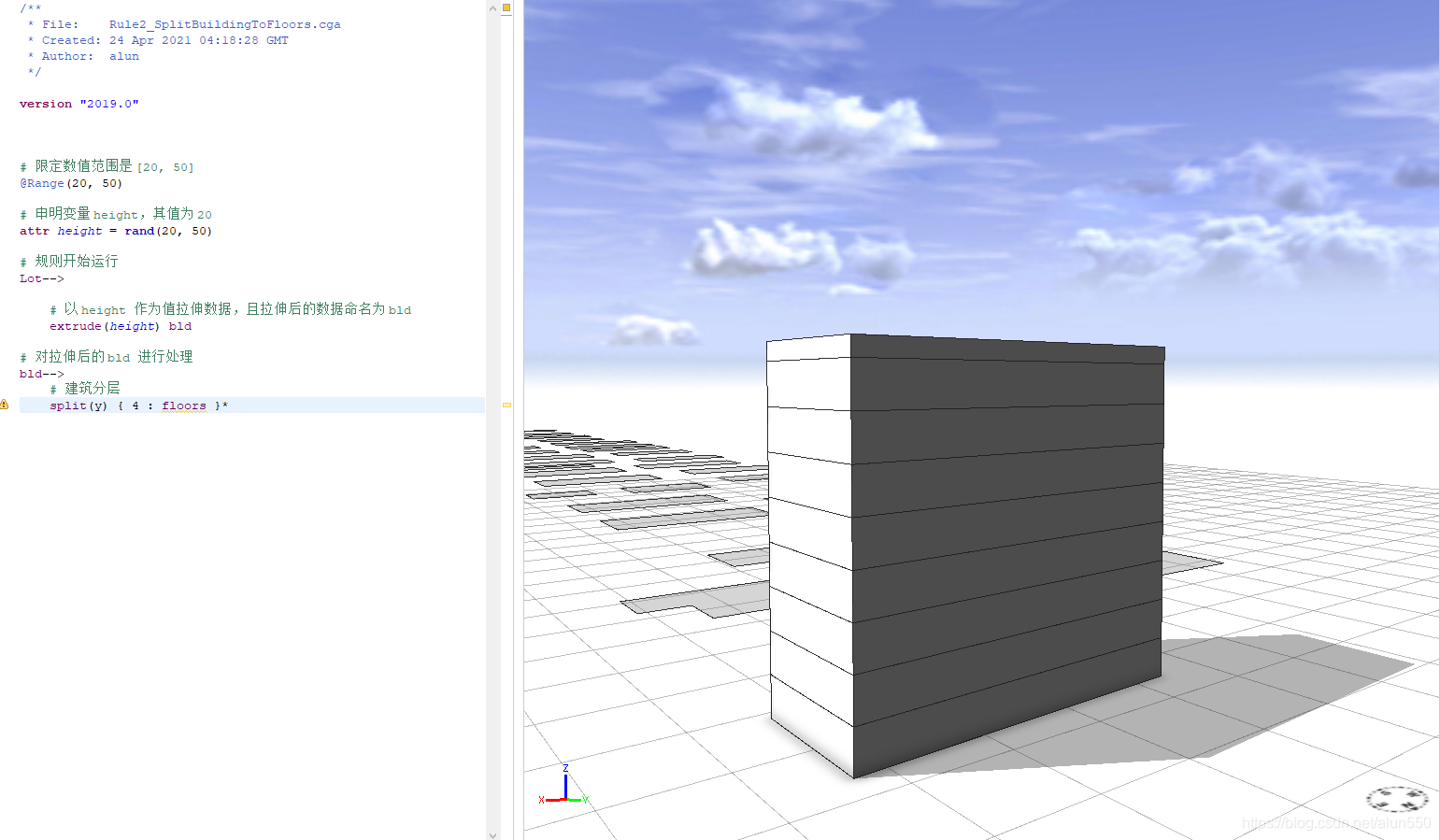
#对拉伸后的bld进行处理

bld-->

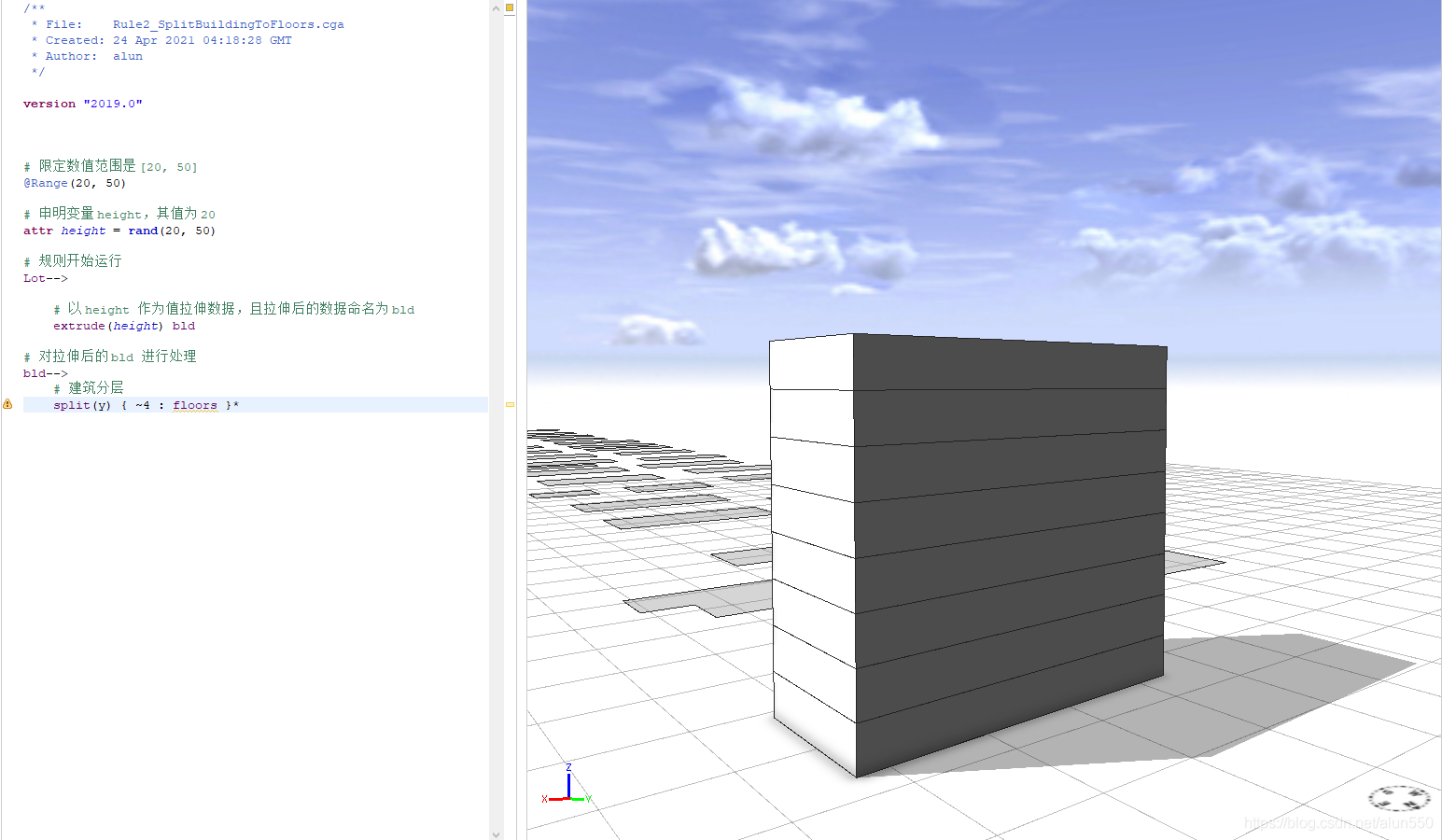
#建筑分层

split(y){~4:floors}\*

使用均值分层前



使用均值分层后



### shape的理解

含义：CityEngine中的shape并非指.shp的shapefile文件，而是指其字面意思——图形。这里的图形是广义图形，既包括二维图形也包括三维图形

属性：

* ShapeSymbol——规则名称
* Parameters——参数
* Attribute——模型的构成与形状
* Geometry——几何对象，包含纹理、颜色、图形
* Scope——外界矩形盒
* Pivot——模型坐标系

