

# Generative Components 101

## Lesson 9: Stair Creation (x2)

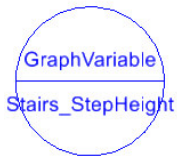
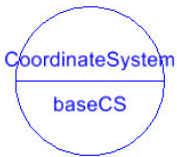
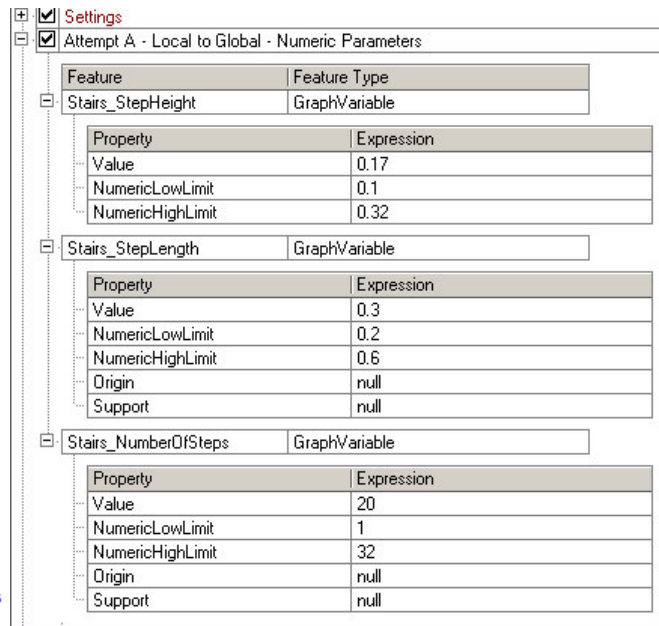
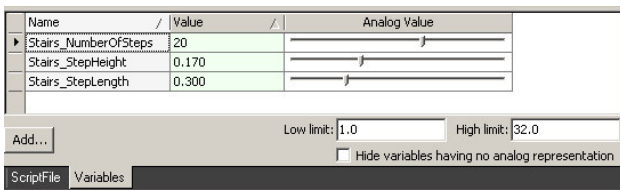
### Lesson 9-A: Using step height and length.

Ok, I'm with ya, where do I start?

Let's create all our GraphVariables first:

#### 1. Define your variables:

##### Graph>Manage Global Variables



#### 2. To start our steps we first want to create a series of Lowerpoints:

##### Point>ByCartesianCoordinates

When we get to the Xtranslation and Ytranslation we can use the GraphVariables created above, within a Series to help us:

**Glossary: Series (start, finish, increment)** A Series is a collection of points along a line or a surface.

E.g. Series(1, 10, 1) returns {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} – starts at 1, ends at 10 and is at intervals of 1.

Series(1, 10, 2) returns {1, 3, 5, 7, 9}, Series(2, 3, 0.25) returns {2, 2.25, 2.5, 2.75, 3}

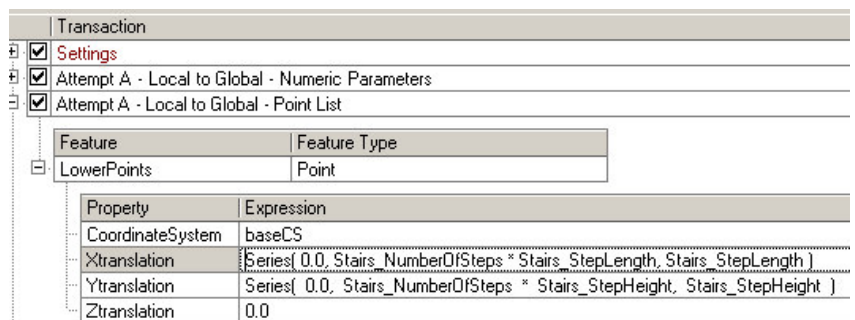
For the X translation:

#### Series(start, finish, increment)

We want to start at 0.0

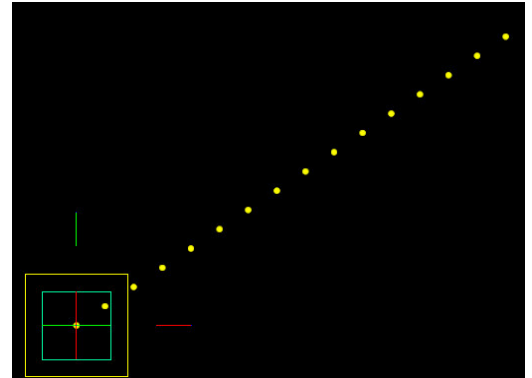
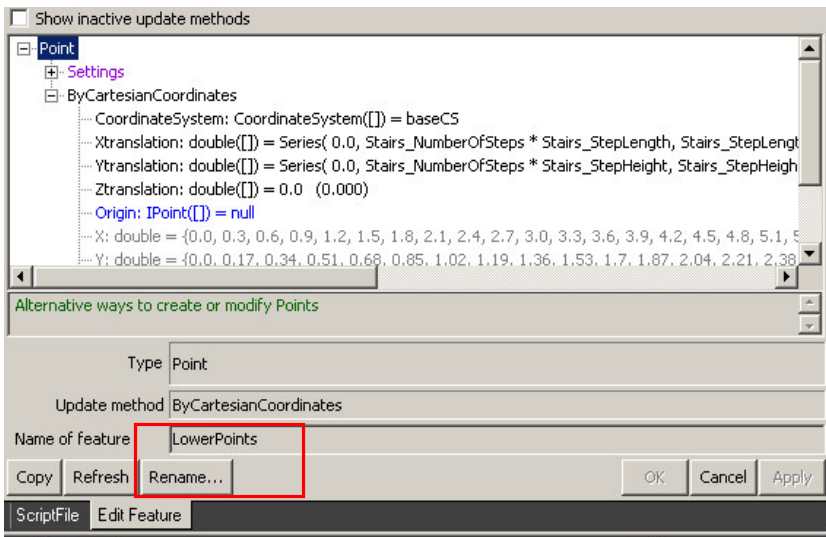
The finish is calculated using the variables: Stairs\_NumberOfSteps x Stairs\_StepLength (Use \* for times)

