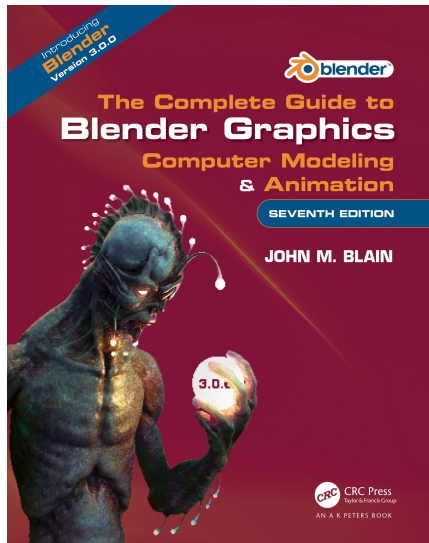


Geometry Nodes

Geometry Nodes in Blender is offered as a **FREE** supplement to:

The Complete Guide to Blender Graphics, 7th Edition



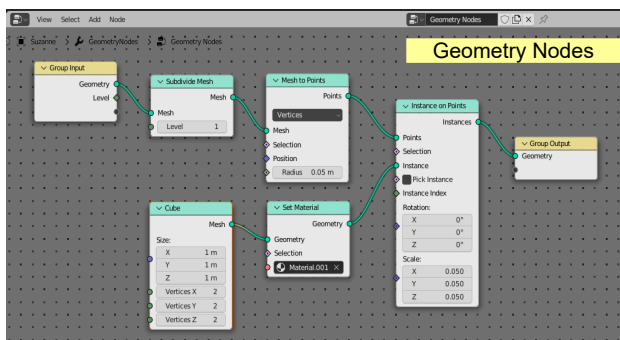
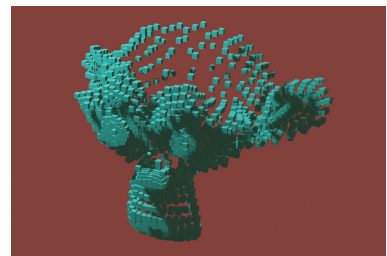
Geometry Nodes are one of the additions to Blender which were incorporated in version number 2.92. At the time of writing the current release of **Blender is 3.0.1**.

Geometry Nodes are a Node System for creating Arrays or Patterns to be used in visual effects or for creating complex shapes.

This topic is continually being developed and even in the course of writing this article have changed significantly. Many of the video tutorials available on the internet providing instruction in using the node system are superseded by developments.

The examples and instruction often make the assumption that the viewer is conversant with Blender and has a reasonably advanced knowledge. A beginner may find the following instruction beneficial

Blender is continually being developed with new features being incorporated and improvements made. In writing an instruction book it is difficult, if not impossible, to keep pace with the developments. The lead time from completing a manuscript to publication to release, prohibits some features from being included. **Geometry Nodes** were not included in the book and are, therefore, offered as a supplement to the book and free to those who are interested.



Suzanne the Monkey constructed as an Array of Cubes using Geometry Nodes

Geometry Nodes – Contents

Introduction.....	I
Contents.....	II
Assumptions.....	1
1.0 Geometry Nodes.....	1
Geometry Node Workspace.....	1
What are Geometry Nodes?.....	2
The Blender Shader Node System.....	2
A Simple Shader Node Arrangement	2
2.0 The Geometry Node Pipeline.....	3
Starting a Pipeline.....	4
How Does the Pipeline Work?.....	7
Connecting Disconnecting and Arranging Nodes.....	7
Minimising Node Display.....	9
Grouping Nodes.....	9
Multiple Pipelines.....	10
The Geometry Node Modifier.....	11
3.0 Spread Sheet Editor.....	12
4.0 Node Selection Menu.....	16
5.0 Editing Shapes.....	17
6.0 Procedural Mesh Objects.....	18
Non Procedural Objects.....	18
Procedural Objects.....	18
7.0 Duplicating and Coloring.....	21
8.0 Minimising Naming and Color Coding.....	21
9.0 Arranging Objects.....	21

10.0 Adding a Subdivision Surface Node	22
11.0 Add Convex Hull	23
Wireframe Display	24
12.0 Materials (Color) for Geometry Nodes	25
Applying the Material	26
13.0 Instance Objects	26
14.0 Bloom - Object Glow	27
15.0 Spiral	28
16.0 Point Cloud and Fields	29
17.0 Creating a Landscape	33
Creating a Ground Plane	34
Appending Objects (Assets)	35
18.0 The Assets Library	37
Creating the Assets Library	37
18.0 Animation	40
19.0 Maths	42
Simple Node Value Rules	43
Socket and Noodle Colors	44
Using Maths	46
20.0 Summary	49

Geometry Nodes

Assumptions

This article assumes the readers have a working knowledge of Blender, and that they are conversant with the **Graphical User Interface** and how the different **Editors** (panels) are arranged into **Workspaces**. In particular it is assumed that the readers know how to Model in Blender, apply Materials (colors) use Shader Nodes and perform Keyframe Animation.

These features are outlined in my book **The Complete Guide to Blender Graphics, 7th Edition**.

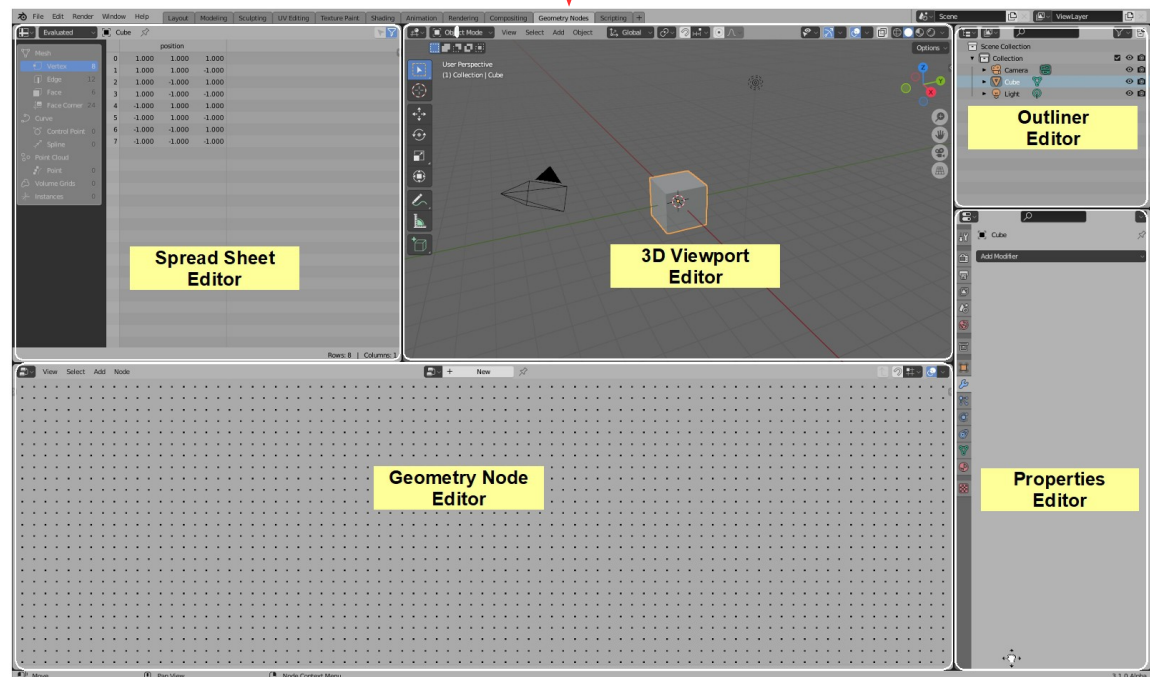
1.0 Geometry Nodes

Geometry Nodes in Blender may be considered as a system for generating Arrays or Patterns which are used as visual effects or for creating complex geometric shapes. The Arrays may be rendered as still images or animated.

Geometry Nodes are accessed in the **Geometry Node Editor** which is part of the **Geometry Node Workspace**.

Geometry Node Workspace

Figure 1.1



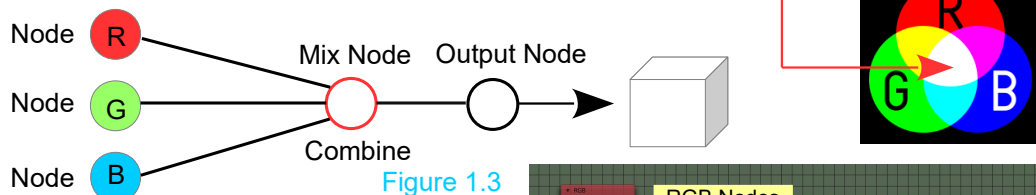
The **Geometry Node Workspace** is accessed by clicking **Geometry Nodes** in the selection menu in the Blender Graphical User Interface Screen Header (Figure 1.1). Alternatively the Workspace may also be configured by dividing and arranging Editors in the default Blender Screen.

What are Geometry Nodes? To answer this question, begin by asking, **What is a Node?**

In Blender and other computer graphics programs a **Node** may be considered a point in a pipeline of information which contributes to a result. In the case of **Material Shader Nodes** the result is the appearance of the surface of an Object in the 3D Viewport Editor. There are usually numerous Nodes connected together producing the result, each of which may be disconnected, rearranged or replaced to vary the final display.

The **Node** is a graphical representation of computer data or instruction which is arranged in a **Pipeline**. Think about mixing colors. The primary colors are Red, Green and Blue, which when mixed in equal proportions produce White (Figure 1.2).

In a **Node System** this would look like: **Figure 1.2** $R + G + B = \text{White}$



The Blender Shader Node system (Figure 1.3)

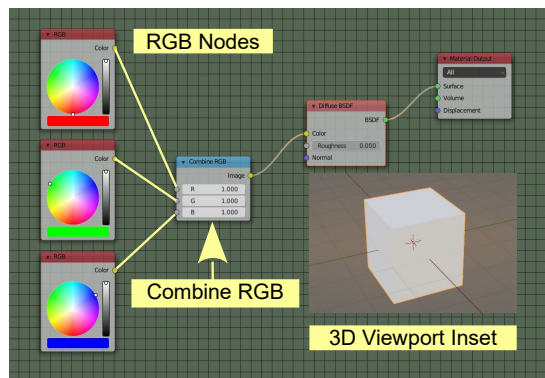
Note: Figure 1.3 is conceptual only (not strictly correct). The RGB Nodes are NOT connected and are represented instead by the **Combine RGB Node**. Red, Green and Blue values all 1.000.

A Simple Shader Node arrangement. **Figure 1.4**

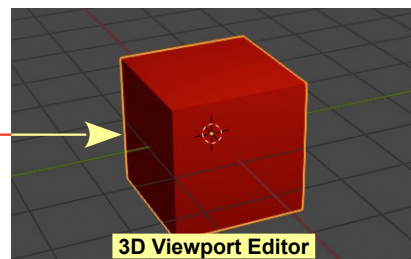
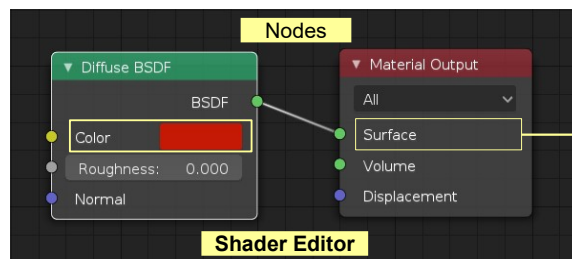
Computer Code

```
shader simple_material(
    color Diffuse_Color = color(0.6, 0.8, 0.6);
    float Noise_Factor = 0.5;
    output closure color BSDF = diffuse(N))
{
    color material_color = Diffuse_Color * mix(1.0, noise(P * 10.0), Noise_Factor);
    BSDF = material_color * diffuse(N);
}
```

Text Editor



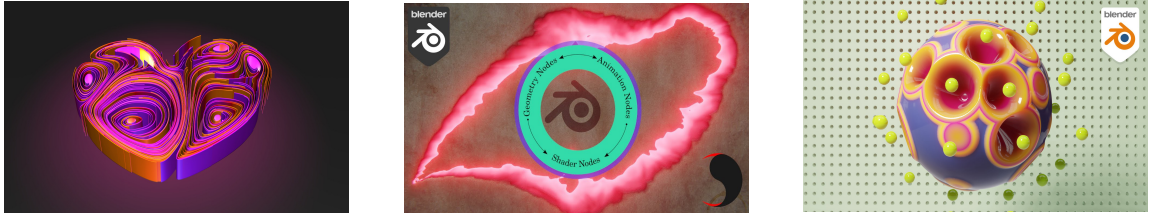
Computer Code written in **Python**, is represented by the **Diffuse BSDF Node**, which outputs data to the **Material Output Node**. This in turn applies Material (color) to the surface of the **Cube** Object in the 3D Viewport Editor.



Individual Geometry Nodes, in essence, are similar to Material Shader Nodes in appearance and the way in which they are arranged and connected in a **Pipeline**. To produce an effect Material Shader Nodes are arranged in the Shader Editor and affect an Object (model) which is selected in the 3D Viewport Editor. **Geometry Nodes** also affect an Object which is selected in the 3D Viewport Editor but they are arranged and connected in the **Geometry Node Editor**.

A reference to the similarities with Shader Nodes may be somewhat misleading. The reference may infer that Geometry Nodes are a different method of modeling and applying Materials but this is not the case. Geometry Nodes are methods for creating visual effects.

Figure 1.5



It has been said that Geometry Nodes change the way environments are made, procedural effects are generated and the way things are scattered around. Procedural assets can be generated for a current project and then reused in future work since a non-destructive workflow is utilised. This means the assets can be modified for new projects. The assets in a current project are easily adjusted when the results are not satisfactory.

2.0 The Geometry Node Pipeline

The Geometry Node Pipeline may be considered as using an Object's Geometry as a basis for generating an **Array**.

A Node in the Pipeline may position an Object in 3D Space as a whole which determines where the Array is placed in the Scene.

A Node may control the position in 3D Space of each of the Objects Vertices which determines the shape of the Array.

The Node may provide a method of adding Vertices to an Object which alters its shape or it may display additional Vertices on the surface of an Object to create an Array.

The Node may provides a method of displaying Vertices as other Objects (instances) which serves to create an **Array of Objects**.

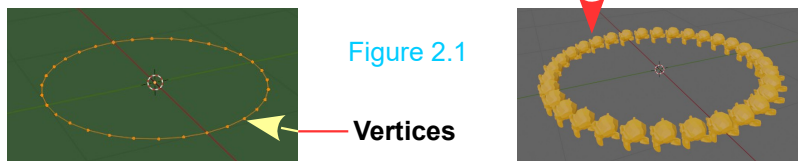


Figure 2.1

Individual Nodes generally contain values which may be edited to affect the display in the 3D Viewport Editor. Individual Nodes are connected in a Pipeline to produce an effect. Note; a Pipeline is sometimes referred to as a **Tree**.

Starting a Pipeline

In the default Blender Screen, with the default Cube Object selected in the 3D Viewport Editor, select the **Geometry Nodes Workspace** in the Blender Screen Header.



Figure 2.2

Selecting **Geometry Nodes** in the Header changes the default Screen arrangement to the Geometry Nodes Workspace (Figure 1.1).

Note that the default Cube Object is selected in the 3D Viewport Editor as indicated by its orange outline. The Geometry Node Editor is empty except for the **New Button** in the Header. The **Spread Sheet Editor** is showing data which is the location in 3D Space of each Vertex making up the Cube Object (more on this to follow).

The procedure in creating a **Node Pipeline** is to click the **New Button** in the Geometry Node Editor Header to introduce a default Node arrangement, then Add Nodes and connect the Nodes to the Pipeline.

A Node Pipeline may be associated with any Object which is selected in the 3D Viewport Editor.

Note: If the default Cube Object In the default Blender Scene is deleted, and the Geometry Node Workspace is entered, you will find that the New Button in the Geometry Node Editor Header does **NOT** display. The New Button only displays when a new Object is entered in the Scene. A new Object entered in a Scene, by default, **is selected** when entered. In other words, an Object must have been selected in the 3D Viewport Editor before the New Button displays.

To demonstrate; in a new Blender Scene **delete the default Cube**. Open the Geometry Node Workspace. The **New Button** does **NOT** display in the Geometry Node Editor Header. In the 3D Viewport Editor, add a **Cone Object**. The New Button is reinstated in the Geometry Node Editor Header.

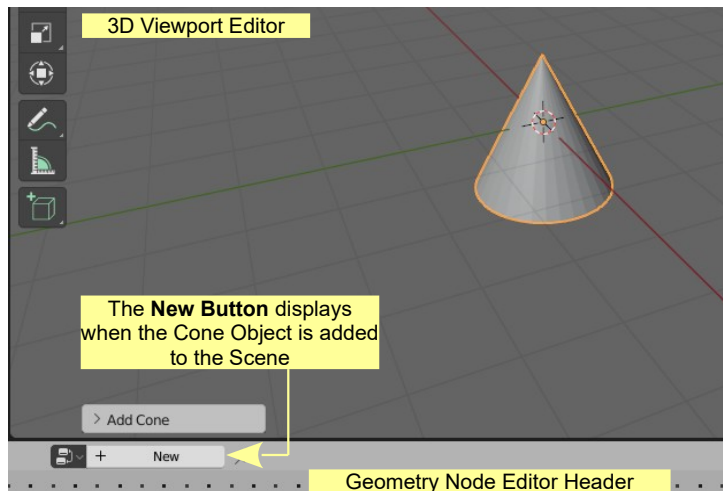
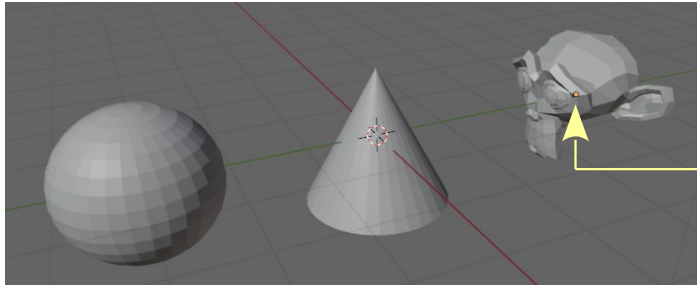


Figure 2.3

Deselect the Cone then add a **UV Sphere** followed by a **Monkey** (Suzanne) separating each along the Y Axis and deselecting each in turn.



Orange dot shows the last Object to be selected.

Figure 2.4

With all three Objects deselected, click the **New Button** in the Geometry Node Editor Header to introduce a **Node Pipeline**. The Pipeline will consist of a **Group Input Node** and a **Group Output Node**. The two Nodes are connected by a **Noodle** (green line) from the Group Input, Geometry Socket to the Group Output Geometry Socket. **Note:** The name **Geometry Nodes**.

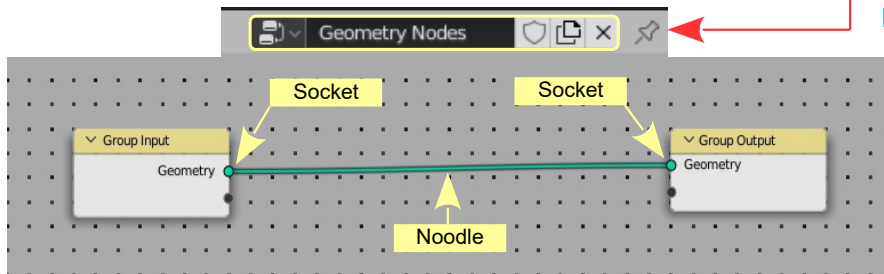


Figure 2.5

Note: None of the Objects are selected in the 3D Viewport Editor. In the Geometry Node Editor Header click **Add** to display the Node Selection Menu. Mouse over on the **Geometry** category then in the sub menu click **Transform**.

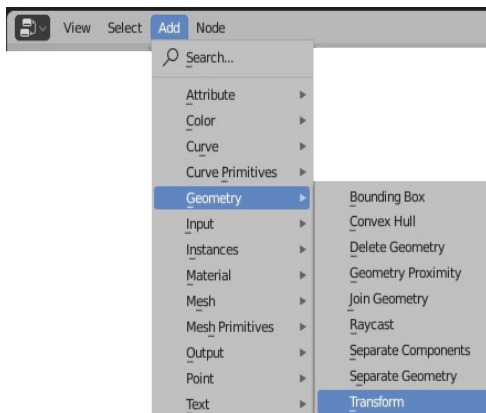
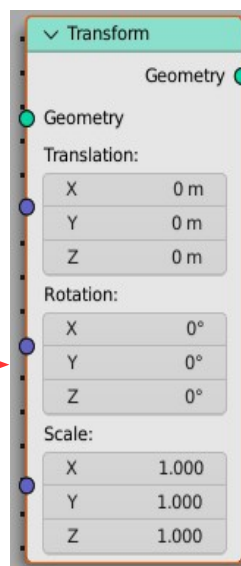


Figure 2.6

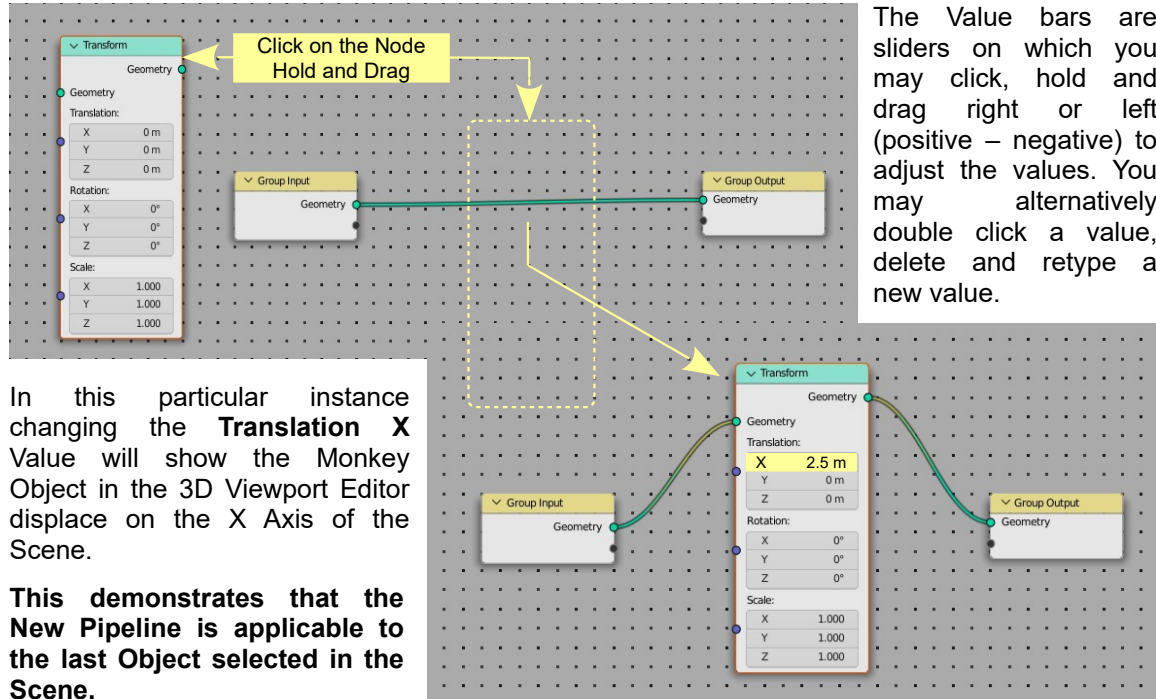


A **Transform Node** displays in the Geometry Node Editor. The Node is NOT connected.

