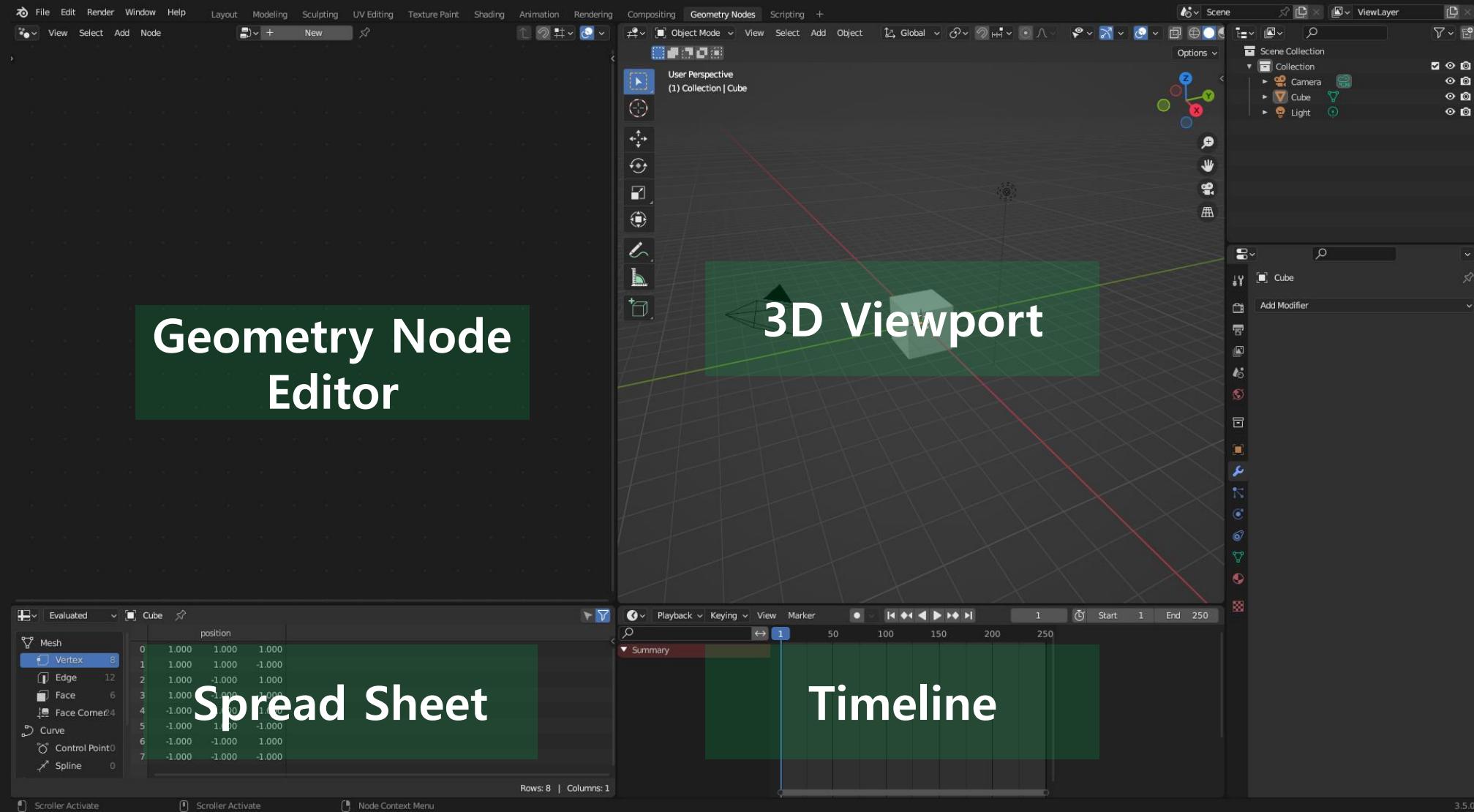


031강 지오메트리 노드 입문

- 지오메트리 노드의 개념과 구조
- Attribute란?
- 지오메트리 정보의 입력과 출력



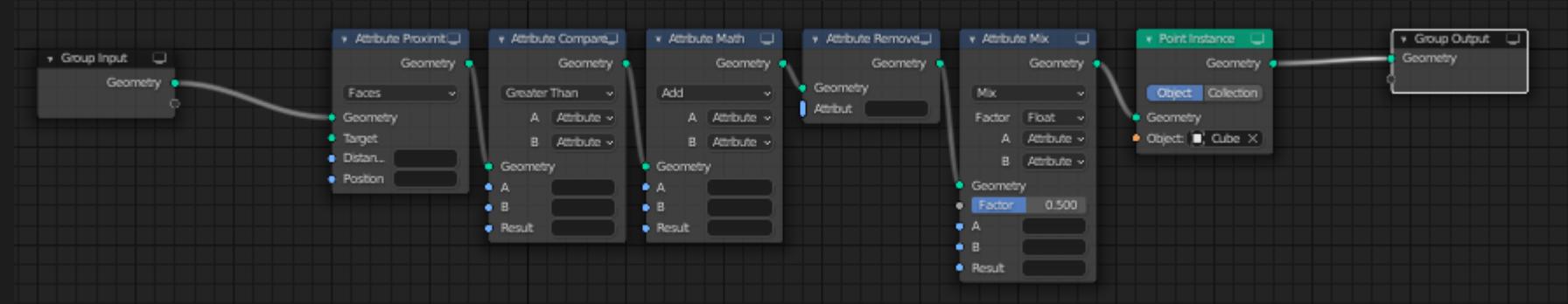
Workspace



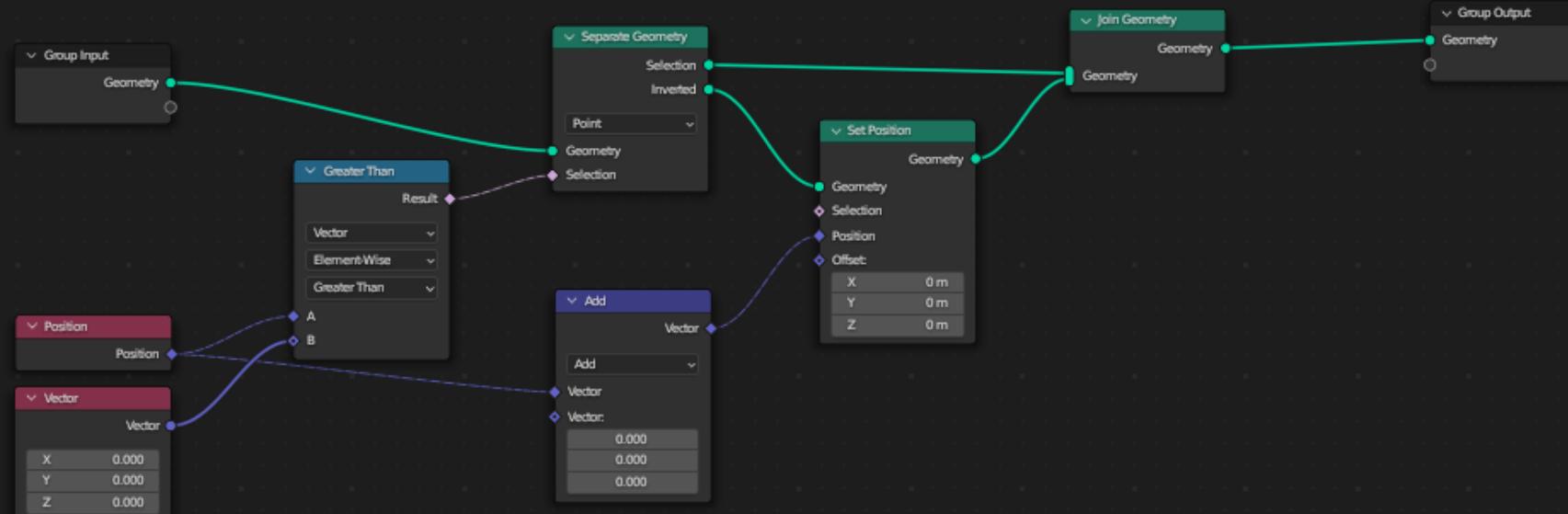
버전 히스토리

※지오메트리 노드는 세이더 노드보다 훨씬 최근에 생긴 기능으로 블렌더에서 가장 빨리 변화하는 기술 중 하나입니다.
그중 가장 큰 변화는 블렌더 버전이 3.0으로 업데이트 되면서 일어났습니다.
3.0에서 지오메트리 노드의 구조 자체가 바뀌었으므로, 만약 인터넷에서 3.0버전 이전의 튜토리얼을 찾으셨다면
따라하는게 거의 불가능함에 유의하세요.