We want the increment to be the Stairs\_StepLength.



For the Y translation is calculated the same way using the Height Variables.

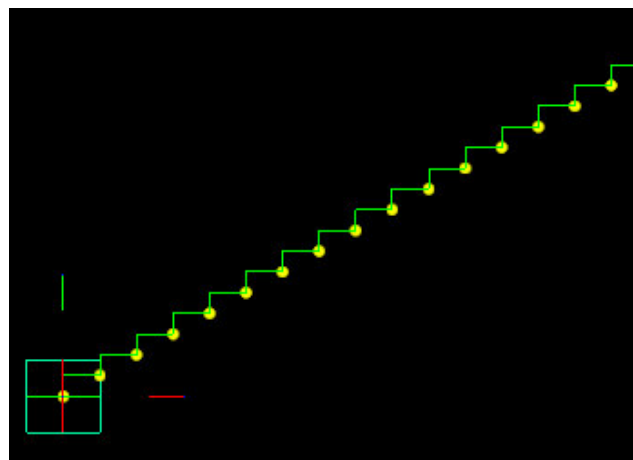
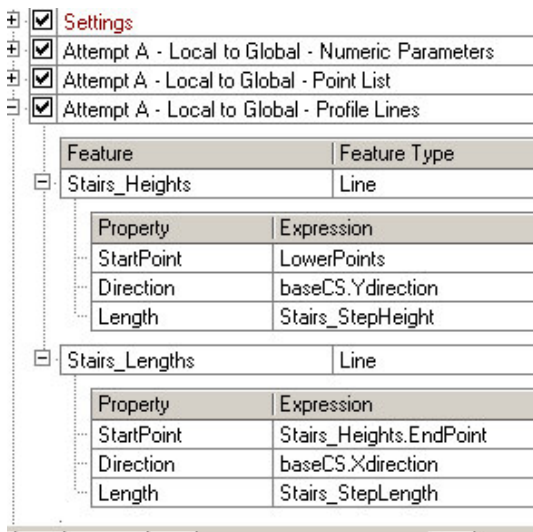
Lets give this Feature a name - LowerPoints that we can recognise it in the future:  
 Forgot to give the feature a name as you were working on it? Just Edit the feature and Rename it.



### 3. Now lets create our profile lines. Line>ByStartPointDirectionAndLength

Lets start with the height lines (risers). Rename your Line feature to Stairs\_Heights.  
 Our start point is the feature LowerPoints.  
 Our direction is from the BaseCS in the Y direction.  
 Our Length is the Variable Stairs\_StepHeight.

And then our lengths (treads). Rename your Line feature to Stairs\_Lengths.  
 This time our start points are the endpoints of the above lines (Stair\_Heights), which we can write by Stairs\_Heights.EndPoint.  
 Our direction is from the BaseCS in the X direction.  
 Our Length is the Variable Stairs\_StepLength.



### 4. Lastly, lets hide our points.



Use the Display Tool (and you thought it was difficult!) and pick the elements you wish to hide – in this case lets hide the points. You'll notice in the Symbolic view that your points graphic has greyed out. How do you unhide something? Use the same tool and pick the greyed out Points graphic.

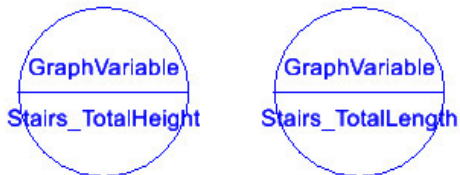
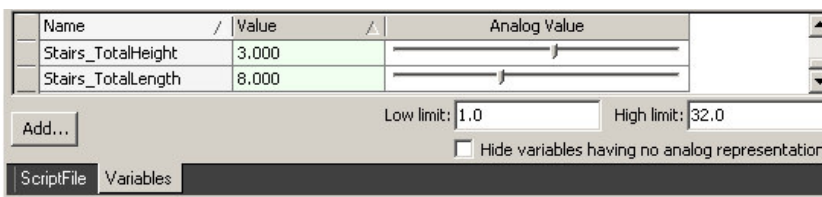
(Notice in the example script that we have hidden everything – so we can show you another way of drawing your stairs...)