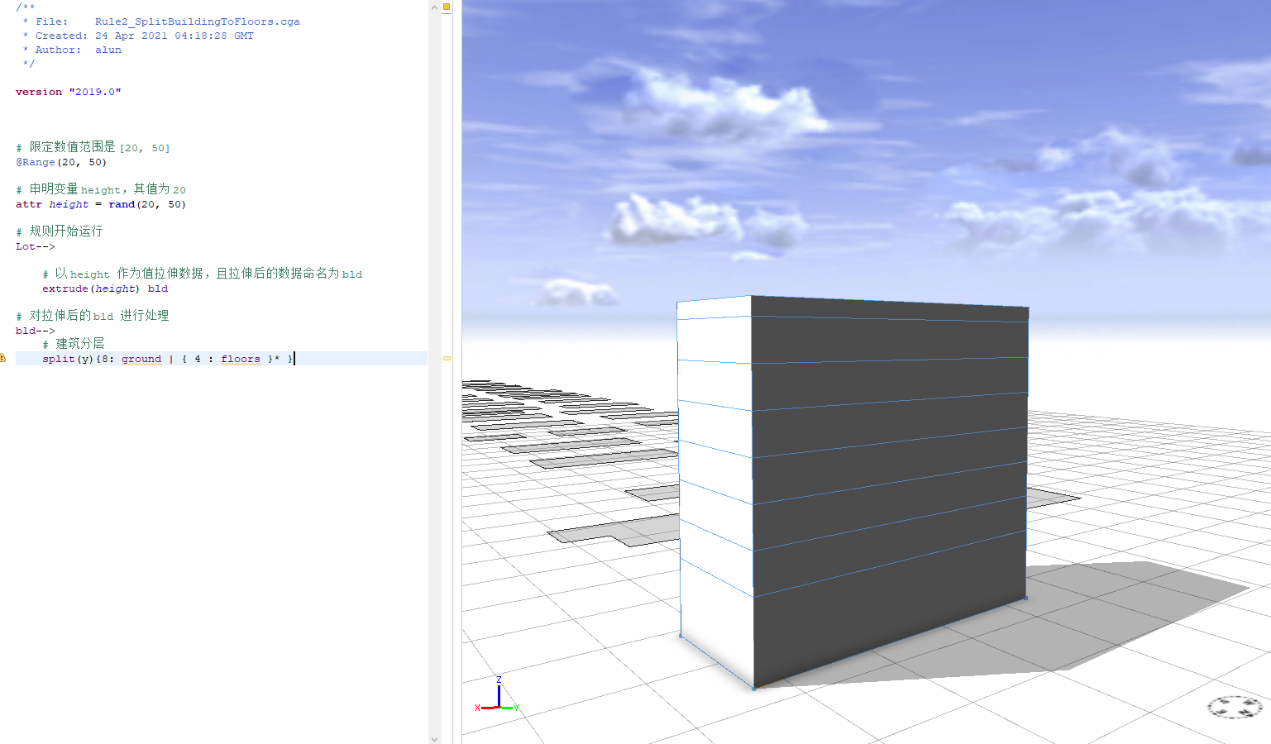
## 常用的函数

### 数据处理函数

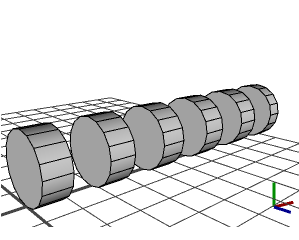
| **函数** | **参数含义** | **效果** |
| --- | --- | --- |
| comp(f){top:roof|side:walls} | 关键字top、side表示屋顶和侧面  top:roof表示将建筑物整体分割，且顶部赋值至变量roof  支持的关键字：top、side、left、right、font、back、bottom… | f—拆分建筑物顶部和侧部  e—拆分边  v—节点分割  fe—面上的边  g—组分割  m—材质分割  h—洞分割 |
| setupProjection(uvset,axes,width,height) | uvset—被操作的uv层的索引值，取值[0,5]指代材质属性  axes—指定原点及哪些轴被指定为u轴或者v轴  width—贴图的宽度，支持~和`（按照百分比去贴）  height—贴图的高度，支持~和` | 初始化基准坐标系统与uv集的投影矩阵，用于设置图片贴图的矩阵 |
| projectUV(uvset) | uvset—被操作的uv层的索引值，取值[0,5]指代材质属性  彩色贴图具有uvset0、凹凸贴图具有uvset1、污垢贴图具有uvset2 | 创建纹理图片坐标系 |
| texture(“picture\_url”) | 略 | 指定并执行具体贴图 |
| split(axes){length:new\_name} | axes—轴向（x,y,z）  length—该轴向上要切割的长度 | 用于切分模型，仅执行一次 |
| split(axes){length:new\_name}\* | 同上 | 用于切分模型，循环执行，直至超出模型范围 |
| i(“the\_utl\_of\_3dmodel”) | 模型地址 | 用三维模型（一般是obj）替换现有的模型结构 |
| roofGable(height,length,width) | height—屋顶的高度  length—建筑长边的挑檐长度  width—建筑短边的挑檐长度 | 尖顶房屋（带额头的大中分） |
| roofHip(height,outlength) | height—屋顶的高度  outlength—延伸出来的屋檐长度 | 尖顶房屋（有刘海的西瓜头） |
| roofPyramid() |  | 屋顶样式控制 |
| roofShed(height) | height—屋顶的高度 | 斜坡屋顶 |
| shapeL() |  | 分割房屋边界 |
| shapeU() |  | 分割房屋边界 |
| shapeO() |  | 分割房屋边界 |
| Scatter() |  | 种植树木 |