2.9x

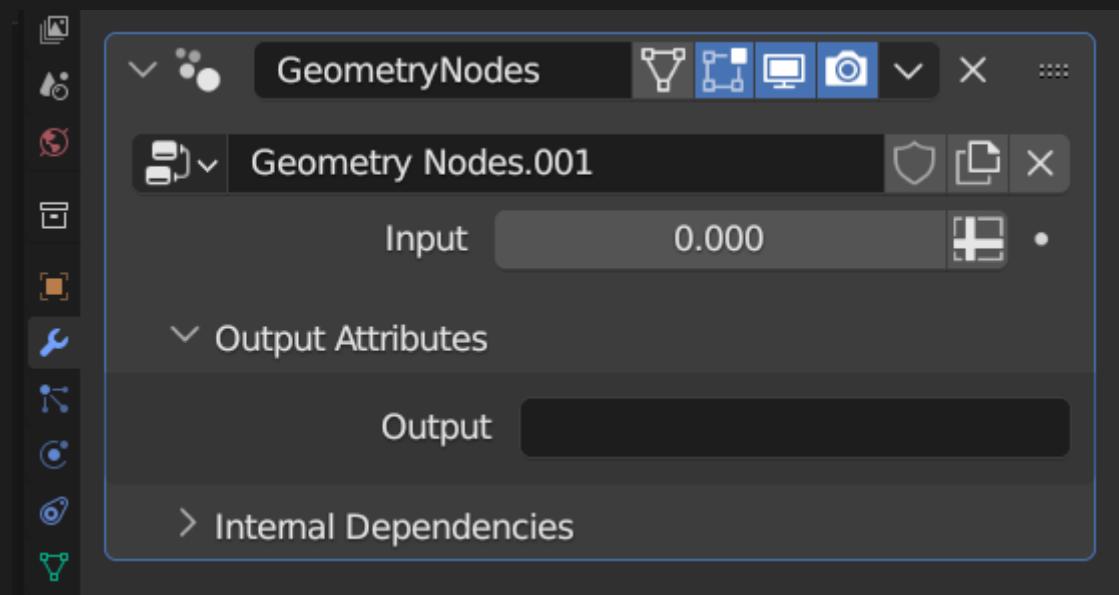


3.0 ~

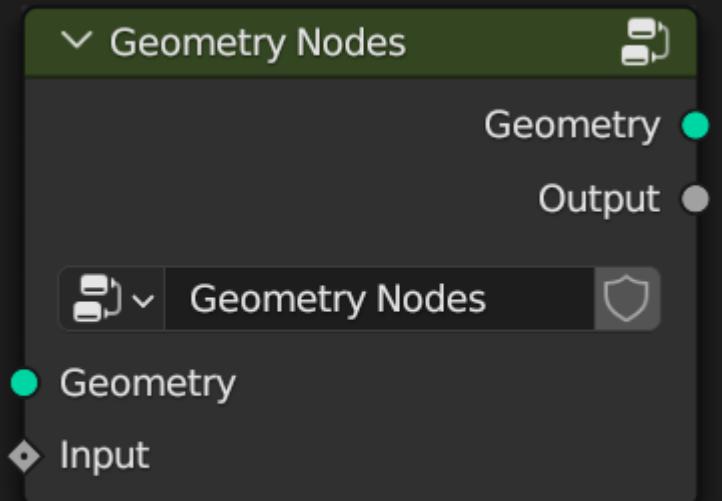


지오메트리 노드의 구조

지오메트리 노드는 모디파이어입니다



지오메트리 노드는 노드그룹입니다

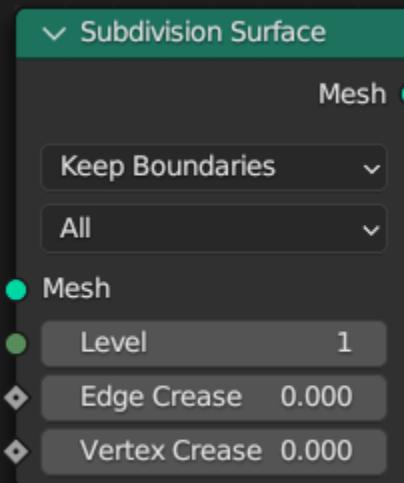
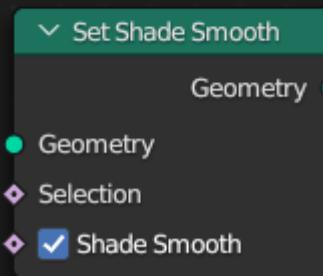
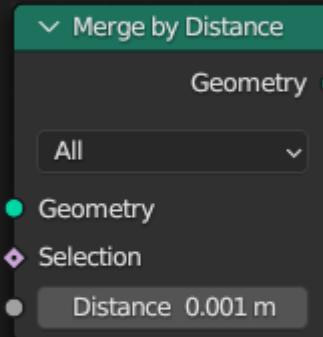


일단 연결해 봅시다

이름만 봐도 알 수 있는 것들이 있습니다

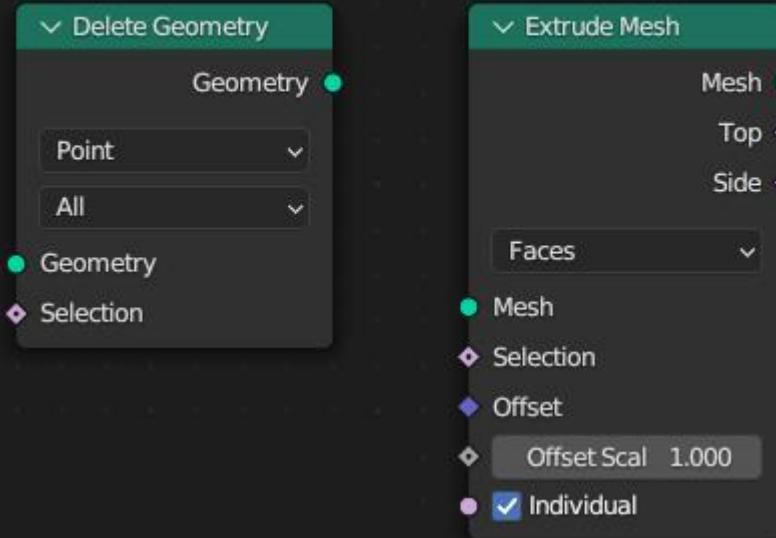


모델링에서 사용하던 용어를 그대로 사용하기 때문에,
어떤 역할을 하는지 직관적으로 알 수 있습니다.



일단 연결해 봅시다

뭔지는 알겠지만..



Delete Geometry에서 무엇을 지울지 어떻게 선택할까요?

Extrude Mesh에서 무엇을 어디로 Extrude할지 어떻게 컨트롤할까요?