Click on the Node, hold and drag the mouse, placing the Node over the green Noodle, between the Input and Output Nodes. This automatically connects the Transform Node in the Pipeline.

Note: If the new Node were incompatible in the Pipeline the Noodle would display red.

In the Transform Node you will see Translation, Rotation and Scale values. [Figure 2.7](#)

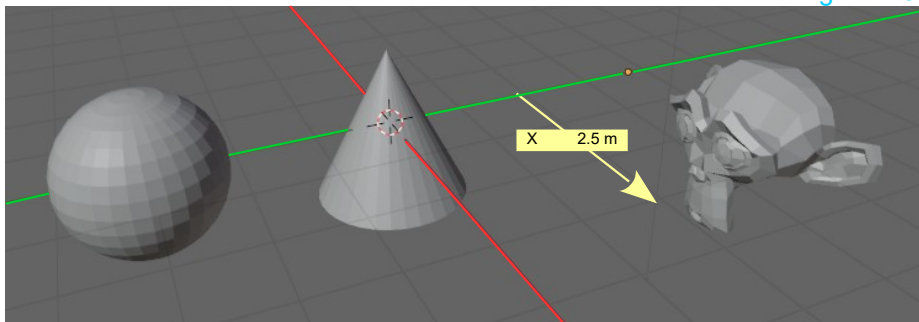


In this particular instance changing the **Translation X** Value will show the Monkey Object in the 3D Viewport Editor displace on the X Axis of the Scene.

This demonstrates that the New Pipeline is applicable to the last Object selected in the Scene.

Note: The last Object that was **selected**, is not necessarily the last Object **entered**.

[Figure 2.8](#)



Note: In the Geometry Node Editor Header, when the New Button was clicked, the **New Button** was replaced by the name **Geometry Nodes**. This is the name of the Pipeline affecting the Monkey Object.

Selecting one of the other Objects in the 3D Viewport Editor shows the name **New Button** again.

How Does the Pipeline Work?

With the very basic **Node Pipeline** consisting of the **Group Input Node** connected to the **Group Output Node**, the Input Node represents the data for displaying the selected Object. This data is transmitted to the Group Output Node which affects the display of the Object in the 3D Viewport Editor. Introducing the **Transform Node** and connecting it in the Pipeline allows that original Input Data to be modified thus affecting the display in the 3D Viewport Editor.

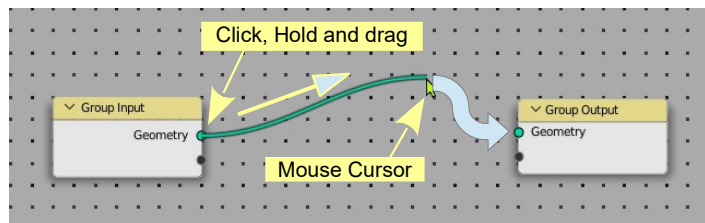
Connecting Disconnecting and Arranging Nodes

It has been demonstrated that Nodes will automatically be connected when they are dragged over a connecting Noodle (Figure 1.12). Nodes may be arranged to suit user preferences by clicking on the Node to select it (a thin red outline is displayed) then holding LMB and dragging the Node to the desired location. If you wish to connect a Node between two other Nodes and the space on the Screen is too small, the space automatically adjusts. At times you may wish to delete a Node in which case, with the Node selected (red outline) press **X Key**.

If you delete the **Transform Node** shown in Figure 1.12 you will be left with the Group Input Node and the Group Output Node **disconnected** (Figure 1.14). With the Nodes disconnected, Suzanne (the last Object selected in the 3D Viewport Editor) disappears from view.

Figure 2.9

To reconnect the Input and Output Nodes, click on the green Group Input, Geometry Socket, hold and drag over to the Group Output green Geometry Socket.



To disconnect Nodes there are two options:

Figure 2.10

Click on the **Input Socket** of a Node (right hand end of the Noodle) hold and drag away from the Socket.

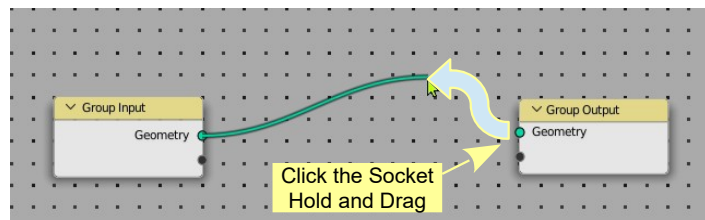
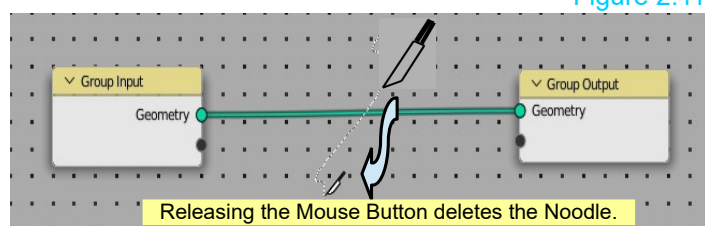


Figure 2.11

Hold **Ctrl** on the Keyboard, click RMB hold and drag the **Knife** tool over the Noodle.



Note: If you click **Shift** on the Keyboard, click and drag the Cursor over the Noodle a new connection Socket is created on the Noodle.

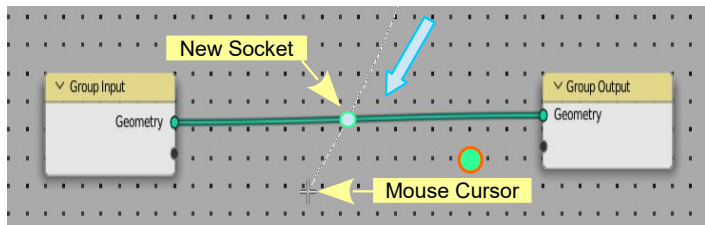
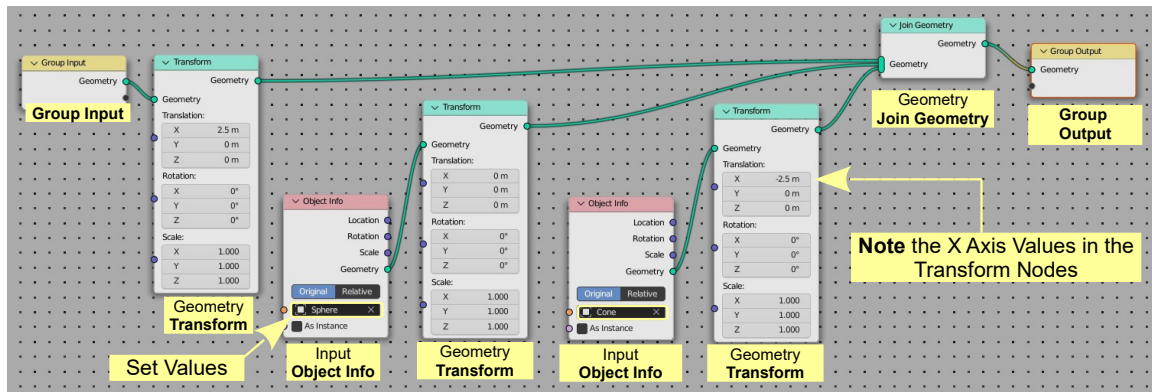


Figure 2.12

Adding and arranging Nodes creates a Node Tree. Figure 2.13 shows a **Simple Node Tree**. The adjective **Simple** must be emphasised since Node Trees can be very complex and have to be organised.

Nodes in the Pipeline - Node Tree

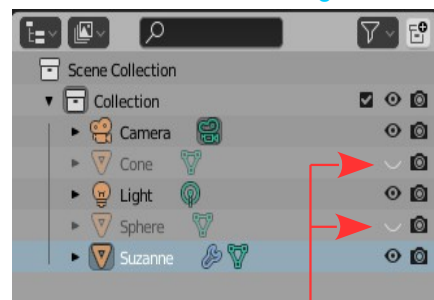
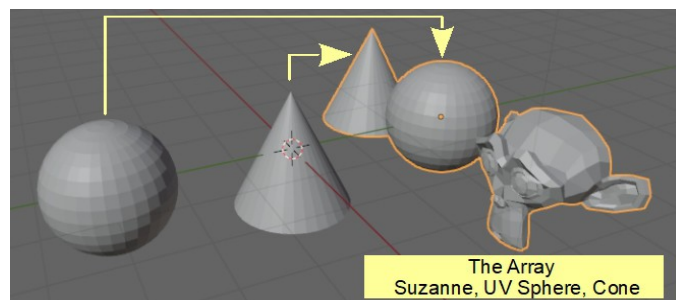
Figure 2.13



Remember: The objective in creating a Geometry Node system is to use an Object for creating arrays or effects. The Pipeline created generates an Array around Suzanne (last Object selected).

In Figure 1.18 the **Group Input Node** transmits data pertaining to the last Object selected (Suzanne) to the **Transform Node** where the data may be modified before being sent on to the **Group Output Node**. Two **Object Info Nodes** have been added connected to additional **Transform Nodes**. The **Object Info Nodes** have been set to gather data from the UV Sphere and Suzanne respectively. This data is transmitted to the additional Transform Nodes. All three Transform Nodes are connected via a **Join Geometry Node** to the **Group Output Node**. The X Axis values in the Transform Nodes determine the position of Objects in the Array.

Figure 2.14

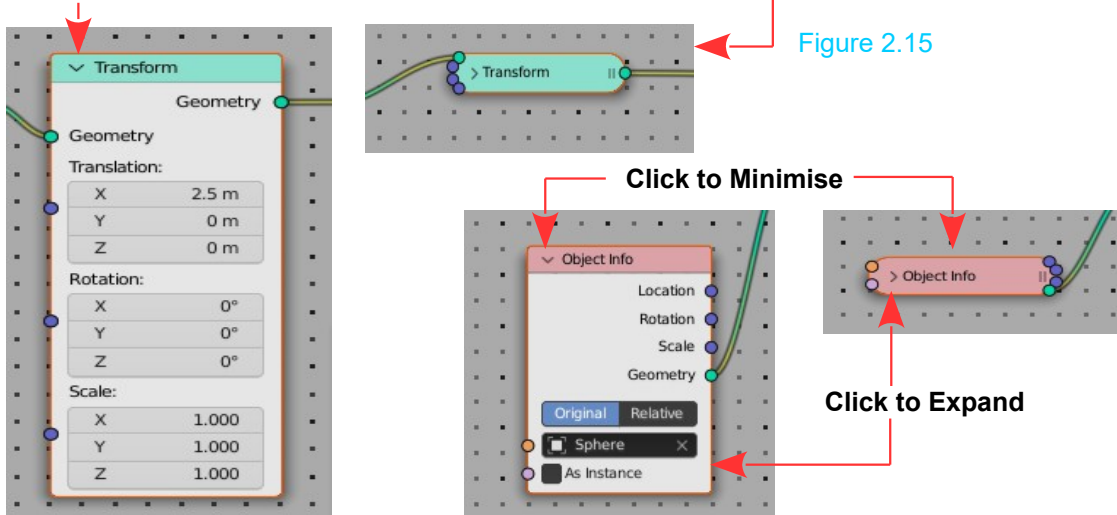


Click the **Eye Icons** in the **Outliner Editor** to hide the original Sphere and Cone.

A complex Pipeline can consume Screen space, therefore, individual Nodes may be collapsed to minimise space. Nodes can be grouped to make it easier to distinguish components of the Tree and the **Groups** themselves may be minimised.

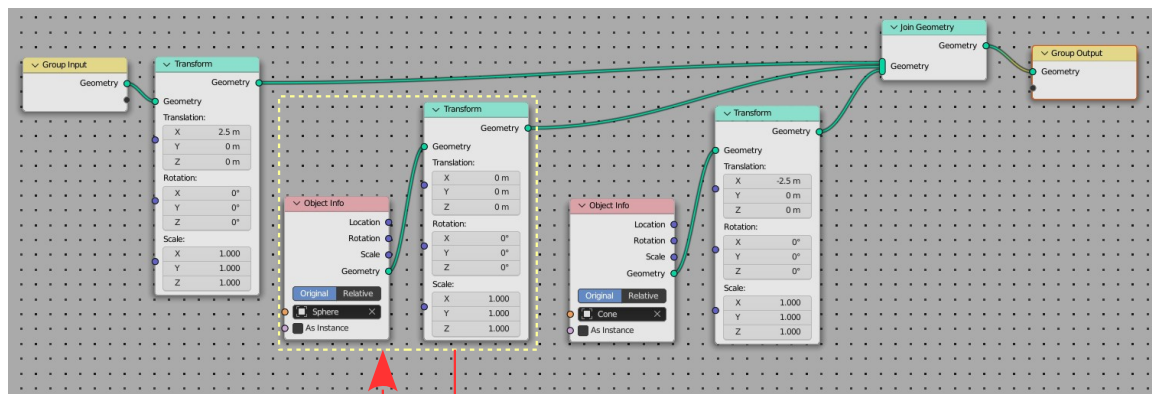
Minimising Node Display

In Figure 2.15 the **Transform** and **Object Info** Nodes may be minimised by clicking the chevron adjacent to the Node Name.



Grouping Nodes

Figure 2.16

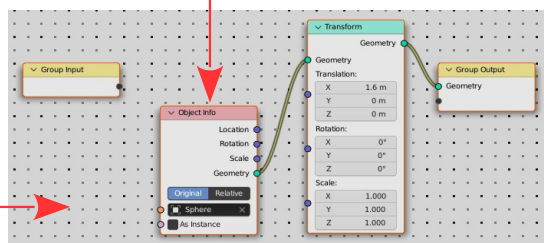


Shift select or **Box select (B Key)**, the Nodes you wish to be included a **Group**.

Figure 2.17

Hold **Ctrl** on the Keyboard and press the **G Key** (Group).

The display changes showing only the Group

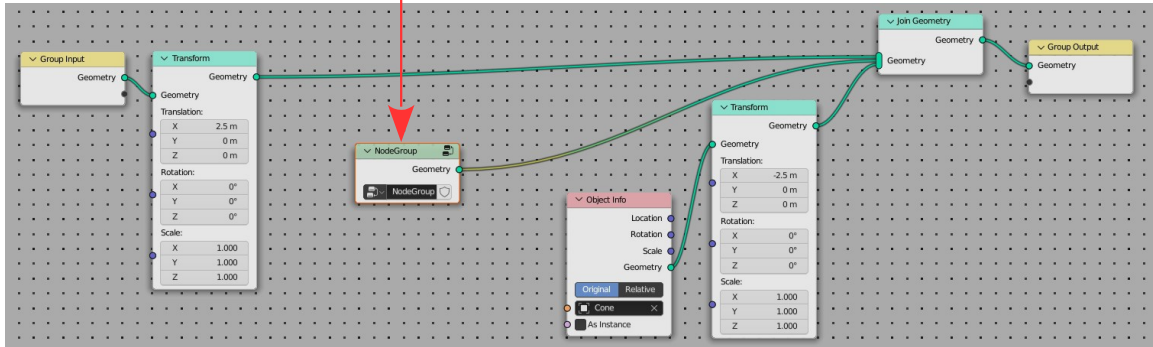


The **Group Input** and **Output** Nodes remain displayed as does the **Join Geometry** Node. All Noodle connections remain in situ.

Remember: The Node Pipeline is associated with the last Object selected in the Scene.

Select either of the Objects in the **Outliner Editor** followed by **Suzanne** to see the Node Group minimised in the Node Tree.

Figure 2.18



In minimising the Node Group you lose the ability to amend values in the Transform Node. Select the minimised Node Group, RMB click and select **Ungroup** in the menu that displays. The Nodes are reinstated where you adjust values then RMB click and select Group again.

Multiple Pipelines

Multiple Pipelines may be created by selecting each Object in the Scene and pressing the New Button in the Geometry Node Editor Header. UV Sphere and Cone in turn. The automatic Pipeline Names will be **Geometry Nodes.001** and **Geometry Nodes.002**.

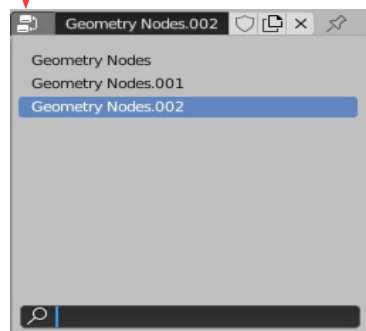
Figure 2.19

Node Tree to be Linked

By clicking the **Node Tree to be Linked Button** you will see the three Node Pipeline names. The data for each Pipeline is stored in a Cache.

If you add a fourth Object to the Scene, then click the New Button in the geometry Node Editor Header, then click the Browse Node tree to be linked button and select (click) one of the stored Pipelines, you apply an instance of that Pipeline (tree) to the new Object.

Note: You may rename the Pipelines to something meaningful to your project.



The foregoing is intended to convey the basic process for creating an Array using the Geometry Node Editor. Understanding the method will assist when following demonstrations.

Before studying examples it should be realised that creating a Node Tree or Pipeline adds a Modifier to the selected Object.

The Geometry Node Modifier

Figure 2.20

When a Pipeline is created in the Geometry Node Editor a Modifier is added in the Properties Editor, Modifier Properties. It follows that, Geometry Nodes are, therefore, Modifiers and that they modify the Geometry of the selected Object.

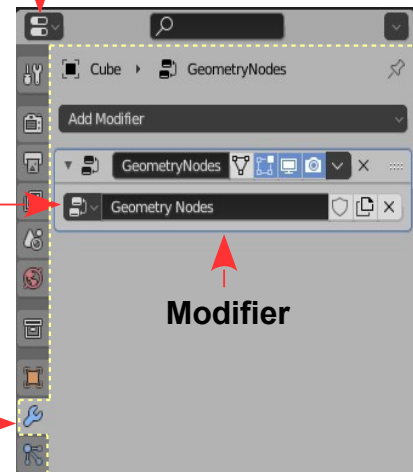
Note: The Pipeline is applicable to the selected Object in the 3D Viewport Editor (the Cube).

In the first instance the name of the **Modifier** is **Geometry Nodes**.

Note: Geometry Node Modifiers, unlike traditional Modifiers are not Applied directly to an Object.

Modifier Properties

Properties Editor



Geometry Node Modifiers are, however, transferable to other Objects which are entered in the 3D Viewport Editor.

For example: Consider the arrangement of the Cone, UV Sphere and Monkey Objects previously described.

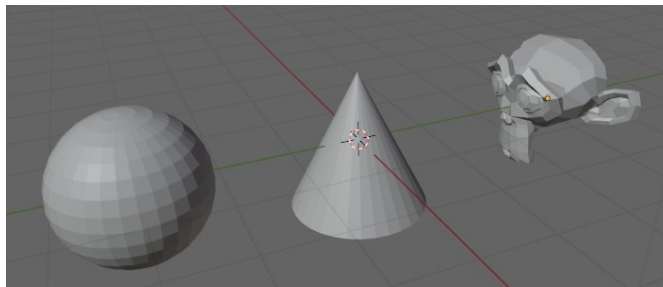


Figure 2.21

Selecting the Suzanne (Monkey) and creating the Pipeline shown in Figure 2.18 adds the Modifier named **GeometryNodes** (Figure 2.20). The Modifier is added in the Properties Editor, Modifier Properties immediately on selecting Suzanne and pressing the New Button in the Geometry Node Editor Header. Adding a Transform Node to the Pipeline incorporates the Transform Attribute in the Modifier. Adjusting the X Axis rotation value in the transform Node causes Suzanne to rotate in the 3D Viewport Editor. The Cone and the UV Sphere are not affected.

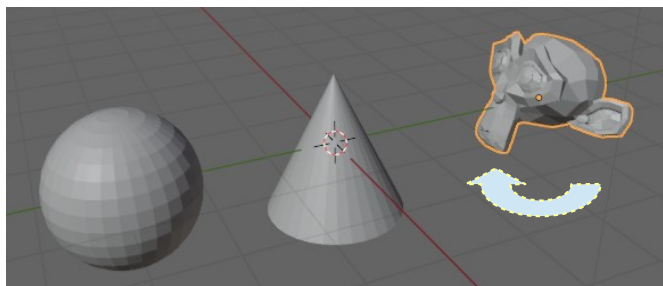
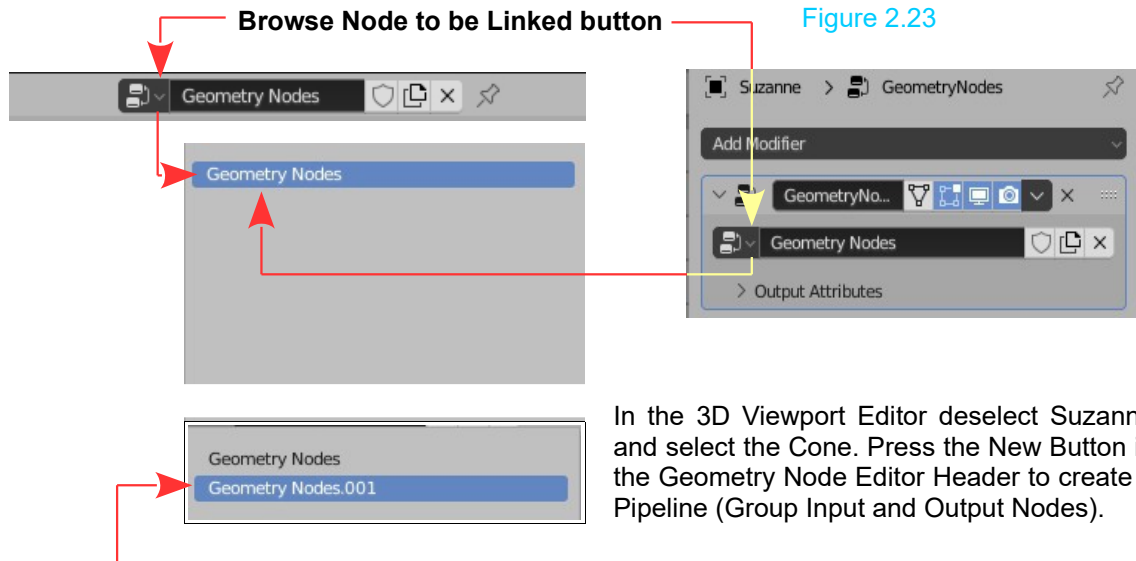


Figure 2.22

Clicking the **Browse Node to be Linked button** in the Geometry Node Editor Header or in the Modifier Properties shows the single Modifier named **Geometry Nodes**.



