Grasshopper 完全学习手册 V1.0

Form | Algorithm

www.jinjieming.com

交流群群号

312673316

参数化设计

建筑学留学申请辅导

建筑学作品集制作及资讯分享

欧美高校建筑学设计及资讯分享

版权声明

本书创作者名单（排名不分先后）：
首先是 E 拓参数化论坛的三位版主：流风 Mario Chester.L
以及：Nightawk 皮佳 善良的北 Burt
当然还有我 DanielJin
排版，美工，校对，后期，勘误等等等等也都是我 T.T
由于本书工作量巨大 206 页 32657 字
为了赶在中秋节前九月初发布送给大家
时间紧迫 任务艰巨 人手不足
所以美工和排版极其烂
书中不足和错误也可能有很多
请各位海涵！
感谢！

序

虽然我知道你们几乎不会有人看序的，但是作为一本完整的电子书，还是随手写一点东西吧。经过了一个月的构思，决定写这本书的想法逐渐成熟。原因有三点。

（1）现在国内没有一本很好很完全的集所有运算器于一身的 Grasshopper 中文说明书。大家普遍在用的基本都是很早以前 NCF 论坛前辈们所创作的《Grasshopper 运算器教程 V1.1》。后来由于前辈们有各自的事情要忙，NCF 的教程 V2.0 只完成了很小的一部分就没有了下文。前一段看到一位前辈说，每个年代都应该有每个年代的 NCF，作为 E 拓参数化论坛的创始人，我认为我应当担负起这个责任，虽然我只是一个还未毕业刚步入大五几天的小屁孩子。

（2）E 拓参数化论坛（csh.eeeetop.com）从最开始只是 E 拓建筑网（www.eeeetop.com）的一个小版块，逐渐发展壮大。很开心我在这个网站里认识了很多志同道合的朋友，他们纷纷和我一样发一些帖子，心得，感悟，让 E 拓参数化大家庭越来越温暖。随着 8 月初我将论坛域名独立出来，论坛的每一步成长和成熟我都看在眼里。然而很遗憾的是，我们除了原创了一些教程以外，完全拿不出可以成为标杆性的成果给大家分享。同样作为参数化论坛，很多网站上的前辈们，都要比我们的创作人员专业技术强；我们的 UI 也没有一些网站好看；背后的财力支持也不如一些商业化运转的论坛。我常常想，我们有这么多比不上别人的地方，那么我们怎么把 E 拓参数化论坛做好呢？想来想去，我们只有比别人更努力付出，也许才能获得大家的认同。因此我邀请论坛上的几位版主和现实生活中的几位好友帮忙，一起来写下这本书，希望能当做我们为 E 拓参数化论坛略尽一份绵薄之力吧！

（3）网络上流传了很多关于 Grasshopper 运算器的总结，从只言片语的帖子，到 gh 格式的文件，再到整理过的汇总图片等。但是一直没有人愿意站出来将这些网络资源整合，提供给读者。既然已经分析到了，我们的实力不如很多前辈所在的论坛，那么我们更应该付出比别人多的努力。因此我们将网上免费提供给大家的资源进行了整合，重新整理，更新了所有计算器的用法并且删除了已经过时的内容，提供给各位读者。

接下来说说构思结束后的创作过程吧！

随着想法成熟以后，我的托福考试也逐渐临近。我在着手开始准备编著工作的同时还要忙网站的事，同时还要复习考试，当时真的是觉得一天为什么只有 24 个小时呢完全不够用啊。和我一起参与编著此书的作者们也十分忙碌，他们常常白天要加班晚上还要熬夜写到一两点，好不容易到了周末还不能休息要努力赶进度。就是这样昼夜不分的拼搏了两周，才最终得以将这本书在 9 月初当做中秋节礼物呈献给各位。这本书中的一些资料是源于 NCF 论坛前辈们编写的《Grasshopper 运算器教程 V1.1》，一些是来自各种渠道的略显粗糙的翻译，在这里我要向所有参考资料的原作者致以最崇高的敬意，正是他们的免费开源得以孕育了这本书的出生。也正是因此原因，我决定将本书免费发布在 E 拓参数化论坛 csh.eeeetop.com 上。任何想要学习的人都是值得尊敬的。

当然我知道你们不怎么看序的，你们也不必感动我们起早贪黑的努力，因为对于许多人来说，免费资料就意味着廉价的劳动力不应该被珍惜，相反那些收费的课程才有存在的意义。能够看到这里的，我代表 E 拓参数化论坛和 E 拓建筑网感谢您对一本免费书籍的重视，也由衷感谢您对我们给予了最起码的尊重。

这本书之后的发展

我们的本意是这本书出了以后不应该就此搁置。每当有变动较大的 Grasshopper 新版本出现时，我们都应该进行校对和重新修订。书名后的版本号代表代表新旧情况。

同时，E 拓参数化论坛（csh.eeeetop.com）上的教程将会和本书进行协调，每当一些教程中有了更详细的阐述时，我们都会在修订时将帖子名称加入本书供大家查看。

这本书出的十分仓促，因此一定会存在很多不足和错误。欢迎您勘误和校对，并登陆 csh.eeeetop.com 在置顶区的本书帖子中，给我留言或发站内信。我们将会由衷感谢并将您的反馈一并放入下一期修订版本中。

致谢

首先，感谢所有参与编著《Grasshopper 运算器教程 V1.1》的所有来自 NCF 的前辈们。没有他们的努力，写这本书的想法不会成型，这本书也不会面世。正是因为前辈们之前的铺垫，我们这些晚辈才可以编著出此书。我代表所有参与创作的人员向 NCF 的所有前辈致以最崇高的敬意！

其次，要感谢之前所有创作过 Grasshopper 相关教程并免费发布于网上的作者，没有您之前的研究和大方共享，同样这本书也无法成型。

接下来，我要感谢所有我团队的所有人。他们多数还是和我一样的大四大五的学生，有些前辈刚刚参加工作不久。团队中的绝大多数人每天都要忙着做竞赛，实习加班，还有鄙人要准备托福考试申请出国。在这本书的两周编写期内，所有人都是挑灯夜战牺牲自己的睡眠时间和课余时间参与编著。无论每个人参与了多少工作，我都向你们致以我个人的谢意。

感谢流风（论坛 ID：沧月），和我一样刚刚进入大五，正在实习，每天要忙着加班晚上回来还要熬夜参与创作，他独立完成了 Surface 和 Display 章节。谢谢您！

感谢 Mario 前辈，经常要加班还是参与了创作。他独立完成了 Param 章节。谢谢您！

感谢 Chester.L 前辈（论坛 ID：zhiaixu2010），也是在工作之余独立完成了 Mesh 章节，同时给我解答了不少疑问让我学习到很多新的知识,并参与了校对。谢谢您！

感谢 Nightawk，他和我一样在申请出国，独立完成了 Intersect 章节。谢谢您！

感谢皮佳，他虽然开学才大四，但是技术上是我的老师！他因为家里有一些事脱不开身，仍然完成了 Vector 的部分章节和 Set 章节的 Tree 部分。感谢您！

感谢善良的北前辈（论坛 ID：443792128）在工作之余帮我完成了 Set 章节的 Array 部分。谢谢您！

当然还要感谢来自我私人好友，Burt 的帮助！他在实习考试前一夜帮我完成了 Set 章节的 Util 部分，第二天一早依旧考了满分！简直是 V587 不解释！

同时感谢所有帮助解答我编写本书时提出的各种幼稚疑问的人，比如 UncleX。

借以此书，感谢对 E 拓参数化论坛提供大力帮助的 E 拓建筑网的站长@深圳老梁。感谢梁哥如此信任我，支持我，鼓励我。

感谢我的学弟@朋朋，他帮我搞定了很多视频教程美化的工作，并帮助我进行了很多网站线下的工作。

感谢我的学弟@离境之尘，他帮我进行了很多网站线下的工作。

感谢我的女朋友给我自由让我创作这本教程。

最后，感谢所有能耐心看到这里的人，感谢您对我们给予了尊重！让我们每天熬得夜受的苦和幸福感相比，微不足道。

By DanielJin
矮丑穷挫黑宅土
你们的小黄人君 :-)
2014-8-31 凌晨 3:37

0. 准备篇

——如何通过此书来完整的了解，学习 Grasshopper？

那么首先简单介绍一下这本书吧。这本书将会全面的介绍 Grasshopper 这个软件。无论是从界面的最基本介绍，还是到各种运算器的介绍都会涵盖。运算器使用的版本是最新的 0.9.0075 原生版本，未包含任何插件（在发布前 3 天的时间传来了噩耗，0076 版本上市了，呵呵呵...），涵盖了所有的运算器，包括运算器的输入端和输出端分别代表什么，应该输入什么类型的数据，以及运算器的用途，如果官方英文直译不通时还会标注实际用法的通俗解释。同时对于很多运算器加入了实际的小的电池演示图示，方便大家更快的理解每个运算器应该如何参与整体的运算。当然，有些非常基础或者非常类似的运算器和输入端，我没有全部都写出，比如 Curve 类运算器，所有接入端 C 几乎都代表输入曲线，在这样的情况下我通常只详细讲解部分具有代表性的输入端，其余类似的输入端不再赘述。当然，所有的运算器都有进行讲解，没有说因为功能类似就直接忽略不讲的。每当遇到极其复杂的运算器时，如果只靠图示不能明白阐述，我通常会在后边用红色标注详细案例请登录 E 拓参数化论坛 csh.eeeetop.com 查看某贴。您可以通过登陆论坛得到更多资源教程。当然，这本书还有很多不足和错误，比如很多运算器我自己都不知道如何用“人话”清楚明白的解释清楚，只能生硬的翻译配上图示希望能帮助大家理解。有些运算器我的理解也十分局限，我通常也会标注上致歉，并邀请您登陆 csh.eeeetop.com 给我发站内信进行勘误和校正。

想要学习 Grasshopper，大致分为这样的顺序：

（1）最基础的部分，比如界面讲解，请登录 csh.eeeetop.com 查看由我代为发布的【SEG】系列教程的基础篇部分。里面对整个 grasshopper 的详细设置都有所讲解。

（2）了解各电池的作用，方法是通读这本书，做到对运算器的大概用法有个粗略的印象即可。

（3）登陆 csh.eeeetop.com 的置顶区，观看由我代发的 UncleX 所录制的视频教程“小苹果系列”:《GH 负基础教学实例之苹果 logo》。UncleX 用苹果 logo 作为例子由浅入深讲解了如何将运算器相互结合做出越来越复杂的苹果 logo。你们会发现最后的立体苹果 logo 和最初版本的平面 logo 相比较，已经有很深的的难度了。当然，看这个教程请自行忽略掉 UncleX 黑我拿我开涮的部分，谢谢！

（4）登陆 csh.eeeetop.com，将系列教程推荐区的教程认真读完，并认真做课后作业。很多人看完我以小黄人为头像发布的【By DanielJin】系列教程然后加入群中总是问很多问题，这些问题一看就是没有按部就班的学习和思考就提问的假大空的问题。我常说，学习必须踏踏实实脚踏实地按部就班，自己才是最好的老师。浅尝辄问不会被别人尊敬，与君共勉！

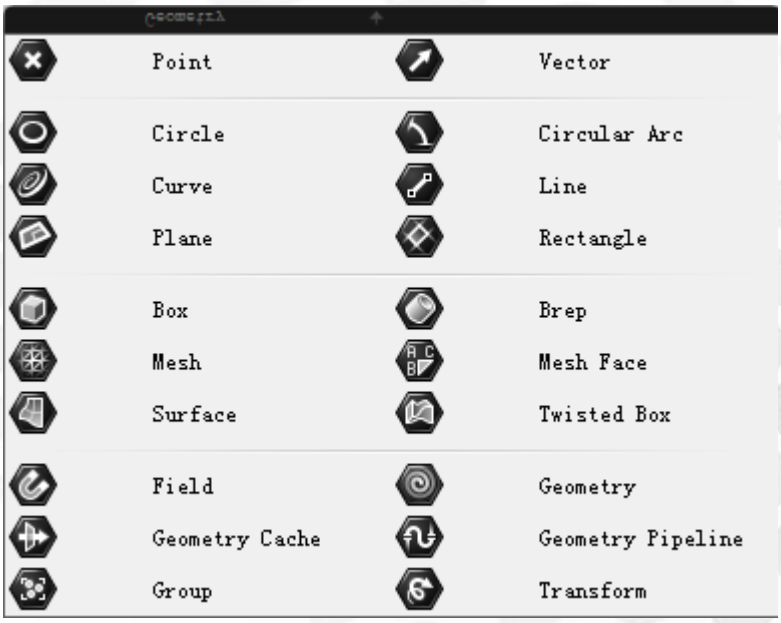
（5）登陆 csh.eeeetop.com 完成我和流风（ID：沧月）老师布置的思考题。如果你能独立完成 Voronoi 全局排序，恭喜你，你已经不能算初学者了。

（6）师傅领进门，修行靠个人，往后关于 Grasshopper 的学习之路，鄙人拙见是您不应该再浪费太多金钱用来学 Grasshopper。到了这个阶段，最好的学习方法就是动手做，硬着头皮做，静下心来逼着自己做。

（7）E 拓参数化论坛将会随后推出每一个小运算器的单独视频教程，如果只靠本书有看不明白的地方，欢迎您登陆 csh.eeeetop.com 关注我们随后推出的视频教程。

1. Params 电池组

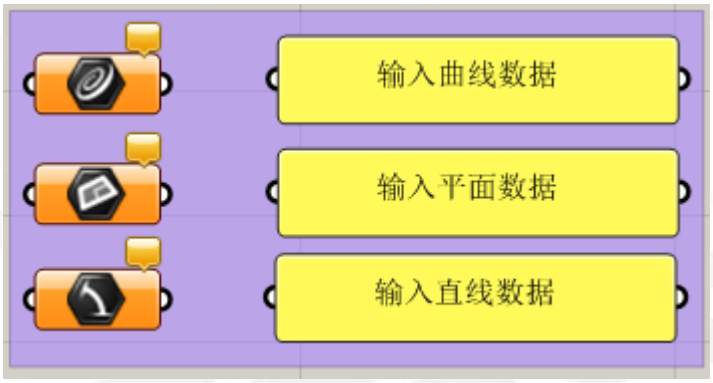
（1）Geometry 电池组



这一组都是对数据的抓取,电池都有左侧输入端和右侧输出端
都有两种输入数据的方法，一种是把相应数据连接到左侧输入端，另一种是电池上点右键 Set one XXX，新设置一个 XXX。Set multiple XXX,即设置多个。但是 Set one curve 只能选取 Rhino 中创建好的。
左侧输入端：任何相应属性数据
右侧输出端：电池所包含的相应属性数据

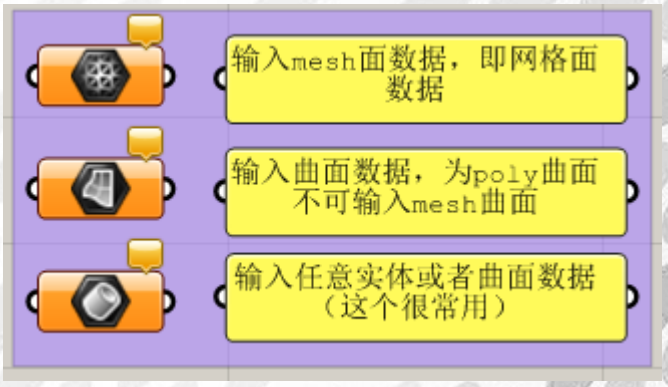
属性对应如下：

Point:：输入点数据
Vector：输入向量数据
Circle:输入圆数据，这个电池只包含圆相关曲线

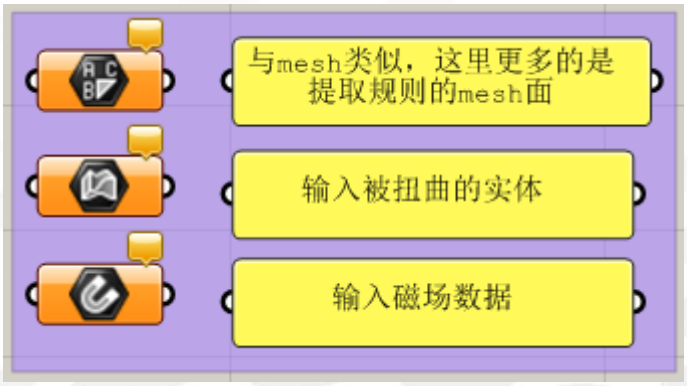


Curve:输入曲线数据
Plane:输入平面数据
Circular Arc:输入圆弧数据

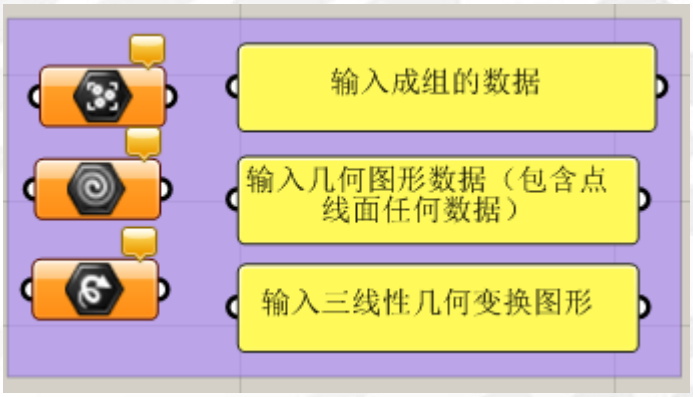
Line:输入直线数据
Rectangle:输入网格数据
Box:输入实体盒子数据



Mesh:输入 mesh 面数据，即网格面数据
Surface:输入曲面数据，为 poly 曲面，不可输入 mesh 曲面
Brep:输入任意实体或者曲面数据（这个很常用）



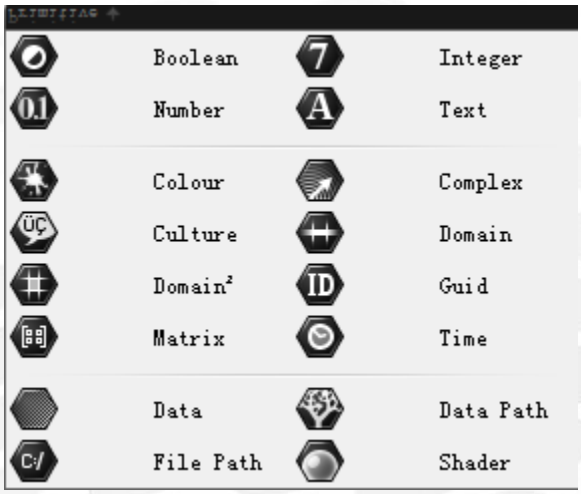
Mesh Face:与 mesh 类似，这里更多的是提取规则的 mesh 面
Twisted Box:输入被扭曲的实体
Field：输入磁场数据

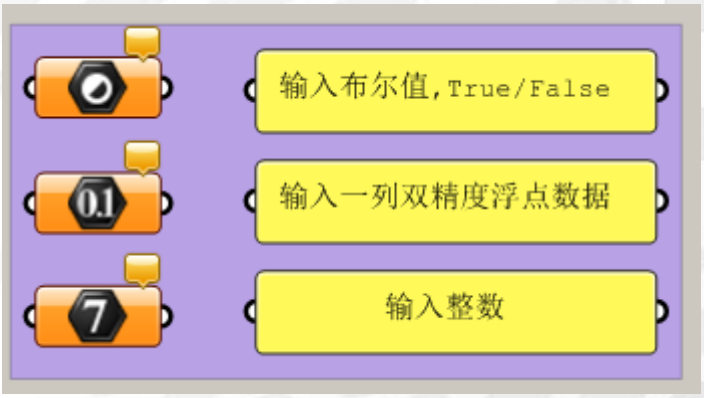


Group：输入成组的数据
Geometry：输入几何图形数据（包含点线面任何数据）
Transform:输入三线性几何变换图形

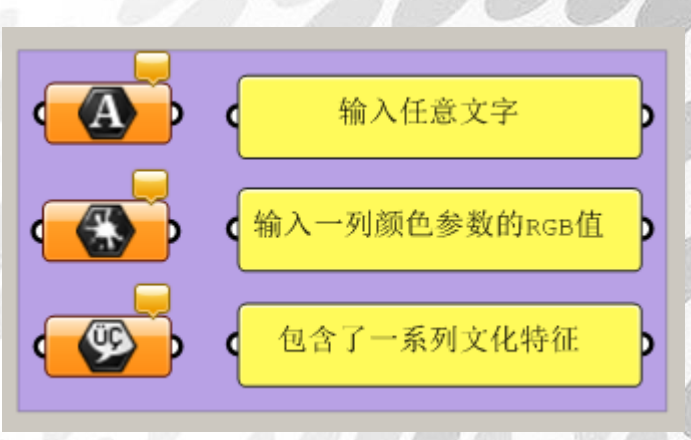
Geometry Pipeline:从犀牛中输入几何管线到 GH 中
Geometry Cache：物体缓存
主要作用：1 快速烘焙 GH 中的物体 2 快速选择已经烘焙到 Rhino 中的物体（Chester.L）

（2）Primitive 电池组:

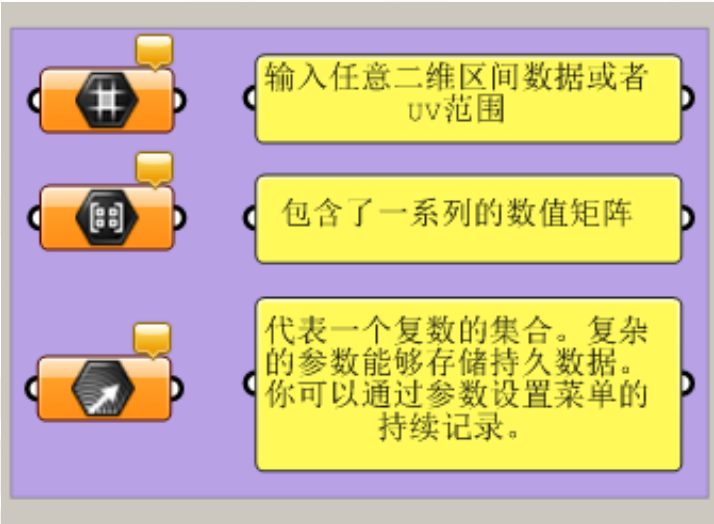




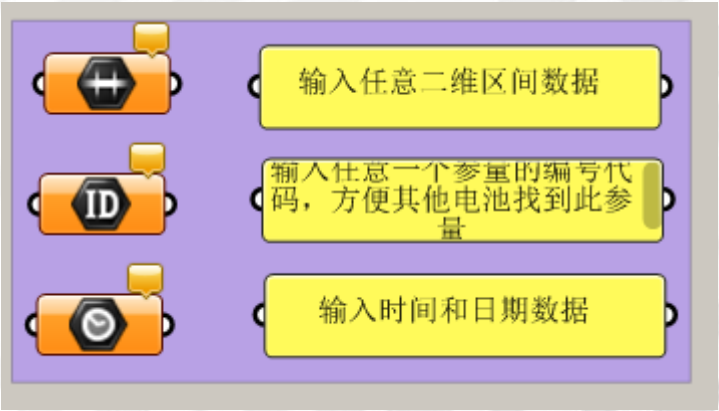
Boolean:输入布尔值
Integer:输入整数
Number:输入一列双精度浮点数据



Text:输入任意文字
Color:输入一列颜色参数的 RGB 值
Culture:包含了一系列文化特征



Domain²:输入任意二维区间数据或者 UV 范围
Matrix:包含了一系列的数值矩阵
Complex:代表一个复数的集合。复杂的参数能够存储持久数据。你可以通过参数设置菜单的持续记录。

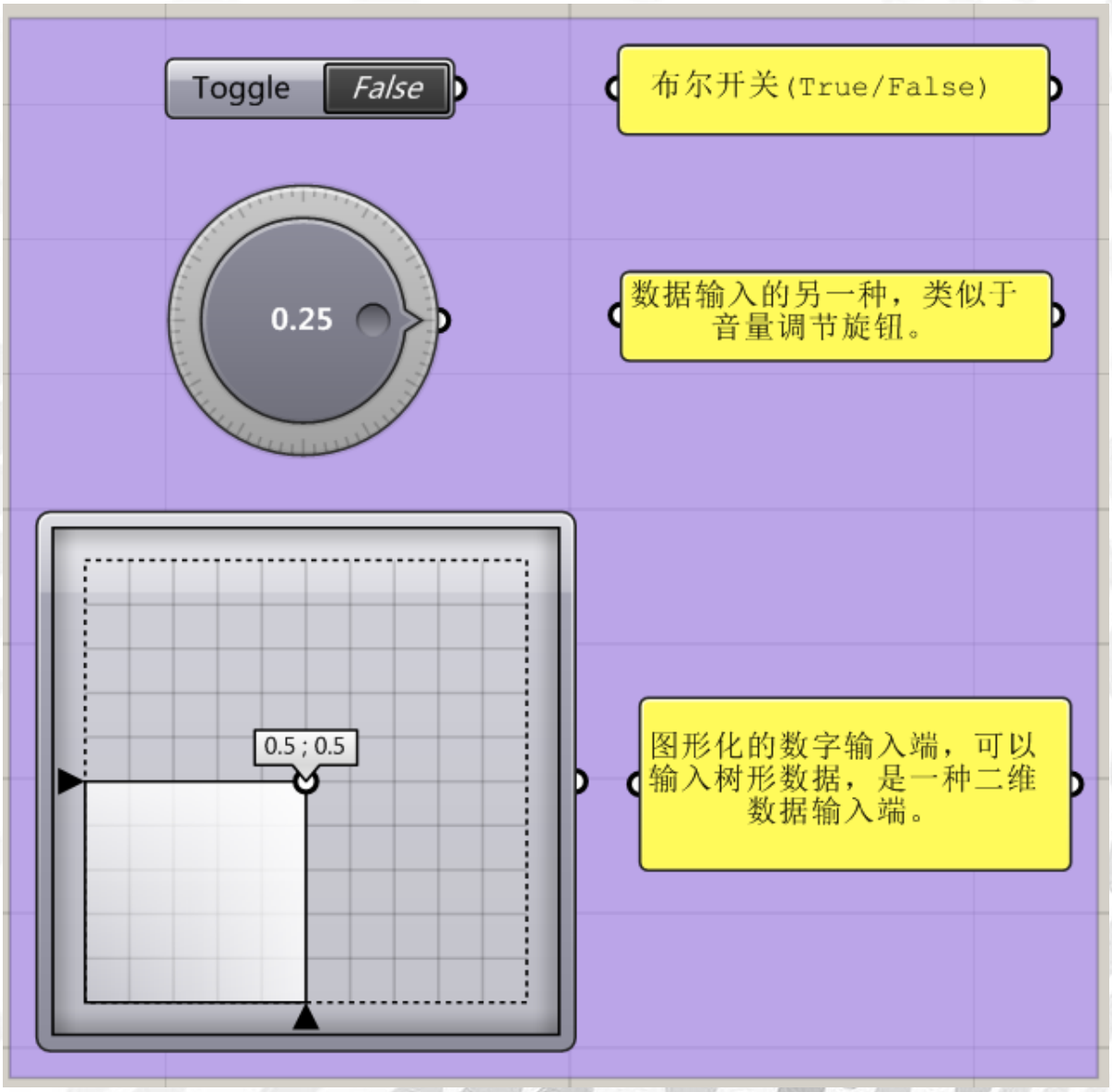


Domain:输入任意二维区间数据
Guide:输入任意一个参量的编号代码，方便其他电池找到此参量
Time:输入时间和日期数据

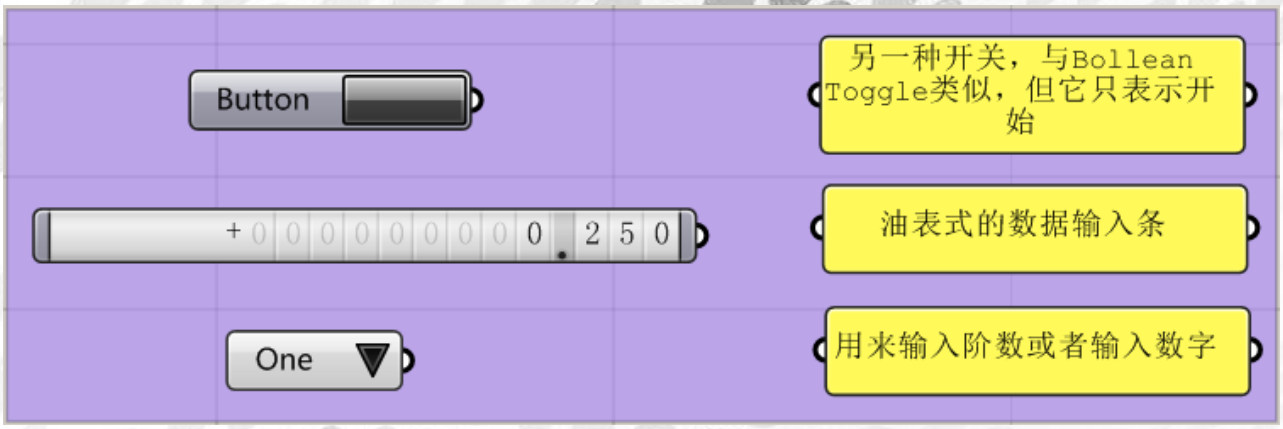
Data:输入任何一列参量
File Path:用于输入硬盘中某个地址的文件
Data Path:通过路径输入一列数据
Shader:输入一列渲染值

（3）Input 电池组：

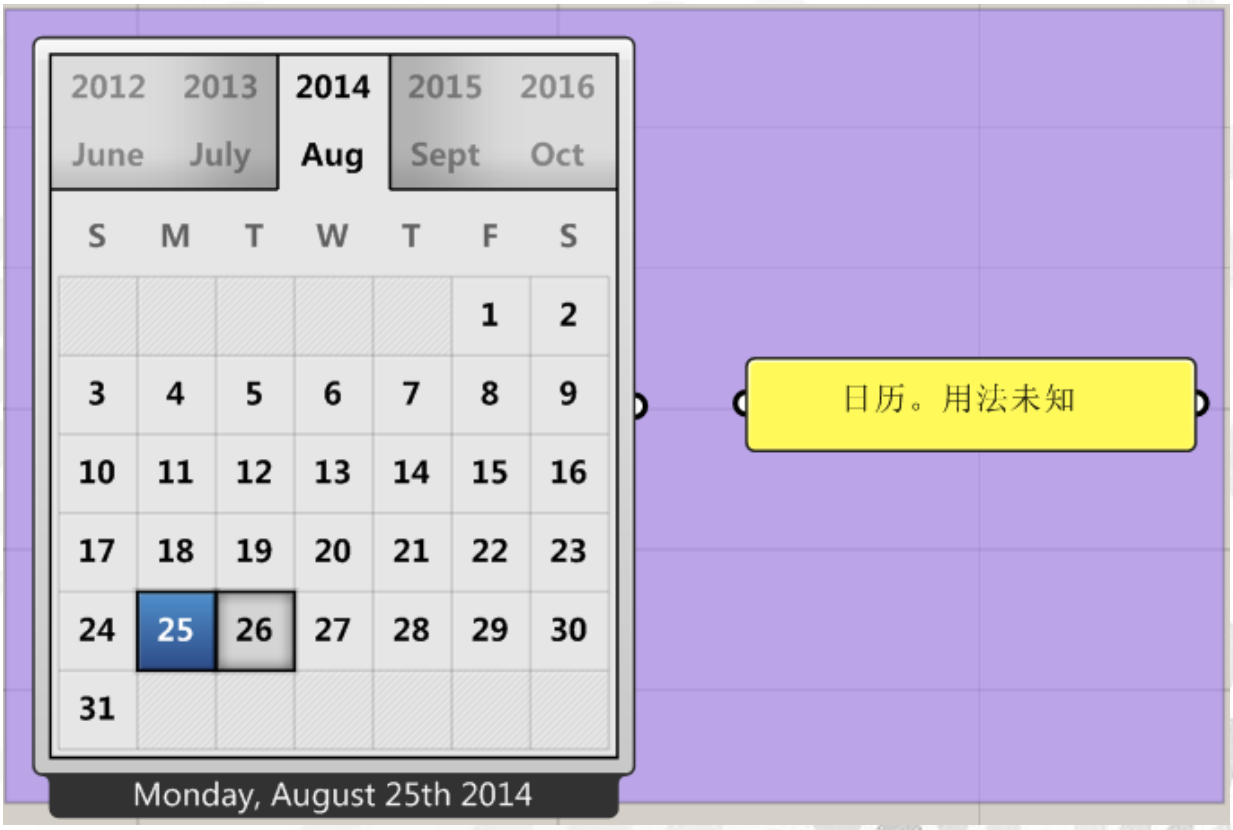
Number Slider:最常用的拉棒，可以输入任意数字
Panel:可以查看电池所包含的数据



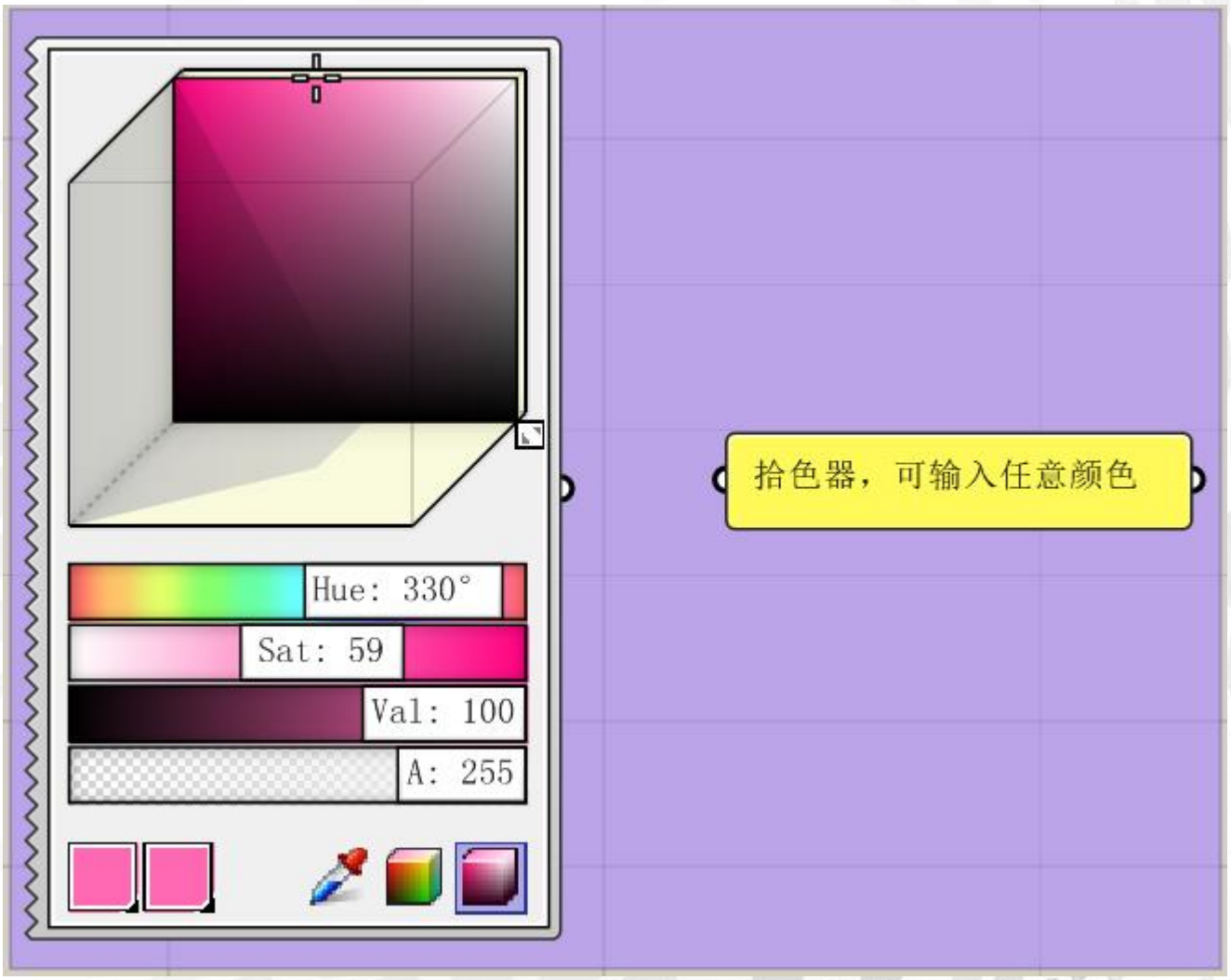
Boolean Toggle:布尔开关(True/False)
Control Knob:数据输入的另一种，类似于音量调节旋钮。
MD Slider:图形化的数字输入端，可以输入树形数据



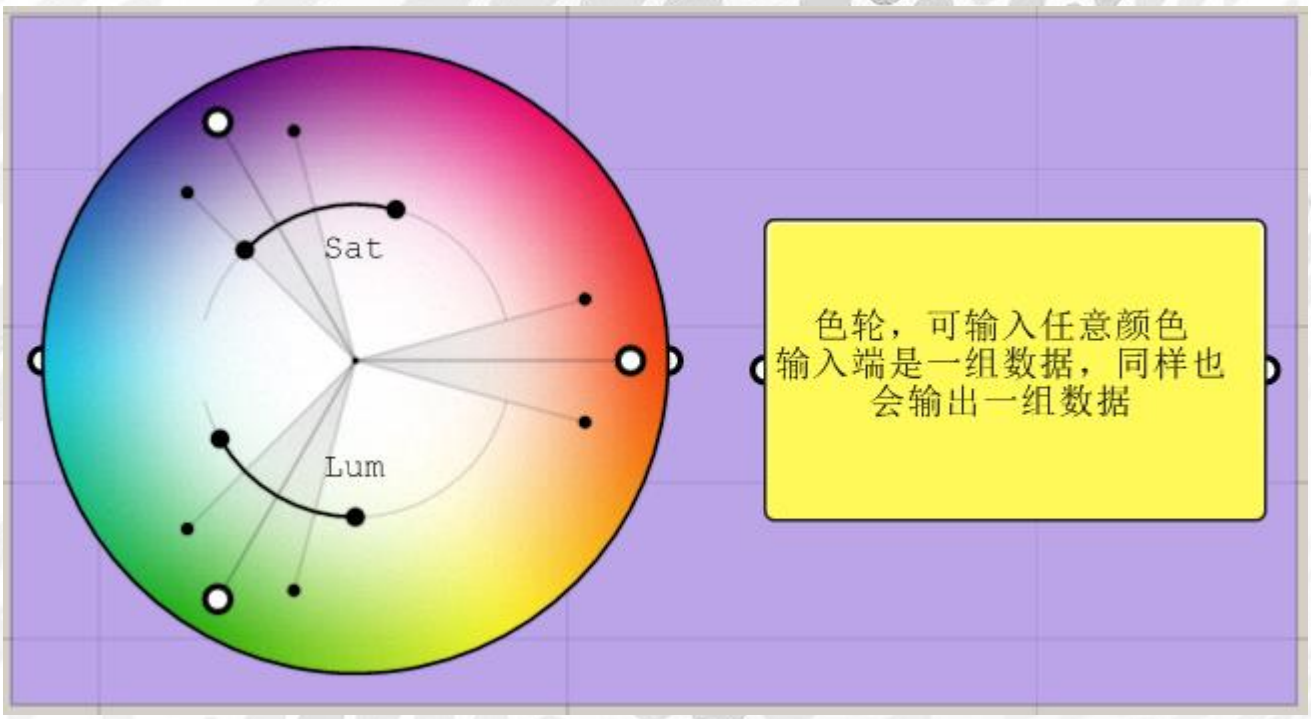
Button:另一种开关，与 Boolean Toggle 类似
Digit Scroller:油表式的数据输入条
Value Lis:用来输入阶数或者输入数字



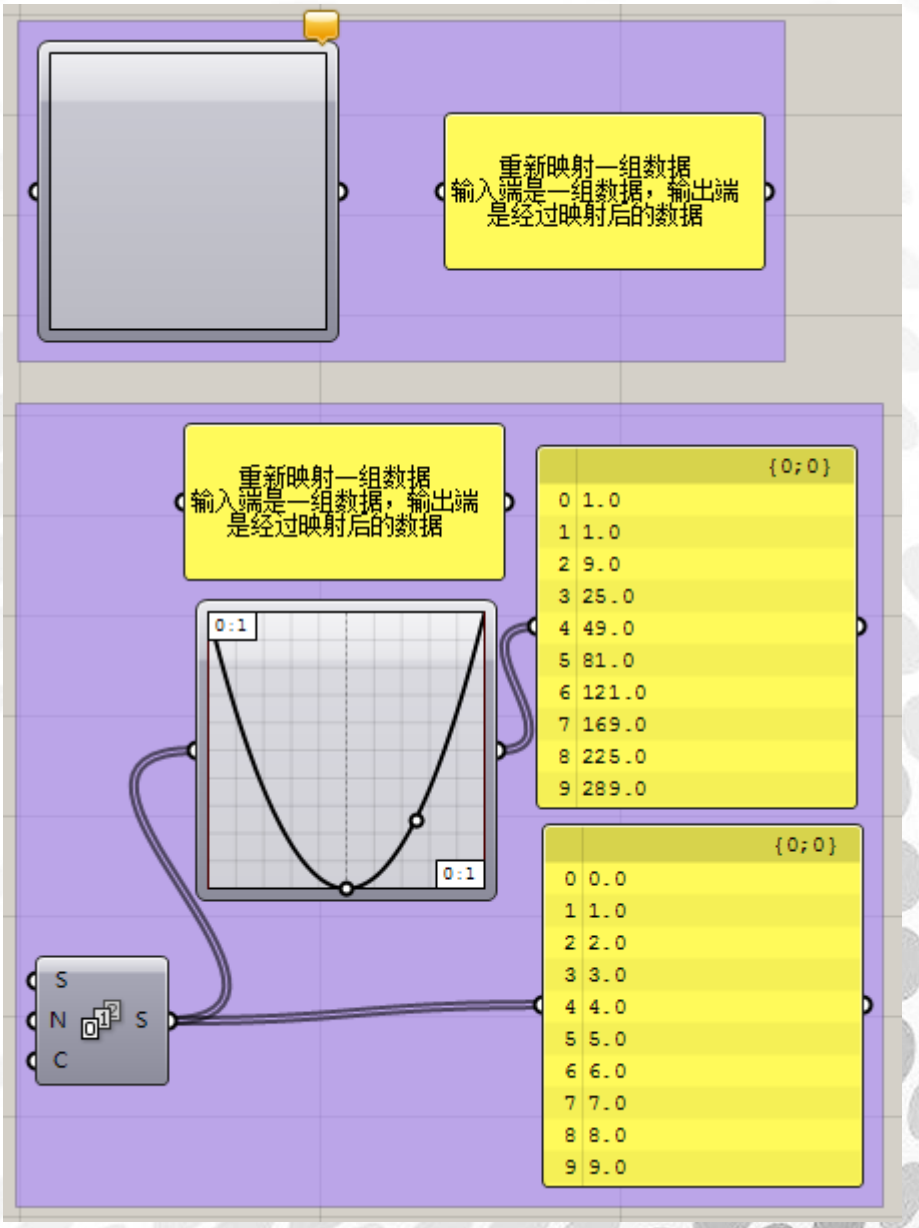
Calendar：日历。说实话没有见过具体案例中使用到...



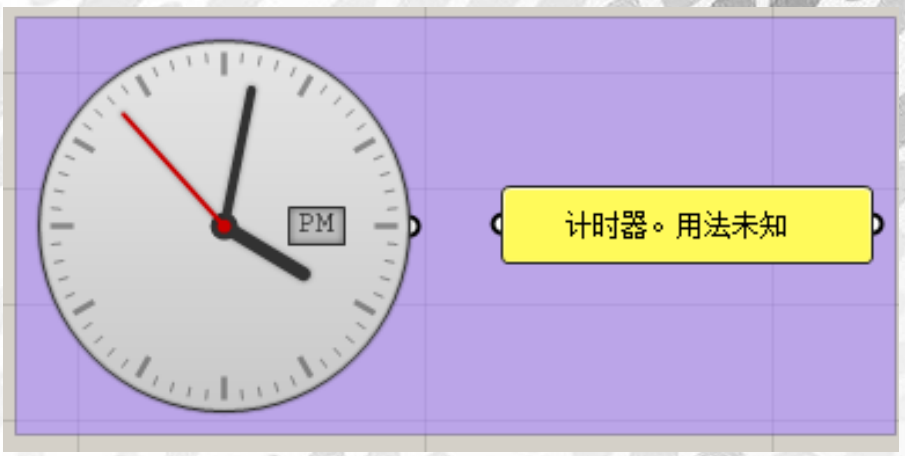
Color Picker:拾色器，可输入任意颜色



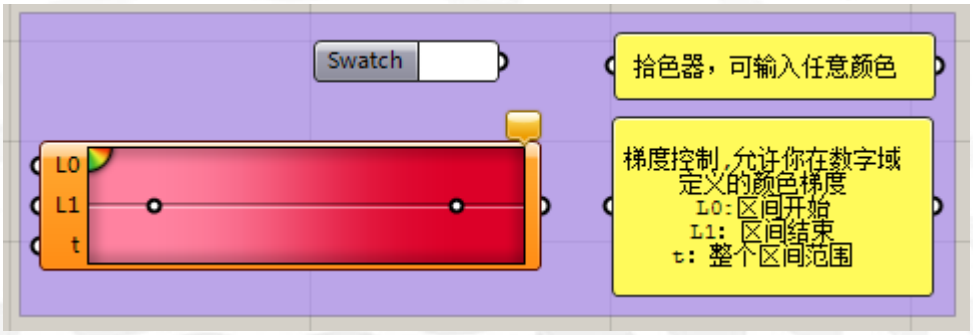
Color Wheel:色轮，可输入任意颜色



Graph Mapper:重新映射一组数据



Clock:计时器。说实话没有见过具体案例中使用到...



Color Swatch:拾色器，可输入任意颜色
Gradient:梯度控制,允许你在数字域定义的颜色梯度
关于此运算器的详细说明，请登录 csh.eeeetop.com 观看置顶区流风老师的视频专题教程

Image Sampler:输入图像数据

以下几个运算器说实话鄙人从没有见过有什么案例中使用，因此不再详细解释。如果阁下有高见，欢迎登录 csh.eeeetop.com 反馈，我会将您的建议收录在修订版的 pdf 中。感谢！

Atom Data:得到一个原子的详细数据

下列运算器均为读取地理信息数据使用，实际上都是读取数据库数据所用。（Chester.L）

Import Coordinates

Import PDB

Read File

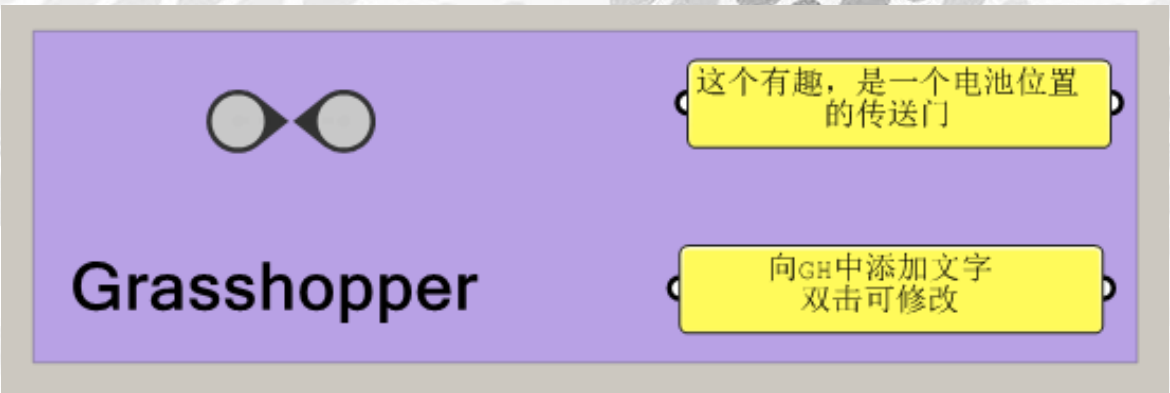
Import 3DM

Import Image

Import SHP:

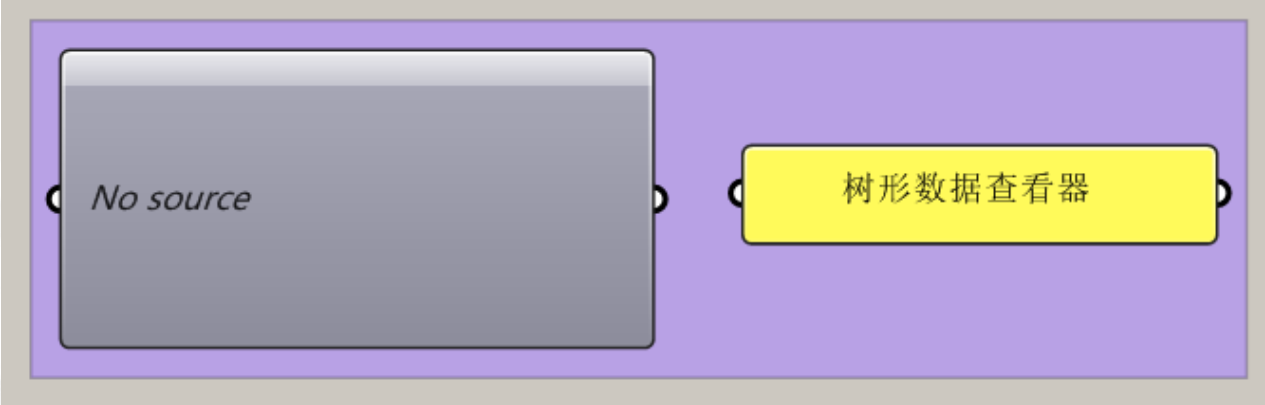
（4）Util 电池组：

Cherry Picker:提取树形数据中的某一节

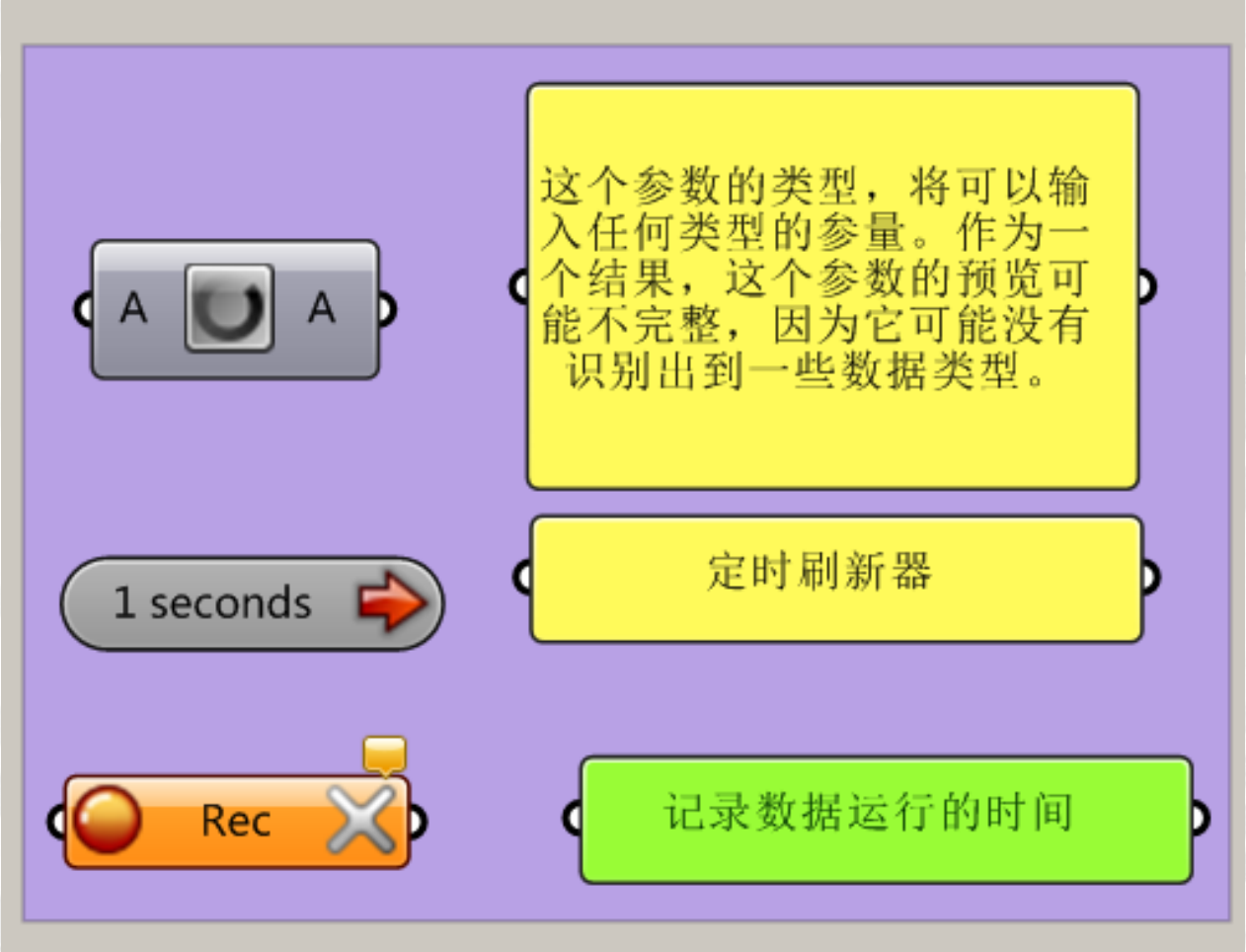


Jump:这个有趣，是一个电池位置的传送门

Scribble:向 GH 中添加文字



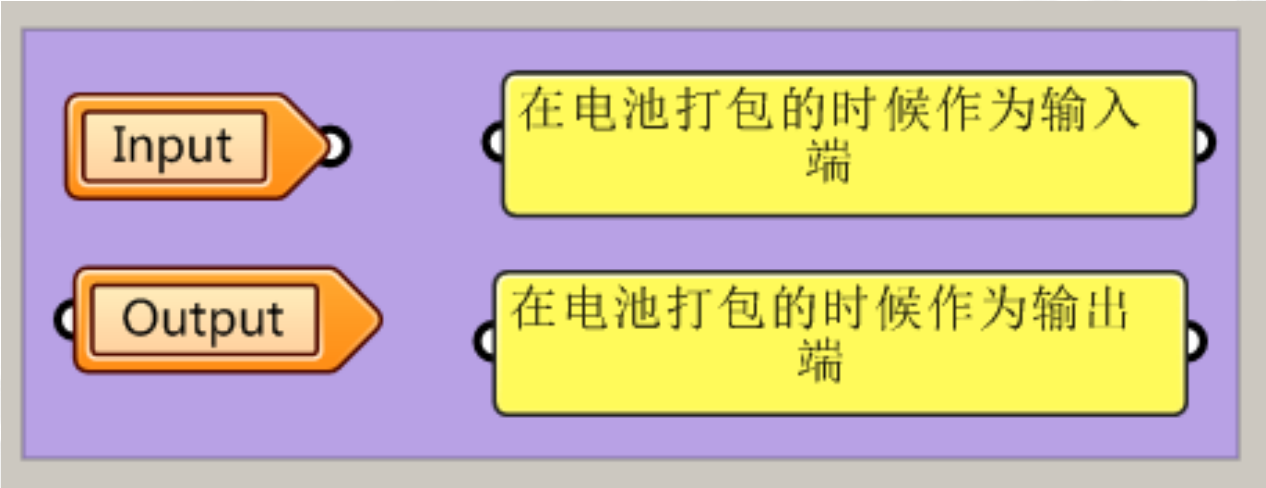
Param Viewer:树形数据查看器



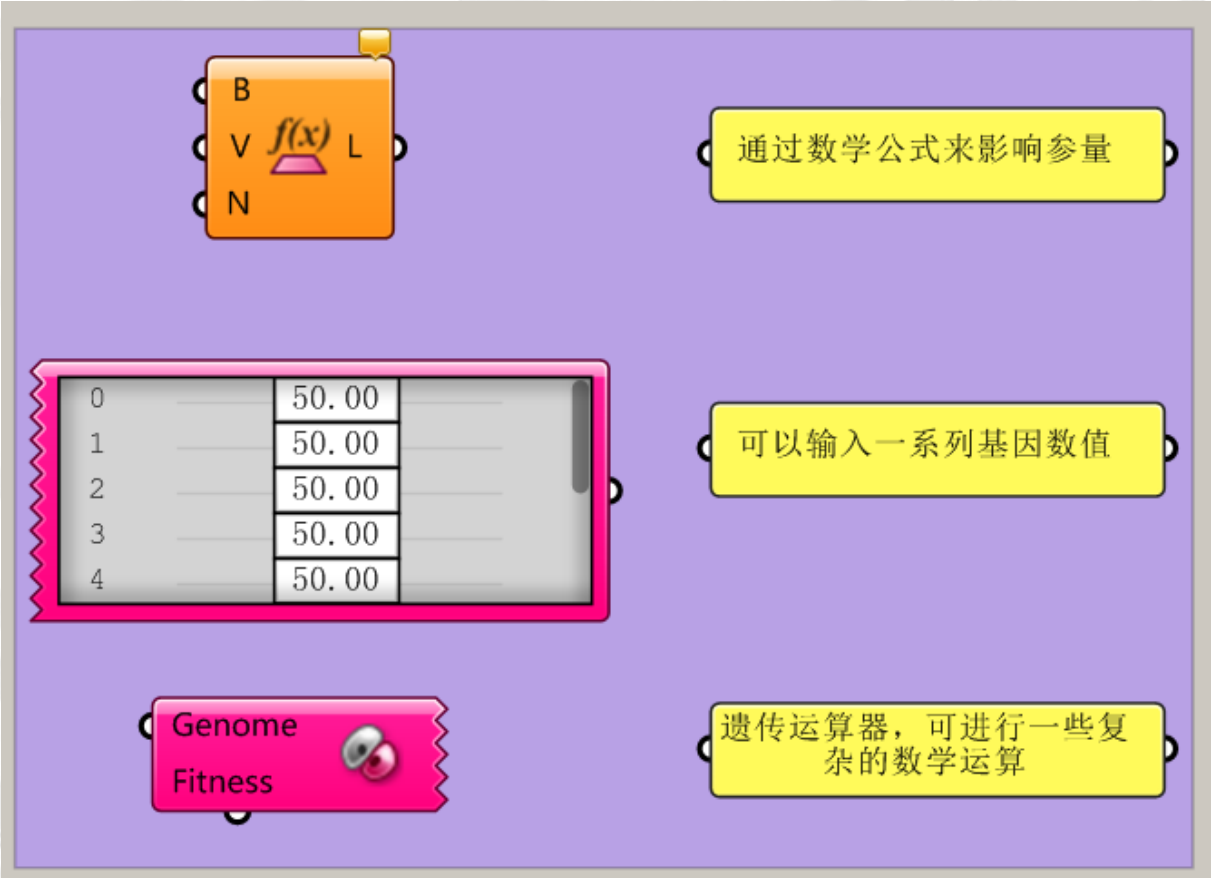
Data Dam:这个参数的类型，将可以输入任何类型的参量。作为一个结果，这个参数的预览可能不完整，因为它可能没有识别出到一些数据类型。

Timer:定时刷新器

Data Recorder:记录数据运行的时间



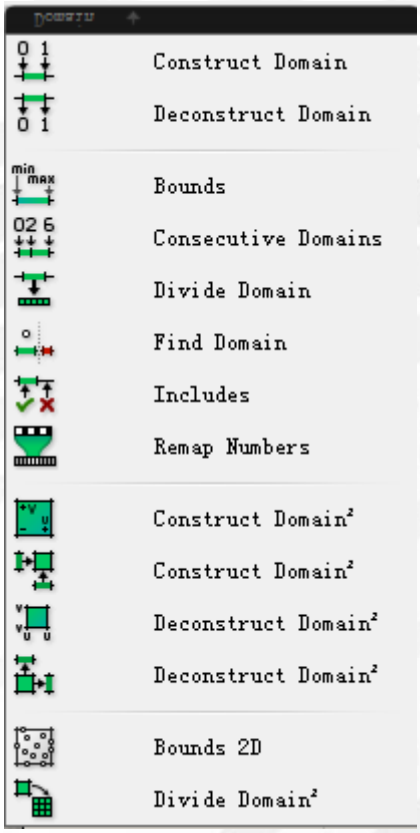
Cluster Input:在电池打包的时候作为输入端
Cluster Output:在电池打包的时候作为输出端



Fitness Landscape:通过数学公式来影响参量
Gene Pool:可以输入一系列基因数值
Galapagos:遗传运算器，可进行一些复杂的数学运算。
这是 Grasshopper 中特立独行的一个运算器，首先不只是因为它的输出端长在了下面，而是因为它的强大，大家鼠标放在运算器上看描述就可以知道了...
这个运算器可以进行复杂的运算从而得出结果。想要详细了解这个运算器，请登录 csh.eeeetop.com 观看流风老师的视频专题课程。

2. Maths 电池组

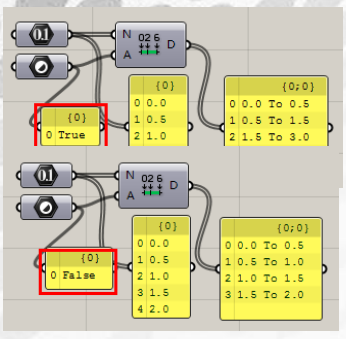
（1）Domain 电池组



Construct Domain:创建从 A 到 B 的一个范围 I
Deconstruct Domain：讲一个范围 I 分解为起始值 S 和结束值 E

Bounds：在一组数中，用最大值和最小值创建一个范围

Consecutive Domains：连续范围
从字面上比较难理解，解释如下。
当 A 取值为 True 则从一系列的数字中，分别取前 n 项的和与 n+1 项的和构成范围。
当 A 取值为 False 则为 n 项和 n+1 项两个数字构成范围。



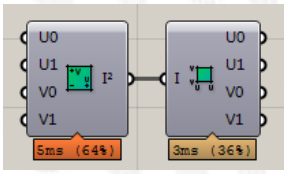
Divide Domain：将 Domain 等分为 C 个小的范围区间

Find Domain：寻找区间（D）内输入的数值（N）的编号（I）。
输入端 S：是否以精确方式查找
输出端 N：得到的旁边相邻数据的编号

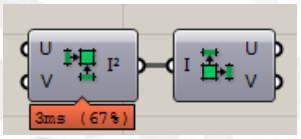
Includes：寻找输入数值（I）是否在区间（D）以内

Remap Numbers：映射两个范围比值得到新范围
输入端 V：映射数值
输入端 S：源范围
输入端 T：目标范围
输出端 R：新的范围
这个比较难翻译其作用，输入端和输出端的关系是：R=（T/S）*V

以下两组计算器重名，且互相可逆：



Construct Domain² ：
通过定义（输入 UV 坐标面的 U 和 V 的上下限）来确定一个面域
Deconstruct Domain² ：分解面域得到 UV 坐标的上下限

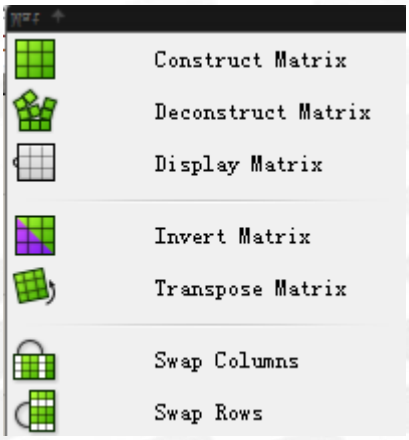


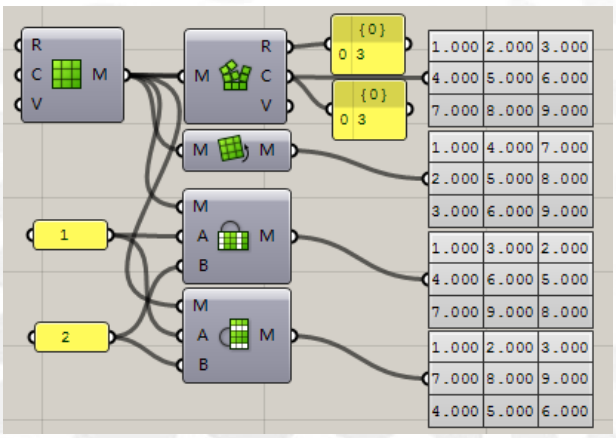
Construct Domain² ：
通过定义 U 和 V 的两个区间来确定一个面域
Deconstruct Domain² ：分解面域得到 U 和 V 的两个区间

Bounds 2D：通过一组点得到一个面域，该面域为包含所有点的最小面域

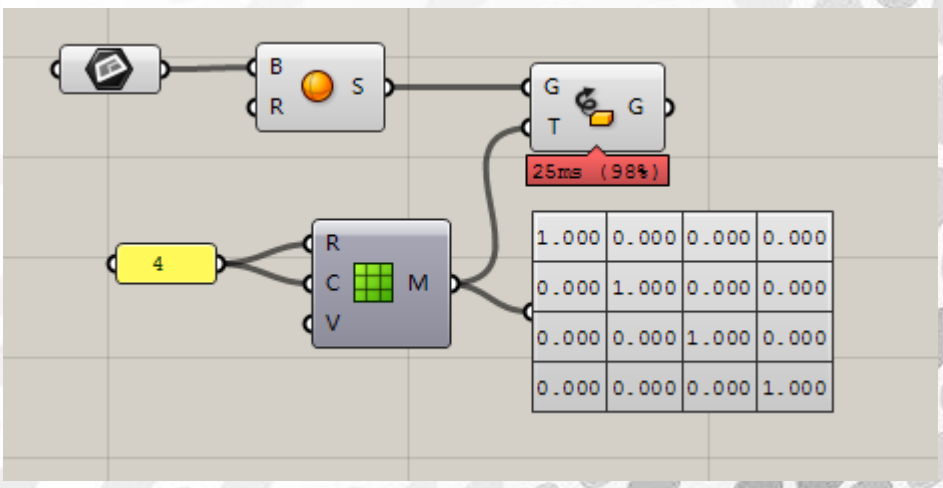
Divide Domain² ：将一个面按照 uv 坐标细分，细分出若干个面域。这是一个非常常用的运算器，常用来细分曲面使用。
这个运算器常常和 Isotrim 连用。如果想要观看实例中的应用，请登录 csh.eeeetop.com 查看我发布的【By DanielJin】由浅入深学 Grasshopper 中级 01 教程

（2）Matrix 电池组



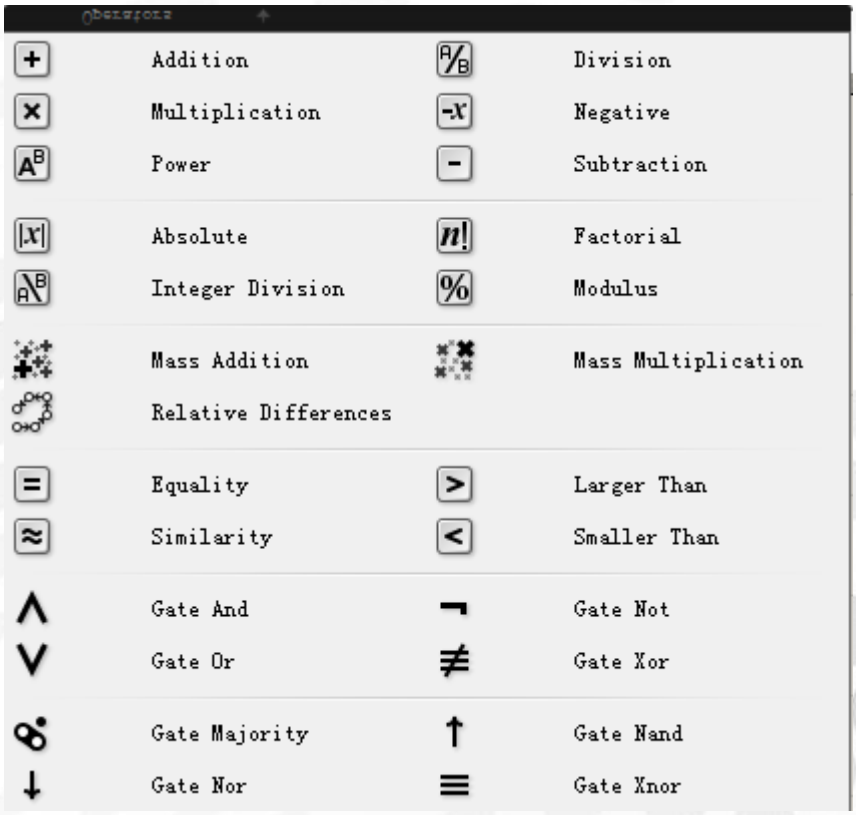


Construct Matrix：创建矩阵
Deconstruct Matrix：分解矩阵
Display Matrix：展示矩阵
Transpose Matrix：将矩阵的列和行调换
Swap Columns：调换矩阵的 A 列和 B 列
Swap Rows：调换矩阵的 A 行和 B 行
输入端或输出端的 R 代表 Rows，行。C 代表 Columns，列。V 代表矩阵里的各项值。
Invert Matrix：改变矩阵。



GH 中的变换本质上都是以矩阵来实现的，上图为平移矩阵示意，其他包括旋转，对称，切变、缩放等都可用矩阵实现。（Chester.L）

（3）Operators 运算 电池组



这一组电池非常好掌握，学过数学的都明白。简单的不做详细翻译了。

前半部分属于算法：

Addition：加法

Division：除法

Multiplication：乘法

Negative：负值

Power：幂

Subtraction：减法

Absolute：绝对值

Factorial：阶乘

Integer Division：除法，得到整数商

Modulus：取余数，用 A 除以 B 输出得到的余数

Mass Addition：总量相加。其中输出端 Pr 是每一步的累加值。

Mass Multiplication：累乘

Relative Differences：每一项和上一项的差值。

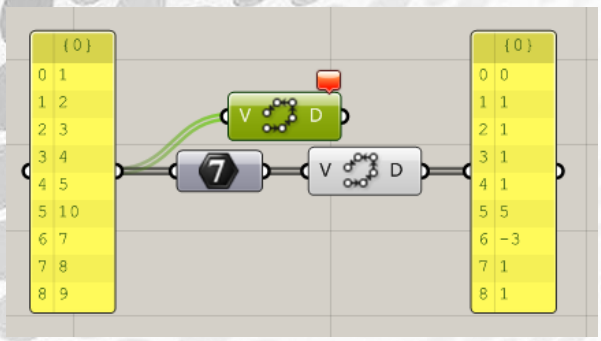
注意输入类型只能是整数，数字，点和向量。

Equality：相等

Larger Than：大于

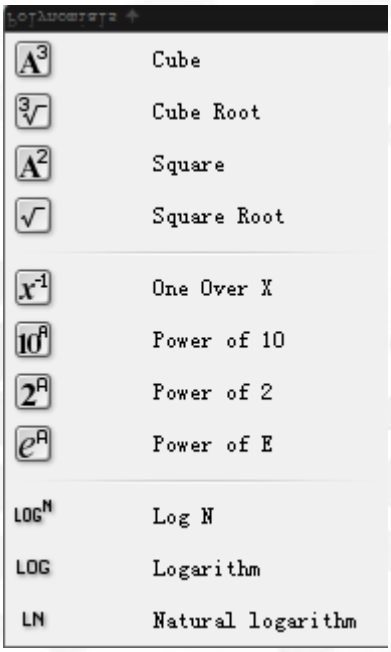
Similarity：约等于

输入端 T%：误差允许值%



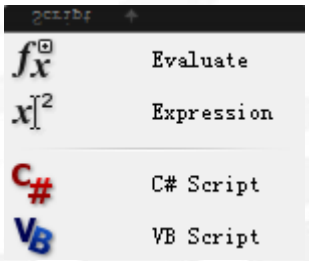
输出端 dt：两者的差
Smaller Than：小于
后半部分属于逻辑
Gate And：当输入端都为 True 时，输出 True，即 A 且 B
Gate Or：A 或 B
Gate Not：将输出结果变为相反的结果
Gate Xnor：输入的布尔值 A=B 时为真
Gate Xor：A 不等于 B 为真
Gate Nand：若 A=B=True，输出假，其他情况输出真。
Gate Nor：若 A=B=False，输出真，其他情况输出假。
Gate Majority：按照输入端的真假结果，输出占得比例大的结果。

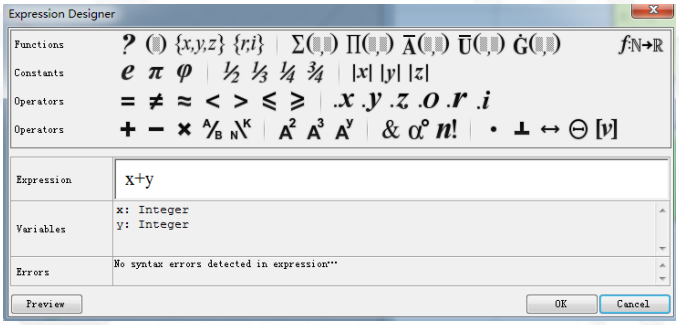
（4）Polynomials 电池组



Cube：立方
Cube Root：立方根
Square：平方
Square Root：平方根
One Over X：x 的负一次方
Power of 10:10 的 x 次方
Power of 2：2 的 x 次方
Power of E：自然系数 e 的 x 次方
Log N：对数 log
Logarithm：以 10 为底的对数 lg
Natural Logarithm：以自然系数 e 为底的对数 ln

（5）Script 电池组

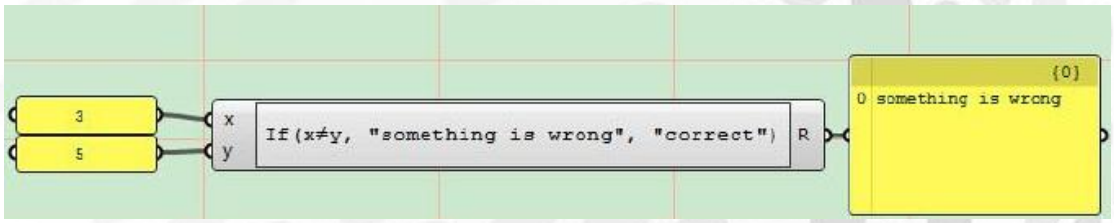




Evaluate：函数运算器，使用表达式进行判定。
放大运算器以后可以添加删除输入端变量。
双击 fx 函数图标会出现上面的窗口，用于创建函数。
同时会显示输入端的输入类型，比如上图的整数。