Attribute

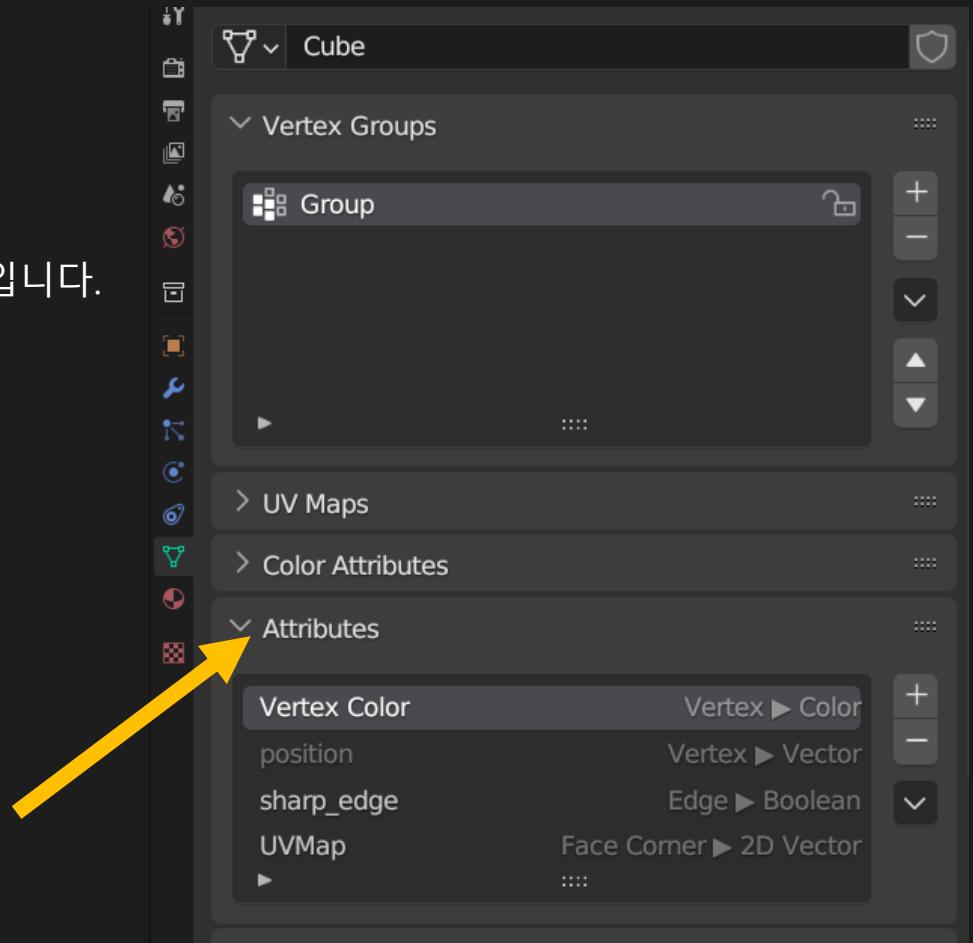
지오메트리의 정보

Vertex Group, Vertex Color, UVMap 모두 지오메트리의 정보이고
지오메트리 노드에서 컨트롤할 수 있습니다.

이들을 Attribute라는 단일 개념으로 통합 관리하는 체계가 만들어지는 중입니다.
예컨대, 이제 UVMap은 생성과 동시에 Attribute에 자동으로 추가됩니다.

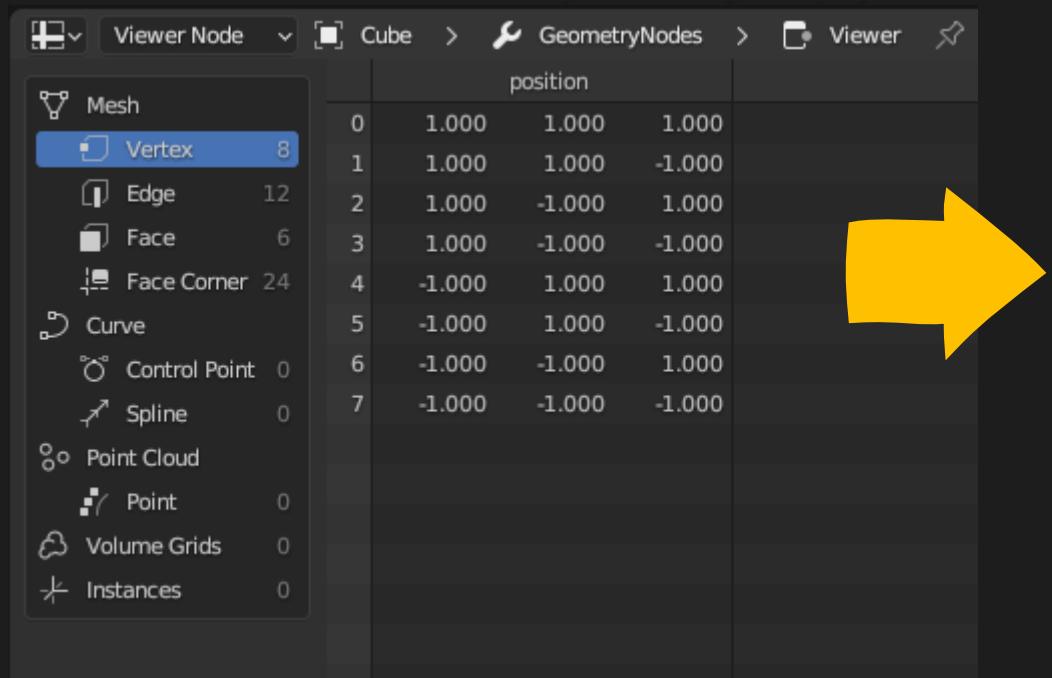
3.5 기준으로 아직 통합되지 않은 정보들도 있습니다.
(Bevel Weight, Mark seam, Face Map...)

Vertex Group은 따로 관리되지만,
지오메트리 노드에서 Attribute로써 불러올 수 있습니다.



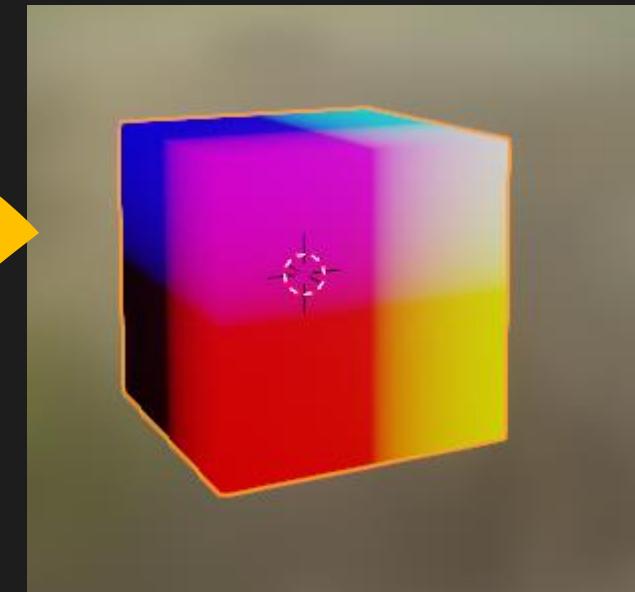
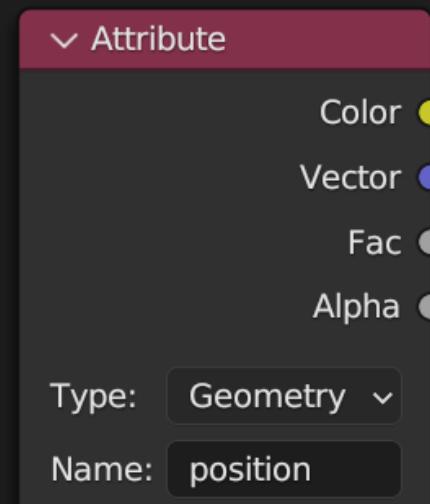
Attribute

Attribute는 Shader에서 불러올 수 있습니다.



position	
0	1.000 1.000 1.000
1	1.000 1.000 -1.000
2	1.000 -1.000 1.000
3	1.000 -1.000 -1.000
4	-1.000 1.000 1.000
5	-1.000 1.000 -1.000
6	-1.000 -1.000 1.000
7	-1.000 -1.000 -1.000

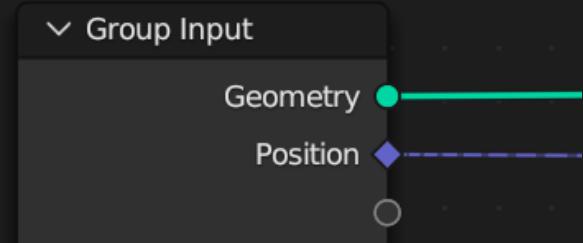
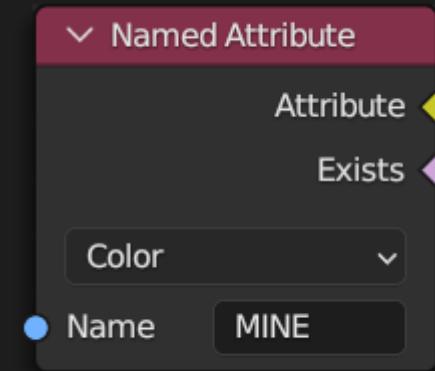
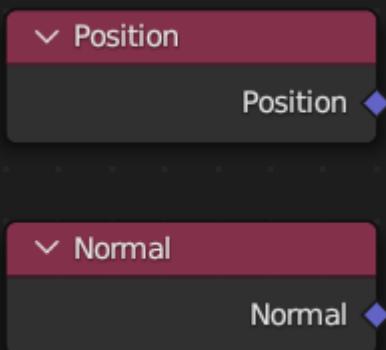
Shader Nodes



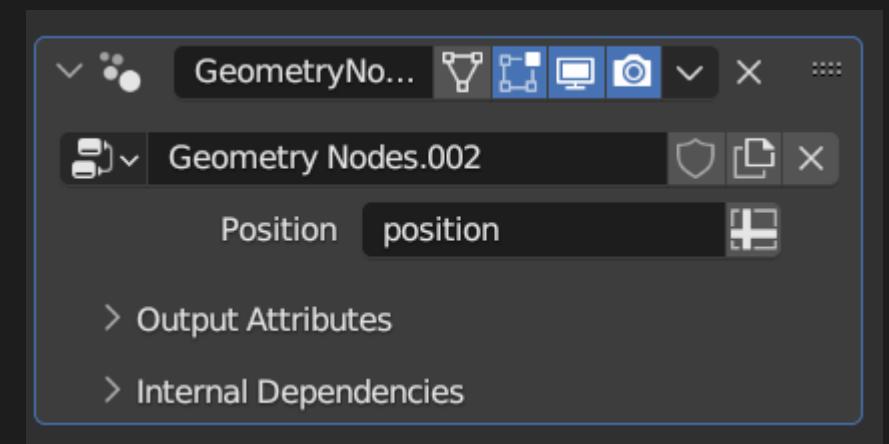
지오메트리 노드에서의 Attribute

지오메트리 노드 내부에서 제공하는
Input 노드로 정보를 불러옵니다.