In the 3D Viewport Editor deselect Suzanne and select the Cone. Press the New Button in the Geometry Node Editor Header to create a Pipeline (Group Input and Output Nodes).

You will now have a new Modifier named **Geometry Nodes.001** in the Modifier Properties and in the Geometry Node Editor Header. Instead of adding Nodes to the Pipeline, click the **Browse Node Tree to be Linked button** and select **Geometry Nodes** in the drop down menu to assign the Modifier named **Geometry Nodes** to the Cone. The Cone is rotated in the 3D Viewport.

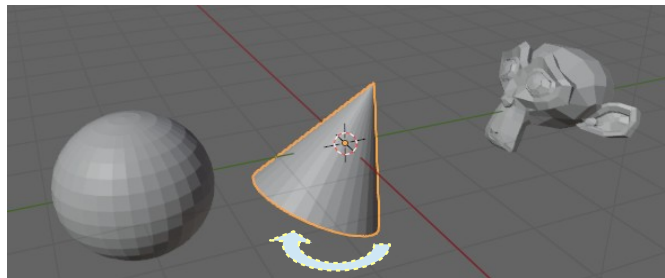


Figure 2.24

3.0 Spread Sheet Editor

When introducing Geometry Nodes it has been shown how a Pipeline is created in the Geometry Node Editor which affects the Object in the 3D Viewport Editor. Objects included in the Scene are listed in the Outliner Editor where they may be selected and hidden from view (Figure 1.18). The Geometry Node Modifier is shown in the Properties Editor. The fifth Editor in the Geometry Node Workspace is the **Spread Sheet Editor** (Figure 3.1).

The Spread Sheet Editor displays coordinates for different aspects of the selected Object in the 3D Viewport Editor i.e. the position in 3D Space of Vertices, Edges, Faces and Face Corners.

Note: You **Cannot** edit coordinate values in the Spread Sheet.

Spread Sheet Editor

Figure 3.1

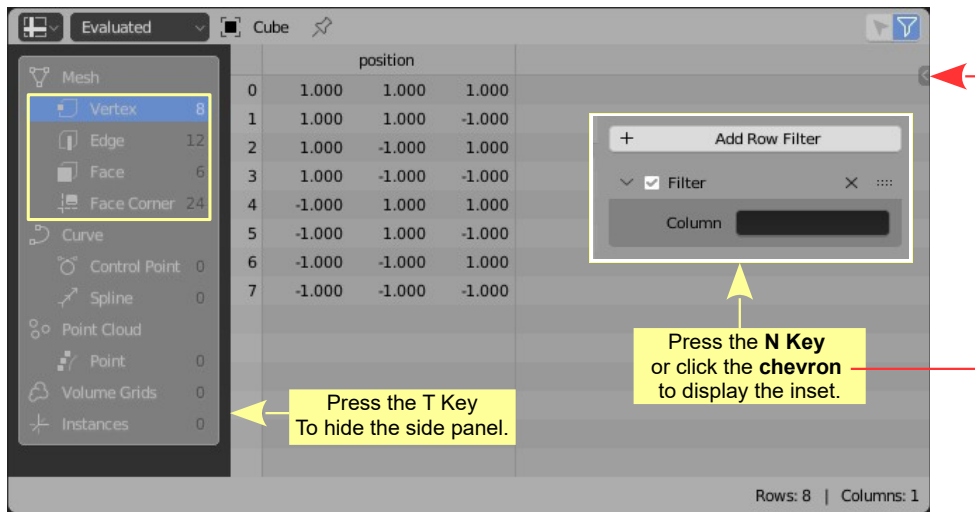
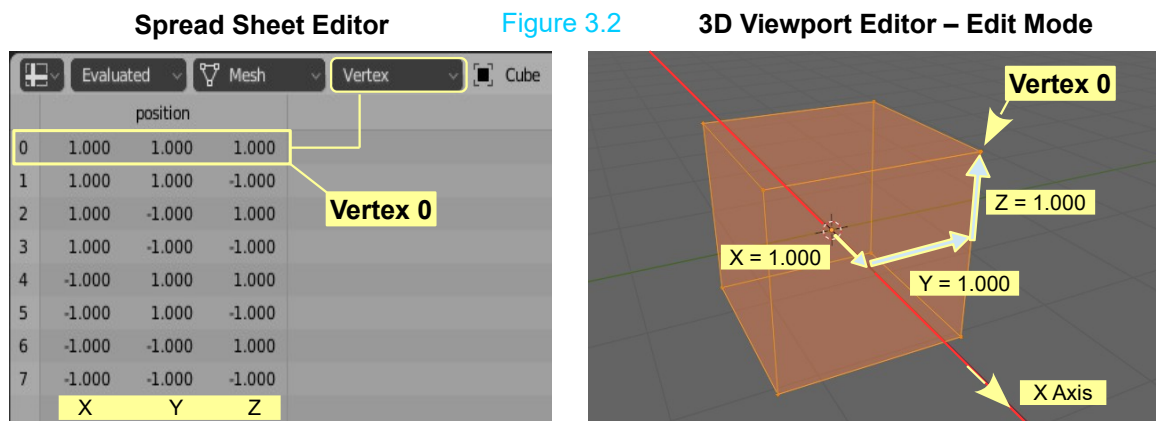


Figure 3.1 shows the default **Spread Sheet Editor** when the Cube Object is selected in **Object Mode** in the **3D Viewport Editor**.

Note: the **Geometry Node Editor** is empty.

The **Spread Sheet Editor** has a column headed **position** containing the location of each of the Cube's Vertices in 3D Space. Note: The Cube in the 3D Viewport Editor is in Object Mode. Changing the 3D Viewport Editor to Edit Mode shows the Cube's Vertices which relate to the values in the Spreadsheet.



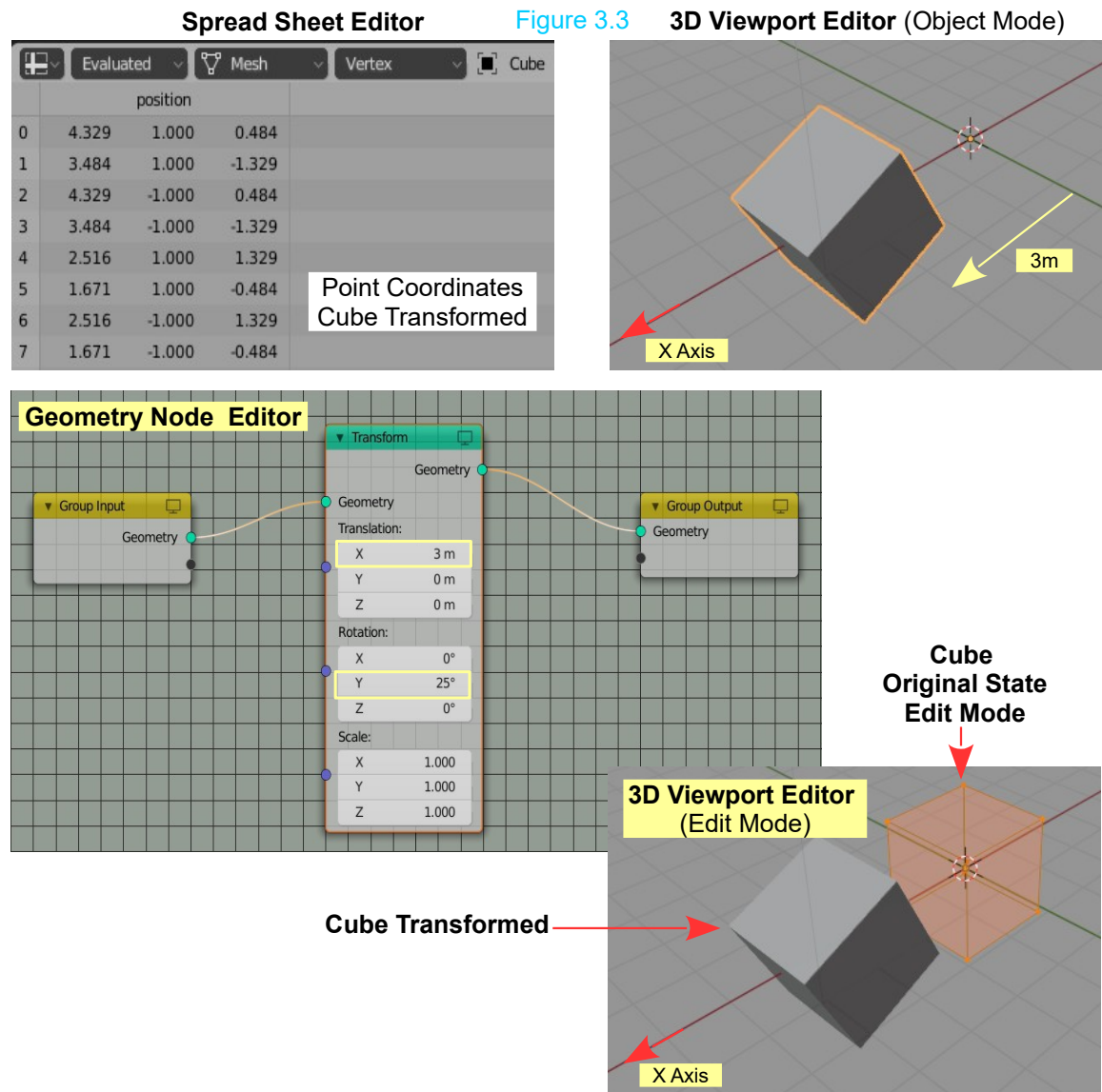
The Spread Sheet Editor shows the XYZ coordinates of the Cube's Vertices in 3D Space. Eight Vertices numbered 0 to 7. Note: In Geometry Nodes, Vertices are considered to be **Points**.

When an Object is Transformed in the 3D Viewport Editor by editing values in the Transform Node in the Node Editor you see the Point coordinates change in the Spread Sheet Editor. You see the Transformation of the Object in the 3D Viewport Editor in Object Mode. By changing the 3D Viewport Editor to Edit Mode you will see the Object in its original state.

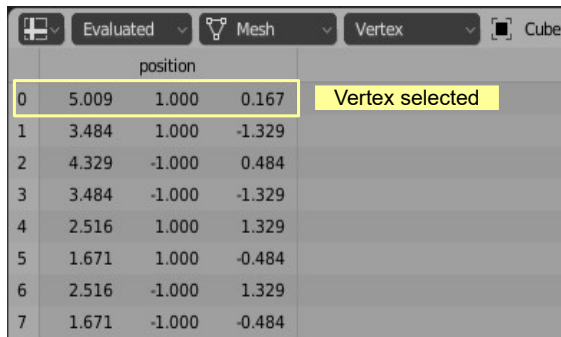
To clarify this concept consider the following:

In a new Scene in the Geometry Node Workspace with the default Cube selected in the 3D Viewport Editor add a Geometry Transform Node to the Pipeline in the Geometry Node Editor.

In the Transform Node change the X Translation value to 3m and the Y Rotation to 25°.

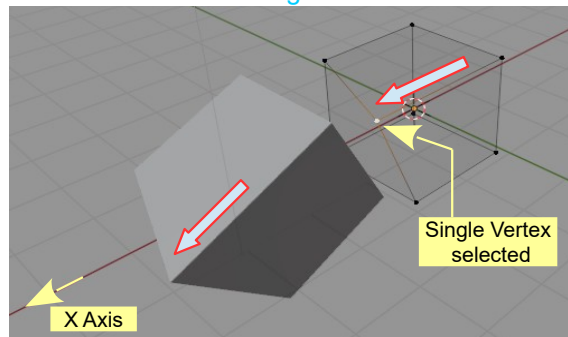


With the 3D Viewport Editor in Edit Mode, select a single Vertex (Vertex Point 0 – Figure 3.4). Translate the Vertex on the X Axis. You see the translation occur on the Rotated Cube and you see the Coordinate Values change in the Spread Sheet Editor.



	position		
0	5.009	1.000	0.167
1	3.484	1.000	-1.329
2	4.329	-1.000	0.484
3	3.484	-1.000	-1.329
4	2.516	1.000	1.329
5	1.671	1.000	-0.484
6	2.516	-1.000	1.329
7	1.671	-1.000	-0.484

Figure 3.4



With the single Vertex selected in the 3D Viewport Editor you may check (tick) **Selected Only** in the upper RH corner of the Spread Sheet Editor to isolate the Point coordinates for the selected Vertex.

Figure 3.5

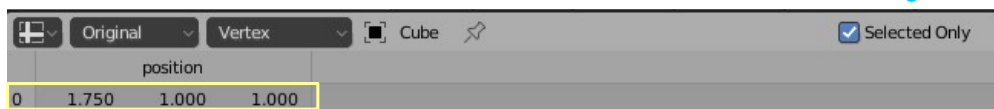


	position		
0	5.009	1.000	0.167

Evaluated Coordinates for the selected Vertex **Point 0**

In the **Spread Sheet Header** change **Evaluated** to **Original** to see the Point Coordinates for the Vertex prior to Translating on the X Axis.

Figure 3.6

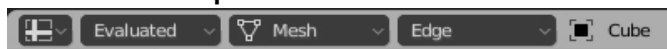


	position		
0	1.750	1.000	1.000

Note: In the Spread Sheet Editor Header you may select the options, **Vertex**, **Edge**, **Face** and **Face Corners** to view coordinates for each. These correspond to the selection modes in the 3D Viewport Editor in Edit Mode.


Figure 3.7

Spread Sheet Header



Vertex
Edge
Face
Face Corner
Attribute Domain

3D Viewport Edit Mode Header



Vertex – Edge - Face

Face Corners: For a Cube – 6 Faces times 4 Corners on each Face = 24 Faces. Face Corners numbered 0 to 23 in the Spread Sheet Editor.

The **Geometry Transform Node** has been employed to demonstrate the basic concept of the Node Editor being used in conjunction with the Spread Sheet Editor and the 3D Viewport Editor.

4.0 Node Selection Menu

Search... **Node** **Shortcut: Shift A**

- Attribute**
 - Attribute Statistic
 - Capture Attribute
 - Transfer Attribute
- Input**
 - Boolean
 - Collection Info
 - Color
 - Integer
 - Is Viewport
 - Material
 - Object Info
 - String
 - Value
 - Vector
- Color**
 - Color Ramp
 - Combine RGB
 - Mix RGB
 - RGB Curves
 - Separate RGB
- Curve**
 - Curve Length
 - Curve to Mesh
 - Curve to Points
 - Fill Curve
 - Fillet Curve
 - Resample Curve
 - Reverse Curve
 - Sample Curve
 - Subdivide Curve
 - Trim Curve
- Geometry**
 - Bounding Box
 - Convex Hull
 - Delete Geometry
 - Geometry Proximity
 - Join Geometry
 - Raycast
 - Separate Components
 - Separate Geometry
 - Transform
- Mesh Primitives**
 - Cone
 - Cube
 - Cylinder
 - Grid
 - Ico Sphere
 - Mesh Circle
 - Mesh Line
 - UV Sphere
- Output**
 - Viewer
- Point**
 - Distribute Points on Faces
 - Points to Vertices
 - Points to Volume
 - Set Point Radius
- Text**
 - Join Strings
 - Replace String
 - Slice String
 - Special Characters
 - String Length
 - String to Curves
 - Value to String
- Vector**
 - Combine XYZ
 - Separate XYZ
 - Vector Curves
 - Vector Math
 - Vector Rotate
- Material**
 - Replace Material
 - Material Index
 - Material Selection
 - Set Material
 - Set Material Index
- Utilities**
 - Align Euler To Vector
 - Boolean Math
 - Clamp
 - Compare Floats
 - Float Curve
 - Float to Integer
 - Map Range
 - Math
 - Random Value
 - Rotate Euler
 - Switch
- Texture**
 - Brick Texture
 - Checker Texture
 - Gradient Texture
 - Image Texture
 - Magic Texture
 - Musgrave Texture
 - Noise Texture
 - Voronoi Texture
 - Wave Texture
 - White Noise
- Curve Primitives**
 - Bezier Segment
 - Curve Circle
 - Curve Line
 - Curve Spiral
 - Quadratic Bezier
 - Quadrilateral
 - Star
- Mesh**
 - Mesh Boolean
 - Mesh to Curve
 - Mesh to Points
 - Split Edges
 - Subdivide Mesh
 - Subdivision Surface
 - Triangulate
 - Is Shade Smooth
 - Set Shade Smooth
- Group**
 - Make Group
 - Ungroup
 - Group Input
 - Group Output
- Layout**
 - Frame
 - Reroute
- Volume**
 - Volume to Mesh

Note: The content of the selection menu may vary since Nodes are continually being developed and amended. This snapshot was taken for Blender 3.0.0 (2 January 2022).

4.0 Node Selection Menu

By navigating through the Add Menu in the Node Editor Header you will discover a selection of Nodes (see diagram included). This paper is intended to demonstrate the basic concept of using Geometry Nodes not to provide instruction in the use of every Node. With an understanding of the basic concept you will be better placed to follow the many video tutorials available on the internet. As you experiment with the use of Geometry Nodes you will discover Node Systems for particular applications. As you discover a system save a Blender file with the Node arrangement for future use thus compiling a library. The systems may be suitable for future applications or may be a starting point for modifications and used in your work.

5.0 Editing Shapes

Editing the shape of an Object in the 3D Viewport Editor when using Geometry Nodes is performed to create a shape for an Array or Pattern. How you shape the Object determines how the Array displays in the 3D Viewport Editor. Bear in mind, what you see in the 3D Viewport Editor, should be considered with respect to Camera View. After all, Camera View shows what will be the final result and this is the ultimate goal. A shape for an Array can be a shape for a complex model such as plant leaves or flower petals.

As an example the simple Array of Monkey Heads shown in Figure 1.6 is created by instancing Objects on the Vertices of a Circle Object. The Circle could be reshaped to represent a plant stem and in place of Monkey Heads you would use a model of a leaf.

To continue; you should be aware that the different Nodes affect the display of the Object in the 3D Viewport Editor in different ways. The Geometry Transform Node has been shown affecting the Cube Object as a whole. To modify the Cube's shape, Vertices in the 3D Viewport Editor were shown to be manipulated in the 3D Viewport Editor in Edit Mode.

Another factor to consider is; to use Geometry Nodes a Mesh Object or a Curve Object must be entered in the 3D Viewport Editor as a starting point. You can not create a Node Tree without the base Object. Objects such as Empty Objects, Surfaces, Cameras and Armatures will not work.

When employing Mesh Objects in conjunction with Geometry Nodes there are two types of Objects to be considered; Procedural Objects and Non-Procedural Objects.

Explaining Procedural Objects will be limited to purely describing the mechanics of using an Object with Nodes in relation to the Graphical User Interface.

To understand what a Procedural Object is you would have to understand Procedural Generation in Computer Graphics and its application when developing Computer Games. This is beyond the scope of this discussion but in a nutshell "procedural" and "generation" imply that you are dealing with computer procedures, or algorithms, that create something and perhaps incorporate AI (Artificial Intelligence). Such procedures are particularly applicable when creating Computer Games.

At this point consider that the use of Geometry Nodes in conjunction with Procedural Mesh Objects as employing computer procedures through the use of Geometry Nodes.

6.0 Geometry Nodes – Procedural Mesh Objects

When using Geometry Nodes it is important to distinguish between **Procedural** and **Non-Procedural** Mesh Objects.

Non Procedural Objects

Geometry Nodes have been demonstrated, thus far, using a Mesh Object by employing one of the Blender's Primitives.

The basic procedure has been to enter an Object in the 3D Viewport Editor and with the Object selected, generate a Node Pipeline in the Geometry Node Editor.

With the Node Pipeline generated the **Group Input Node** obtains data from the selected Object and creates the display in the 3D Viewport Editor, secretly superseding the original display.

Nodes are added and connected in the Pipeline which modifies the display of the Mesh Object.

Figure 6.1

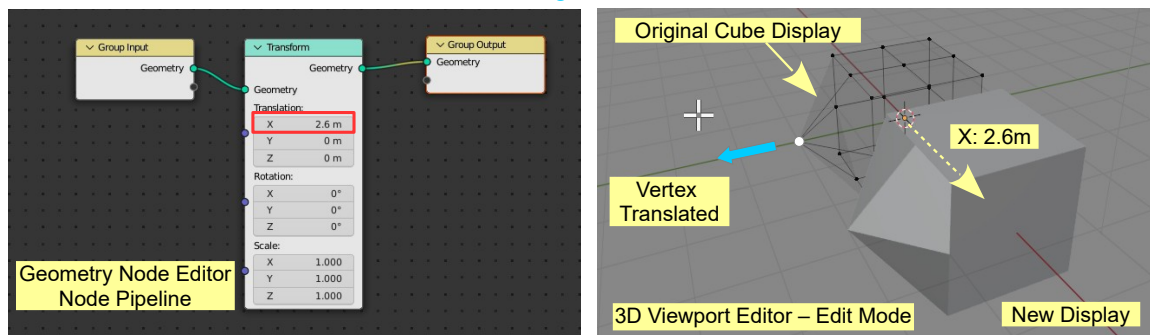


Figure 6.1 shows the Node Pipeline in the Geometry Node Editor producing the display in the 3D Viewport Editor. The Transform Node has moved the new Cube display 2.6 m on the X Axis. The data for the original Cube displays in the 3D Viewport Editor (in Edit Mode) in the original position. The Cube has been subdivided. Translating a single Vertex on the original alters the new display which demonstrates that **Non-Procedural Objects** may be edited in Edit Mode.

Disconnecting or deleting the Node Pipeline will show the edited Cube in its original default position at the center of the Scene.

Procedural Objects

A **Procedural Object** is introduced to a Scene by adding one of the **Mesh Primitive Nodes**.