Expression：表达式

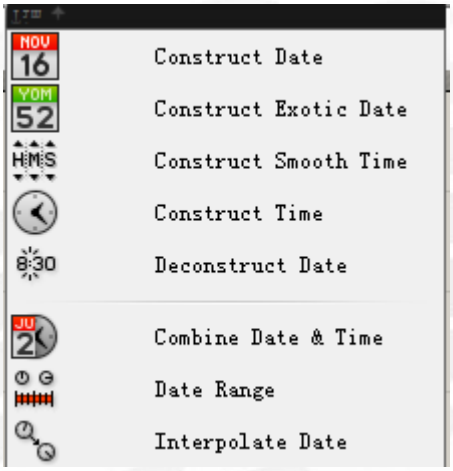
C# Script,VB Script:C#和 VB 脚本运算器
需要编辑时请右键图表中心 logo



Expression 支持预设了很多合法的表达式，用来作为条件选择或者逻辑判断有时候比 Script 更为方便。（Chester.L）

（6）Time 电池组

Time 电池组用的比较少，简单翻译大家略有了解即可。

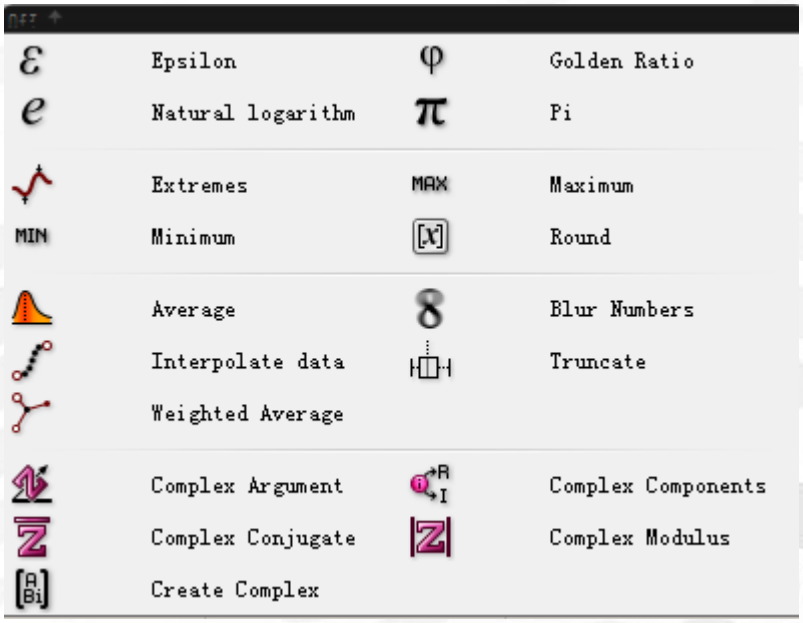


Construct Date：创建日期
Construct Exotic Date：创建一个“异国”的日期
就是创建一个时间，运用指定的日历
Construct Smooth Time：创建一个光滑连续的计时器
Construct Time：创建时间
Deconstruct Date：将现在的时间分解为年月日时分秒
Combine Date & Time：结合日期和时间
Date Range：日期范围
Interpolate Date：插入时间

（7）Trig 三角函数电池组

Cosine：余弦
Sinc：辛格函数，不好理解，自己百度吧！
Sine：正弦函数
Tangent：正切函数
ArcCosine：反余弦函数
ArcSine：反正弦函数
ArcTangent：反正切函数
CoSecant：输入值 x 的 sine 函数值的倒数值
CoTangent：输入值 x 的 tangent 函数值的倒数值
Secant：输入值 x 的 cosine 函数值的倒数值
Degrees：弧度转化为角度值
Radians：角度转化为弧度值

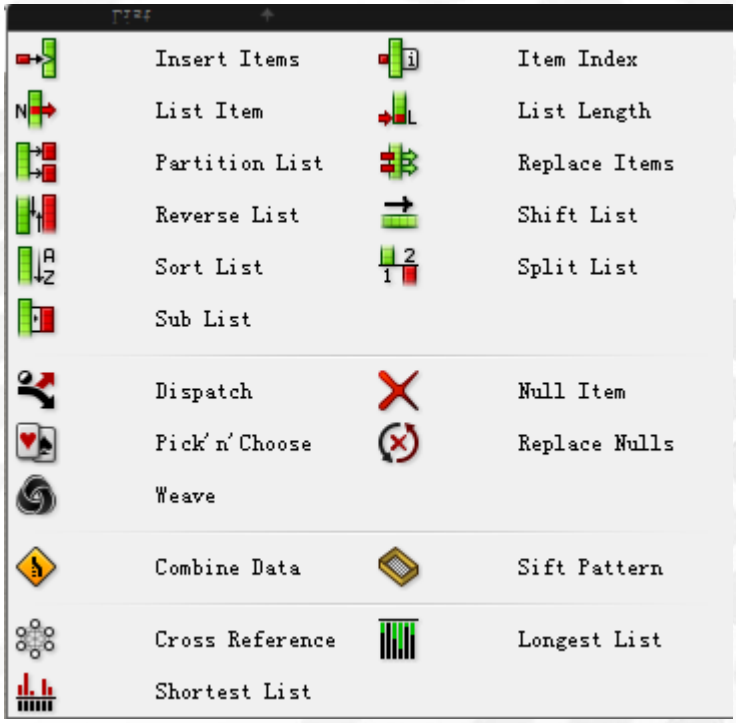
（8）Util 电池组



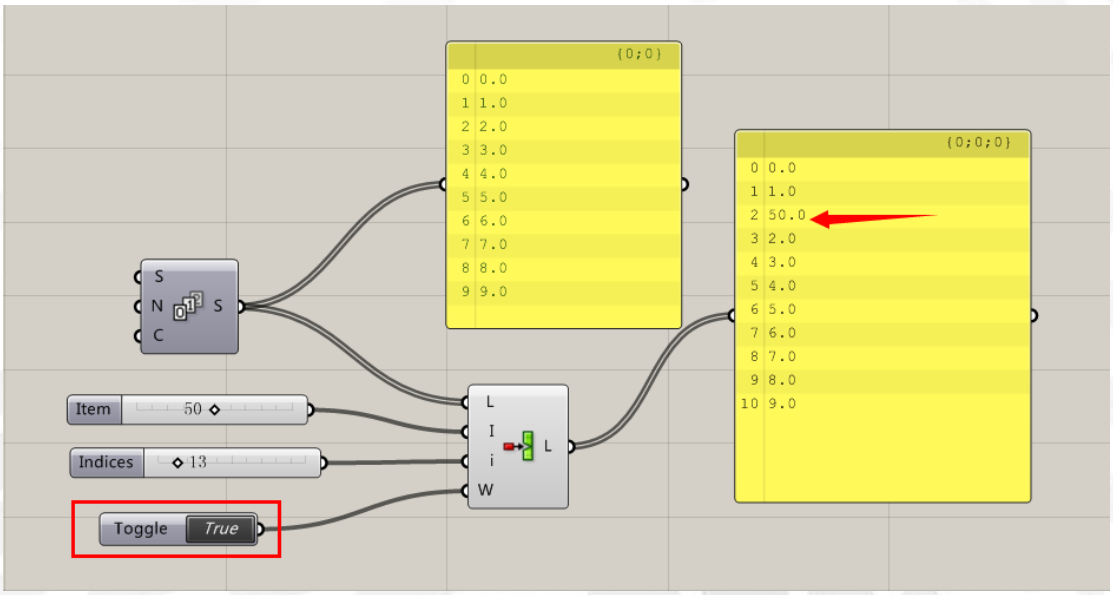
输入端 N：倍数
Epsilon：一个无限趋近于 0 的数，却不等于 0
Natural Logarithm：自然对数
Golden Ratio：黄金比例
Pi：圆周率
Extremes：极值
Maximum：最大值
Minimum：最小值
Round：四舍五入
输出端 N：输出结果
输出端 F：向下取整
输出端 C：向上取整
Average：求平均数
Interpolate Data：插入数据
通过输入端 t（0-1）控制列表数据 D 的数值之间的插入值。t 相当于百分比因子。
Truncate：截断数据
在列表中根据输入 t（0-1）依次剔出首位两端的数据
Weighted Average：判定输入端 I 和 W 这两组数据的平均权重值
Complex Argument：复数的幅角
Complex Conjugate：复数的共轭
Complex Components：分解复数的实部和虚部
Complex Modulus：复数的取模
Create Complex：通过输入端 R 和 i，创建一个复数 C（R+i）

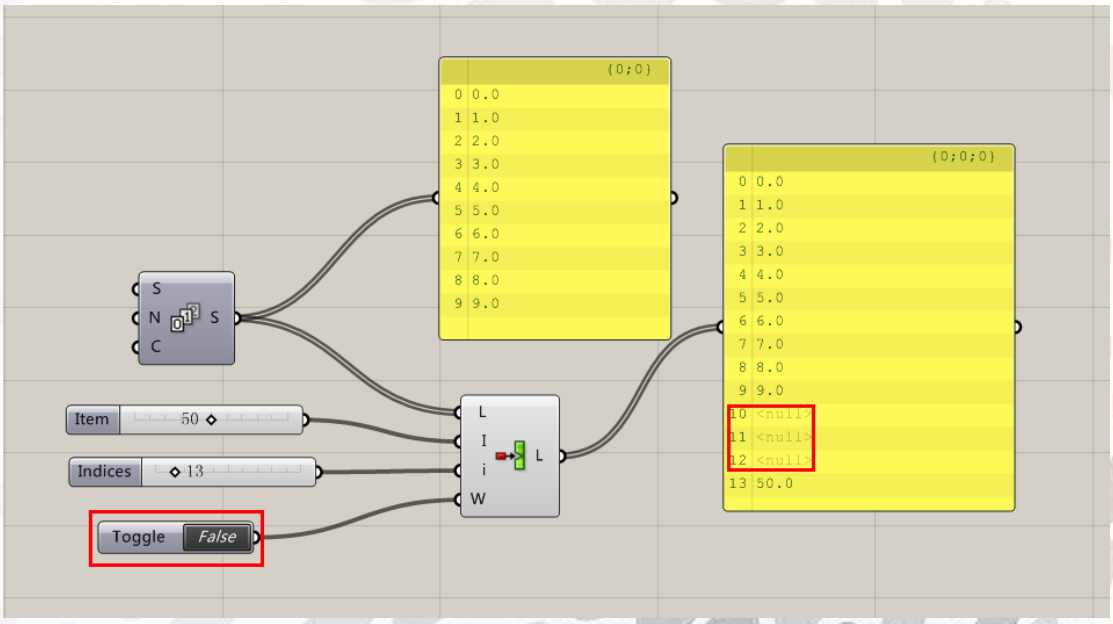
3. Set 电池组

（1）List 电池序列

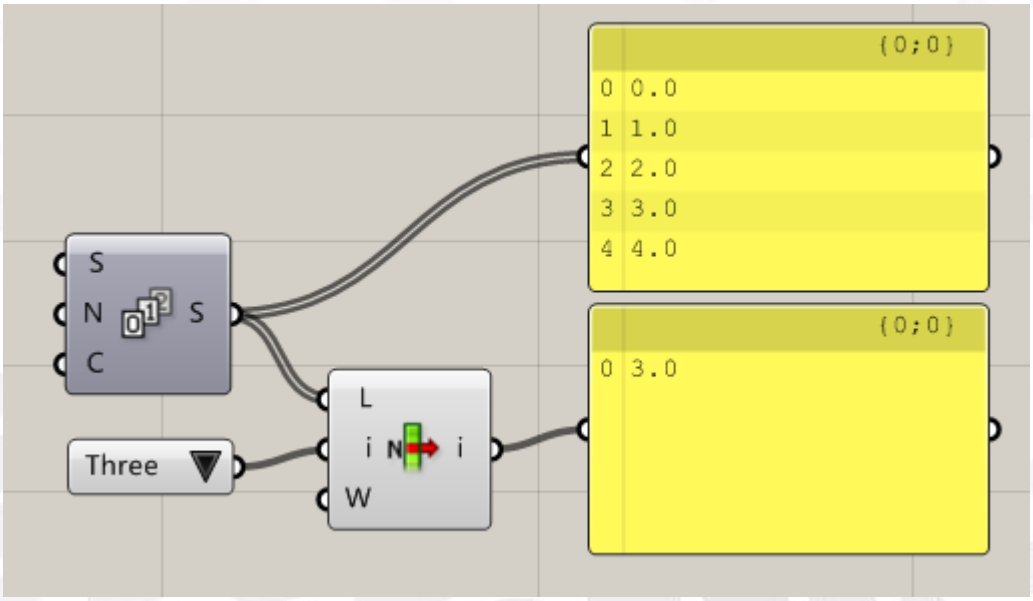


Insert Items:插入数据到列表中
输入端 L:要插入数据的目标列表
输入端 I:插入什么数据
输入端 i:插入数据的编号
输入端 W:很难用语言简练的讲清楚,因此附上图片示例供大家对比.
之后的布尔值输入端学习方法也类似,请大家举一反三,自行实践.

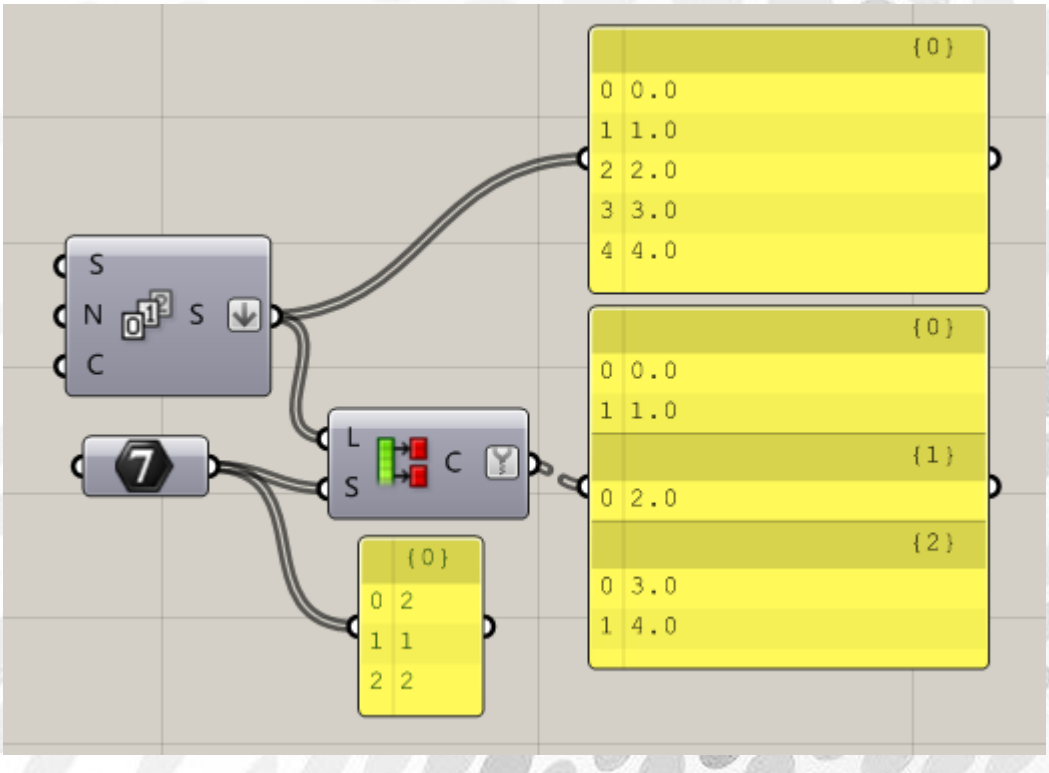




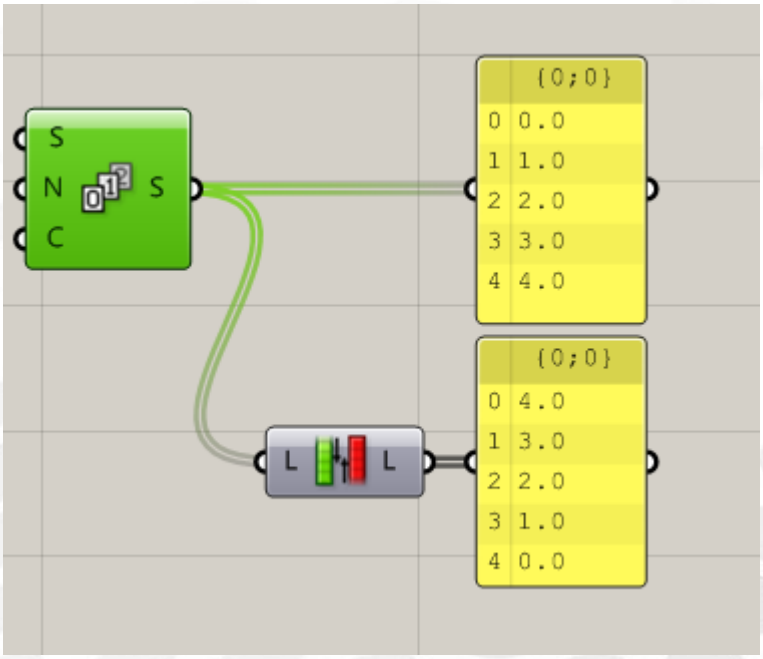
List Items:根据编号选择列表里的数据
输入端 L:需要选择数据的原始列表
输入端 i:选择数据的编号



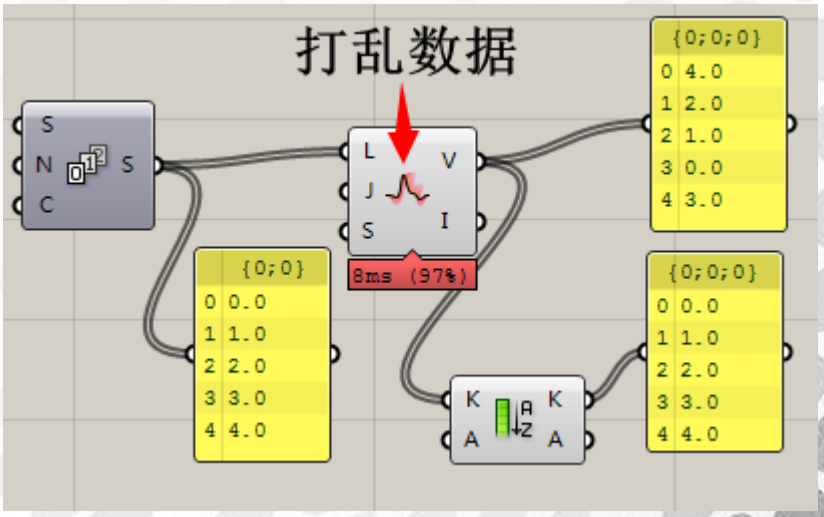
Partition List:按数量划分列表
输入端 L:需要作为划分数据的原始列表
输入端 S:指定多少个数据划分在一起



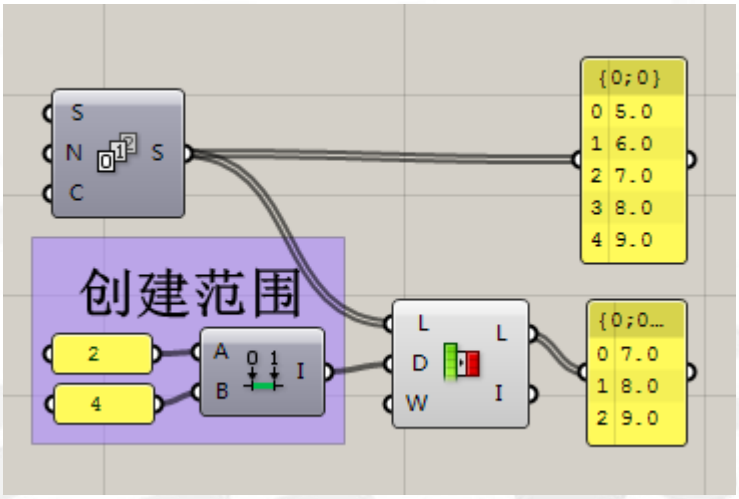
Reverse List:反转数据列表的顺序



Sort List:排序列表,按照编号的大小顺序排列编号和与编号关联的对象
输入端 K:需要排列的列表数据(数值,字符等)
输入端 A:需要排序的物体对象(需要和输入 K 关联)



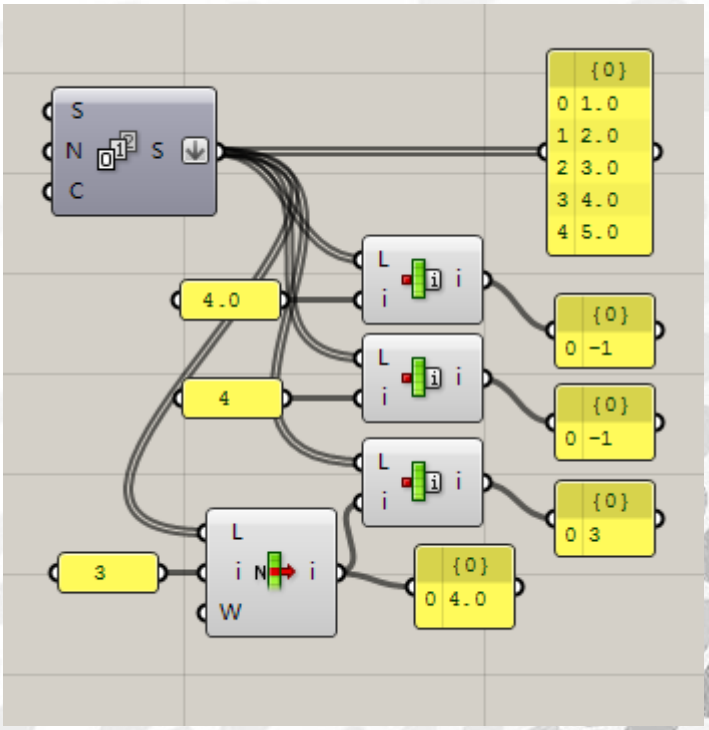
Sub List:输入一个区间,将元列表在指定区间内的项选择出来.
输入端 L:原始数据列表
输入端 I:选取数据的区间,作为分割依据



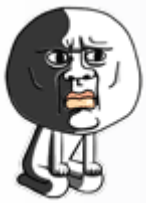
Item Index:检索数据列表中的某一项，输出他的标号。类似在操场点名，你喊最帅的那个出来，我就会站出来，迅速找到这一项的位置。

输入端 L:检索的目标数据列表

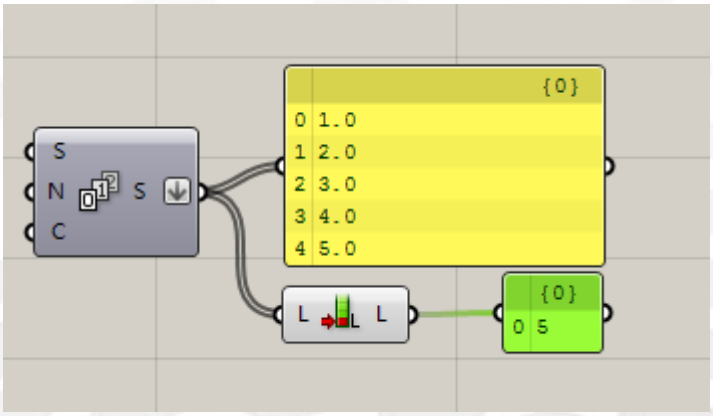
输入端 i:检索的数据编号



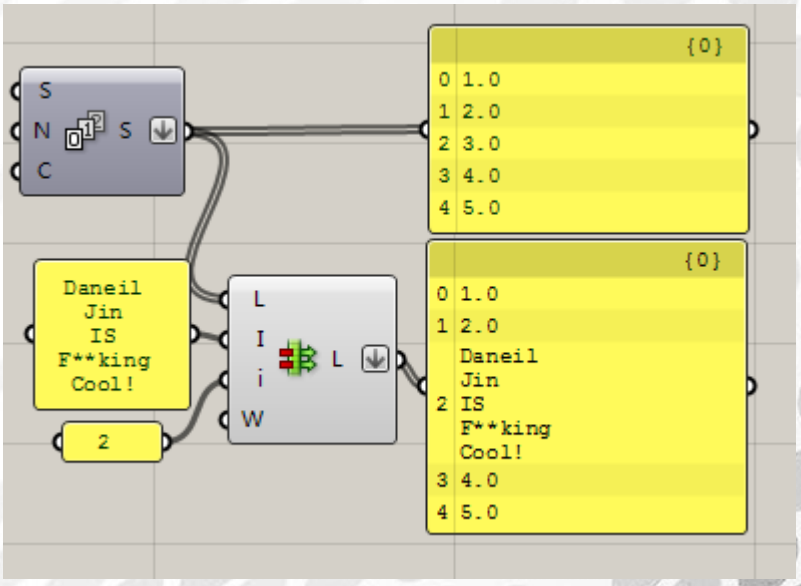
这个运算器比较奇葩。我本来需要从原数据中找到 4 这一项在第几个位置，答案应该输出在{0}分组里的底 3 项。但是结果...有很多情况显示了-1 项...因此经过讨论，总结如下：该运算器完美的证明了，4≠4.0;4.0≠4.0



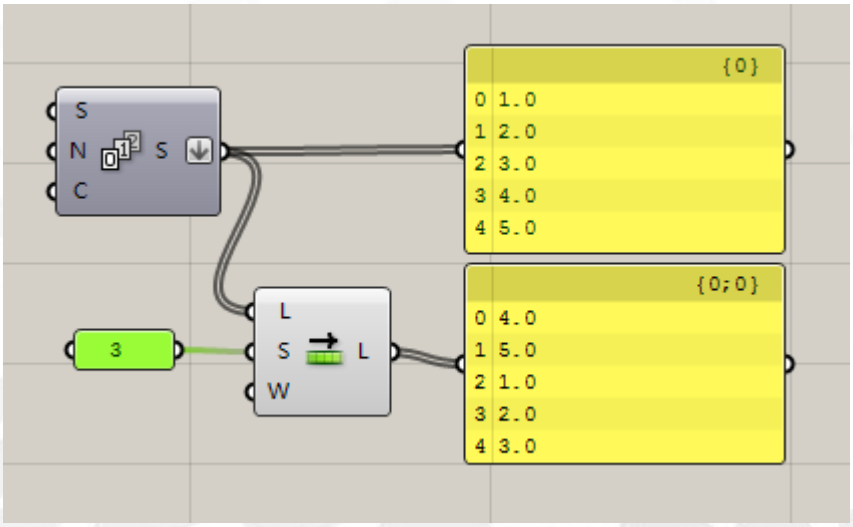
List Length:计算数据列表长度



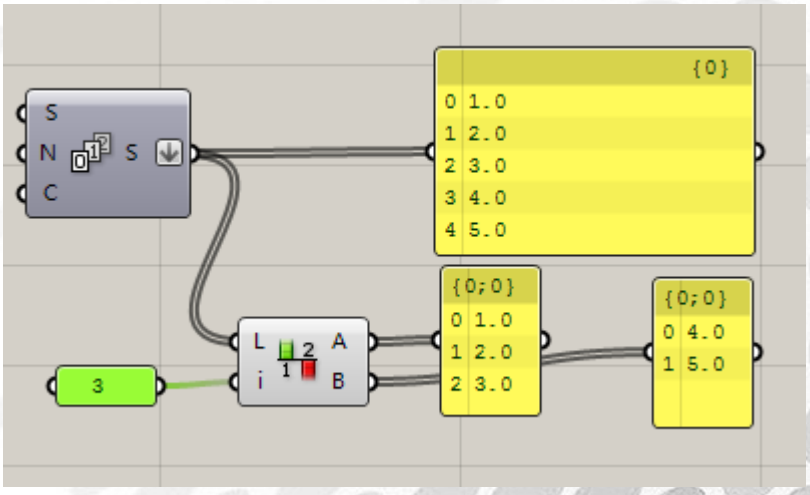
Replace Items:替换列表指定项的数据内容
输入端 L:需要作为替换数据的原始列表
输入端 I:需要替换的数据
输入端 i:替换数据第几项的编号



Shift List:根据输入值偏移数据,向上或向下滚动列表
输入端 L:需要滚动数据的原始列表
输入端 S:滚动数量(正值为向上滚动,负值为向下滚动)
输入端 W:True 时保留数据,False 则删除数据

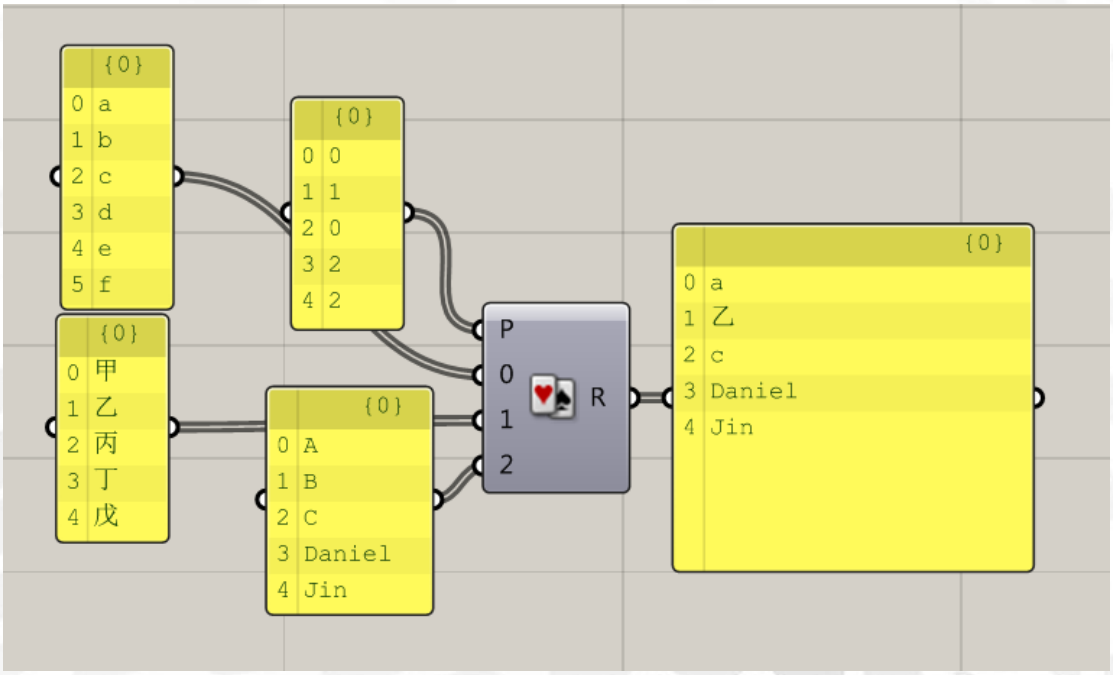


Split List:根据输入编号,将数据列表划分为两个部分
输入端 L:需要划分的原始数据列表
输入端 i 在哪个编号上进行划分



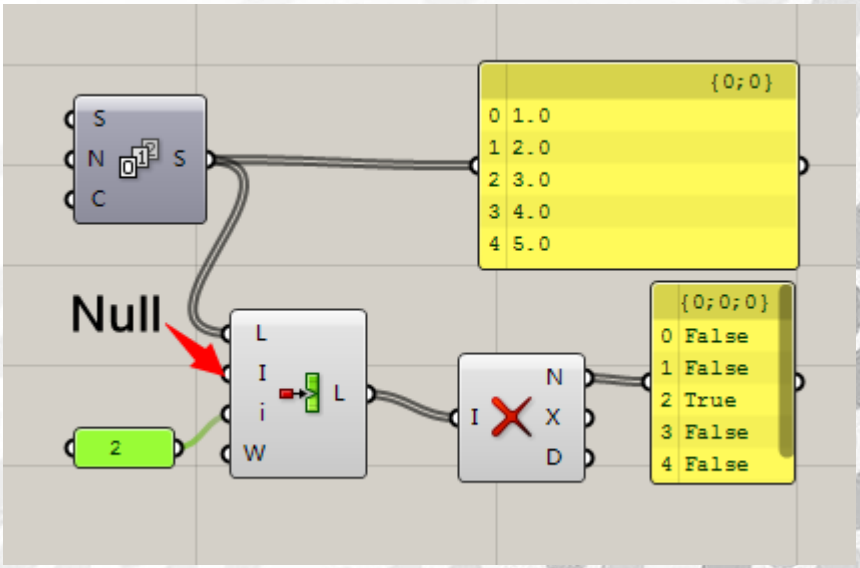
Dispatch:数据分流
（关于此运算器的运用，欢迎登陆 csh.eeeetop.com 观看流风老师的视频公开课演示）
输入 L:需要根据布尔值分流出的原始数据列表
输入 P:布尔值(真为 A 输出，假为 B 输出)

Pick’n’Choose:按条件输出列表数据
输入端 P:指定入口数据的条件
输入端 0,1,2(放大到足够大时可以看到出现加减号,可以按照自己需求添加输入端):
指定具体哪些数据进入次入口,配合 P 条件,输出符合相对入口的数值
这个运算器非常牛，但是也比较难用语言表述。请大家结合下图图示理解。输入的规律是{0,1,0,2,2}，那么输出的 panel 的 0,1,2,3,4 项分别将会从输入端 0,1,2 中选择对应项。

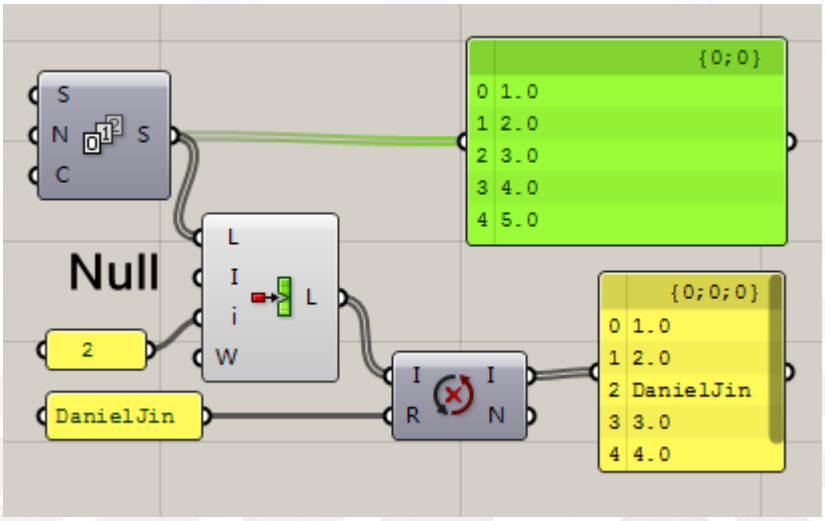


Weave:编织
为列表设定编号后,按顺序提取编号下的数据

Null Item:空变量评价.
以布尔方式来评价列表中对象是否为空

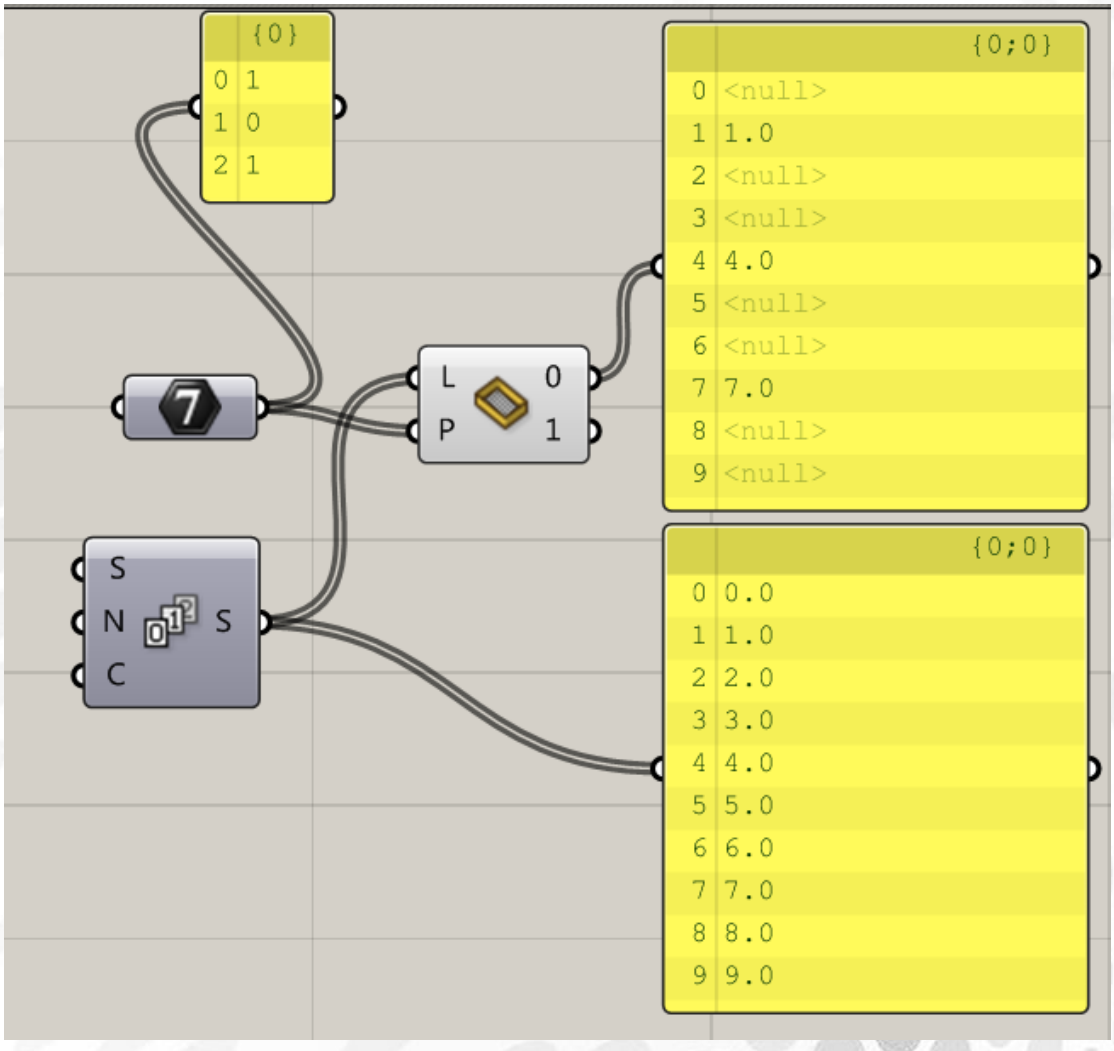


Replace Null:替换 Null 空数据
输入端 I:具有 Null 的数据列表
输入端 R:作为替换 Null 物体的数据列表
输出端 I:替换后的列表结果
输出端 N:有多少 Null 被替换了

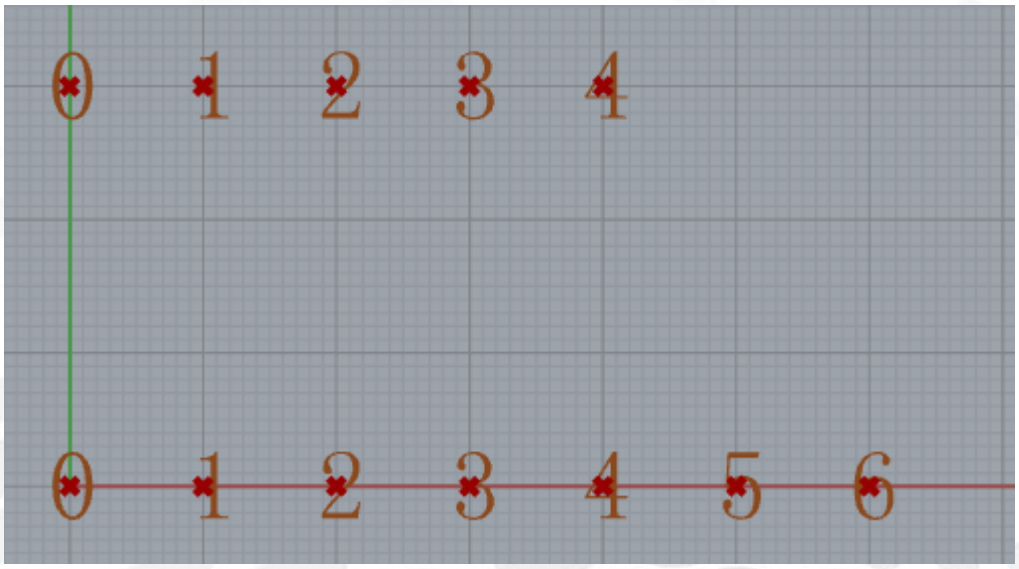


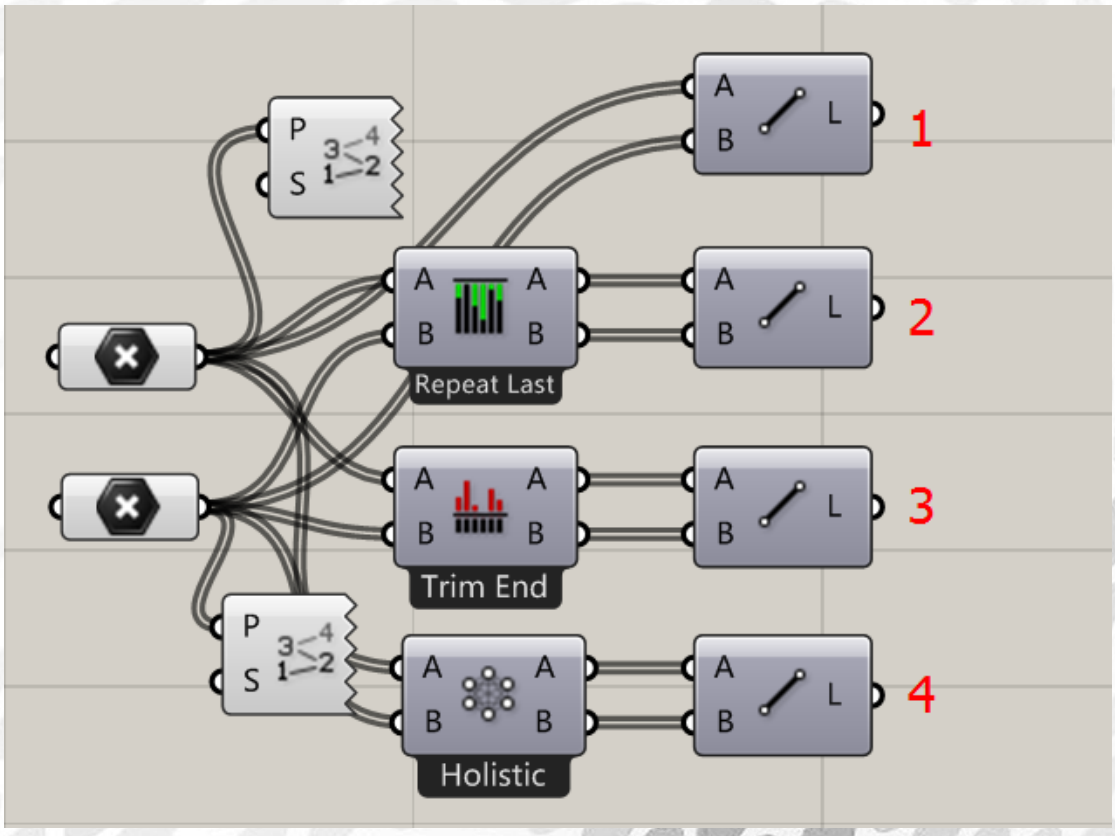
Combine Data:组合数据
将 0 输入端数据做为输出端 R 的数据,依照所有输入端数据列表最多的进行参照.如果数量不匹配,将会把最后一个数据重复排列下去

Shift Pattern:根据条件分流输出 Null 数据
输入端 L:需要分流 Null 的数据列表
输入端 P:布尔分流条件(循环的)
输出:根据条件,有效数据输出,其他数据 Null

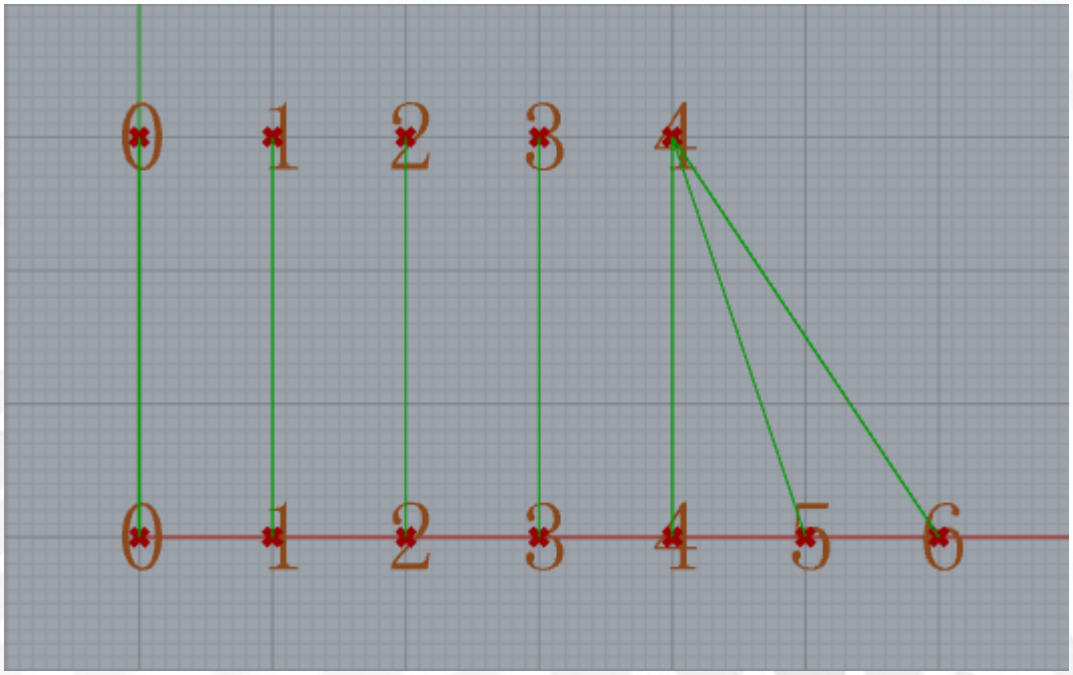


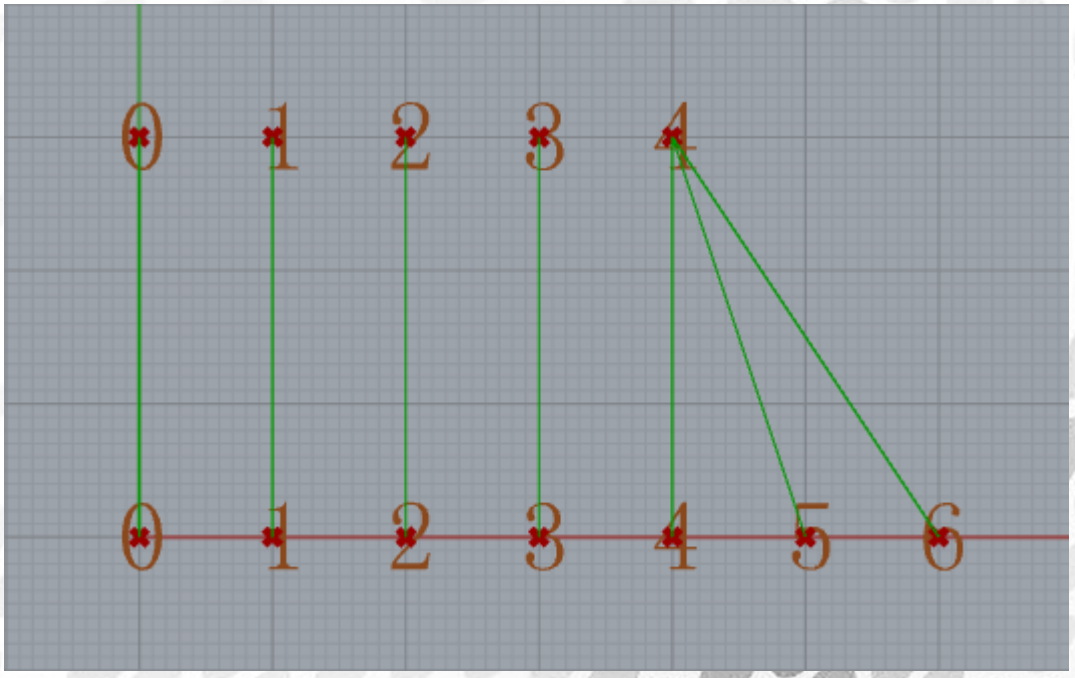
Cross reference:交错排列两组数据
Shortest list, Longest List:短排法，长排法。
新版本的运算器中这些成为了单独的运算器。
为了示范，我们给出示例图：分别在上排点了 5 个点，下排点了 7 个点。

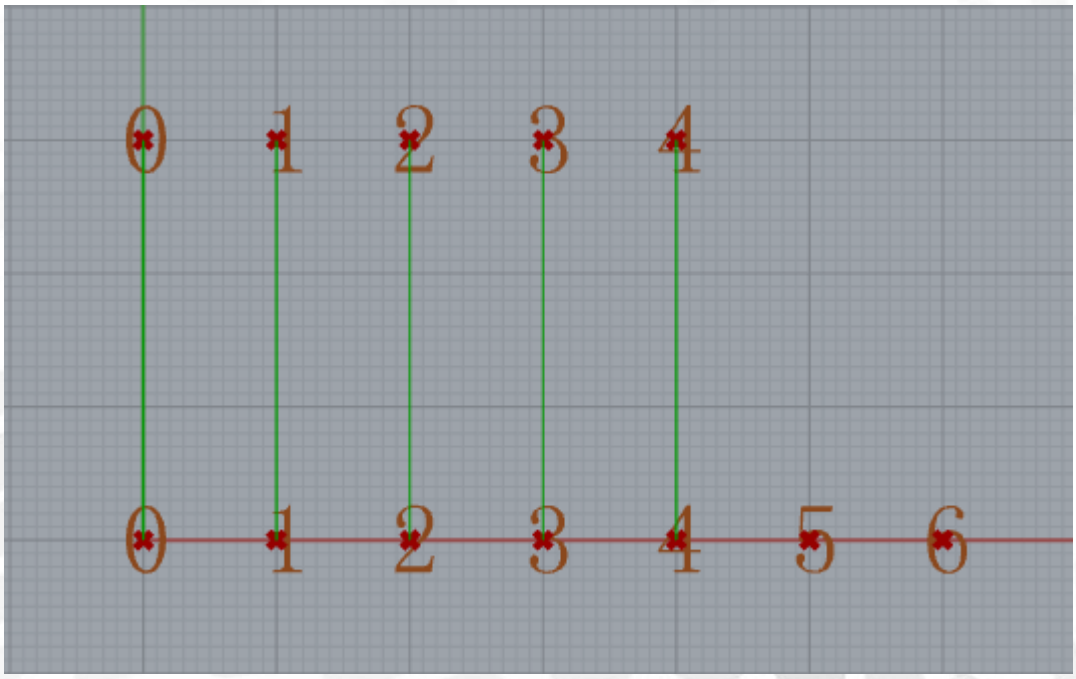


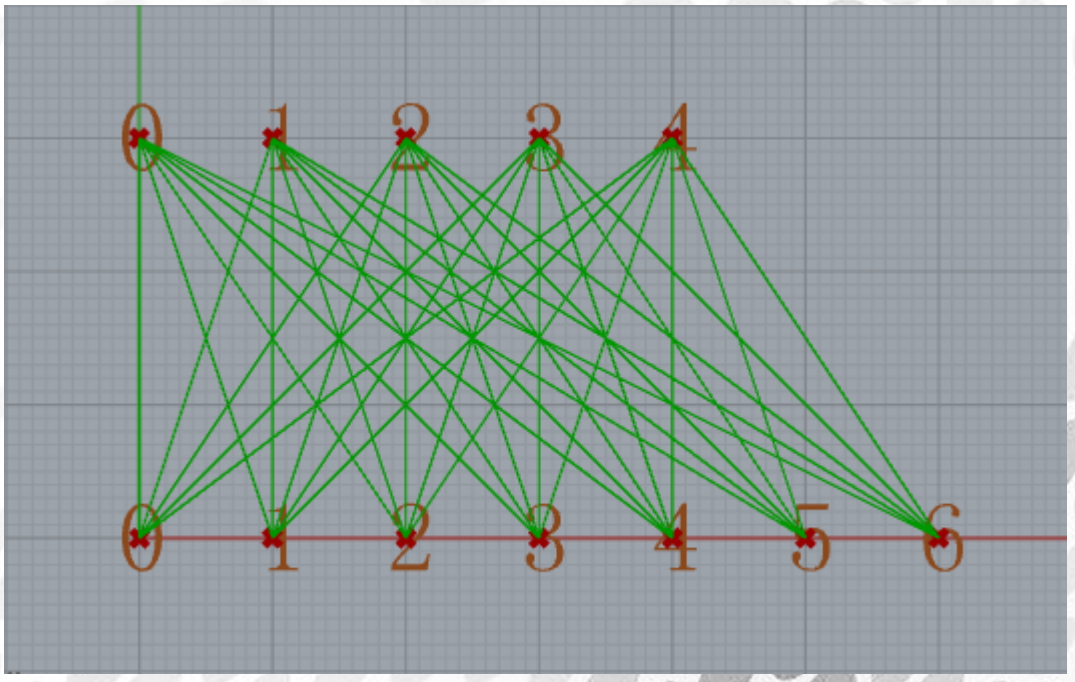


分别打开 1,2,3,4 连线，展示不同数据匹配方式连线的差别。



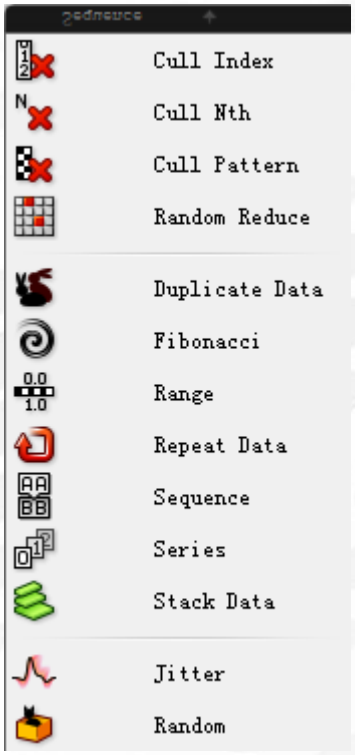




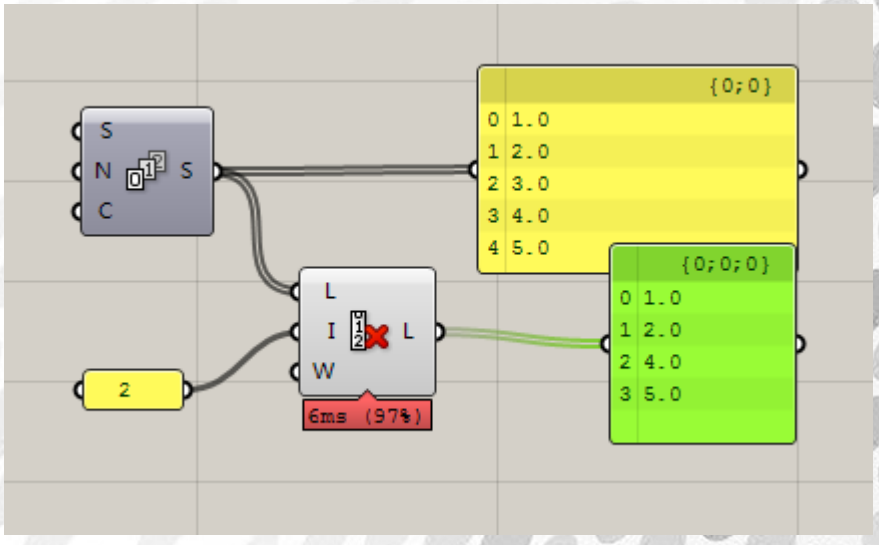


关于树形数据的认识，请在 csh.eeeetop.com 中阅读我发布的【By DanielJin】系列教程中的树形数据专题教程
关于数据匹配（数据对应原则），请查阅我转发的由 zhangzs 所著的《全面认识 Grasshopper 树形数据》

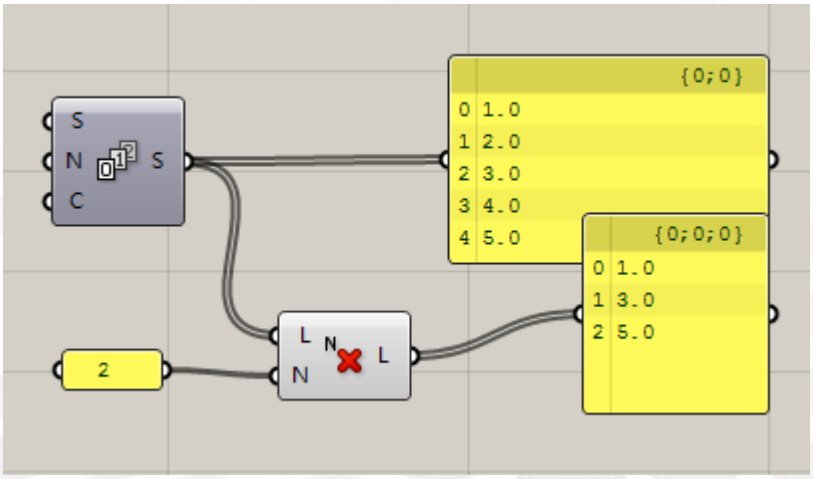
（2）Sequence 电池序列



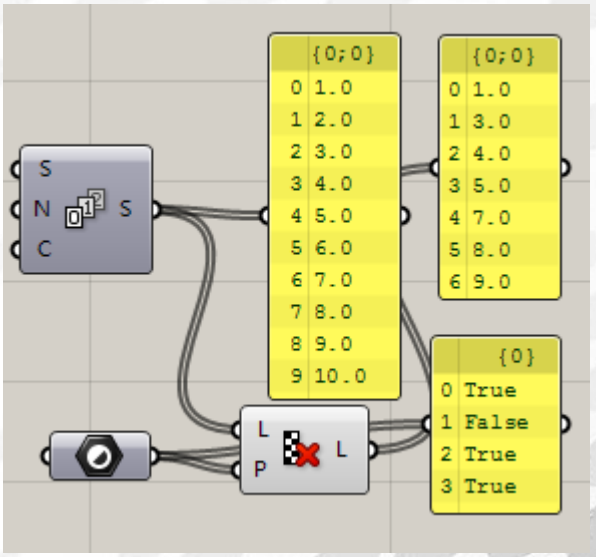
Cull Index:将列表上指定编号的数据删除
输入端 L:要删除的数据所属的原数据列表
输入端 I：要删除的数据编号



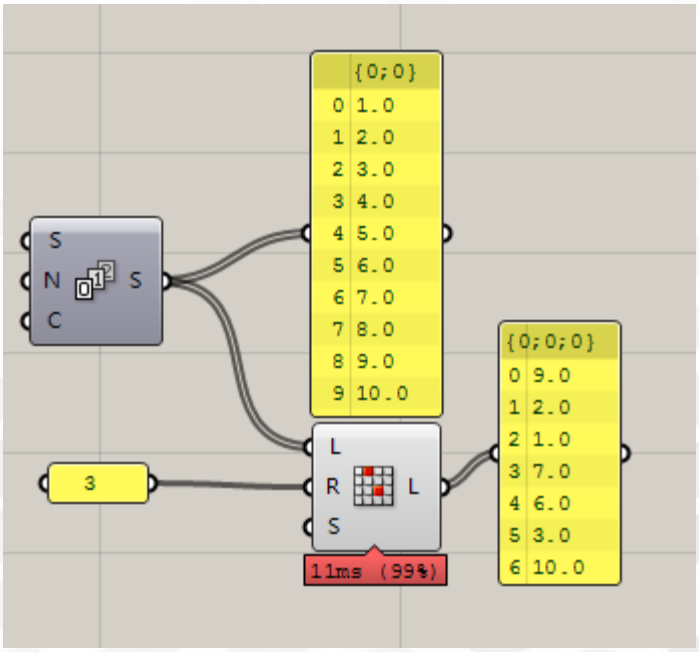
Cull Nth：移除列表中的第 N 个数据（往后循环，直到列表结束）
输入端 L:需要移除的数据列表
输入端 N:列表中的排列编号



Cull Pattern：根据布尔值来保留或删除数据
输入端 L:需要操作的数据列表
输入端 P:布尔值（循环）



Random Reduce:随即从一个列表中删除掉指定数目的数据。但是注意,该运算器会打乱数据!
输入端 L:输入被删除数据
输入端 R:删除掉数据的数量
输入端 S:随机值控制 Seed



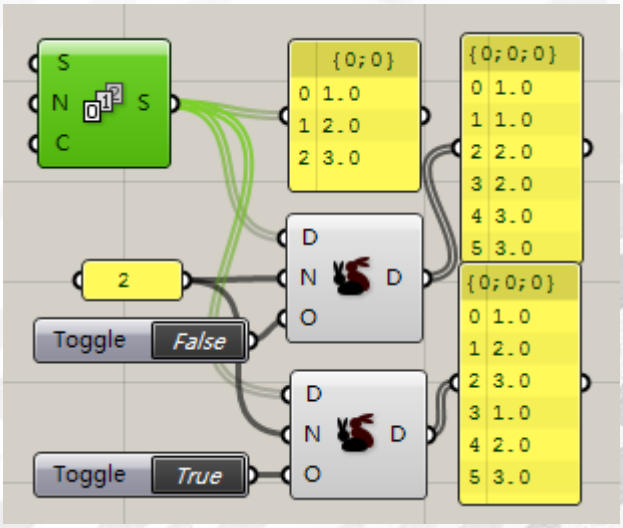
Duplicate Data:复制数据

要注意,该运算器的图标因为有些不够纯洁,在”和谐模式”中图标是另一个样子

输入端 D:需要复制的原始数据

输入端 N:需要复制多少次

输入端 O:布尔值,是否保持数据排序



Fibonacci:大名鼎鼎的斐波那契数列,设置 A,B 两项，第 N 项等于前两项综合

Range:将一个给定范围区间等分

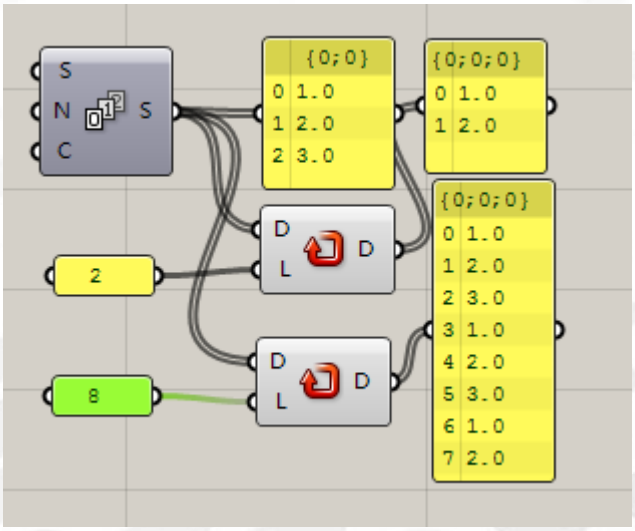
输入端 D:范围区间

输入端 N:平均分成多少分

Repeat Data:按照给定长度循环数据列表

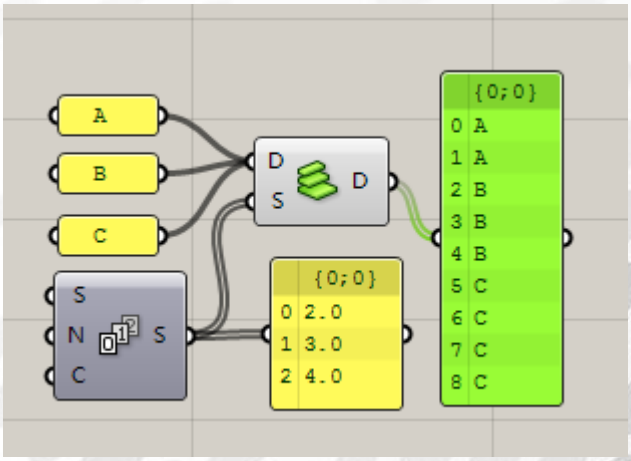
输入端 D:输入数据列表

输入端 L:循环的长度



Sequence:根据设置列表长度和预排文字产生列表数据
输入端 C:列表数据的数量
输入端 P:预排文字

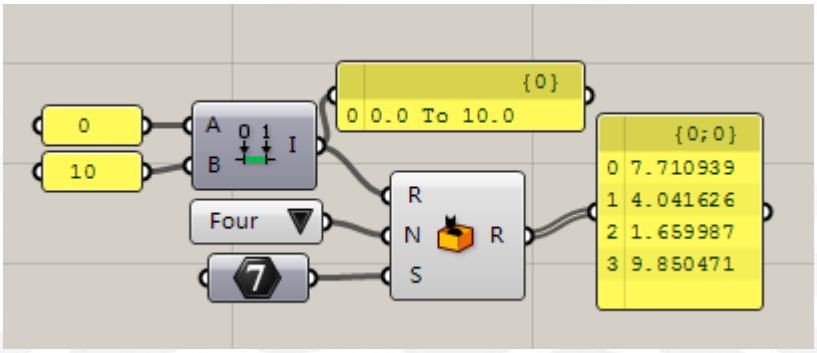
Stack Data:根据指定堆栈数量,来生成列表数据
输入端 D:需要堆栈的数据列表
输入端 S:堆栈的数据列表



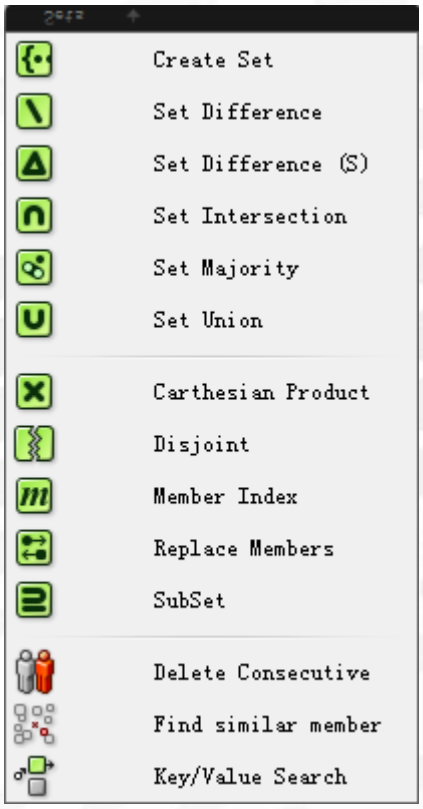
Jitter:打乱数据重新排列
输入端 L:需要打乱的数据
输入端 J:打乱强度
输入端 S:打乱 seed
运用实例图示上边已经提供了一个了。

Random:非常常用的运算器,随机产生一组数.
输入端 R:随机生成的范围(Domain)
输入端 N:随机生成的个数
输入端 S:随机的 seed

常用组合情况如下:



（3）Set 电池序列



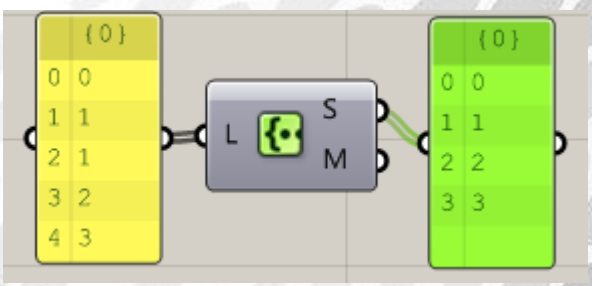
Create Set：创建集合

官方直译是创建一个只包含不同独立数据的集合，翻译成通俗的人话就是：去重复...