직접 이름을 지정해서 불러올 수도 있습니다.

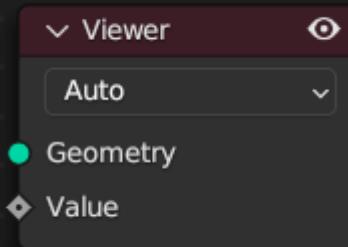


모디파이어에서 불러올 수도 있습니다.



Attribute를 확인하기

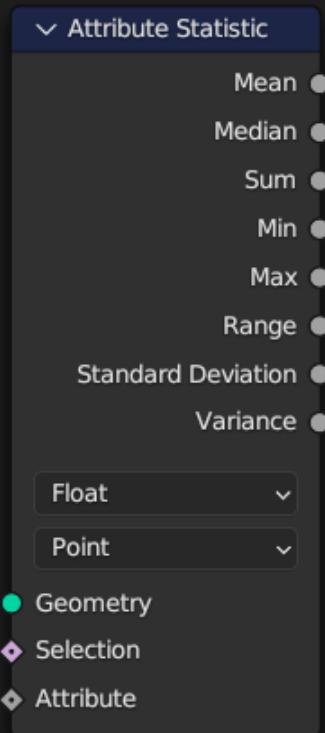
Viewer Node



연결된 Attribute를 Viewport와 Spreadsheet에 보여줍니다.

※ 셰이더와 다르게 Ctrl+Shift+클릭이 이쪽으로 연결됩니다.
만약 실제 출력으로 내보내려면 Alt+Shift+클릭을 하셔야 합니다.

Attribute Statistic

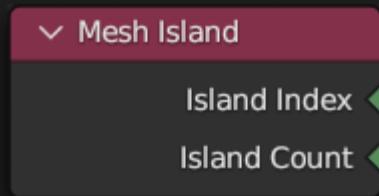


연결된 Attribute의 평균, 최대 최소 등 통계자료를 출력합니다.

이름없는 Attribute

스프레드시트나 Mesh Properties에서 확인할 수 없지만 노드로 연결하면 나타나는 Attribute가 있습니다.

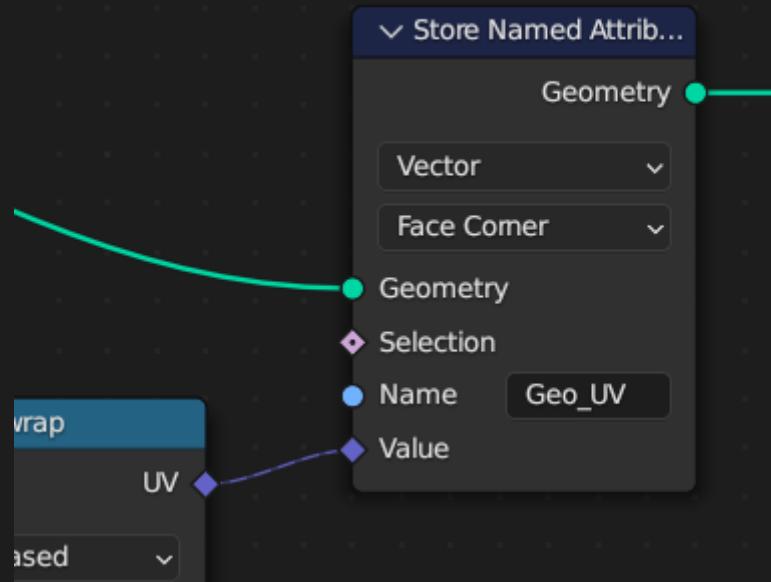
노드를 연결하면 지오메트리 노드 내부에서 자유롭게 사용 가능하지만,
만약 외부에서 사용하려면 이름을 부여해서 저장해주어야 합니다.



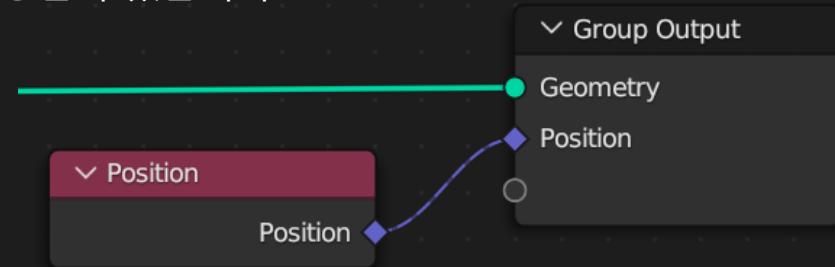
Mesh Island는 점선면이 분리된 지오메트리마다 번호를 매깁니다.
다음 페이지를 참고하여 Attribute를 저장하면, 셰이더 노드에서 사용하실 수 있습니다!

Attribute의 저장

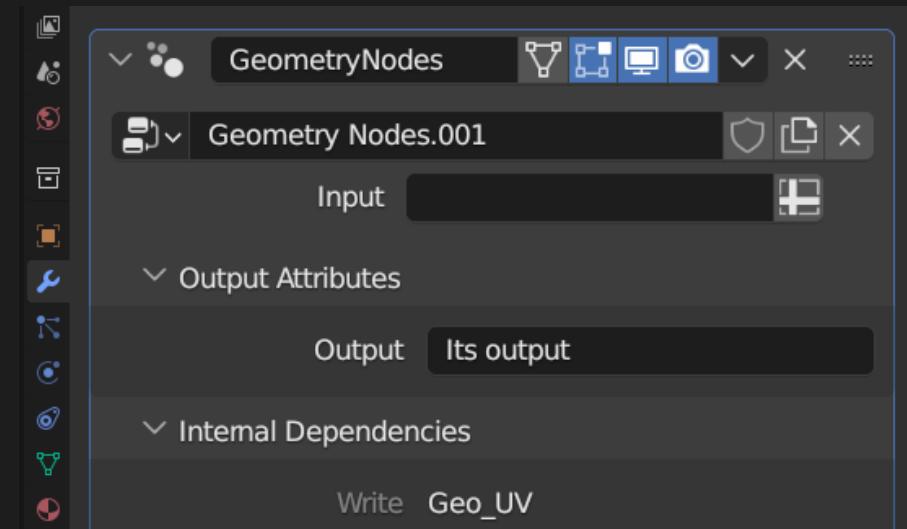
Group Output을 이용하는 방법과 Store Named Attribute를 이용하는 방법이 있습니다.



Store Named Attribute는 앞의 Named Attribute와 대응되는 노드로, Value 소켓에 꽂힌 정보를 이름을 붙여 저장합니다.