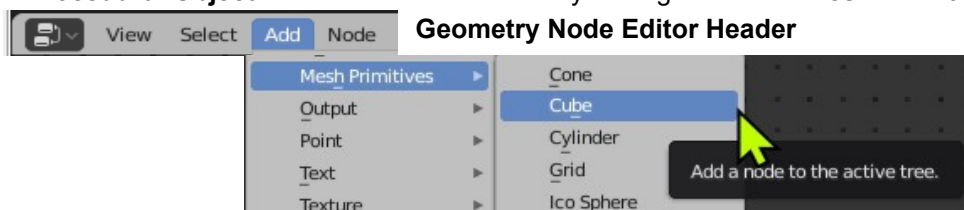


Figure 6.2

Placing a **Cube Mesh Primitive Node** in the Pipeline will not produce a display in the 3D Viewport Editor until the red Noodle is disconnected. A red Noodle indicates that the connection is invalid.

**Invalid Connection
No Display**

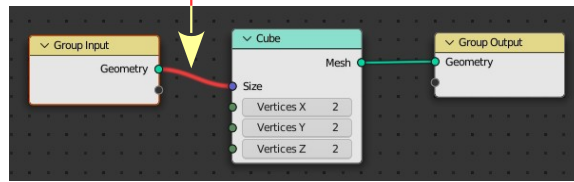
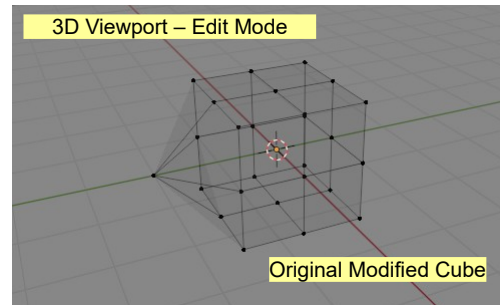
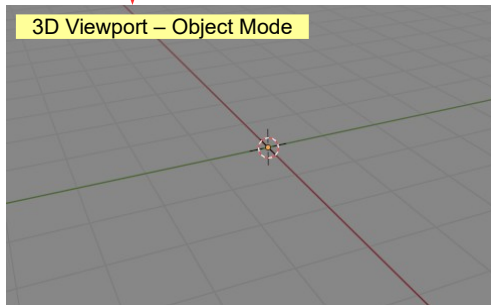
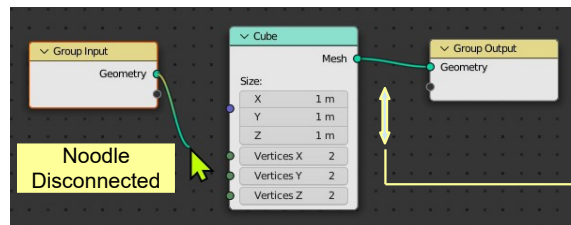


Figure 6.3

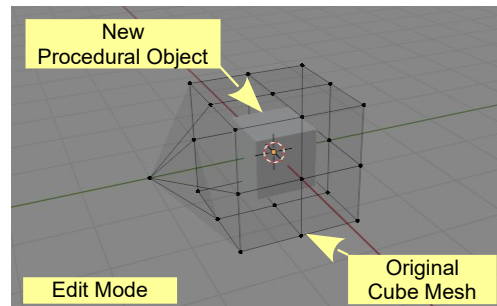
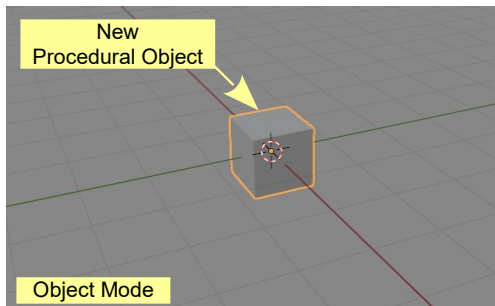


Following on from the previous example (Figure 6.3) shows that the original modified Cube Object remains with the 3D Viewport Editor in Edit Mode. There is no display in Object Mode and consequently nothing Renders. This demonstrates that the data for the original Cube Object remains.



The Procedural Cube Node expands when the invalid noodle is disconnected.

Figure 6.4



Disconnecting the invalid connection displays the new Procedural Object (Cube) in the 3D Viewport Editor in Object Mode and in Edit Mode. In Edit Mode the new Cube is encapsulated by the original Cube Mesh which again demonstrates that the original Cube data remains in the Scene.

The new Procedural Cube display is being generated by the computer code contained in the Mesh Primitive, Cube Node. The original Non Procedural Cube display is generated by Blender's internal coding. The Procedural Cube supersedes the original.

Note: Editing the original Cube, in Edit Mode, has no effect on the new Procedural Cube. A rendered view of the Scene captured by the Camera only displays the Procedural Cube.

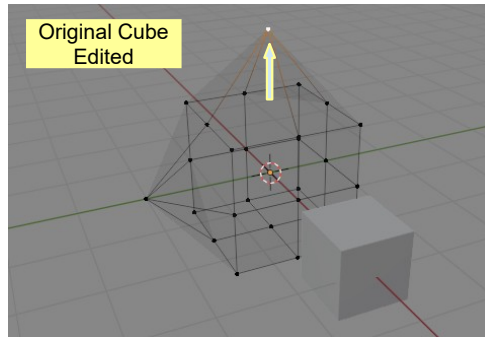
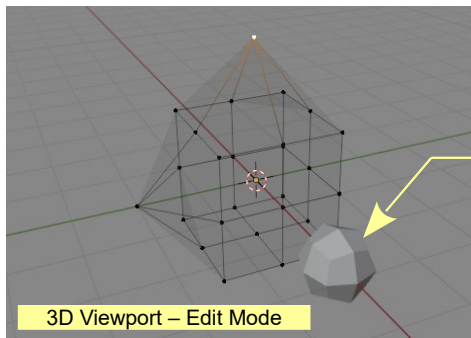


Figure 6.5

To edit the Procedural Cube, Nodes are entered in the Node Pipeline in the Geometry Node Editor.

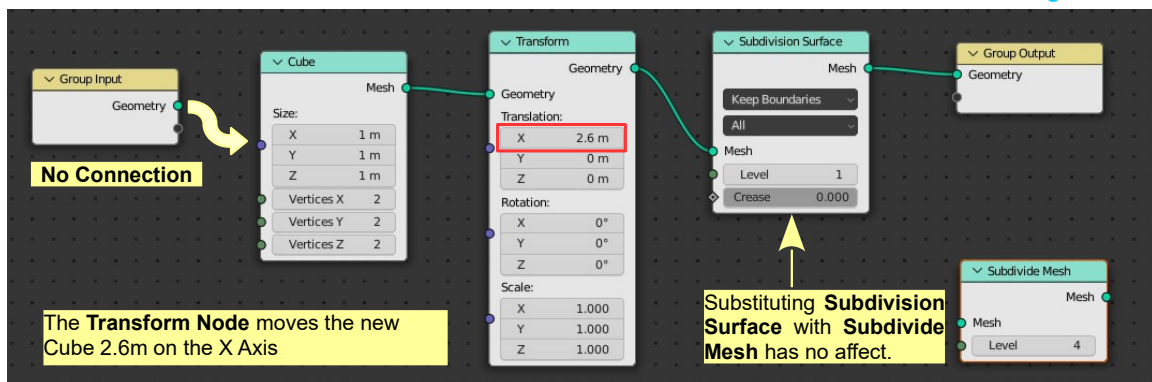
Figure 6.6



In Figure 6.6 the new Procedural Cube is Translated on the X Axis by the transform Node entered in the Pipeline. **The new Procedural Cube is subdivided by the Subdivision Surface Node.**

Note: Not all Nodes work as you may expect. For example, substituting a **Subdivide Mesh Node** in place of the **Subdivision Surface Node** has no effect on the Cube.

Figure 6.7



Node Pipeline in the Geometry Node Editor

The foregoing will allow you to more easily understand the following examples. By following examples and experimenting you will create your own Node Systems.

8.0 Minimising Naming and Color Coding

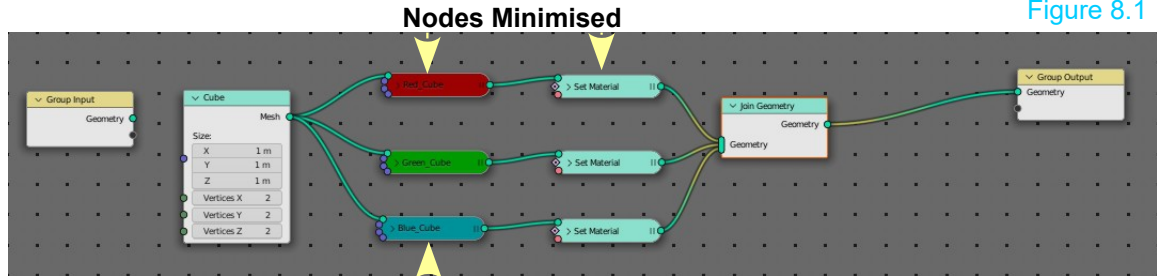


Figure 8.1

Nodes Color Coded

To conserve Screen space when a Node Pipeline becomes complicated Nodes may be collapsed (minimised) by clicking the **chevron** adjacent to the Node Name.

To color code and name Nodes, with the Mouse Cursor in the Geometry Node Editor, press the **N Key** to display the **Tool Panel** at the right hand side of the Editor.

In the Tool Panel with the **Node Panel** displayed type a name in the **Label Panel**.

Figure 8.3

To color code Nodes, check (tick) **Color**.

Click on the **color bar** and select a color in the color picker circle that displays.

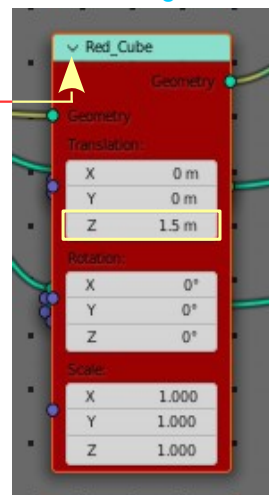
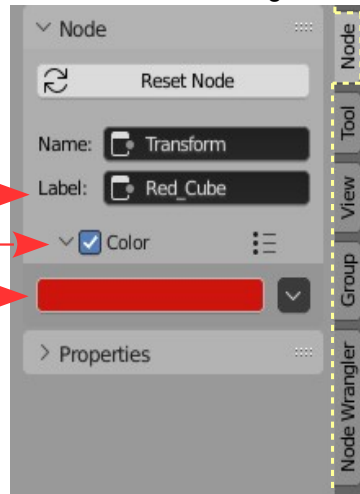


Figure 8.2

Red Cube Z Axis value adjusted to 1.5m.

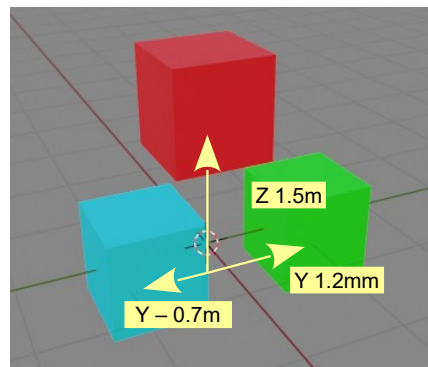
9.0 Arranging Objects

Figure 9.1

By adjusting the values in the **Transform Node** you can manipulate and arrange the Array of Objects in the 3D Viewport Editor.

In Figure 9.1 the Translate values are:

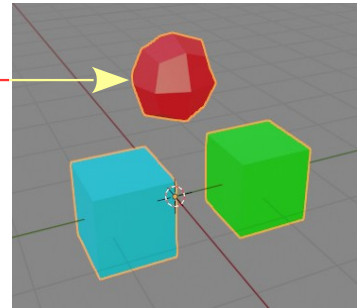
Red Cube: Z 1.5m
Green Cube: Y 1.2m
Blue Cube: Y minus 0.7m



10.0 Adding a Subdivision Surface Node

Adding a Subdivision Surface Node to the Pipeline will affect a single Cube Object or all Cube Objects depending on where the Node is placed.

Figure 10.1



Subdivision Surface Node affects the Red Cube only

Figure 10.2

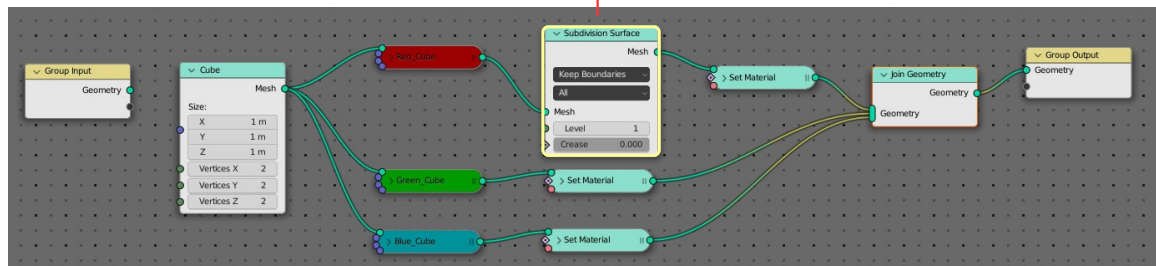


Figure 10.3

Subdivision Surface Node affects all Cubes

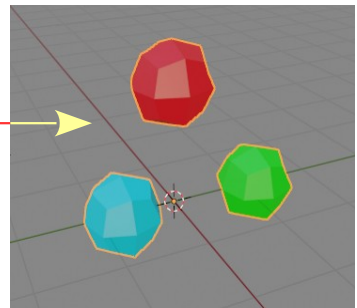
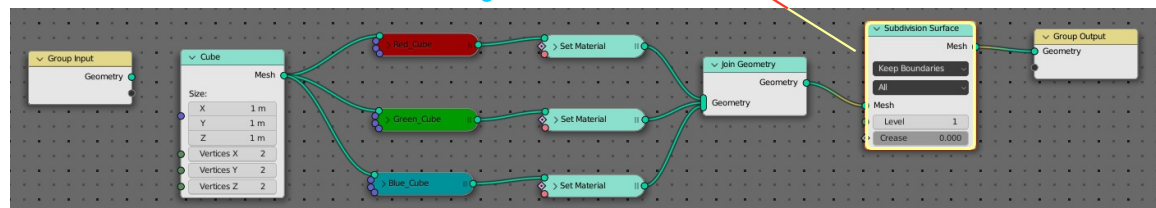


Figure 10.4



A Node placed in a Pipeline affects the Nodes preceding when read from left to right. In Figure 10.4 the **Subdivision Surface Node** could be placed between the **Set Material Node** for the Red Cube and the **Join Geometry Node** to produce the same effect.

11.0 Add Convex Hull

Figure 11.1

Placing a Convex Hull Node in the Pipeline between the Subdivision Surface Node and the Group Output Node wraps the Array in a tight Mesh.

Remember, you are producing a visual effect not necessarily a model of a particular object. This wrapping may or may not be what you want displayed.

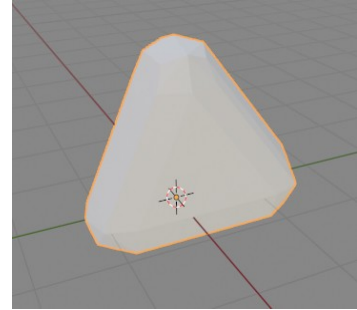
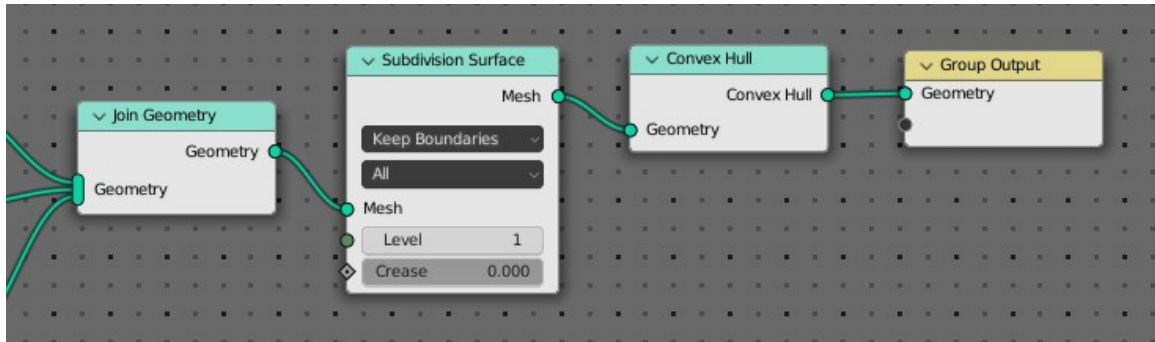


Figure 11.2



Placing the Convex Hull Node as shown in Figure 11.4 and adding a second Join Geometry Node into the Pipeline will wrap only the Red Cube and the Green Cube. This again demonstrates that a Node affects the preceding Nodes.

Figure 11.3

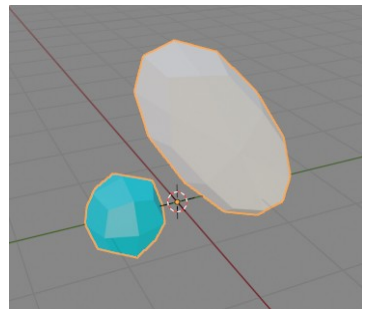
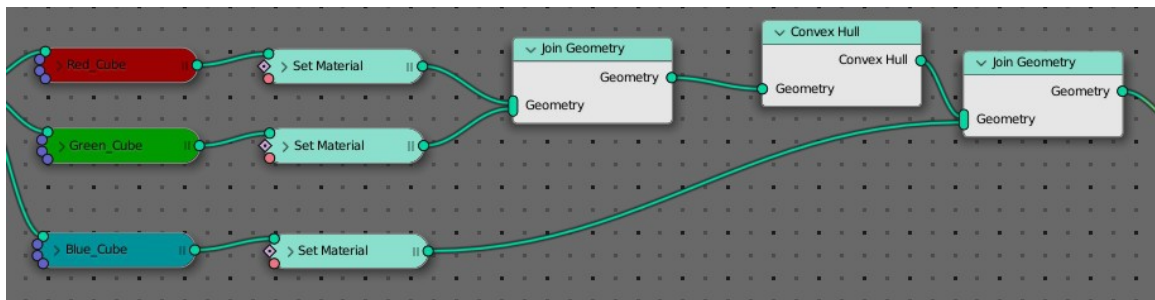


Figure 11.4



A further variation on this effect would be to convert the solid mesh wrap to a **Wireframe Display**.

Wireframe Display

Figure 11.5

Placing a Mesh to Curve Node in the Pipeline between the Convex Hull Node and the Group Output Node converts the solid mesh wrap to Wireframe.

You now have the original Cubes with Subdivided Surface producing the shape of the Array, wrapped in a Mesh which has been converted to a Wireframe. The Cube Array shapes the Wireframe Mesh.

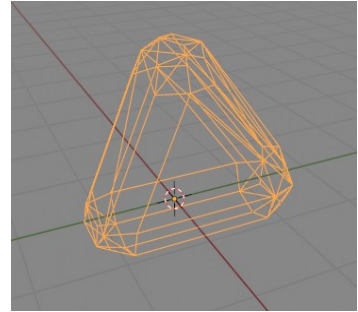
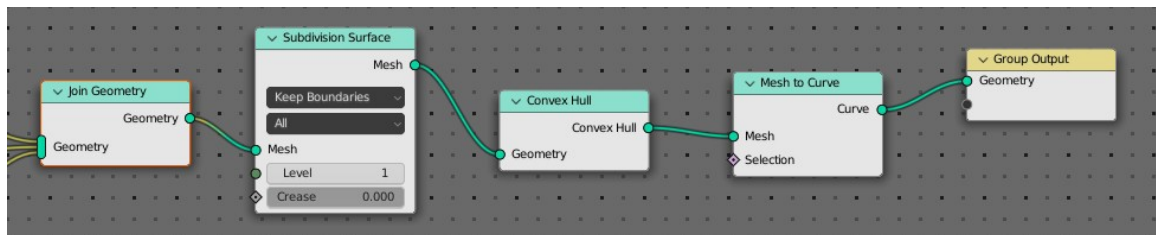


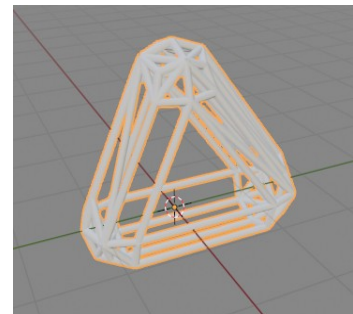
Figure 11.6



A Wireframe does NOT Render, therefore, perhaps you would prefer that the Wireframe should have a solid appearance.

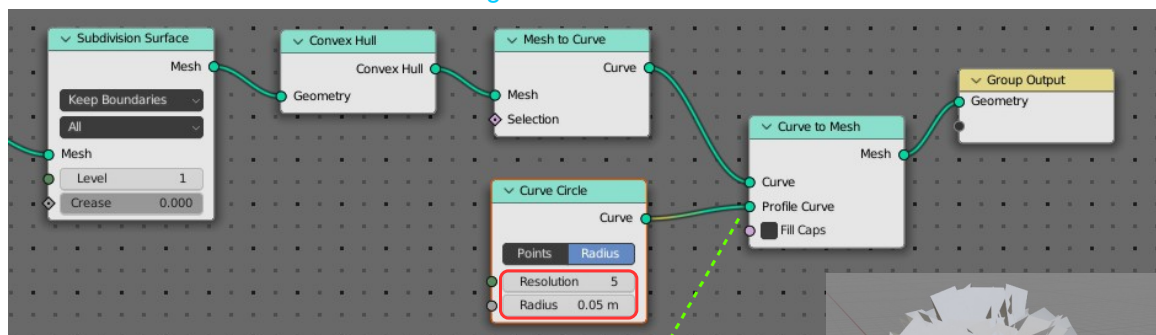
Figure 11.7

Convert the Wireframe (the Curve) back to a Mesh giving it a solid profile. To do this place a Curve to Mesh Node between the Mesh to Curve Node and the Group Output and connect a Curve Circle Node to provide the profile.



Instead of a Curve Circle Profile try a Curve Primitive-Star. You have to play with the settings in the Nodes of the Curve to Mesh Node to obtain the look you want.

Figure 11.8



The name of the game is **Experiment and Experiment** to discover Node Arrangements that produce the desired effect.

What works for you works. Save the Blender File.