输入端 L：建立集合前原始数据

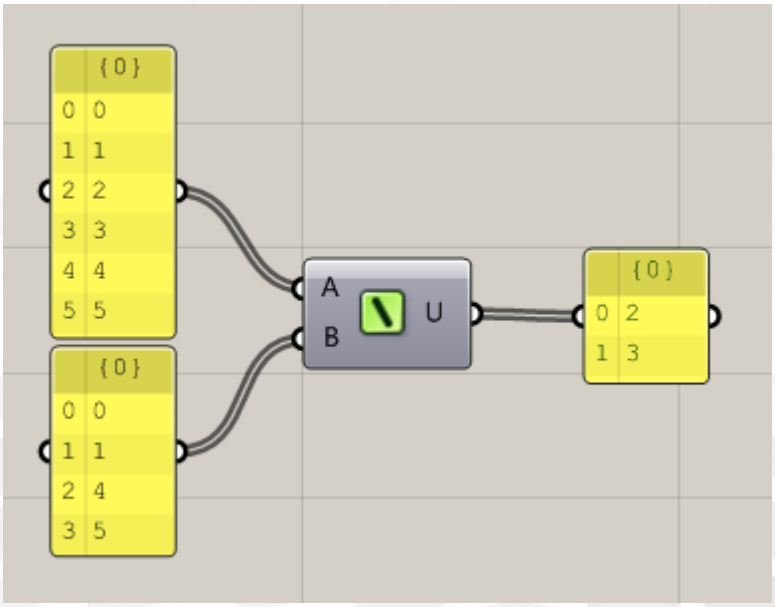
输出端 S：建立后的集合

输出端 M：集合编号



Set Difference：设置集合差异

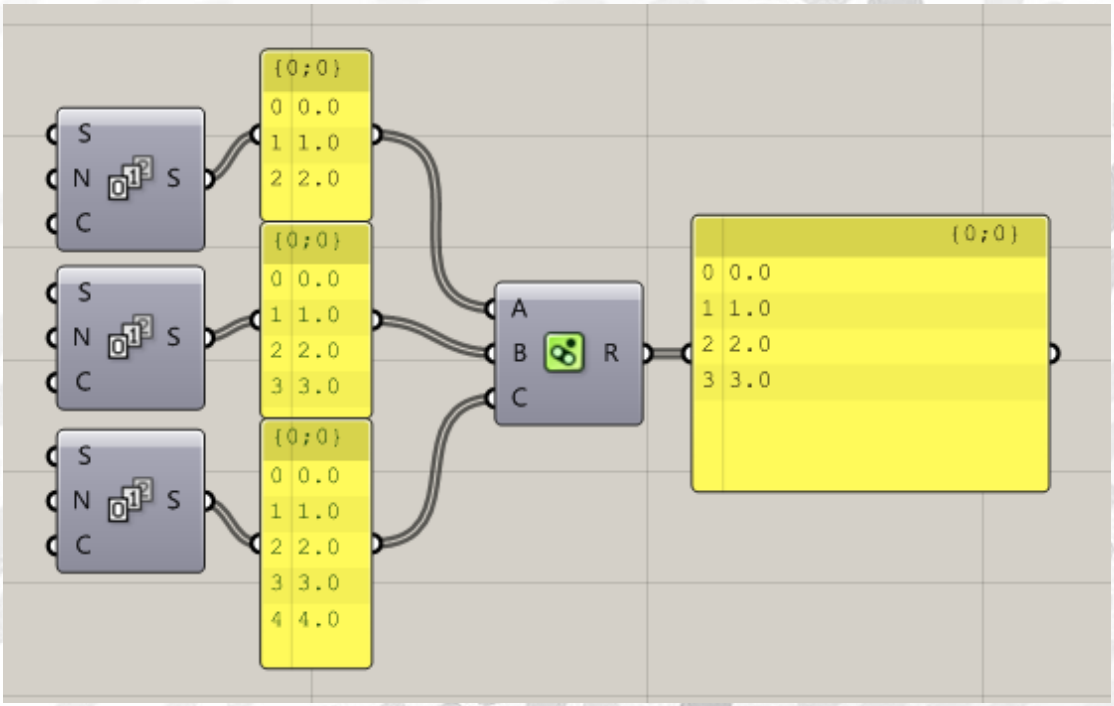
将输入端 A 集合中不同于 B 的部分输出。



Set Difference(S)：集合差异数据的合并

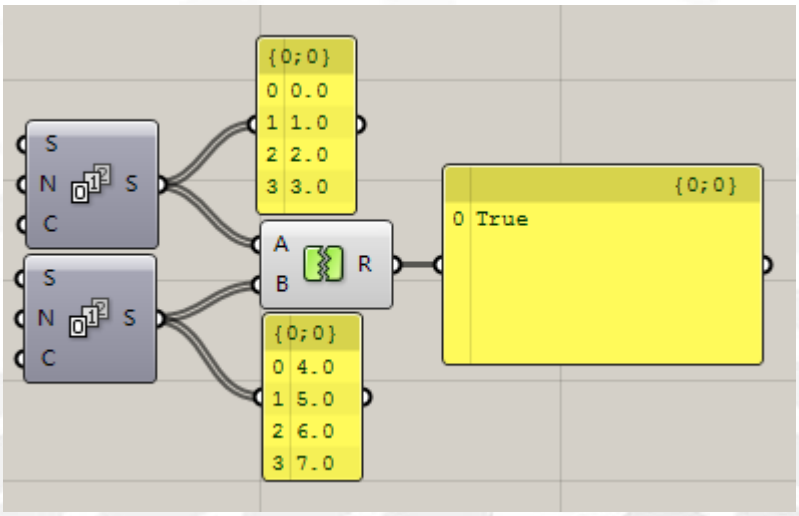
Set Intersection：交集
Set Union：并集

Set Majority：输出三个集合中至少两个相同的数据，少于两个相同则不输出。

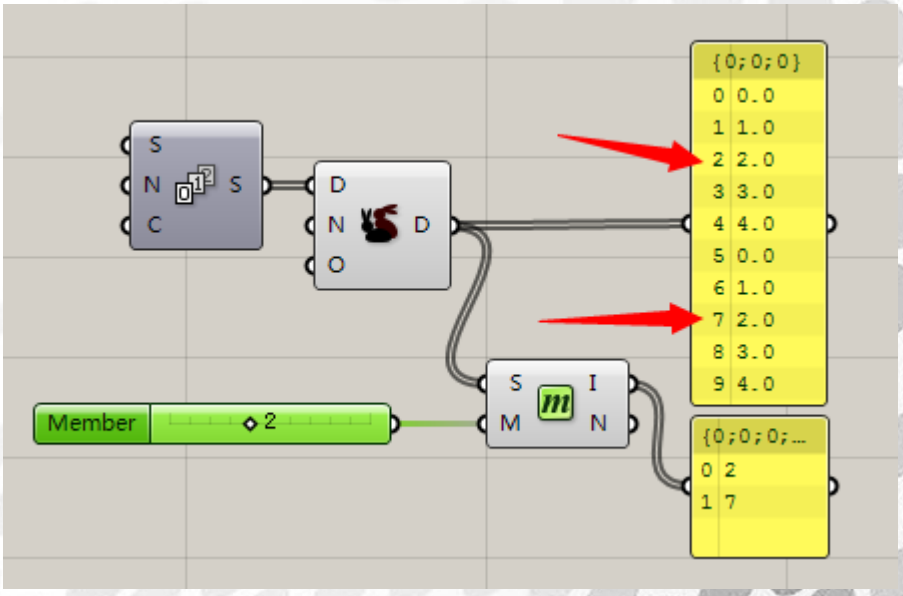


Carthesian Product：笛卡尔乘积：两类集合（A 和 B）按照次序组成一个个局部列表（树形结构），但：A 或者 B 集合内，他们本身内不能存在相同的数据，否则判定错误。

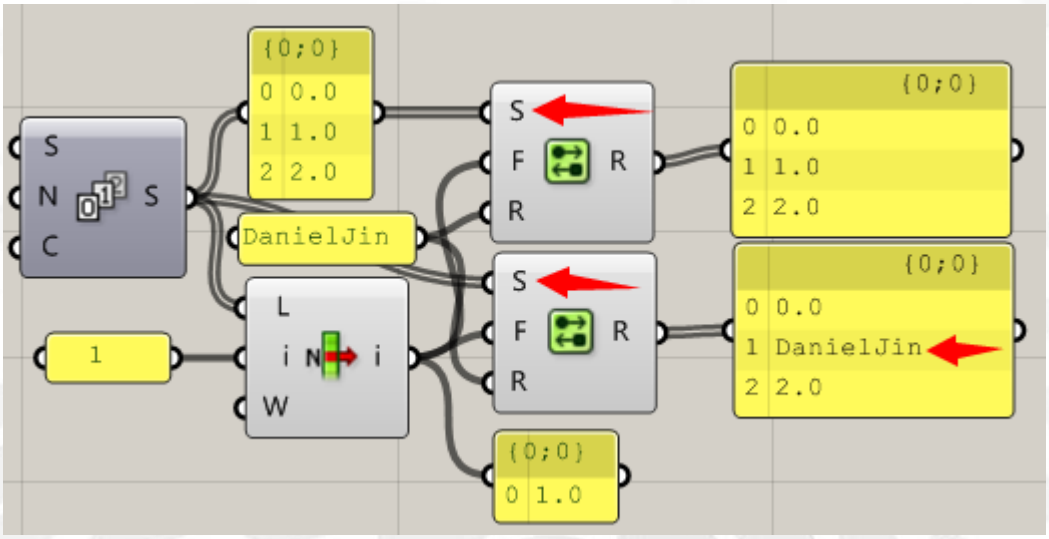
Disjoint：集合 A 与 B 若不相交则为真，相交为假。



Member Index：查找 M 是否在集合 S 中。
输入端 S:查找集合
输入端 M：查找的数据
输出端 I：查找到的数据 M 在 S 中的编号
输出端 N：查找到的数据的个数

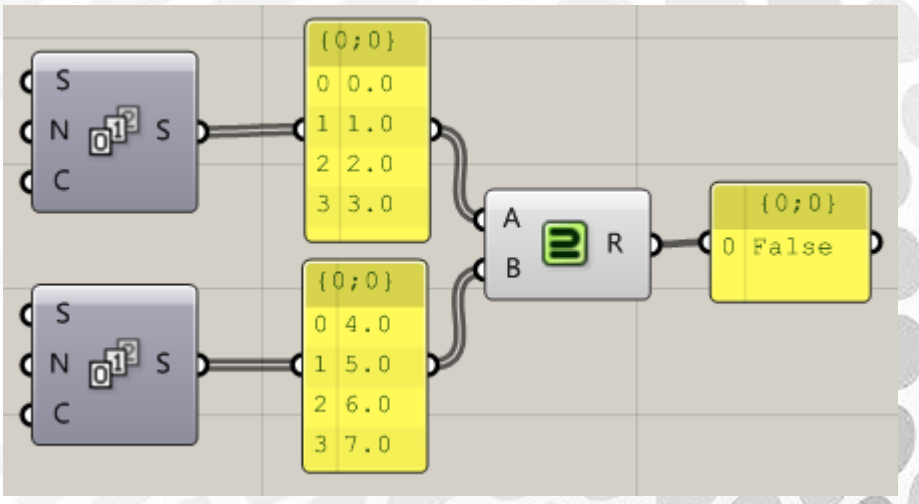


Replace Members：在集合 S 中寻找数据 F，并替换成 R

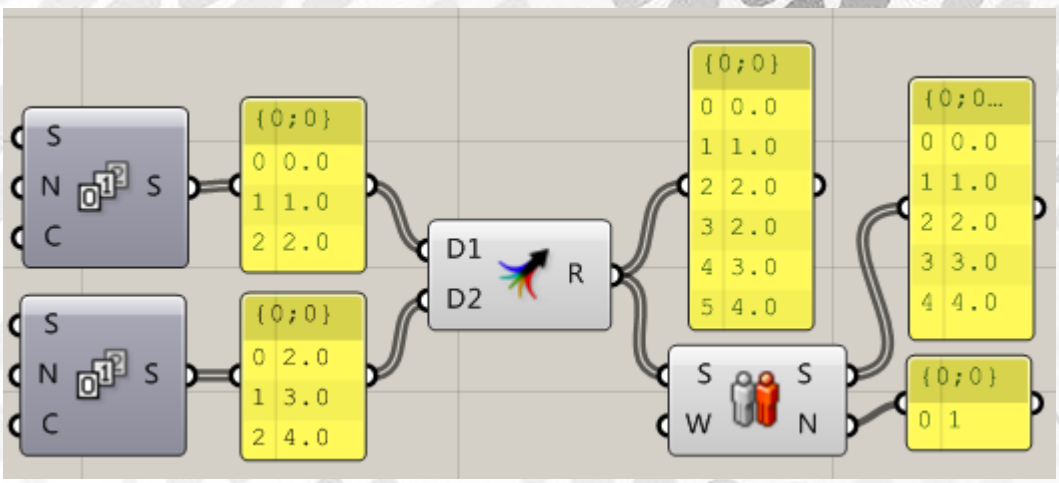


同时提醒各位，Panel 的不适用范围可以从这个案例中看出一些。

SubSet：集合 A 与 B 若存在相互包含关系则为真。



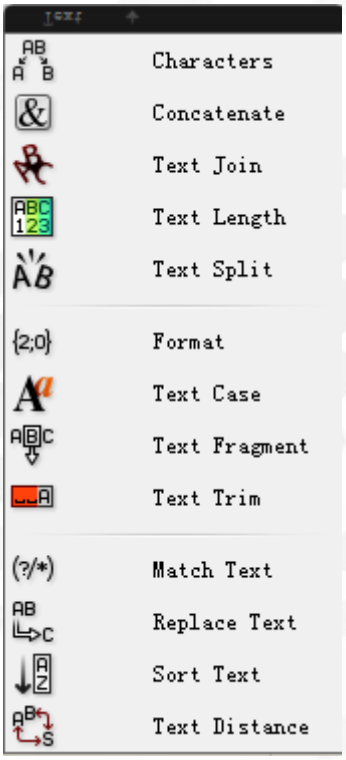
Delete Consecutive：在集合数据中删除掉连续排列相同的数据。
输出 S：删除后的结果
输出 N：删除数据的数量



Find Similar Member：寻找近似成员
对比输入数据 P 和输入集合 S，将 S 中近似于 D 的数据从 H 端输出，i 为他们的编号。

Key/Value Search：通过 key 搜索得到对应值
输入端 K：关键 key
输入端 V：寻找 Key 对应的值
输入端 S：寻找 key，得到对应值

（4）Text 电池序列



Characters:分解字母
输出端 C：得到分解后的字母
输出端 U：Unicode 值

Concatenate：拼接输入 A 和 B，相当于 A+B

Text Join：在列表 T 的空当处插入 J，合并输出结果 R

Text Length：Text 的字节长度

Text Split：按照指定的输入端 C 分割输入端 T，得到分割后的结果 R

Format：将输入端的值按照输入端 F 按照统一格式输出

Text Case：转换大小写

Text Fragment：根据输入的 i 值删掉输入 T 端的第 i 个字节。N 端决定保留随后的几个字节

Text Trim：删除掉输入 T 前后的空格

Match Text：检查输入端 T 与 P 是否一致，从而得出布尔值

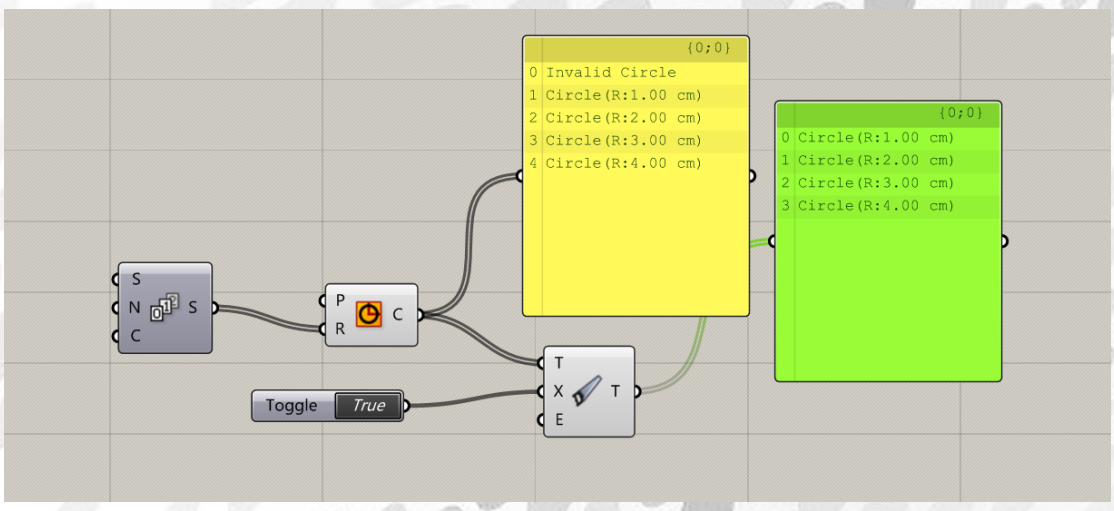
Replace Text：替换字符
输入端 T:准备替换的 text
输入端 F：找到的 text
输入端 R：用来替换的 text

Sort Text：排列 Key 和 Value

Text Distance：输入端 A 的 text 总长度减去输入端 B 的 text 长度然后输出结果。

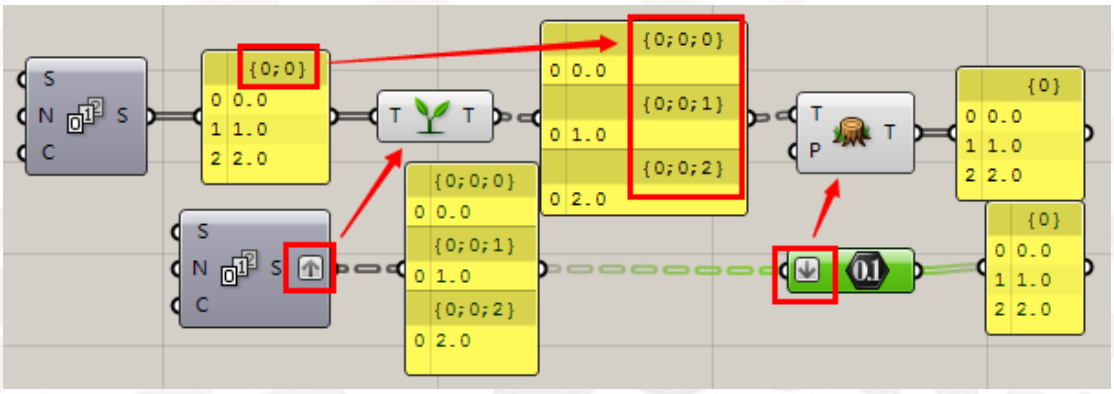
（5）Tree 电池序列

Clean Tree：清理一组数据的 Null 和 Invalid 部分
输入端 X：布尔值，X 为 true 删除 Invalid 项
输入端 E：布尔值，E 为 true 删除 Null 项

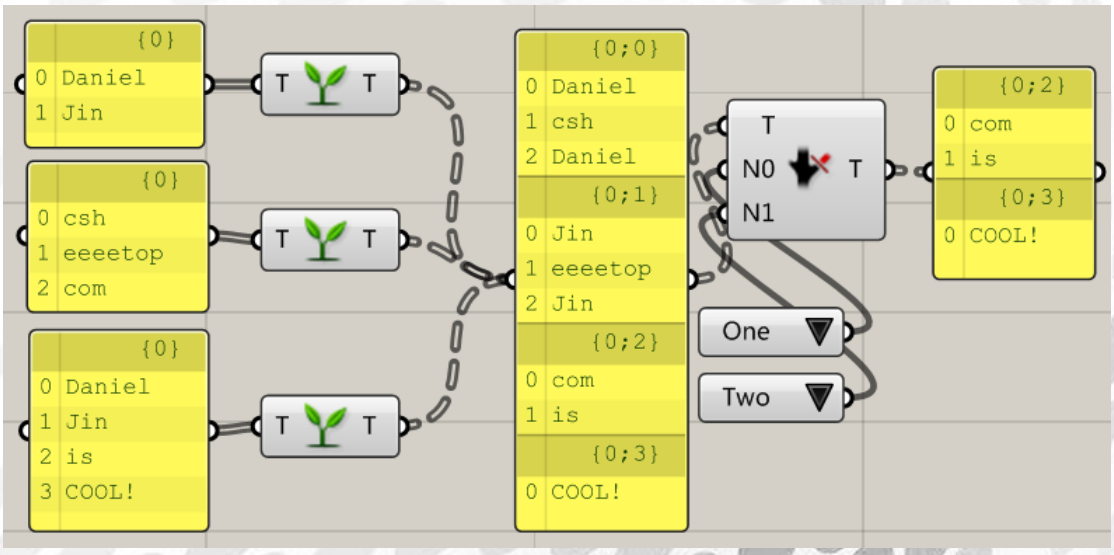


Flatten Tree：拍平树形数据
输入端 P：输入路径编号
Graft Tree：转为为树形数据
输入端 P：输入路径编号

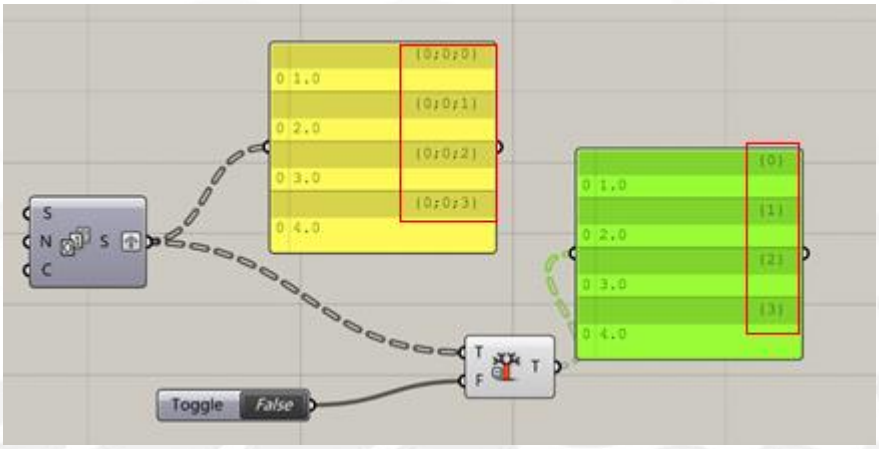
这里需要注意的有两点，1 这两个运算器和右键菜单中的 Graft，Flatten 作用一样。2 请各位仔细看路径名称变化，这些无意义的上级路径{0,0}我习惯称为无效路径。当然这是我个人习惯的称呼。我在个人教程中专程花了两篇帖子来讲解该部分内容。请登录 csh.eeeetop.com 学习树形数据的教程。



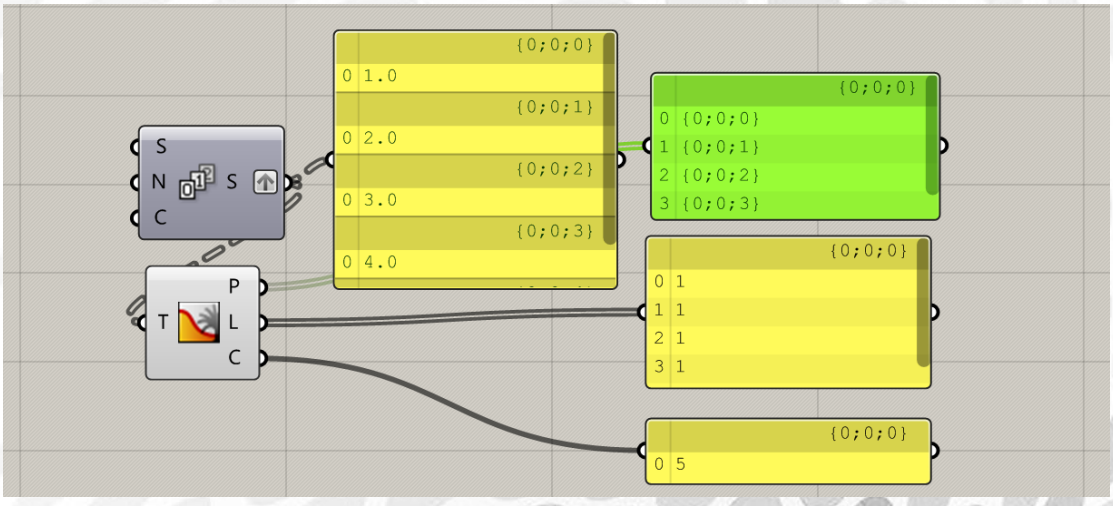
Prune Tree：用于砍掉一个树上的树叶数在指定范围之外的树枝
输入端 N0：范围上限
输入端 N1：范围下限



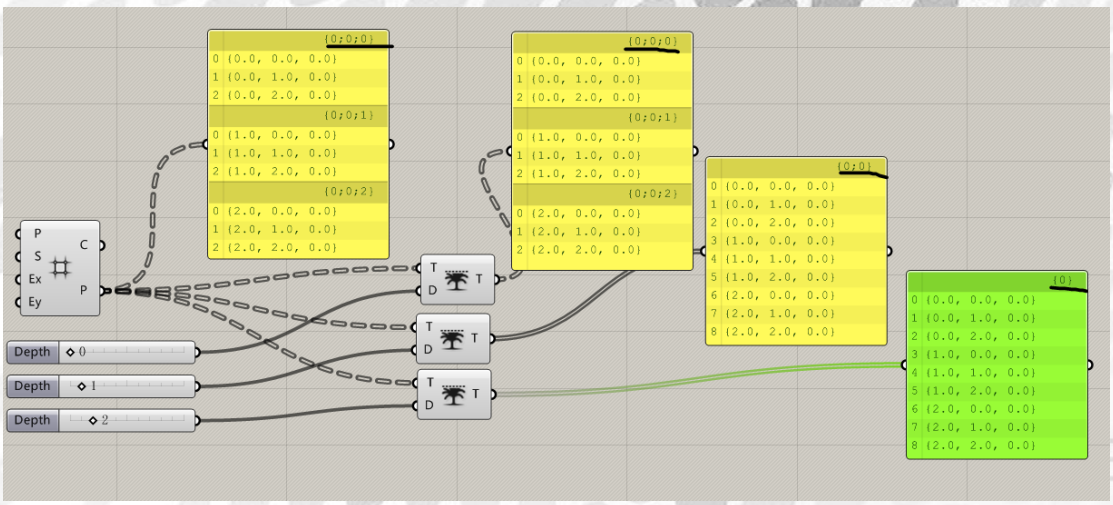
Simplify Tree：简化树形数据
该功能和右键运算器菜单中的 Simplify 等同
输入端：布尔值，在开始简化的时候限制路径的路径指数



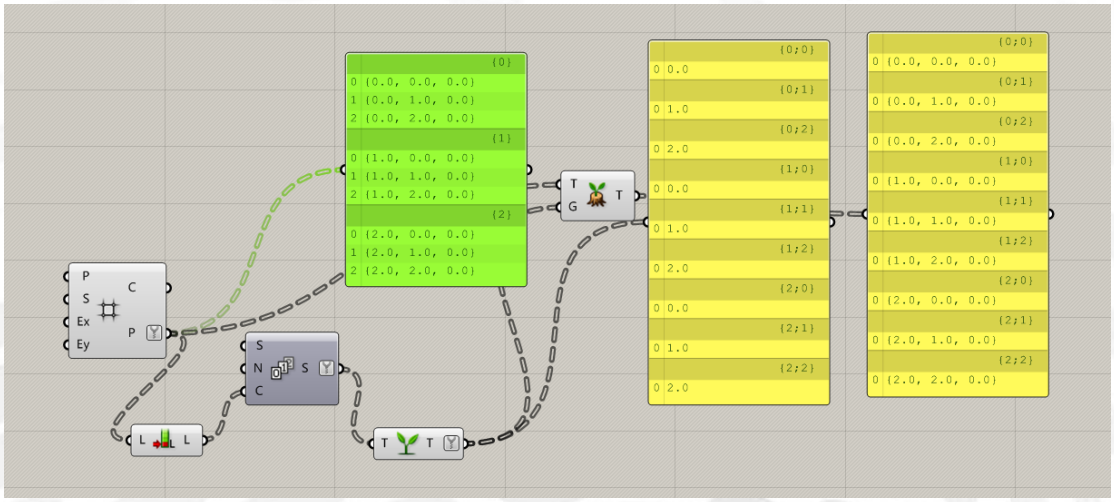
Tree Statistics：路径分析
输出端 p：输出路径编号
输出端 L：输出树枝下的树叶数
输出端 C：输出树枝数



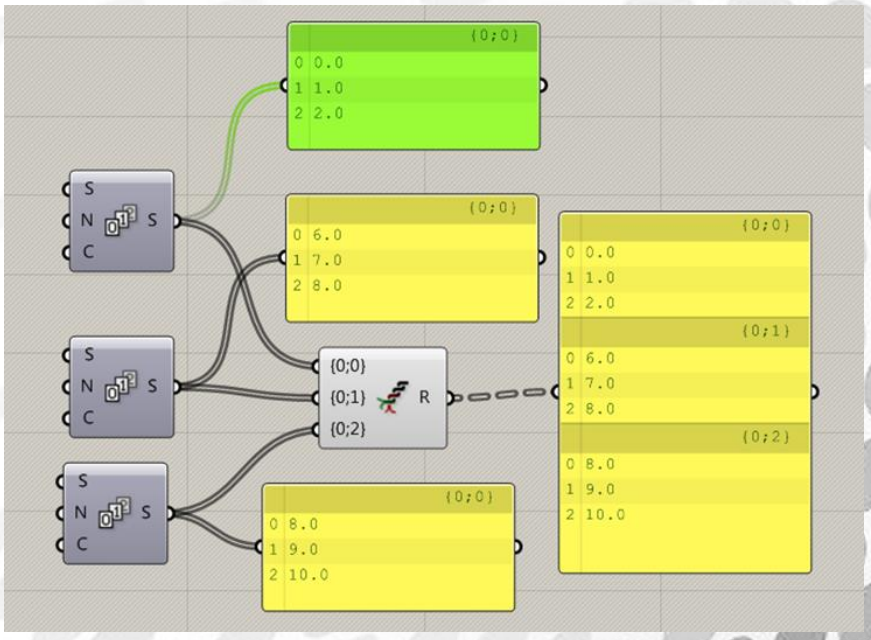
Trim Tree：修剪树形数据
输入端 D：向前修剪级别数



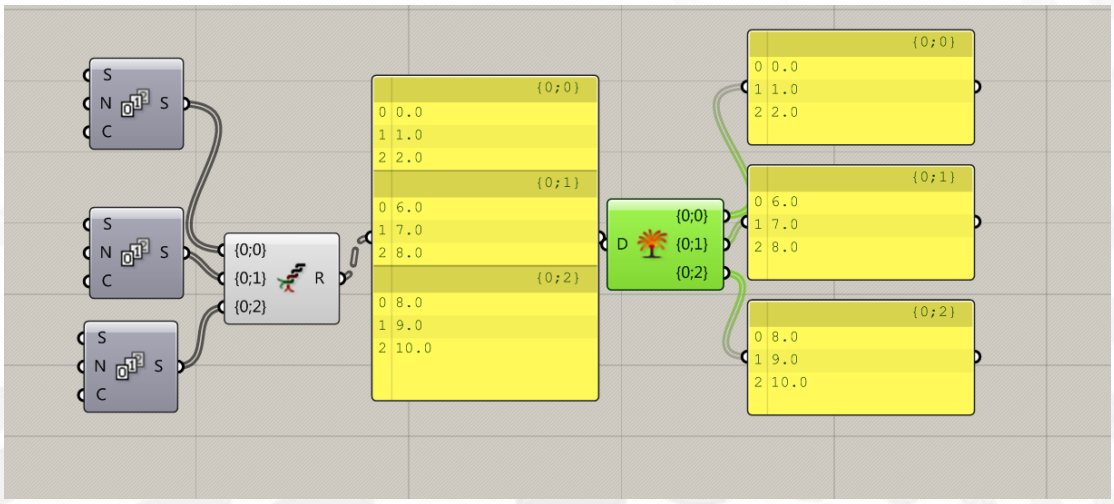
Unflatten Tree：
输入端 G：指定编号转化为树形数据



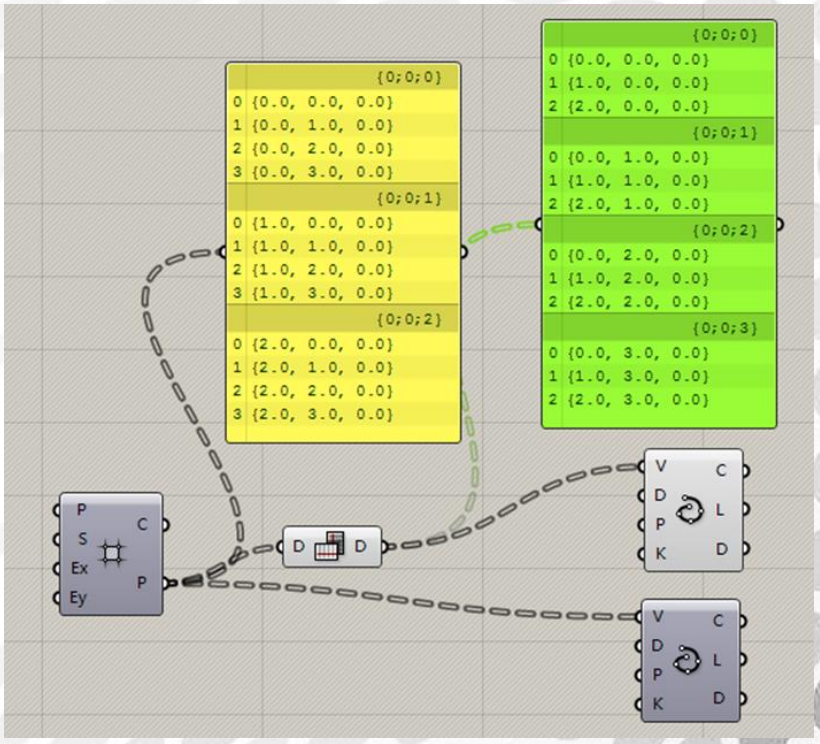
Entwine：数据合并
输入各树枝组成树形数据

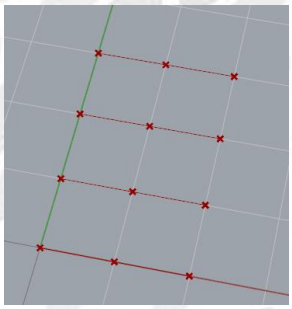


Explode Tree：分解数据
将树形数据拆散成单个树枝

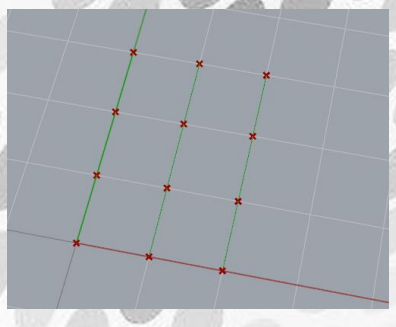


Flip Matrix：数据倒置

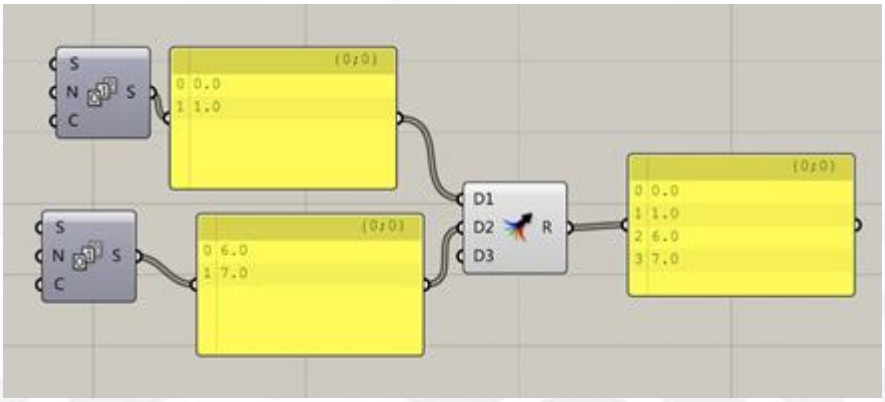




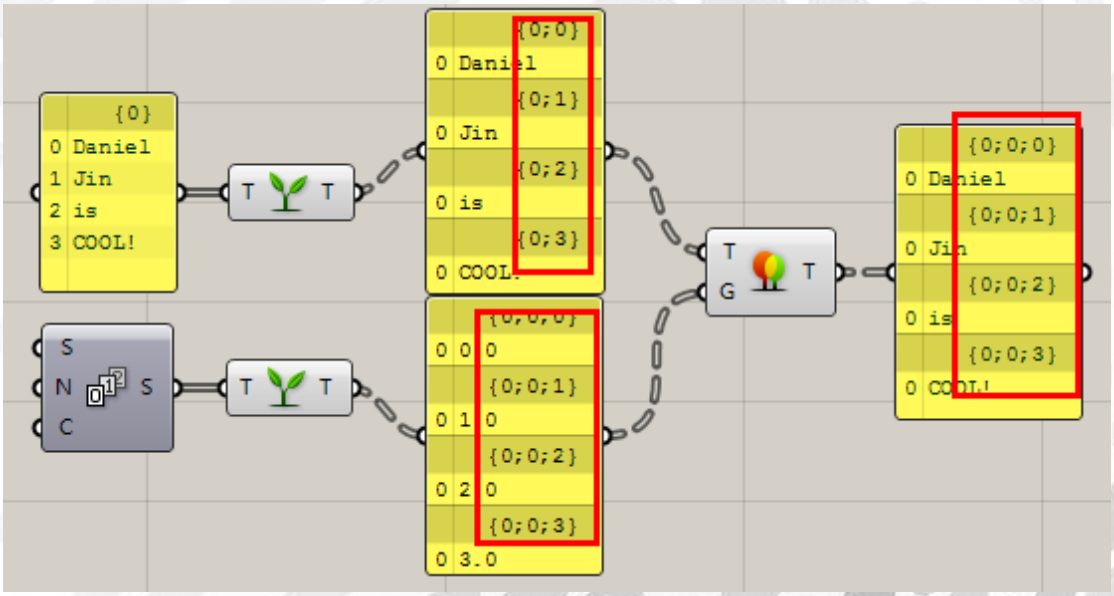
数据倒置后重新连线



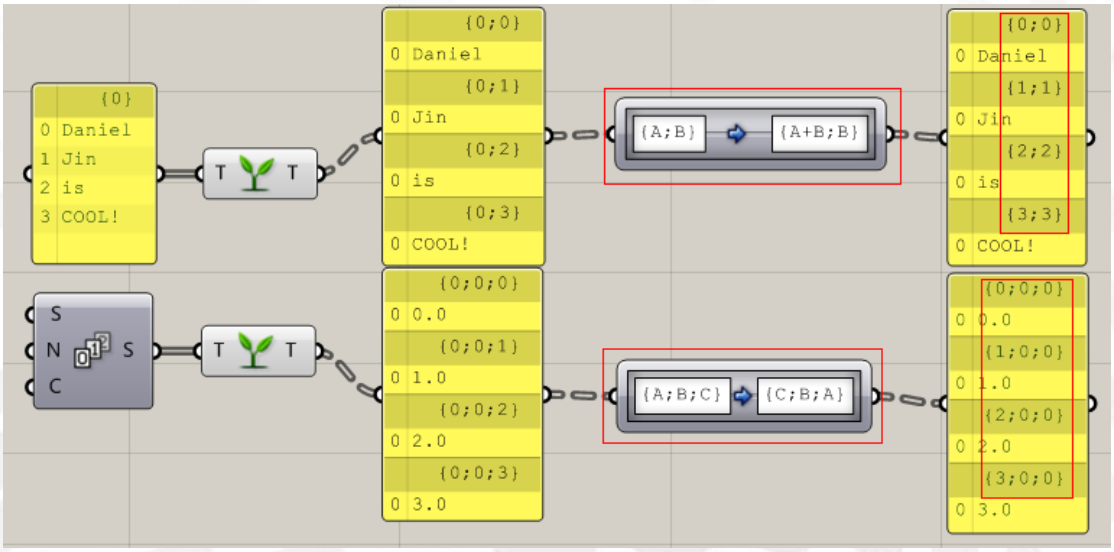
Merge：合并数据



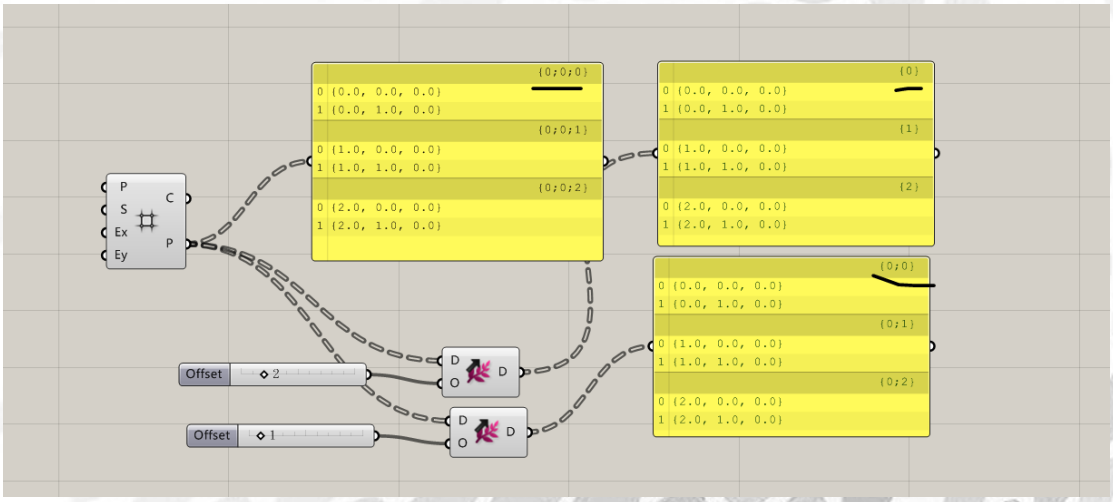
Match Tree：匹配数据路径
输入端 T：要匹配路径编号的数据
输入端 G：匹配路径的数据



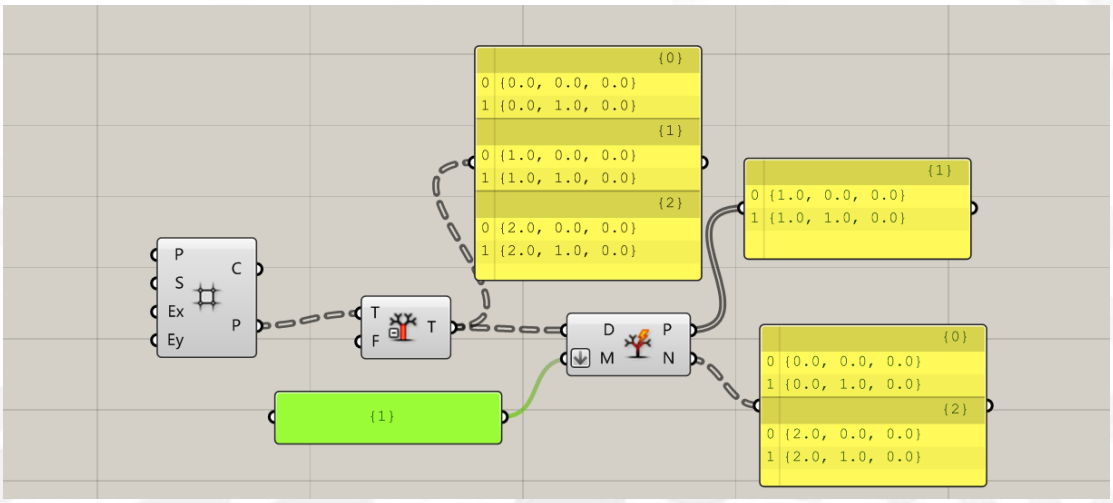
Patch Mapper：路径编辑器
此运算器用法较多，请在右键菜单中观察详细使用情况。在此举两例：



Shift Trees：移除路径
输入端 O：向前剔除级数

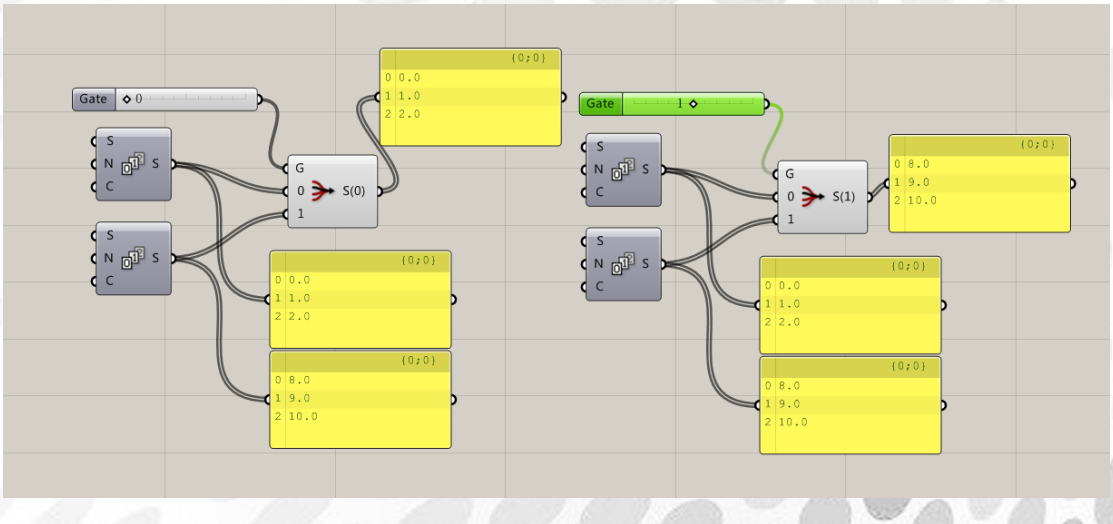


Split Tree：分裂树形数据
输入端 M：输入开始分裂的树枝
输出端 P：分裂出的树枝
输出端 N：剩余的树枝



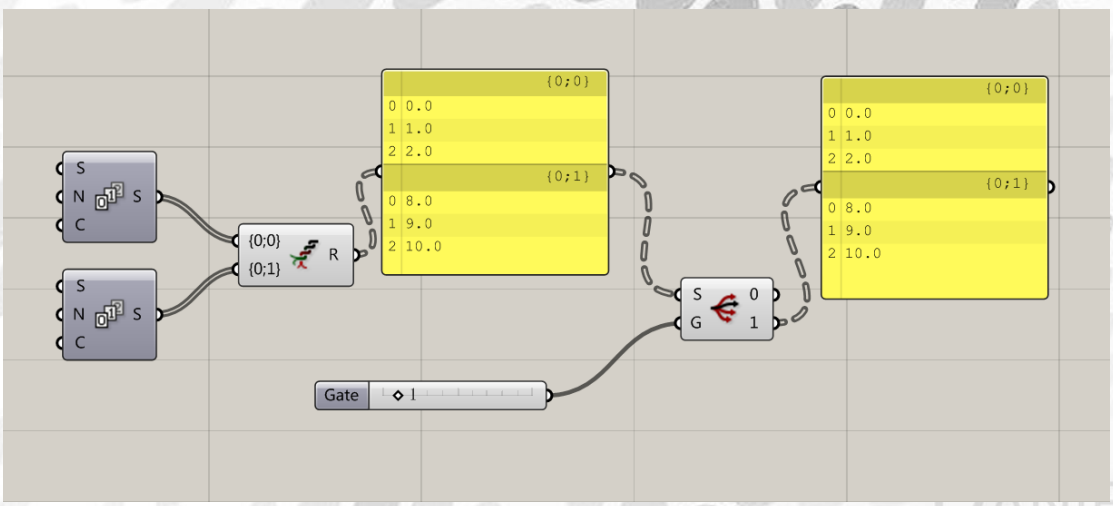
Stream Filter：数据分流开关

输入端 G：指定输出端输出那组数据



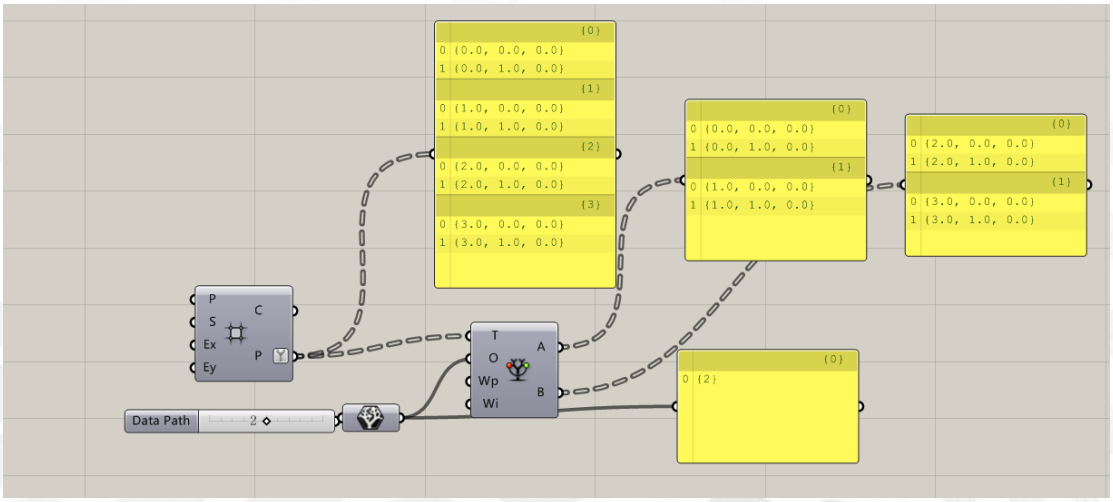
Stream Gate：数据分流开关

输入端 G：控制数据从哪一端输出

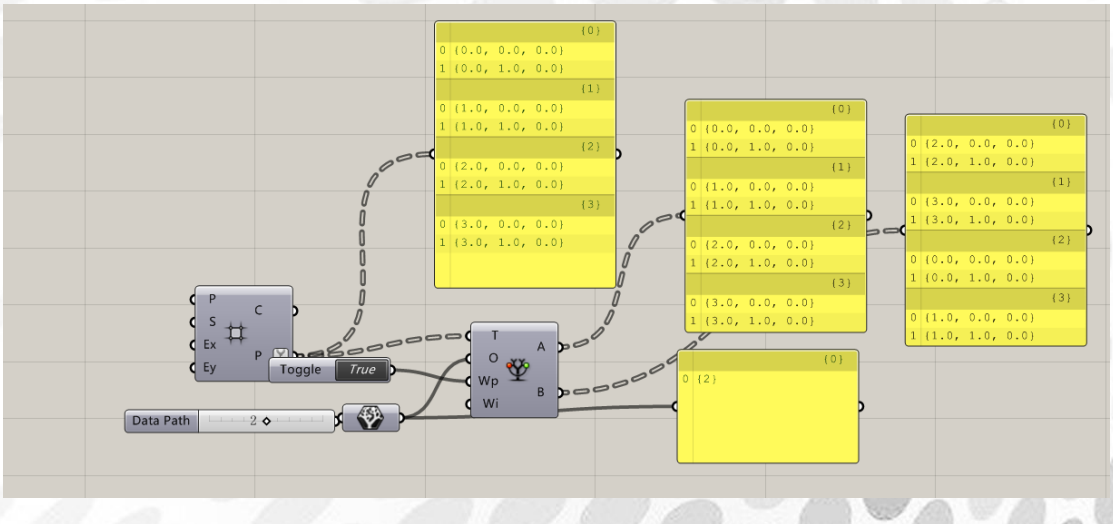


Relative Item：相对分裂数据

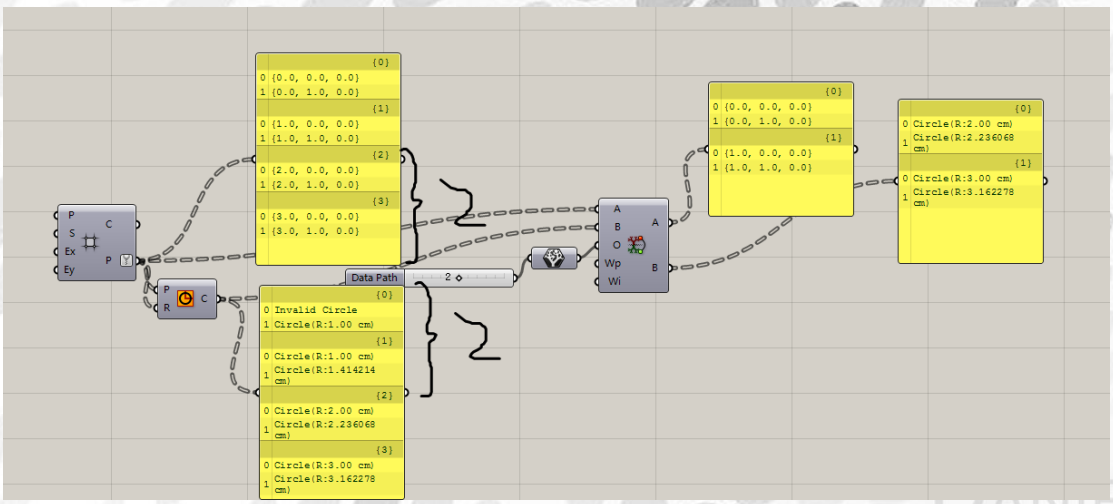
输入端 O：指定分裂数据开始路径
输入端 Wp：布尔值，为 true 则追加分裂出去的数据



Wp 为 true 时

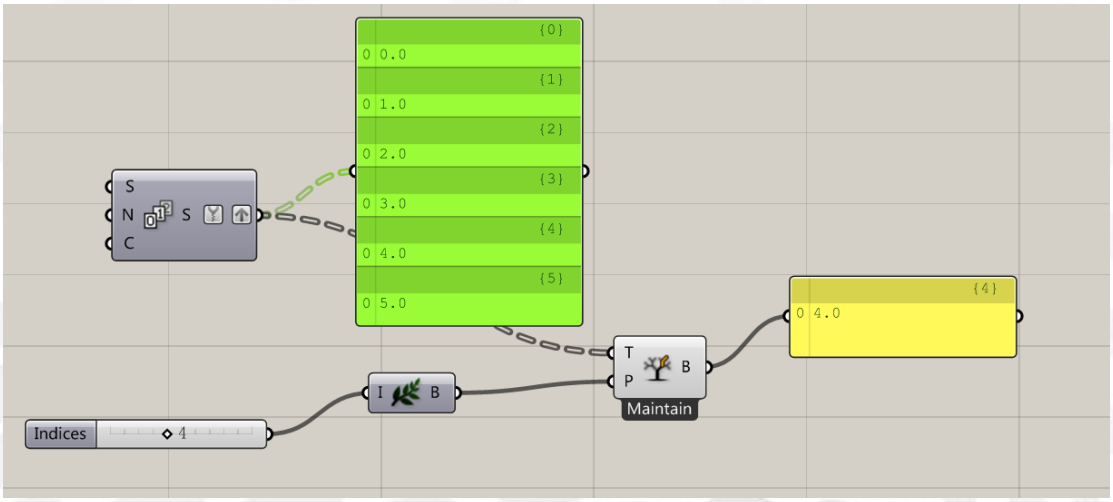


Relative Items：相对分裂两组数据
输入端 O：剔除 A 组数据的末端和 B 组数据的始端树枝数

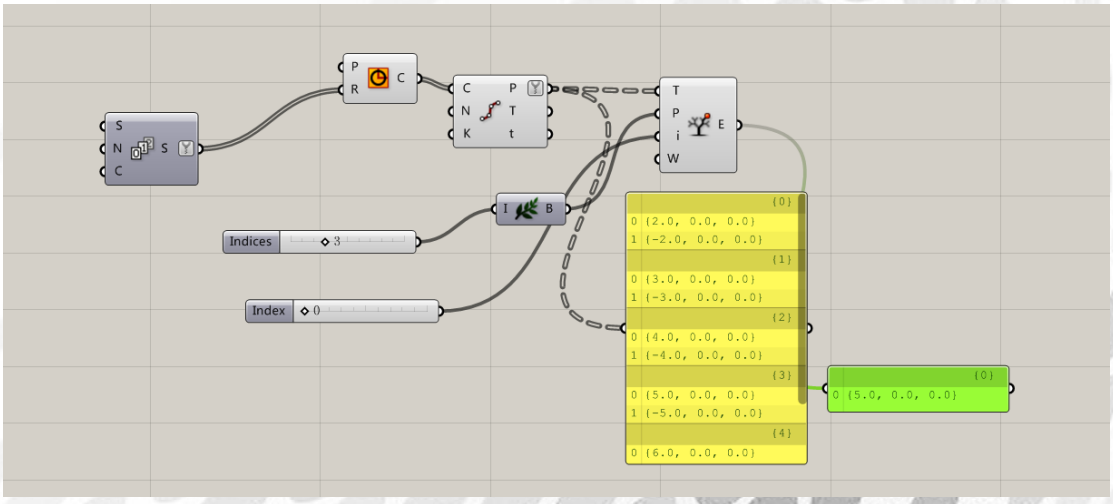


Tree Branch：抽取特定树枝

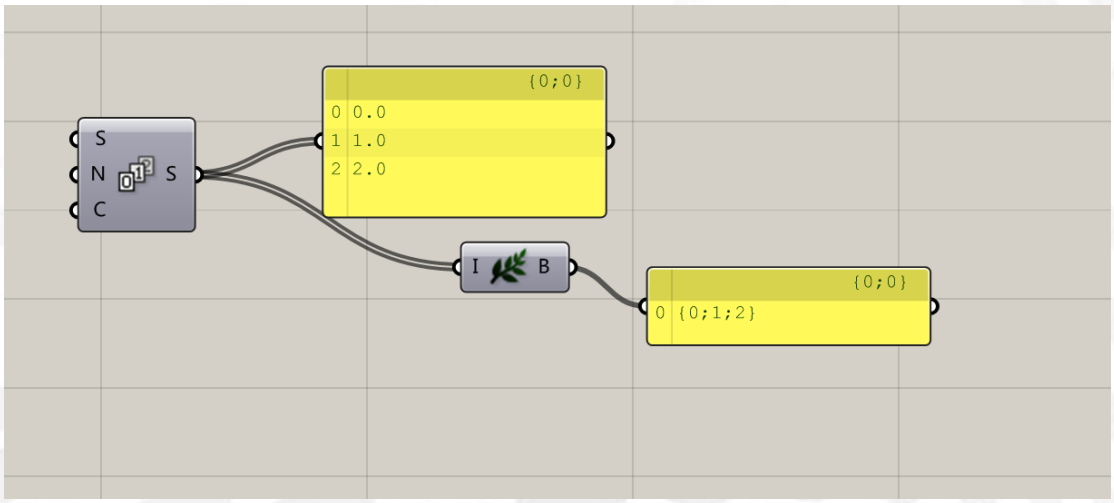
输入端 P：输入树枝编号
输出端 B：输出抽取到树枝



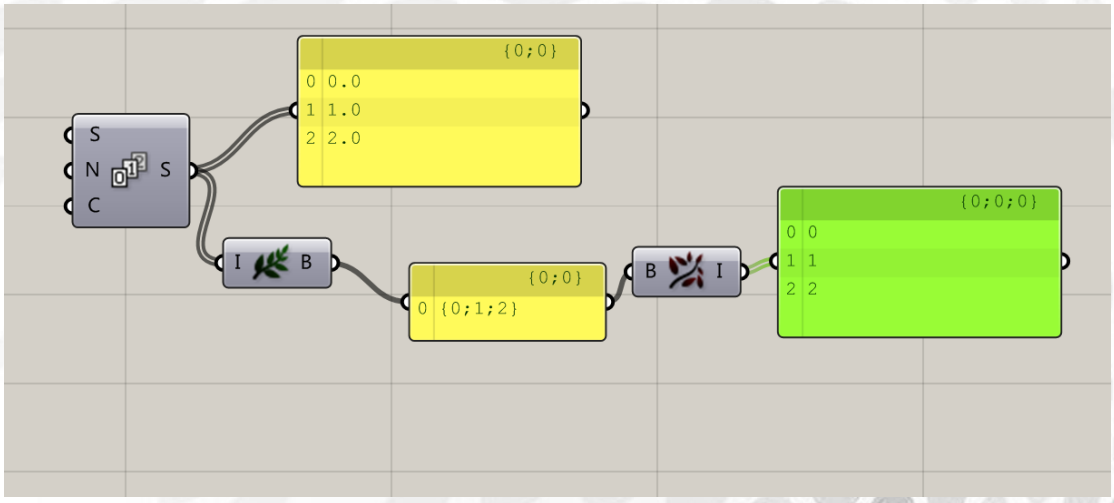
Tree Item：抽取特定树枝上的特定数据
输入端 P：输入要抽取的树枝编号
输入端 i：输入数据编号
输入端 W：布尔值



Construct Path：将一组数据转化为路径编号



Deconstruct Path：将路径编号转为一组数据

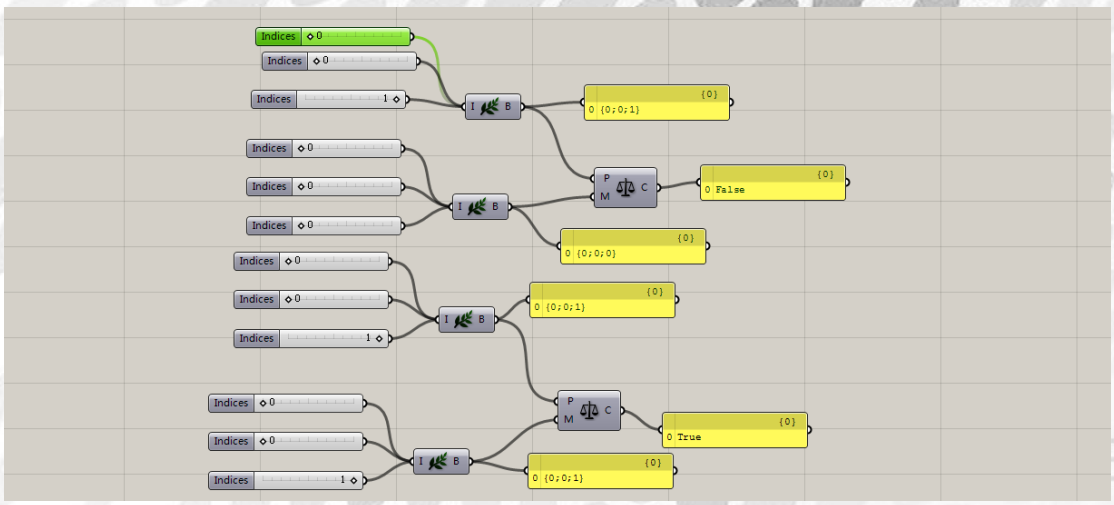


Path Compare：用于判断路径是否与掩码匹配

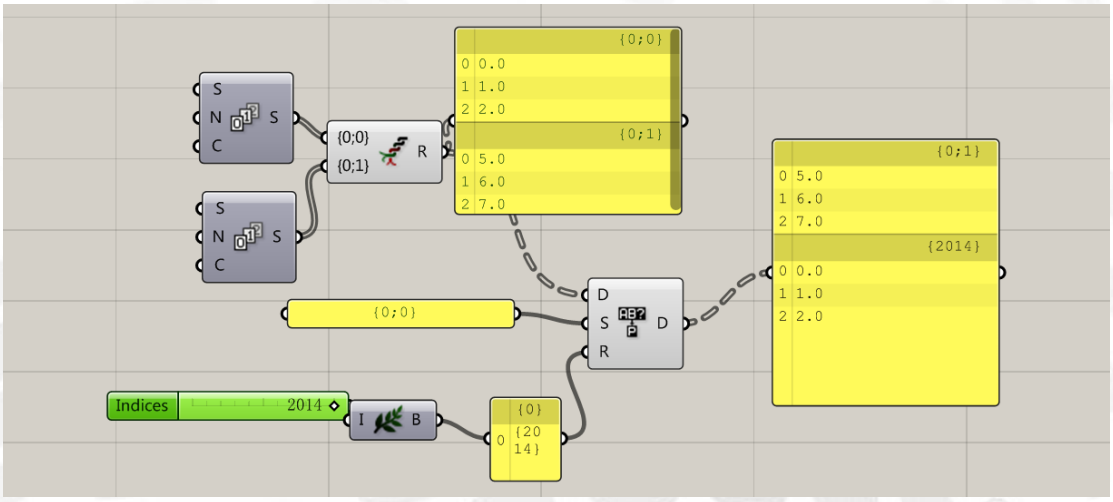
输入端 P：要判断的路径

输入端 M：掩码

输出端 C：路径相同输出 true 反之为 false



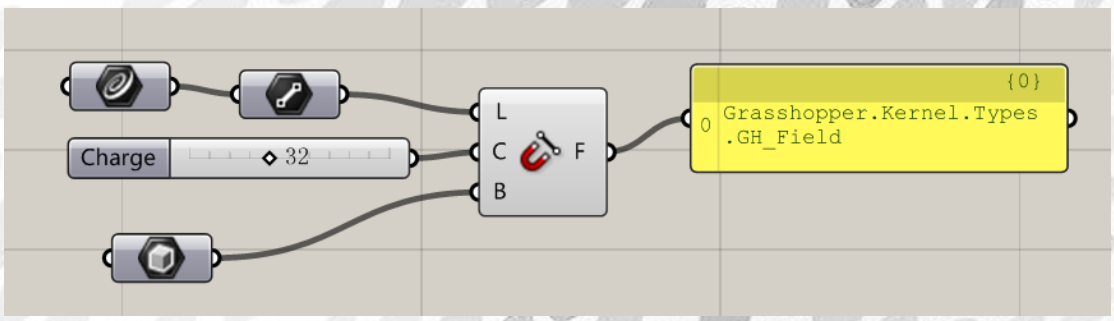
Replace Paths：利用掩码把树上的树枝选出来，再集合到指定的树枝上
输入端 S：输入掩码指定抽取的树枝
输入端 R：输入要重新集合的路径编号

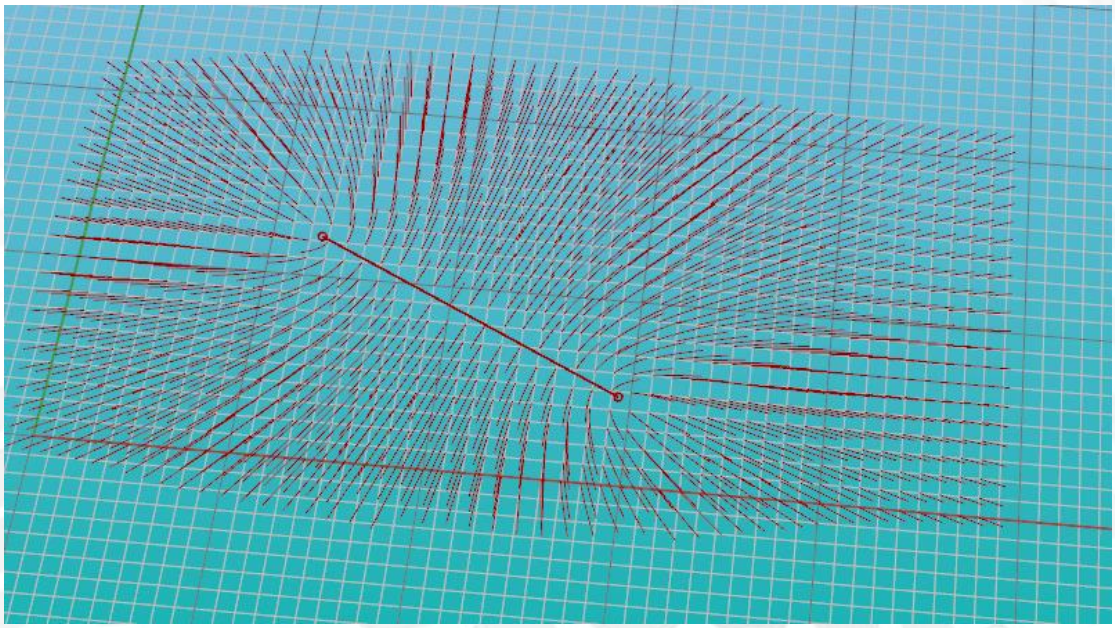


4. vector 电池组

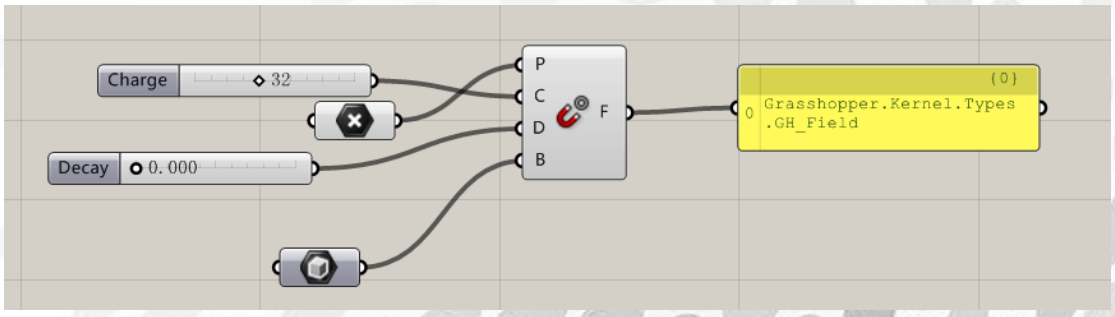
（1）Field 电池序列

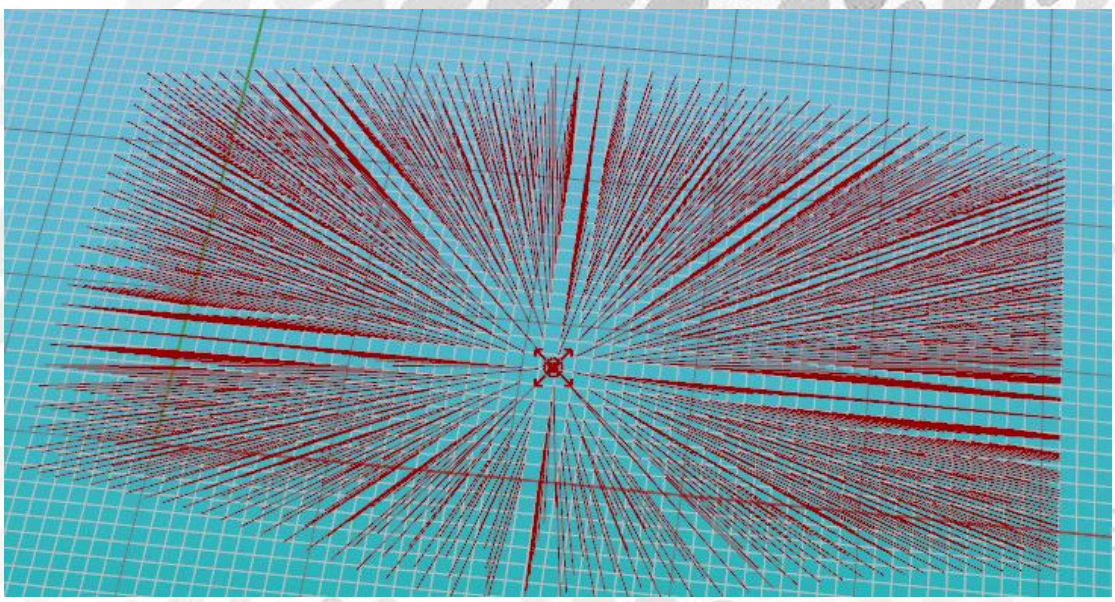
Line Charge:建立线电荷场
输入端 L:产生电荷场的线段
输入端 C:电荷强度
输入端 B:产生电荷的区域



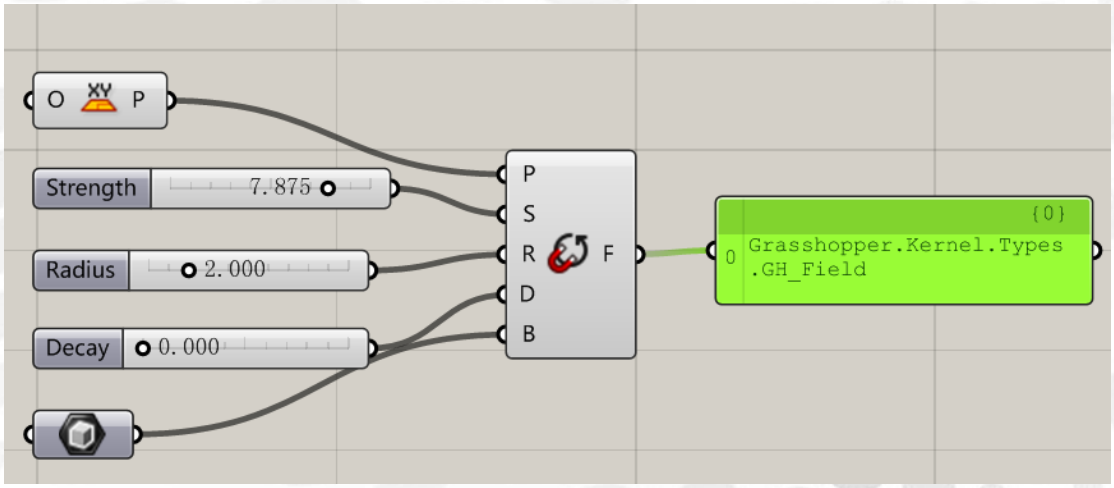


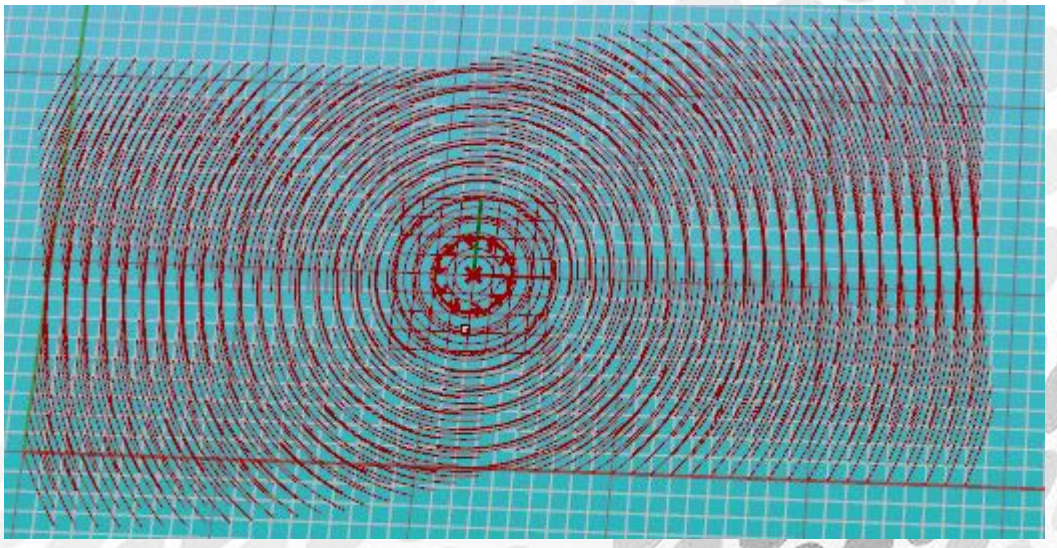
Point Charge:创建一个点电荷场
输入端 P:产生电荷场的点
输入端 C：控制电荷强度
输入端 D:电荷衰减值
输入端 B:生成电荷的区域



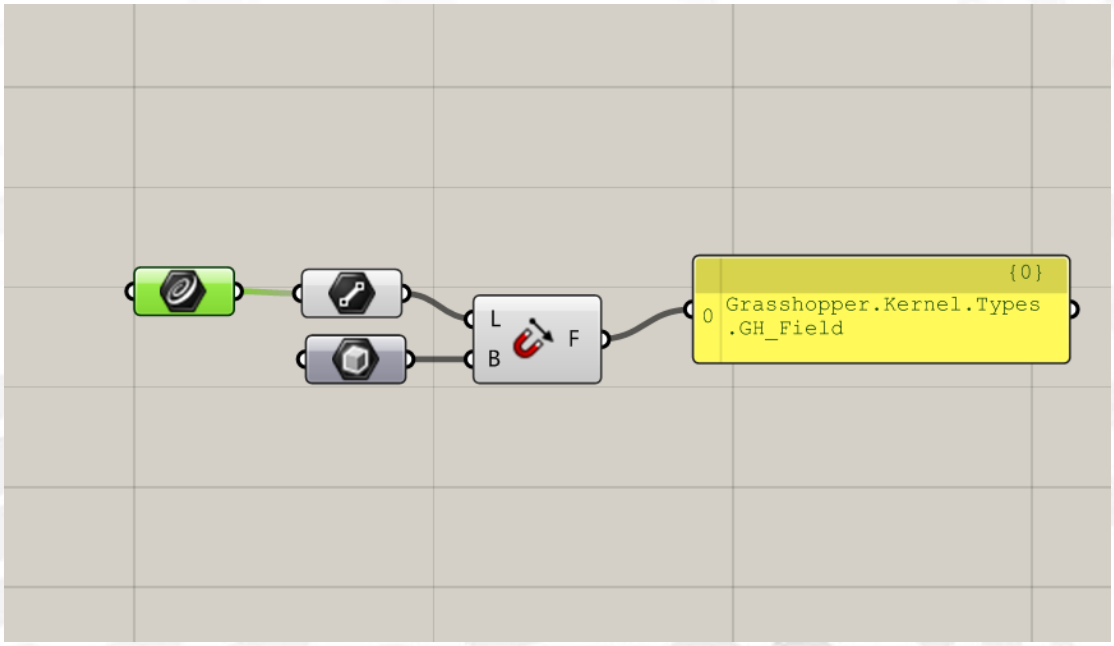


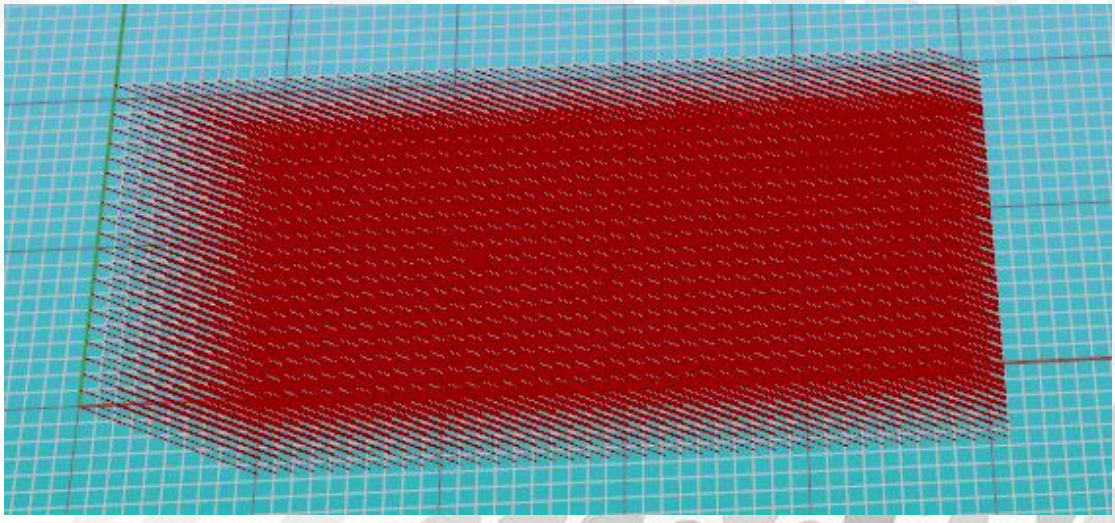
Spin Force:生成一个带旋转力的电荷场
输入端 P:电荷生成的开始点
输入端 S:旋转力度
输入端 R:旋转的半径值
输入端 D:电荷衰减值
输入端 B:生成电荷的区域