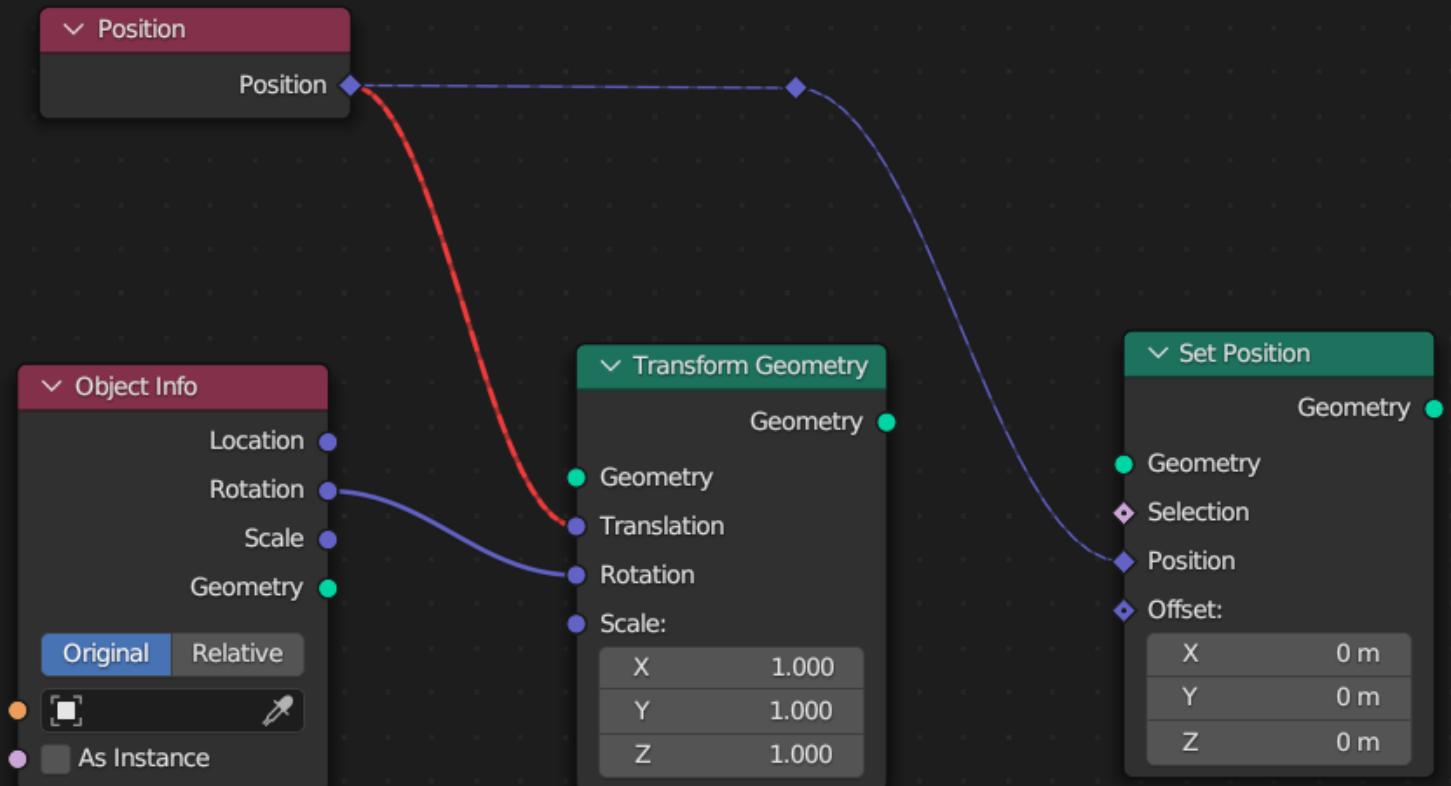


모디파이어 Output을 사용해도 내보낼 수 있습니다.
출력되는 이름은 모디파이어에서 적어줍니다.



032강 지오메트리 노드의 연결구조

Field 란?
지오메트리 노드의 기본 연결방식



소켓 종류

소켓은 색깔 뿐만 아니라 모양으로도 구분됩니다



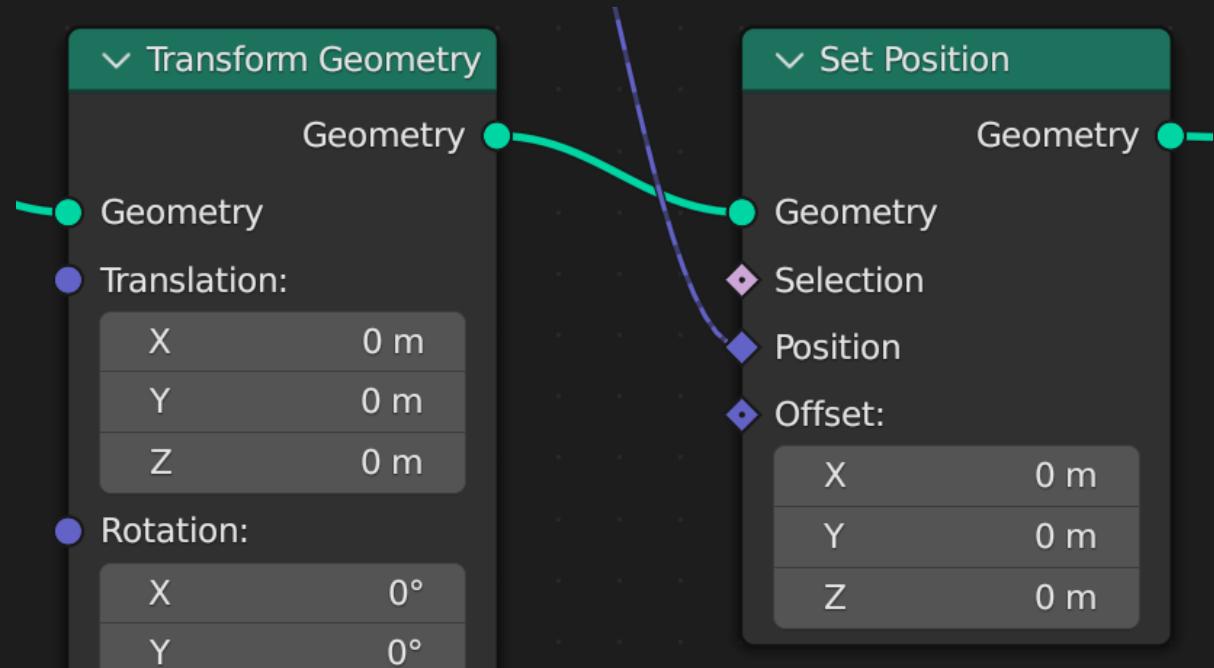
원형 : 단일한 값을 입/출력합니다.
이 소켓에는 마름모 소켓을 입력할 수 없습니다.



마름모 : Attribute 정보를 입/출력합니다.



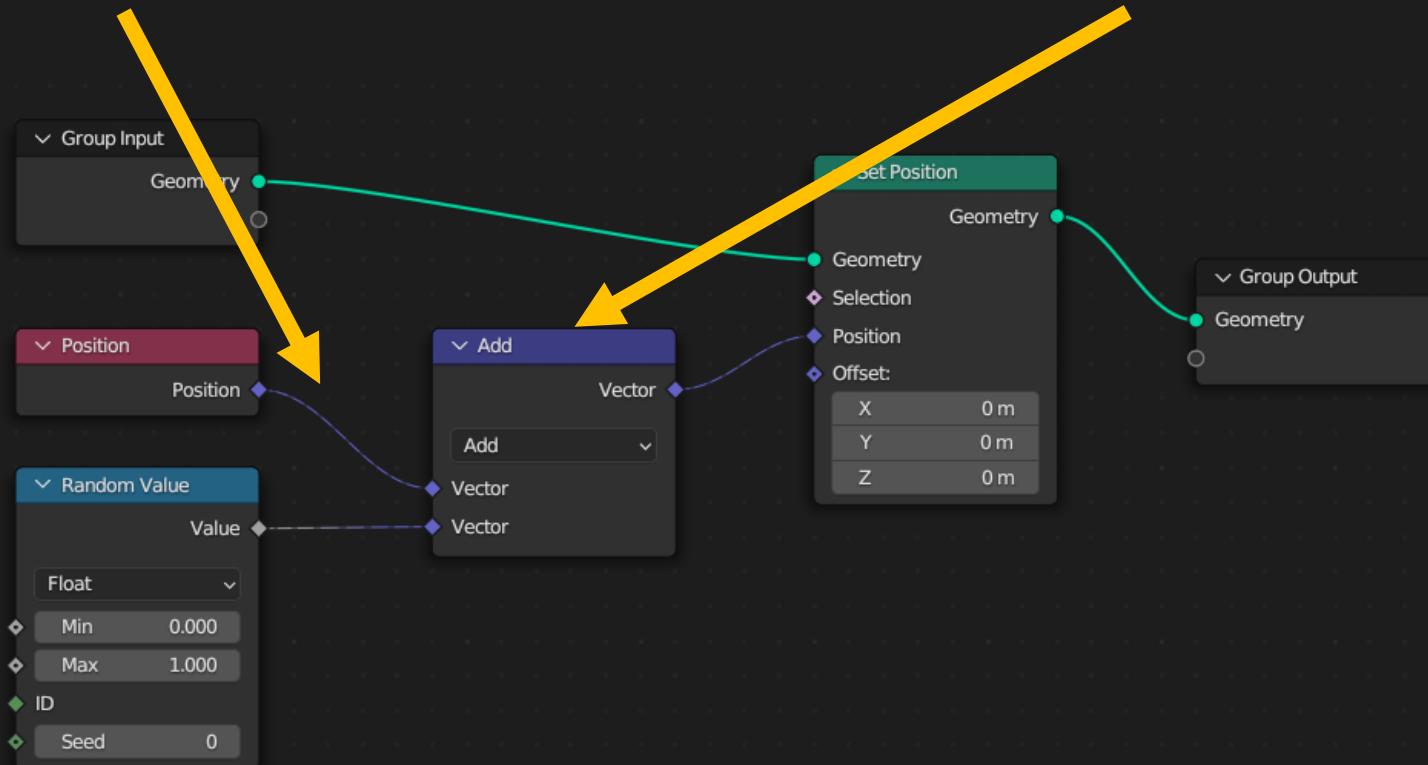
Attribute 정보를 꽂을 수 있지만,
현재 단일값을 입/출력하는 상태입니다.