split(){}的复杂用法：

* 常规分割和重复分割组合使用——split(y){2:ground|{~3:floors}\*}



* 隔一层删除一层（在spilit(){}的变量命名中使用NIL）——split(x){{~0.75:xx|~1:NIL}\*|~0.5:xx}

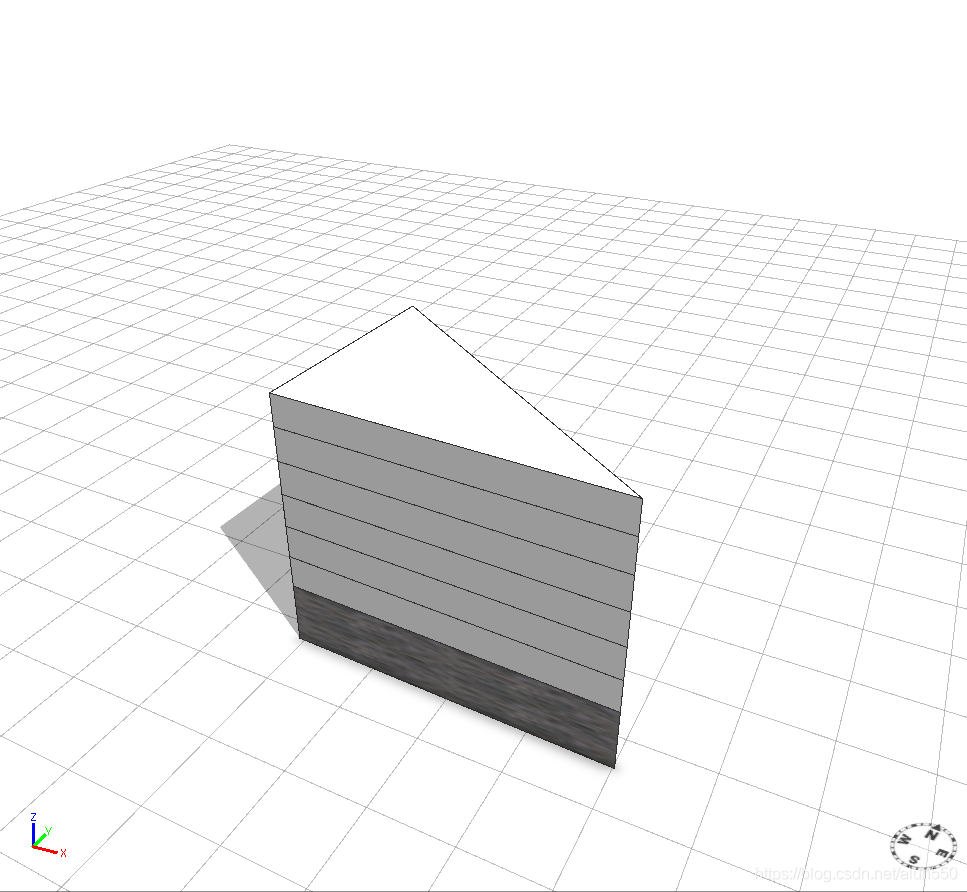


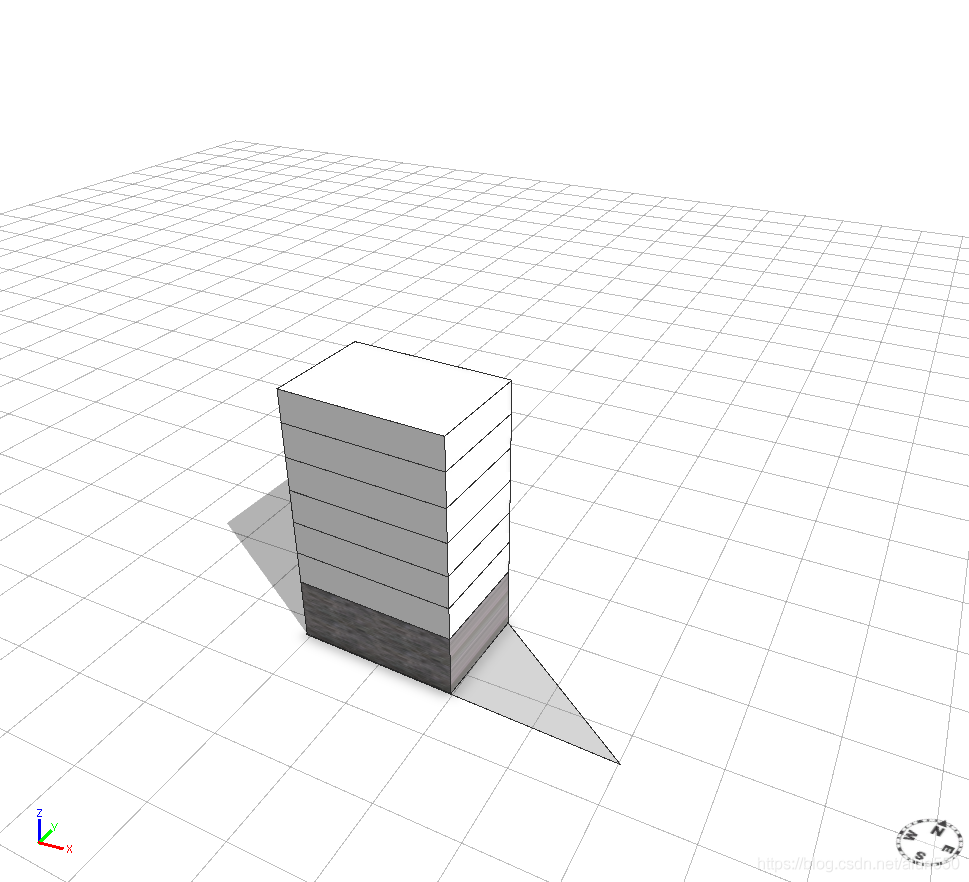
### 数据处理函数

| **函数** | **参数含义** | **效果** |
| --- | --- | --- |
| rand(min,max) | 最小最大值 | 获取值域范围的一个随机值 |

### 重要常量

* NIL——删除此部分（对谁调用就删除谁）
* innerRect——将Shape转化为内接矩形，建立规则的房屋（如对三角形的地块，构建矩形房屋会被裁剪成三角形。如果使用innerRect就可将地块的形状，以其内接矩形作为规则应用的区域）





# 实例

## CGA的高级用法

以百分比划分区域，并分类进行规则引用

描述：如将一块区域按照20%,80%进行随机划分，20%种树80%建造房屋，此时可用如下语法

Lot-->

#将20%的区域划分给变量planttree

20%:planttree

#将其余的划分给变量building

else:building

## 各样的屋顶

尖顶房屋

@range(20,50)

attrheight=rand(20,50)