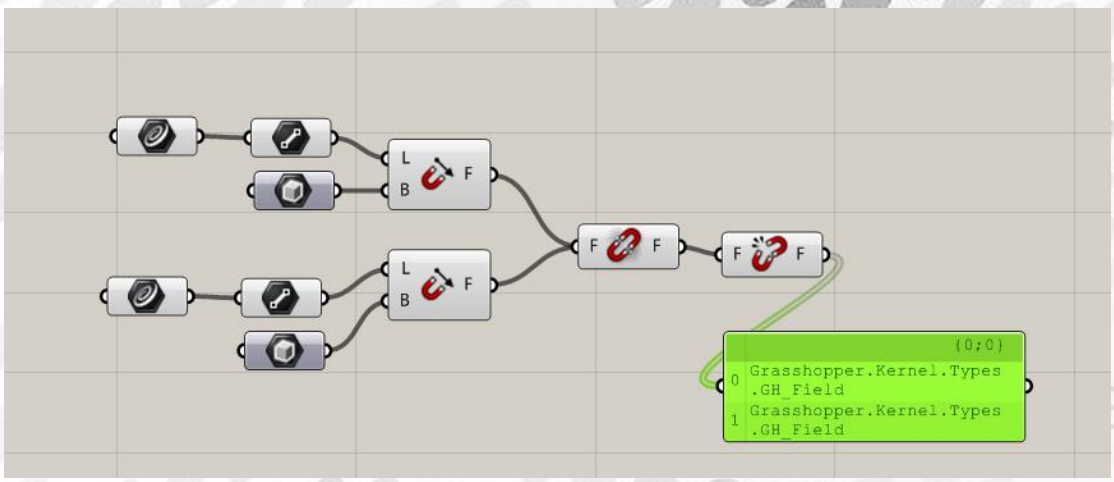


Vector Force:产生一个带直线力的电荷场
输入端 L:产生电荷场的线段
输入端 B:产生电荷场的区域

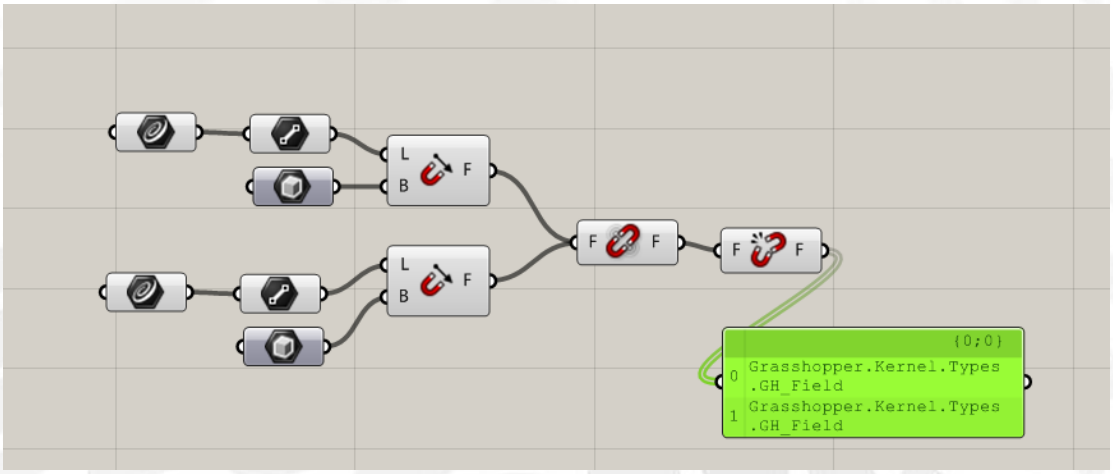




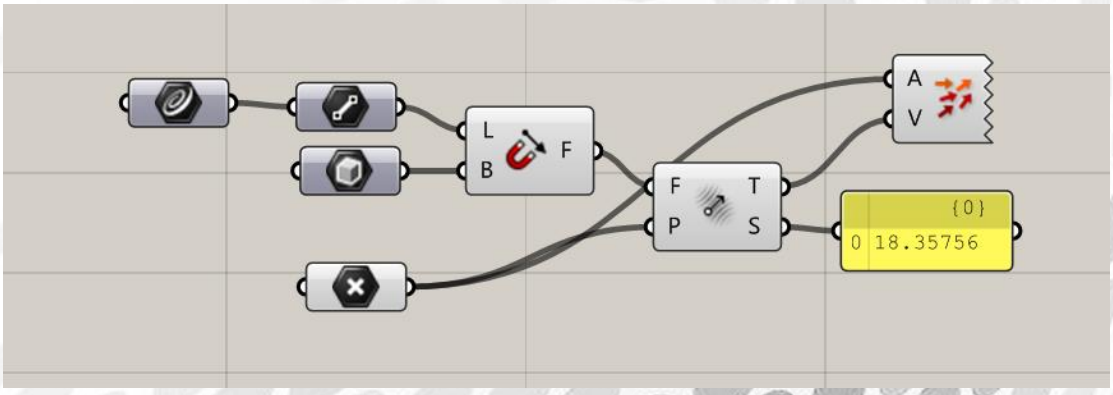
Break Field:电荷场分解
输入端 F:输入多组电荷场

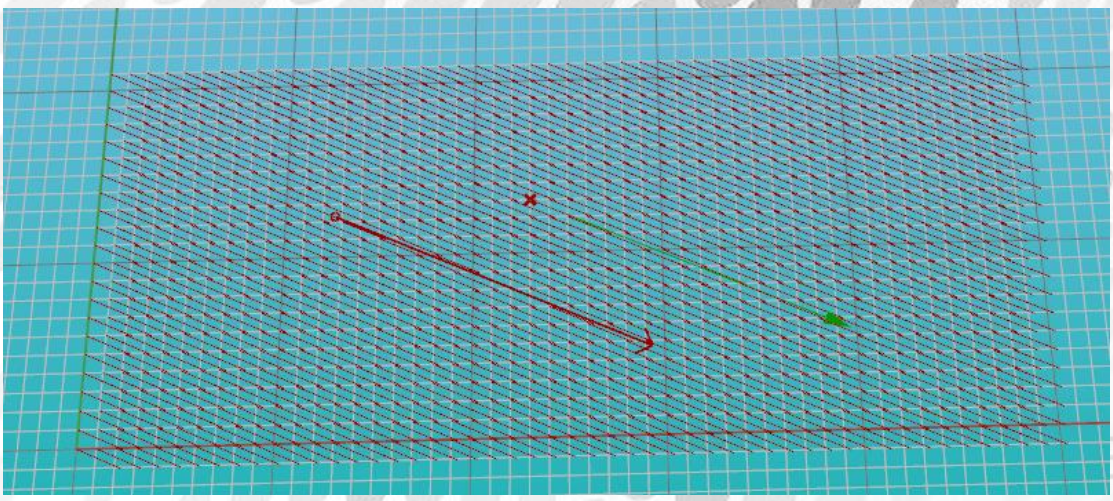


Merge Field:合并电荷场
输入端 F:输入要合并的电荷场

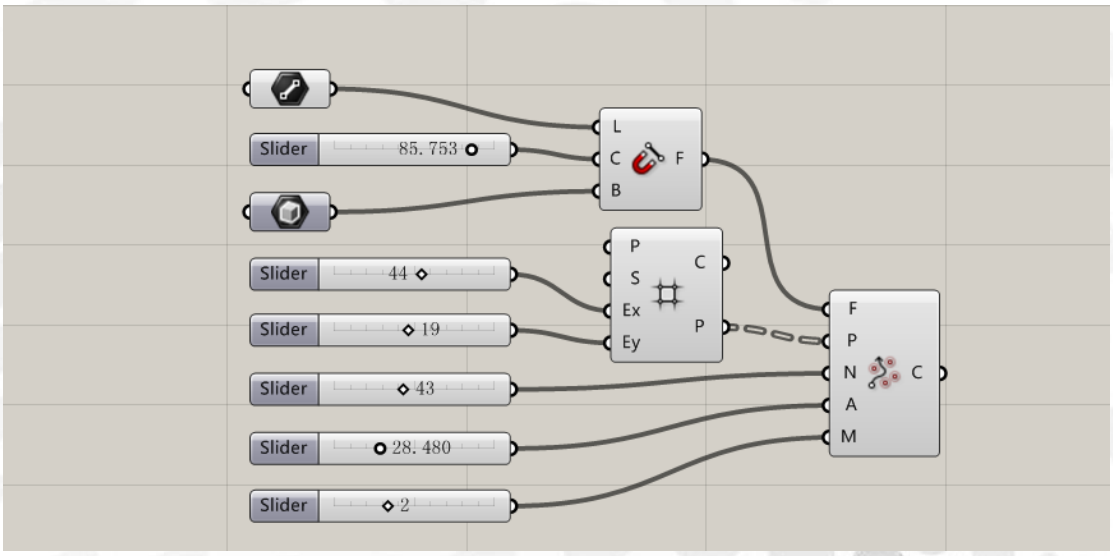


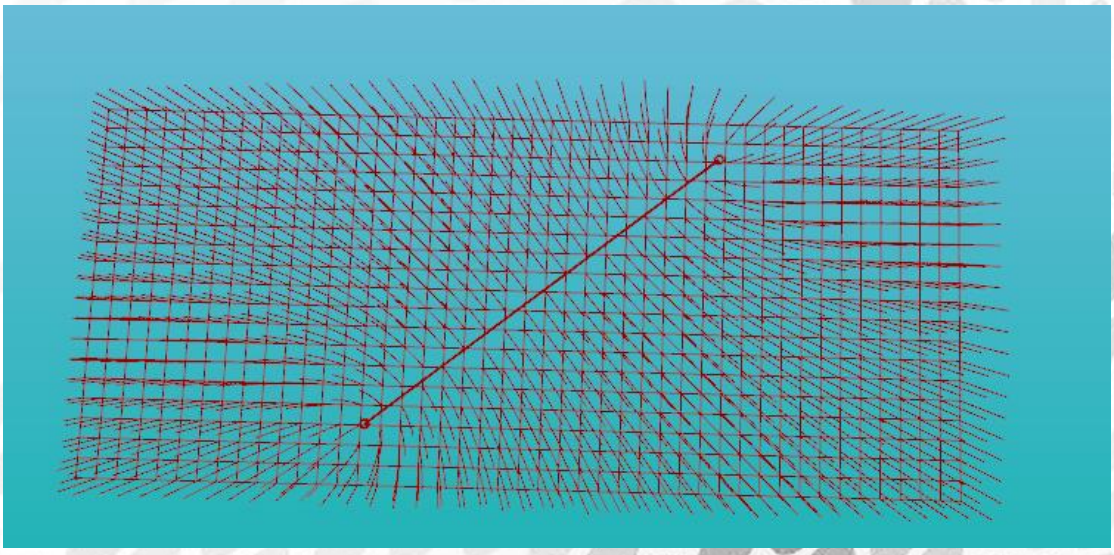
Evaluates Field:计算电荷场内某点的强度和方向
输入端:F 输入电荷场
输入端 P::输入电荷场内某点
输出端 T：点向量
输出端 S：点的电荷强度



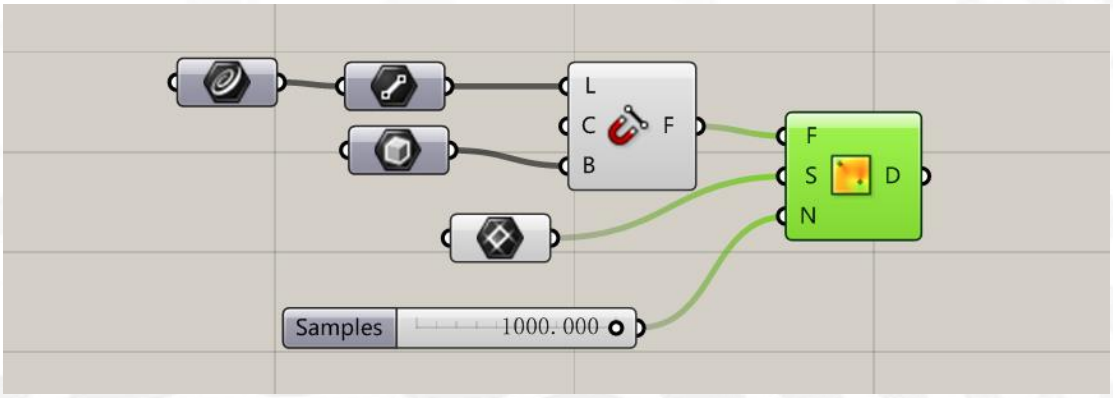


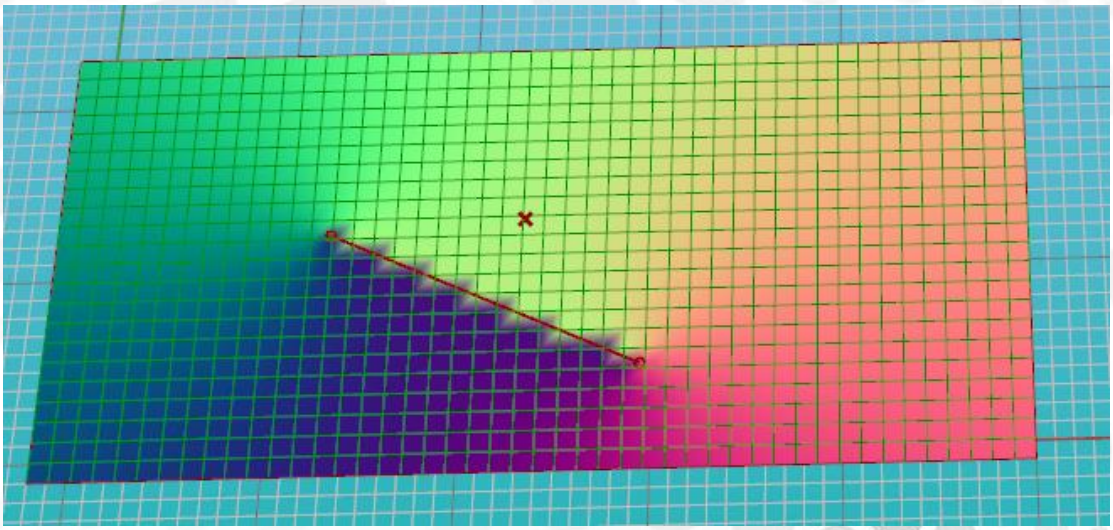
Field Line:产生电荷场线
输入端 F:输入电荷场
输入端 P:在电荷场内建立点阵
输入端 N：线的延生长度，值越大，线越长
输入端 A：精度提示，也可以控制线的长短
输入端 M：方法可取的值为 1，2， 3， 4.



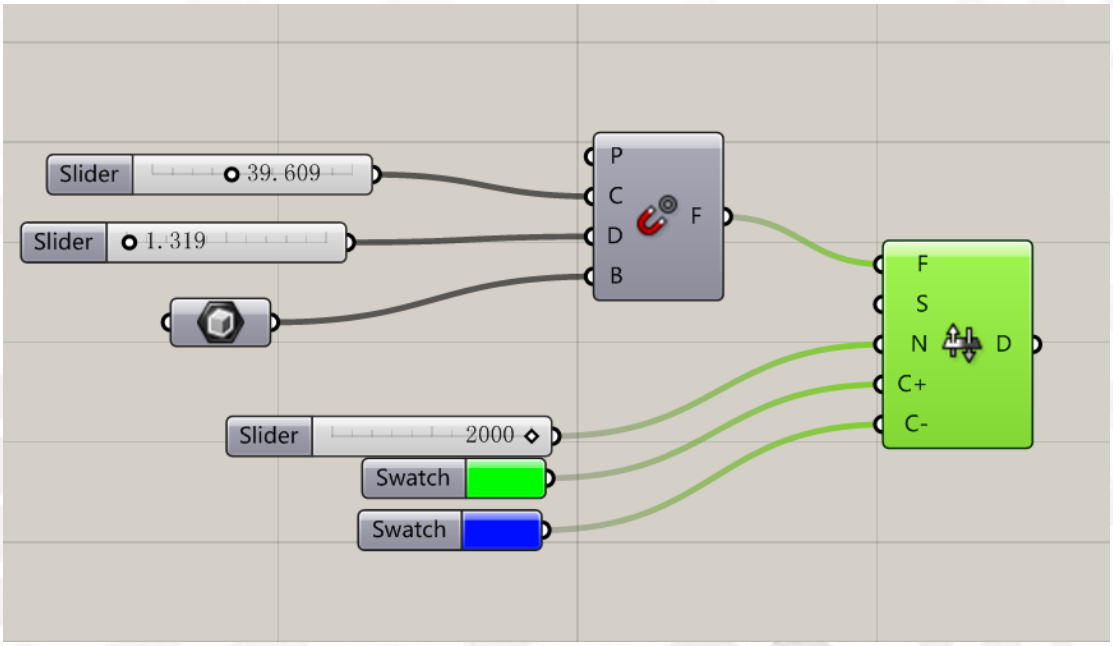


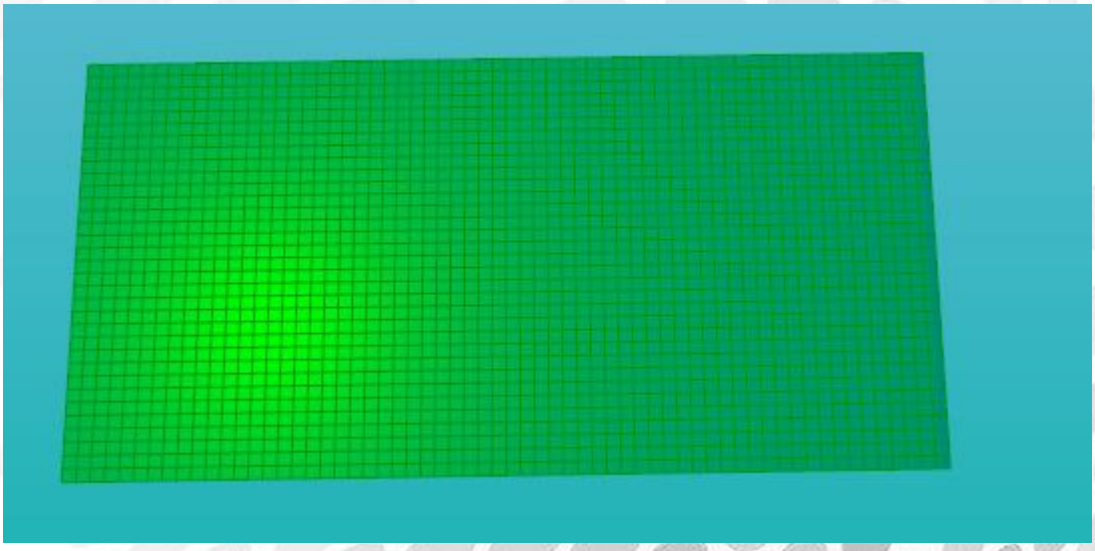
Direction Display ： 用颜色来表示场的方向
输入端 F：输入电荷场
输入端 S：区域值
输入端 N：显示精度



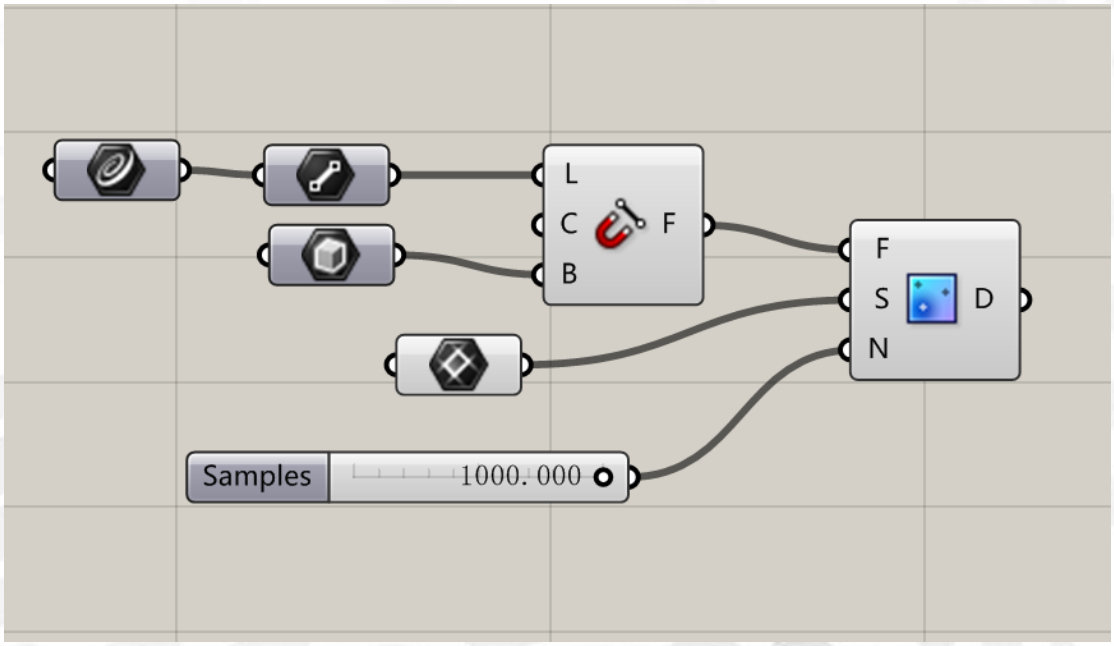


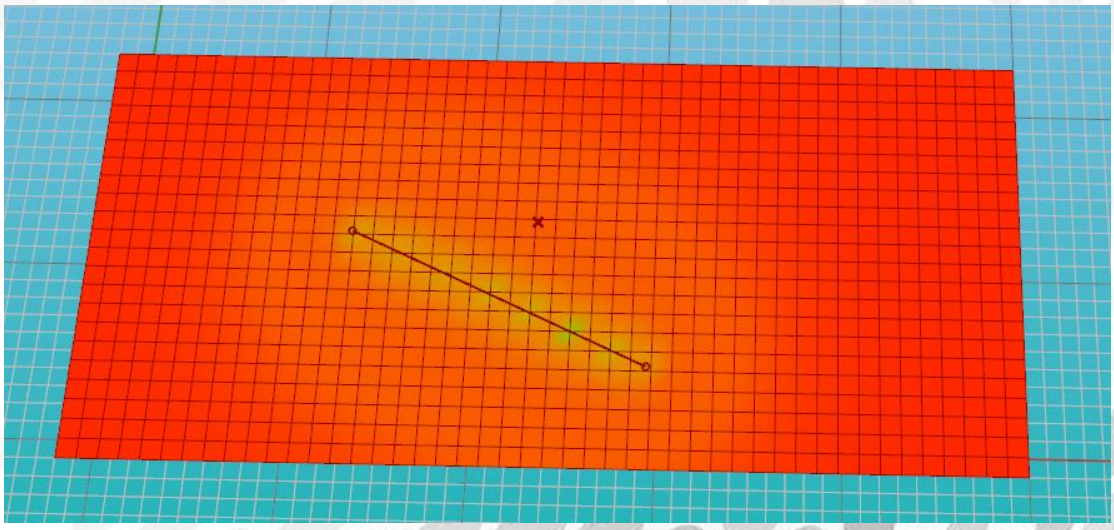
Perpendicular Display:用颜色来表示电荷场与指定面的垂直
输入端 F:输入电荷场
输入端 S:区域值
输入端 N：显示精度
输入端 C+:颜色阈值上限
输入端 C-:颜色阈值下限



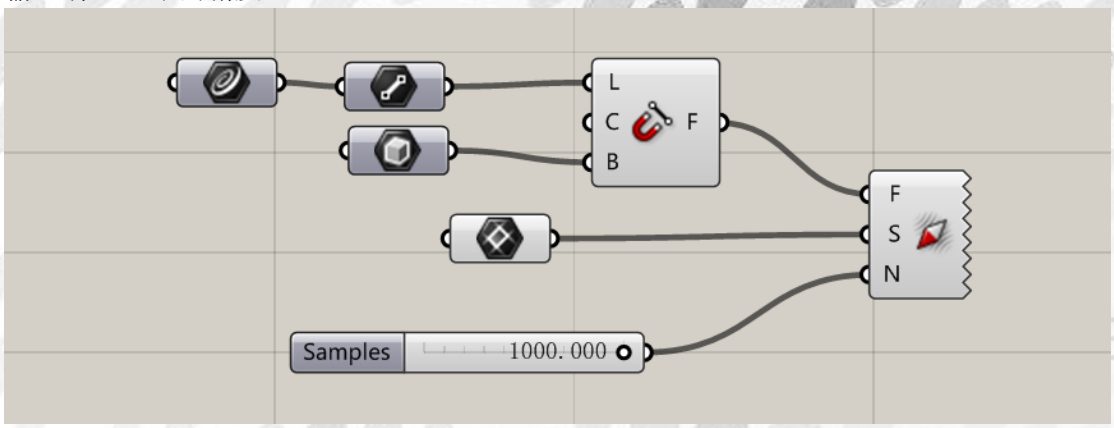


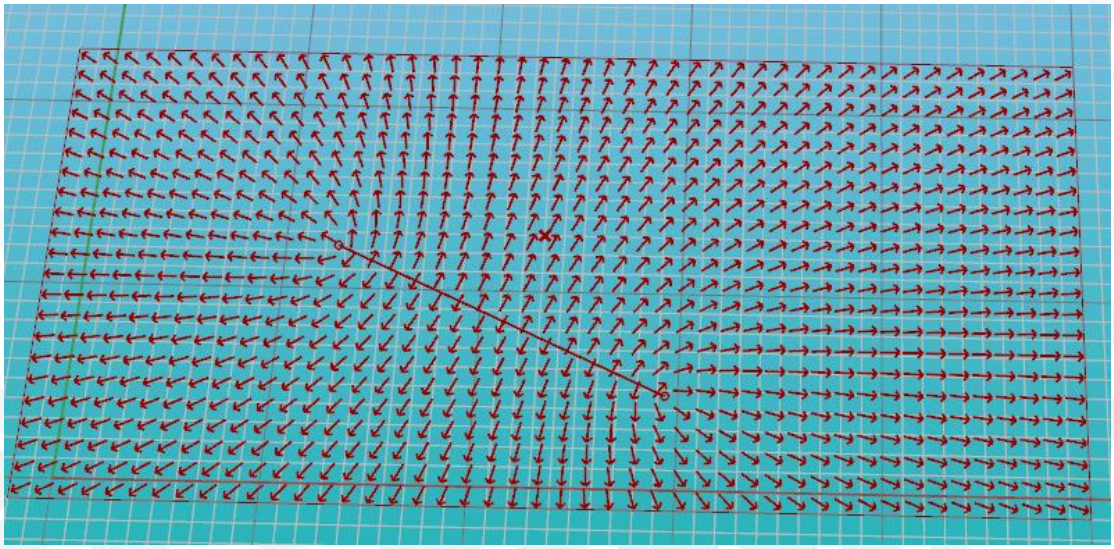
Scalar Display:用明暗来表示场的强弱
输入端 F:输入电荷场
输入端 S:输入需要显示的区域
输入端 N:显示精度



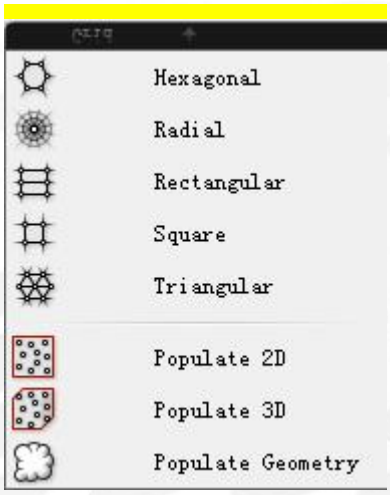


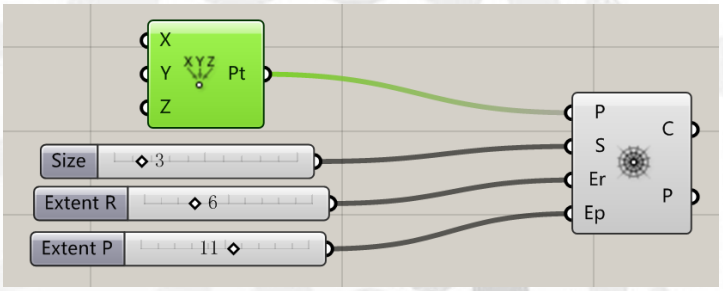
Tensor Display：以向量来显示电荷场
输入端 F：输入电荷场
输入端 S:需要显示的区域
输入端 N:显示的精度



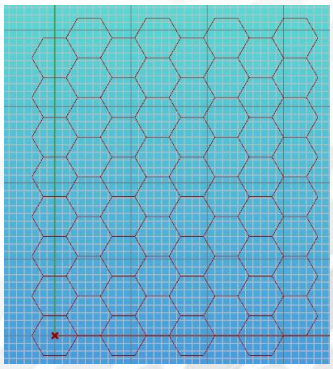


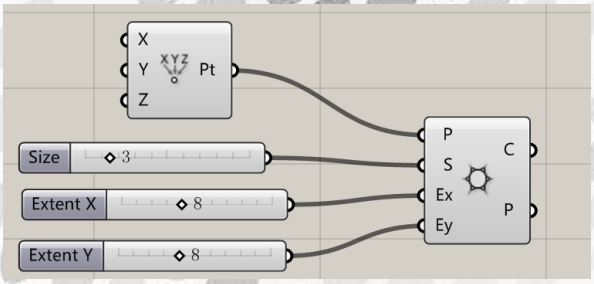
（2）Grid 电池序列



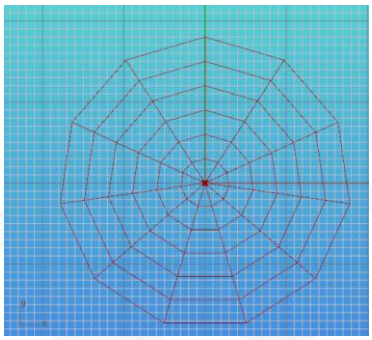


Hexagonal:六边形阵列
输入端 P：输入平面坐标
输入端 S：六边形半径
输入端 Ex：X 轴方向数量
输入端 Ey :Y 轴方向数量
输出端 C: 六边形单体
输出端 P：六边形中心点

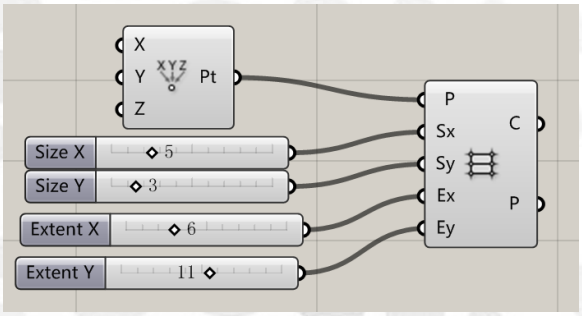


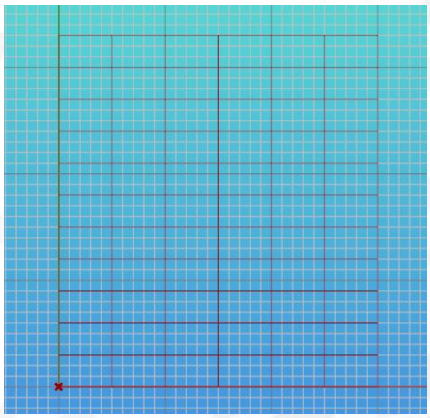


Radial：多边形阵列
输入端 P：坐标平面
输入端 S：多边形的半径
输入端 Er：多边形个数
输入端 Ep：多边形边数
输出端 C: 六边形单体
输出端 P：六边形中心点

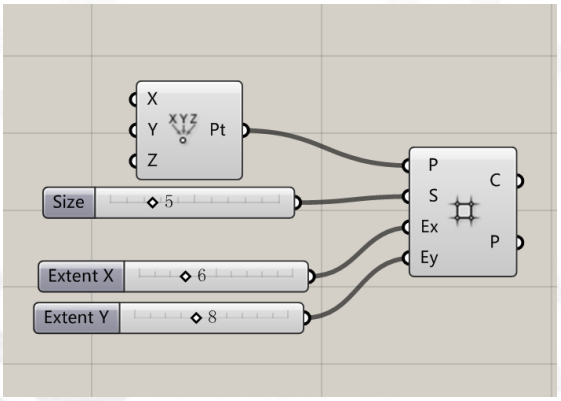


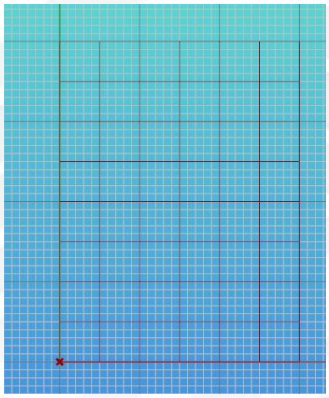
Rectangular：矩形阵列
输入端 P:坐标平面
输入端 Sx：矩形宽度
输入端 Sy：矩形长度
输入端 Ex：x 轴方向矩形个数
输入端 Ey：y 轴方向矩形个数
输出端 C: 矩形单体
输出端 P：矩形形顶点



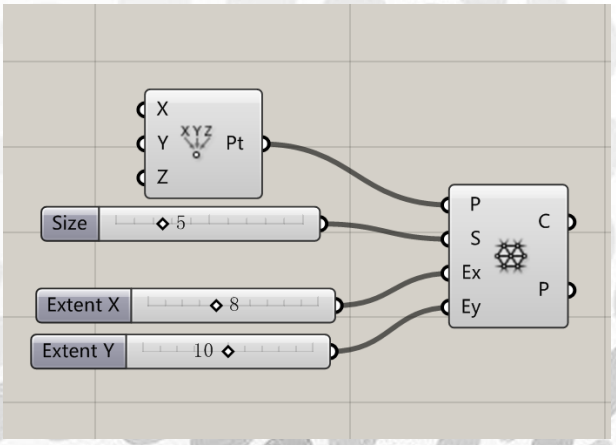


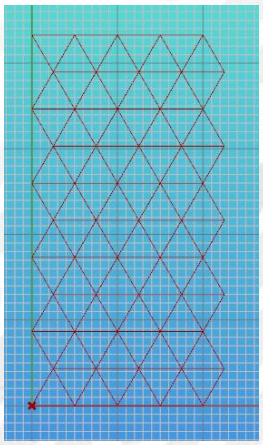
Square：方形阵列
输入端 P:坐标平面
输入端 S：正方形长宽
输入端 Ex：x 轴方向数量
输入端 Ey：y 周方向数量
输出端 C: 方形单体
输出端 P：方形顶点



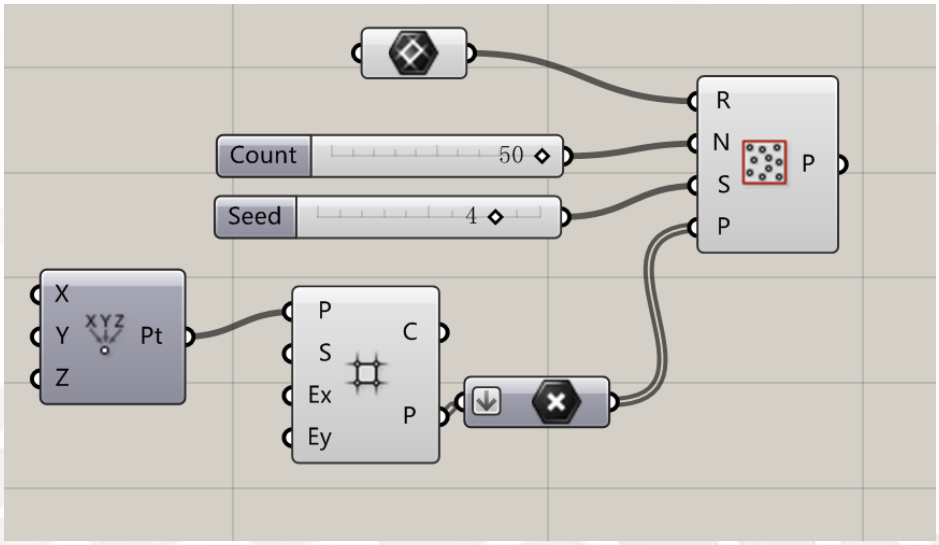


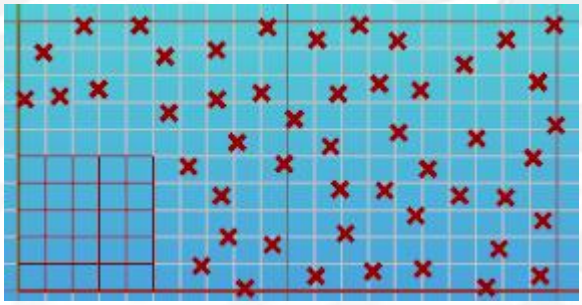
Triangular：菱形阵列
输入端 P：坐标平面
输入端 S：菱形边长
输入端 Ex：x 轴方向数量
输入端 Ey：y 轴方向数量
输出端 C: 菱形单体
输出端 P：菱形顶点



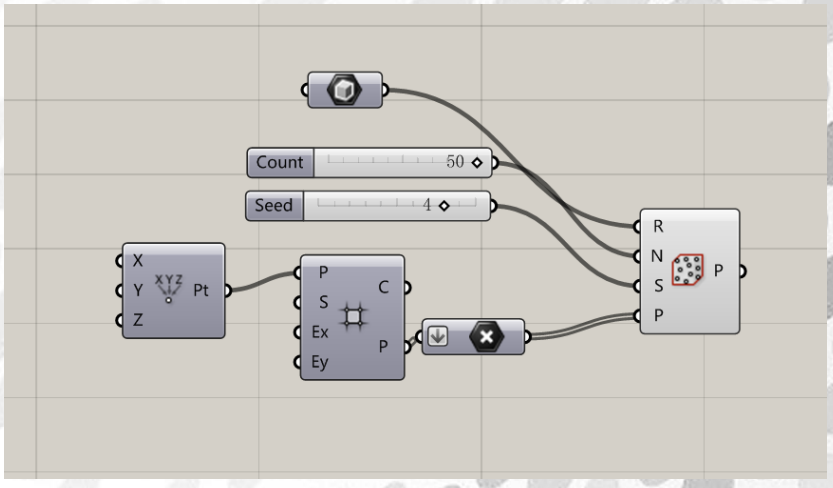


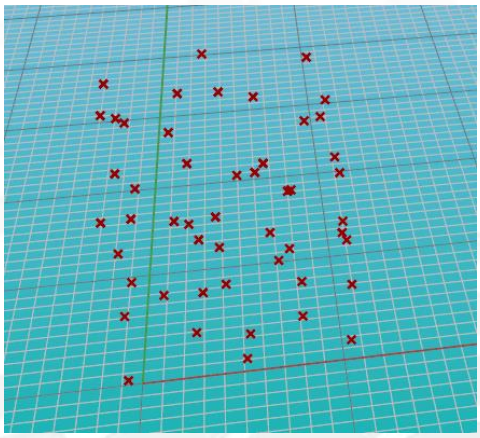
Populate 2D：随机生成二维点
输入端 R：随机生成区域(矩形)
输入端 N：随机点数量
输入端 S：随机种子
输入端 P：干扰点，使得随机点生成范围不在其内



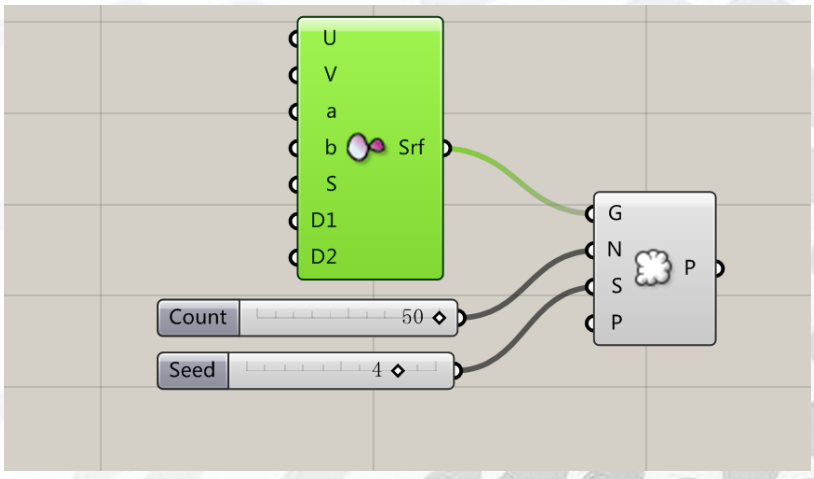


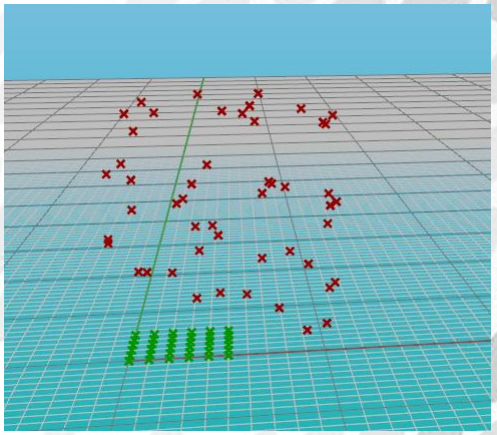
Populate 3D：随机生成三维点
输入端 R：随机生成区域（box）
输入端 N：随机点数量
输入端 S：随机种子
输入端 P：干扰点，使得随机点生成范围不在其内





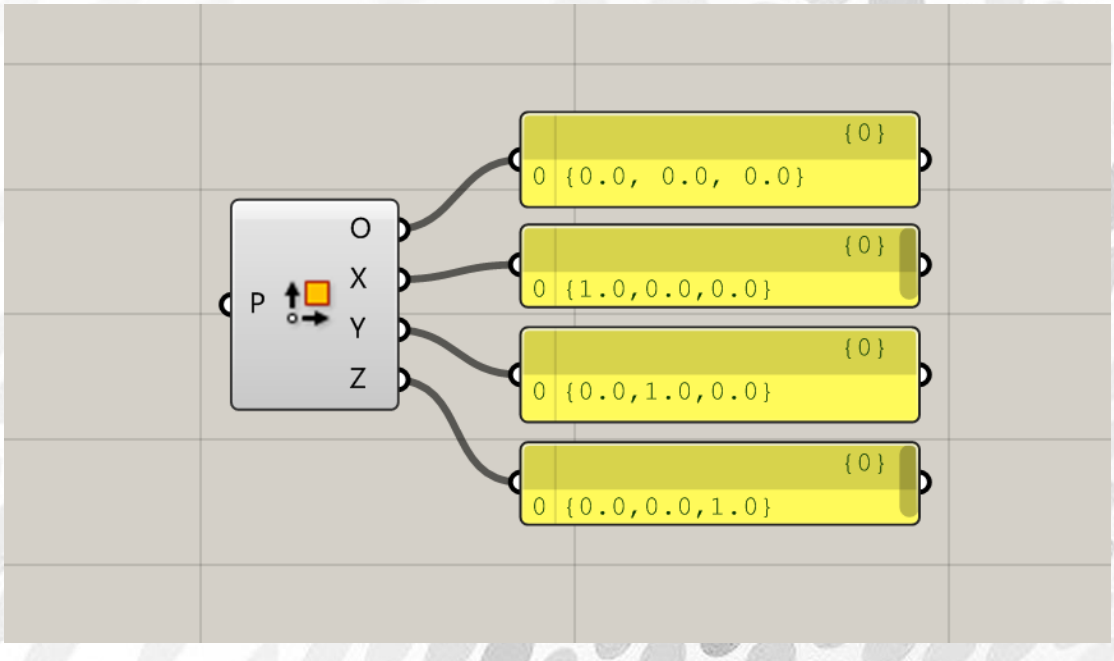
Populate Geometry：在物体表面成随机点
输入端 G：拾取几何体
输入端 N：随机点数量
输入端 S：随机种子
输入端 P：干扰点，使得随机点生成范围不在其内



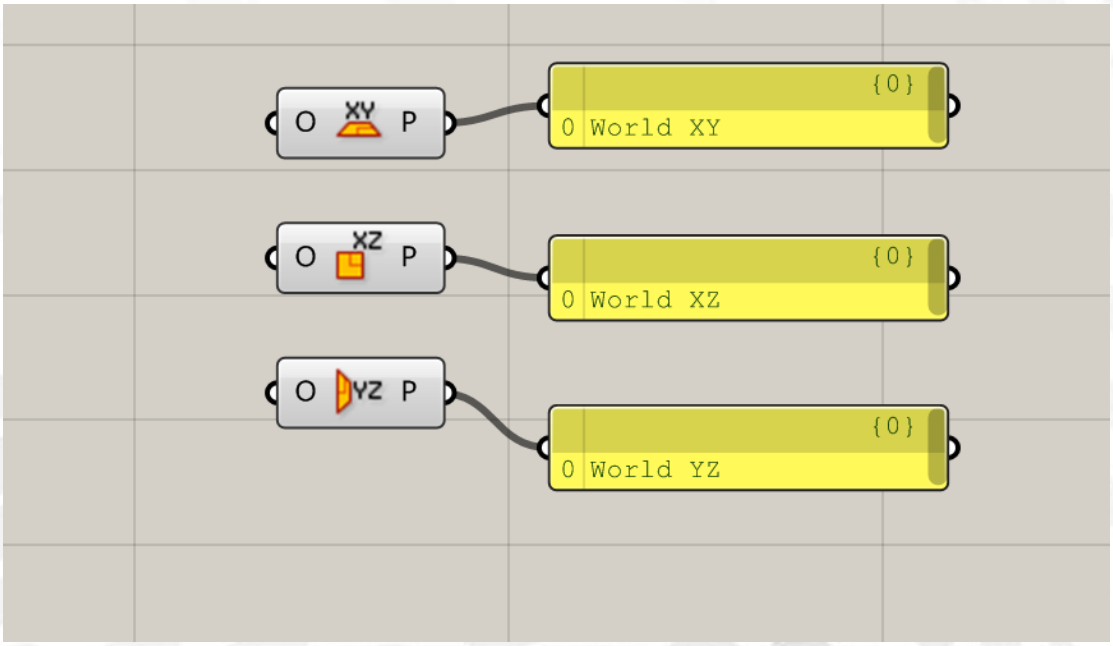


（3）Plane 电池序列

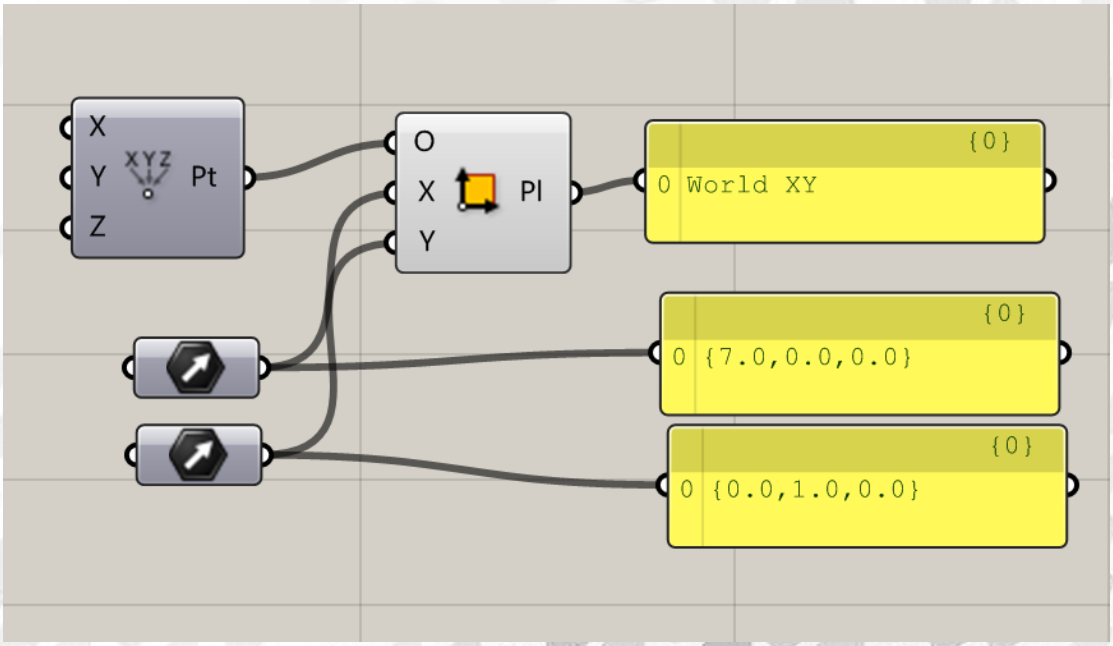
Deconstruct Plane：描述一个平面
输出端 O：坐标原点
输出端 X：X 轴单位向量坐标
输出端 Y：Y 轴单位向量坐标
输出端 Z：Z 轴单位向量坐标



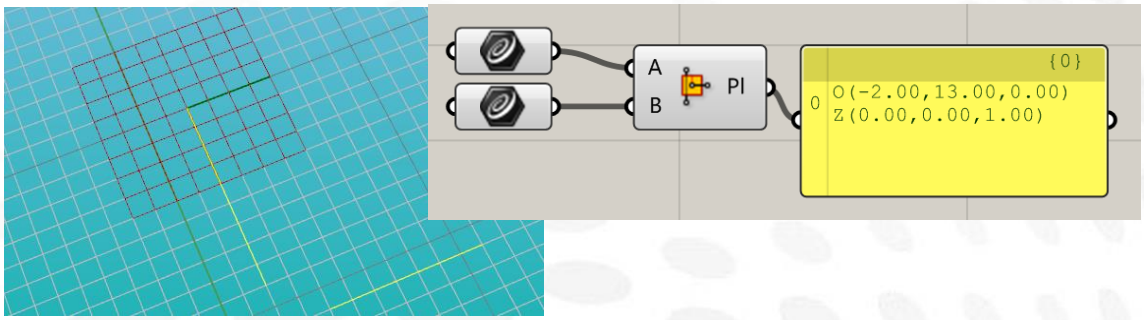
XY XZ YZ Plane：分别建立 xy xz yz 平面
输入端 O:输入坐标原点
输出端 P：输出一个工作平面，分别是 xy 平面，xz 平面和 yz 平面



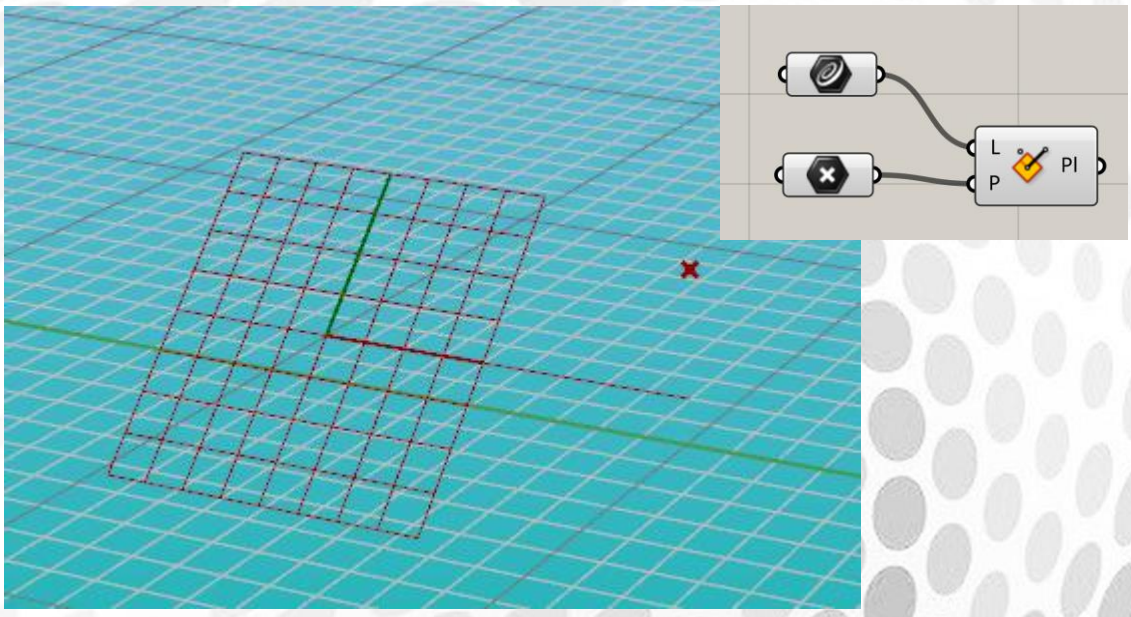
Construct Plane：通过定义 xy 轴向量值来生成平面
输入端 O:坐标平面原点
输入端 X:X 轴向量
输入端 Y:Y 轴向量



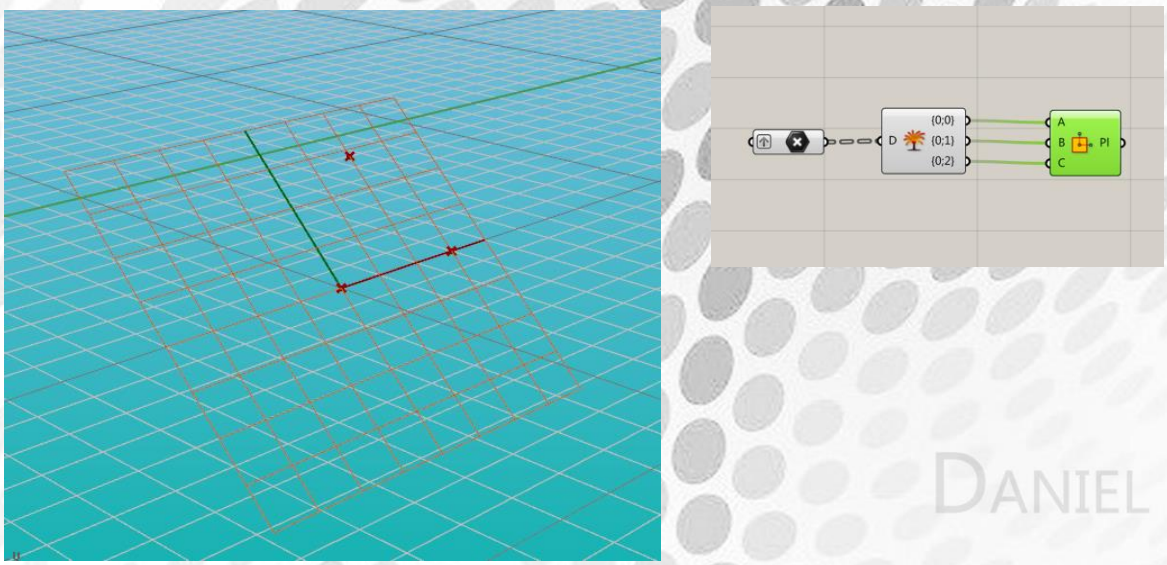
Line+line：通过两条线条定义一个平面
输入端 A B:输入 A B 两条线段
平面原点坐标是优先识别 A 线段的起始端点作为原点



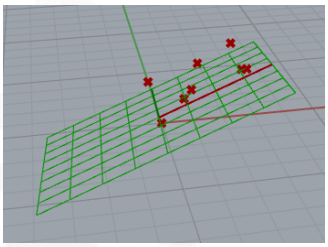
Line +Pt：通过一条线段和点来定义一个平面
输入端 L:输入线段
输入端 P：输入点



Plane 3Pt：通过三个点定义平面
输入端 ABC：分别输入三个点



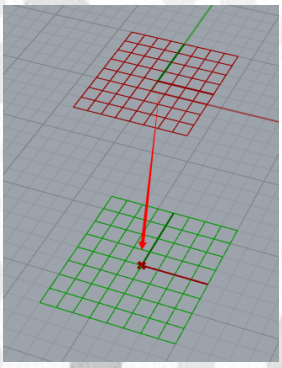
Plane Fit：通过一系列点找到最适合平面
输入端 P：点群
输出端 Pl：平面
输出端 dx：点与平面最大距离



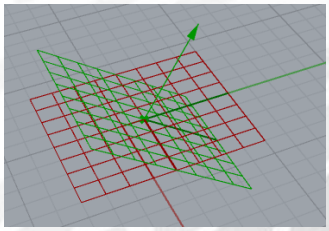
Plane Normal：创建一个垂直于某向量的平面
输入端 O：向量起始点
输入端 Z：向量
要注意，该平面 P 永远垂直于向量 Z

Plane Offset：平面偏移
输入端 P：平面
输入端 O：偏移距离
要注意的是，和 Rhino 里的 offset 不同，这里的平面是无限大的，因此不存在便宜缩放。
所以在显示的时候还是会显示一样的平面缩略图，而不像 Rhino 中 offset 以后 curve 会缩放。

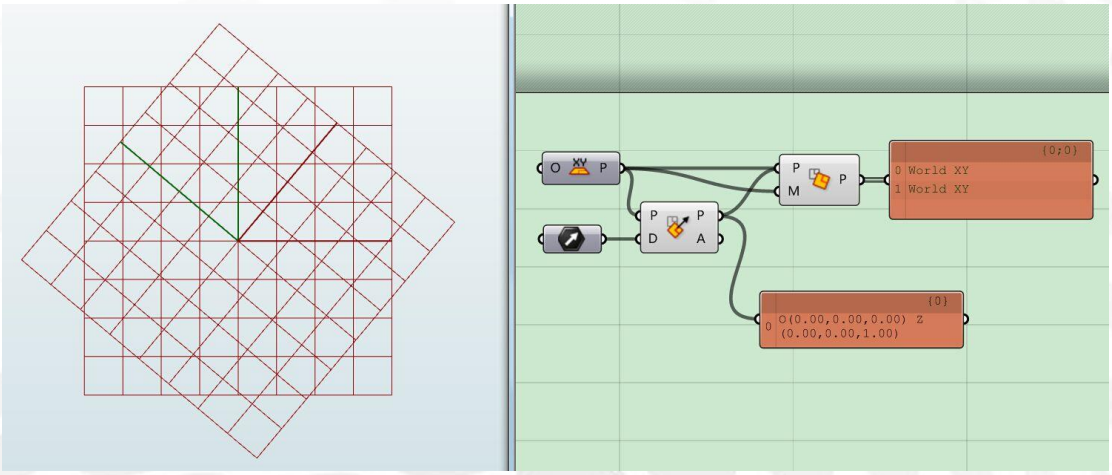
Plane Origin：改变一个平面（缩略图）的中心点（虽然翻译应该为起始点），从而平面（缩略图）的位置随之发生更改。



Adjust Plane：调整平面。
输入端 P：原始平面
输入端 N：调整后垂直的向量



Align Plane：对齐平面，在一个已知向量引导下，通过向量调整平面指向性
输入端 P：平面
输入端 D：向量
Align Planes：通过另一个平面调整平面指向性



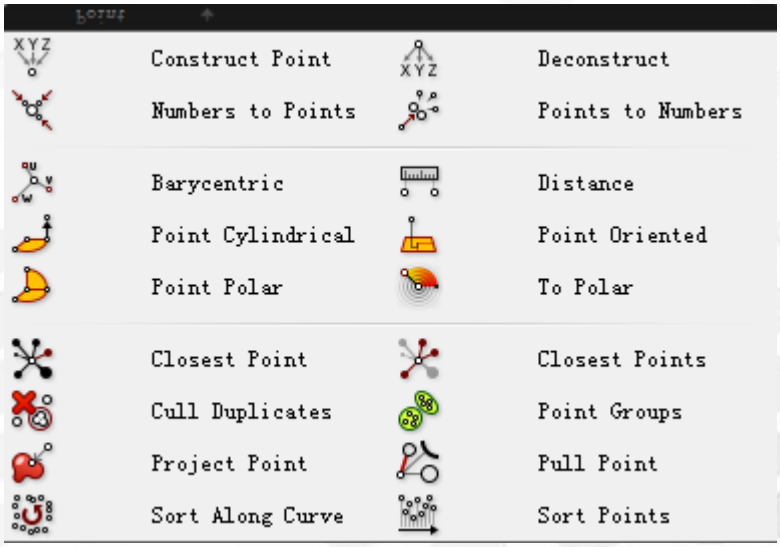
图片由 Chester.L 提供

Plane Closest Point：距离平面最近投影点
输入端 S：样点
输入端 P：平面
输出端 P：S 点在 P 面上的投影点
输出端 uv：点的 uv 值
输出端 D：S 点离 P 平面的距离

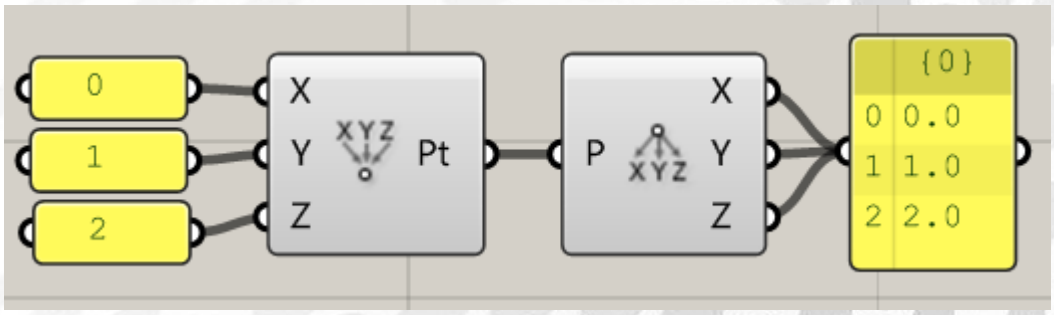
Plane Coordinates：输入点得到其在某平面坐标系中的坐标

Rotate Plane：旋转平面
输入端 P：平面
输入端 A：旋转角度

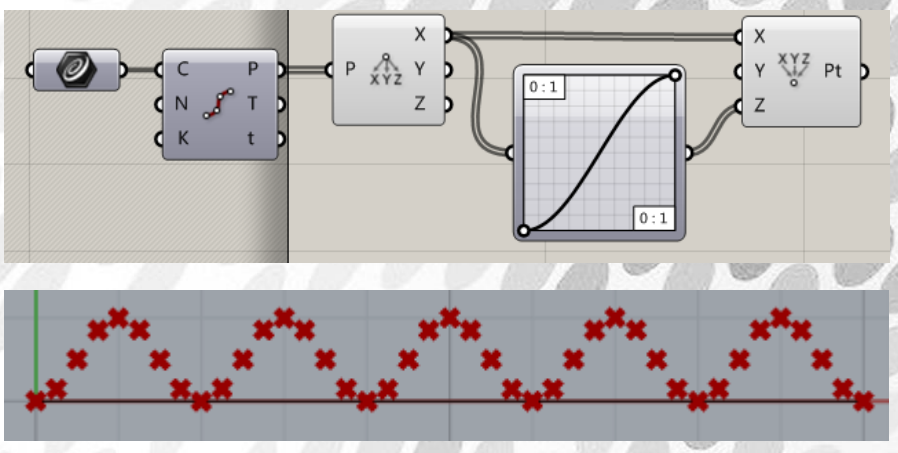
（4）Point 电池序列



Construct Point：通过 x,y,z 坐标建立一个点
Deconstruct Point：分解一个点得到 x,y,z 坐标



事实上，这两个运算器经常反着来串联。



Numbers to Points:输入三个数字，按照 M 中的排序成为三坐标构成一个点
Points to Numbers:输入一个点，按照 M 中的排序分解成一个点的三坐标

Barycentric：重心点
输入端 ABC：三个锚点
输入端 UVW：三个重心坐标

Distance：测量 AB 两点间距离

Point Cylindrical：通过一个圆柱坐标创建点
输入端 P：基准平面
输入端 A：角度
输入端 R：半径
输入端 E：升高

Point Oriented:通过一个平面的 UVW 坐标建立点

Point Polar：通过极坐标建立点

To Polar：将一个 3D 坐标系中的点转换到极坐标上

Closest Point：在一群点中找到某一点的最近点
输入端 P：要找最近点的点
输入端 C：点群
输出端 P：最近点
输出端 i：最近点的序号
输出端 D：两点间的距离

Closest Points：和上边的运算器类似，只不过是找出最近的几个点，并按照距离从小到大输出这些点。

Cull Duplicates:剔除距离近的点
输入端 P：点群
输入端 T：距离容忍度
输出端 P：剔除后的剩余点

Point Groups：将距离近的点自动化成一组

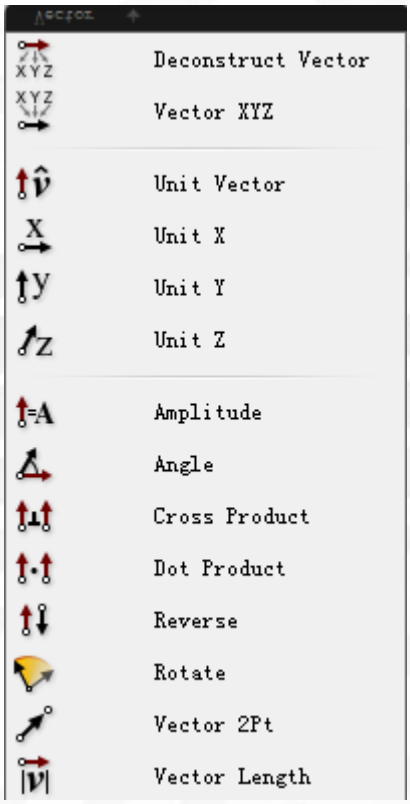
Project Point：投影点
输入端 P：点
输入端 D：方向
输入端 G：被投影至的物体
输出端 P：投影的点
输出端 I：投影指数。0 代表投影上，-1 代表未投影

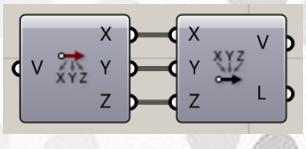
Pull Point：将点拉至某几何形体
输入端 P：点
输入端 G：需要拉至的几何形体
输出端 P：几何形体上的最近点
输出端 D:距离

Sort Along Curve：延曲线将点分类，比如画一条逆时针的曲线，点将会按照逆时针顺序排列。
输入端 P：点群
输入点 C：曲线
输出端 P：排列后的点
输出端 I：点的序号
对这个运算器，想要详细了解，欢迎登陆 csh.eeeetop.com 观看由我发布的【By DanielJin】Voronoi 全局排序思考题中的答案示例。

Sort Points：对点群线性排序

（5）Vector 电池序列





Deconstruct Vector:分解一个向量，得到 xyz 三个坐标轴上的投影长度（数字）
Vector XYZ:通过 xyz 三个坐标上的投影长度（数字）合成一个向量

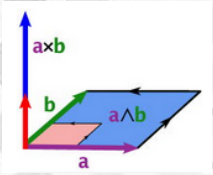
Unit Vector:单位向量，将一个向量转化为同方向长度为 1 的单位向量 

以下三个向量是 xyz 三个方向的单位向量，输入端 F 可以决定向量长度，默认为 1
Unit X
Unit Y
Unit Z

Amplitude：官方起的名字叫振幅，实际上是通过向量长度和方向创建一个向量
输入端 V：基础向量，可以理解为方向
输入端 A：振幅，可以理解为长度

Angle：求两输入向量 A 和 B 之间的角度

Cross Product：求两向量的向量积
Dot Product：求两向量的标量积



Reverse：反转向量

Rotate：将一个向量 V 沿一个轴 X 旋转 A 的角度（或弧度）

Vector 2Pt：通过两点 A 和 B 建立向量 AB

Vector Length：向量长度

5. Curve 电池组

（1）Analysis 电池序列

Control Point:得到一个 Curve 的控制点和节点

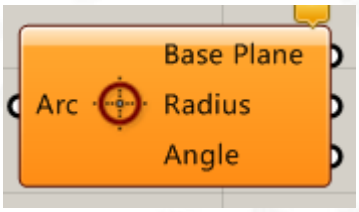
Control Polygon：提取出一个 nurbs 曲线的控制多边形

Deconstruct Arc：分解一个圆弧得到其相关参数。

注意！这个图标坑了无数人!!!无数人来问我为什么这个运算器得不到圆心了!!!

在 long long time 以前，这个图标对应的功能是找平面中心，比如圆心。

那个时候他长这个样子：



但是现在他不是这个样子了!!! 现在是这样子了：

对应的找圆心的图标要用这个了：

大家不要再拿着老版教程问，咦我看到别人用的计算器怎么和我不一样啊是不是我这 bug 了这种问题了！要崩溃了!!!