Field

Field는 블렌더 3.0에서 정착된 지오메트리 노드의 구조를 의미하는 용어입니다.

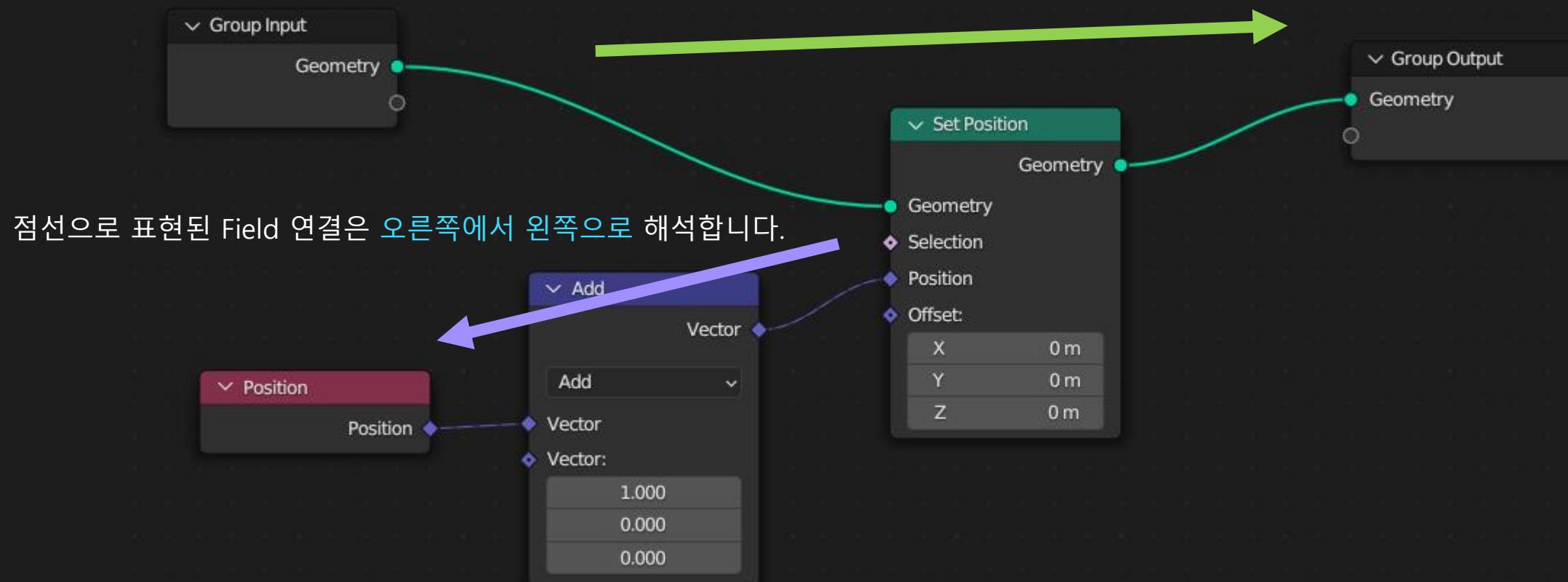
점선으로 표현되는, 지오메트리 노드의 연결방식



Attribute를 받아 계산하는 노드

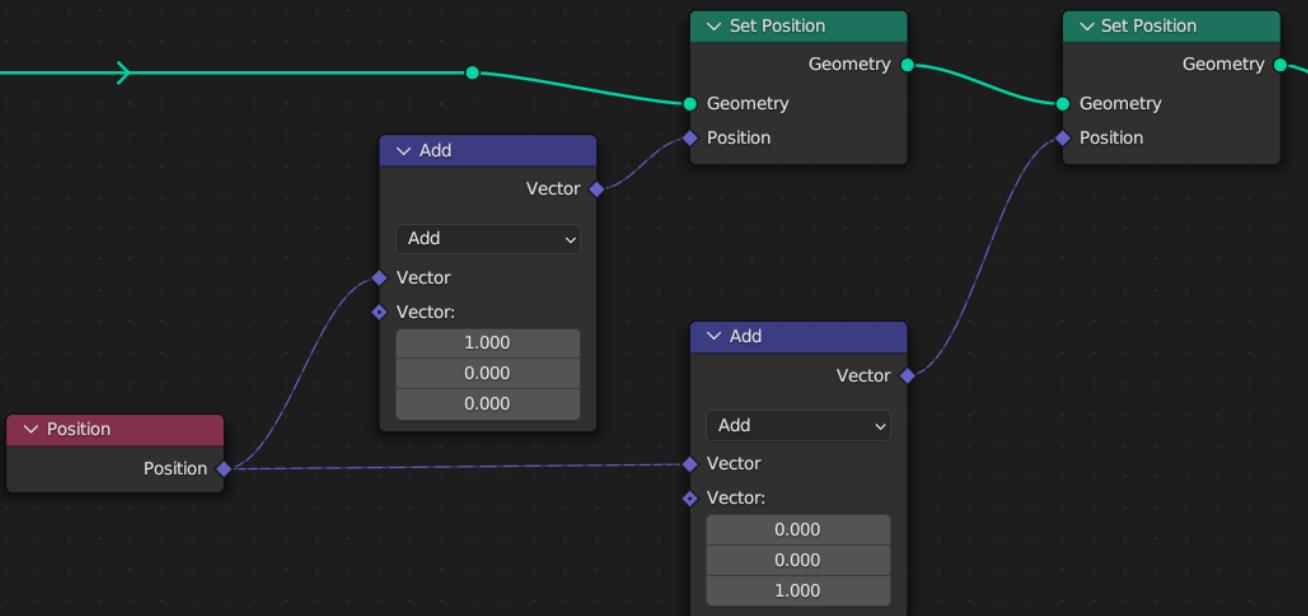
어떻게 작동하나요?

지오메트리의 변형은 초록색 선을 따라 [왼쪽에서 오른쪽으로](#) 이루어집니다.



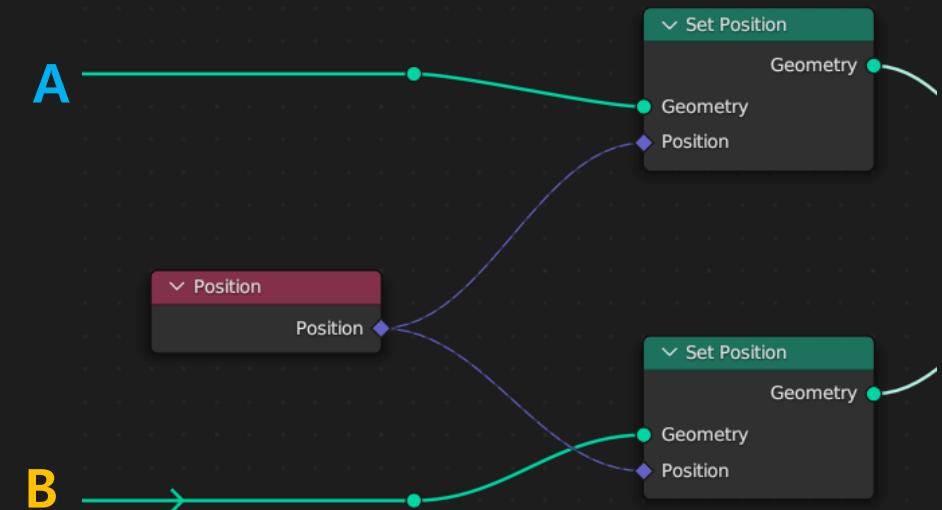
어떻게 작동하나요?