Lot-->

extrude(height)bld

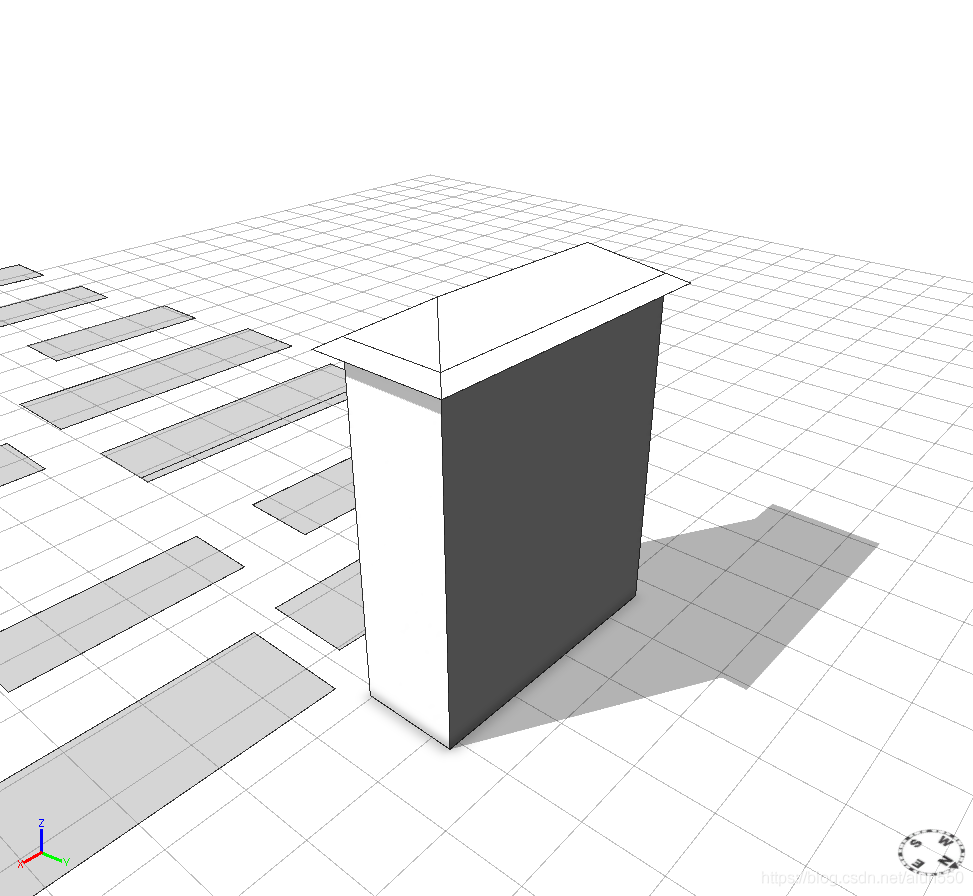
bld-->

comp(f){top:roof|side:walls}

roof-->

#尖顶房屋

roofHip(30,2.5)

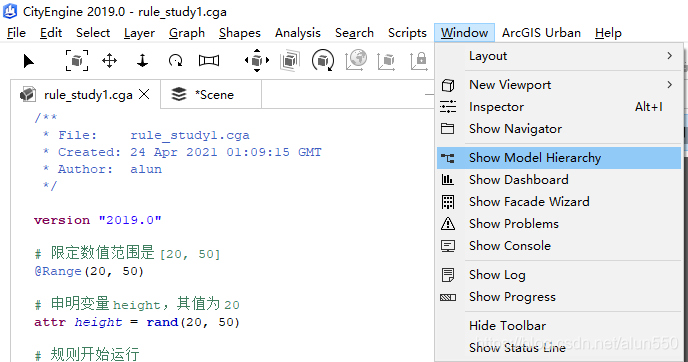


# 辅助工具

## ModelHierachy

功能：用于可视化的显示cga规则文件的流程

打开方式：Window——ShowModelHierachy



# CityEngine实例

## 导出为带属性的slpk

原始数据具有5个属性（OBJECTID、name、x、y、height），若直接从cityengine导出为slpk则全部属性都会丢失，通过如下方式保留其属性

1、导入gdb中三维数据

2、编写cga以保留属性

version"2019.0"

attrOBJECTID=1

attrheight=1

attrname=""

attrx=1

attry=1

@StartRule

Lot-->

report("height",geometry.height)

report("name",name)

report("OBJECTID",OBJECTID)

report("x",x)

report("y",y)

3、全选所有的三维模型，并应用inspector中应用规则文件

4、导出为slpk文件