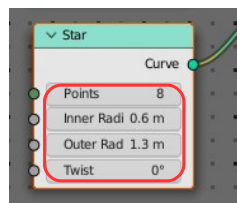
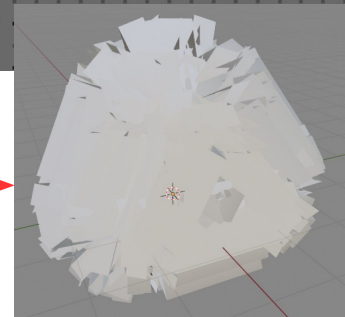


Figure 11.9

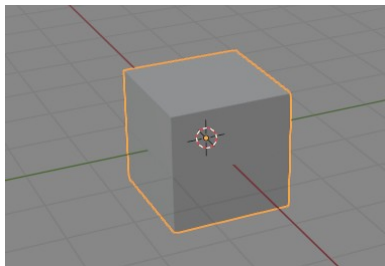


12.0 Materials (Color) for Geometry Nodes

In examples where **Set Material Nodes** are used, the Material is entered in the Node. The Material entered is selected from Materials pre-generated in **Properties Editor, Material Properties**.

To generate Materials you may have Blender opened in the General Layout Workspace or in The Geometry Node Workspace. The default Cube Object in the 3D Viewport Editor has a default gray Material pre-applied.

You see this Material in the **Properties Editor, Material Properties**.



Cube Object – 3D Viewport Editor

Figure 12.1

The default Material is named **Material**.

By clicking the **Browse Material to be linked** button you see the Material named Material stored in the **Material Cache**.

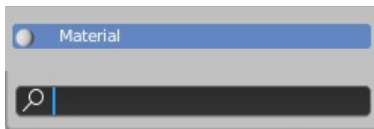
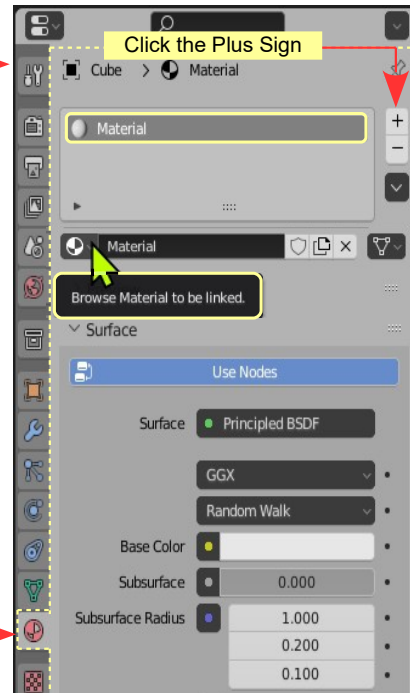


Figure 12.2

Material Properties

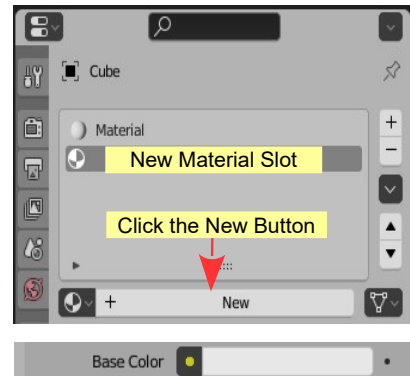


Note: The Material Properties has Use Nodes activated and the 3D Viewport Editor by default is in Solid Viewport Shading Mode. To see new Materials being generated change the 3D Viewport Editor to Material Preview or Rendered Viewport Shading Mode.

Figure 12.3

To generate a new Material click the plus sign in the Properties Editor then click the New Button that displays. This creates a new **Material Slot**. The Properties Editor, Material Properties expands. Click on the **Base Color Bar** and select a color in the Color Picker Circle to assign the color to the new Material Slot.

Repeat the procedure creating new Materials. The Materials are added to the Material Cache.



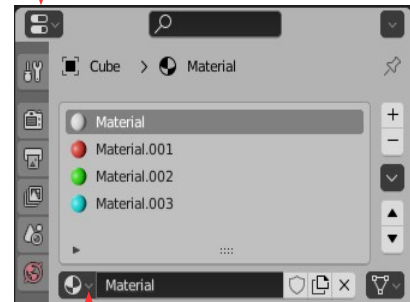
Applying the Material

The **Materials** display in the top panel of the **Properties Editor** and by clicking the **Browse Material to be linked** button you see them stored in the **Material Cache**.

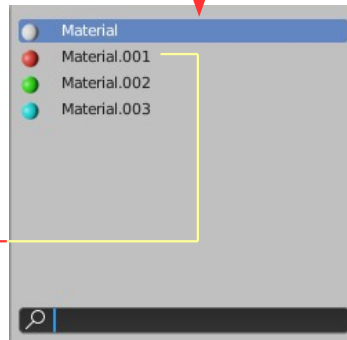
Clicking the Material bar in the Set Material Node displays the Material Cache where you select one of the pre-generated Materials to be assigned to the selected Object in the 3D Viewport.

Figure 12.4

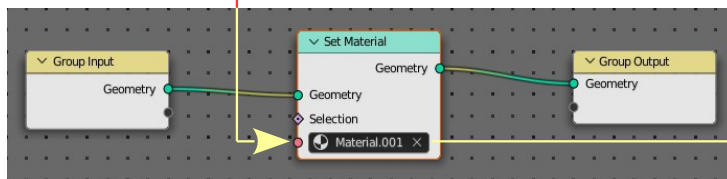
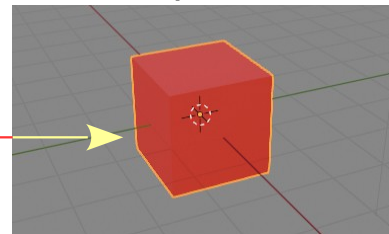
Properties Editor



Browse material to be linked



3D Viewport Editor



Geometry Node Editor

13.0 Instance Objects

Creating an Array by arranging Objects whether they are Procedural or Non-Procedural is one method. Another method, as shown in Figure 1.6 is to **Instance Objects** on the **Vertices of an Object**. Add a Material to the Instance.

Figure 1.6 replicated

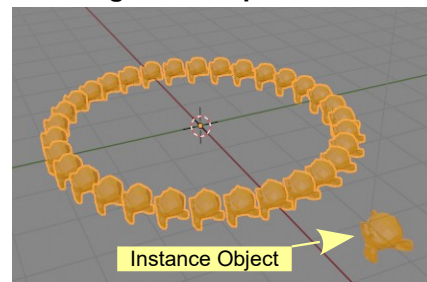
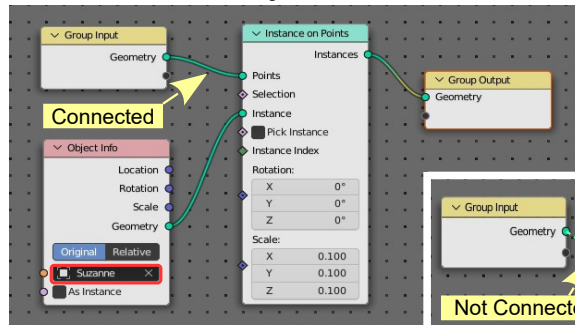


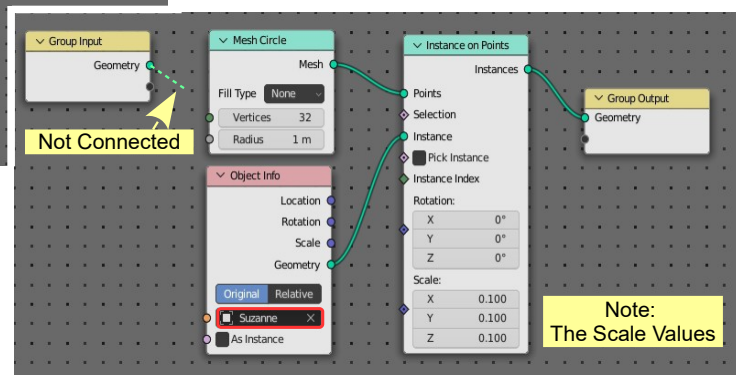
Figure 13.1



Non-Procedural Object

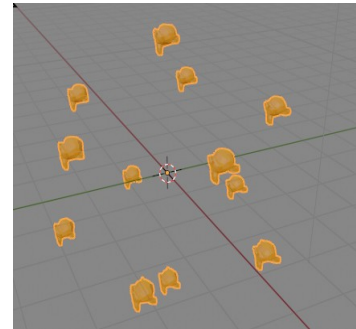
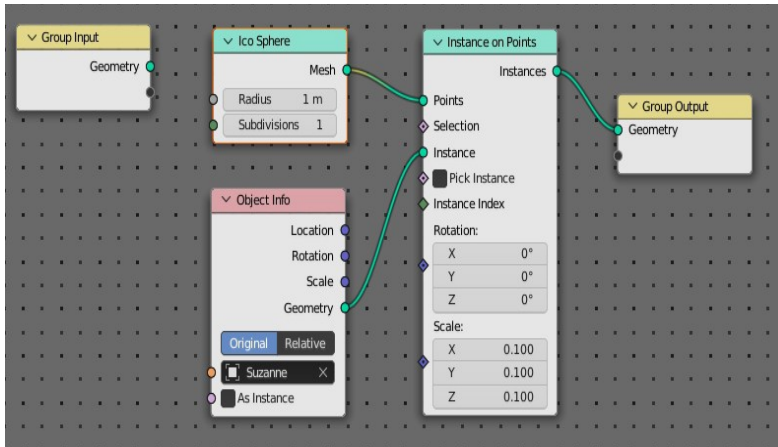
Procedural Object

The **Object Info Node** provides the **Instance Object** information to the **Instance on Point Node**.



By substituting the Mesh Circle with an **Icosphere** you will appreciate that the the shape of the Array is infinite.

Figure 13.2



Note: The Instance Object (Suzanne) has been hidden from view by clicking the **eye icon** in the **Outliner Editor**



14.0 Bloom - Object Glow

Figure 14.1

By making the Instance Object glow against a dark background the effect may be further advanced.

To set the background, go to the **Properties Editor, World Properties, Surface Tab** and set the Surface Background making it black, Strength: 0.000.

With the Instance Object (Suzanne) selected go to the **Properties Editor, Material Properties** and add a Material with Use Nodes active.

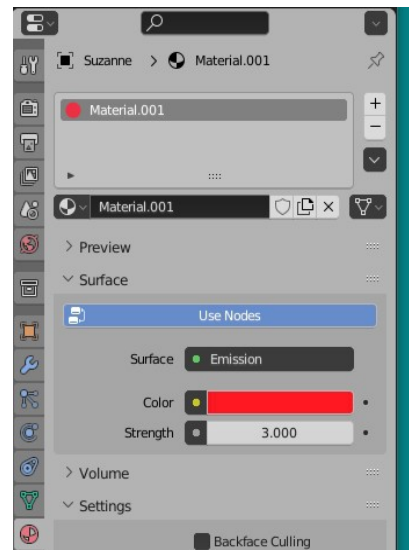
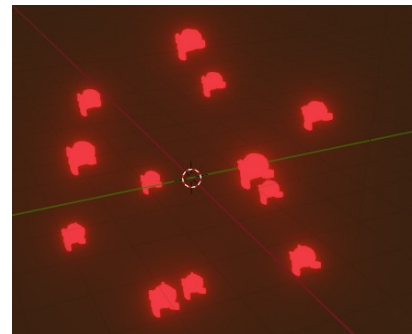
Figure 14.2

In the Surface Tab, set Surface to Type **Emission** and make Strength: 3.000.

Type: Emission makes the Object emit light (glow) when viewed in the 3D Viewport Editor in Rendered Viewport Shading Mode.

Even with a static display as shown in Figure 14.1 the display begins to show signs of becoming impressive.

Remember, the Instance Objects are assigned to the Vertices of an Icosphere. The Icosphere may be animated to rotate and change in size which would see the Array of Monkey Heads rotate and change position in 3D Space as the animation is played.



15.0 Spiral

Using a Spiral to create an Array can produce impressive displays.

Start with the default Cube Object selected in the 3D Viewport Editor and in the Geometry Node Editor click the New Button in the Header to create a Pipeline. Have an Instance Object in the Scene (UV Sphere scaled way down with a Material added). Add a **Spiral Node** and connect together with an **Instance on Point Node** and an **Object Info Node**. The UV Sphere is entered in the Object Info Node as the **Instance Object**.

Figure 15.1

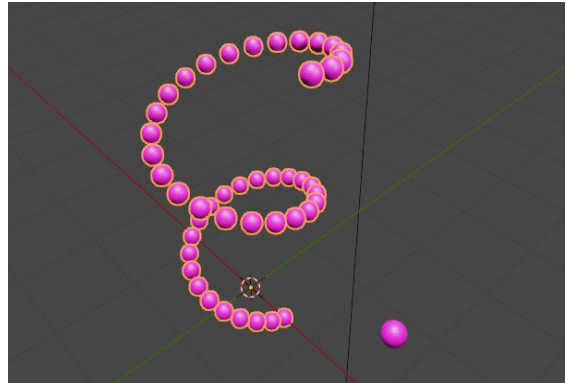
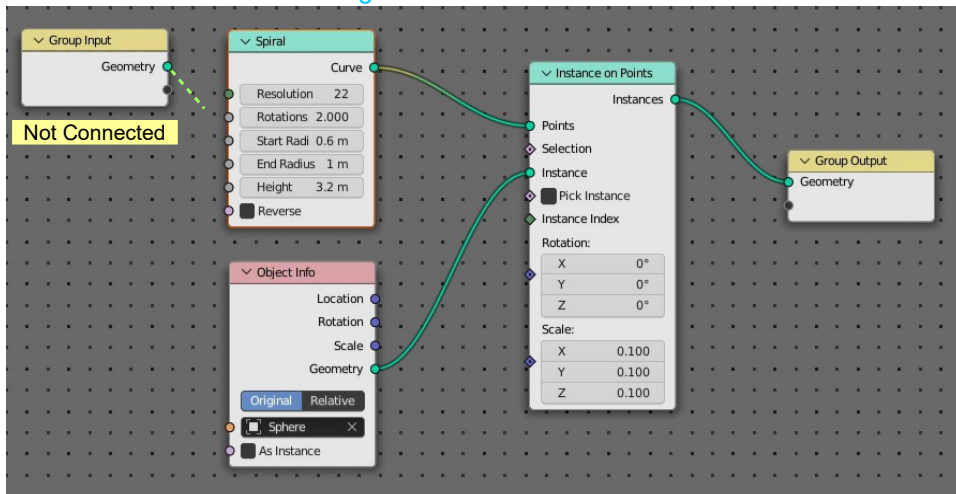
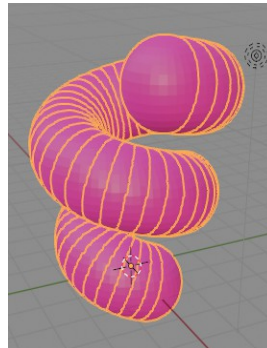
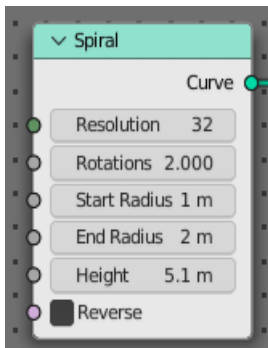


Figure 15.2



Disconnect the Group Input Node to see the spiral. Adjust the values in the Spiral Node to achieve the effect you want. Note: In Figure 15.1 an Area Light or Sun Light added to the Scene will add to the effect.

Figure 15.3



16.0 Point Cloud and Fields

At this point, Arrays have been demonstrated by manually arranging Objects in the 3D Viewport Editor or shaping a single Object and Instancing a secondary object (the Instance Object) to Vertices. Geometry Nodes allow the creation of **Point Clouds** or **Fields** which automate the manual process.

Point Clouds and **Fields** are treated as separate subjects but at a basic level they are very similar with Instance Objects being displayed on Points which have been designated as the Faces or Vertices of the selected Object.

In the previous example (15.0 Spiral) Instance Objects (UV Spheres) have been displayed on the Points (Vertices) of a Spiral.

Point Cloud

To demonstrate a Point Cloud, start with a Plane Object selected in the 3D Viewport Editor.

Create a Pipeline in the **Geometry Node Editor** by clicking the **New Button** in the Header.

Add a **Distribute Points on Faces Node** (Figure 16.1).

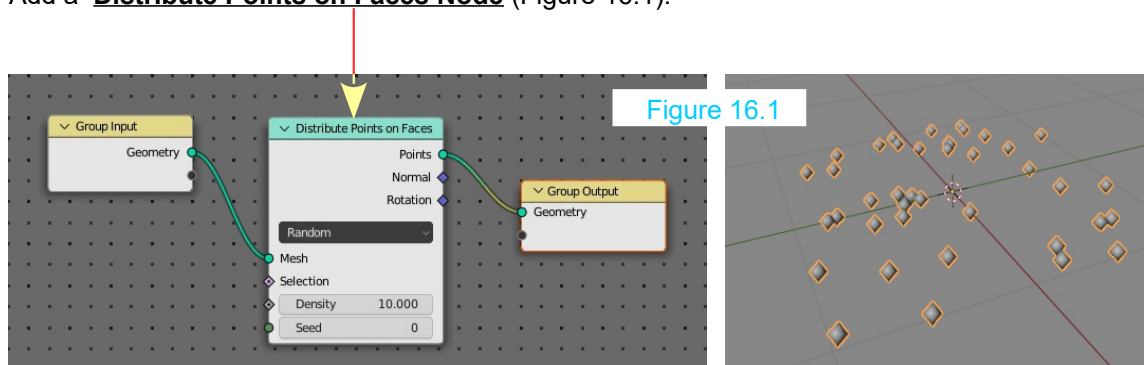
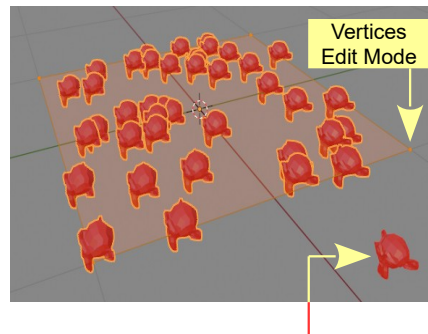


Figure 16.1

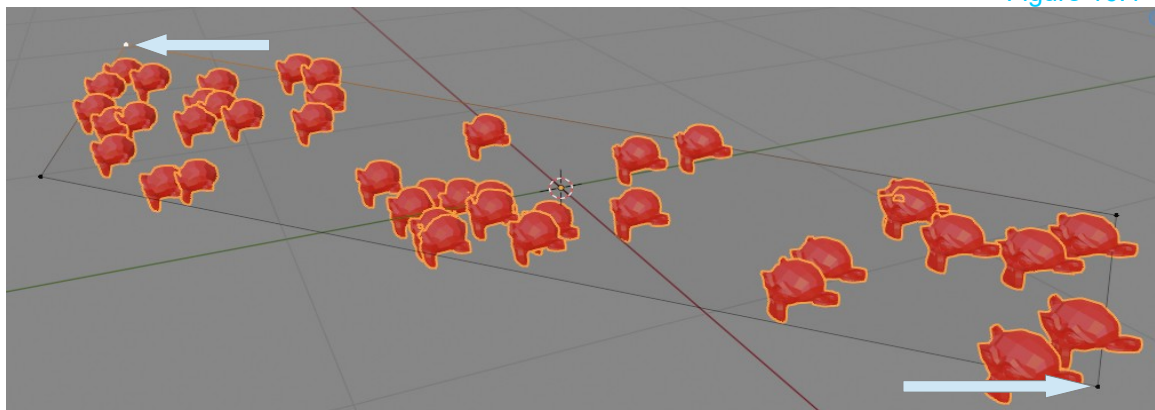
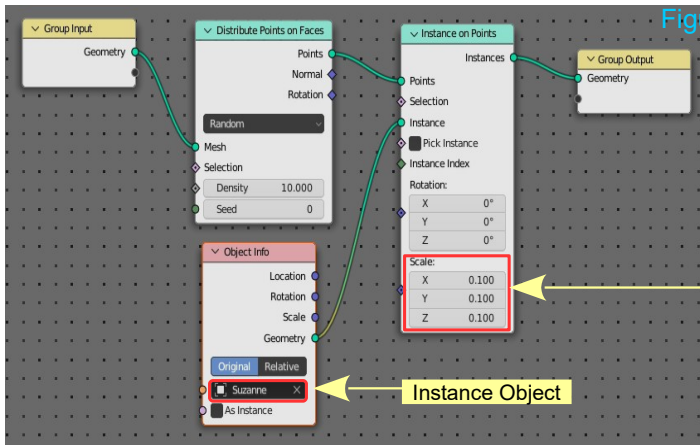
With the **Distribute Points on Faces Node** connected Points are scattered over the Face of the Plane Object.

Figure 16.2

Note: The Plane Object has only four Vertices (Figure 16.2). Points are distributed on the Face of the Plane in accordance with the **Density** value set in the Distribute Points on Faces Node (Figure 16.1).

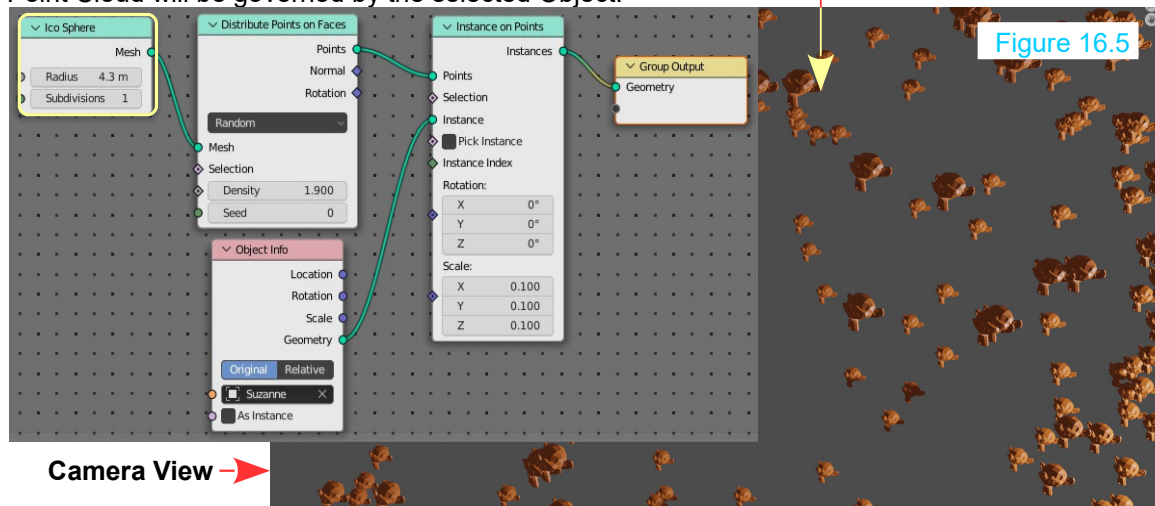


By introducing an **Instance Object** to the Scene (Monkey – Suzanne) and Instancing to the Points you generate an Array of Monkey Heads.