Deconstruct Rectangle：分解一个矩形得到相应参数。

End Points：求一段 Curve 的起始终止点。

Polygon Center:求一个 Polygon 的一些 Center。该运算器有三个输出中心点，中文翻译分别为：输出端 Cv，垂直中心点。输出端 Ce，由边平均得出的中心点。输出端 Ca，由面平均得出的中心点。在正 N 边形中这三点是一样的，不同的 polygon 会导致三点不同，读者可在课下自己尝试区别。

Closed：检验一段 curve 是否为周期、闭合曲线。输出端分别为闭合，周期。输出值为 True 则曲线闭合或呈周期性。反之则 false。

Curvature Graph：绘制标准犀牛曲率圆
输入端 C：要绘制的曲线
输入端 D：图表样点拾取密度
输入端 S：图表输出比例

Curve Closest Point：找到曲线上的最近点
输入端 P：输入点
输入端 C：输入曲线
输出端 P：曲线上距离样点最近的点
输出端 t：曲线定义域上最近点的值
输出端 D：样点和最近点的距离

Curve Nearest Object：距离曲线最近的物体
输入端 C：输入曲线
输入端 G：输入几何物体
输出端 A，B：在曲线，几何物体上的最近点
输出端 I：最近的几何物体的个数

Curve Proximity：两曲线最近点连线

输入端 A,B：两根曲线
输出端 A,B：曲线上的最近点
输出端 D：两点之间距离

Discontinuity：查找曲线上所有不连续点（即曲线方程的导数有变化）
输入端 C：输入曲线
输入端 L：判定的等级（要求输入整数，1=tangency 相切，,2=curvature 弯曲相吻合，大于 2 的整数则视为无限大）
输出端 P：不连续点
输出端 t：不连续点在曲线区间上的 t 值

Extremes：输入一个 Curve，得到曲线最高最低点。（即导数等于零）
输入端 C：输入 curve
输出端 H，L：最高最低点

Planar：检查一个 curve 是否在一个平面内。是则输出 True，输出平面。

Curvature：分析出曲线上某点的曲率
输入端 C：输入曲线
输入端 t：待分析的曲线定义域上的曲率参数
输出端 P：曲线上 T 值上的点
输出端 K：曲线上 T 值上的曲率向量
输出端 C：与曲线该点相切曲率等于该点曲率的圆

Curve Frame：求出曲线某 t 值点的切线框架平面
输入端 C：要输入的 curve
输入端 t：待分析曲线定义域上的 t 值
输出端 F：曲线上 T 值处点切线所在框架平面

Derivatives：求指定曲线某点的导数
输入端 C：要输入的 curve
输入端 t：待分析曲线定义域上的 t 值
输出端:：对应 t 值点的曲线导数

Evaluate Curve，Point on Curve:都可以理解为求曲线上一点
输入端 C：要输入的 curve
输入端 t：待分析曲线定义域上的 t 值

Horizontal Frame, Perp Frame:求曲线某点的水平面，与切线垂直的面
输入端 C：要输入的 curve
输入端 t：待分析曲线定义域上的 t 值

Torsion：测量曲线一点上扭力
输入端 C：要输入的 curve

输入端 t：待分析曲线定义域上的 t 值

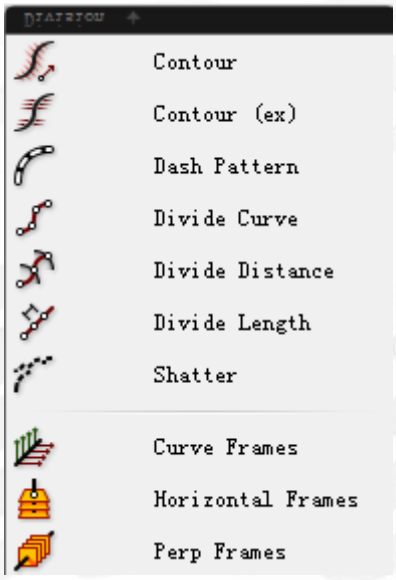
Evaluate Length：测量长度
输入端 C：要输入的 curve
输入端 L：待分析曲线定义域上的长度
输入端 N：若 N 为 true 则长度区间为 0.0-1.0
输出端 P：曲线上 L 长度的点
输出端 T：P 点对应切线的向量
输出端 t：P 点所对应的 U 值

Length：测量曲线长度
Length Domain：给定曲线上一个范围，测量长度
Length Parameter：给定曲线上一个分割点，测量两段分别的长度

Segment Length：查找曲线最长和最短的区段
输入端 C：输入 curve
输出端 SI：最短部分长度
输出端 SD：最短部分区间
输出端 LI：最长部分长度
输出端 LD：最长部分区间

Point in Curve（s）：检查点是否在封闭曲线内
输入端 P:要检查的点
输入端 C：输入的封闭曲线
输出端 R：0 在曲线上，1 在曲线内，2 在曲线外
输出端 P：点在曲线面域上的投影
输出端 I：指定一个包含该点的区域

（2）Division 电池序列



Contour:创建一组曲线轮廓
输入端 C：输入曲线
输入端 P：轮廓的起始点
输入端 N：轮廓的发现方向
输入端 D：等高线间的距离
输出端 C：生成的轮廓点
输出端 t：所有点的曲线参数

Contour(ex)：创建一组曲线轮廓
输入端 C：输入曲线
输入端 P：轮廓的基础平面
输入端 O：基面的轮廓偏移量
输入端 D：等高线之间的距离
输出端 C：生成的轮廓点
输出端 t：所有点的曲线参数

Dash Pattern：把曲线转换成虚线
输入端 C：输入曲线
输入端 Pt：生成虚线的空间值
输出端 D：输出的虚线
输出端 G：虚线空隙中的曲线

Divide Curve：将曲线 N 等分，比较简单。需要说明的是，K 为 true 时将会在节点处拆开曲线。
输入端 N：分割的份数
输出端 T：分割点处的曲线切线向量

输出端 t：分割点在曲线区间上的值

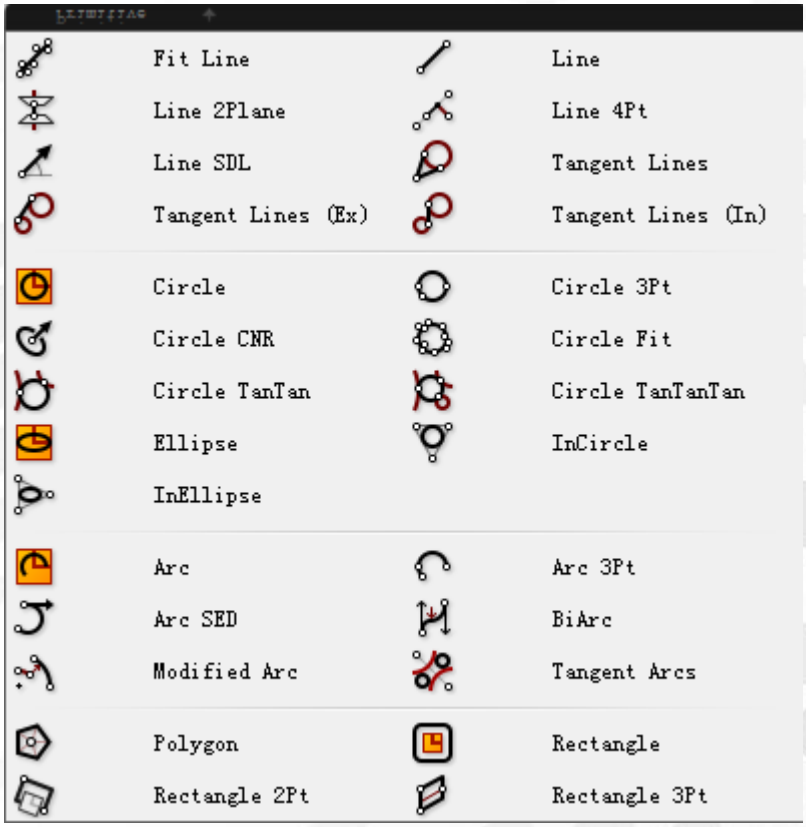
Divide Distance：用已知距离分割曲线，与上一个运算器不同的是输入端 D 为已知距离。
Divide Length：将曲线分割成某个长度的曲线段

Shatter：打碎曲线
输入端 C：输入曲线
输入端 t：要裁剪处的系数，即在曲线定义域上的值
输出端 S：输出的曲线碎片

Curve Frames：将曲线等分成多个曲线方向的平面
输入端 C：输入曲线
输入端 N：等分的数量
输出端 F：输出的 Frame
输出端 t：等分点在曲线上的长度系数

Horizontal Frames：将曲线按照长度等分并生成一系列水平方向的平面
Perp Frames：将曲线按照长度等分并生成一系列垂直方向的平面
输入端 C：输入曲线
输入端 N：等分数量
输入端 A：True 表示垂直平面不跟随曲线扭转
输出端 F：输出等分点上的平面
输出端 t:等分点在曲线上的长度系数

（3）Primitive 电池序列



Fit Line：通过一系列点，创造出一条符合的直线

Line：两点之间连线

Line 2Plane:求出一条穿透两个平面的线在两个平面之间的部分（就如同两把刀切黄瓜！）

Line 4Pt：求出两点之间连线在另两点之间连线上的投影
输入端 L：被投影线段
输入端 AB：待投影线段起始终止点
输出端 L：投影后的线段

Line SDL：已知起点（S）和长度（D）方向（L），建立一条线段

Tangent Lines：创建一个点和一个圆之间的两条切线
Tangent Lines（Ex）：创建两个圆外部的切线
Tangent Lines（In）：创建两个圆内部的切线

Circle：已知平面和半径，建立圆。圆心为输入端 P 的几何中心，默认为世界平面
Circle 3Pt：已知三点建立圆

Circle CNR：已知圆心，圆平面垂直向量和半径建立圆
输入端 C：圆心
输入端 N：垂直圆平面的向量
输入端 R：半径

Circle Fit：通过已知的一群点建立最适合的圆

Circle TanTan：通过一个已知点，创立一个和两条已知曲线相切的圆
Circle TanTanTan：通过一个已知点，创立一个和三条已知曲线相切的圆

Ellipse：已知平面和长短半径，建立圆

In Circle：求三角形的内切圆
In Ellipse：求三角形的内切椭圆

Arc：已知点，半径和弧度范围得出圆弧。
输入端 P：已知点
输入端 R：圆弧的半径
输入端 A：弧度范围，需要输入 Domain 区间

Arc 3Pt：已知三点求圆弧
Arc SED：根据已知起始结束点，和起始点切线方向求圆弧

Bi Arc：通过起始点和结束点和两点处向量得到双圆弧
输入端 S，E：起始点和结束点
输入端 Ts，Te：圆弧起点处和终点处向量
输入端 R：两端圆弧的权重
输出端 A1，A2：第一段，第二段圆弧
输出端 B：求出的双圆弧

Modified Arc：根据一个已知弧线求得另一个弧线
输入端 A：已知弧线
输入端 R：新弧线半径
输入端 A：新弧线的范围，要求输入 domain

Tangent Arc：建立与同一平面上两个圆相切的圆弧
输入端 A，B：已知两个圆
输入端 R：相切圆半径
输出端 A，B：求得的两段圆弧

Polygon：可倒角的正多边形
输入端 P：建立的平面
输入端 R：多边形的半径
输入端 S：多边形的边数

输入端 Rf：倒角半径
输出端 L：边长

Rectangle：可倒角的矩形
输入端 P：指定平面
输入端 X，Y：矩形的长宽
输入端 R：倒角半径
输出端 L：周长

Rectangle 2Pt：根据两对角点创建可倒角矩形
输入端 P：指定平面
输入端 A，B：两个对角
输入端 R：倒角半径

Rectangle 3Pt：根据三个已知点创建矩形

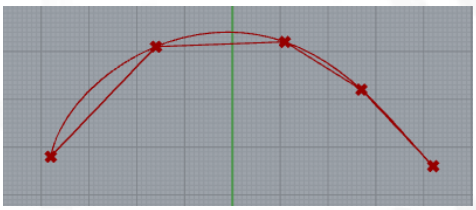
（4）Spline 电池序列

Bezier Span:由已知两点和起始向量创建曲线
输入端 A,B：起始两点
输入端 At，Bt：两点的起始向量
输出端 C：输出 curve
输出端 L：曲线长度
输出端 D：曲线范围

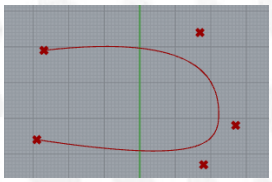
Interpolate：穿越点曲线
输入端 P：点
输入端 D：Degree，曲线的维度或称阶数。我对数学不太了解，不清楚详细原因但是 D 输入端只能输入奇数。如果输入 D 为奇数，将会生成 D+2 个控制点或 D+4 个（取决于 P），可以用 Control Point 运算器来验证。我曾简单理解为曲线圆滑程度，数值越大越圆滑。
输入端 P：是否呈周期性，True 则曲线会闭合，此时曲线没有起始点和终止点，故控制点将会变成 D+2 个。
输入端 K：0,1,2 代表三种曲线样式

Interpolate(t)：用一系列向量和向量起始点创建曲线

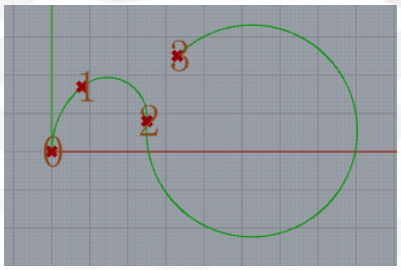
Kinky Curve：通过一系列点创建曲线。不同的是，中间的点为内插点，可以设置不同的角度，得到如图示曲线。

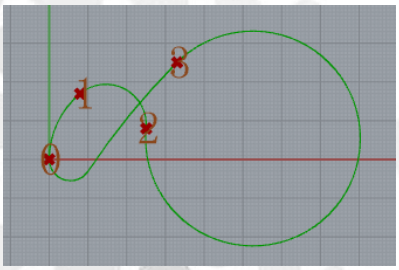


Nurbs Curve：创立一条 Nurbs 曲线。除了起始点和结束点外，中间的点为控制点。



Poly Arc：创建一条 Poly 圆弧，需要输入穿越点和起始向量。输入端 C 为是否闭合。





Poly Line：根据已知点创建一条折线

Tangent Curve：根据已知点和向量创建一条 Curve。需要注意的是输入点数量要等同于输入向量数量

Curve On Surface：通过在一个面上的一系列点创建一个 curve

Geodesic：通过在一个面上的起始和终止点创建曲线。
输入端 S：输入面
输入端 S：起始点
输入端 E：终止点
要注意的是，如果点不在面上，将会按照投影方式来投影到面上

Iso Curve：通过一个点，提取面上的 U 和 V 方向的结构线。Uv 即输入的点

Sub Curve：通过一个范围，截取曲线上的一部分。
输入端 C：基础曲线
输入端 D：截取范围

Tween Curve：通过两条已知曲线（A 和 B）和中间值（F），创建两条已知曲线的渐进曲线。

Knot Vector：建立一条 Nurbs 曲线节点处的向量
输入端 N：控制点数量
输入端 D：曲线阶数

输入端 P：是否闭合
输出端 K：Nurbs 曲线的节点矢量

Nurbs Curve：已知控制点及权重，建立 Nurbs 曲线
输入端 P:曲线控制点
输入端 W：控制点权重（与 P 数量要一致）
输入端 K：控制点向量
输出端 C：输出曲线
输出端 L：求得曲线长度
输出端 D：输出曲线范围

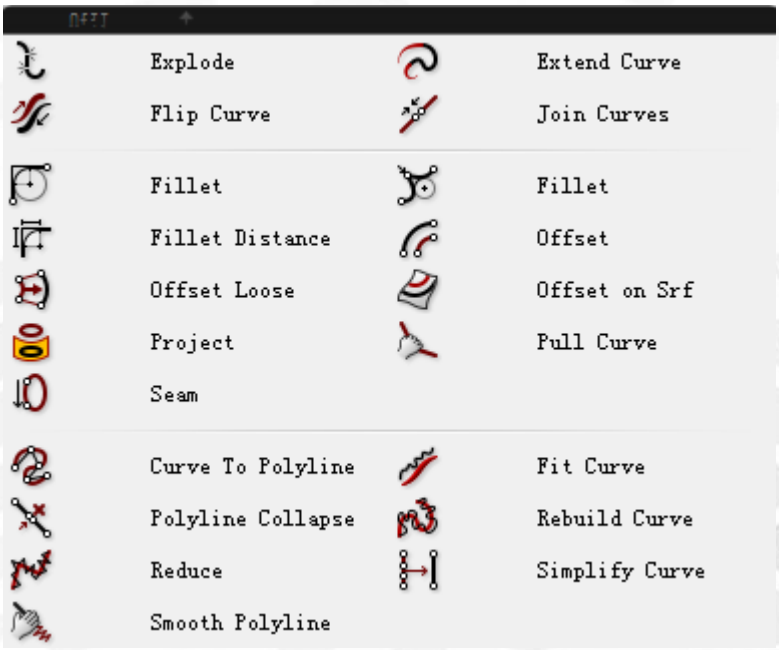
Blend Curve：类似 Rhino 中的命令，通过权重衔接两条曲线。
输入端 C：三种连接方式

Blend Curve Pt：通过穿过一个已知点衔接两条曲线

Catenary：创建一个自重下垂线
输入端 A,B：起始点和终止点
输入端 L：下垂线长度（要比 AB 两点的距离长）
输入端 G：重力强度

Connect Curves：连接曲线
输入端 C：曲线
输入端 G：连接方式
输入端 L：是否闭合
输入端 B：膨胀系数，请大家自行更改系数尝试形态的改变

（5）Util 电池序列



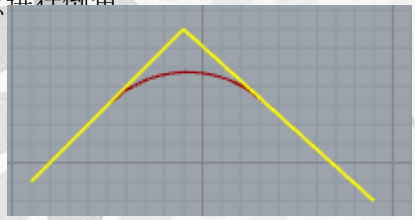
Explode:炸开曲线
输入端 C：曲线
输入端 R：True 表示循环分解直到成为最小的曲线
输出端 S：炸开后的碎片
输出端 V：炸开后曲线的顶点

Extend Curve：延伸曲线
输入端 T：起始点处延伸的长度（0=位置，1=相切，2=长度）
输入端 L0,L1：起始点和终止点延伸的长度

Flip Curve：用一个参考曲线来反转曲线
输入端 G：参考曲线
输出端 F：若曲线被反转则输出 True

Join Curves:将曲线合并成一条，类似犀牛 Join
输入端 P：如果为 True 表示保留原来的方向

Fillet：将曲线尖锐地方倒角。输入端 R 为半径，类似 Rhino 和 CAD 的 Fillet
另一个 Fillet：根据已知曲线的系数 t 对曲线进行倒角
Fillet Distance：根据输入的距离 D 对曲线不平滑折点进行倒角



需要说明的是，折线也属于曲线 Curve

Offset：将已知曲线偏移指定距离
输入端 C：输入曲线
输入端 D：偏移距离
输入端 P：受偏移的平面
输入端 C：倒角圆类型（0=不倒角圆，1=锐角，2=圆角，3=平滑，4=切角）

Offset Loose：将已知曲线的控制点偏移指定距离
输入端 C：输入曲线
输入端 D：偏移距离
输入端 P：受偏移的平面

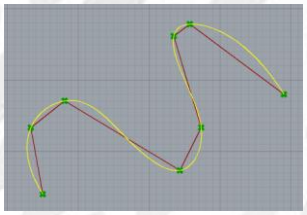
Offset on Surface：将已知面上的曲线偏移指定距离
输入端 C：输入曲线
输入端 D：偏移距离
输入端 S：输入平面

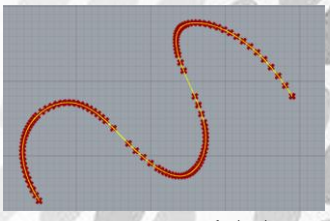
Project：投影
输入端 C：被投射的曲线
输入端 B：被投至的某 Brep 面
输入端 D：投影方向

Pull Curve：拉回一条曲线至一个面上，但是我至今没有搞明白和 Project 的区别

Seam：调整一条闭合曲线的接缝
输入端 C：闭合曲线
输入端 t：新的接缝参数

Curve to Polyline：将一条曲线变为折线
输入端 C：输入曲线
输入端 Td：误差距离容忍度
可以理解为细分程度，值越小折线越接近曲线，当输入值为 0 时原曲线有 a 个控制点（用 Control Point 运算器得到，而非犀牛里打开的控制点），就将会产生相应 a-1 段线段组成的折线。但是当输入值过大时，折线最不吻合程度不会超过 Td=0 的情况


Td=0 以及 Td=100


Td=0.01 几乎吻合


Td=1.8 比 Td=0 吻合

输入端 Ta：误差角度容忍度，和上边图示 Td 类似
输入端 E-，E+：单个线段最小，最大边长。

Fit Curve：调整曲线
输入端 D：曲线的阶数，默认为原阶数

输入端 Ft：调整容差，默认为 Rhino 当前文件设置的容差值

Polyline Collapse：将多折线坍塌
输入端 C:虽然显示输入曲线，但实际必须输入多折线！
输入端 T：多段线调整容差。数值越小，越多的单一线段会被认定为碎片进行坍塌转换

Rebuild Curve:重建曲线
输入端 D：曲线的阶数，默认为原阶数
输入端 N:控制点的数量
输入端 T：True 表示保留原切点的切线方向

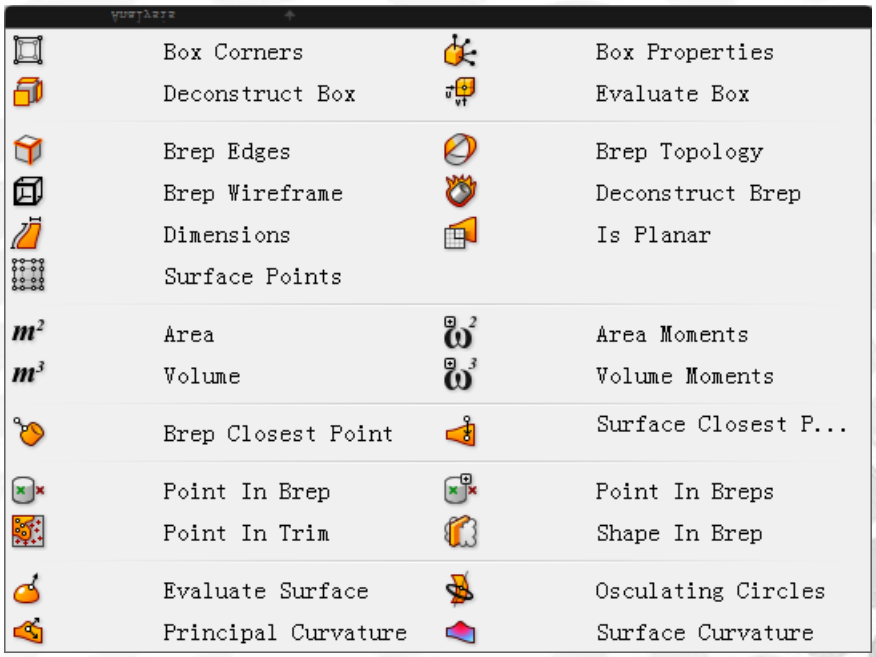
Reduce：删除不重要的顶点简化多折线
输入端 T：简化容忍度
输出端 R：简化顶点数量

Simplify Curve：简化曲线
输入端 t：偏移容差，默认为 Rhino 当前文件设置的容差值
输入端 a：角度容差，默认为 Rhino 当前文件设置的容差值

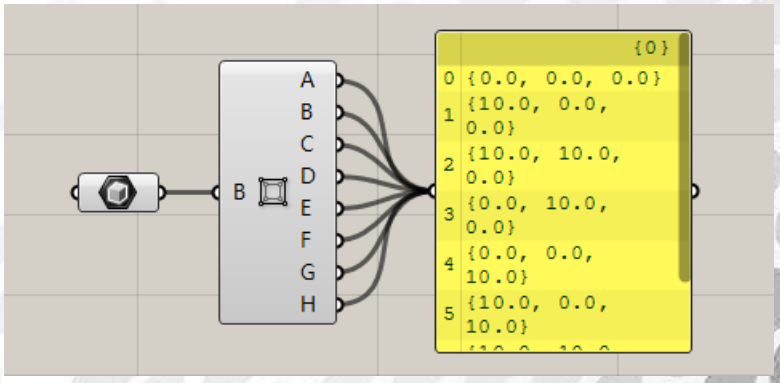
Smooth Polyline：把多段线顶点撸顺
输入端 P：输入的多段线
输入端 S：撸平滑的强度，0=不平滑，1=最平滑
输入端 T：平滑次数

6. Surface 电池组

（1）Analysis 电池序列



Box Corners:提取 box 的八个顶点（10*10*10box 为例）



Box Properties:解读 box 属性（10*10*10box 为例）

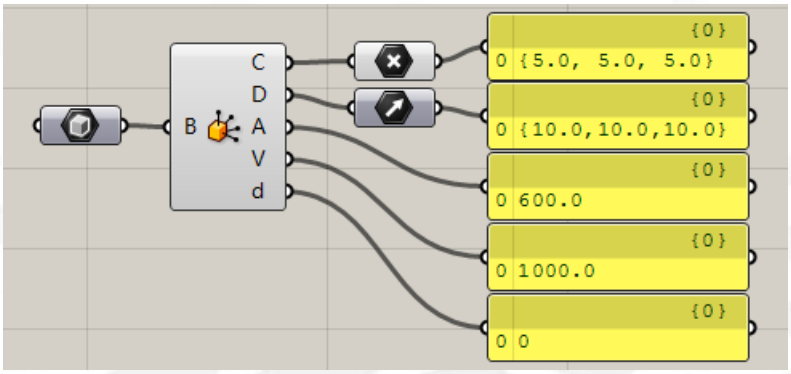
输出端 C：box 中心点

输出端 D：box 斜对角向量

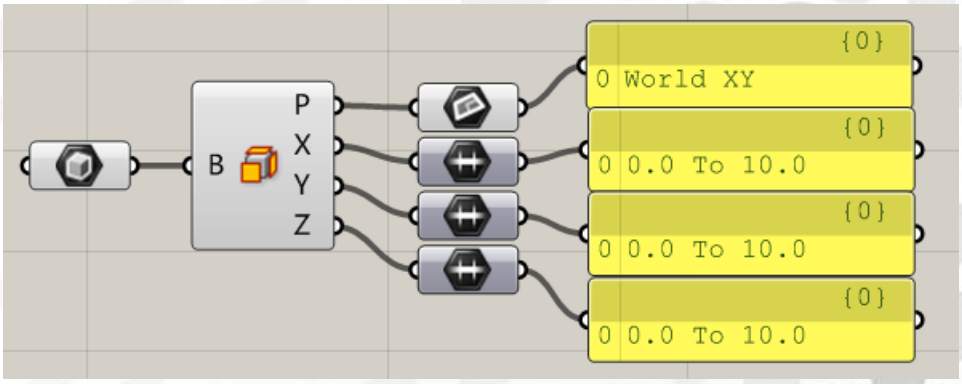
输出端 A：box 表面积

输出端 V：box 体积

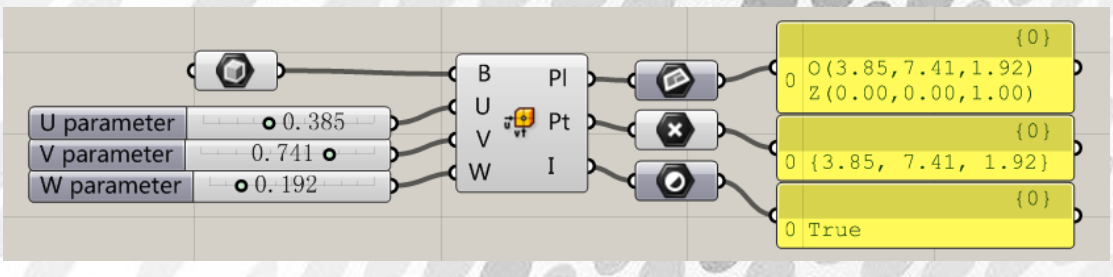
输出端 d：box 简并性



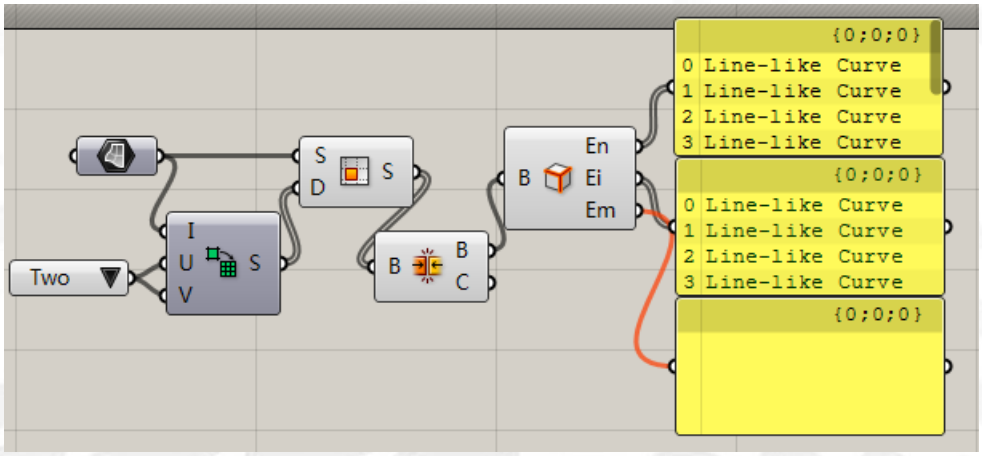
Deconstruct Box :解构 box（10*10*10box 为例）
输出端 P：box 基准平面
输出端 X：x 轴方向尺寸
输出端 Y：y 轴方向尺寸
输出端 Z：z 轴方向尺寸



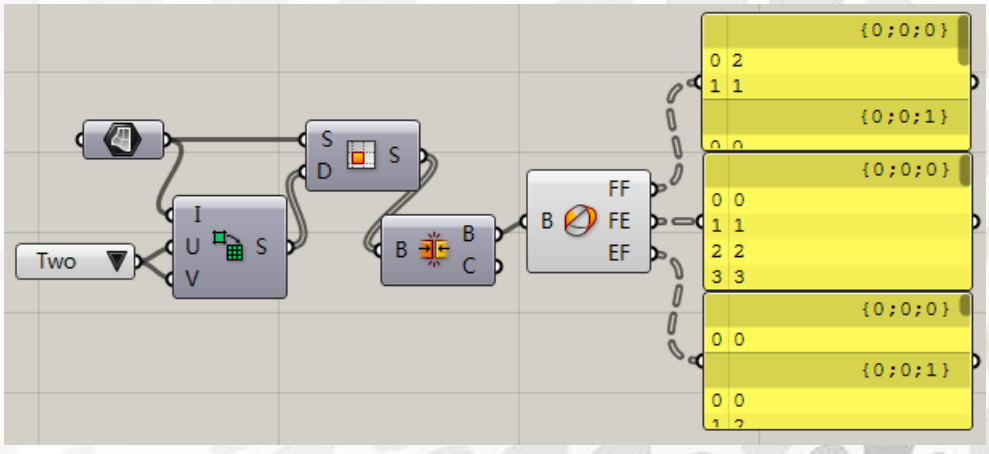
Evaluate Box:通过 UVW 生成坐标并评估在 box 内部或外部（10*10*10box 为例）
输入端 UVW：输入 UVW 的参量（当值在 0~1 内时在 box 内）
输出端 Pl：UVW 生成坐标处的平面
输出端 Pt：UVW 生成坐标处的平面
输出端 I：UVW 生成坐标是否在 box 内，True 为在 box 内



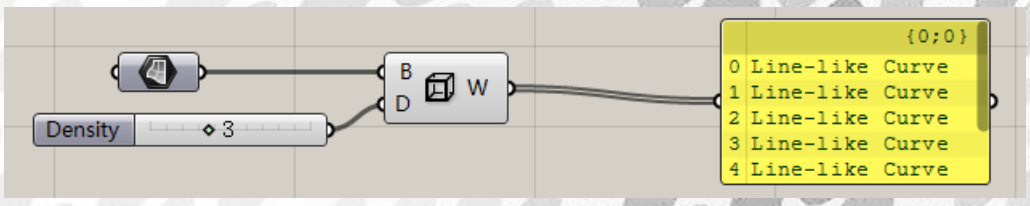
Brep Edges :提取物件的边缘曲线
输出端 En：裸露的边缘
输出端 Ei：内部的边缘
输出端 Em：非交叠边缘



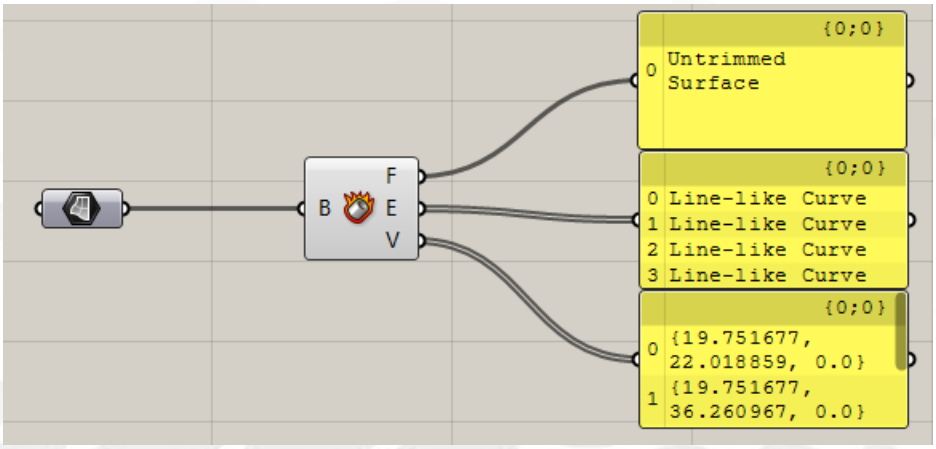
Brep Topology :获取并显示物件的拓扑属性（右键可仅在 Perspective 视窗显示）
输出端 FF：每个面与相邻所有面的拓扑
输出端 FE：每个面与相邻所有边的拓扑
输出端 EF：每条边与相邻所有面的拓扑



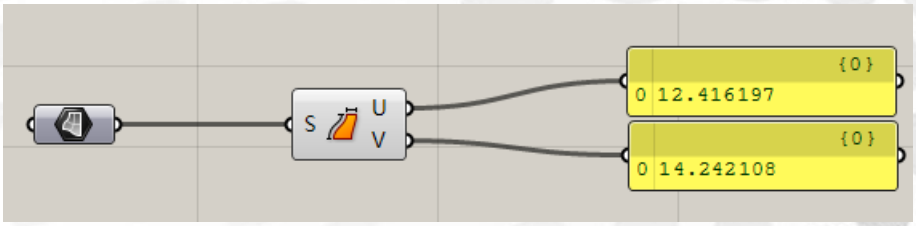
Brep Wireframe:提取物件线框
输入端 D：线框等参密度



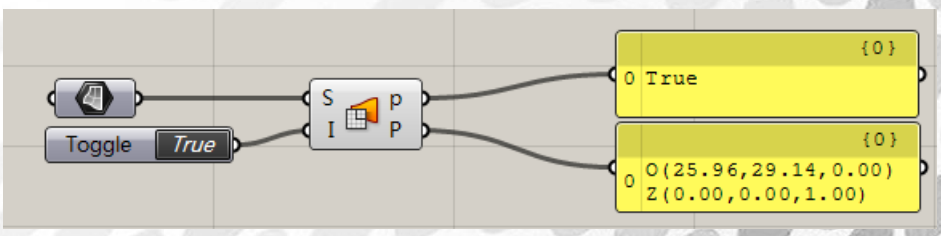
Deconstruct Brep:解构物件
输出端 F：面
输出端 E：边缘
输出端 V：顶点



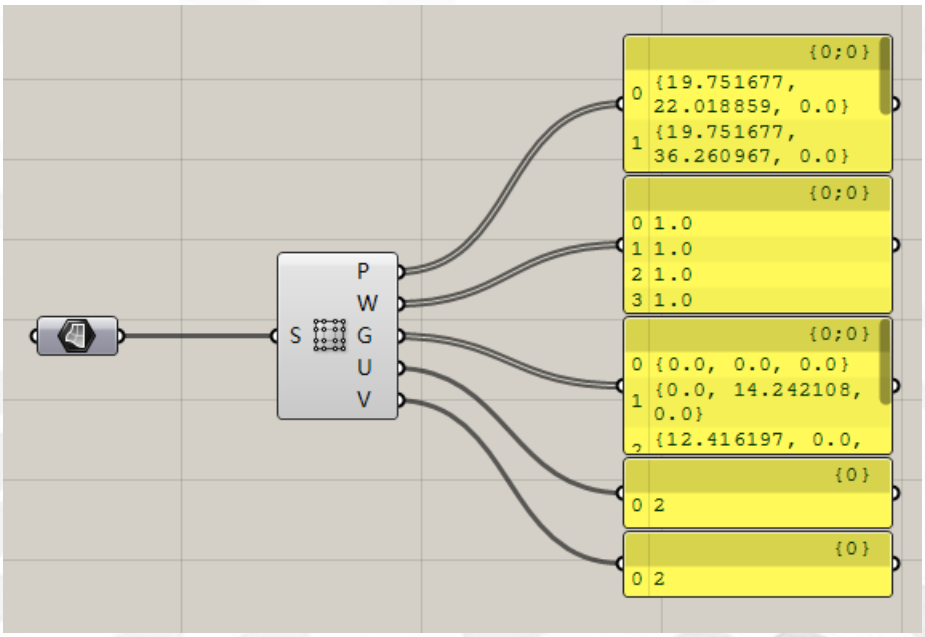
Dimensions:提取曲面大致尺寸
输出端 U：U 方向大致尺寸
输出端 V：V 方向大致尺寸



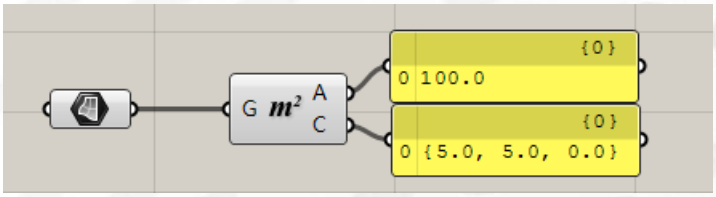
Is Planar：测试曲面是否为平面
输入端 I：是否对修剪曲面的平面性测试限定于内部，True 为限制
输出端 P：平面性（布尔值，True 为是）
输出端 P：曲面所在平面



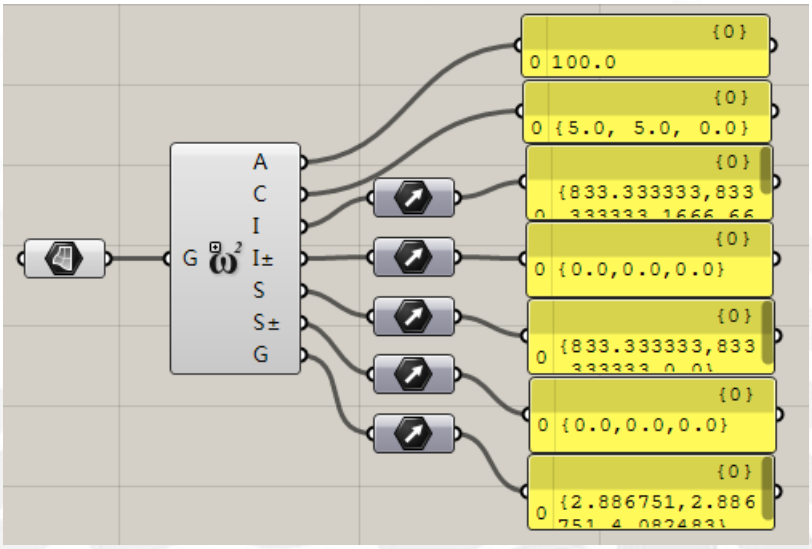
Surface Points：提取 Nurbs 曲面控制点
输出端 P：控制点位置
输出端 W：控制点权重
输出端 G：葛瑞维尔 UV 点
输出端 U：U 方向上点数量
输出端 V：V 方向上点数量



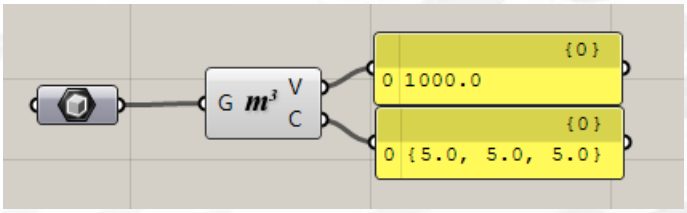
Area：获取实体、网格、平面闭合曲线的面积
输出端 A：面积
输出端 C：形心



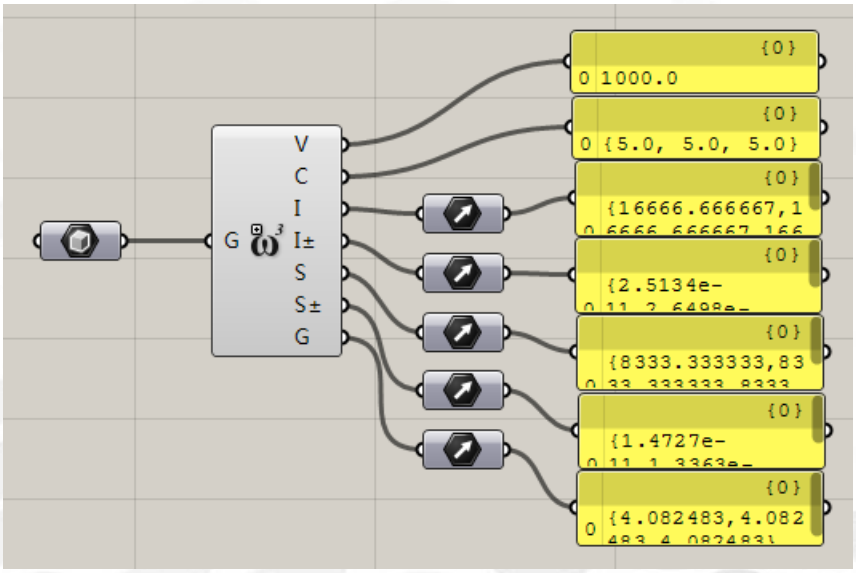
Area Moments：面积力矩
输出端 A：面积
输出端 C：形心
输出端 I：于形心处惯性力矩
输出端 I±：惯性力矩误差范围
输出端 S：于形心处惯性二次矩
输出端 S±：惯性二次矩惯性误差范围
输出端 G：旋转偏心距



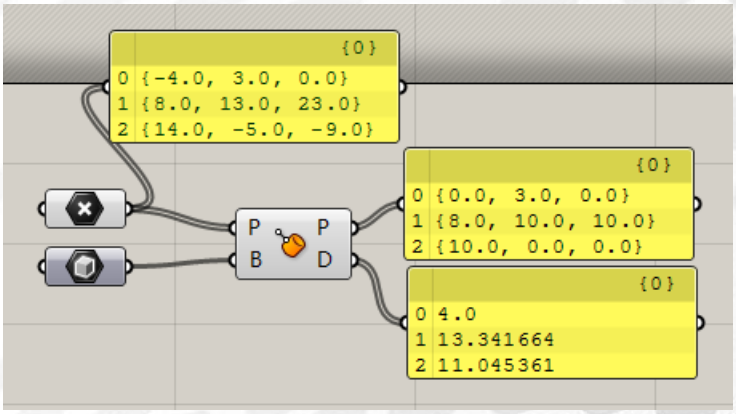
Volume：获取闭合的实体、网格的体积
输出端 V：体积
输出端 C：形心



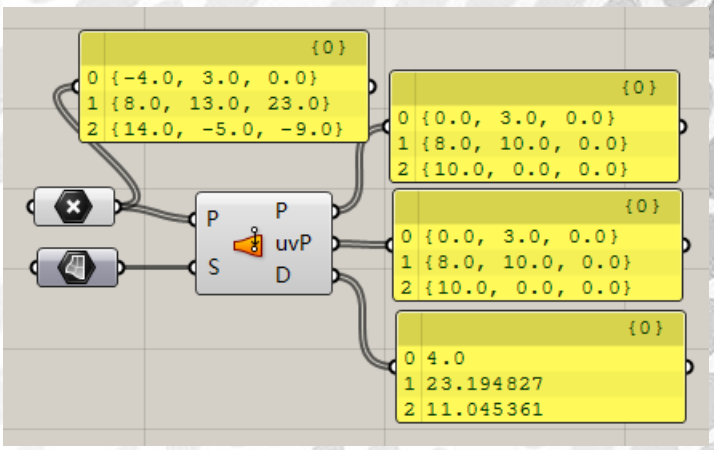
Volume Moments：体积力矩
输出端 V：体积
输出端 C：形心
输出端 I：于形心处惯性力矩
输出端 I±：惯性力矩误差范围
输出端 S：于形心处惯性二次矩
输出端 S±：惯性二次矩惯性误差范围
输出端 G：旋转偏心距



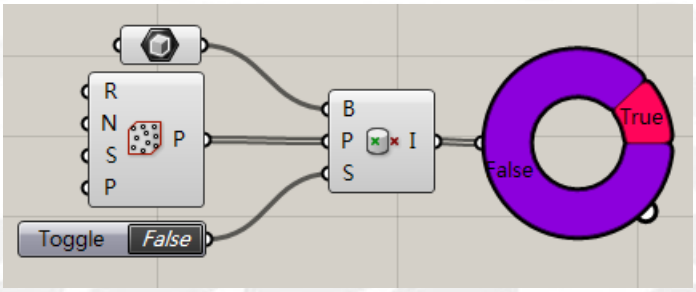
Brep Closest Point：在物件上寻找距采样点最近的点
输出端 P：最近点
输出端 D：采样点与物件的距离



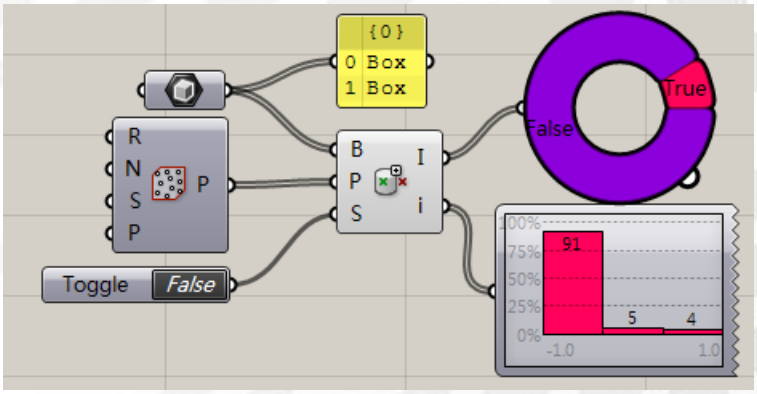
Surface Closest Point：在曲面上寻找距采样点最近的点
输出端 P：最近点
输出端 uvP：最近点在曲面上的 UV 坐标
输出端 D：采样点与曲面的距离



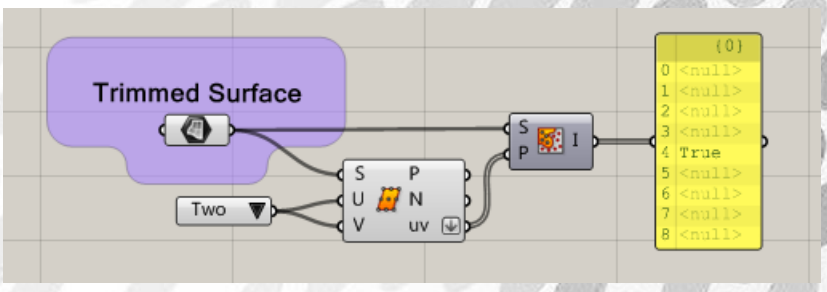
Point In Brep ： 判断点是否在闭合实体内部
输入端 S：布尔值，True 为严格判断方式（点在实体表面不算内部）
输出端 I：布尔值，True 为在内部



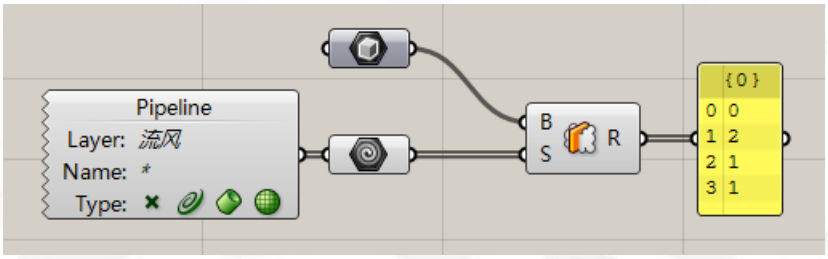
Point In Breps ： 判断点是否在多个闭合实体内部
输入端 S：布尔值，True 为严格判断方式（点在实体表面不算内部）
输出端 I：布尔值，True 为在内部，只要点在任一实体内部即 True
输出端 i：第一个包含点的实体的索引编号，-1 为未被任一实体包含的点



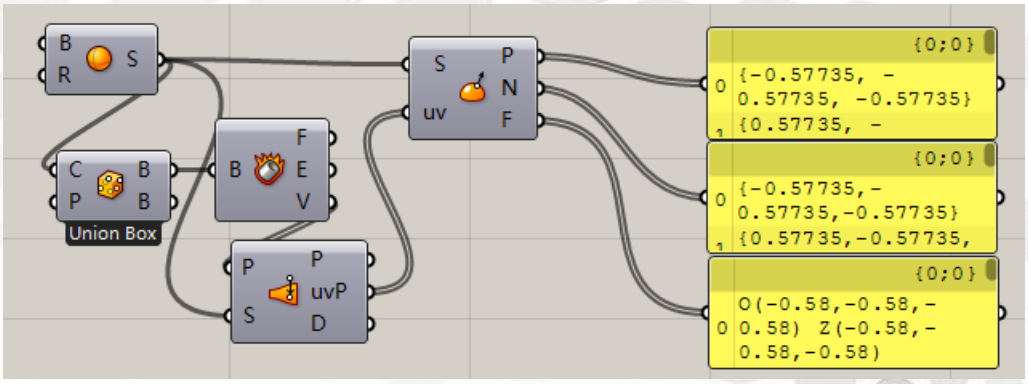
Point In Trim ： 判断坐标是否在曲面修剪的部分内
输出端 I：布尔值，True 为在修剪边界内部



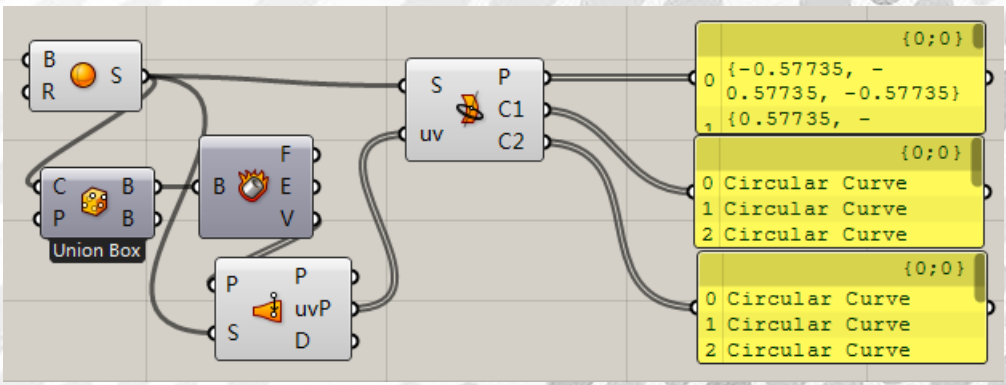
Shape In Brep ： 判断模型是否在实体内部
输出端 R：模型与实体的关系，0 表示在内部，1 表示相交，2 表示在外部



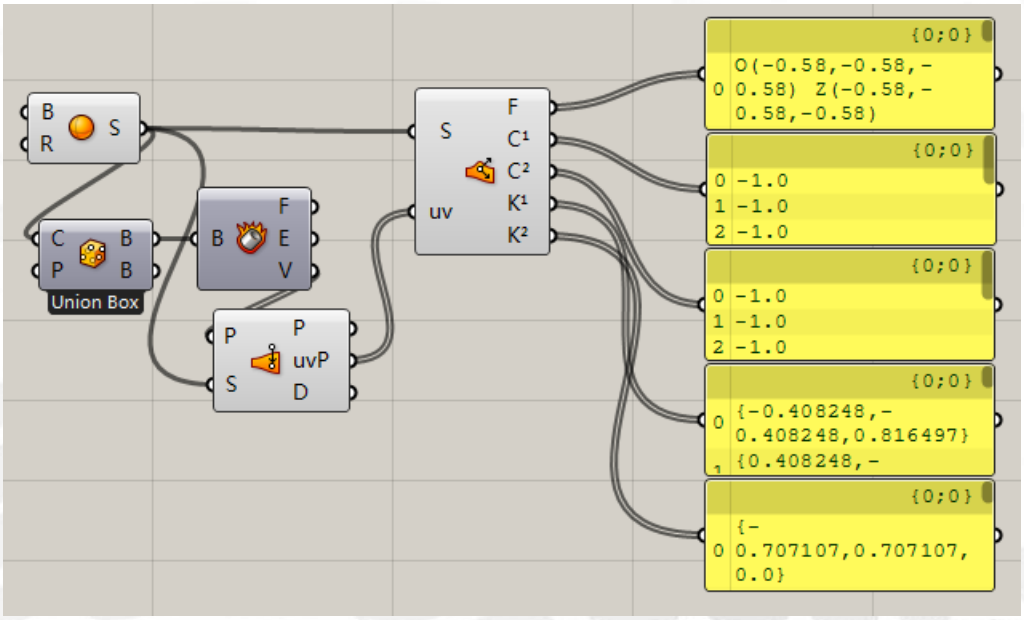
Evaluate Surface :评估曲面在一个 UV 坐标处的属性
输出端 P：UV 坐标处的点
输出端 N：UV 坐标处的法线
输出端 F：UV 坐标处的平面



Osculating Circles ： 计算曲面在一 UV 坐标处的密切圆
输出端 P：UV 坐标处的点
输出端 C1：UV 坐标处的第一个密切圆
输出端 C2：UV 坐标处的第二个密切圆



Principal CurvaTrue：评估曲面在一 UV 坐标处的主曲率
输出端 F：UV 坐标处的平面
输出端 C^1：UV 坐标处的最小主曲率
输出端 C^2：UV 坐标处的最大主曲率
输出端 K^1：UV 坐标处最小主曲率方向
输出端 K^2：UV 坐标处最大主曲率方向

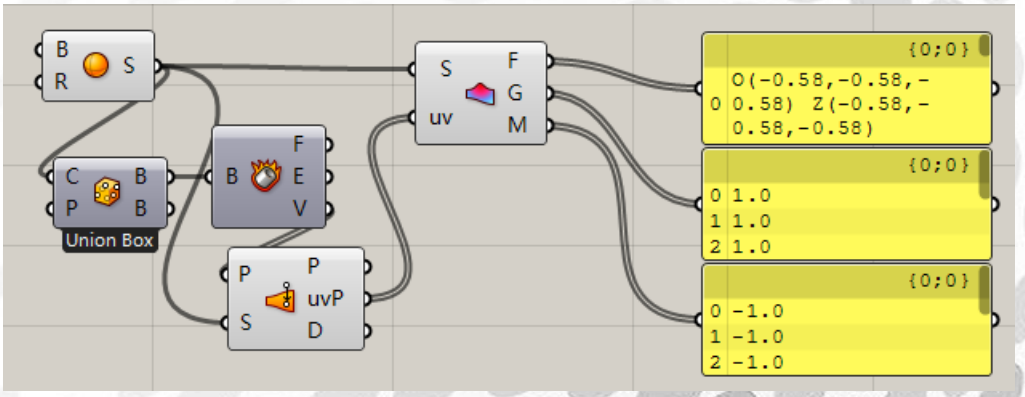


Surface CurvaTrue ： 评估曲面在一 UV 坐标处的表面曲率

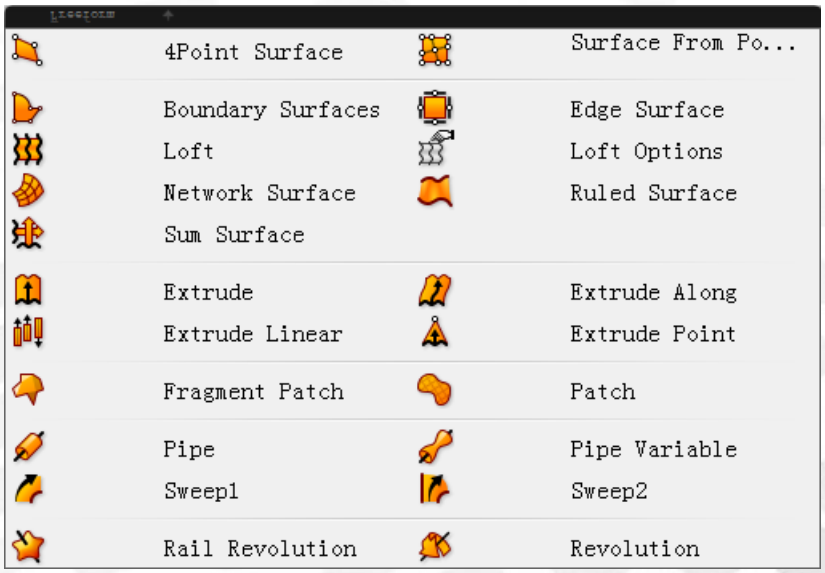
输出端 F：UV 坐标处的平面

输出端 G：高斯曲率

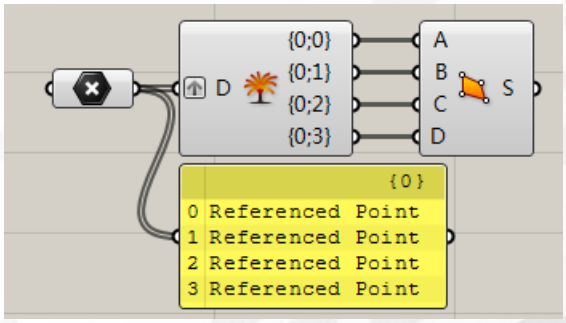
输出端 M：平均曲率



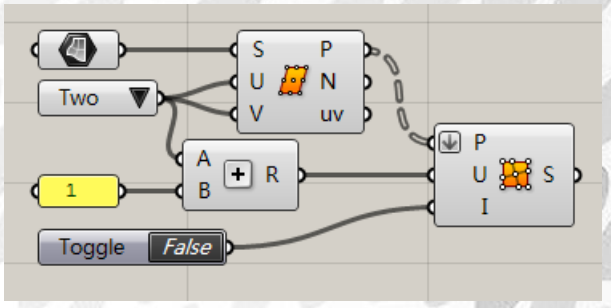
（2）Freeform 电池序列



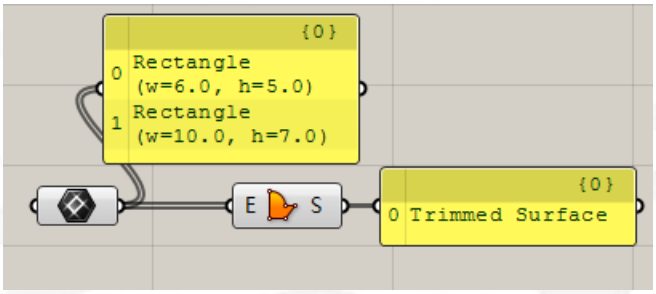
4Point Surface:连接三个或四个点生成一个曲面



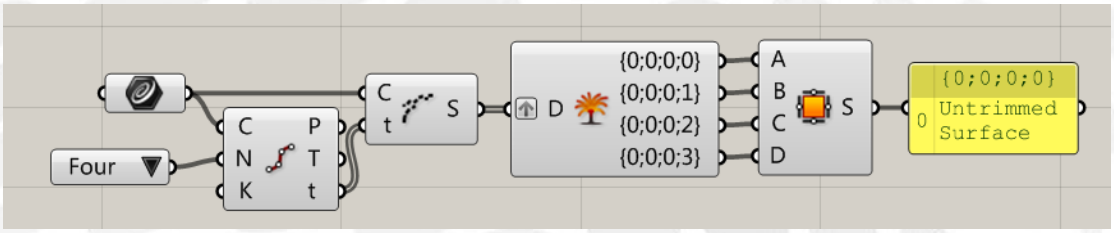
Surface From Points ： 从点阵生成一个 Nurbs 曲面
输入端 U：在 U 方向上的控制点数量
输入端 I：布尔值，True 为采取插值法采样



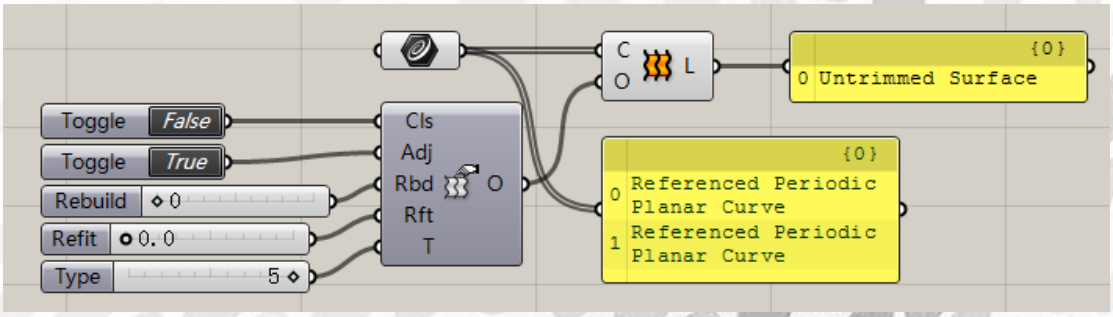
Boundary Surfaces :由边界曲线生成平面



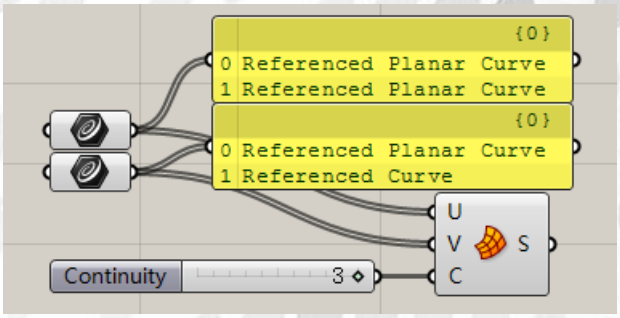
Edge Surface :由 2、3、4 条边界曲线生成曲面



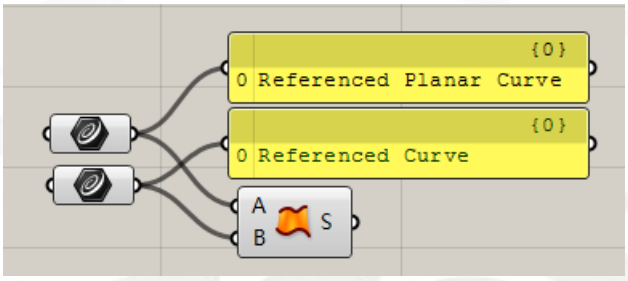
Loft :由一系列截面曲线生成放样曲面
Loft Options：放样选项
输入端 O:放样选项（右键快捷设置）
输入端 Cls:是否封闭放样，True 为是
输入端 Adj:是否调整接缝，True 为是
输入端 Rbd:重建曲面
输入端 Rft:调整容差
输入端 T:放样类型



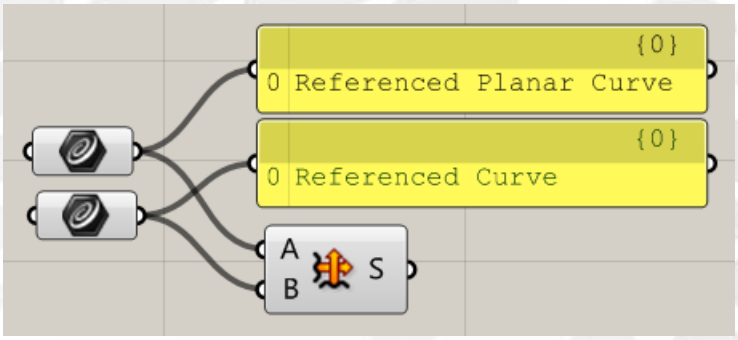
Network Surface :由网格线生成曲面
输入端 U:U 方向上的曲线（至少两条）
输入端 V:V 方向上的曲线（至少两条）
输入端 C:曲面连续性



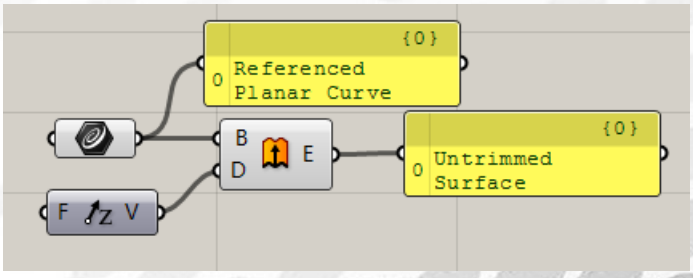
Ruled Surface :由两条曲线生成直纹曲面



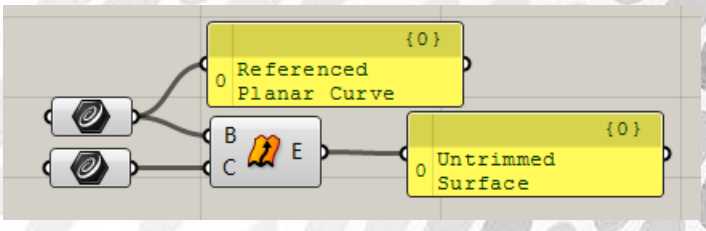
Sum Surface :由两条曲线生成和表面



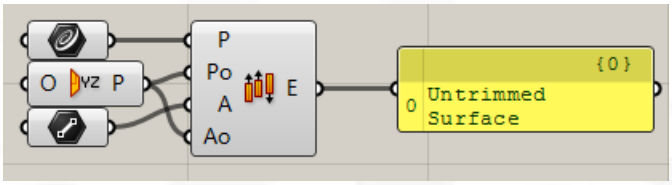
Extrude :沿一个向量挤压线或曲面
输入端 D：挤压方向



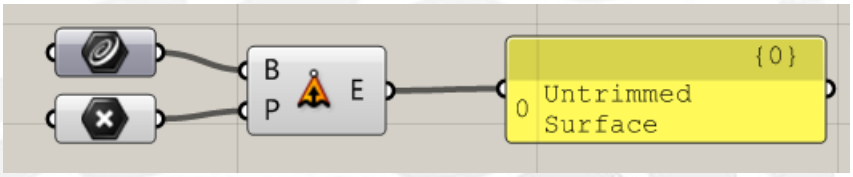
Extrude Along :沿一条曲线挤压线或曲面
输入端 C:挤压路径曲线



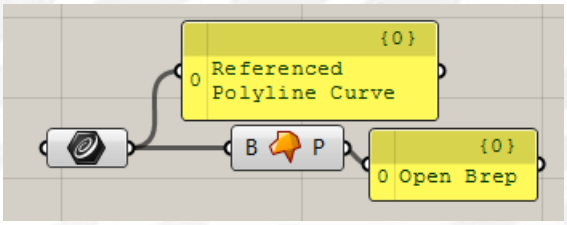
Extrude Linear :线性挤压
输入端 P:轮廓曲线或曲面
输入端 Po:轮廓形状所在指向平面
输入端 A:挤压轴
输入端 Ao:挤压轴可选方向平面



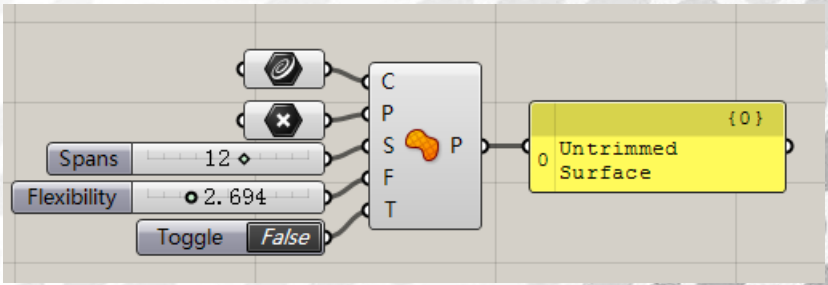
Extrude Point :将曲线或曲面挤压到一点



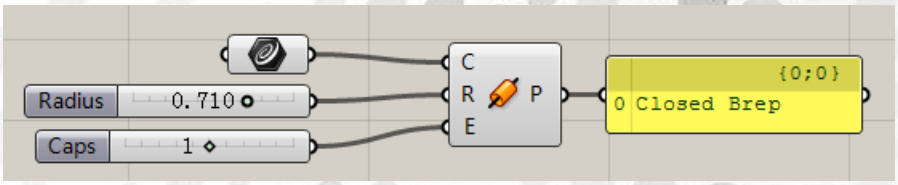
Fragment Patch ： 由多段线边界修补成面



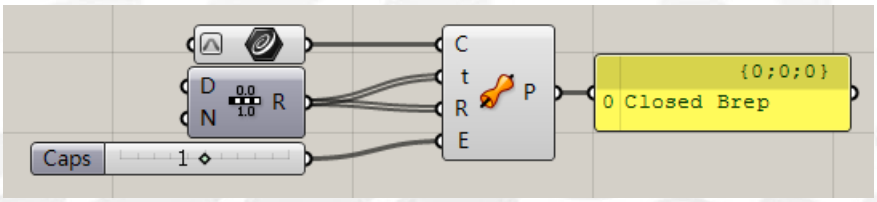
Patch ： 修补曲面
输入端 C:需要修补的曲线
输入端 P:需要修补的点
输入端 S:跨度
输入端 F:修补弹性（数值越小，弹性越小）
输入端 T:是否修剪生成结果，True 为是



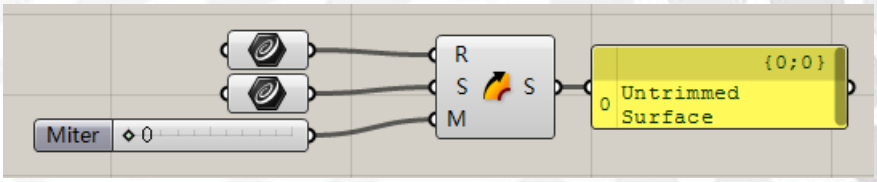
Pipe ： 沿着路径曲线生成圆管
输入端 C：路径曲线
输入端 R：圆管半径
输入端 E：指定加盖类型（0 为无，1 为平坦，2 为圆滑）



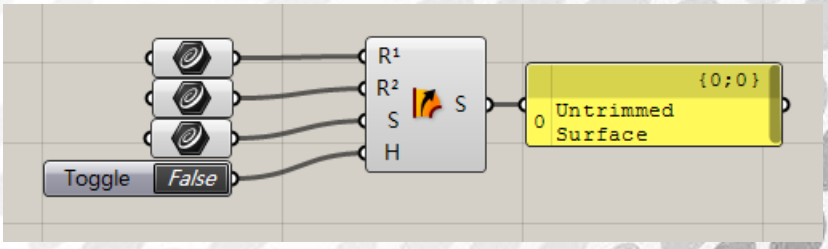
Pipe Variable ：沿着路径曲线生成半径可变的圆管
输入端 C：路径曲线
输入端 t：路径曲线半径改变处参数
输入端 R：在被定义参数处的半径
输入端 E：指定加盖类型



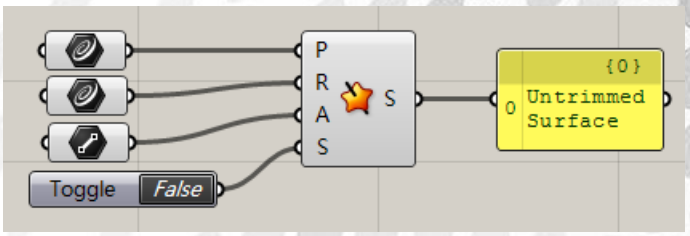
Sweep1 ：单轨扫掠
输入端 R：轨道曲线
输入端 S：截面曲线
输入端 M：扭结斜接类型（0 为无，1 为修剪，2 为循环）



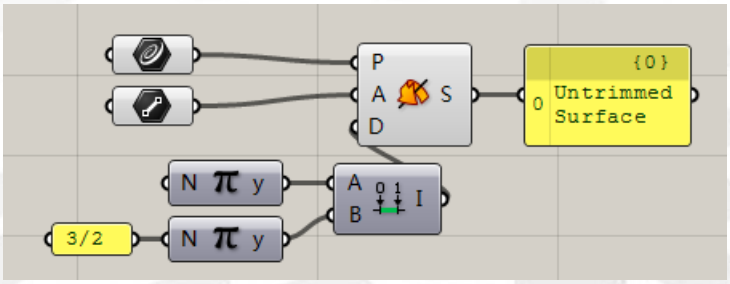
Sweep2：双轨扫掠
输入端 R^1：第一条轨道曲线
输入端 R^2：第二条轨道曲线
输入端 S：截面曲线
输入端 H：是否以等高属性扫掠，True 为是



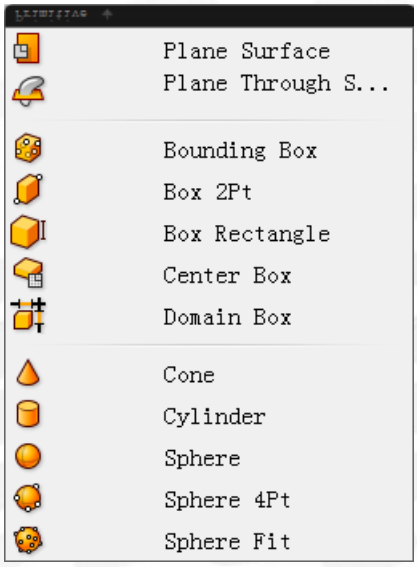
Rail Revolution ：旋转放样
输入端 P：轮廓曲线
输入端 R：轨道曲线
输入端 A：旋转轴
输入端 S：是否缩放轮廓高度，True 为是



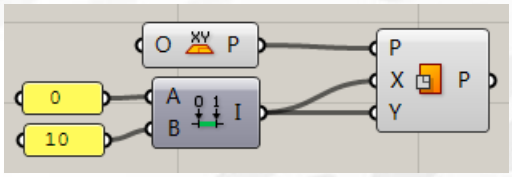
Revolution :旋转成面
输入端 P：轮廓曲线
输入端 A：旋转轴
输入端 D：旋转角度区间



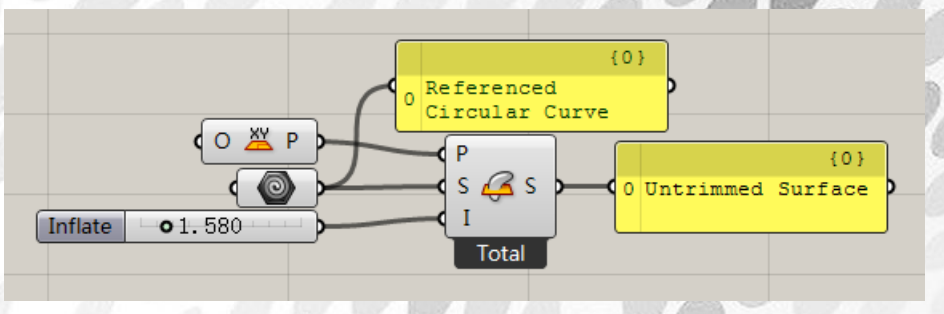
（3）Primitive 电池序列



Plane Surface ： 创建一个平面曲面
输入端 P：基准平面
输入端 X：X 方向区间
输入端 Y：Y 方向区间

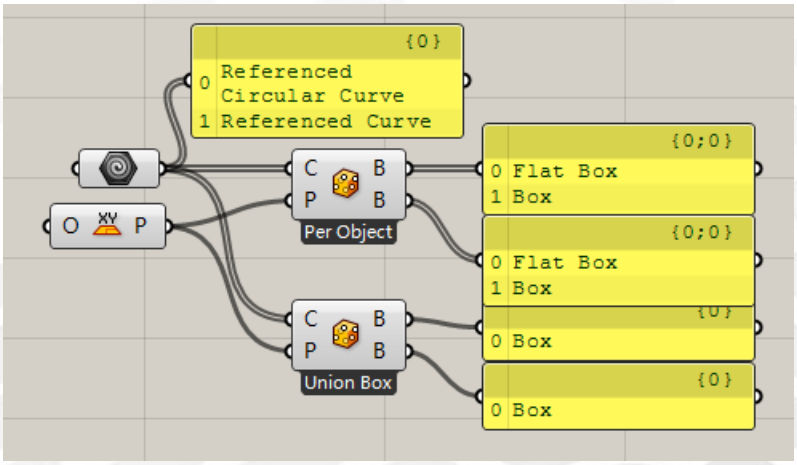


Plane Through Shape ： 创建一个围合给定模型并扩大一定范围的矩形平面（右键可选择仅围合断面）
输入端 P：基准平面
输入端 S：需要围合的模型
输入端 I：围合扩大距离

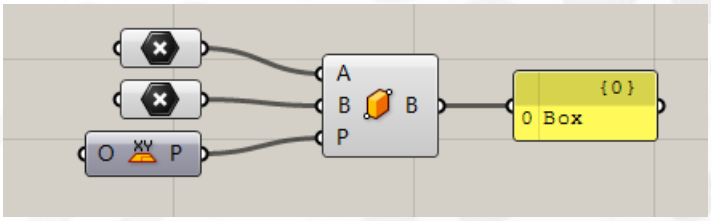


Bounding Box ： 生成围合给定模型的边界 Box
输入端 C：需围合的模型

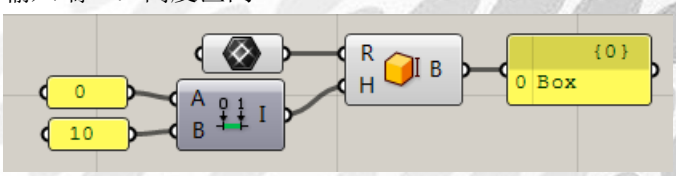
输入端 P：基准定位平面
输出端 B：世界坐标内的围合 box
输出端 B：定位平面上的围合 box



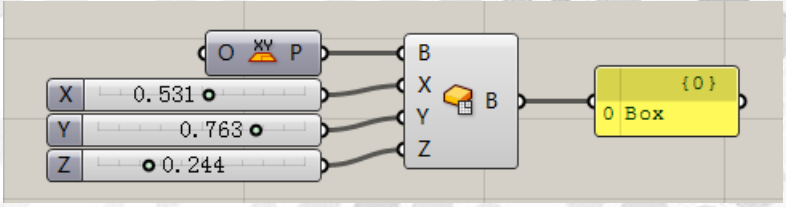
Box 2Pt ： 由两点定义一个 Box
输入端 A：第一点
输入点 B：第二点
输入端 P：基准平面



Box Rectangle ： 由矩形和高度定义一个 Box
输入端 R：基准矩形
输入端 H：高度区间

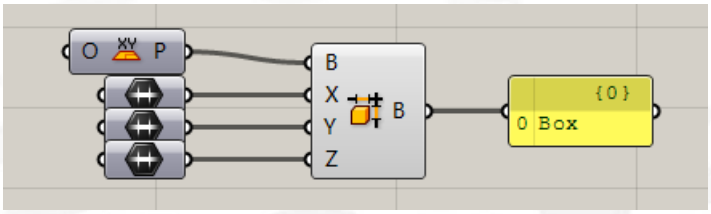


Center Box ： 由点定义一个中心 Box
输入端 B：基准中心点平面
输入端 XYZ：X、Y、Z 方向上的尺寸

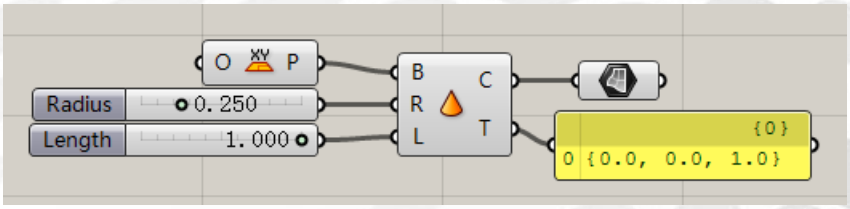


Domain Box ： 由尺寸区间定义一个 Box

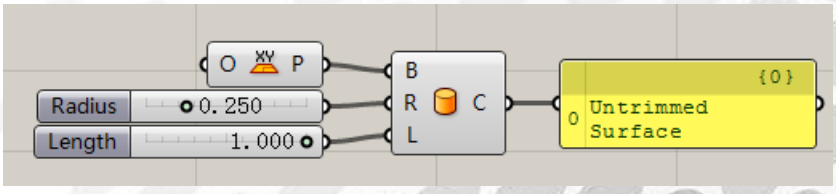
输入端 B：基准曲面
输入端 XYZ：XYZ 方向上的尺寸区间



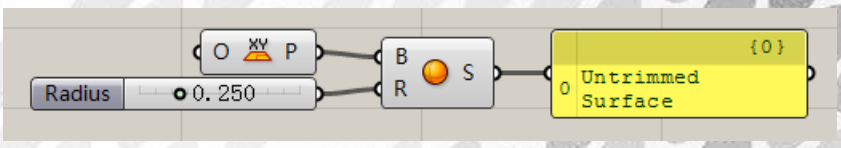
Cone ： 圆锥体
输入端 B：基准平面
输入端 R：底面半径
输入端 L：高度
输出端 C：锥形面
输出端 T：顶点



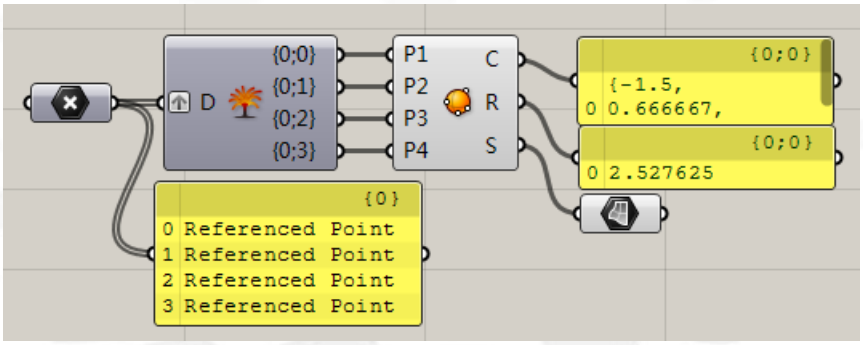
Cylinder：圆柱体
输入端 B：基准平面
输入端 R：半径
输入端 L：高度