같은 Position 노드를 사용하지만, 첫번째 Set Position과 두번째 Set Position에는 다른 정보가 들어갑니다.



Position 노드는 항상 현재 위치를 내보냅니다.
두번째 Set Position 노드에는 첫번째 노드에 의해
변형된 이후의 위치정보가 들어갑니다.

서로 다른 지오메트리에 같은 Position 노드를 사용해도
같은 결과입니다.



위쪽 Set Position에는 A의 위치정보가,
아래쪽 Set Position에는 B의 위치정보가 들어갑니다.

추가된 데이터타입



Boolean : 참/거짓을 표현하는 데이터타입입니다. Boolean Math도 같이 제공되므로, 불린 연산을 쉽게 할 수 있습니다.



Integer : 정수를 표현하는 데이터타입입니다.
※ 세이더 노드로 곧장 출력시 제대로 작동하지 않으므로 유의.

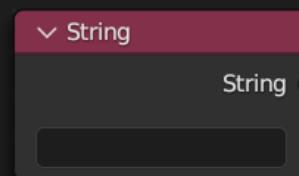
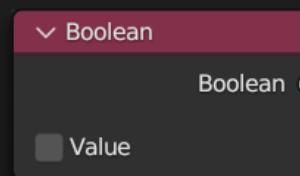
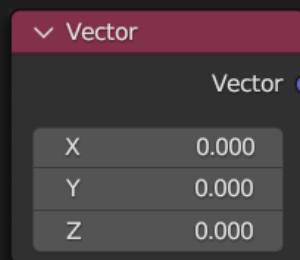
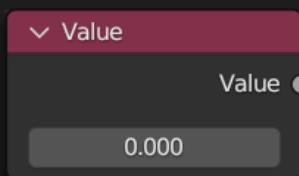


String : 문자를 나타냅니다.
대개 Attribute의 이름을 지정하는 등 제한적으로 사용되지만, 노드를 통해 문자를 시각화 할 수도 있습니다.



Image : 이미지 객체를 나타냅니다.
Color와는 다른 개념이며, 이미지 파일 자체를 나타냅니다. 자주 사용하는 타입은 아닙니다.

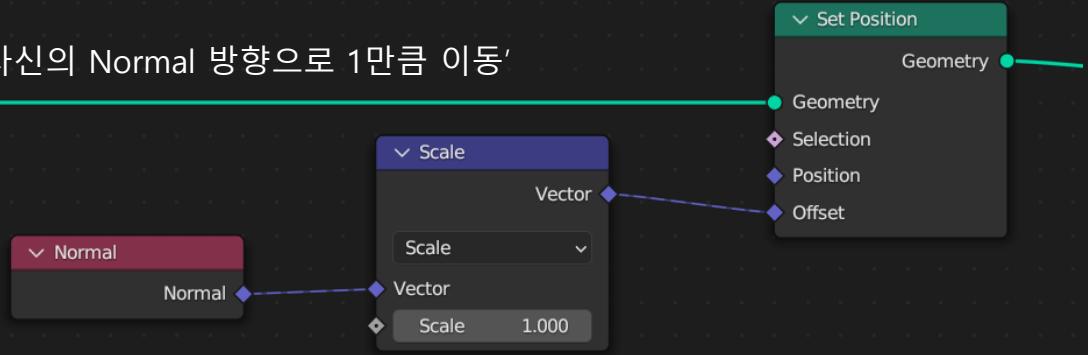
추가된 데이터타입에 맞춰, 기존의 Value노드 같은 입력 노드도 추가되었습니다.



기본적인 사용법

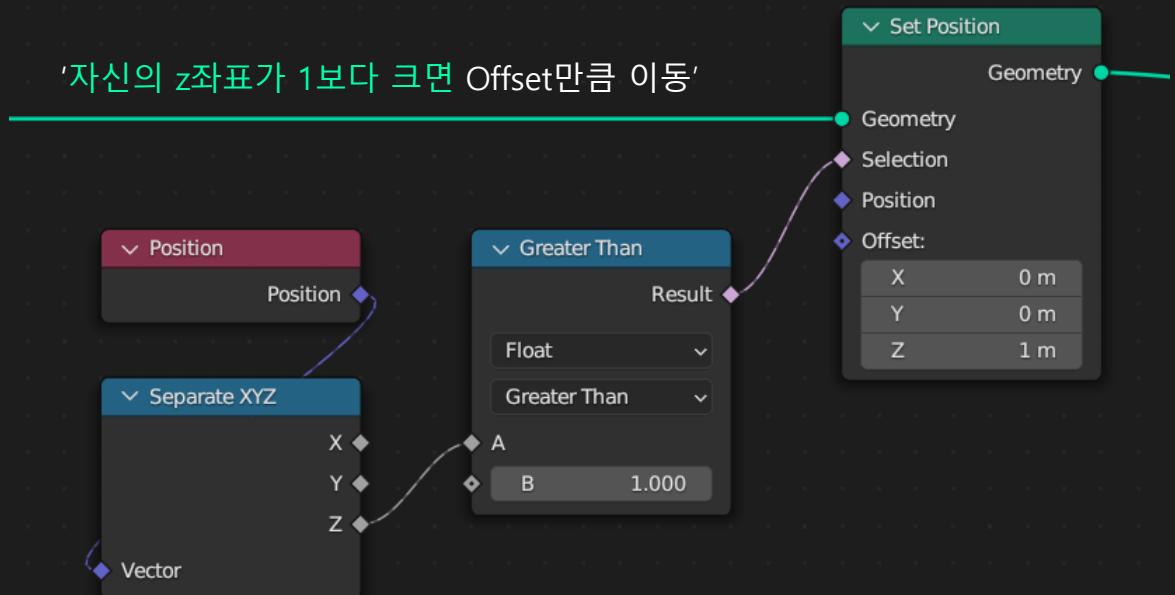
- Attribute를 Math (vector math), Map range 등으로 편집해서 사용한다.

'자신의 Normal 방향으로 1만큼 이동'

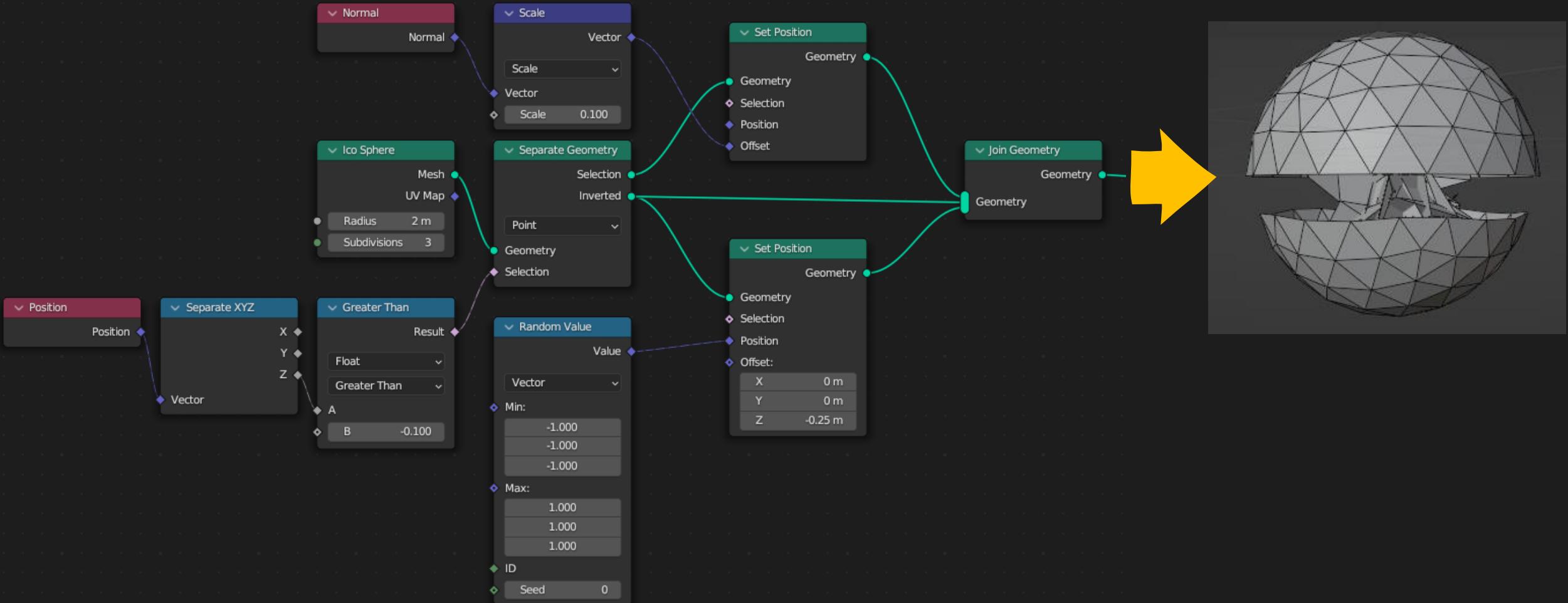


- Attribute 특성을 이용해 Compare node로 골라 선택한다.