Manipulating the Plane's Vertices in Edit Mode shapes the Monkey Array (Figure 16.4).

From the perspective of **Camera View** the Array constitutes a **Point Cloud**. The 3D shape of the Point Cloud will be governed by the selected Object.



Fields

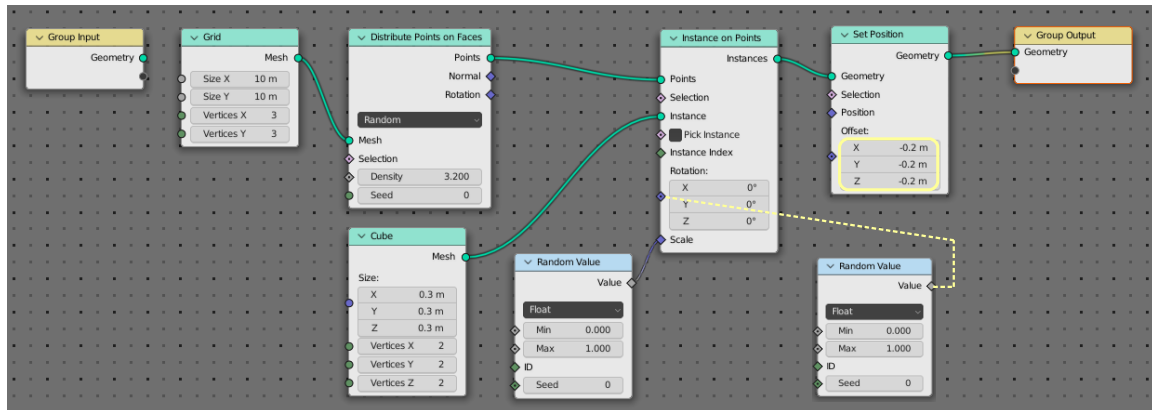
Fields, at a basic level, are similar to **Point Clouds** in that they are concerned with distributing Points in a formation to generate an Array.

By comparing Figure 16.5 with Figure 16.6 you see that the **Procedural Icosphere** Object is replaced by a **Grid Object** which displays similar to a Plane in the 3D Viewport Editor. The **Distribute Points on Faces Node** and the **Instance on Points Node** are retained but instead of the **Object Info Node** being employed a **Procedural Cube Node** is connected directly as the Instance Object.

The Scale of the Cube in the **Cube Node** (3m x 3m x 3m) is being overridden by the values in a **Random Node** (Min 0.000 – Max 1.000) connected the Scale in the **Instance on Points Node**.

A **Set Position Node** is included in the Pipeline which at this point allows positioning of the display in the 3D Viewport Editor on the X, Y and Z Axis.

Figure 16.6



By connecting a **second Random Value Node** to the Rotation value in the Instance on Points Node you will randomly Rotate each Instance Object in the 3D Viewport Editor.

Note. In the 3D Viewport Editor all Instance Cubes are aligned on the XY Plane (Figure 16.7).

Figure 16.7

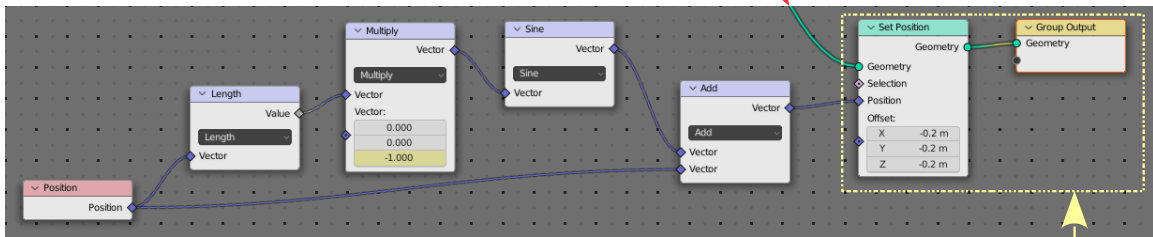


Maths and the use of Maths Nodes were described in Section 10 and it was explained that to control the shape of the Array you may connect **Maths Nodes to perform calculations** which affects an Object selected in the 3D Viewport Editor.

To affect the display shown in Figure 16.6 above the following Nodes will be used and added to the Pipeline (Figure 16.8).

Noodle connecting the Instance on Points Node (Figure 16.6)

Figure 16.8



Set Position and Group Out Nodes (Figure 16.6)

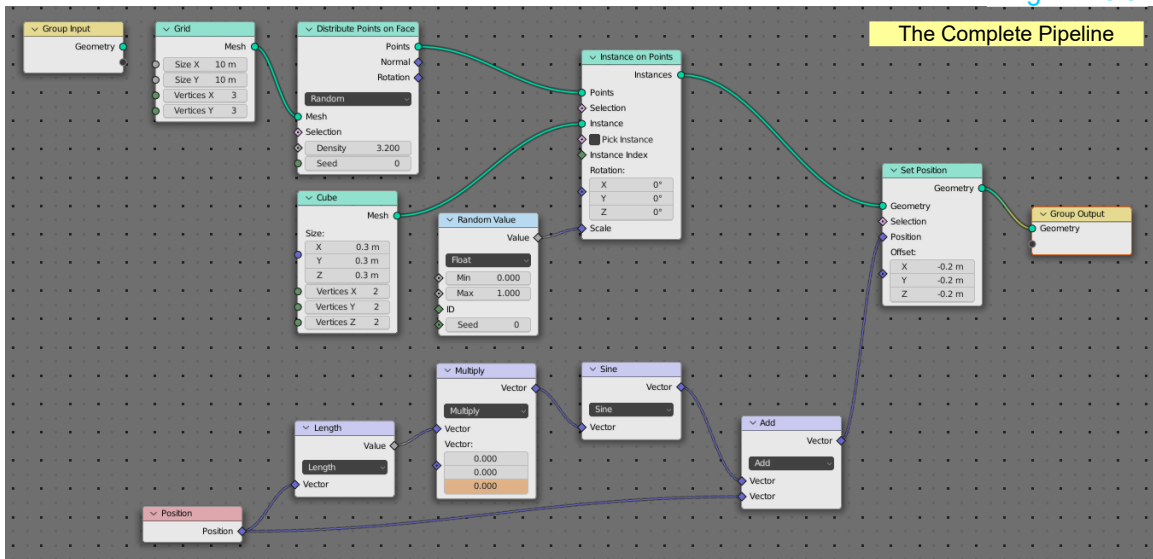
The **Position Node** outputs the vector of each **Point** of the geometry that the Node is connected to. In this case the Node being the **Grid** and more precisely, the **Points assigned to the Grid**.

The **Multiply Node** is a **Vector Math Node** changed from the default Add configuration to Multiply.

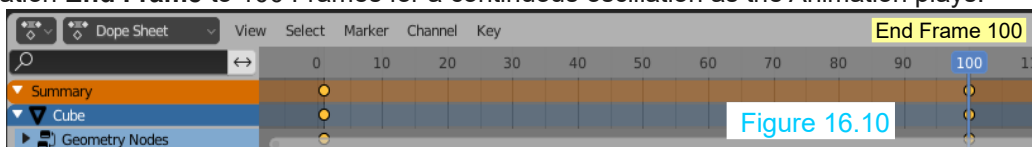
The **Sine Node** is a **Vector Math Node** changed from the default Add configuration to Sine.

Finally, the **Add Node** is a **Vector Math Node** which, in this case, is combining the outputs from the three previous Nodes and the data from the **Position Node**.

Figure 16.9



By Animating the **X Vector** value in the **Multiply Node** from 0.000 to 1.300 over 100 frames in the Timeline the display oscillates in the 3D Viewport Editor following a Sine Wave form. Set the Animation **End Frame** to 100 Frames for a continuous oscillation as the Animation plays.



17.0 Creating a Landscape

There are many ways to create a Terrain or Landscape for a Scene. Geometry Nodes in conjunction with **Foliage Assets Packs** provide a quick method.

Assets Packs , in this context, are collections of pre-constructed models of plants, rocks, trees and stumps which you would find in a landscape.

There are many sites on the internet which offer Assets Packs or individual models, some free and others require a payment.

One site is: <https://www.motionblendstudio.com/blender-vfx-assets>

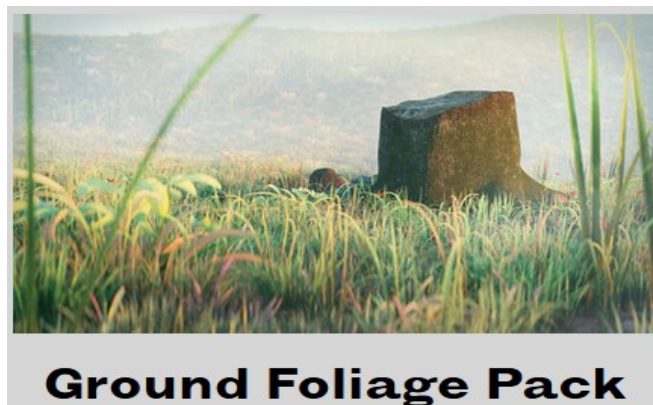
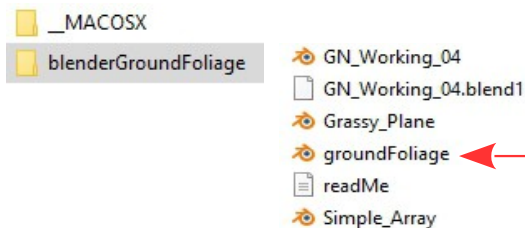


Figure 17.1

This particular **Assets Pack** downloads as a ZIP file name **blenderGroundFoliage.zip**

Unzipping the file produces two Folders:



The Folder named **blenderGroundFoliage** contains several Blender Files. The Ground Foliage Pack is contained in the File named: **groundFoliage** which is a **Blender File**.

Opening **groundFoliage**, in Blender, displays a selection of models which may be **Appended** into a Blender File in which you create your landscape.

Seperate Models

Figure 17.2



Creating a Ground Plane

Before starting an exercise it's nice to know where you are heading. Figure 17.3 shows the final result.

Sand Duns covered in Grass

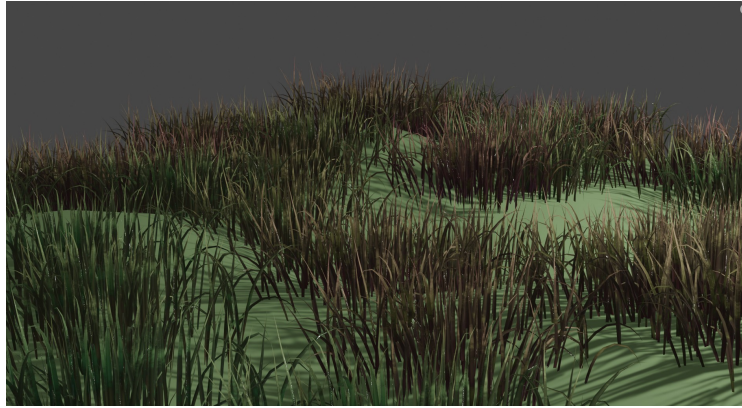
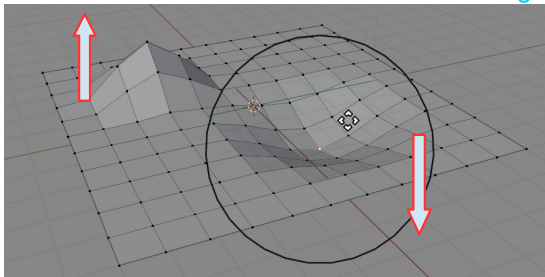


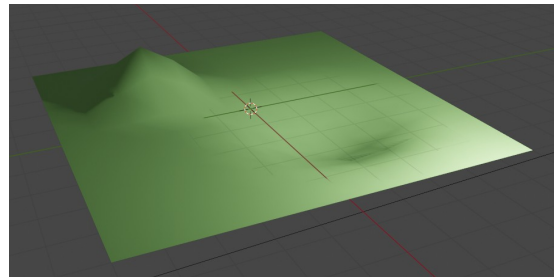
Figure 17.3

The Ground Plane for the Scene is simply a Plane Object, Scaled and Subdivided in Edit Mode with Vertices manipulated with Proportional Editing activated and Set Smooth.

Figure 17.4



Proportional Editing Circle of Influence



Object Mode – Material Applied

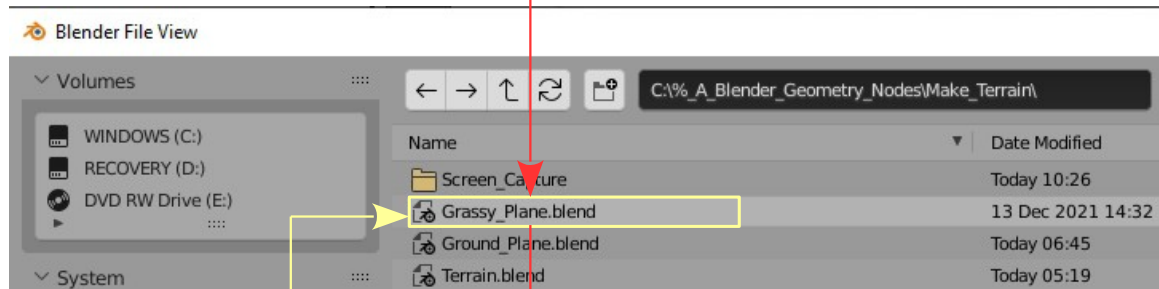
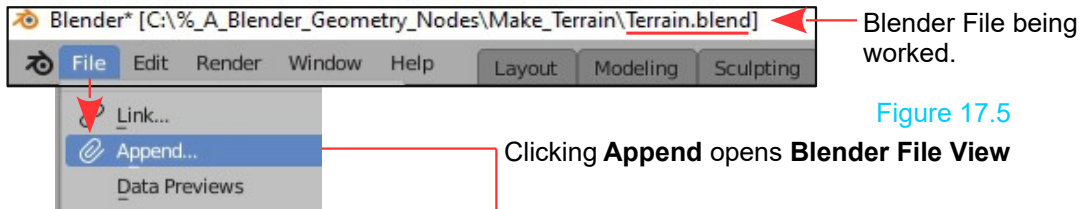
Figure 17.4 is shown for method only. You may configure your ground plane to suit the Scene you are creating. To achieve something like the Figure 17.3 above, a gently undulating plane is preferable.

The method to be employed in generating the Landscape will be the **Point Cloud** procedure previously described with a similar Node arrangement to that shown in Figure 16.3. A Node Pipeline is generated for the ground Plane with a **Distribute Points on Faces Node** being added. A Foliage model is then used as an **Instance Object** which is applied to each of the Points. Bear in mind that once **Cloud Points** are generated the Plane on which they are generated disappears from view in the 3D Viewport Editor. Obviously you want the Ground Plane to remain in view, therefore, before adding the Distribute Points on Faces Node make a duplicate of the Ground Plane complete with Material.

To **Instance Foliage Objects** to the Points in this exercise you **Append** one of the Models from **Ground Foliage Pack**.

Appending Objects

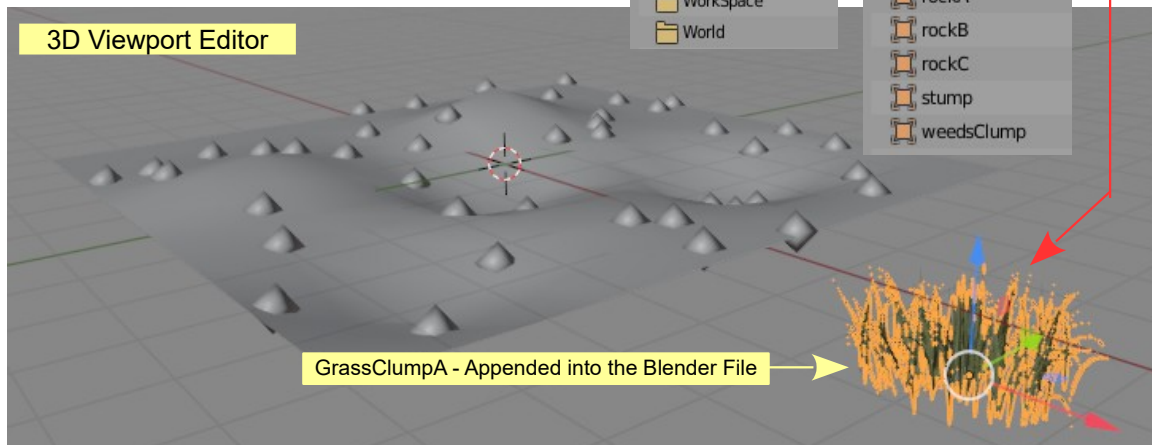
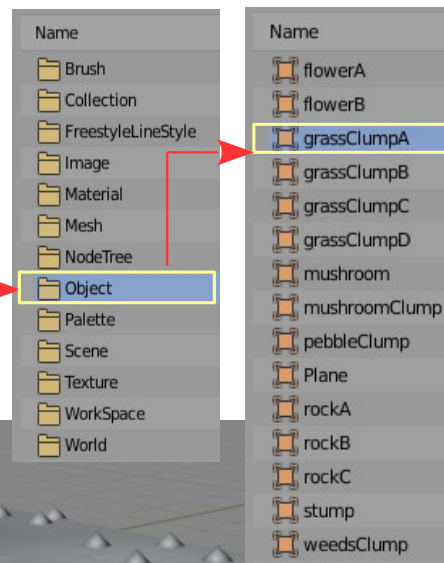
To **Append**, click **File** in the Blender Screen Header and navigate to the Model.



Double click the File containing the Model

Figure 17.6

Double click on **Object** and select the Model (grassClumpA).



Remember there are two ground planes in the Scene, one is named simply **Plane** the other is named **Plane.001**.

Figure 17.7

Plane has a pale green Material Applied.

With **Plane.001** selected create the **Node Pipeline** shown in Figure 17.8 which generates a Point Cloud on the Plane and Instances the Grass Model, **grassClumpA** at each Cloud Point.

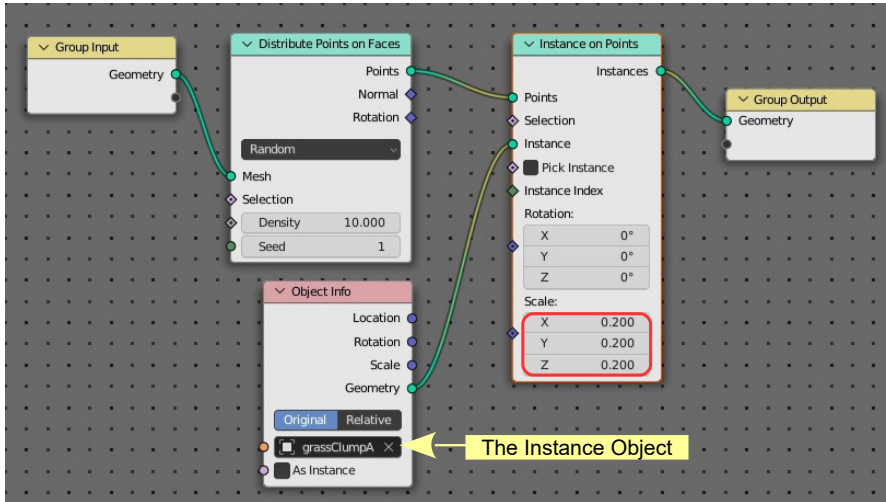


Figure 17.8

Note: The Scale Values

In this particular case, Plane and Plane.001 are **Non-Procedural Objects**, therefore, with the **Node Pipeline** created for **Plane.001** the **Group Input Node** remains connected to the **Distribute Points on Faces Node**.

By giving the World Background a nice blue Material Color and changing the default Point Light in the Scene to an Area Light the Scene Renders as shown in Figure 17.9.

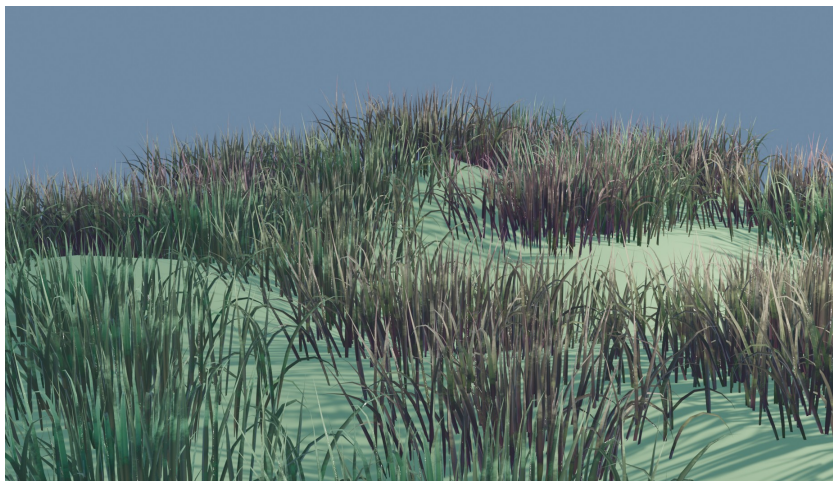


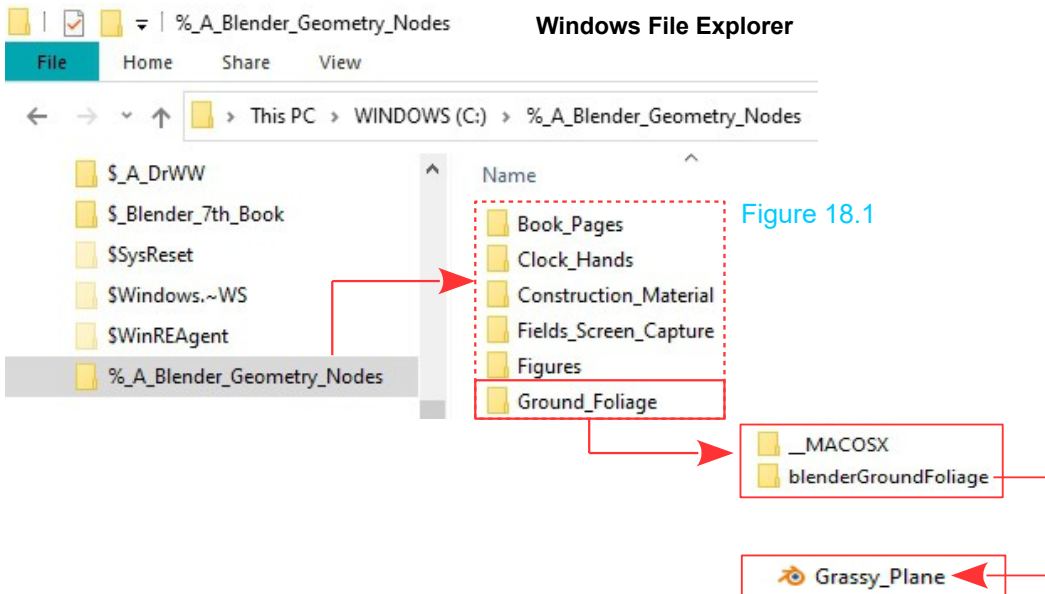
Figure 17.9

18.0 The Assets Library

The **Assets Library** is not a component of Geometry Nodes but, when set up, it is very useful for entering Objects to be used as Instance Objects. The Assets Library may be used to enter Objects into any Blender File once you have stocked the Library with Assets (Models). By default the Library is empty.