Sphere ： 球体
输入端 B：基准中心点平面
输入端 R：半径



Sphere 4Pt ： 由四个点生成一个球体
输出端 C：球体中心点
输出端 R：球体半径
输出端 S：球面

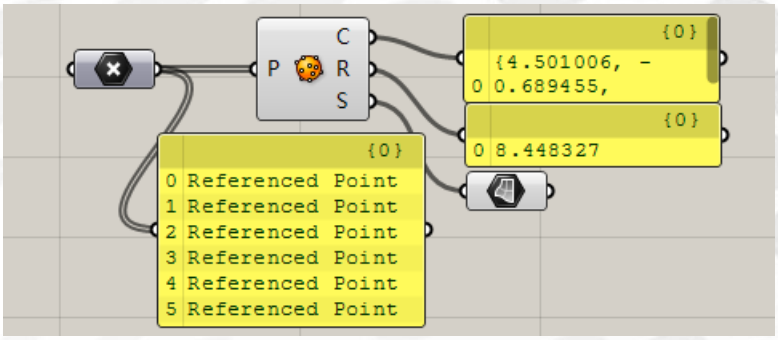


Sphere Fit ： 由一系列 3D 点生成最适合球体

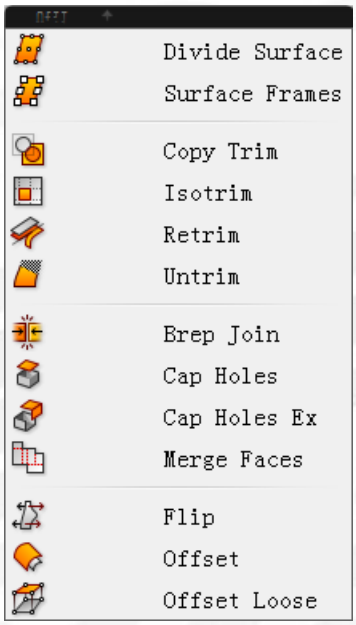
输出端 C：球体中心点

输出端 R：球体半径

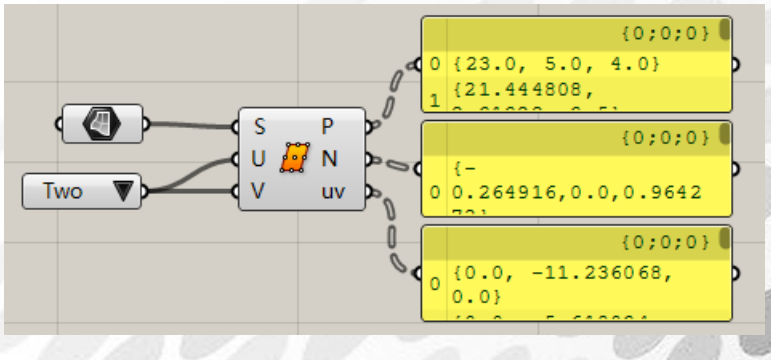
输出端 S：球面



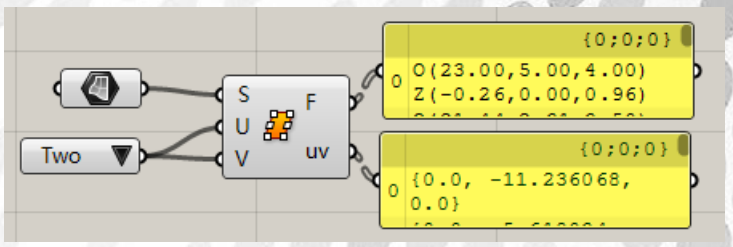
（4） Util 电池序列



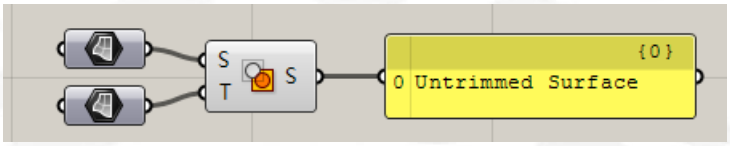
Divide Surface :细分曲面
输入端 UV：U、V 方向细分数量
输出端 P：细分点
输出端 N：细分点处的法线
输出端 uv：细分点在曲面上的 uv 坐标



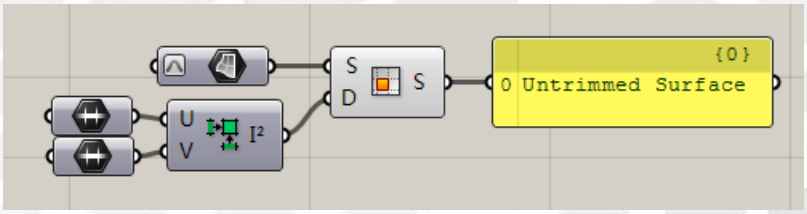
Surface Frames ： 曲面平面
输入端 UV：U、V 方向细分数量
输出端 F：细分点处的平面
输出端 uv：细分点在曲面上的 uv 坐标



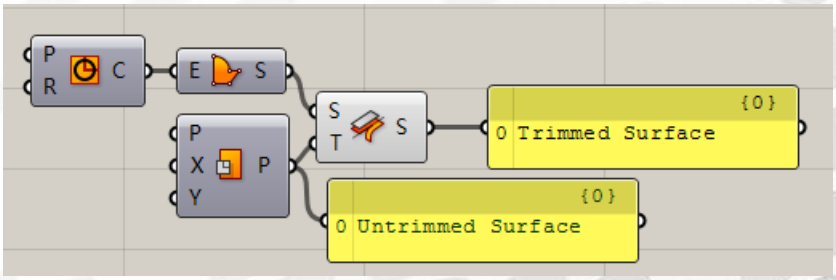
Copy Trim ： 将一个曲面的 UV 信息复制到另一个曲面上
输入端 S：提供 UV 的源曲面
输入端 T：目标曲面



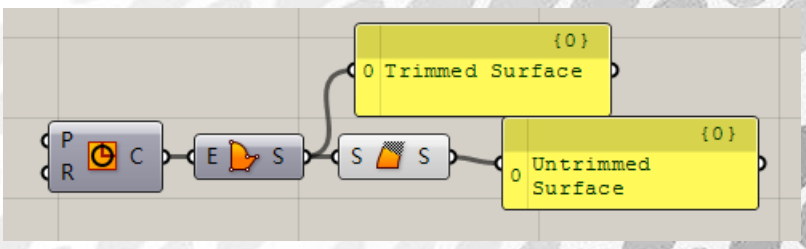
Isotrim ： 提取曲面的一个参量子集
输入端 S：基准曲面
输入端 D：子集二维区间



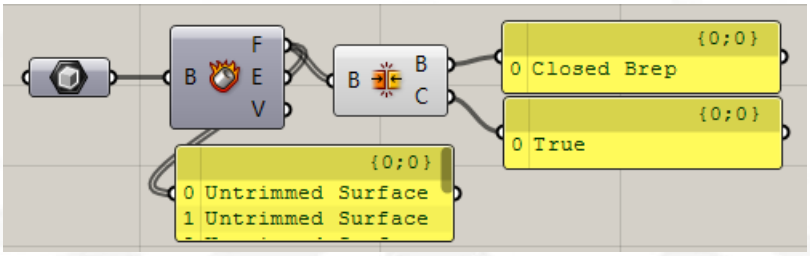
Retrim ： 基于另一个曲面的修剪数据重新调整曲面
输入端 S：提供修剪数据的源曲面
输入端 T：目标曲面



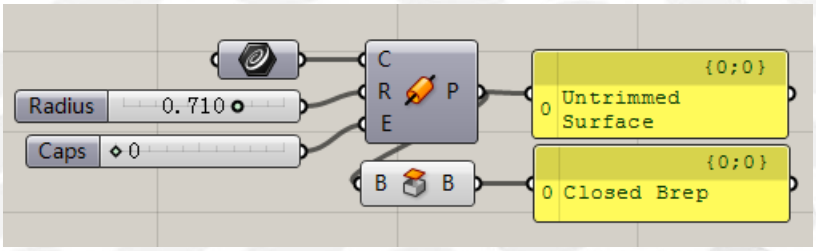
Untrim ： 取消修剪



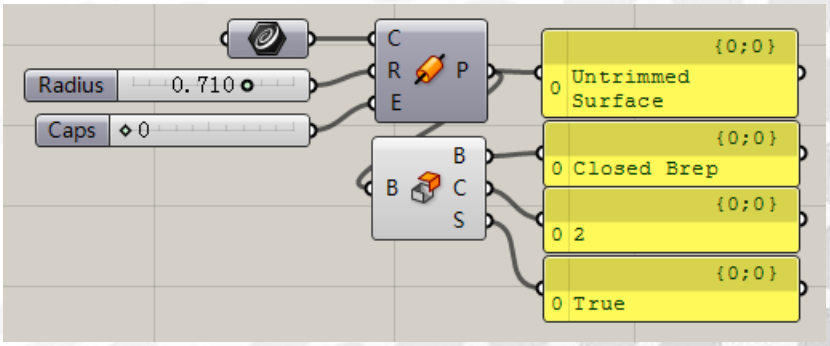
Brep Join ： 合并实体
输出端 B：合并后的实体
输出端 C：检验合并后的实体是否闭合，True 为是



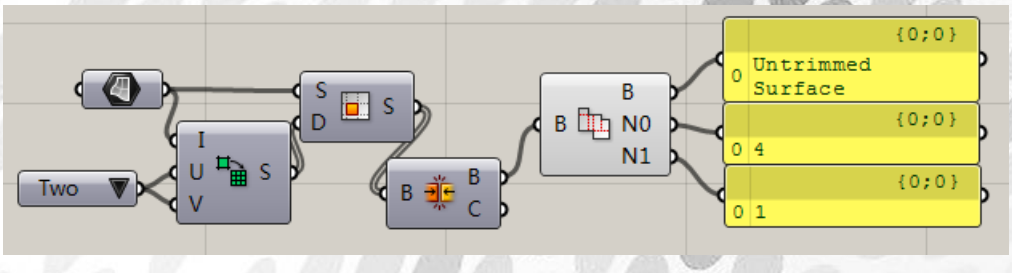
Cap Holes ： 封洞



Cap Holes Ex ： 尽可能对物件封洞
输出端 B：封洞后的物件
输出端 C：封洞数量
输出端 S：判断封洞后的物件是否为实体，True 为是

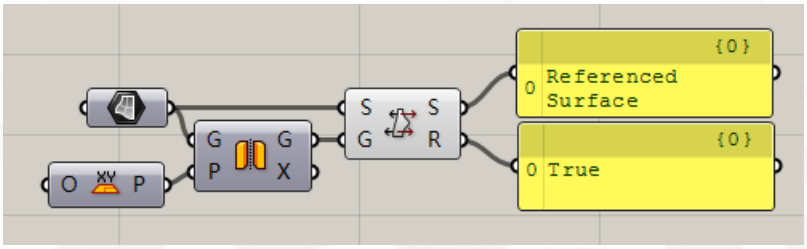


Merge Faces ： 融合所有相邻的二维平面
输出端 B：简化后的物件
输出端 N0：简化前的面数量
输出端 N1：简化后的面数量

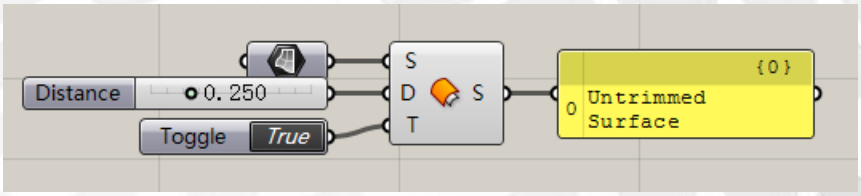


Flip：基于另一个曲面法线调整曲面法线
输入端 S：需要调整法线的曲面
输入端 G：可选的引导曲面

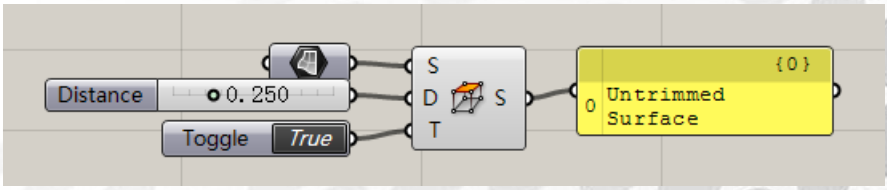
输出端 S：调整后的曲面
输出端 R：检查是否法线被调整，True 为是



Offset ： 以定值偏移曲面
输入端 S：基准曲面
输入端 D：偏移距离
输入端 T：是否重新调整，True 为是

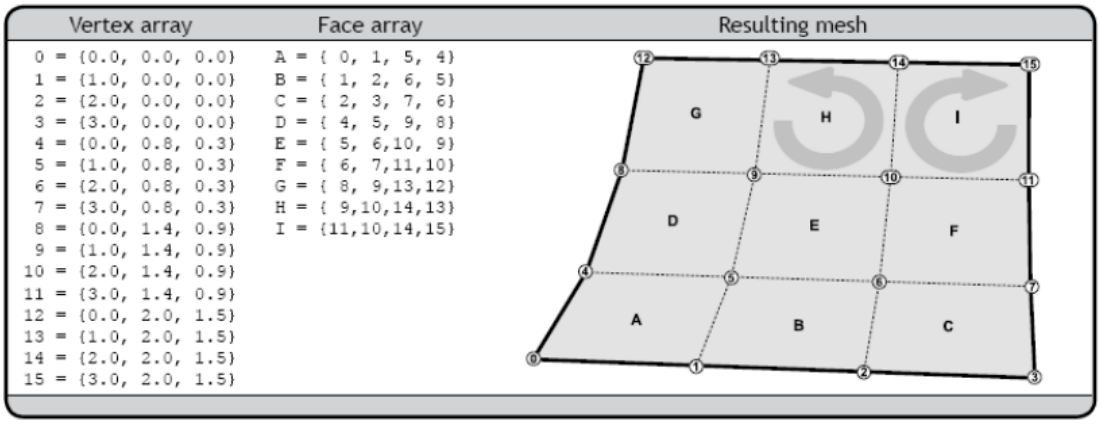


Offset Loose ： 以偏移曲面控制点的方式偏移曲面
输入端 S：基准曲面
输入端 D：偏移距离
输入端 T：是否重新调整，True 为是



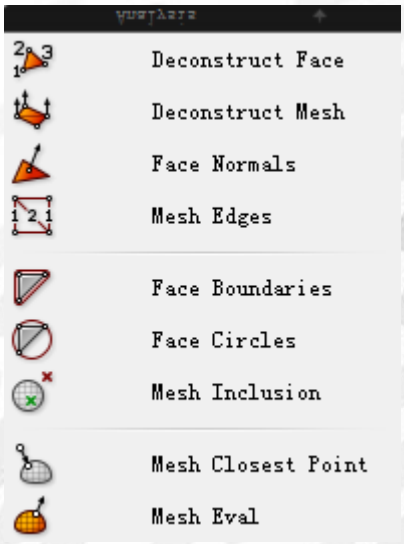
7. Mesh 电池组

Mesh 的构成

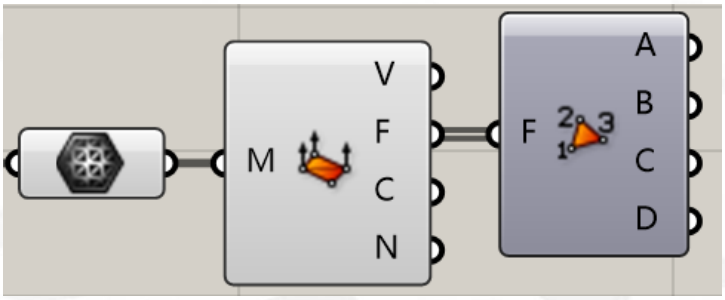


Rhino 网格模型：由四边形或三角形面构成
构成：顶点列表、面列表（顶点序号）、顶点颜色、顶点法向量

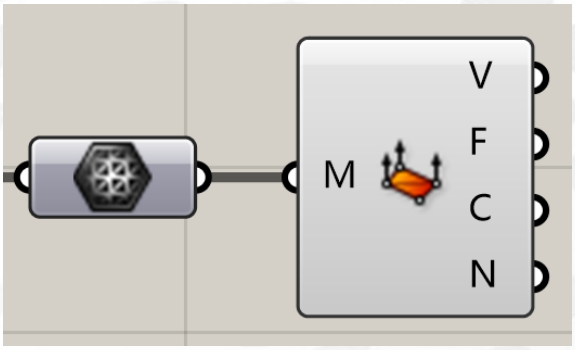
（1）Analysis 电池序列



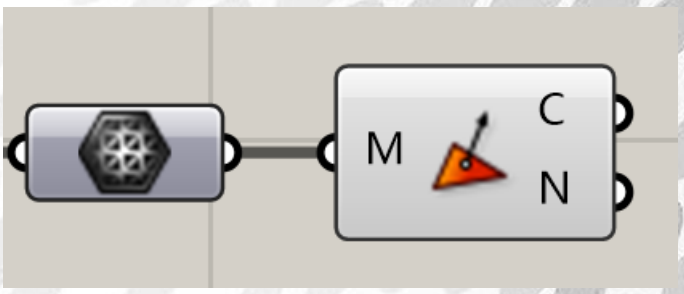
Deconstruct Face:分解网格面
输入端 F:要分解的网格面
输出端 A/B/C/D:处于相同网格顶点标号位置的顶点序号



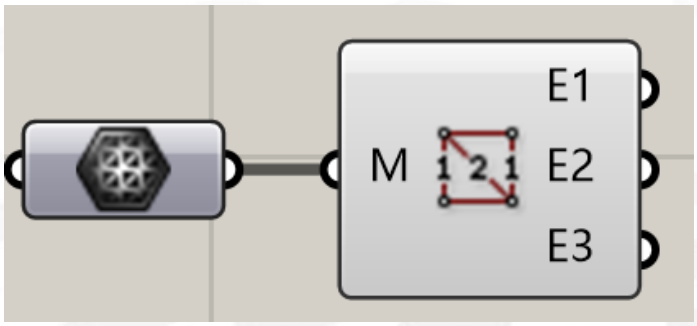
Deconstruct Mesh:分解网格
输入端 M:要分解的网格
输出端 V:网格顶点
输出端 F:网格面
输出端 C:网格颜色
输出端 N:网格面法向量



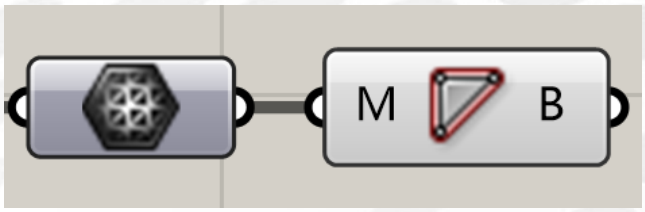
Face Normals:网格面法向量
输入端 M:输入的网格
输出端 C:网格面的中心点
输出端 N:网格面的法向量



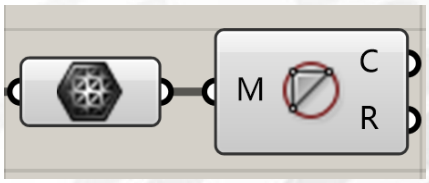
Mesh Edges:网格边线（注意输出类型为 Line）
输入端 M:输入的网格
输出端 E1:只属于 1 个网格面的边线
输出端 E2:属于 2 个网格面的边线
输出端 E3:属于 3 个或 3 个以上的网格面的边线



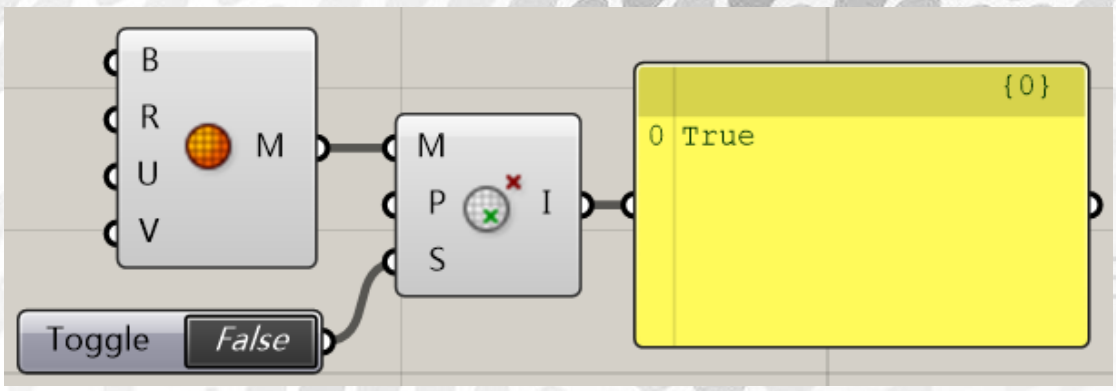
Mesh Edges:网格边线（注意输出类型为 Polyline）
输入端 M:输入的网格
输出端 B:输出每一个网格面的边线



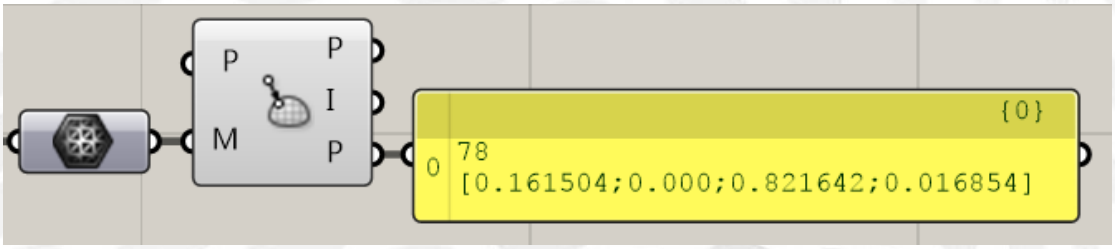
Mesh Edges:获取网格面(三角面)的外接圆
输入端 M:输入的网格
输出端 C:输出每一个网格面的外接圆
输出端 R:输出每一个网格面的高度与最长边的比值（注意以此可以判断网格面的扁平程度）



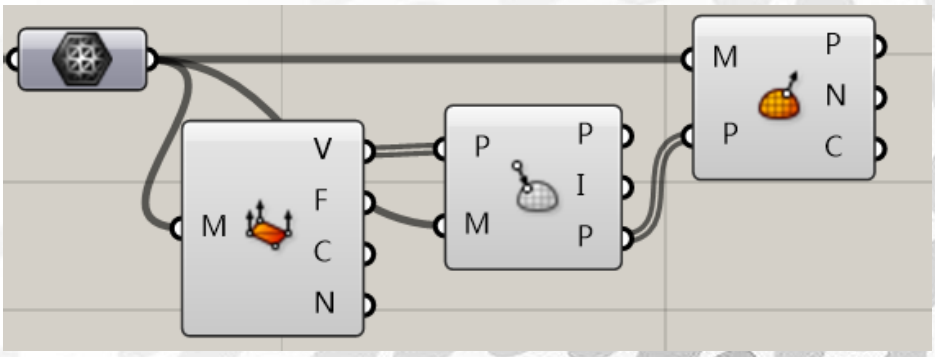
Mesh Inclusion:判断点是否在被包含在网格内
输入端 M:输入的网格（注意网格需要封闭）
输入端 P:需要判断的测试点
输入端 S:是否严格判断
输出端 P:True，点在网格内或者网格上；False，点在网格外



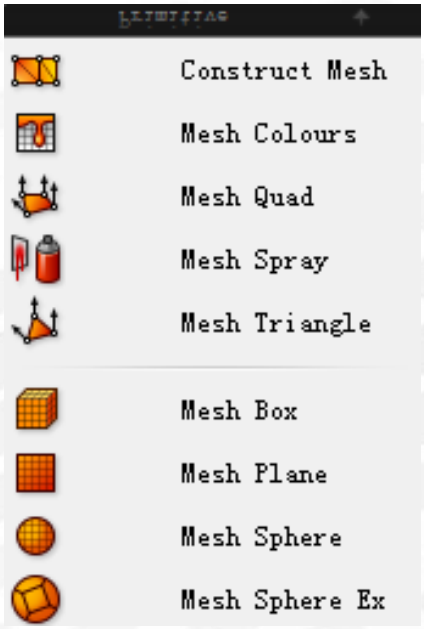
Mesh Closest Point:网格上距离测试点最近的点
输入端 M:输入的网格
输入端 P:需要判断的测试点
输出端 P:网格上距离测试点最近的点
输出端 I:最近点所在的网格面
输出端 P:最近点的相关参数(所在网格面和最近点的插值系数,格式:面序号[f1;f2;f3;f4];f1,f2,f3,f4:面顶点,(V1,V2,V3,V4)的插值系数， 最近点坐标=V1*f1+V2*f2+V3*f3+V4*f4)



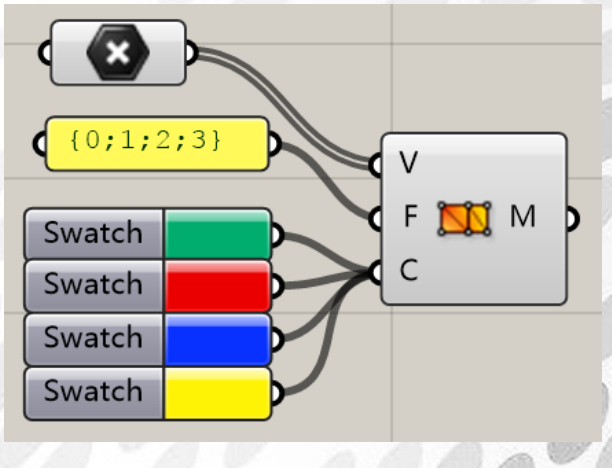
Mesh Eval:根据给定的网格参数评估网格
输入端 M:输入的网格
输入端 P:需要评估的网格参数
输出端 P:该网格参数下的点
输出端 N:该点所在网格面的法向量
输出端 C:网格面颜色

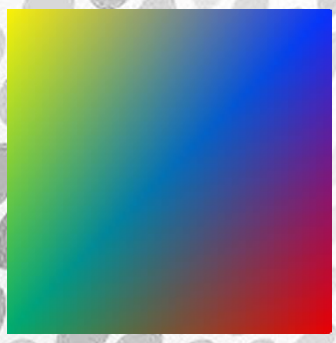


（2）Primitive 电池序列

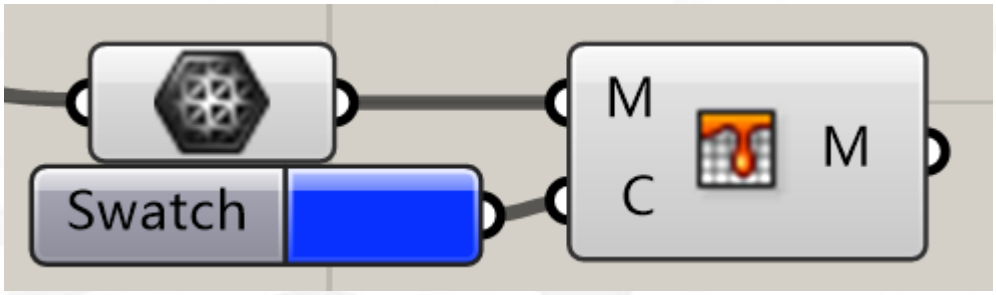


Construct Mesh:构造网格
输入端 V:组成网格面的顶点
输入端 F:网格面顶点序号组成列表
输入端 C:网格点的颜色
输出端 M:输出的网格

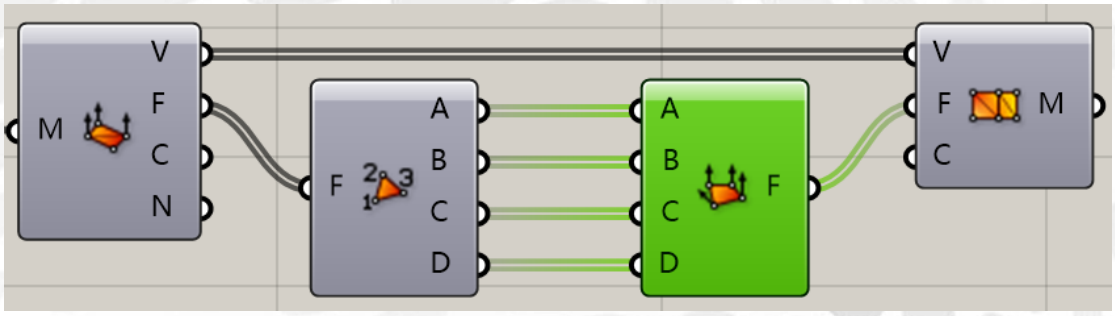




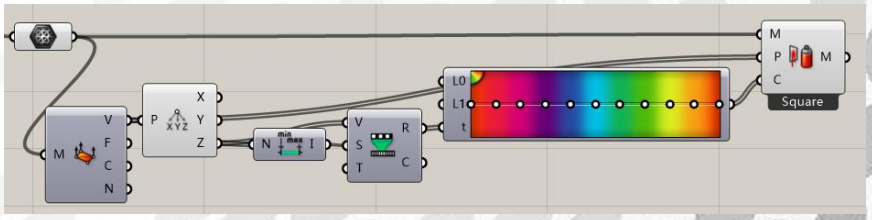
Mesh Colors:网格颜色
输入端 M:输入的网格
输入端 C:网格面的颜色
输出端 M:输出的网格

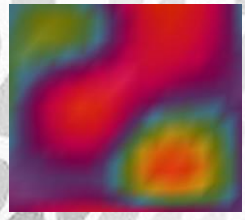


Mesh Quad:四边网格面
输入端 A/B/C/D: 处于相同网格顶点标号位置的顶点序号
输入端 F:网格面顶点序号组成列表

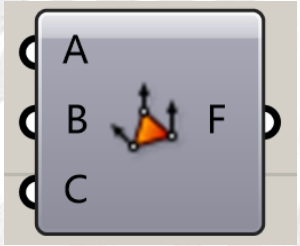


Mesh Spray:网格喷枪
输入端 M:输入的网格
输入端 P:着色的点
输入端 C:颜色
输出端 M:着色的网格



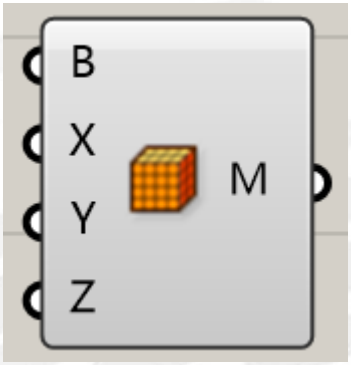


Mesh Triangle:三边网格面
输入端 A/B/C/: 处于相同网格顶点标号位置的顶点序号
输出端 F:网格面顶点序号组成列表

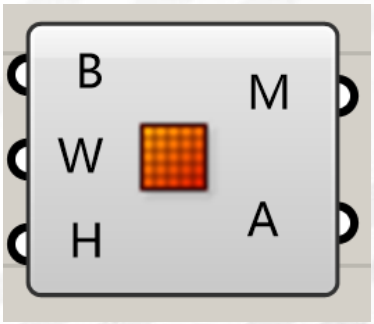


Mesh Box:网格立方体
输入端 B:Box 几何体

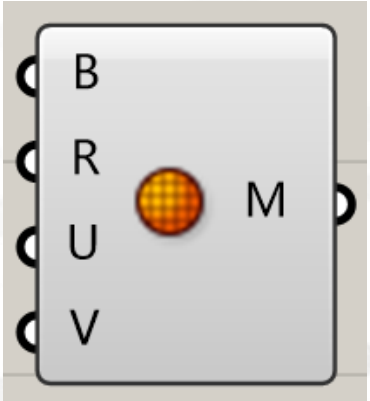
输入端 X:X 方向细分
输入端 Y:Y 方向细分
输入端 Z:Z 方向细分
输出端 M:网格立方体



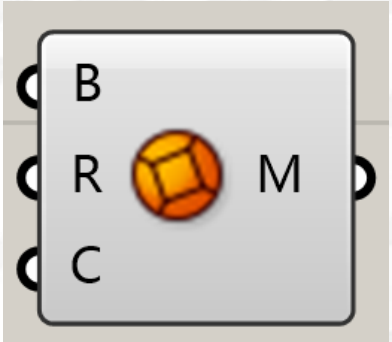
Mesh Plane:网格平面
输入端 B:网格边界（Rectangle）
输入端 W:X 方向长度
输入端 H:Y 方向长度
输出端 M:网格平面
输出端 A: 网格平面面积



Mesh Sphere:网格球体
输入端 B:坐标平面
输入端 R:球体半径
输入端 U:U 方向细分
输入端 V:V 方向细分
输出端 M: 网格球体

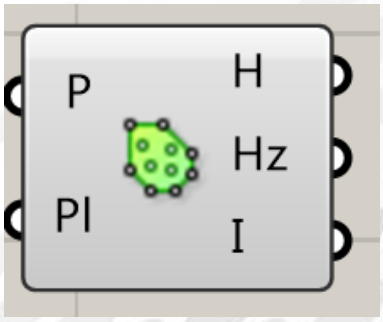


Mesh Sphere EX:拓扑网格球体
输入端 B:坐标平面
输入端 R:球体半径
输入端 C:每个嵌面的细分
输出端 M:拓扑网格球体

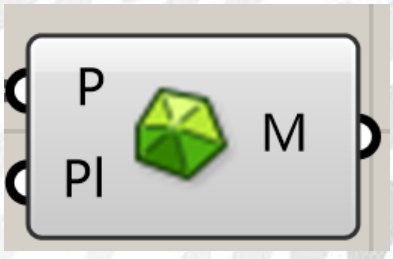


（3）Triangulation 电池序列

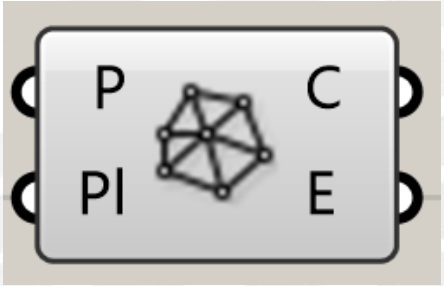
Convex Hull：凸包
输入端 P:用于求解凸包的点集
输入端 PI:工作平面（凸包投影平面）
输出端 H:在工作平面内的凸包边界（Ployline）
输出端 Hz:基于工作平面投影的空间凸包边界（Ployline）
输出端 I:空间凸包顶点在原点集中的序号



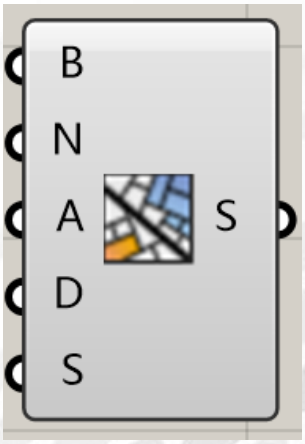
Delaunay Mesh：Delaunay 三角剖分
输入端 P:用于求解三角剖分的点集
输入端 PI:工作平面（凸包投影平面）
输出端 M:三角剖分网格



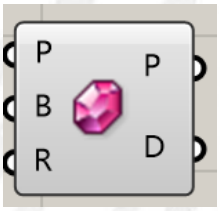
Delaunay Edges 三角剖分网格边界
输入端 P:用于求解三角剖分的点集
输入端 PI:工作平面（凸包投影平面）
输出端 C:每个顶点连接的拓扑关系（对应和该顶点相连的顶点序号）
输出端 E:三角剖分网格边界（Line）



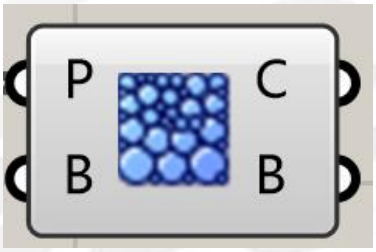
Substrate：生成 Jared Tarbell 肌理
输入端 B:肌理边界
输入端 N:肌理图线的数量
输入端 A:肌理图的角度（弧度）
输入端 D:新线段的最大变化角度（弧度）
输入端 S:随机种子
输出端 S: Jared Tarbell 肌理



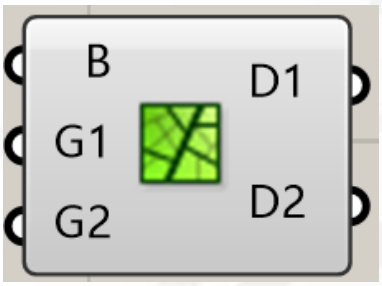
Facet Dome：根据点集 P 拟合球面 D，并生成外切 D 的多边形集(切点为 P 在球面上的投影)
输入端 P:输入的点集
输入端 B: Box 范围
输入端 R:多边形半径
输出端 P:生成的多边形
输出端 D:拟合的球面



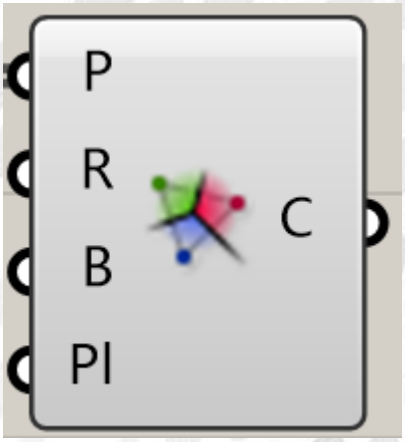
Voronoi 3D：根据点集生成 voronoi 多面体
输入端 P:输入的点集
输入端 B:范围 Box
输出端 C: Voronoi 多面体
输出端 B: 判断多面体是否在边界上



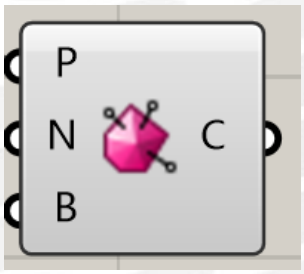
Voronoi Groups：分组划分 Voronoi 多边形
输入端 B:矩形范围
输入端 G1/G2/G3/…: 输入的点集
输出端 D1/D2/D3/…: 根据点集 G1 生成 Voronoi 多边形 D1，再根据点集 G2，将 D1 进行 Voronoi 划分



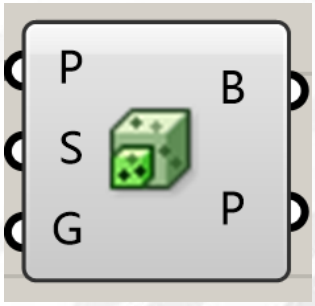
Voronoi (2D)：根据点集生成 voronoip 多边形
输入端 P:输入的点集
输入端 R:多边形半径
输入端 B:矩形范围
输入端 PI:工作平面
输出端 C: Voronoi 多边形



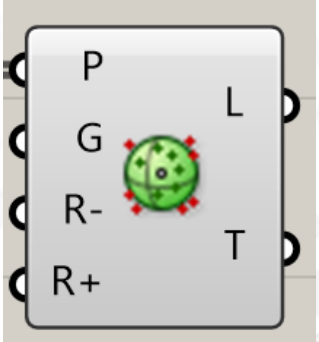
Voronoi Cell：根据点集生成 voronoip 多面体单元
输入端 P:输入的点集 P
输入端 N:点集 P 周围的点集
输入端 B:Box 范围
输出端 C: Voronoi 多面体单元



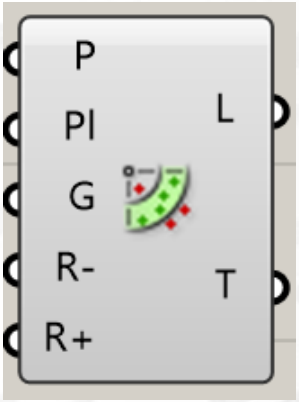
OcTree：根据点集 P 生成空间八分化网格
输入端 P:输入的点集 P
输入端 S:是否为立方体
输入端 G:每个网格单元做多包含的点数
输出端 B:各网格长方体
输出端 P:各长方体包含的点



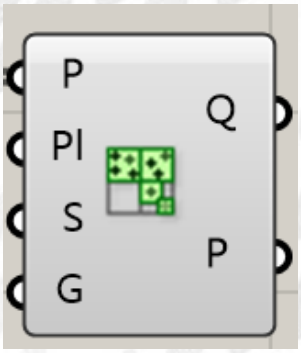
Proximity 3D：求点集 P 中与各点距离最近的 G 个点
输入端 P:输入的点集 P
输入端 G:距离 P 最近点个数 G（可能小于 G）
输入端 R-:搜寻半径下限
输入端 R+:搜寻半径上限
输出端 L:满足条件的顶点连线
输出端 T:各个顶点的拓扑关系



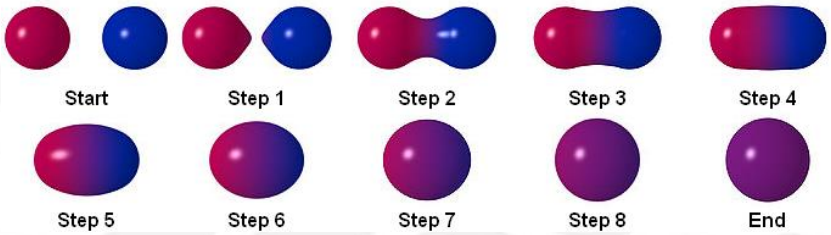
Proximity 2D：通过投影到工作平面 PI 计算，求点集 P 中与各点距离最近的 G 个点
输入端 P:输入的点集 P
输入端 PI:工作平面
输入端 G:距离 P 最近点个数 G（可能小于 G）
输入端 R-:搜寻半径下限
输入端 R+:搜寻半径上限
输出端 L:满足条件的顶点连线
输出端 T:各个顶点的拓扑关系



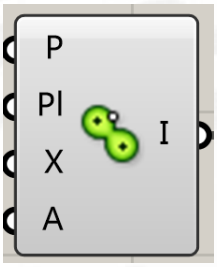
QuadTree：根据点集 P 生成平面四分化网格
输入端 P:输入的点集 P
输入端 PI:网格所在的工作平面
输入端 S:是否为正方形网格
输入端 G:每个网格单元做多包含的点数
输出端 Q: 各网格矩形
输出端 P: 各矩形包含的点（投影到 PI）



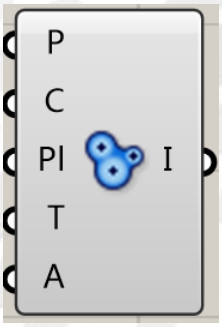
MetaBall 原理：数学模型——$f(x,y,z)=1/((x-x_0)^2+(y-y_0)^2+(z-z_0)^2)$



MetaBall：根据点集 P 求投影于 PI 平面的且经过点 X 的多段线
输入端 P:输入的点集 P
输入端 PI:工作平面
输入端 X:经过的点
输入端 A:采样精度(数值越小 Ploy line 越平滑)
输出端 I:输出的 Ploy line

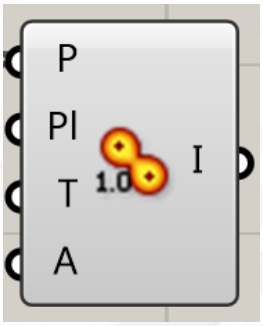


MetaBall (t) Custom：根据点集 P 求投影于 PI 平面的且经过点 X 的多段线
输入端 P:输入的点集 P
输入端 C:各点引力强度
输入端 PI:工作平面
输入端 T:各点强度权值(阈值)
输入端 A:采样精度(数值越小 Ploy line 越平滑)
输出端 I:输出的 Ploy line



MetaBall (t) Custom：根据点集 P 求投影于 PI 平面的且经过点 X 的多段线
输入端 P:输入的点集 P
输入端 PI:工作平面
输入端 T:各点强度权值(阈值)
输入端 A:采样精度(数值越小 Ploy line 越平滑)

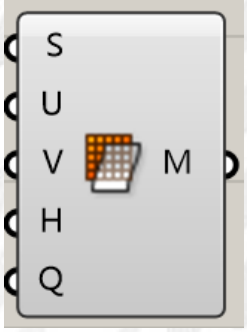
输出端 I:输出的 Ployline



（4）Util 电池序列

Mesh Brep：将 Brep 转换为网格
输入端 B:输入的 Brep
输入端 S:网格参数设定(Setting)
输出端 M:输出的网格

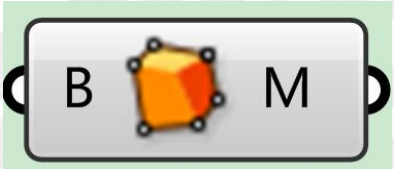
Mesh Surface：将 Surface 转换为网格
输入端 S:输入的 Surface
输入端 U/V:U/V 方向的细分
输入端 H:是否允许网格超过修建的边界
输入端 Q:是否平衡网格边长
输出端 M:输出的网格



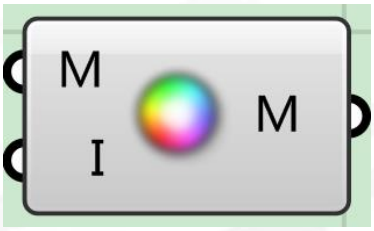
Setting (Custom)：网格参数
输入端 Stitch: 相邻面细分时是否匹配边数
输入端 Planes: 是否将平面分成最小数量的三角网格
输入端 Refine: 超出误差精度时是否优化网格
输入端 Min, Max: 每个面转为四边形网格时的最小、最大数量
输入端 Aspect: 四边形网格面的最大长宽比
输入端 Max Dist：边线中心与原始面的最大允许距离
输入端 Max Angle: 两个相邻四边形网格面法线的最大允许角度
输入端 Min Edge, Max Edge: 网格边线的最小、最大允许长度
输出端 S：网格参数

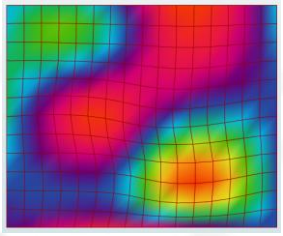
Setting (Speed), Setting (Quality)：预定义的参数设置：粗糙快速、平滑
输出端 S：网格参数

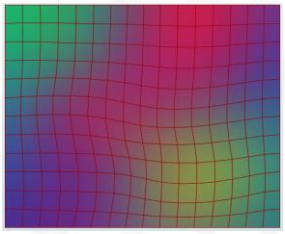
Simple Mesh：简化网格
输入端 B：输入的 Brep
输出端 M：输出简化后的网格



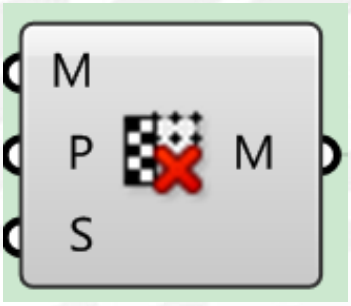
Blur Mesh：模糊网格
输入端 M：输入的网格
输入端 I：模糊的次数
输出端 M：模糊后的网格



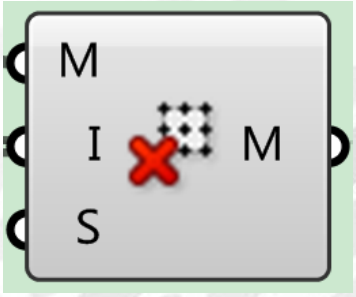




Cull Vertices：按照 Pattern P 删除顶点
输入端 M：输入的网格
输入端 P：选择条件（Boolean）
输入端 S：是否将四边形转为三角形
输出端 M：输出的网格



Delete Vertices：按照序号删除顶点
输入端 M：输入的网格
输入端 I：顶点序号
输入端 S：是否将四边形转为三角形
输出端 M：输出的网格



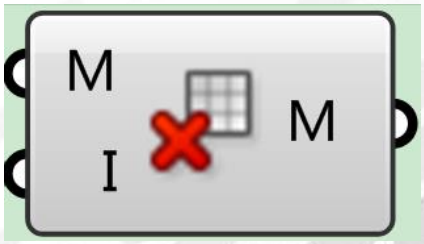
Mesh Join/ Disjoint Mesh：合并/分解网格
输入端 M：输入的网格
输出端 M：输出的网格

Mesh Split Plane：用工作平面 P 切开网格
输入端 M：输入的网格
输入端 P：工作平面

输出端 A：位于平面上方的网格
输出端 B：位于平面下方的网格

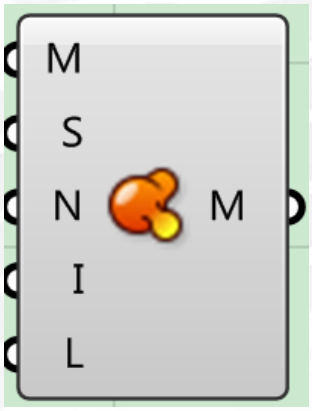
Cull Faces：按照 Pattern P 删除面
输入端 M：输入的网格
输入端 P：选择条件（Boolean）
输出端 M：输出的网格

Delete Faces：按照序号删除面
输入端 M：输入的网格
输入端 I：网格面序号
输出端 M：输出的网格

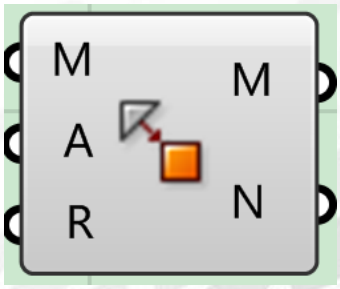


Mesh Shadow：计算网格在光线（向量）L 下在工作平面 P 上的阴影轮廓输入端
输入端 M：输入的网格
输入端 L：光线向量
输入端 P：工作平面
输出端 M：输出的网格

Smooth Mesh：平滑网格
输入端 M：输入的网格
输入端 S：平滑强度
输入端 N：是否忽略裸露的顶点
输入端 I：执行次数
输入端 L：顶点位移最大值
输出端 M：平滑后的网格



Quadrangulate：将网格的三角形面尽可能合并成四边形面
输入端 M：输入的网格
输入端 A：三角形进行转换的最大角度
输入端 R：四边形最大长宽比
输出端 M：输出的网格
输出端 N：转化的三角形数量

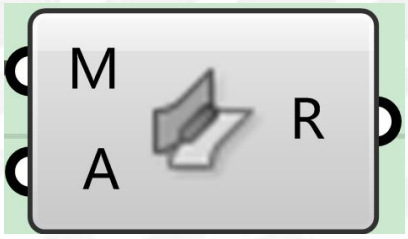


Quadrangulate：将网格的四边形面转化为三角形面
输入端 M：输入的网格
输出端 M：输出的网格
输出端 N：转化的四边形数量

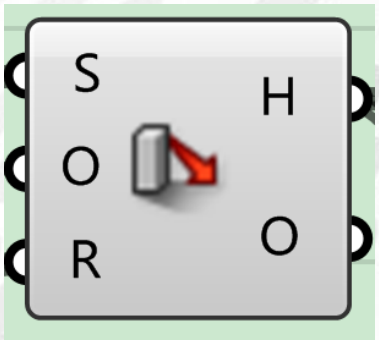
Weld Mesh：焊接网格

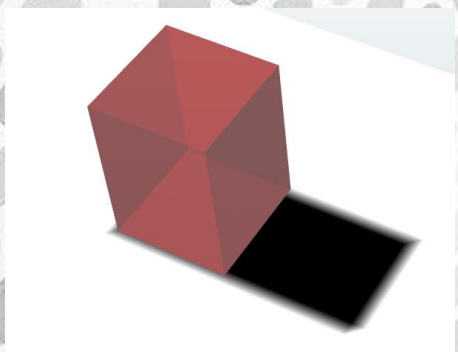
输入端 M：输入的网格
输入端 A：焊接相邻网格面的最大角度
输出端 N：焊接后的网格

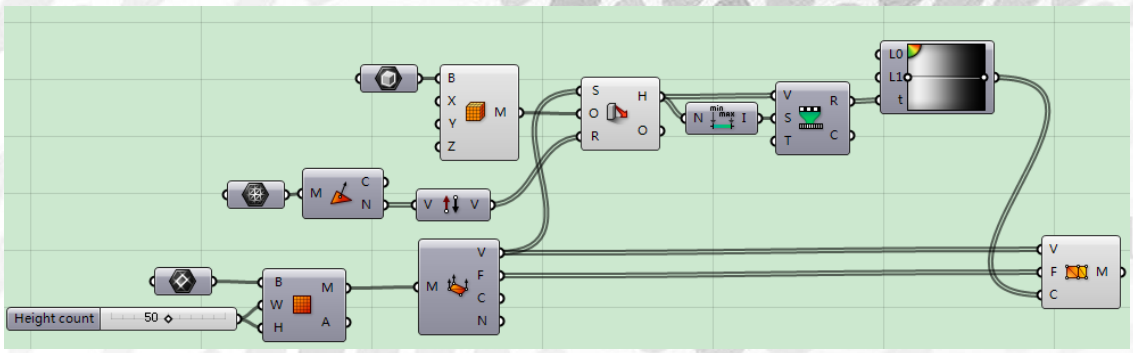
Weld Mesh：取消焊接网格
输入端 M：输入的网格
输入端 A：取消焊接相邻网格面的最大角度
输出端 N：取消焊接后的网格



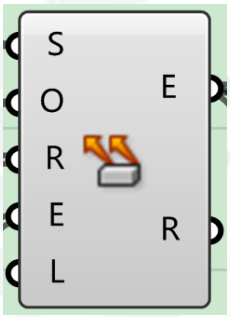
Occlusion：点采样投影
输入端 S：采样点
输入端 O：遮挡物体
输入端 R：光线向量
输出端 H：每个采样点的光通量
输出端 O：采样点拓扑关系

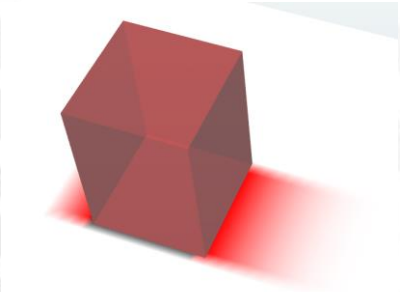


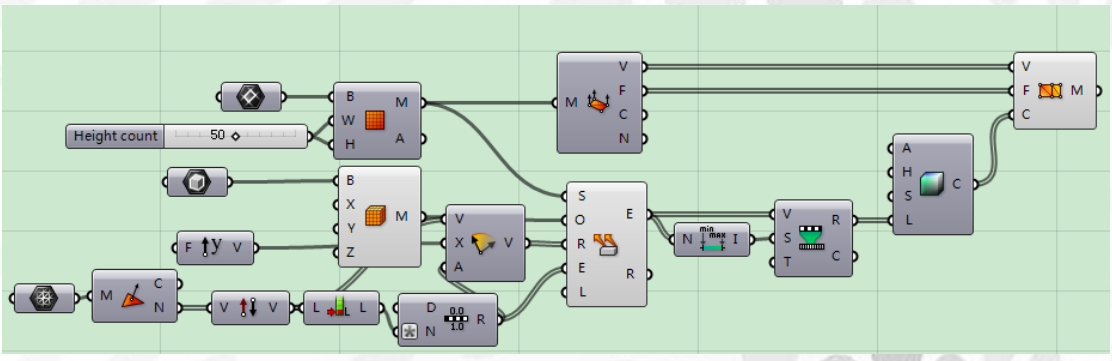




Exposure：网格采样投影(曝光)
输入端 S：采样网格
输入端 O：遮挡物体
输入端 R：光线向量
输入端 E：光线能量值
输入端 L：是否开启 Lambertian 反射
输出端 E：每个网格顶点的曝光量
输出端 R：整个网格曝光量的值域

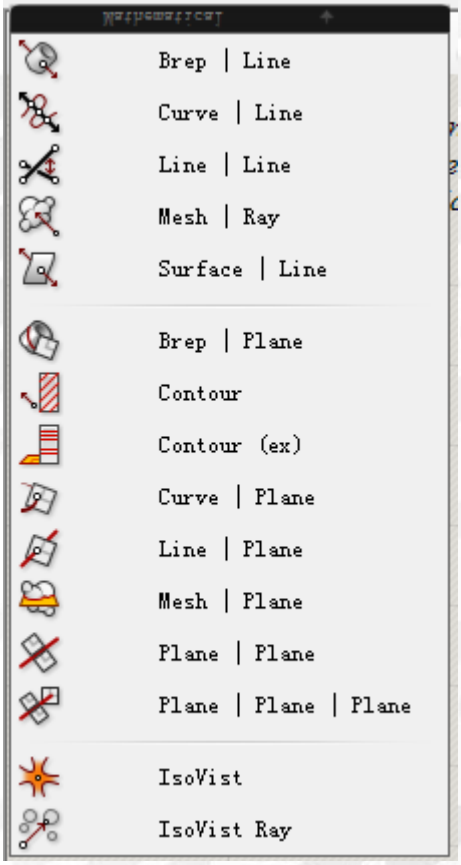




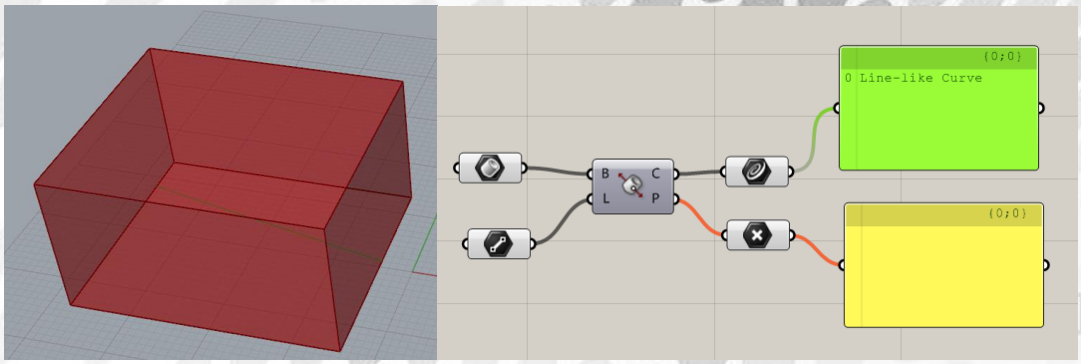


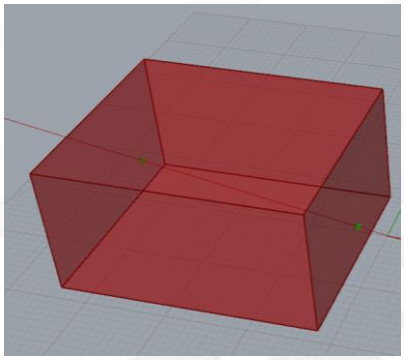
8. Intersection 电池组

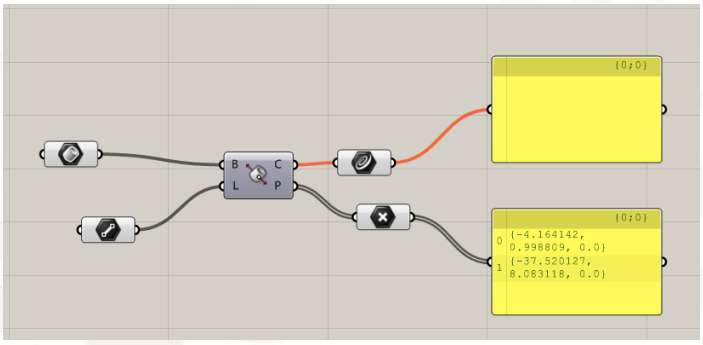
（1）Mathematical 电池序列



Brep/Line(BLX) 实体与线相交
输入端 B：物体
输入端 L：与之相交的线
输出端 C：物体与线相交的一部分线段（当线和物体的面相交时候就会产生线段
输出端 P：相交产生的点

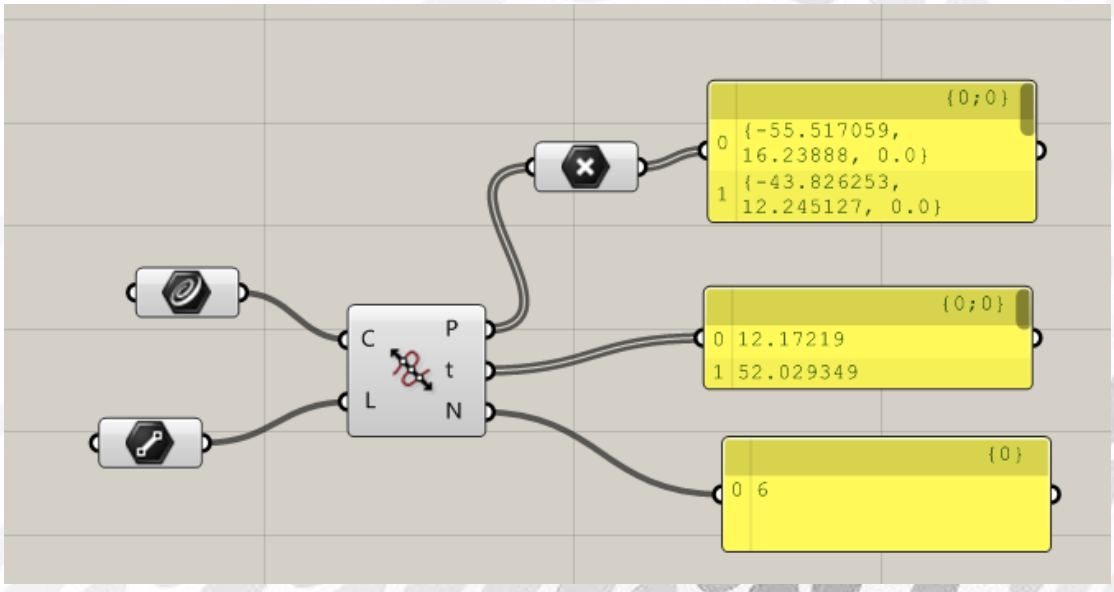




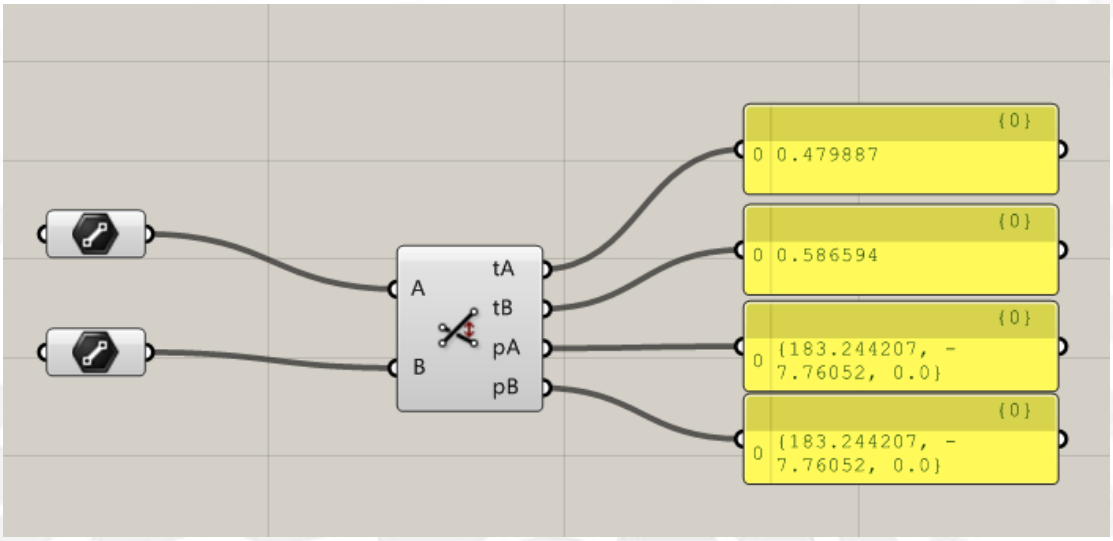


Curve/Line（CLX）：直线与曲线相交
输入端 C：曲线
输入端 L：直线

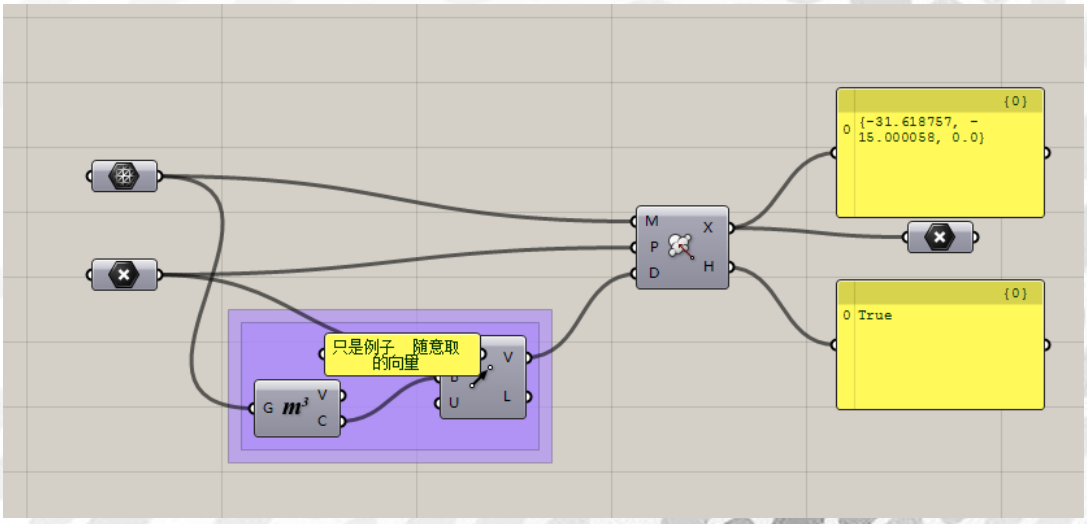
输出端 P：相交的点
输出端 t：相交点在曲线上的评估参数
输出端 E：相交点的个数



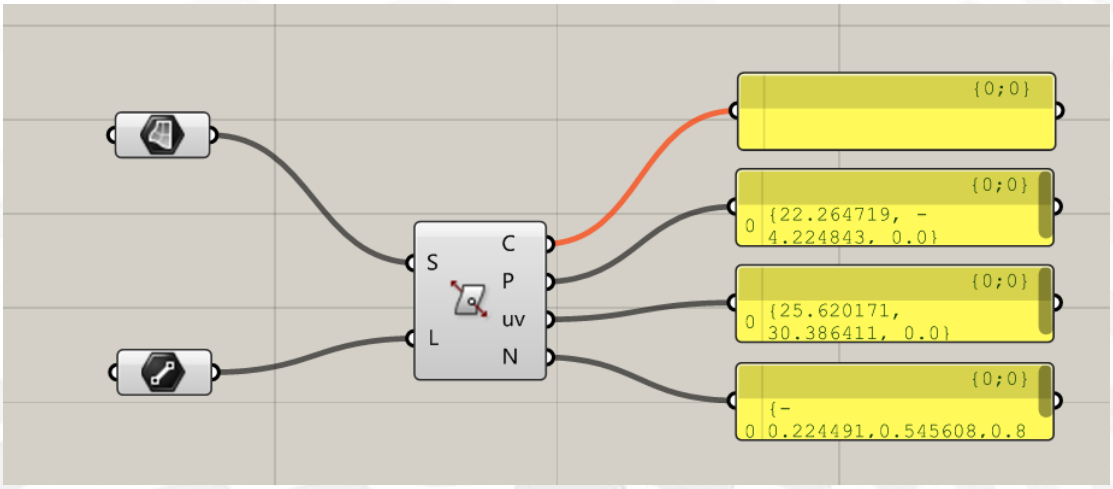
Line/Line(LXL)
输入端 A：直线
输入端 B：直线
输出端 tA：相交的点在直线 A 上的评估参数
输出端 tB：相交的点在直线 B 上的评估参数
输出端 pA：在直线 A 上的点
输出端 pB：在直线 B 上的点



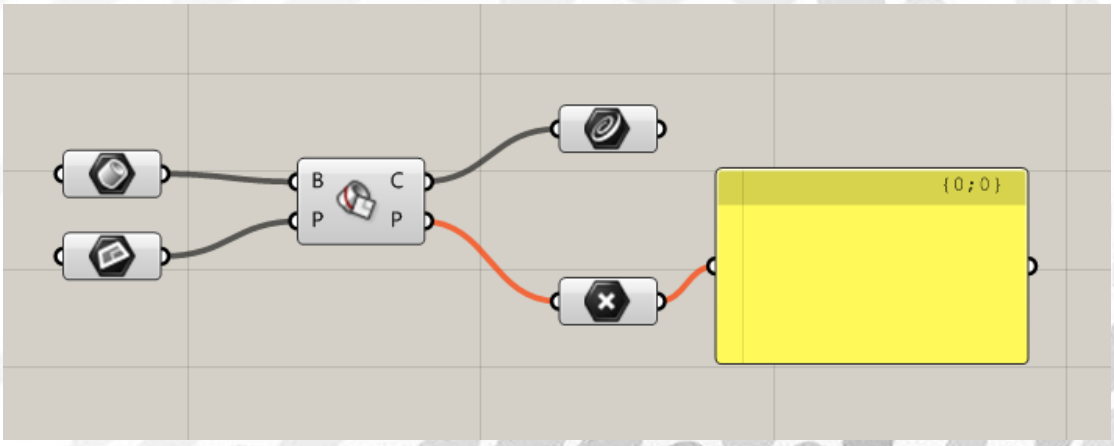
Mesh/Ray:用单放线无线的射线与网格的交集
输入端 M：网格
输入端 P：射线开始的点
输入端 D：射线的放线
输出端 x：首次射线与网格相交的点
输出端 H：是否射线与网格布尔成功



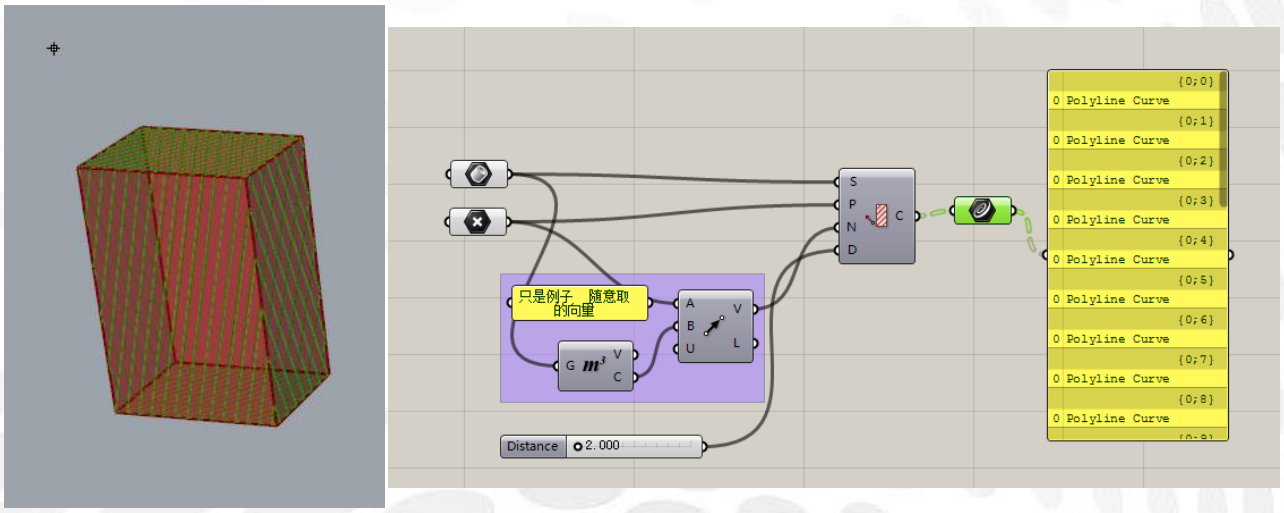
Surface/Line(SCX):线与面相交
输入端 S：面
输入端 C：曲线
输出端 C：相交曲线（当存在相交线时）
输出端 P：相交点
输出端 UV：相交部分的曲面的 UV 坐标
输出端 N：相交部分在曲面标准向量



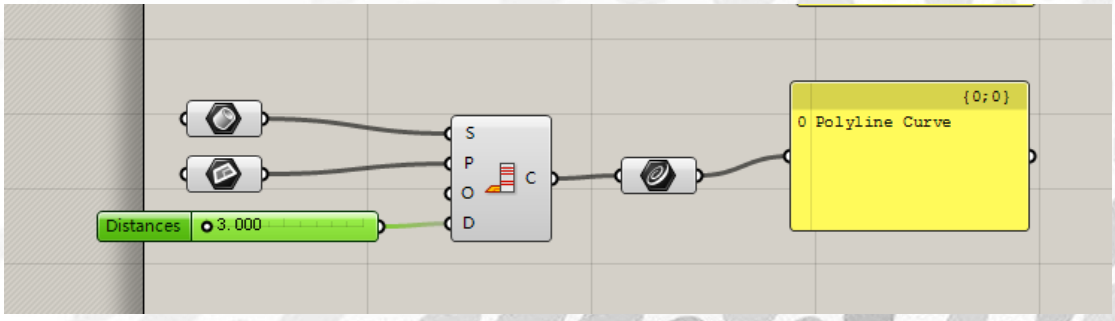
Brep/Plane
输入端 B：物体
输入端 P：平面
输出端 C：相交产生的曲线
输出端 P：相交产生点



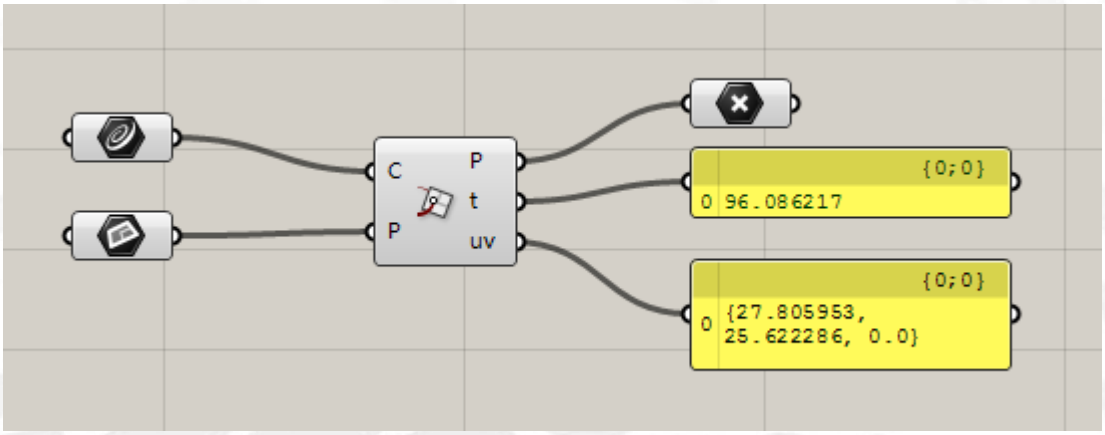
Contour：产生一系列的等位线
输入端 S：实体或网格
输入端 P：起始点
输入端 N：起始的标准方向
输入端 D:轮廓线的距离
输出端 C：曲线



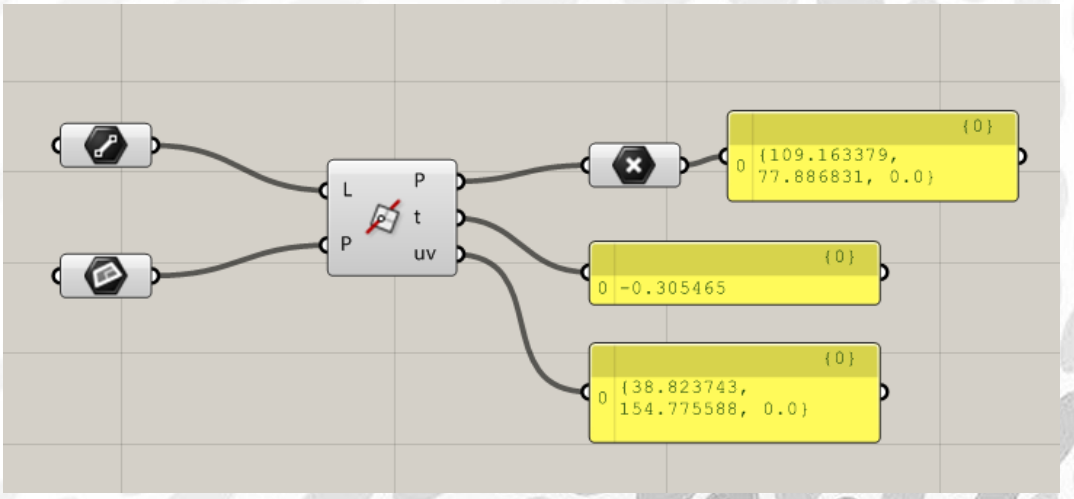
Contour（ex）：产生一系列的轮廓线
输入端 S：物体或网格
输入端 P：基础平面
输入端 O:基于基础平面偏移产生的一系轮廓线(如果不输入数值，你定输入特别的距离代替)
输入端 D:轮廓线的距离（如何不输入数值，你一定要输入特别的偏移距离代替）
输出端 C：曲线