## Lesson 9-B: Using stair total height and length.

### 1. Define your variables:

This time we are going to add two new variables for the Total height and Total length of the stair:

#### Graph>Manage Global Variables



2. Again we need to create the lower points of the stairs (we'll give them the name Origin Points in this example) and use our new variables in the Series.

#### Point>ByCartesianCoordinates

Remember **Series(start, finish, increment)**

Feature	Feature Type
OriginPoints	Point

Property	Expression
CoordinateSystem	baseCS
Xtranslation	Series( 0.0, Stairs_TotalLength, Stairs_TotalLength / Stairs_NumberOfSteps )
Ytranslation	Series( 0.0, Stairs_TotalHeight, Stairs_TotalHeight / Stairs_NumberOfSteps )
Ztranslation	0.0

In our X direction:

The finish is the Stairs\_TotalLength

The increment is Stairs\_TotalLength divided by the Stairs\_NumberOfSteps

In our Y direction we use the Height variables instead.

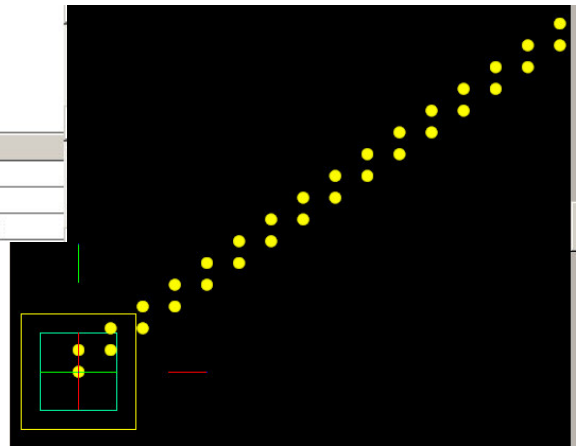
### 3. Now we need to create the top points.

#### Points>ByDirectionDistanceAndOrigin

Our Origin points will be the points we just created (and named the feature OffsetPoints).

Our direction is in the Y direction of our BaseCS.  
 And our distance is our Stairs\_TotalHeight divided by the Stairs\_NumberOfSteps

Transaction	
[x] Attempt B - Global to Local - High Points	
Feature	Feature Type
OffsetPoints	Point
Property	Expression
Origin	OriginPoints
Direction	baseCS.Ydirection
Distance	Stairs_TotalHeight / Stairs_NumberOfSteps

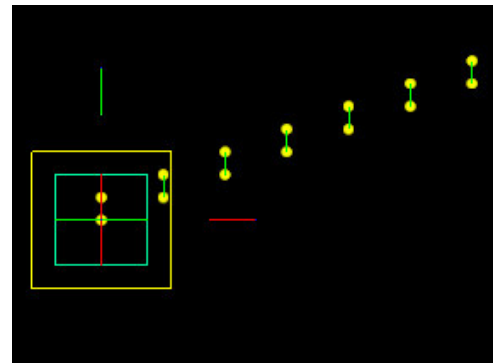


#### 4. Now we can create our step lines.

First of all the Height lines. This one is easy as we just have to match the StartPoint to our OriginPoints (lower) and the Endpoint to our OffsetPoints (top points). Don't forget to take off the bracketed point number so that you apply the lines to all the points (OriginPoints[4] is just a line from point 4, not all the points).

#### Line>ByPoints

Transaction	
[x] Attempt B - Global to Local - Height Lines	
Feature	Feature Type
Heights	Line
Property	Expression
StartPoint	OriginPoints
EndPoint	OffsetPoints



#### 5. And now our horizontal lines.

The Line Lengths we do a bit differently (don't forget to name your line feature "Lengths"..or treads or..).

Transaction	
[-] Lengths	
Feature	Feature Type
Lengths	Line
Property	Expression
StartPoint	OffsetPoints
EndPoint	Subcollection( OriginPoints, Series[ 1, OffsetPoints.Count, 1 ] )

The StartPoint is defined by all our Offset Points (top points).

The End point is defined by the origin points (bottom points) – but how can we tell the line to finish at the next bottom point along? We use a Subcollection.

**Glossary: Subcollection (collection, indices)** creates a collection from specific items obtained from a given collection.

For example you have a prison full of inmates (collection). Inmates can be identified by their unique tag number. We can create a subcollection of inmates by specifying a group of unique tag numbers (indices).

Our collection is the OriginPoints (bottom points). The indices are our criteria for selection. In this case we are going to use a Series to help define our criteria.

**Series(start, finish, increment)** start at 1. Why 1 and not 0? *Our first line starts at the 0 and finishes at OriginPoint 1. We are defining all the end points, the first of which is point 1.* We finish at `offsetPoints.count` ie at the end of all the offset points we stop (however many that may be).

Our increment is 1, as we want a line to end on every origin point on the stairs.

Phew.

Because our start points start at 0 and our endpoints start at one – we achieve the horizontal line going from one point to the next in the step. Ya get it?