To demonstrate stocking the Library, that is, setting up the Assets Library it will be assumed you have downloaded and saved the **Ground-Foliage_ZIP** file and have unzipped to a folder on your PC. Models from the Ground Foliage Pack will be added to the Assets Library.

On my PC **Ground-Foliage_ZIP** is unzipped to a Folder named **Ground_Foliage**:

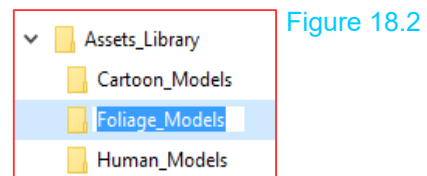


When Ground-Foliage_ZIP is unzipped the **Ground_Foliage** Folder contains two sub Folders (_MACOSX and blenderGroundFoliage) The sub Folder named **blenderGroundFoliage** contains the Blender File **groundFoliage**. This file contains the Foliage Models (Flowers, Grass Clumps, Rocks etc.) which are the **Assets**.

Creating the Assets Library

Using Windows Explorer or File Browser create a new Folder on your PC's Hard Drive and name it **Assets_Library**. In this Folder create sub Folders for different categories of Assets i.e Foliage Models, Human Models, Cartoon Models etc.

Copy and paste the Blender file **groundFoliage** from blenderGroundFoliage to the Folder named Foliage_Models in the Assets_Library.





Blender File **groundFoliage** pasted into the sub Folder named **Foliage_Models**

Figure 18.3

In Blender open **groundFoliage.blend**, and divide the 3D Viewport Editor. Make one part the **Assets Browser Editor**. In the **Outliner Editor** expand the **groundFoliagePack**.

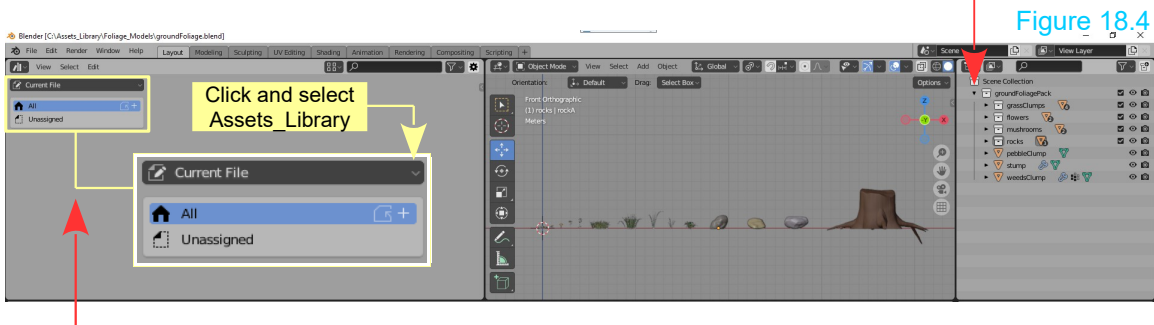


Figure 18.4

In the upper left hand corner of the Assets Browser change **Current File** to **Asset_Library**.

Warning: If a message displays stating that the Path to the Library does NOT exist you add the **File Path** in the **Preferences Editor**.

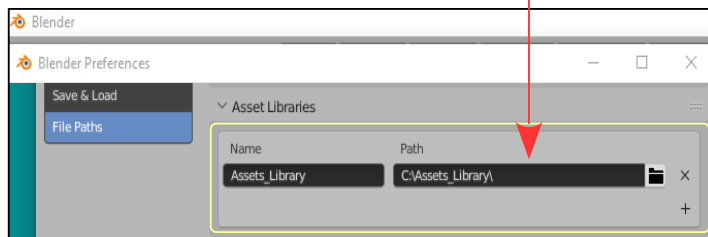


Figure 18.5

Outliner Editor

At this point you add Assets to the Library by marking entries in the **Outliner Editor**.

Figure 18.6

By expanding the entries in the Outliner Editor for the **groundFoliagePack** you display the separate entries for each of the Models displayed in the 3D Viewport Editor.

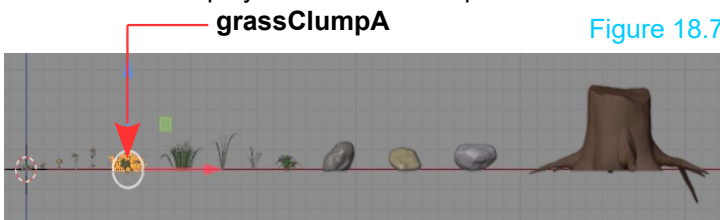
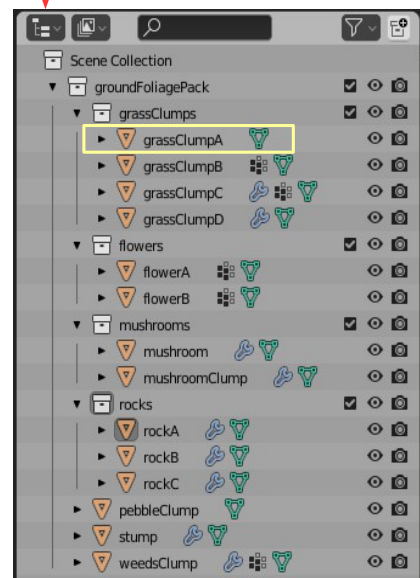


Figure 18.7

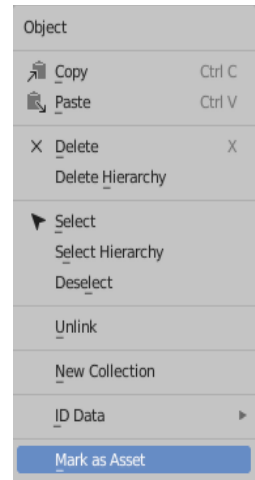
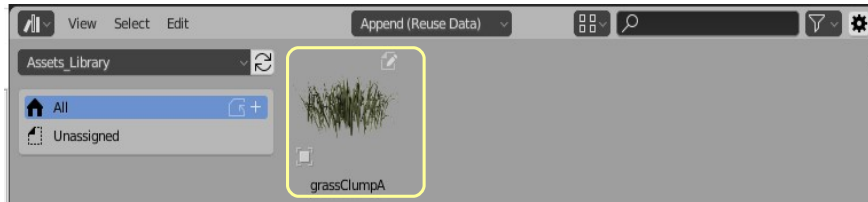
The orange triangle in the Outliner Editor gives an indication that the entry is suitable for marking as an Asset.



To add the Asset (Model) to the Assets Library, right click on the entry in the Outliner Editor and select Mark as Asset in the menu that displays.

In the Asset Browser Editor you will see a Thumbnail Image display for the entry selected in the Outliner Editor.

Figure 18.8

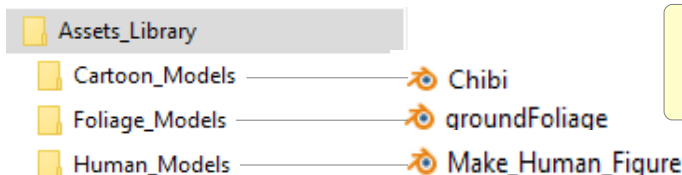
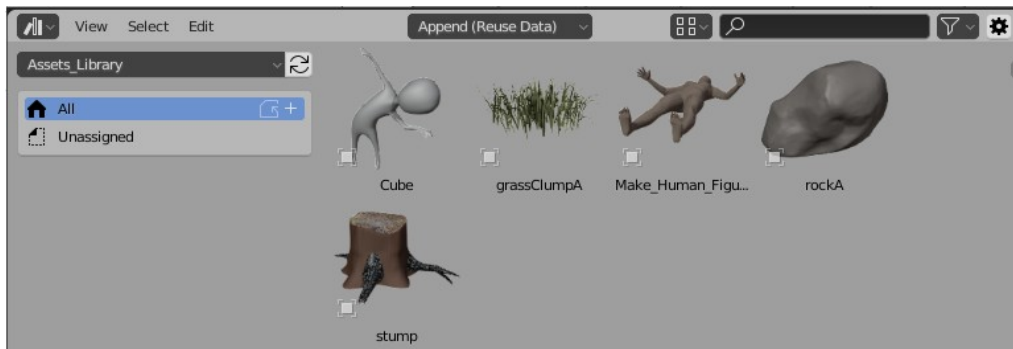


Mark Assets for each Model you wish to add to the Library.

To add more Assets to the Assets Library you copy Blender Files containing Models to the Assets_Library sub Folders then open each Blender File, open the Assets Browser Editor and mark the Model in the Outliner Editor as an Asset.

Figure18.9 shows, **grassClumpA**, **rockA** and **stump** as Assets from the File **groundFoliagePack.blend**, **Cube** is a character named **Chibi** from a **Chibi.blend** File and **Make_Human_Figure** is a model of a Human created in the **Make Human Program** and imported into a Blender File named **Make_Human_Figure.blend**.

Figure 18.9



Save the Blender File when Assets have been Added to the Library.

With Assets (Models) entered in the Assets Library you may open the Assets Browser in a new Blender File, select the Assets Library then click, hold and drag an Asset (Model) into the 3D Viewport Editor.

When Models are entered in the new File they will require Translating, Rotating and Scaling.

19.0 Animation

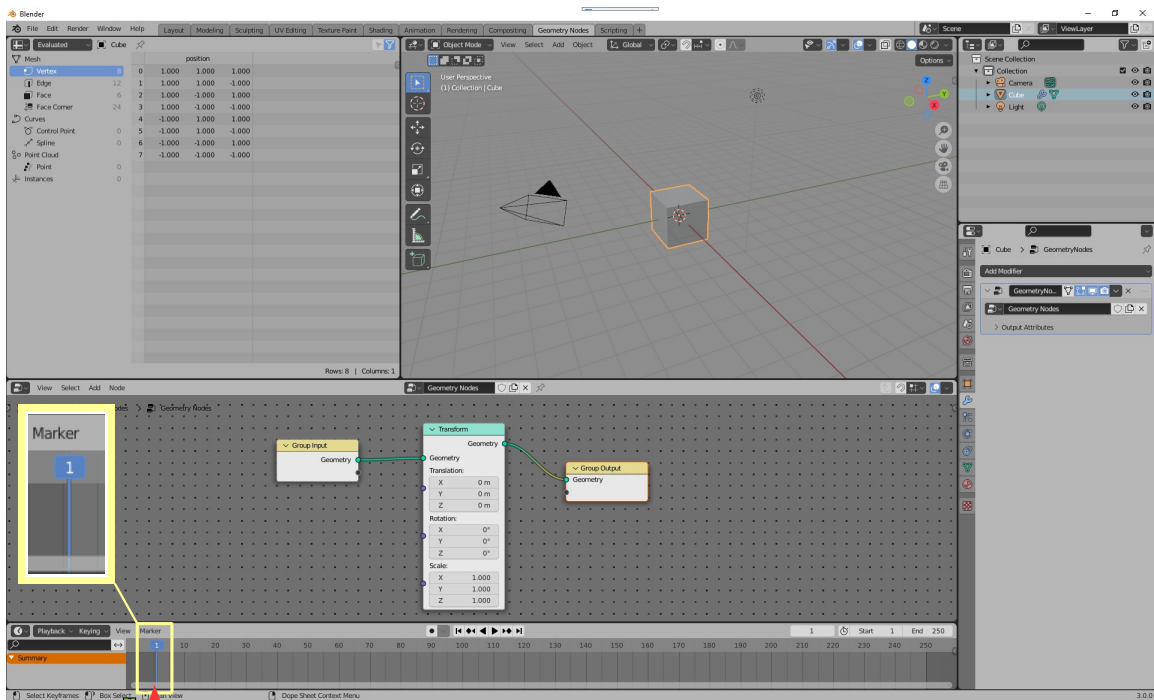
As with many features in Blender, values in Geometry Nodes may be animated to create an animation in the 3D Viewport Editor and then Rendered.

To demonstrate, values in the **Transform Node** will be animated.

With the default Cube Object selected in the 3D Viewport Editor change the Screen Display to the Geometry Node Workspace. To demonstrate Animation divide the Geometry Node Editor horizontally and change the lower part to the **Timeline Editor**.

Click the New Button in the Geometry Node Editor Header to create a Node Pipeline for the Cube Object and insert a **Transform Node** in the Pipeline.

Figure 19.1



In the Timeline Editor the Cursor (blue line) will be at Frame 1.

Remember: The Cube is selected in the 3D Viewport Editor, therefore, the Node Pipeline is applicable to the Cube. The values in the Transform Node control the Translation, Rotation and Scale of the Cube.

To demonstrate Animation the X Axis value in the Transform Node will be Animated to affect the X Axis translation (movement) of the Cube in the 3D Viewport Editor.

In the Transform Node, right click on the **X Translation** value and select **Insert Single Keyframe**. This inserts a single Keyframe at Frame 1 in the Timeline Editor. At this point the Translation X value bar is colored yellow indicating that a Keyframe has been inserted.

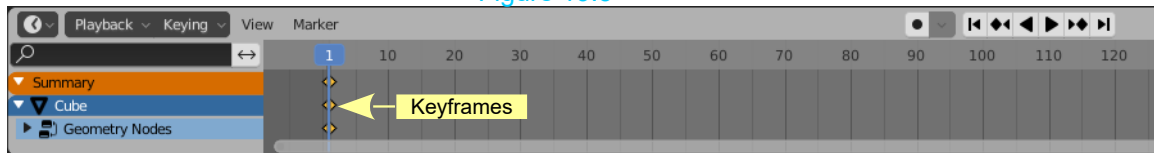
Figure 19.2

In the Timeline Editor small orange diamond icons are displayed at Frame 1. The diamonds also represent Keyframes.

In the Timeline Editor move the Cursor (blue line) to a different Frame (Frame 40).

In the transform Node increase the Translation X value (X: 6.7m). The X transform bar changes color. Right click and select Insert Single Keyframe. Keyframes are indicated in the Dope Sheet Timeline at frame 40.

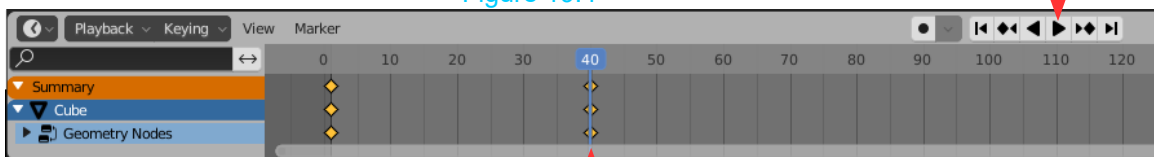
Figure 19.3



In the 3D Viewport Editor the Cube has relocated 6.7m along the X Axis of the Scene.

Move the Cursor in the Dope Sheet Editor back to Frame 1. Scrub the Animation (Drag the Cursor) to see the Cube move in the 3D Viewport Editor. Alternatively, press the **Play Button** in the Header to see the Cube move as the Animation is played.

Figure 19.4



Keyframes at Frame 40

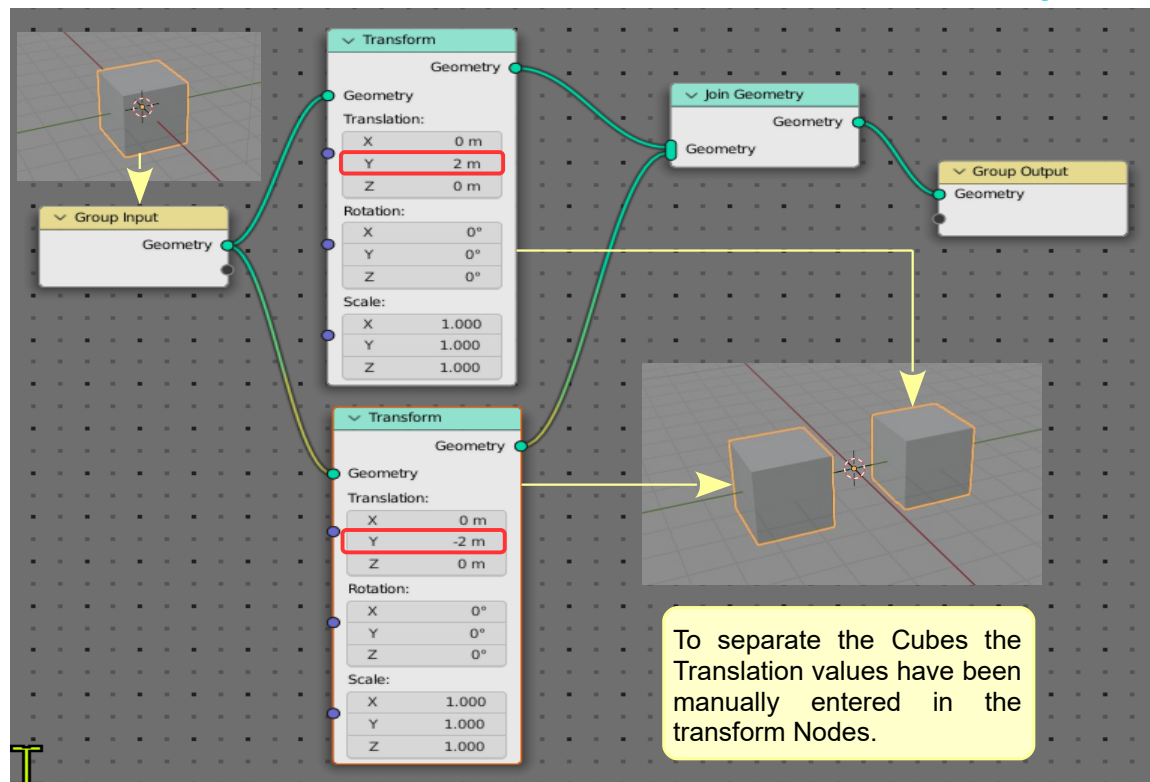
Note: When the Animation is played the Cube moves in the 3D Viewport Editor from the center of the Scene and stops at 6.7m along the X Axis. The Animation continues to play with the Timeline Cursor travelling on to Frame 250 where the Animation replays. Frame 250 is set as the length of the Animation.

The foregoing has demonstrated the Animation of a single value. By adjusting multiple values at different Frames you select **Insert Keyframes** instead of **Single Keyframe**.

20.0 Maths

Maths in Geometry Nodes refers to the use of Nodes to perform calculations which affect an Object which is selected in the 3D Viewport Editor. By stepping through the following exercise you will gain an appreciation of how this is performed and the power of using Nodes.

Figure 20.1



The **Node Pipeline** in Figure 10.1 is applied to the default Cube Object in the 3D Viewport Editor. The Node arrangement with two **Transform Nodes** creates two Cubes spaced 2m either side of the X Axis of the Scene.

Note: Translation plus 2m value in the upper transform Node and Translation minus 2m value in the lower Transform Node. Both values are for the Y Translation.

Manually entering values in the Nodes is fine but were you to Animate the Translation (movement) of the Cubes you would have to Animate the plus and minus values separately. For a simple Scene this is a legitimate method but for complicated Scenes with multiple Objects all moving at the same time you may wish to arrange a Node System such that only one value needs to be Animated.

To employ a single value you add a Value Node to the Pipeline.

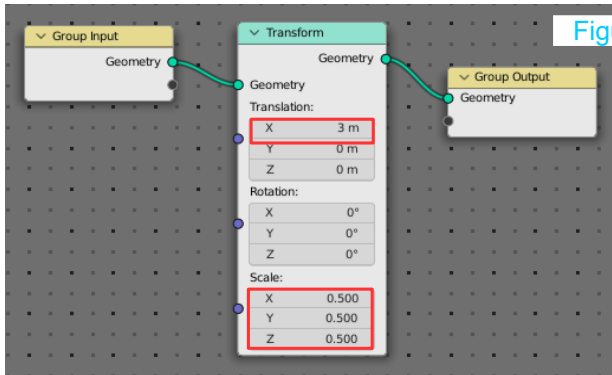
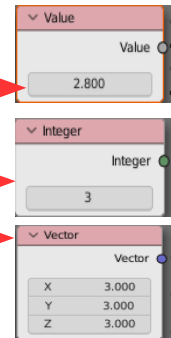
Bear in mind that all Nodes do not output the same type of value and that although they may connect legitimately, the result may or may not be what you require.

Simple Node Value Rule

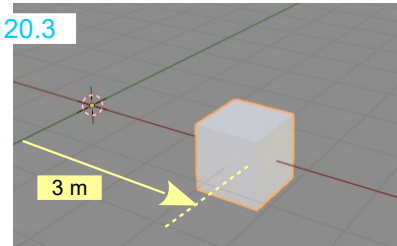
Input Nodes [Figure 20.2](#)

1. A Value with a decimal component or fraction (2.800) is a **Floating Number**.
2. A Value without a decimal component, a whole number(3.000) is an **Integer**.
3. A Value which combines three Floating Numbers or Integers is a **Vector**.

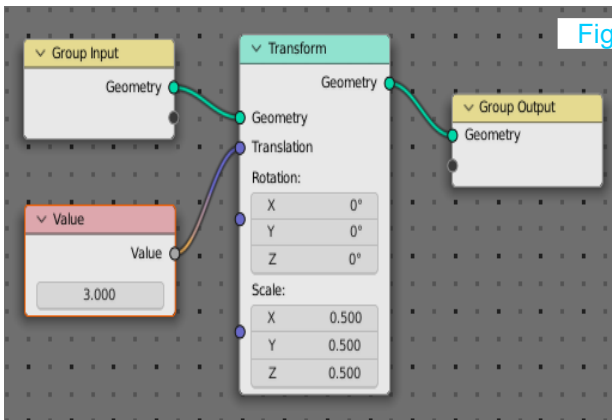
The rules, when applied to Nodes, can be demonstrated by observing the application of a **Value**, **Integer** and **Vector** Node. Figure 19.3 shows the default Cube Scaled down and positioned in the Scene using a **Transform Node**.