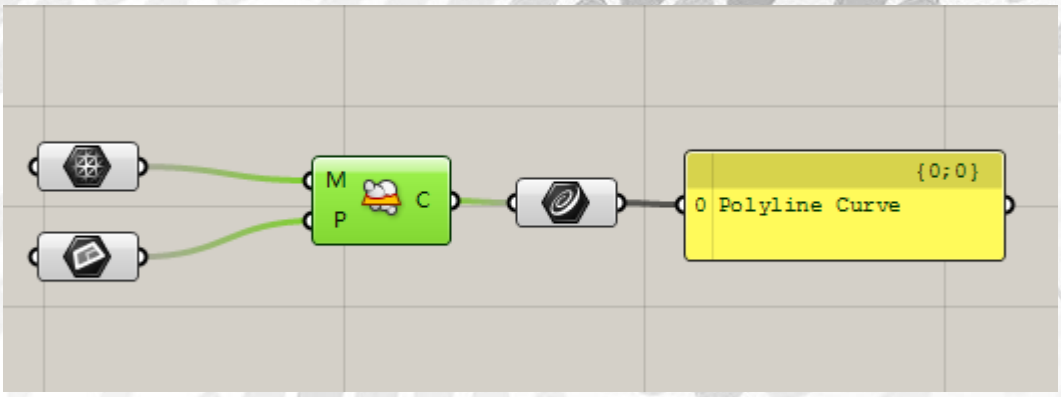
Curve/Plane:曲线余平面的交点
输入端 C:曲线
输入端 P：平面
输出端 P：曲线与平面的交点
输出端 t：相交的点在曲线上的评估参数
输出端 UV：相交部分在平面上的 UV 坐标



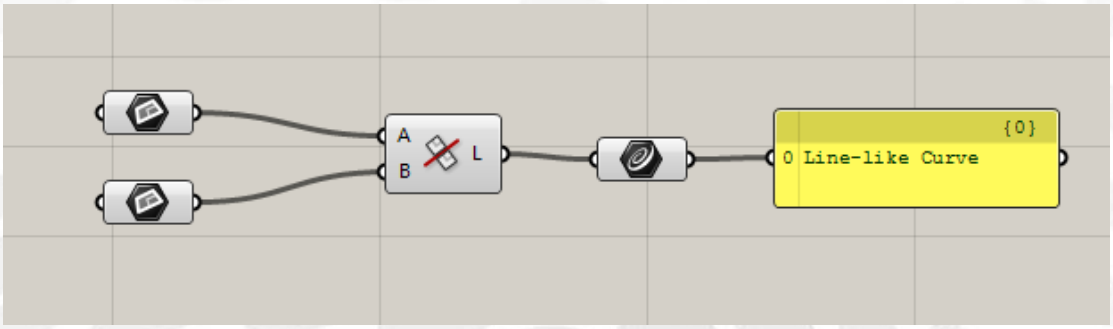
Line/Plane:线与平面的交点
输入端 L：直线
输入端 P：平面
输出端 P：相交的点
输出端 t：相交的点在曲线上的评估参数
输出端 UV：相交部分在平面上的 UV 坐标



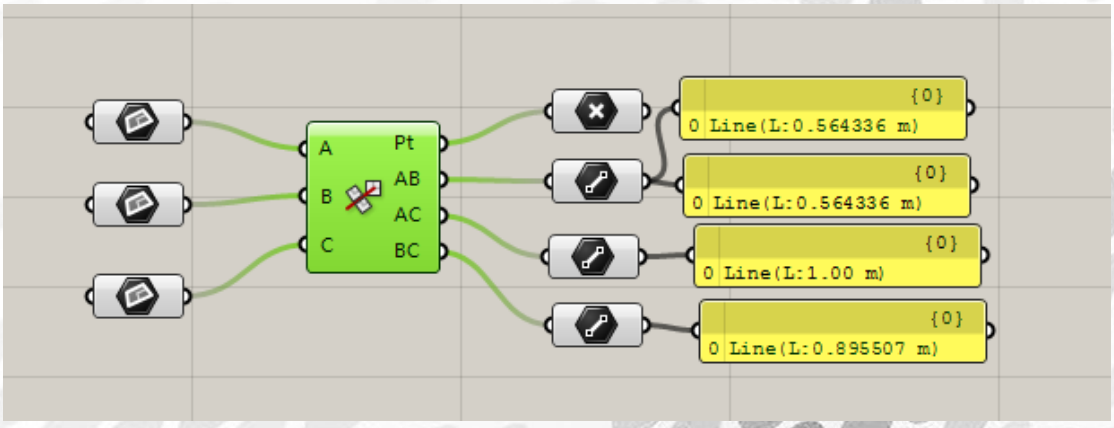
Mesh/Plane:网格和平面的交线
输入端 M：网格
输入端 P：平面
输出端 C：曲线



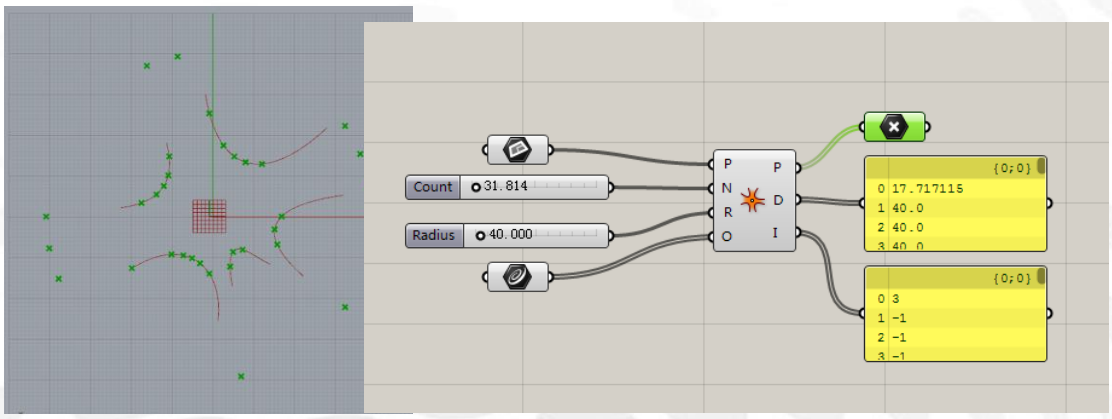
Plane/Plane:平面与平面相交得的交集
输入端 A：第一个平面
输入端 B：第二个平面
输出端 L：2 个平面相交得到的线



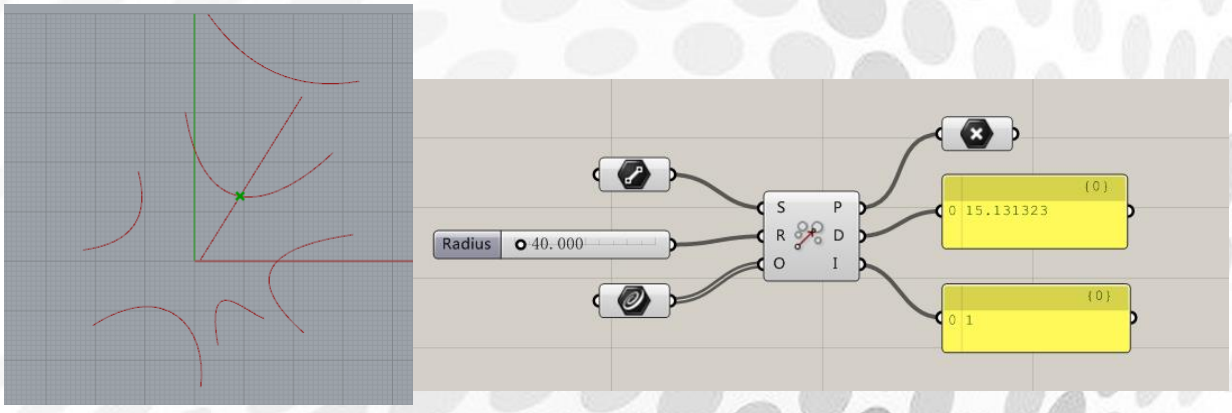
Plan/Plane/Plane：3 个平面所产生的交集
输入端 A：第一个平面
输入端 B：第二个平面
输入端 C：第三个平面
输出端 Pt：3 个平面相交所得的点
输入端 AB：平面 A 和平面 B 相交所得的直线
输入端 AC：平面 A 和平面 C 相交所得的直线
输入端 BC：平面 B 和平面 C 相交所得的直线



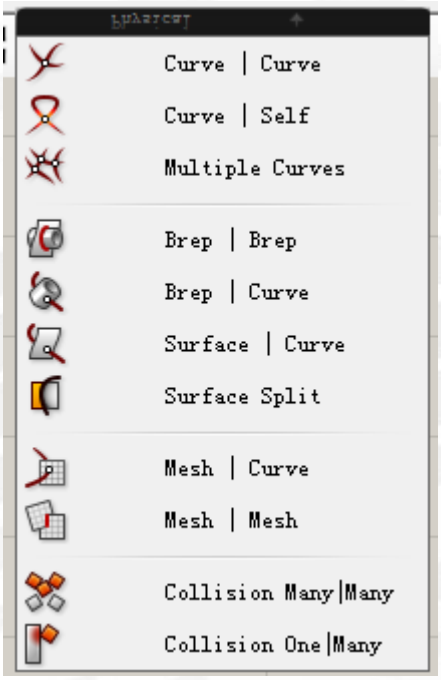
IsoVist:产生一系列的索亚取样位置
输入端 P：平面
输入端 N：样点的个数
输入端 R：范围
输入端 O：阻碍物体的轮廓
输出端 P：射线与阻碍物的交点
输出端 N：个个点与平面原点的距离
输出端 I：得到的每个输出的指数列表或没有障碍（简单的说就是结果的代码，-1 表示没有交点，0 表示与第一条线相交，1 表示于第二天线有交点，以此类推）



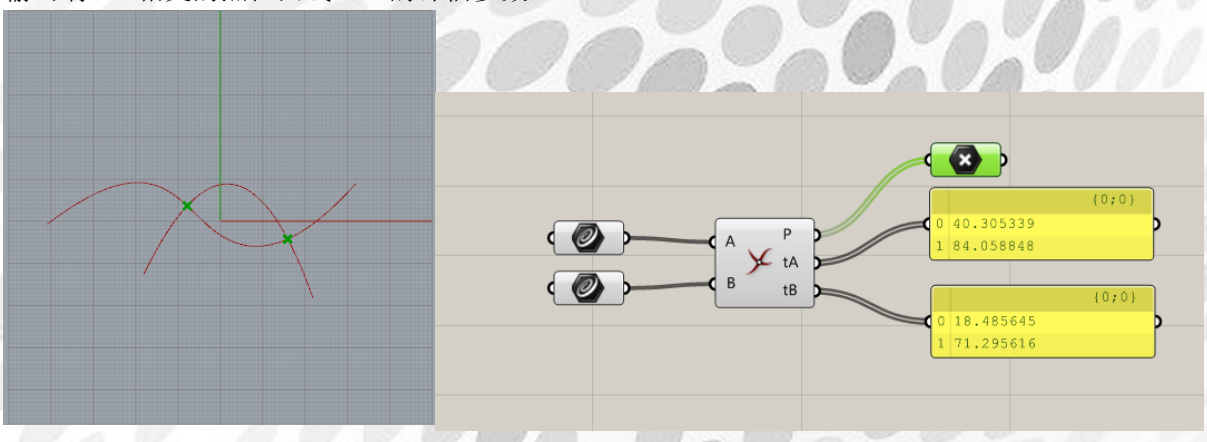
IsoVist Ray：产生一个索亚取样位置
输入端 S：线
输入端 R：范围
输入端 O：障碍物
输出端 P：直线与障碍物点交点，如不交着为最大范围上的点
输出端 D：从射线的起始点到交点的距离
输出端 I：得到的输出的指数列表或没有障碍



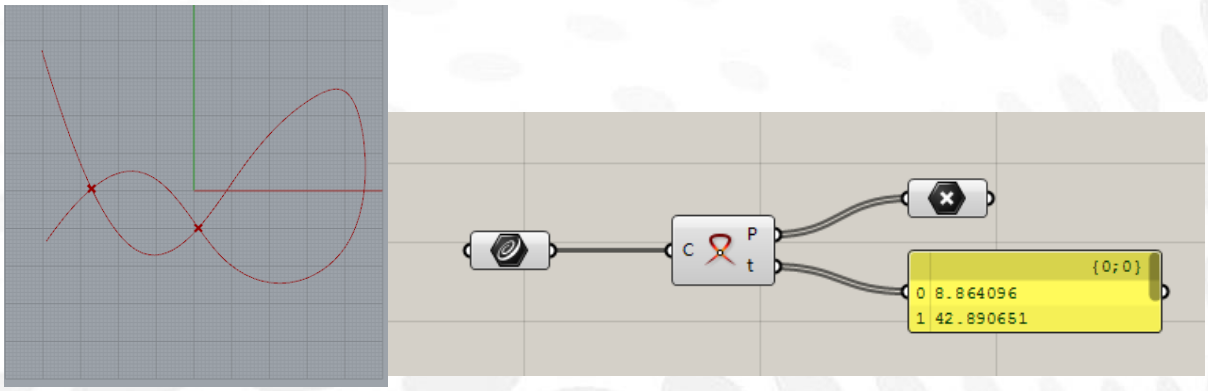
（2）Physical 电池序列



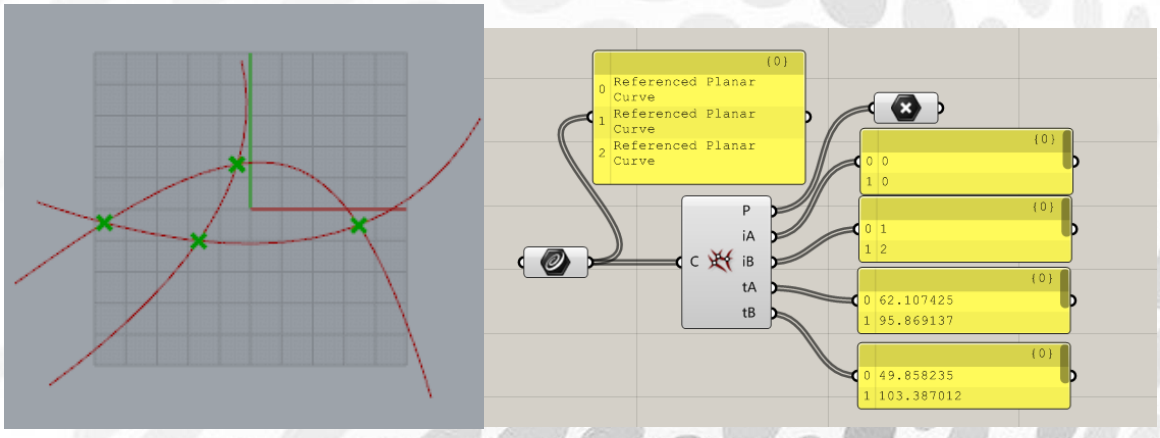
Curve/Curve:曲线与曲线的交点
输入端 A：第一个曲线
输入端 B：第二个曲线
输出端 P：曲线与曲线相交的点
输出端 A：相交的点在曲线 A 上的评估参数
输出端 B：相交的点在曲线 B 上的评估参数



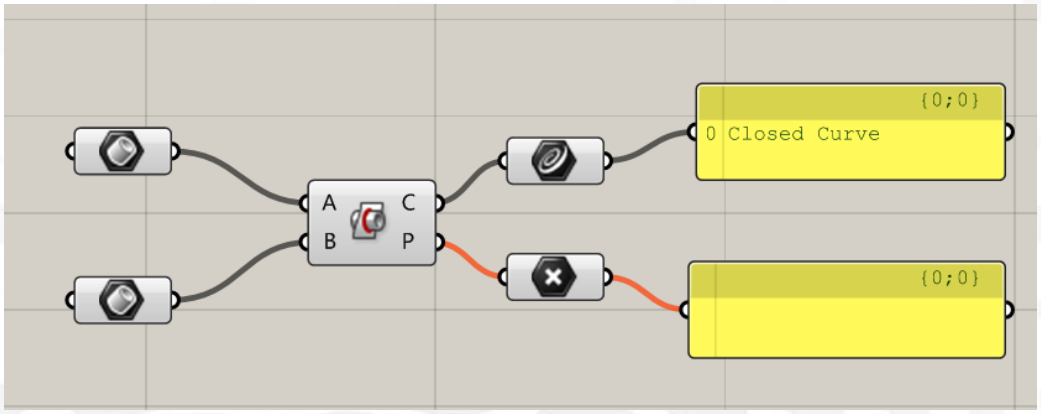
Curve/Self:曲线自身的交点
输入端 AB：曲线
输出端 P：交点
输出端 t：相交的点在曲线上的评估参数



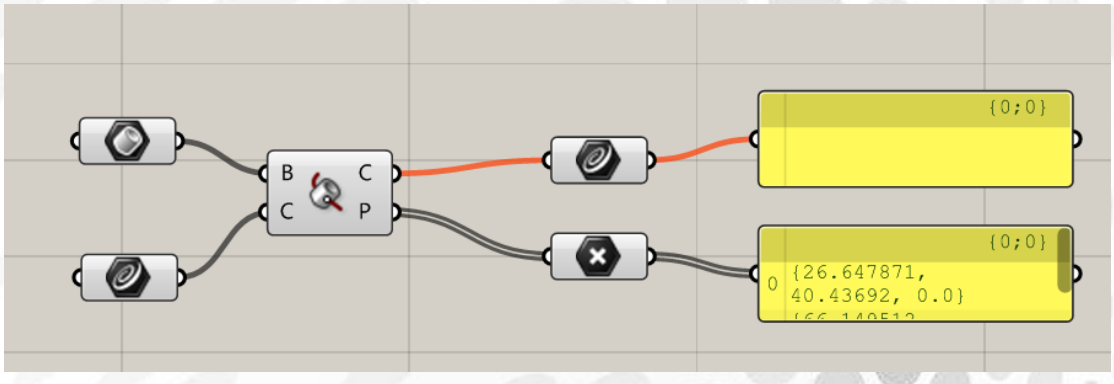
Multiple Curve：多个曲线相交
输入端 C：多个曲线
输出端 P：相交的点
输出端 Ia：得到的第一个相交曲线的输出的指数列表
输出端 Ib：得到的第二个相交曲线的输出的指数列表
输出端 tA：相交的点在第一个相交曲线上的评估参数
输出端 tB：相交的点在第二个相交曲线上的评估参数



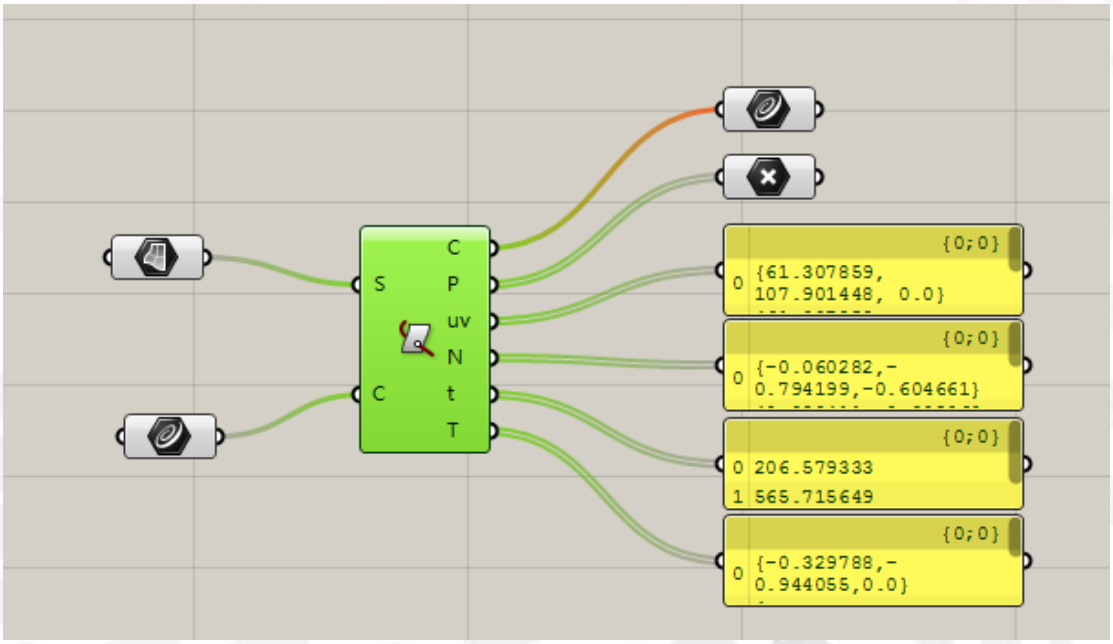
Brep/Brep:实体与物体的交点
输入端 A:第一个物体
输入端 B:第二个物体
输出端 C:相交的曲线
输出端 P:相交的点



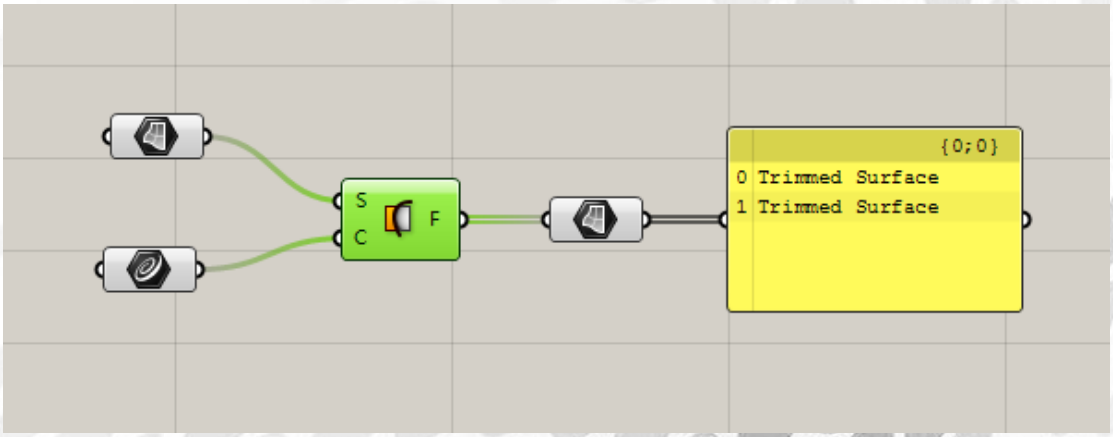
Brep/Curve:实体与曲线的交点
输入端 B: 物体
输入端 C:曲线
输出端 C:相交的曲线
输出端 P:相交的点



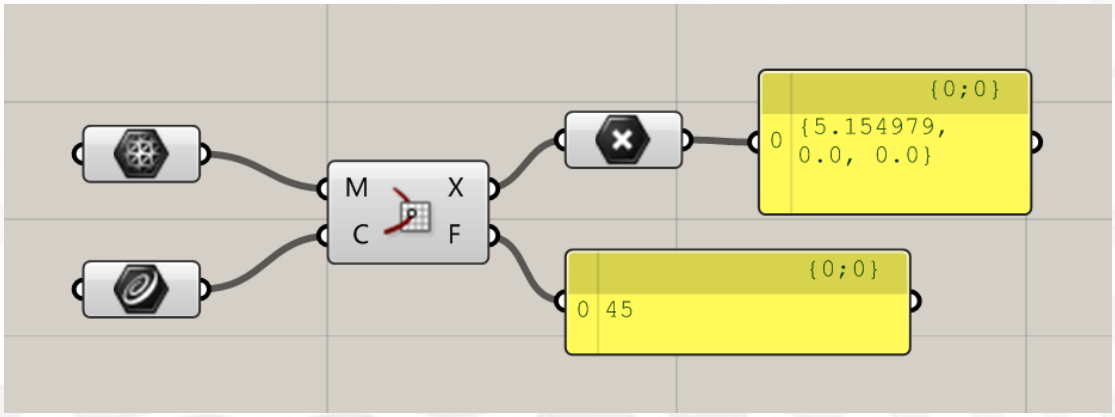
Surface/Curve:直线与曲面相交的点
输入端 S：面
输入端 P:曲线
输出端 C：相交的曲线
输出端 B：相交的点
输出端 UV：相交的点在平面上的 UV 坐标
输出端 N：相交的点在平面上的法向量
输出端 t：相交的点在曲线上的评估参数
输入端 T：相交的点在曲线上的切线向量



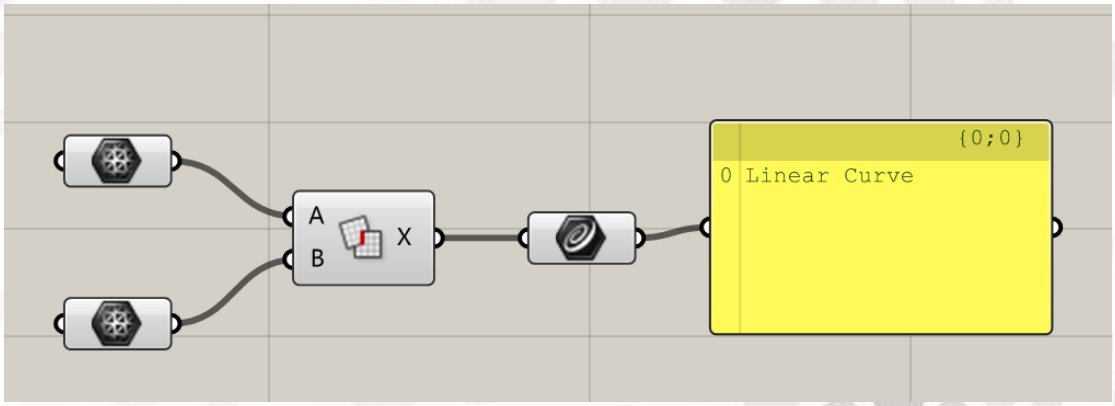
Surface Split：曲面分割
输入端 S：曲面
输入端 C：切割曲线
输出端 F：分离的部分



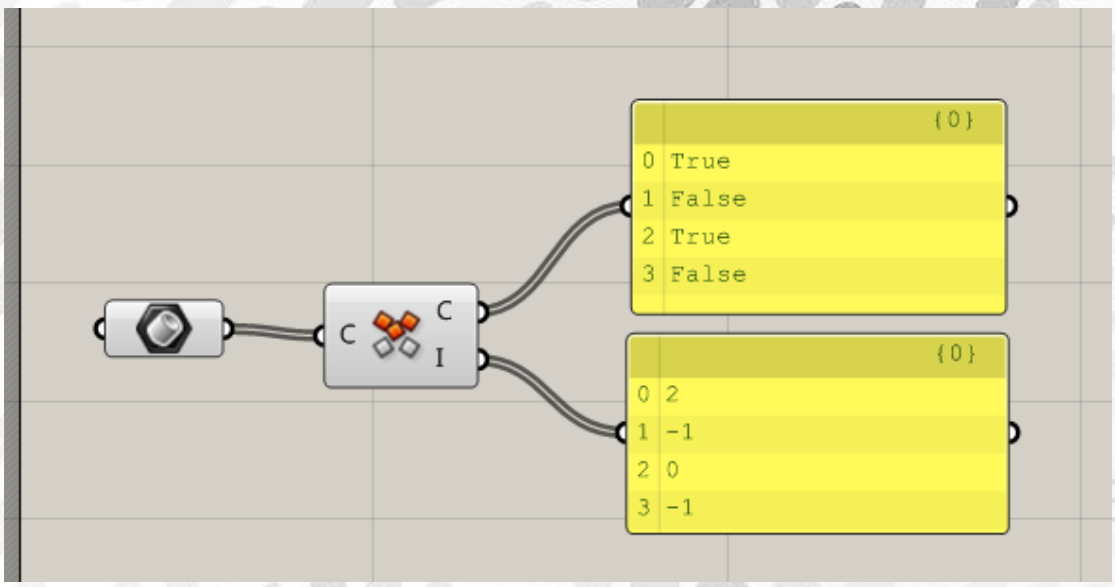
Mesh/Curve
输入端 M：网格
输入端 C：曲线
输出端 X：相交的点
输出端 F：每个相交的点在网格上的指数



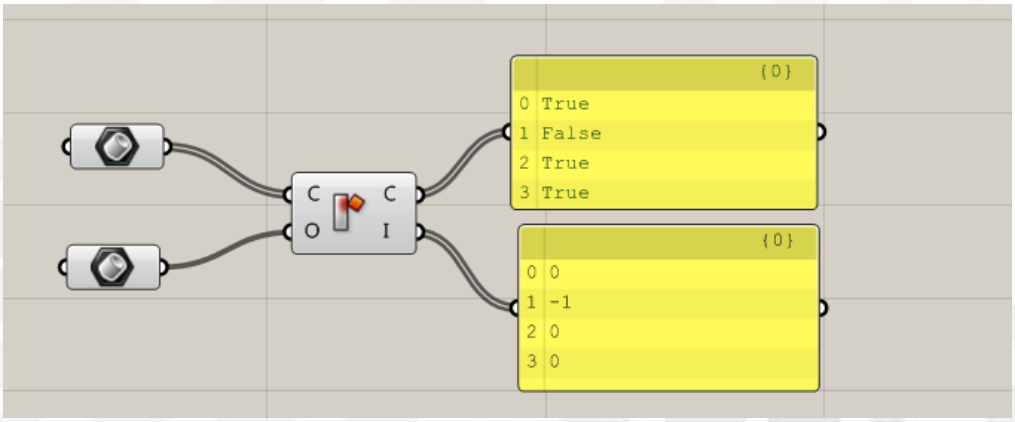
Mesh/Mesh：网格与网格的交集
输入端 M:第一个网格
输入端 M:第二个网格
输出端 X：相交的线



Collision Many/Many:许多物体的碰撞
输入端 C：物体
输出端 C：物体是否碰撞，以 true 或者 false 显示
输出端 I：物体的碰撞，以指数的形式表达

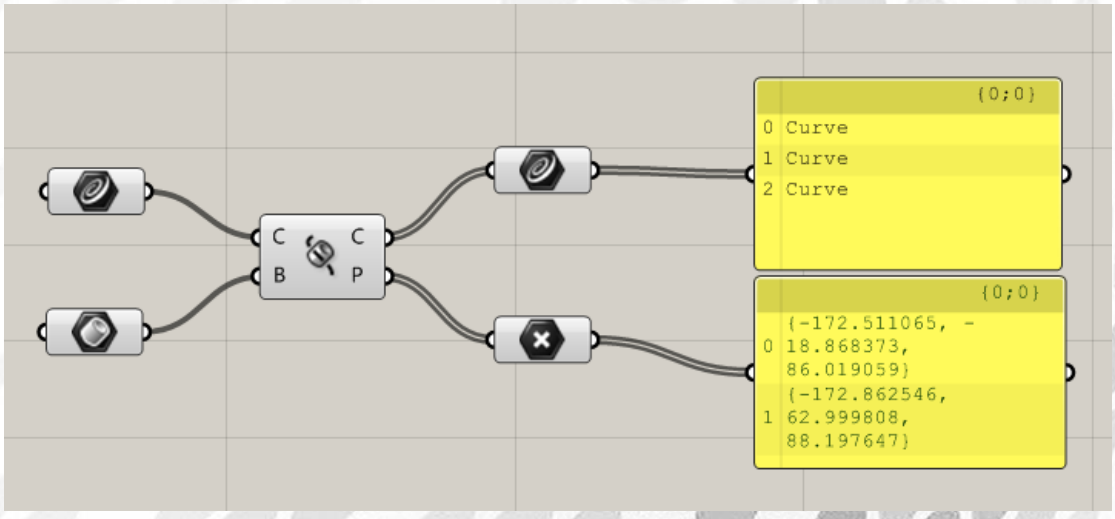


Collision One/Many：物体的碰撞一个或着多个
输入端 C：测试碰撞的物体
输入端 O：阻碍碰撞的物体
输出端 C：在很多阻碍物的情况下，物体是否碰撞以 ture 或 false 显示
输出端 I：在很多阻碍物的情况下，物体的碰撞以指数的形式表达

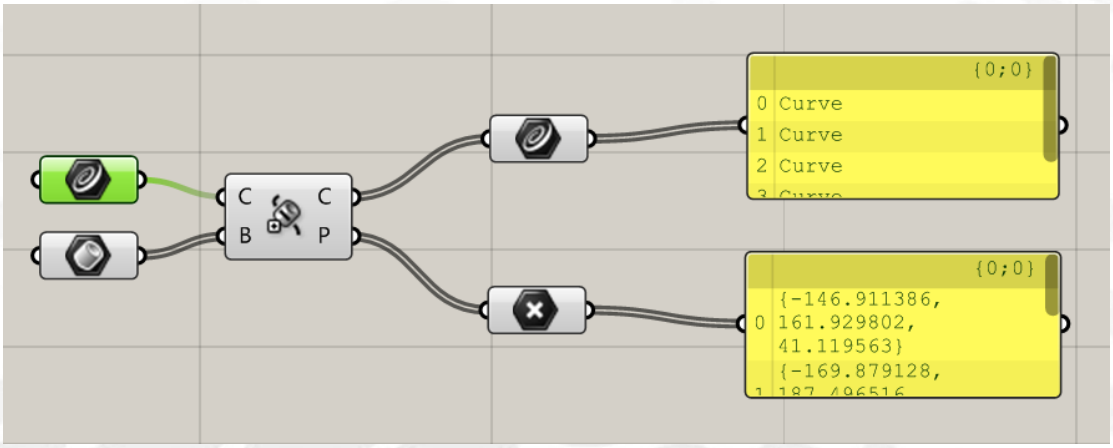


（3）Region 电池序列

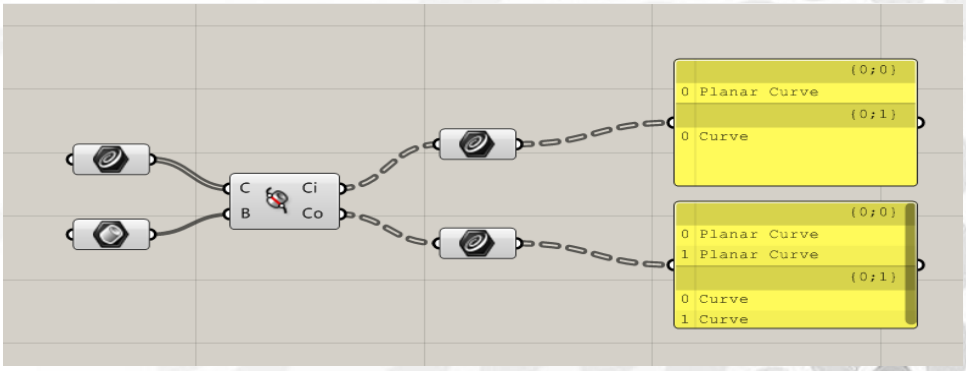
Split with Brep:曲线被一个实体切分
输入端 C：被切割的曲线
输入端 B：切割用的物体
输出端 C：被物体切割后的 N 个线段
输出端 P：曲线与物体的交点



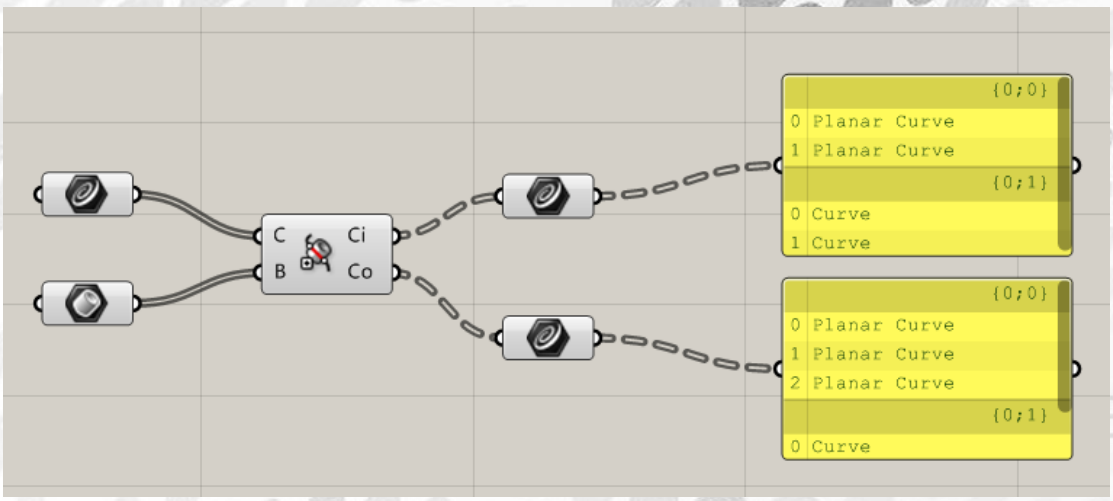
Split with Brep:曲线被一个物体切分
输入端 C：被切割的曲线
输入端 B：切割用的多个物体
输出端 C：被物体切割后的 N 个线段
输出端 P：曲线与物体的交点



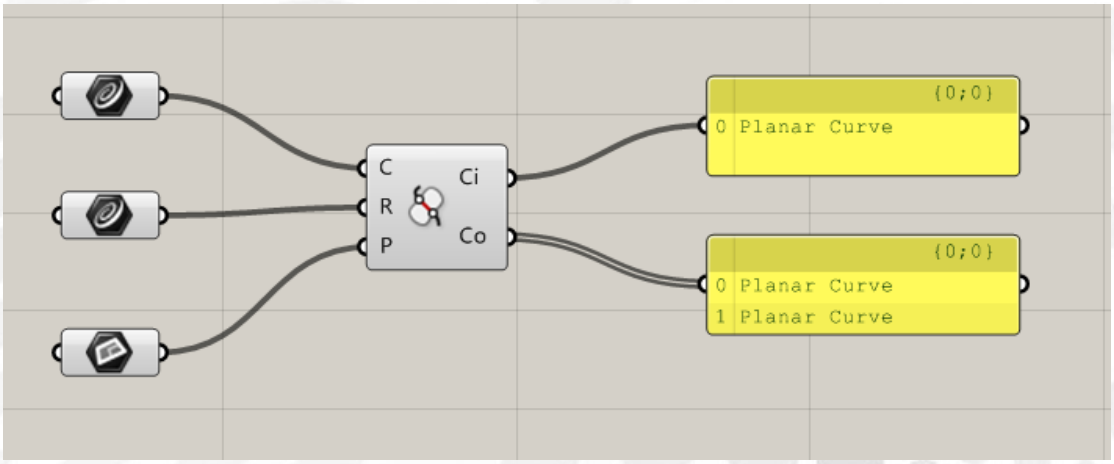
Trim with Brep：曲面被一个物体切割分成内外 2 部分
输入端 C:被切割曲线
输入端 B：切割用物体
输出端 Ci：被物体切割后的实体外的曲线
输出段 Co：被物体切割后的实体内的曲线



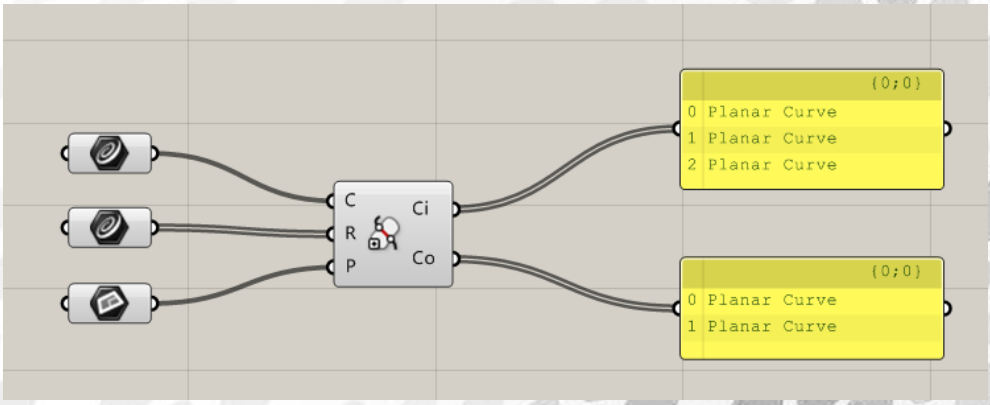
Trim with Brep：曲面被多个物体切割分成内外 2 部分
输入端 C:被切割曲线
输入端 B：切割用物体
输出端 Ci：被实体切割后的物体外的曲线
输出段 Co：被实体切割后的物体内的曲线



Trim with Region：曲面被一个区域切割分成内外 2 部分
输入端 C:被切割曲线
输入端 B：切割用区域
输入端 P：平面
输出端 Ci：被区域切割后的物体外的曲线
输出段 Co：被区域切割后的物体内的曲线

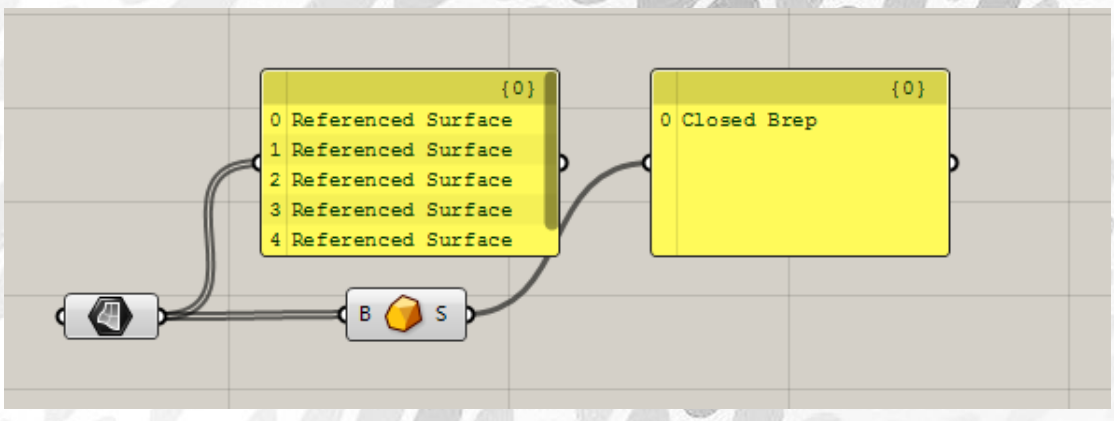


Trim with Region：曲面被多个区域切割分成内外 2 部分
输入端 C:被切割曲线
输入端 B：切割用区域
输入端 P：平面
输出端 Ci：被区域切割后的物体外的曲线
输出段 Co：被区域切割后的物体内的曲线



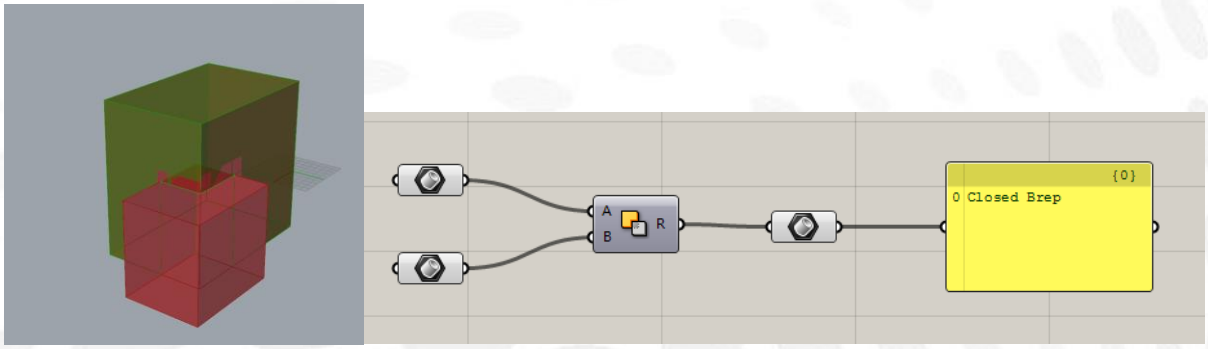
（4）Shape 电池序列

Boundary Volume:通过边界表面创建一个封闭的实体
输入端 B：边界表面
输出端 S：一个有体积的固体

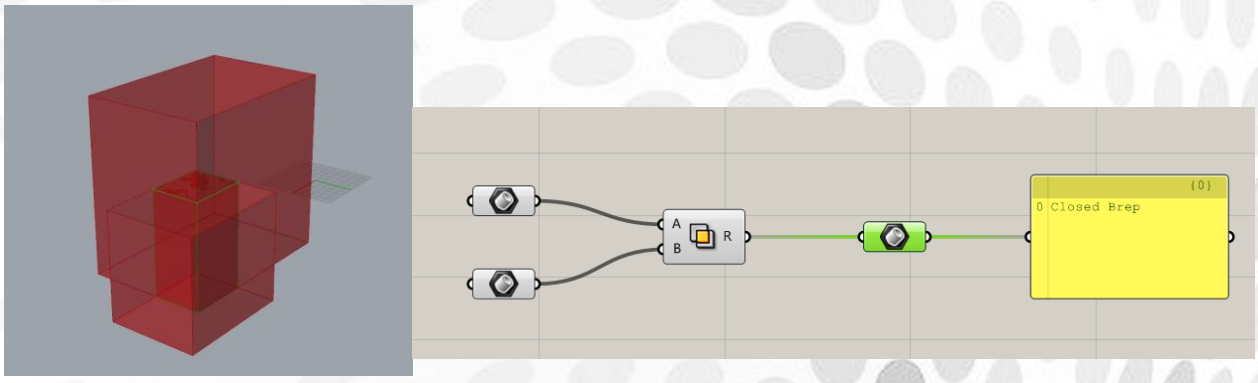


Solid Difference：物体 B 去切割物体 A
输入端 A：第一个物体
输入端 B：第二个物体

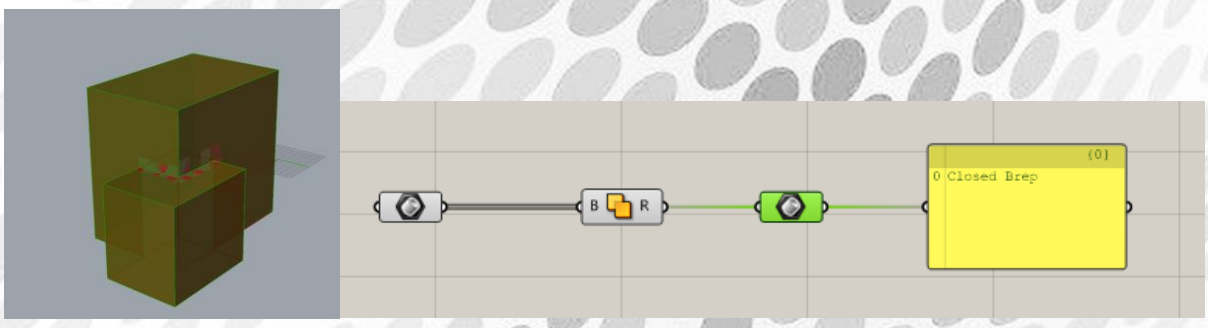
输出端 R：切割后的物体，含切面



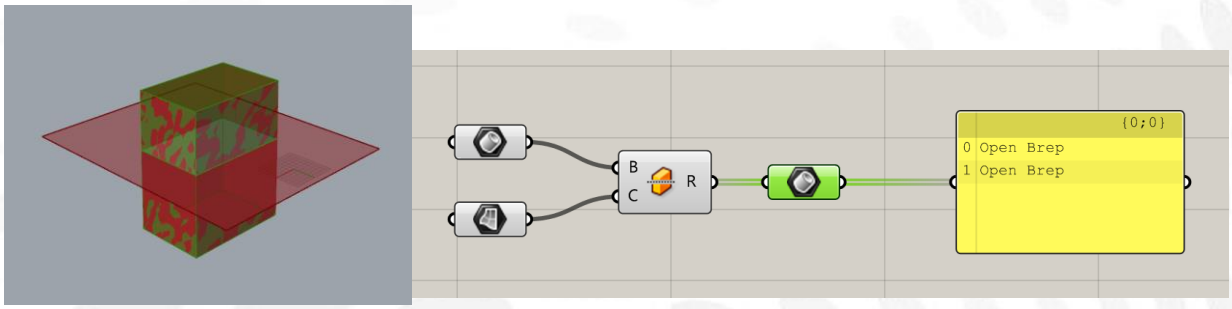
Solid Intersection：2 个物体相同的地方
输入端 A：第一个物体
输入端 B：第二个物体
输出端 R：相同的地方



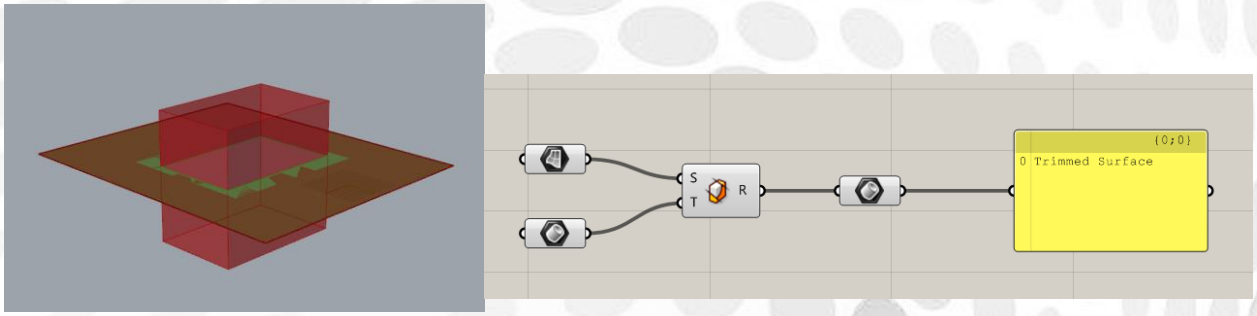
Solid Union：多个物体的并集
输入端 A：第一个物体
输入端 B：第二个物体
输出端 R：相同的地方



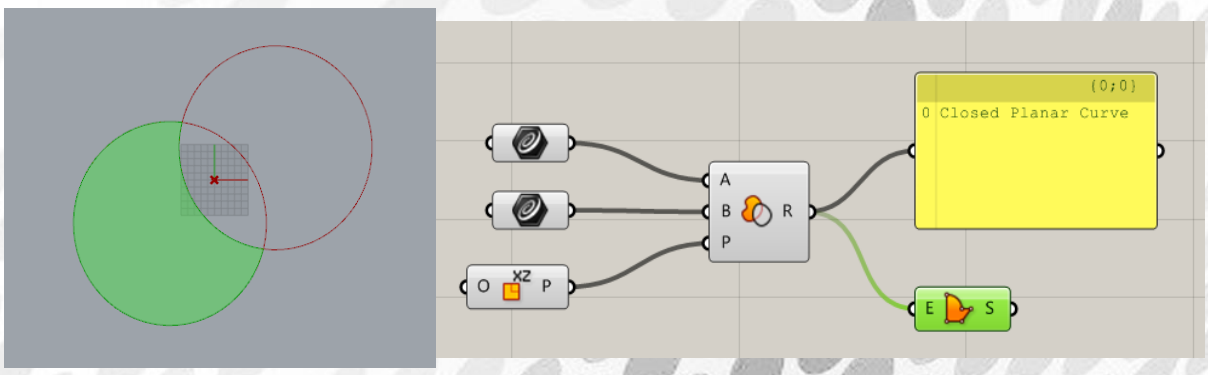
Split Brep：用实体去切割物体
输入端 B：切割用实体
输入端 C：被切割的物体（可以为线，平面，物体）
输出端 R：切割后所得的物体



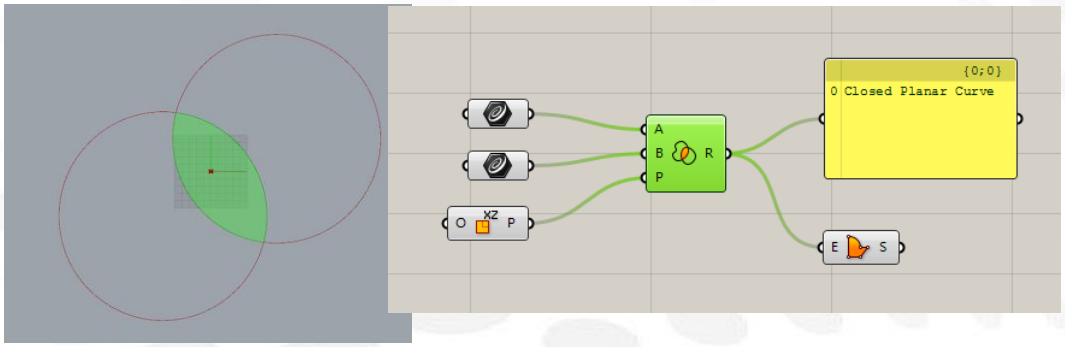
Trim solid：用一些实体去剪切形状
输入端 S：被切割的形状
输入端 T：切割用实体
输出端 R：被切割后得到的形状



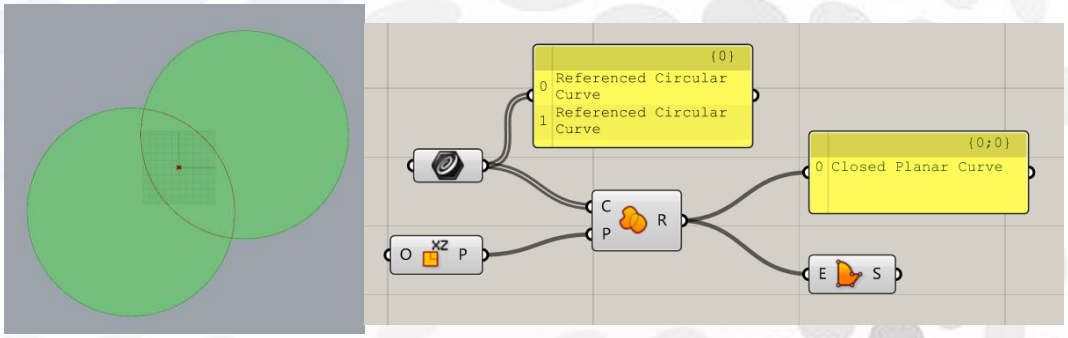
Region Difference：曲线（区域）B 分割曲线 A
输入端 A：闭合曲线（区域）A
输入端 B：闭合曲线（区域）B
输出端 P：2 条曲线投影平面
输出端 R：分割后的曲线或区域，含切边



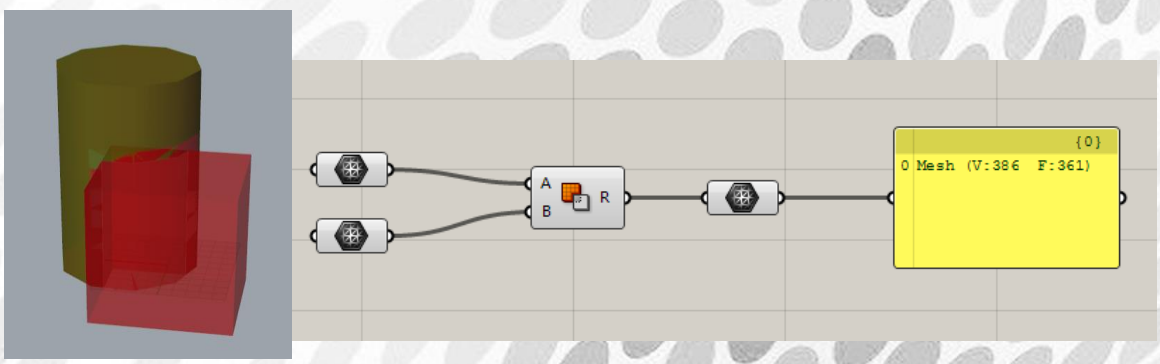
Region Intersection：曲线（区域）之间交集
输入端 A：闭合曲线（区域）A
输入端 B：闭合曲线（区域）B
输出端 P：2 条曲线投影平面
输出端 R：交集后的曲线或区域



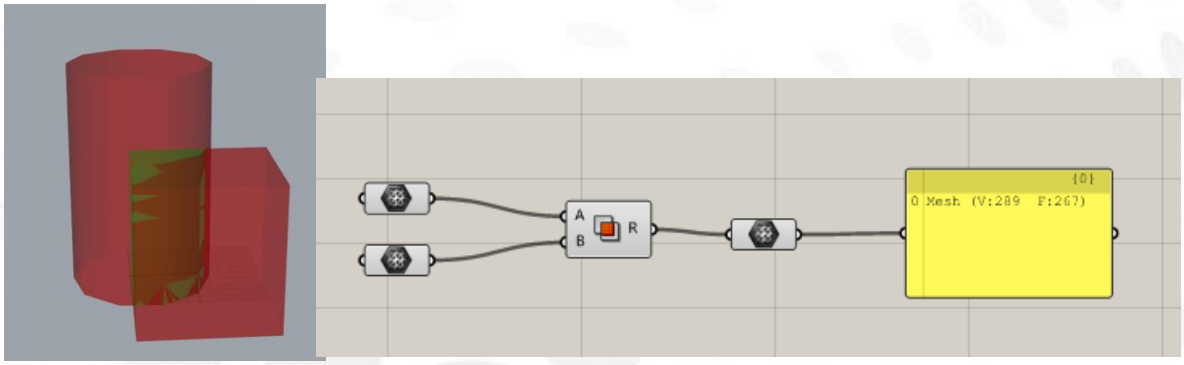
Region Intersection：一些曲线（区域）之间并集
输入端 C：一些闭合曲线（区域）A
输出端 P：曲线投影平面
输出端 R：并集后的曲线或区域



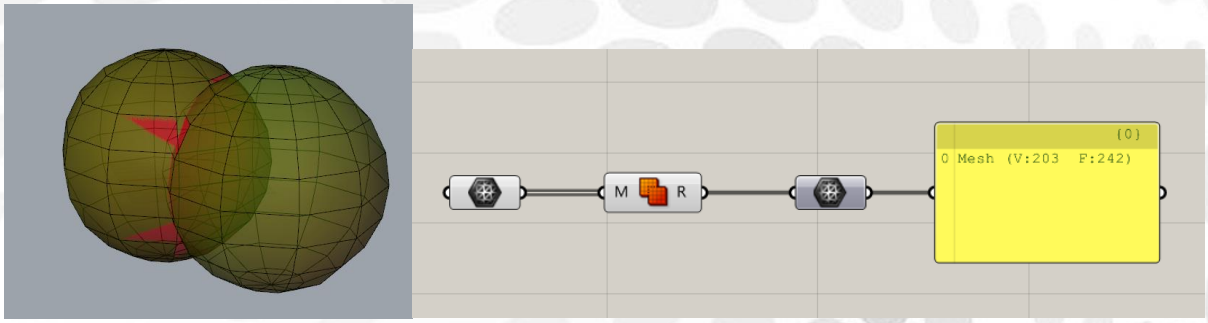
Mesh Difference：网格 B 切割曲线 A
输入端 A：网格 A
输入端 B：网格 B
输出端 R：切割后的网格，含切边



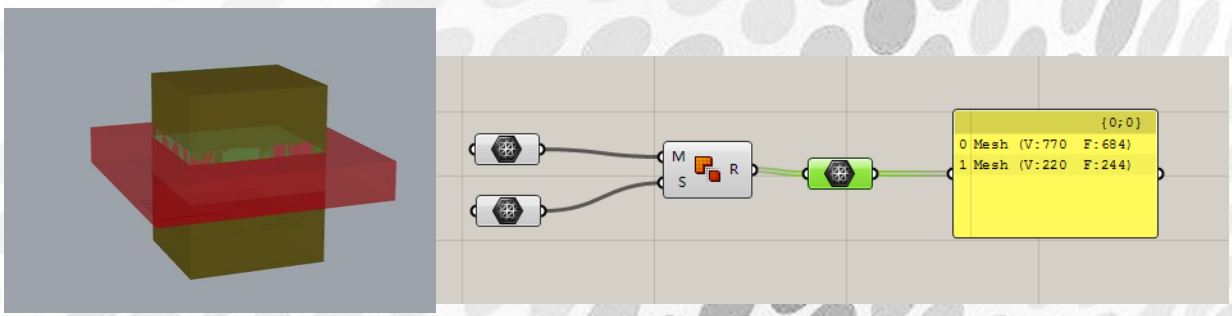
Mesh Intersection：网格之间交集
输入端 A：网格 A
输入端 B：网格 B
输出端 R：交集后的网格



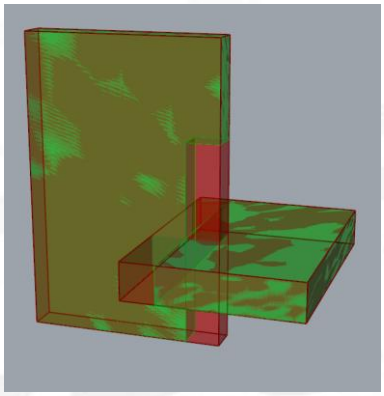
Mesh Intersection：一些网格之间并集
输入端 M：一些网格
输出端 R：交集产生的实体的网格

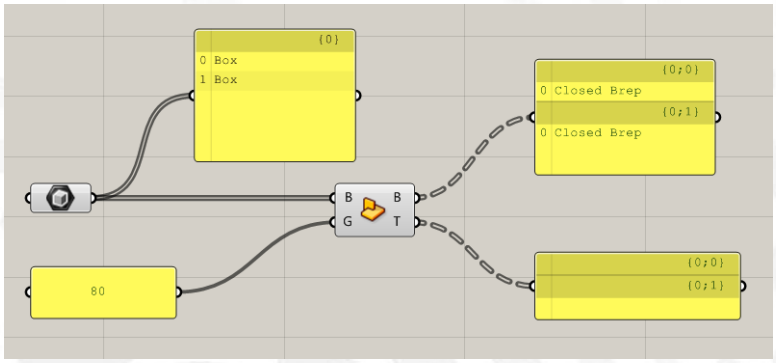


Split Brep：用网格去分割网格
输入端 B：切割用网格
输入端 C：被切割的网格
输出端 R：分割割后所得的网格

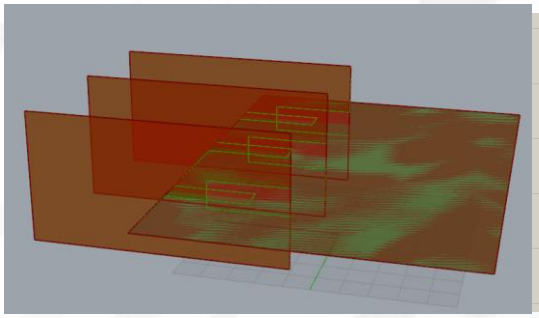


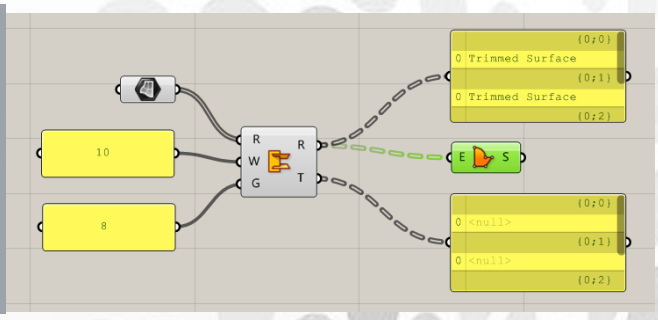
Box slits：在盒子的相交处加入一些裂缝
输入端 B：一些相交的盒子
输入端 G:间隙的宽度
输出端 B：加入了裂缝后的一些盒子
输出端 T：裂缝拓扑





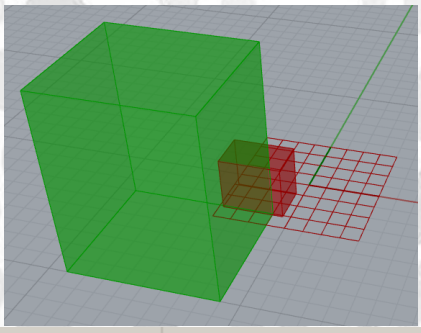
Box slits：在平面区域的相交处加入一些裂缝
输入端 B：一些相交的平面区域
输入端 W:裂缝的宽度
输入端 G 列分的长度
输出端 B：加入了裂缝后的一些盒子
输出端 T：裂缝拓扑



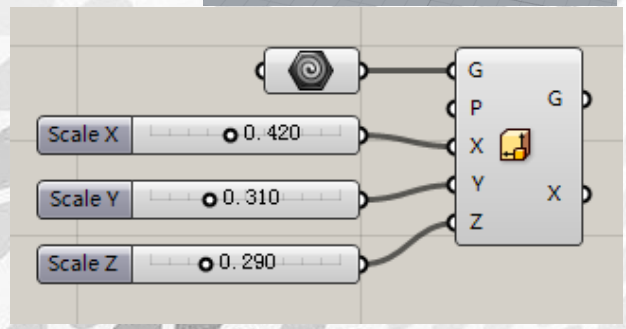


9. Transform 电池组

（1）Analysis 电池序列

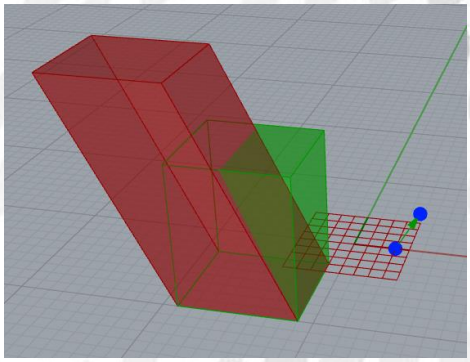


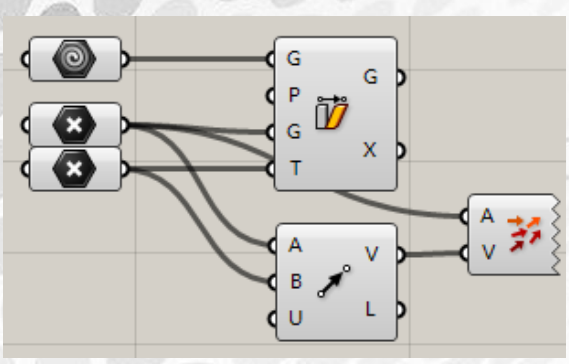
Scale:同犀牛的 Scale，缩放。
输入端 G：需要缩放的物体
输入端 C：缩放中心
输入端 F：缩放系数
输出端 G：缩放后的物体
输出端 X：输入的缩放系数



Scale NU:不等比缩放。
输入端 G：需要缩放的物体
输入端 P：缩放平面，默认是世界平面 World xy
输入端 XYZ：三个轴分别的缩放系数

Shear：具体的用法我并不清楚，直译叫做：通过一个修建向量，改变物体形状
输入端 G：需要缩放的物体
输入端 P：缩放平面，默认是世界平面 World xy
输入端 G：参考点
输入端 T：目标点



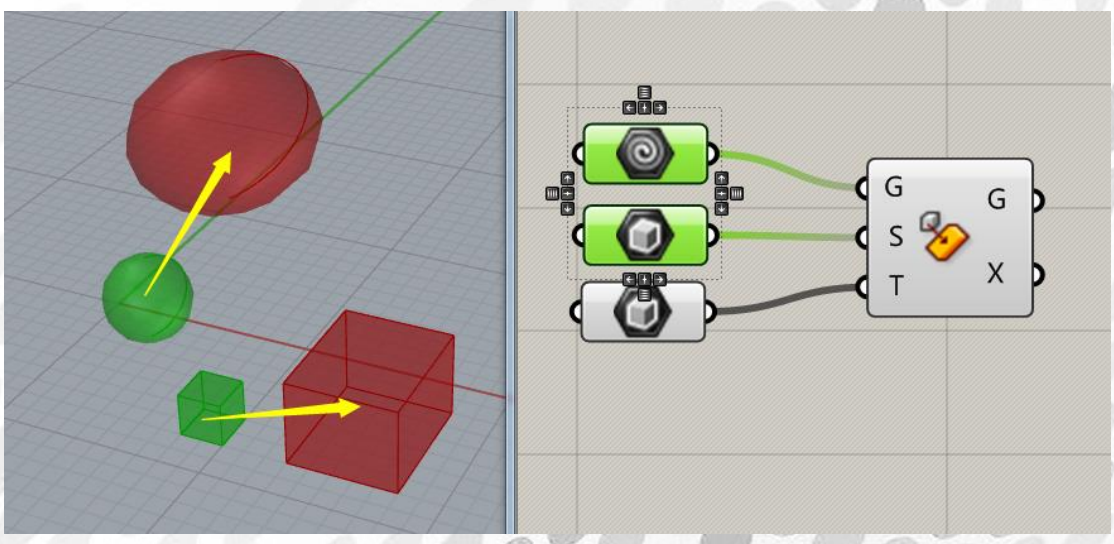


Shear Angle:根据一个角度值，来改变物体形状
输入端 XY：x 和 y 方向的改变角度。通过输入该角度来改变物体。

Orient Direction：用一组方向约束改变物体。
输入端 pA：起始点 A
输入端 dA：起始点 A 的向量
输入端 pB：起始点 B
输入端 dB：起始点 B 的向量

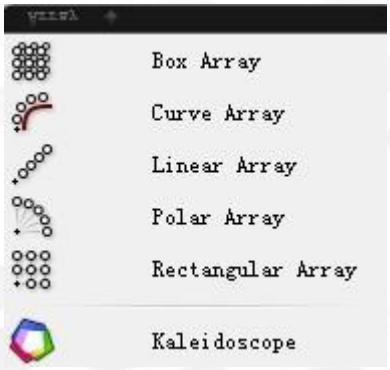
Project：将几何物体投影至平面
Project Along：将几何物体按照某一个方向投影至平面
输入端 G：几何物体
输入端 P：投影平面
输入端 D：投影方向

Box Mapping：通过参考两个 box 的改变来改变输入的物体形状。
输入端 G：需要改变的几何物体。
输入端 S：参考 box
输入端 T：改变后的 box
通过对比 S→T 的改变情况，将 G 端输入的几何物体进行改变。如图所示，绿色 Box 即为输入端 S，绿色球为输入端 G。输入端 T 为红色 Box，红色球为所得输出几何物体。

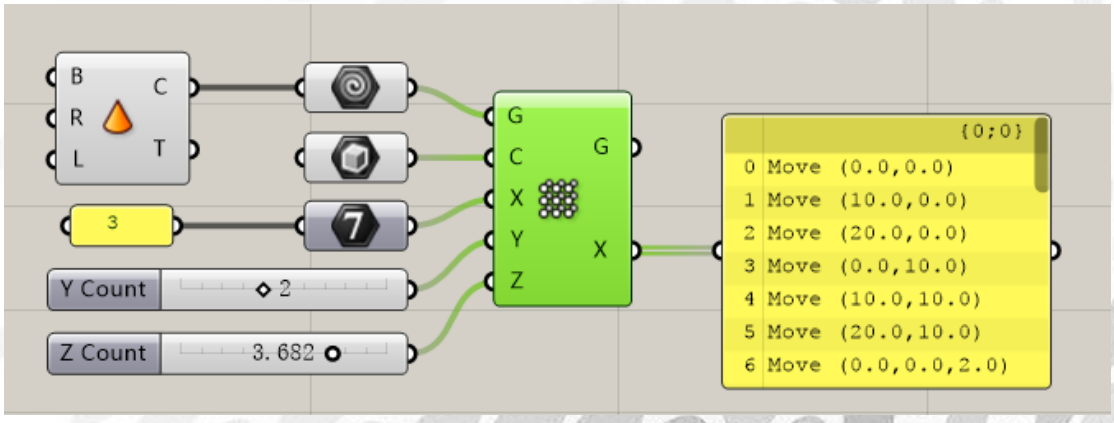


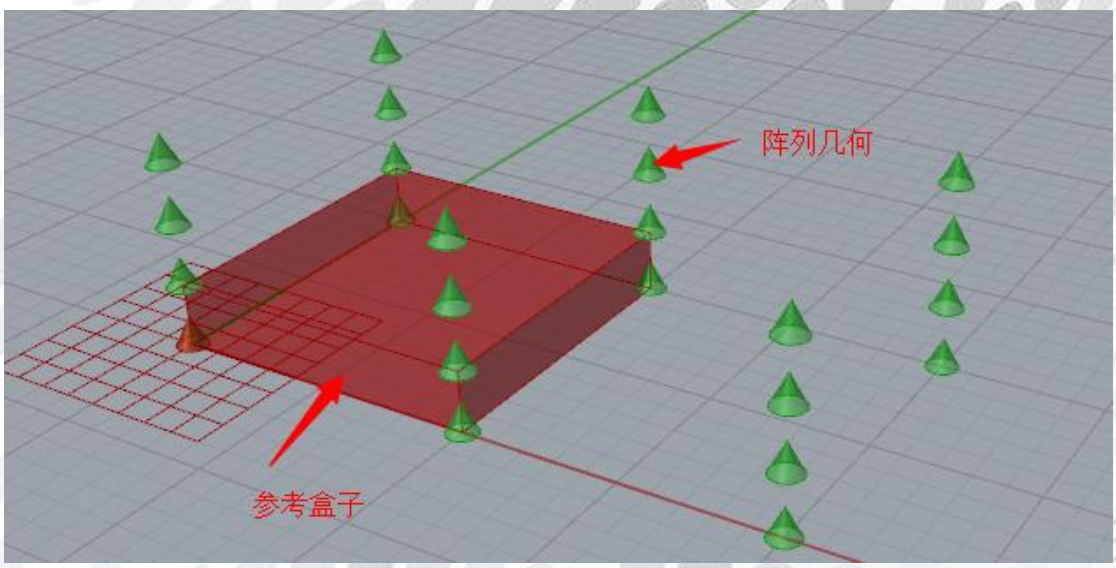
Rectangle Mapping：通过参考两个矩形的改变来改变输入的物体形状。
Triangle Mapping：通过参考两个三角形的改变来改变输入的物体形状。

（2）Array 电池序列



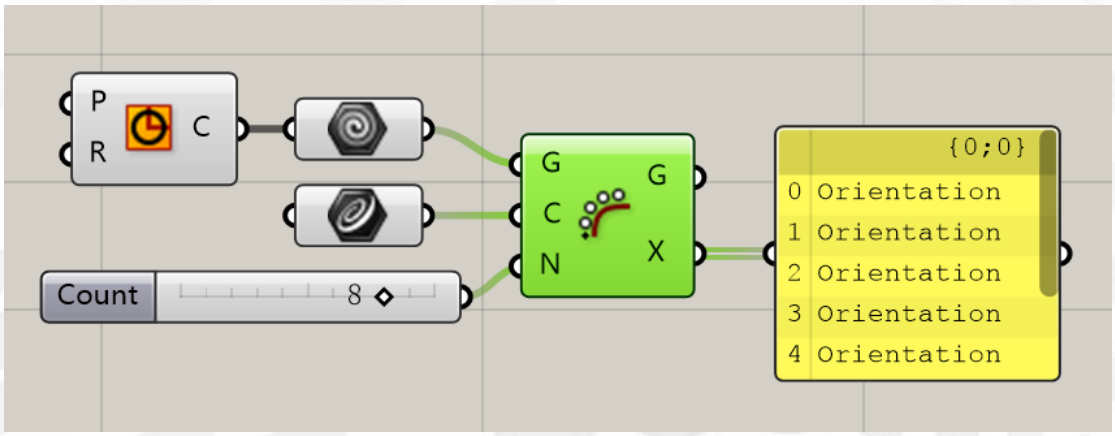
Box Array:根据参考盒子对几何进行阵列
输入端 G:要阵列的几何
输入端 C:参考盒子
输入端 X:X 方向阵列个数（整数）
输入端 Y:Y 方向阵列个数（整数）
输入端 Z:Z 方向阵列个数（整数）（遇到浮点软件会自动取最接近整数）

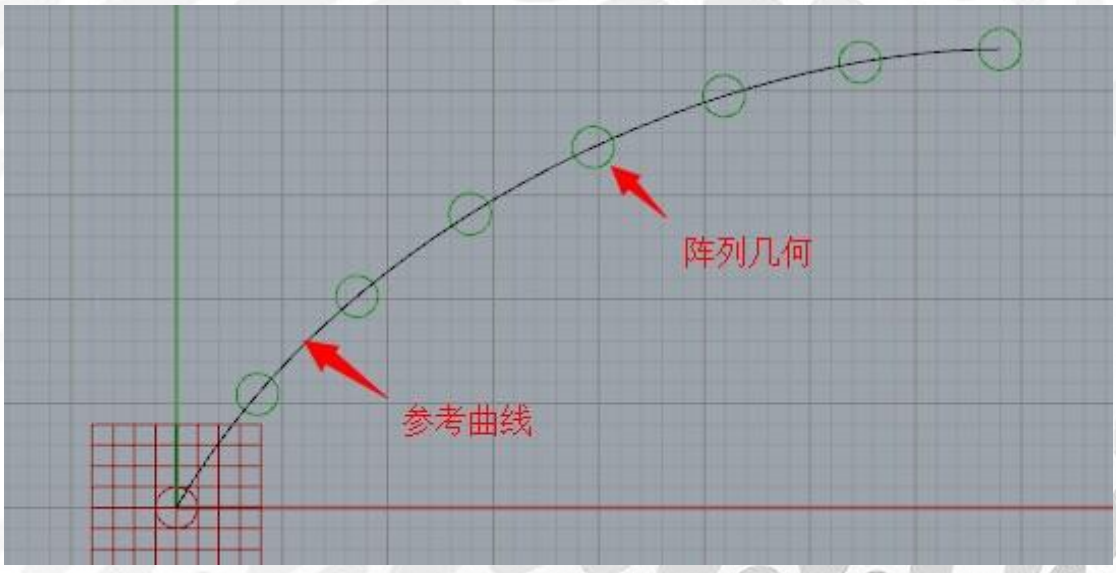




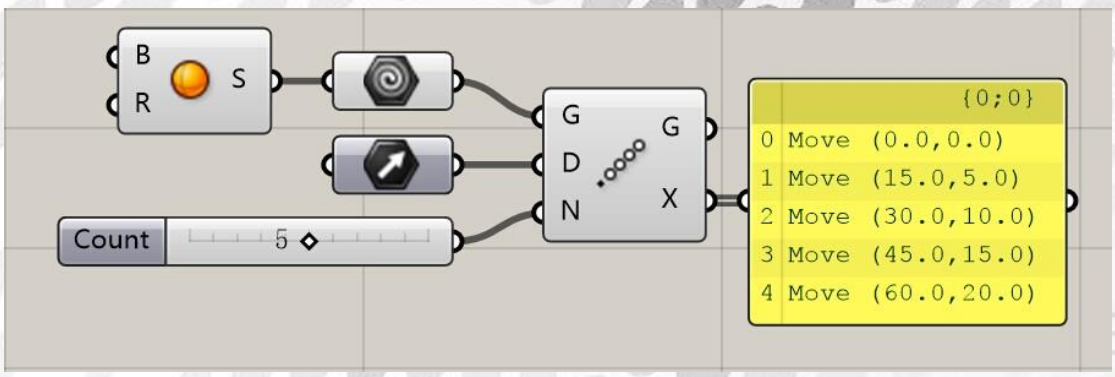
Curve Array:沿着曲线阵列

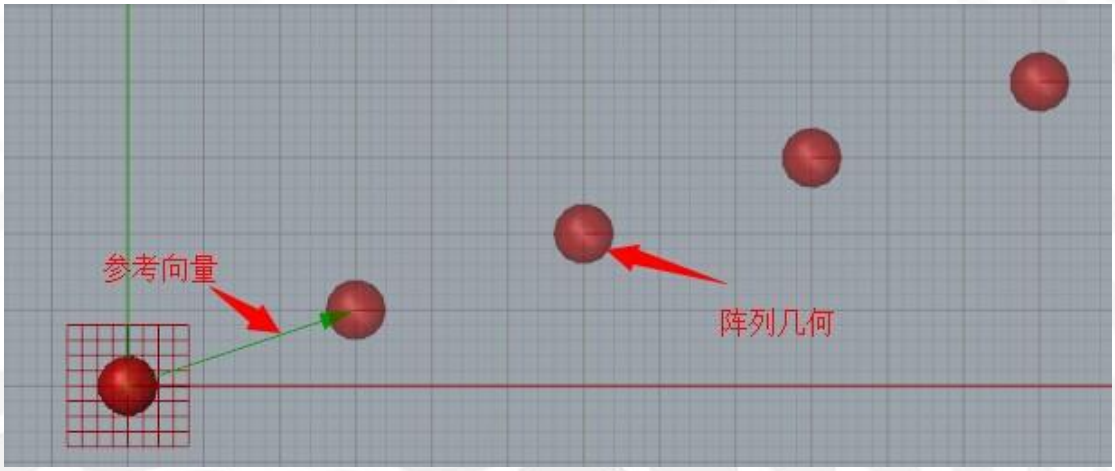
输入端 G: 要阵列的几何
输入端 C:参考曲线
输入端 N:阵列数量



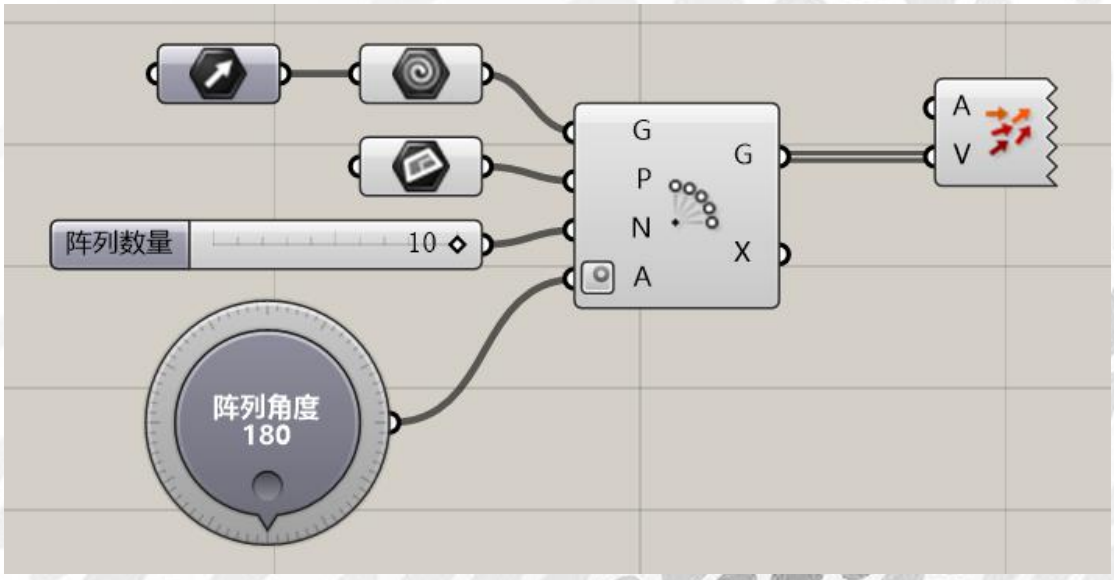


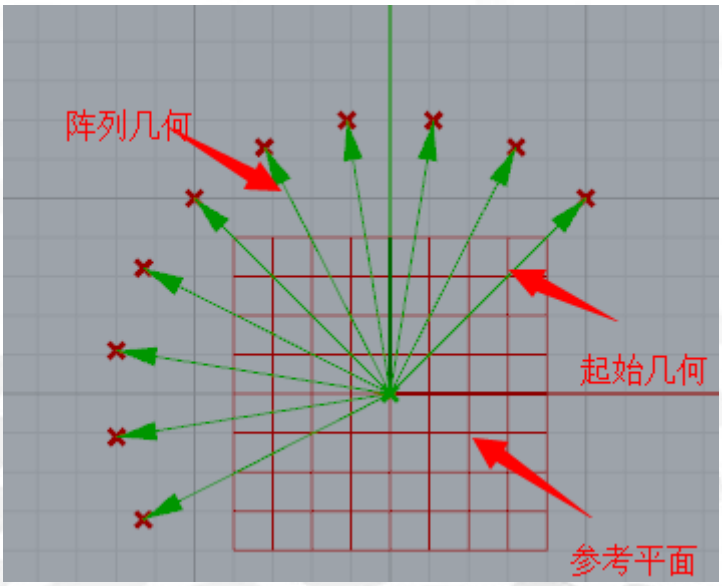
Linear Array:通过参考向量对几何进行直线阵列
输入端 G: 要阵列的几何
输入端 D:参考向量
输入端 N:阵列数量



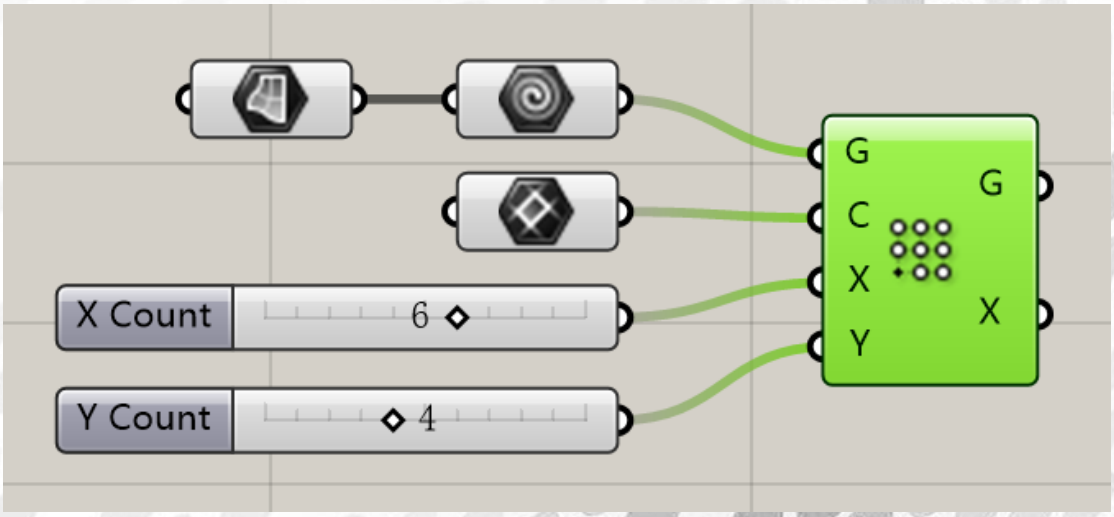


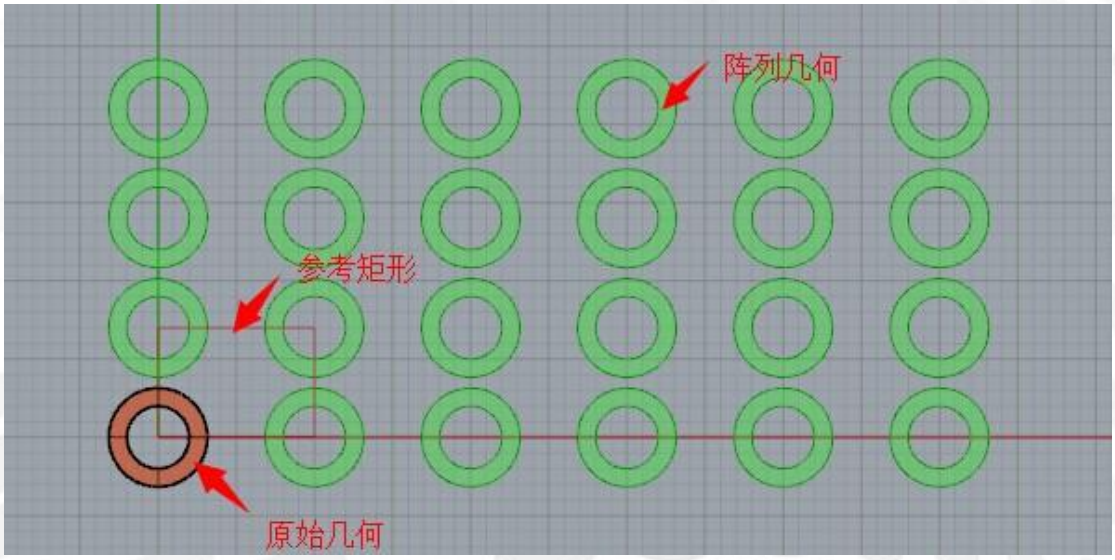
Polar Array:环形阵列
输入端 G: 要阵列的几何
输入端 P:要阵列的平面
输入端 N:阵列数量
输入端 A:阵列角度



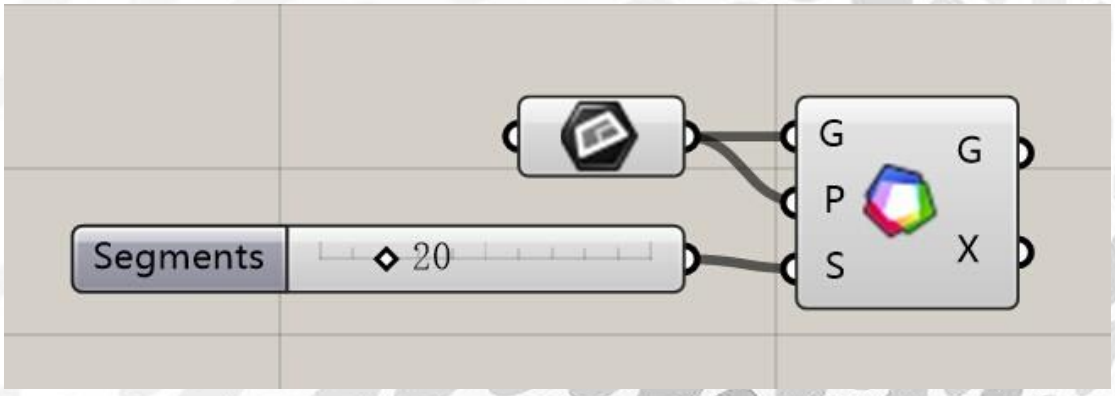


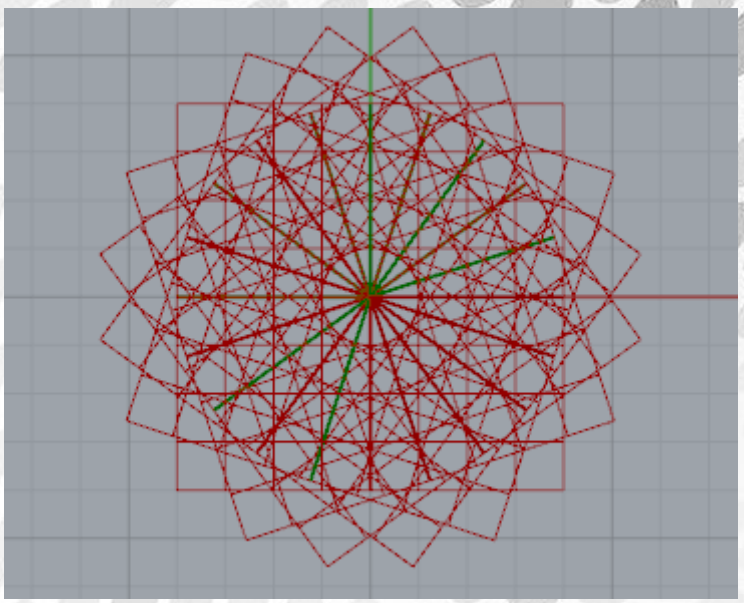
Rectangular Array:矩形阵列
输入端 G: 要阵列的几何
输入端 C:参考矩形
输入端 X:X 轴阵列数量
输入端 Y:Y 轴阵列数量



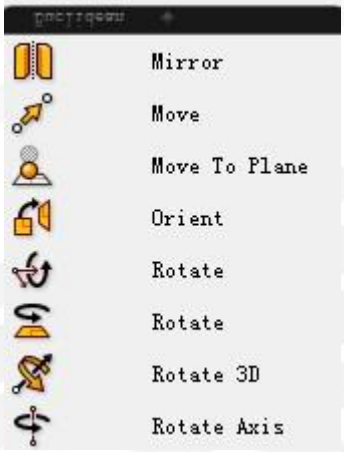


Kaleidoscope:如万花筒般阵列几何
输入端 G: 要阵列的几何
输入端 P:参考 P 平面
输入端 S:应该类似万花筒镜子数目





（3）Euclidean 电池序列



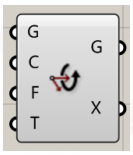
Mirror：将输入物体对照某一平面镜像。同 Rhino 中 mirror

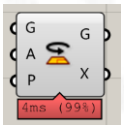
Move：移动。
输入端 G：输入几何物体
输入端 T：移动向量

Move To Plane：将某一几何体移动到某一平面上
输入端 G：输入几何物体
输入端 P：输入平面，默认为世界平面 world xy
输入端 A：当物体在该平面上方时，移动。（True 或 False）
输入端 B：当物体在该平面下方时，移动。（True 或 False）

Orient：通过参考输入平面 A 和 B 之间的变化量，旋转输入端 G 的几何物体

旋转命令：该部分命令建议大家都拖出来对比一下效果差别，会更方便理解掌握。
如果您希望看到实例运用中的作用，欢迎登陆 csh.eeeetop.com 观看我发布的【By DanielJin】系列教程中跟随阳光高度渐变的可调节窗户的案例。





Rotate：根据一个中心点和两个方向，来旋转一个物体。
Rotate：根据角度和平面来旋转物体。
Rotate 3D：通过绕一根轴（X）和给定角度（A）和中心点（C）来 3D 旋转一个物体
Rotate Axis：通过绕一根轴（X）和给定角度（A）来旋转物体

（4）Morph 电池序列

Blend Box：通过两个面创建一个扭曲的盒子
输入端 Sa，Sb：输入的上下两个面
输入端 Da，Db：上下两个面的范围
要注意的是，如果输入面 S 不是平面，盒子也将会相应的扭曲。
Box Morph：通过参考两个盒子 R 到 T 的变形，改变输入端 G 的几何物体
Surface Box：通过一个输入面 S 和平面的范围，得到一个盒子。
要注意的是，如果输入面 S 不是平面，盒子也将会相应的扭曲。

Twisted Box：通过输入的八个点来得到一个盒子

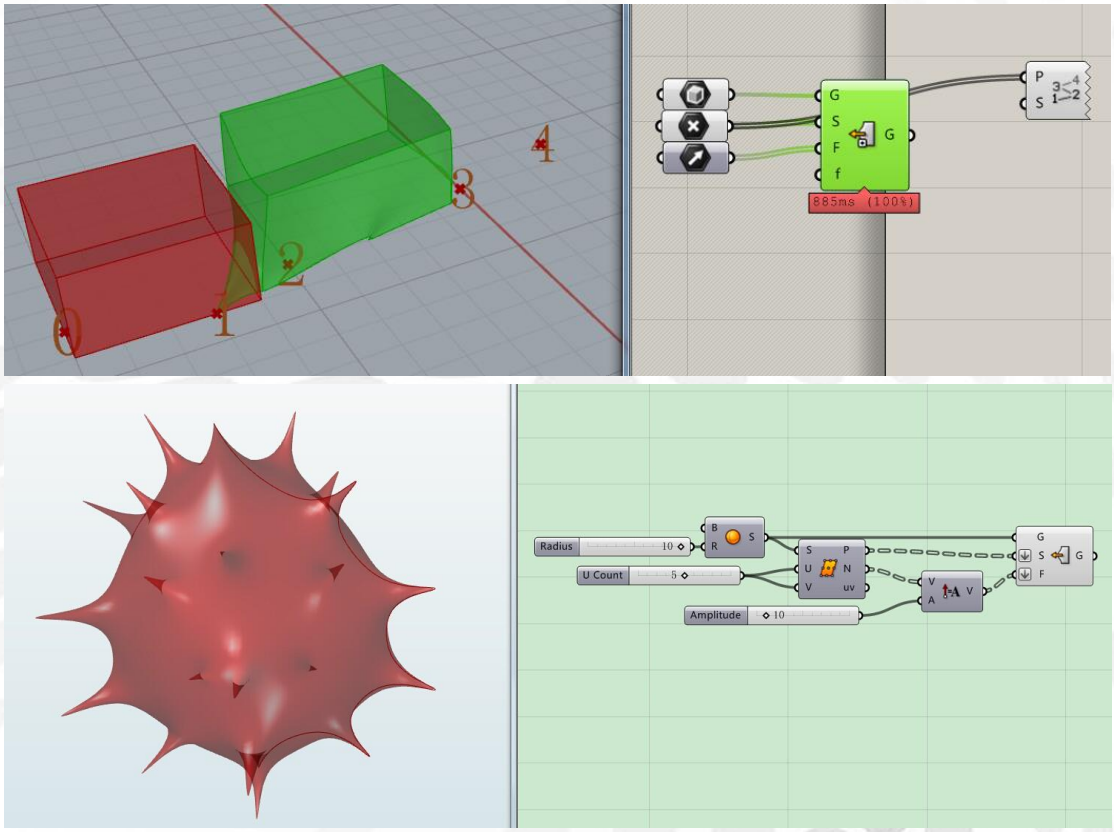
Camera Obscura：小孔成像
输入端 G：输入几何物体
输入端 P：类似“小孔”的点
输入端 F：缩放系数

Map to Surface：通过参考两个面 S 到 T 的变形，改变输入端 C 的曲线

Mirror Curve：将输入物体对照某一曲线镜像
Mirror Surface：将输入物体对照某一面镜像，可以为非平面

Spatial Deform
Spatial Deform(Custom)
输入端 G：输入几何物体
输入端 S：输入变形规则的起始点
输入端 F：输入变形规则向量

输入端 f：衰减值



上图为 Chester.L 的示范

要注意，输入 G 的数量需要等于 S 的数量。

这两个命令虽然应翻译成：基于自定义执行空间进行变形的空间规则，我实在不清楚应该如何用文字表述用法。如果您知道更多，欢迎您在 csh.eeeetop.com 提出解答或向我发送站内信，我会在修订版中更新。

Surface Morph：沿曲面流动，同 Rhino 中沿曲面流动

输入端 G：需要流动的物体

输入端 R：包裹的盒子

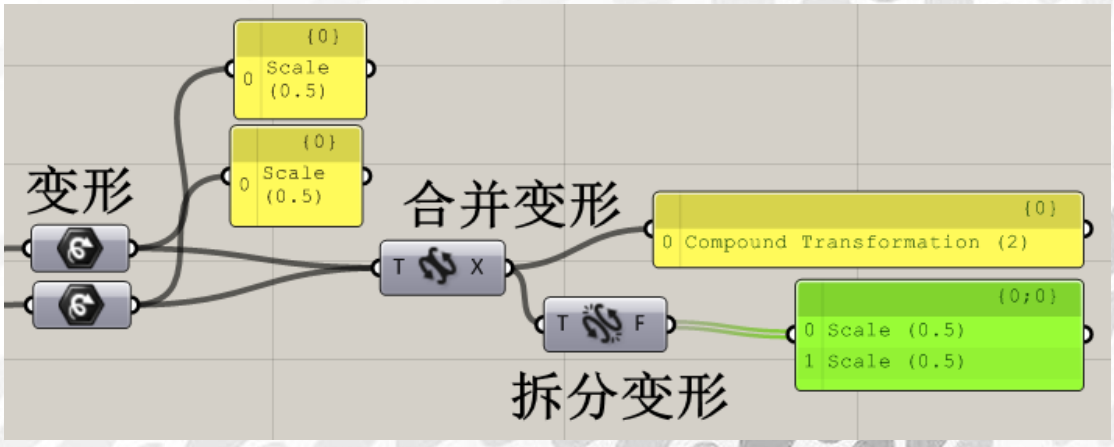
输入端 S：流动的曲面

输入端 UVW：流动曲面的范围

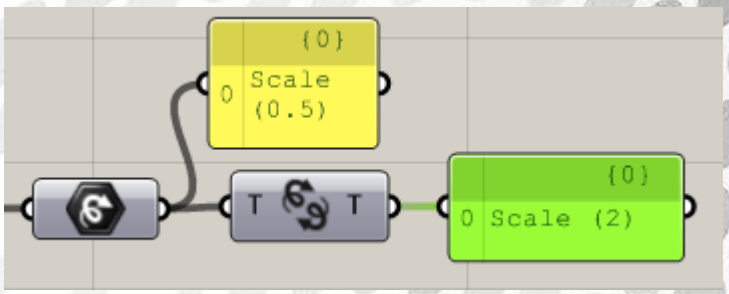
流动的原理是将需要流动的几何物体包裹在一个 box 中，然后 box 按照曲面的 uvw 进行变形同时 box 中的几何形体也随之变形。

（5）Util 电池序列

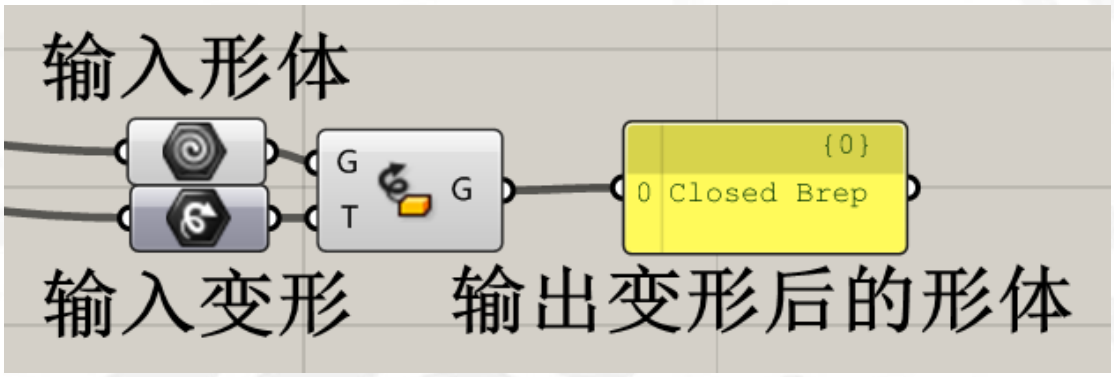
Compound：合并变形，可以将两个单独的变形条件（如变高，变胖）合并成一个。
Split：拆分变形



Inverse Transform：调转变形条件，比如缩放 1/2 倍会被变成缩放 2 倍，即扩大 2 倍

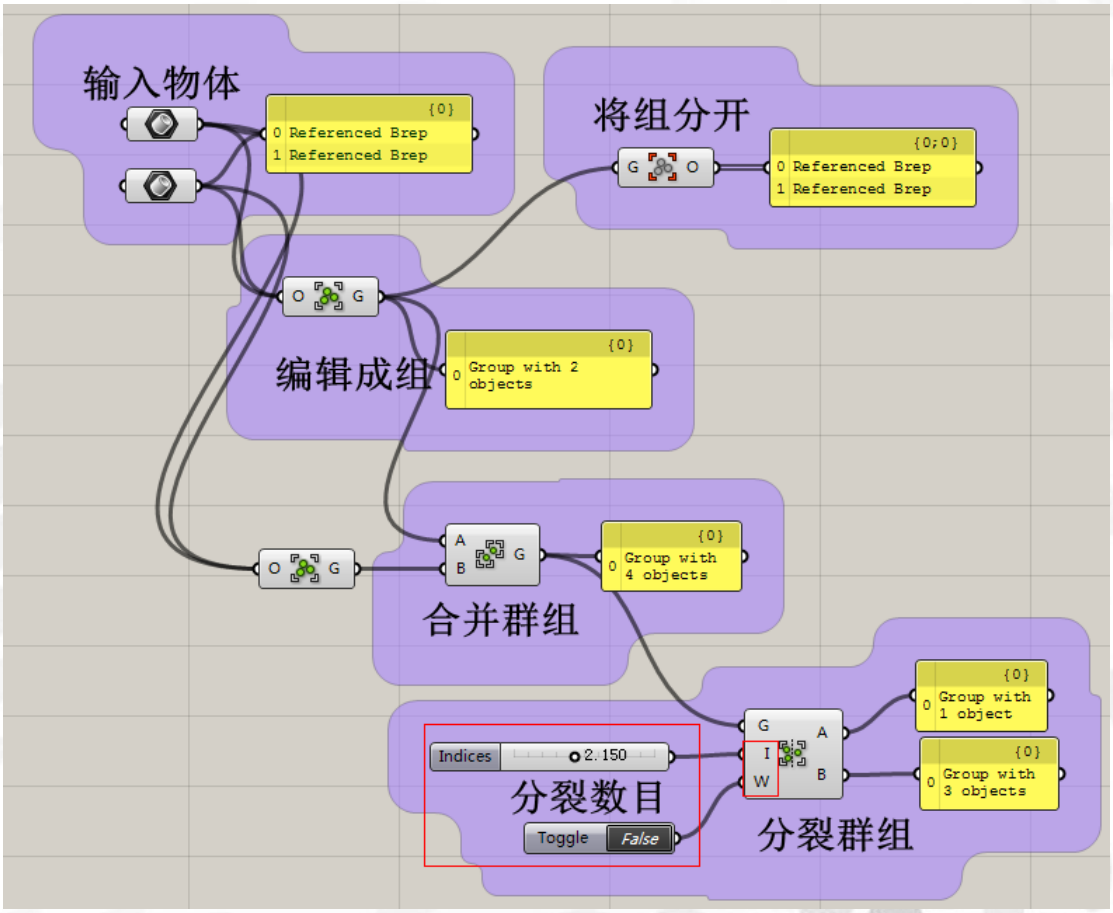


Transform：通过输入形体和变形条件，改变形体



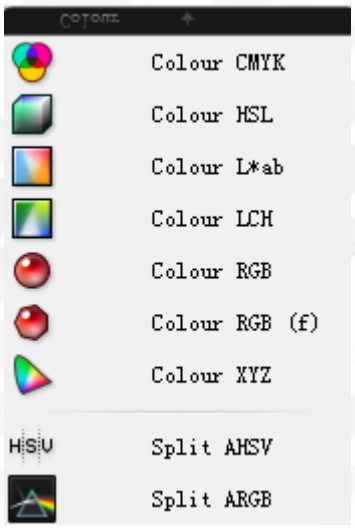
Transform Matrix：改变矩阵
具体用法未知，欢迎您在 csh.eeeetop.com 提出解答或向我发送站内信，我会在修订版中更新。

Group：编辑成组
UnGroup：将组分解
Merge Group：合并群组
Split Group：分开群组
输入端 G：待分离的群组
输入端 I:分离数目
但是我并不清楚在 Split Group 中 I 端和 W 端的具体作用。欢迎您在 csh.eeeetop.com 提出解答或向我发送站内信，我会在修订版中更新。

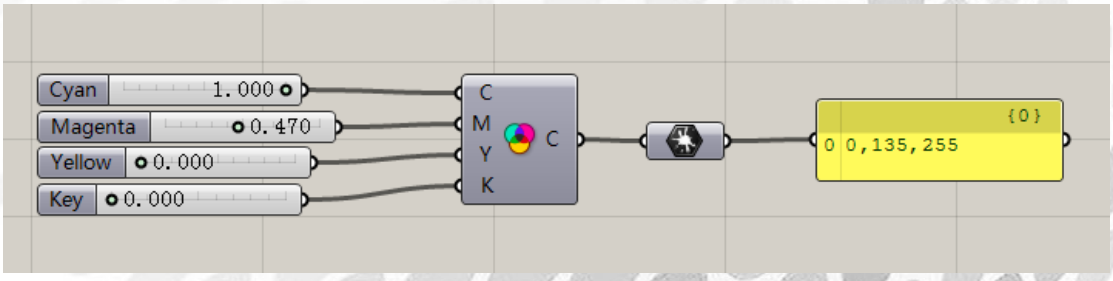

输入物体
{0}
0 Referenced Brep
1 Referenced Brep
将组分开
G O
{0;0}
0 Referenced Brep
1 Referenced Brep
O G
编辑成组
{0}
0 Group with 2 objects
O G
A
B
G
合并群组
{0}
0 Group with 4 objects
Indices
2.150
分裂数目
Toggle
False
G
I
W
A
B
分裂群组
{0}
0 Group with 1 object
{0}
0 Group with 3 objects

10.Display 电池组

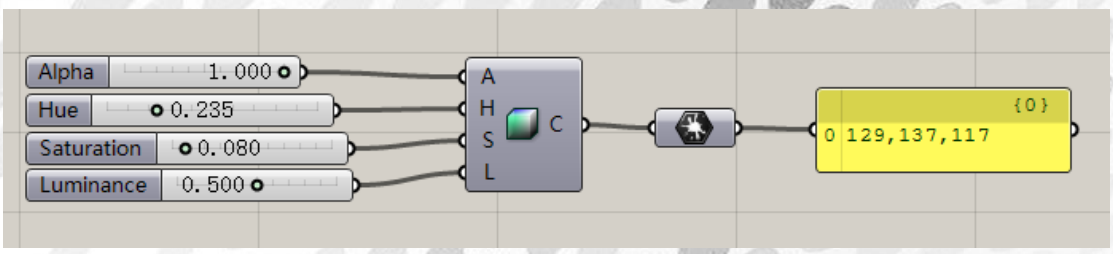
（1）Colour 电池序列



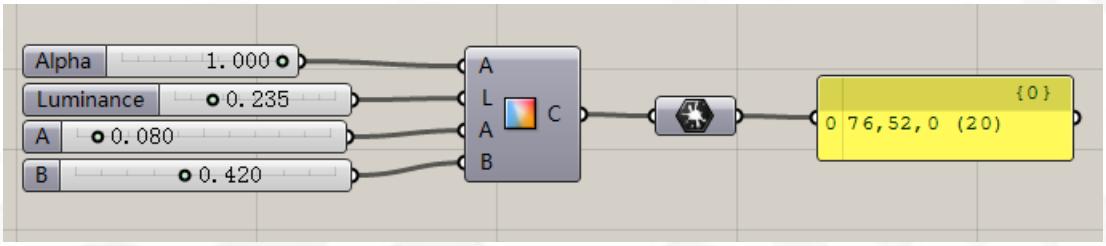
Colour CMYK:通过 CMYK 创建一个或多个颜色
输入端 CMYK：青，品红，黄，黑，数值需定义在 0~1 内



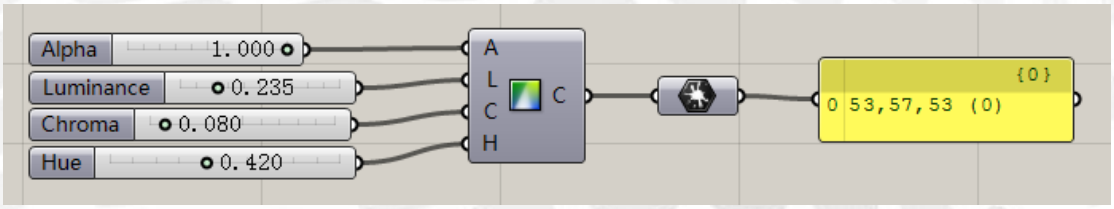
Colour HSL:通过 HSL 创建一个或多个颜色
输入端 A：alpha 通道
输入端 HSL：色相，饱和度，明度，数值需定义在 0~1 内



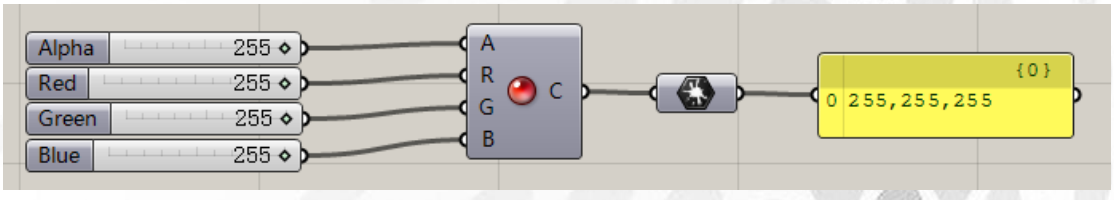
Colour L*ab:通过 LAB 创建一个或多个颜色
输入端 A：alpha 通道
输入端 LAB：亮度，洋红~绿，黄~蓝，数值需定义在 0~1 内



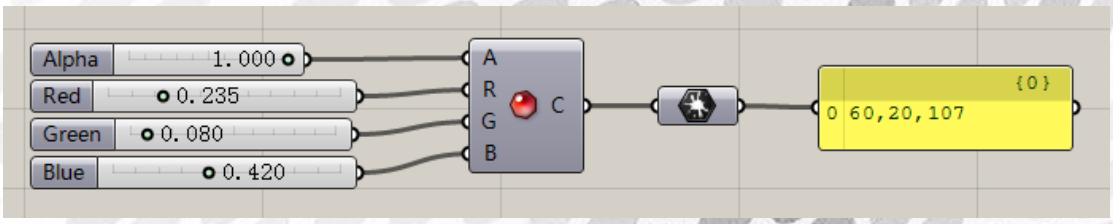
Colouur LCH:通过 LCH 创建一个或多个颜色
输入端 A：alpha 通道
输入端 LCH：亮度，色度，色相，数值需定义在 0~1 内



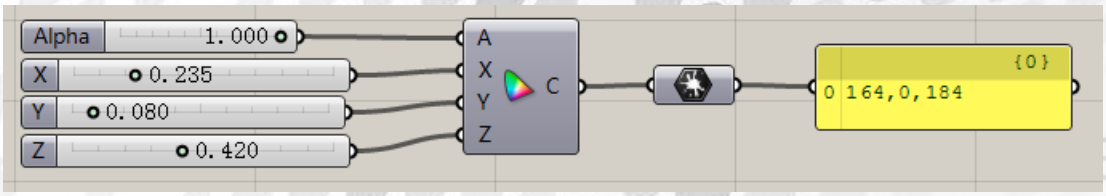
Colour RGB:通过 RGB 创建一个或多个颜色
输入端 A：alpha 通道
输入端 RGB：真实三原色，数值需定义在 0~255 内



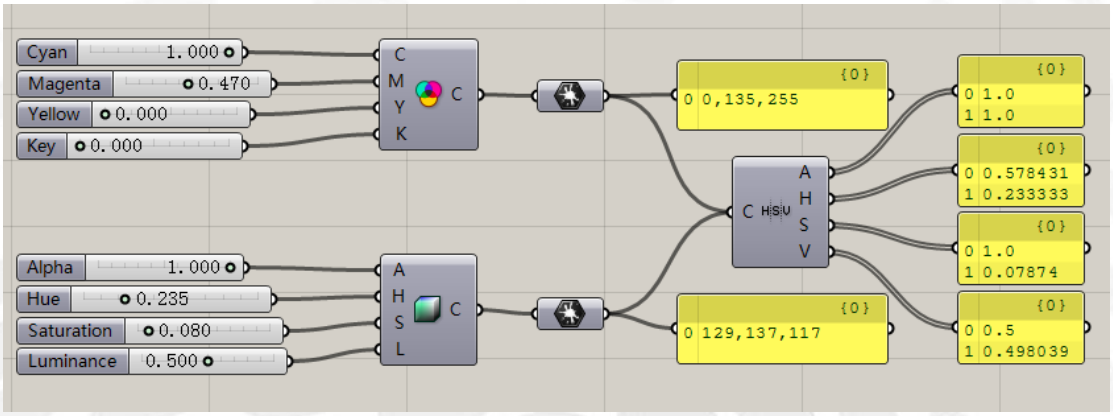
Colour RGB(f):通过 RGB 创建一个或多个颜色（区别于 Colour RGB）
输入端 A：alpha 通道
输入端 RGB：真实三原色，数值需定义在 0~1 内



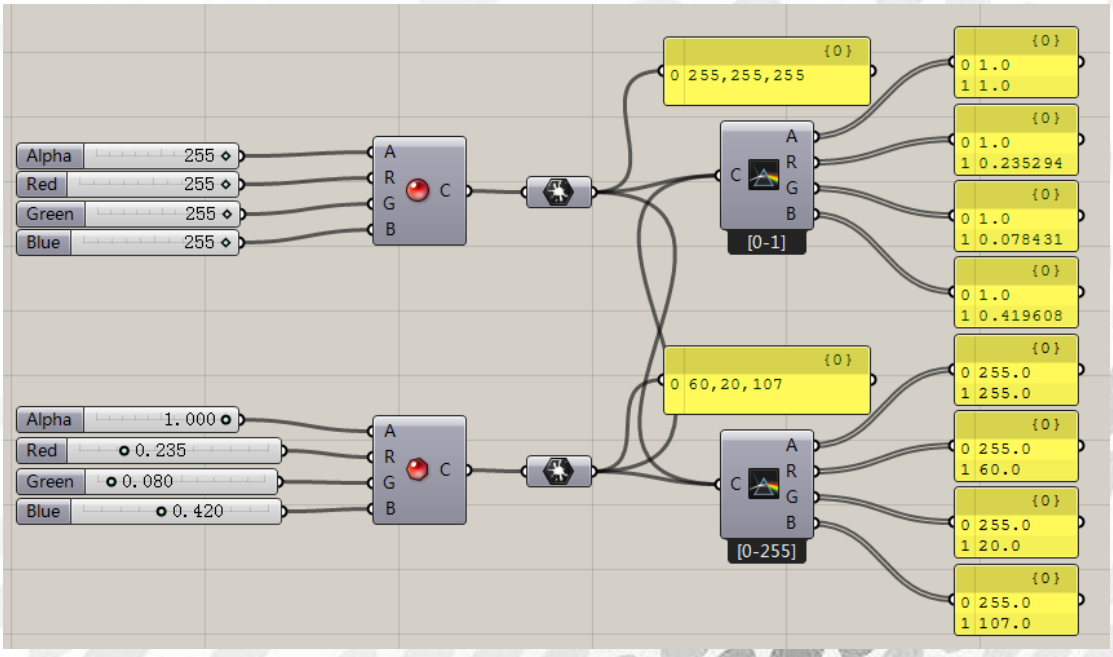
Colour XYZ:通过 XYZ 创建一个或多个颜色
输入端 A：alpha 通道
输入端 XYZ：理想三原色，数值需定义在 0~1 内



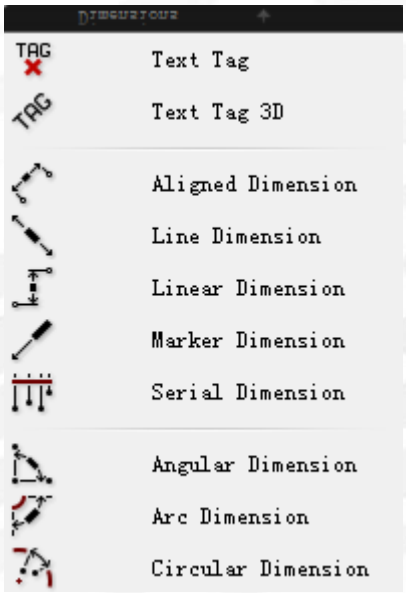
Split AHSV:从一个或多个颜色中提取 HSV 数值
输出端 A：alpha 值
输出端 HSV：色相，饱和度，色调



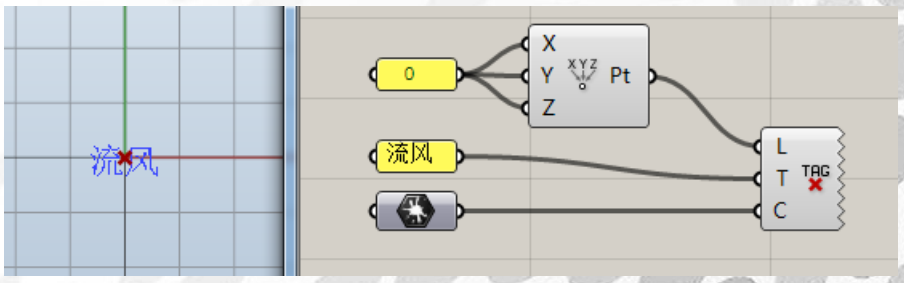
Split ARGB:从一个或多个颜色中提取 RGB 数值
输出端 A：alpha 值
输出端 RGB：真实三原色（右键 integer channel 以 0~255 输出）



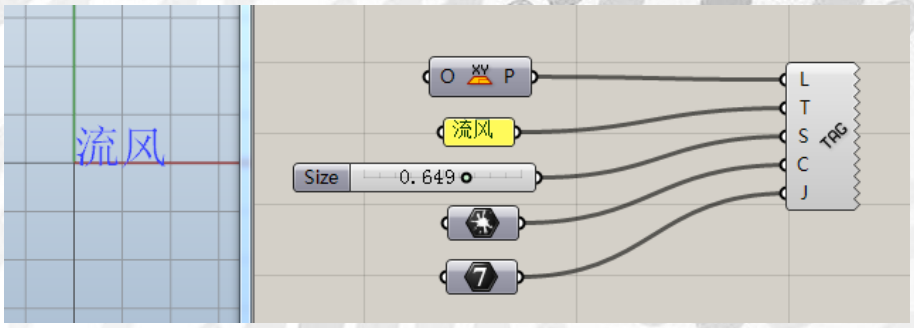
（2）Dimension 电池序列



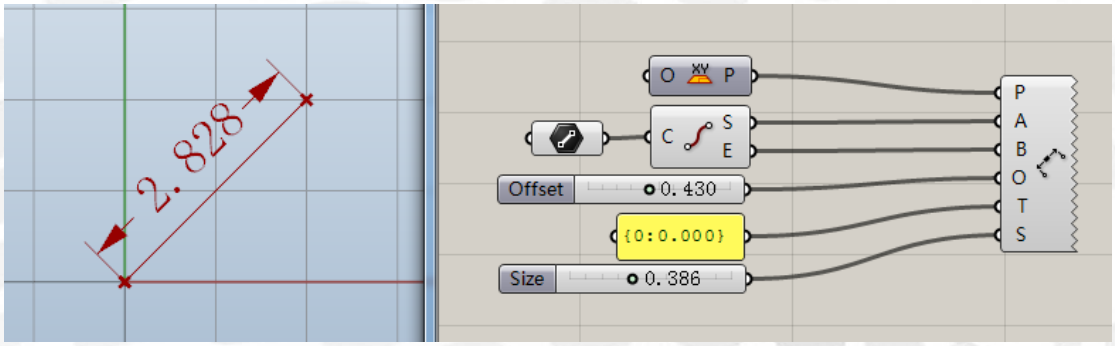
Text Tag:在犀牛界面内显示文本标签(随视窗调整保持尺寸不变，右键可选择显示尺寸)
输入端 L：文本标签位置
输入端 T：文本内容
输入端 C：显示颜色



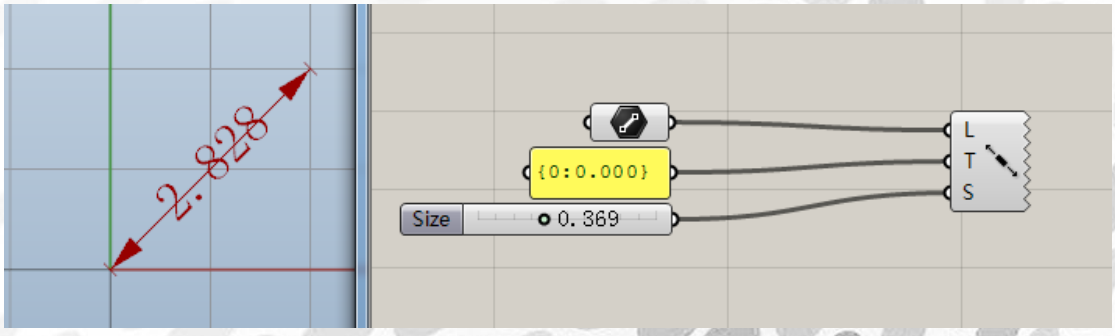
Text Tag 3D:3D 文本标签
输入端 L:文本标签位置和方向
输入端 T:文本内容
输入端 S:显示尺寸
输入端 C:显示颜色
输入端 J:对齐方式（建议右键设置）



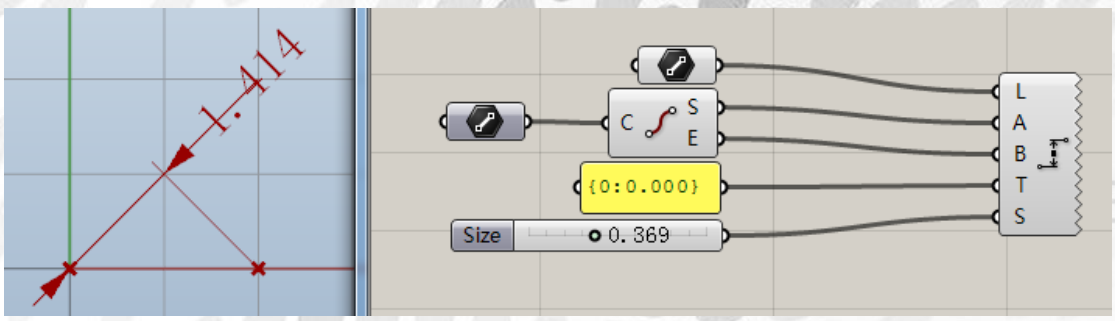
Aligned Dimension:对齐标注
输入端 P:标注平面
输入端 A:标注起点
输入端 B:标注终点
输入端 O:标注偏移距离
输入端 T:标注内容（可替换）
输入端 S:标注尺寸



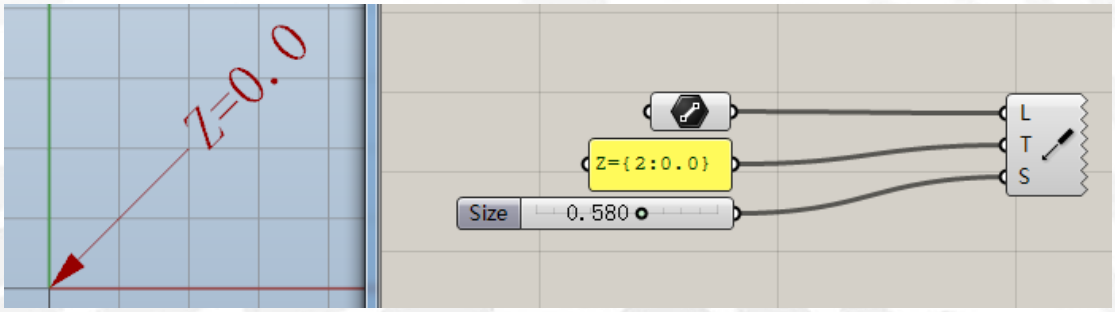
Line Dimension:直线标注
输入端 L:标注基准直线
输入端 T:标注内容（可替换）
输入端 S:标注尺寸



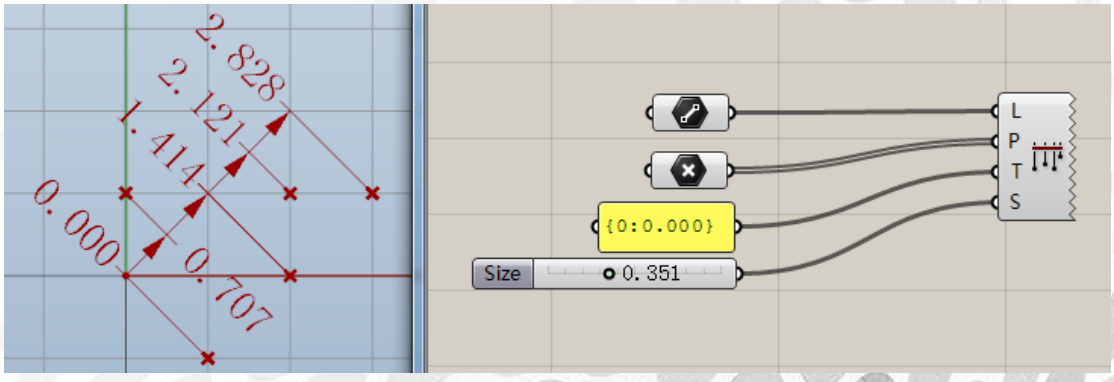
Linear Dimension:线性标注
输入端 L:标注基准直线
输入端 A:标注起点
输入端 B:标注终点
输入端 T:标注内容（可替换）
输入端 S:标注尺寸



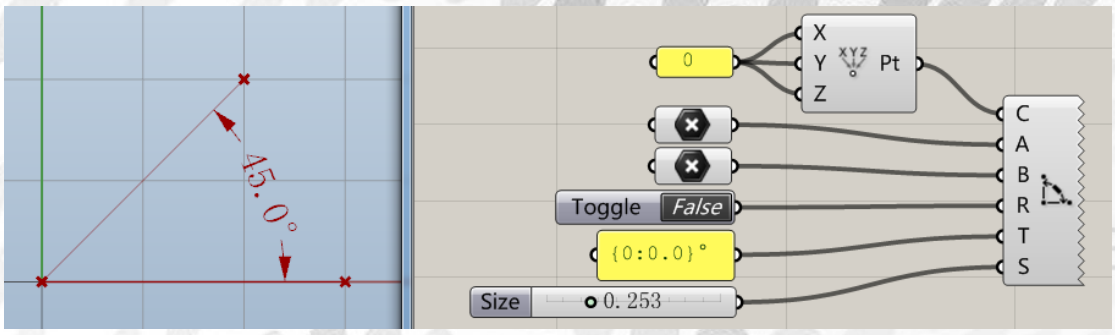
Marker Dimension:注释标注
输入端 L:标注基准直线
输入端 T:标注内容（可替换）
输入端 S:标注尺寸



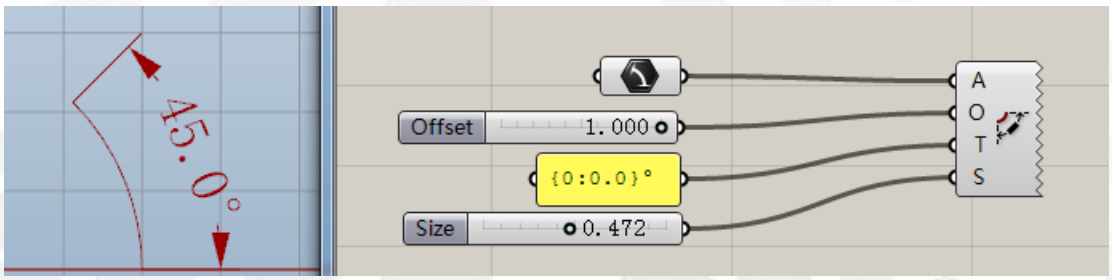
Serial Dimension:数个点投影到直线上创建连续标注
输入端 L:标注基准直线
输入端 P:标注点（第一点默认为 0）
输入端 T:标注内容（可替换）
输入端 S:标注尺寸



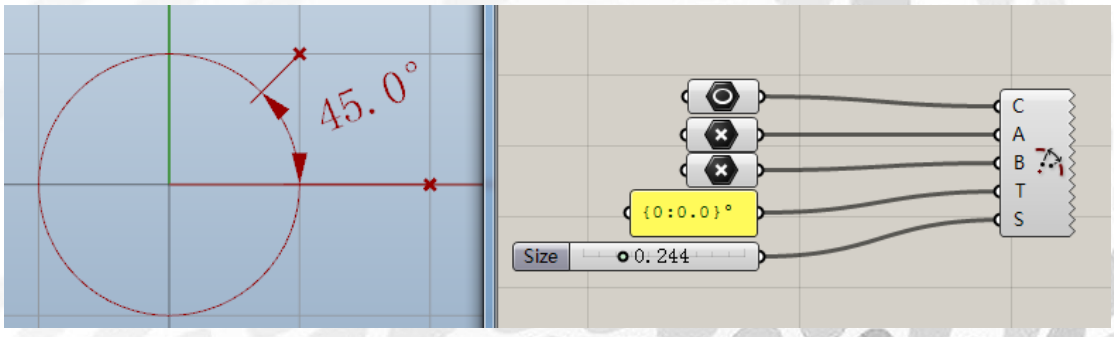
Angular Dimension:点控制角度标注
输入端 C:中心点
输入端 A:起始方向端点
输入端 B:终结方向端点
输入端 R:布尔值，F 为劣角显示，T 为优角显示
输入端 T:标注内容（可替换）
输入端 S:标注尺寸



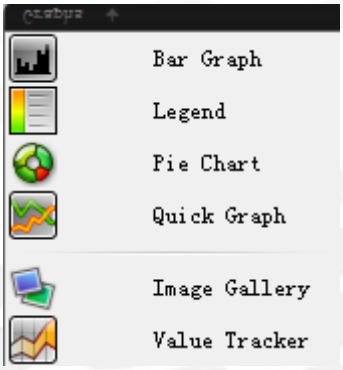
Arc Dimension:弧线控制角度标注
输入端 A:弧线
输入端 O:标注偏移距离
输入端 T:标注内容（可替换）
输入端 S:标注尺寸



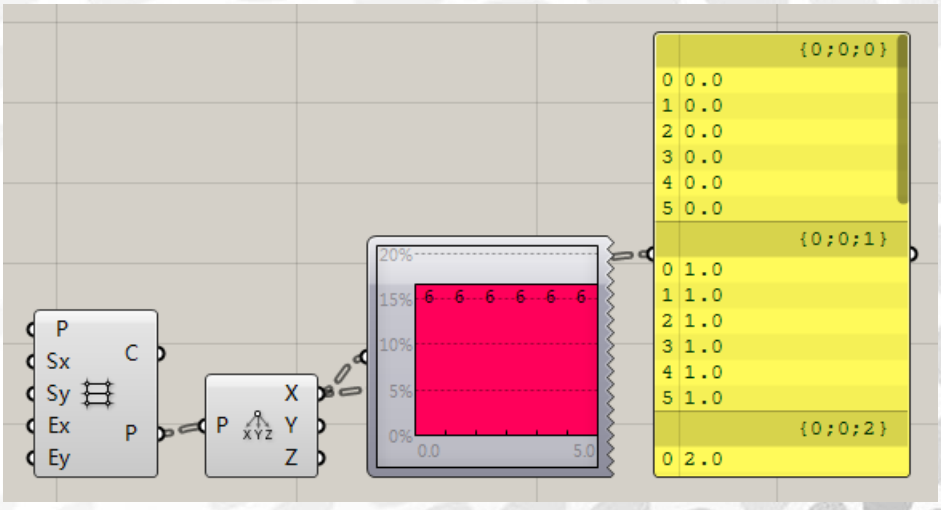
Circular Dimension:点投影圆控制角度标注
输入端 C:圆
输入端 A:起始方向端点
输入端 B:终结方向端点
输入端 T:标注内容（可替换）
输入端 S:标注尺寸



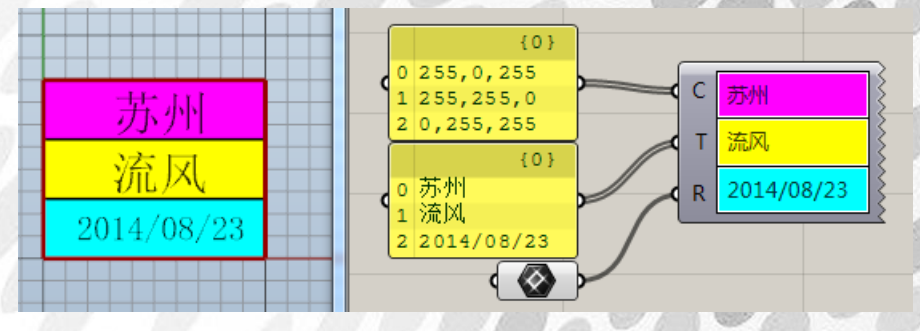
（3）Graphs 电池序列



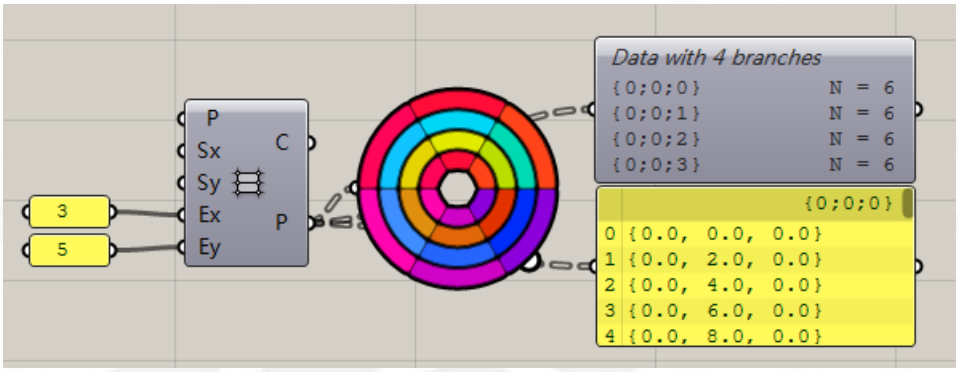
Bar Graph：直方图
用于数据统计，右键可设置 bars 数量，正常情况下总百分比为 100%



Legend：图例
以一组相应的文标签和颜色创建图例，右键可设置显示类型



Pie Chart：饼图
用于显示输入数据的具体内容，左击拖动右下角半圆可缩放图标，放大到一定程度后可全面显示输入数据。个人觉得类似于 Param Viewer 与 Panel 运算器的结合版，但运用不是很方便。



Quick Graph：快速图表
以线状图的形式反映数据的变化形态

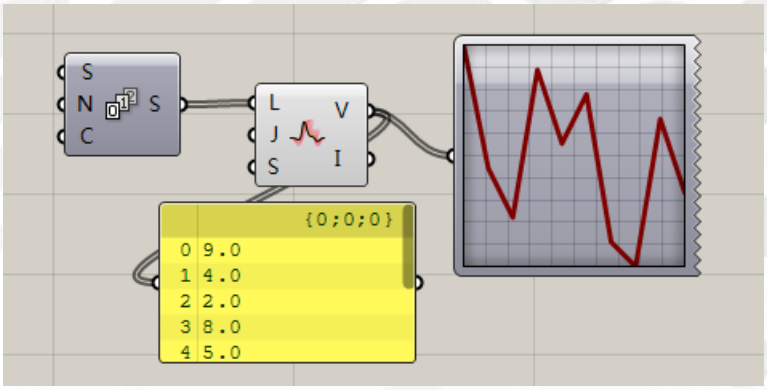
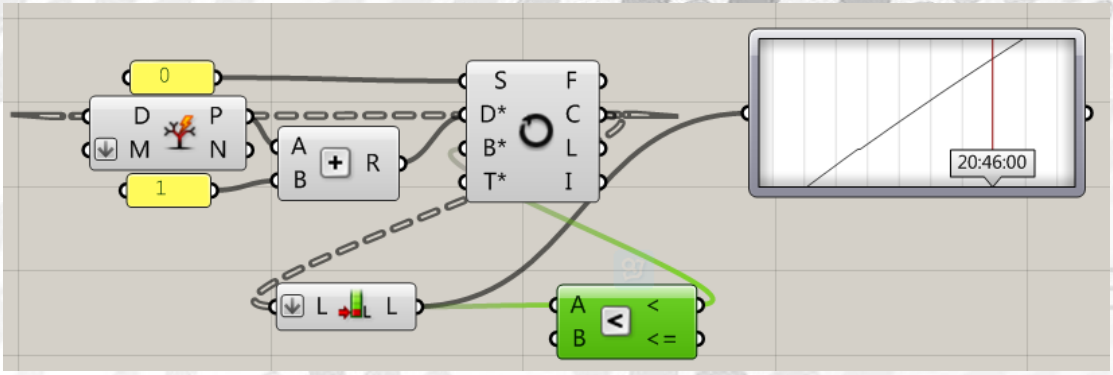
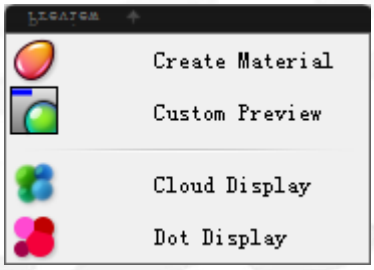


Image Gallery：画廊
双击进入具体设置，输入图像文件路径，图片会根据循环间隔时间重复播放

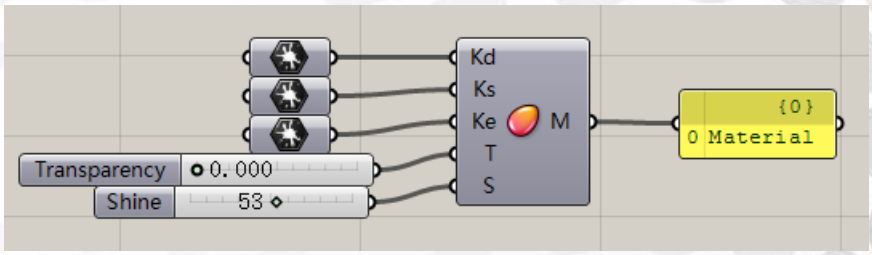
Value Tracker：值追踪器
以动态的形式反映数据的变化规律，右键可做具体设置改动



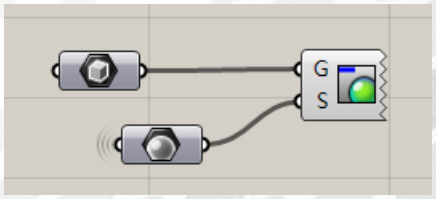
（4）Preview 电池序列



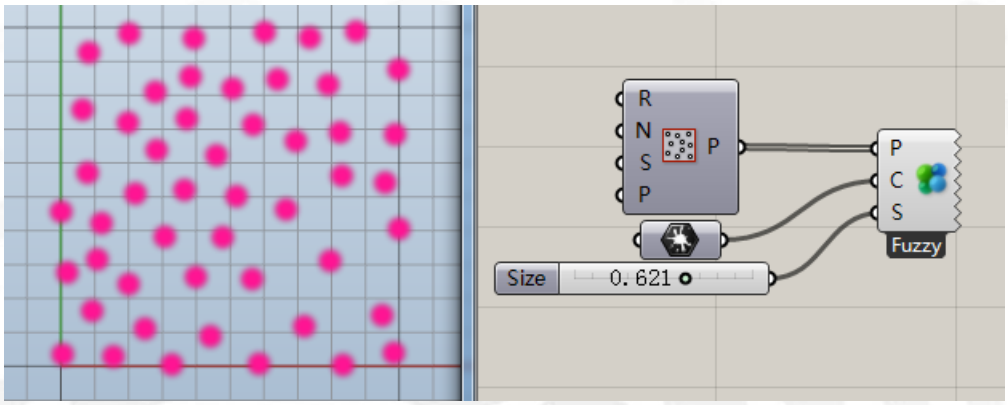
Create Material:创建材质
输入端 Kd：漫反射颜色
输入端 Ks：反射颜色
输入端 Ke：自发光颜色
输入端 T：透明度（数值定义在 0~1）
输入端 S：光泽度（数值定义在 0~100）



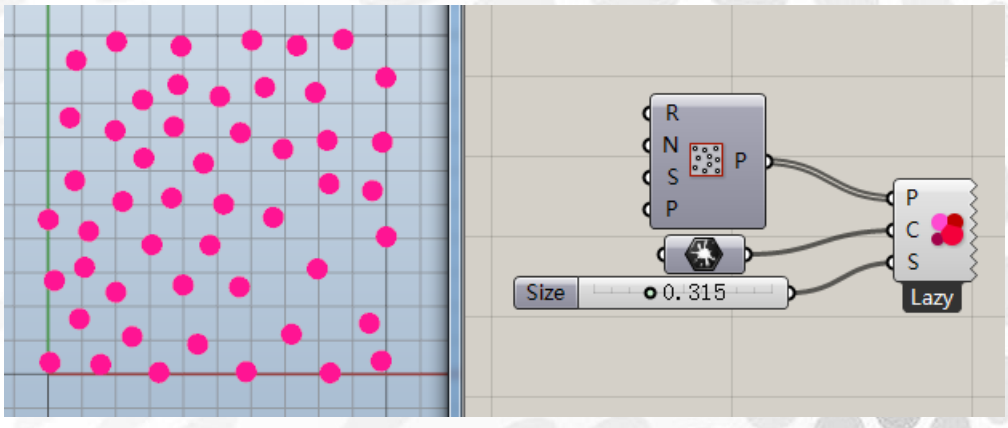
Custom Preview：预览被定义材质的几何体
输入端 G：几何体
输入端 S：材质



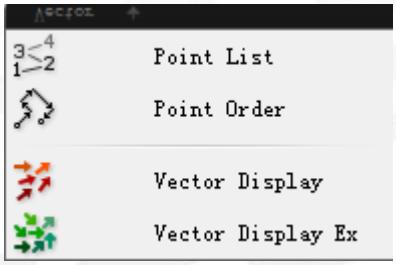
Cloud Display：将点模糊处理显示，右键选择模糊方式
输入端 P：点
输入端 C：显示颜色
输入端 S：显示尺寸



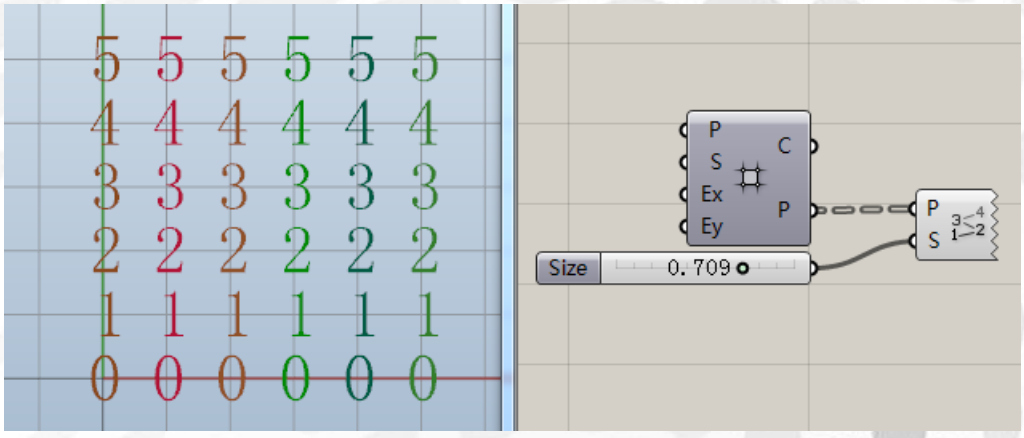
Dot Display：点显示，右键选择处理方式
输入端 P：点
输入端 C：显示颜色
输入端 S：显示尺寸



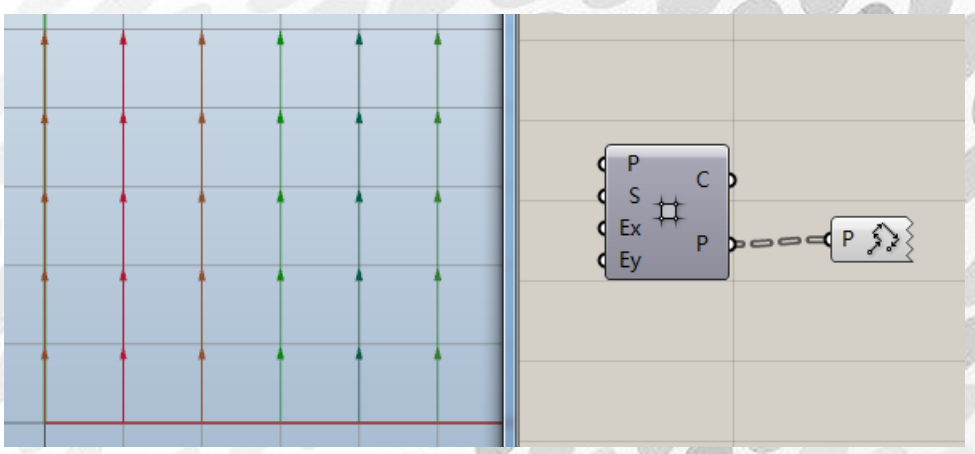
（5） Vector 电池序列



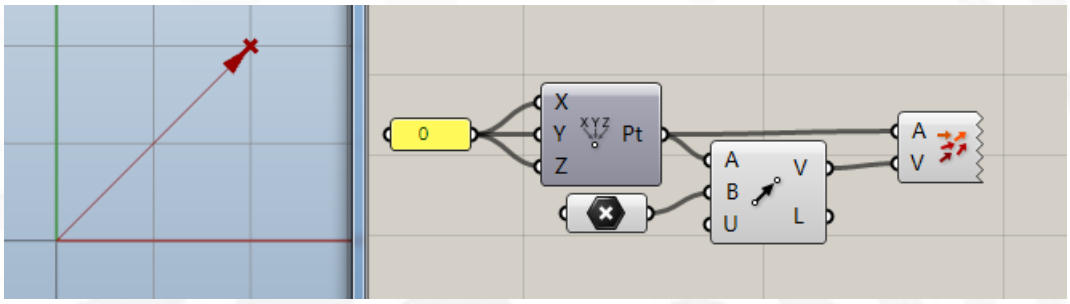
Point List：显示点列表
输入端 P：要显示的点
输入端 S：显示尺寸



Point Order：显示点顺序
输入端 P：要显示的点



Vector Display：显示向量
输入端 A：起始锚点
输入端 V：要显示的向量



Vector Display EX：自定义显示向量
输入端 P：起始锚点
输入端 V：要显示的向量
输入端 C：显示颜色
输入端 W：显示宽度