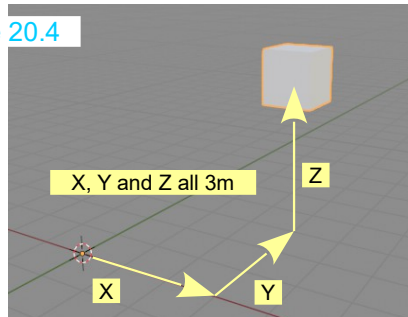
[Figure 20.3](#)



By connecting a **Value Node** to the Translation component of the **Transform Node** the Value is applied to the X, Y and Z Axis.

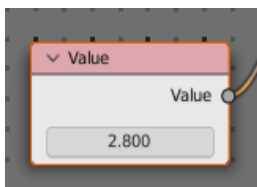


[Figure 20.4](#)

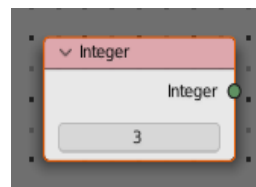


You may use an **Integer Node** in place of the **Value Node** for the same result.

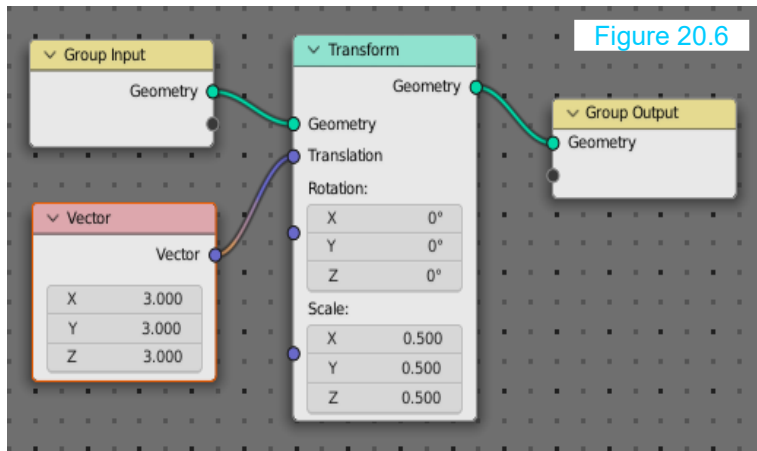
Note: The value 3.000 in the **Value Node** is a **Floating Number** value. The value could be amended to 2.8. In contrast if you enter value 2.8 in an **Integer Node** the value will be rounded up to 3.



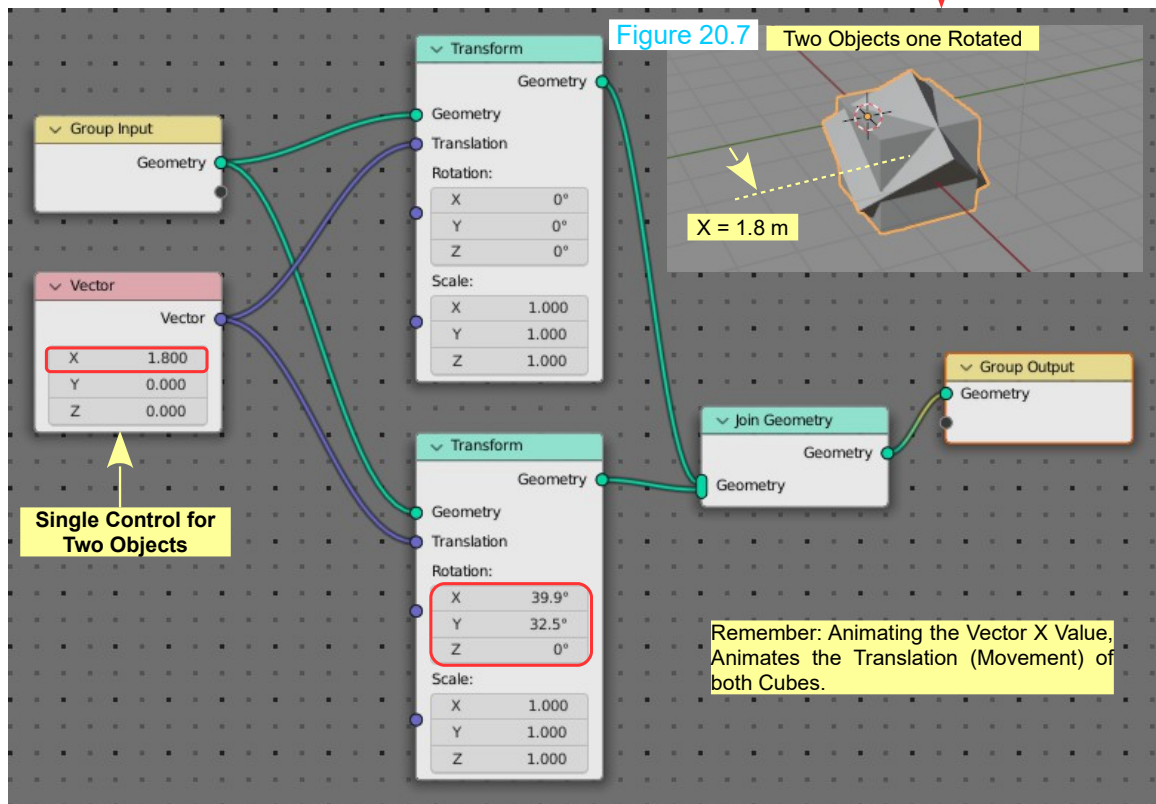
[Figure 20.5](#)



Connecting a **Vector Node** to the **Translation Input Socket** of the Transform Node maintains control of the X, Y and Z Values external to the Transform Node.



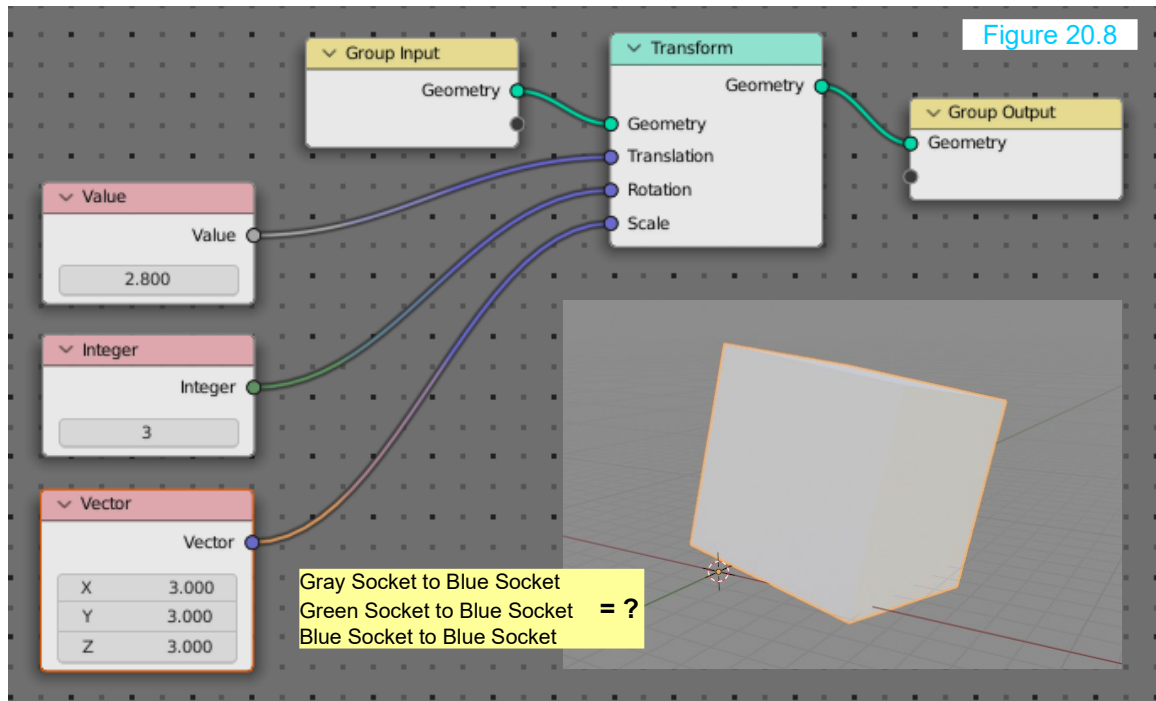
The **Vector Node** controls the X, Y and Z values of the Translation socket which in a system where there are two **Transform Nodes** creating two Objects, creates a **single control** for both Objects.



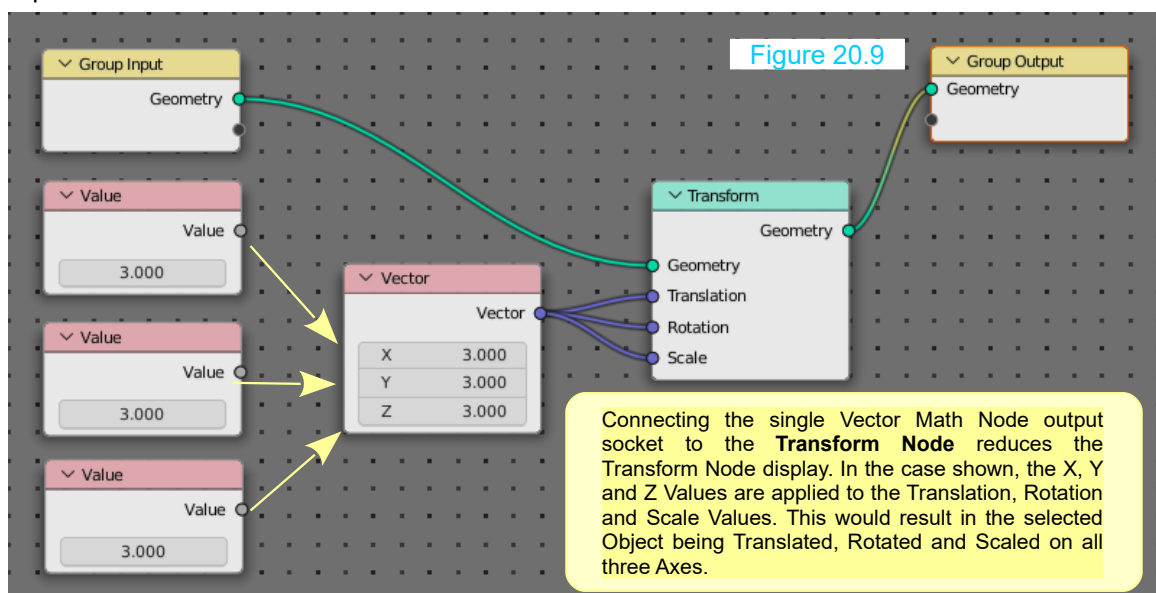
Socket and Noodle Colors

In the above, you will observe that sockets and noodles are connected green to green, blue to blue which indicates that the connections are valid. A **red noodle** means; connection is invalid.

At this point you may well be asking; what has any of this to do with Maths? Well, the Nodes actually perform Maths calculations according to the values you enter. Different Nodes perform different calculations, therefore, you have to select the correct Node for the right job. The colored sockets give an indication only but when connected may not produce the correct result.



Consider the **Vector – Math Node** shown above to comprise three Float Values. This may be represented as:



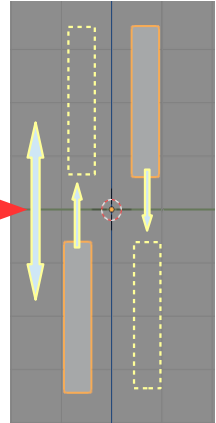
Using Maths

Reciprocating Cubes Figure 20.10

Maths or rather Maths Nodes may be used to generate effects. As a demonstration, two elongated Cube Objects will be Animated to display reciprocated motion. Imagine the Cubes as pistons in an engine. When one piston moves up the other moves down.

Reciprocating Motion →

When considering machine components dimensions are very important, therefore, the physical dimensions of the elongated Cubes determine the length of the motion. The dimensions of the default Cube in the 3D Viewport Editor are 2m x 2m x 2m and the Scale is X, Y and Z = 1.000. The divisions of the Grid Floor in the default Scene are 1m x 1m.



Set the Dimensions of the Piston (Cube)

Figure 20.11

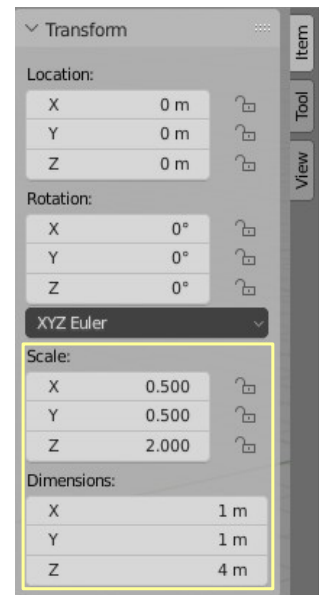
With the default Cube selected press **S + 0.5 + Enter** to reduce the Cube to 1m x 1m x 1m. Press **S + Z + 4 + Enter** to elongate the Piston to 4m.

With the elongated Cube (piston) selected and with the Mouse Cursor in the 3D Viewport Editor, press the **N Key** to display the Object Properties and see the dimensions of the Piston.

In the Object Properties panel the Dimensions correspond to the values that have been set but the Z Axis Scale value is 2 **NOT** the value 4 that was entered?

This occurs since the original default dimensions of the Cube were 2m x 2m x 2m, therefore, 2m x Scale 2 = 4m.

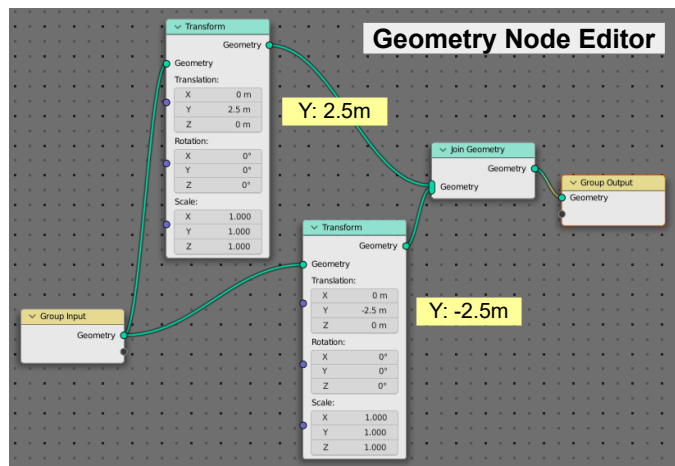
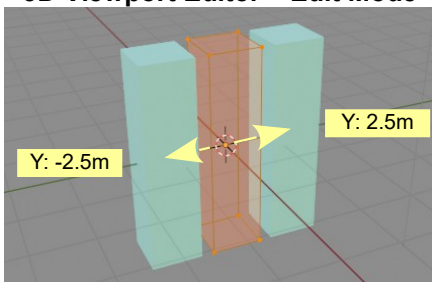
This may seem obvious but is emphasised since setting dimensions in relation to Object Properties is important.



Duplicate the Piston Figure 20.12

To duplicate the Piston use the Transform Nodes in the Geometry Node Editor.

3D Viewport Editor – Edit Mode



From the foregoing, the elongated Cube is 4m in length. To set the Translation (movement) shown in Figure 19.13 the Value is $1\text{m} + \text{A}$. Assume $\text{A} = 0.3\text{m}$

Figure 20.13

The total movement from the upper to the lower positions will be: 2.6m which will be from +1.3m to -1.3m.

The total movement has to be considered with respect to the length of the Animation. Assuming the total Animation length is 100 Frames the movement will be from +1.3m to -1.3m over 100 Frames.

With the Node Pipeline containing two Transform Nodes as shown in Figure 19.14 you would have to Animate each Z Transformation Value separately to create a reciprocal motion.

To simplify the Animation process, especially where multiple Objects are involved, you add Nodes as shown in Figure 19.14.

The **Value Node** contains the value which sets the position of the right hand piston at Frame 1 in the Animation.

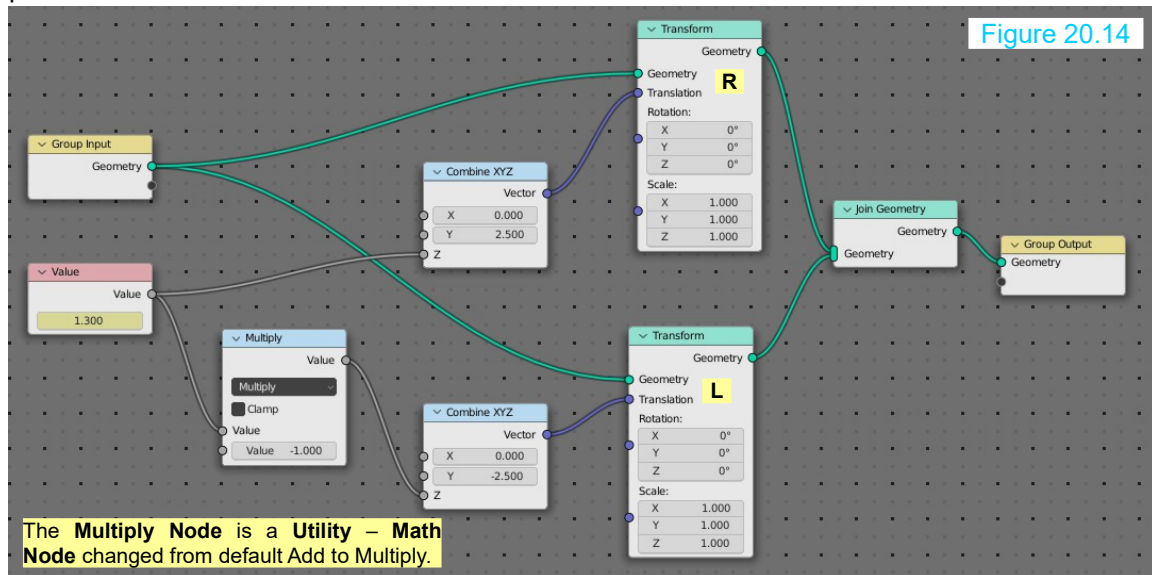
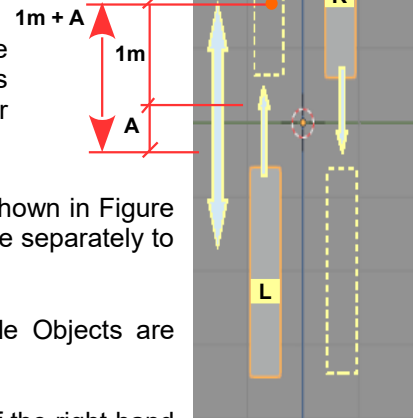


Figure 20.14

The **Multiply Node** is a **Utility – Math Node** changed from default Add to Multiply.

The value in the **Value Node** is a **Floating Number** which does not give a **Vector Value** required for the **Translation values in the Transform Node**, therefore a converter **Combine XYZ Node** is inserted. The Value Node is connected to the **Z input Socket** on the converter then the Vector output socket is connected to Translation input socket on the Transform Node. This allows the value from the Value Node to be conveyed as the **Translation value** and at the same time retain the Y Axis control for the right hand piston.

To produce a reciprocal Translation for the left hand piston the **Multiply Node** with a **minus 1** value converts the plus 1.3 value from the **Value Node** to a minus 1.3.

To Animate the reciprocal motion right click on the value in the **Value Node** and select **Insert Keyframe**. Keyframes are inserted in the Timeline at Frame 1.

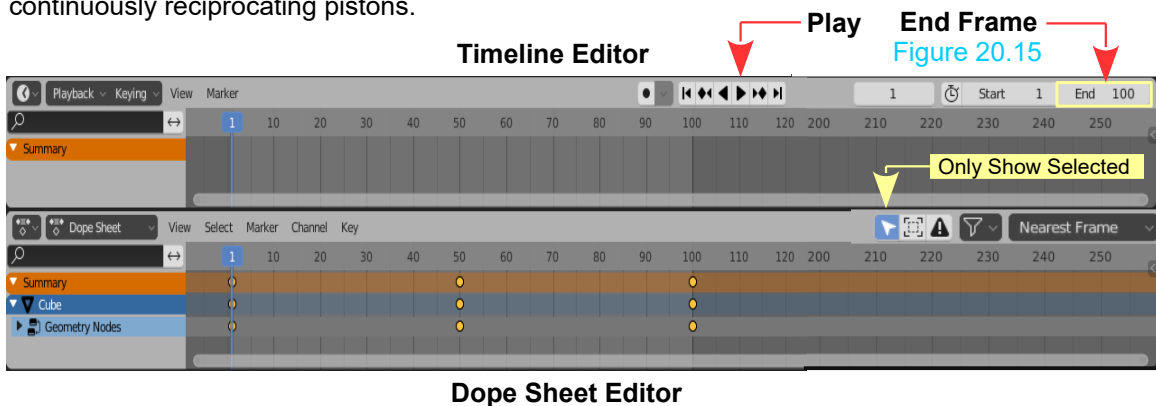
Note: With the Timeline Editor displayed you may NOT see the Keyframes. If this is the case change to the **Dope Sheet Editor Timeline** and click the **Only Show Selected** button.

Move the Timeline Cursor to Frame 50. Change the value in the **Value Node** to minus 1.3 and insert a second Keyframe.

Move the Timeline Cursor to Frame 100, change the value back to plus 1.3 and insert a Keyframe.

Playing the Animation at this point will see the pistons reciprocate from Frame 1 to Frame 100 then stop as the Animation plays on to Frame 250 (the default End Frame) before the Animation replays.

Change the **End Frame** in the **Timeline Editor** to 100 and play the Animation to see continuously reciprocating pistons.



21.0 Summary

This brief introduction to **Geometry Nodes** is intended to get you started. The subject is extensive and is continually being developed, therefore, it is felt that with a basic knowledge, following instruction provided in the many video tutorials on the internet will be a little easier.

There are many tutorials available but be aware that some of the instruction contained in the tutorials has been superseded as development have been applied. Always check the date the tutorial was published and the version of Blender for which the tutorial was written.

Most importantly, be prepared to experiment and record your findings.

Geometry Nodes are a fantastic tool for creating a multitude of effects. It is hoped you will be enthused to pursue a study and develop skills allowing the creation of fantastic effects.