'자신의 z좌표가 1보다 크면 Offset만큼 이동'



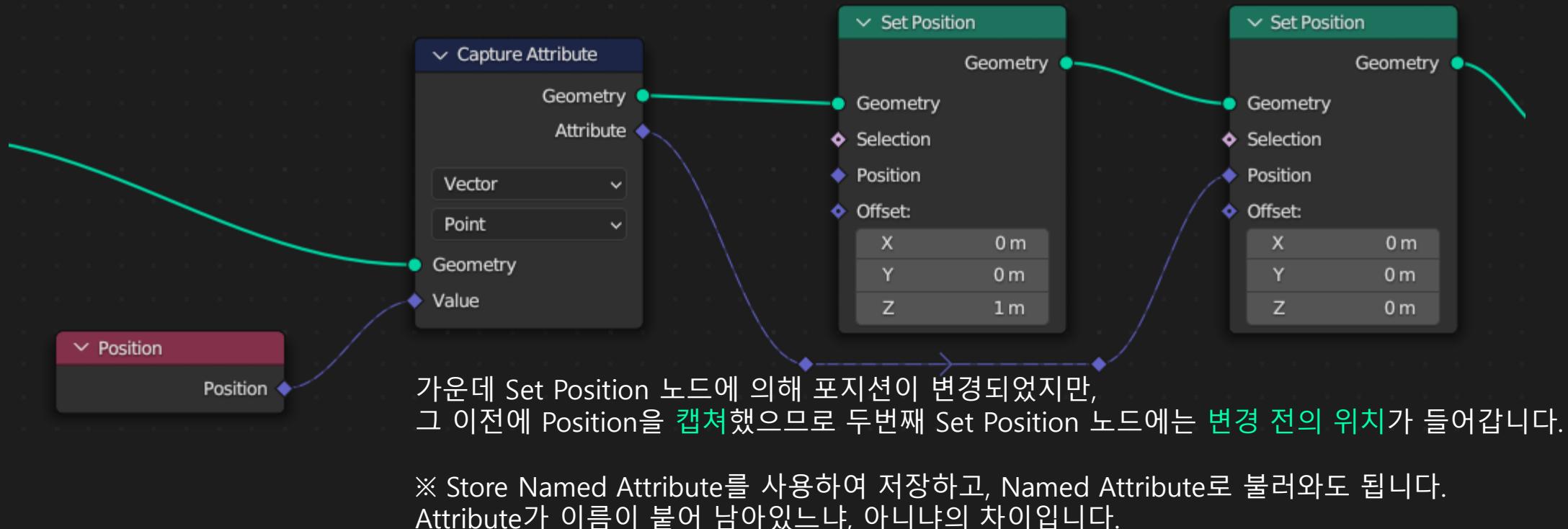
예시



Capture Attribute

정보의 보존

지오메트리가 변화하면, Attribute도 따라서 변화하거나 사라집니다.
현재 상태를 보존하기 위해서는 Capture Attribute를 사용합니다.



지오메트리가 가진 정보

- Domain Size
 - Point Count
 - Edge Count
 - Face Count
 - Face Corner Count
- Mesh
- Geometry

세이더의 텍스쳐 좌표처럼 어떤 위치정보만 사용할 수 있는 것이 아닙니다.
지오메트리 노드는 200개가 넘고, **대부분의 정보에 모두 접근할 수 있습니다.**

예컨대 Domain Size 노드는 지오메트리를 이루는 구성요소의 개수를 셹니다.

- Vertex Neighbors
 - Vertex Count
 - Face Count

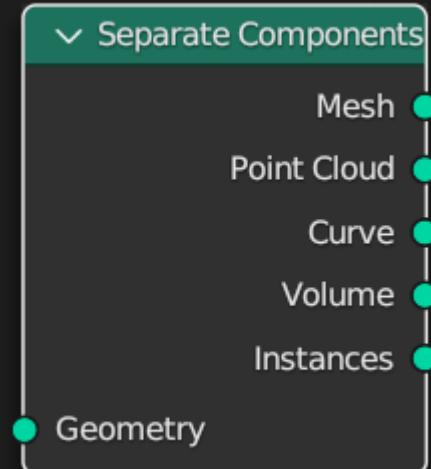
Vertex Neighbors 노드는 점과 이웃한 다른 점이나 면의 개수를 셹니다.
마찬가지로, Edge Neighbors는 선이 만나는 면을,
Face Neighbors는 면이 만나는 점이나 면의 개수를 셹니다.

- Edge Neighbors
 - Face Count

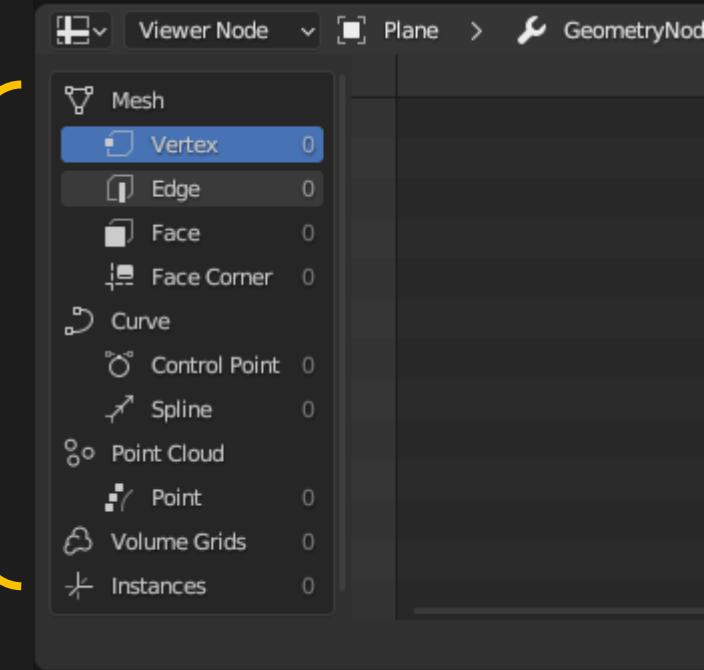
이제부터, 어떤 노드들이 있고 / 그리고 그것을 어떻게 연결하는지 알아보겠습니다.

- Face Neighbors
 - Vertex Count
 - Face Count

지오메트리의 종류

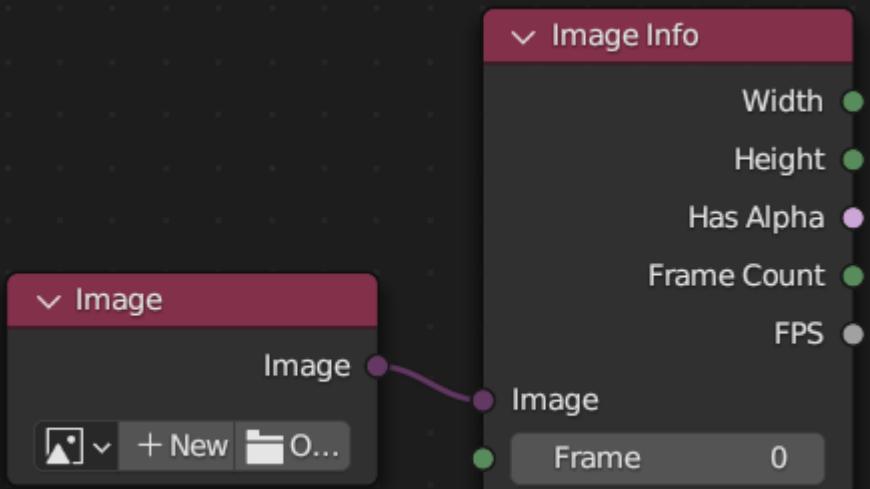


하나의 오브젝트에 여러 종류의 지오메트리를 담을 수 있고,
이렇게 담긴 지오메트리는 [Separate Components](#) 노드를
사용하여 종류별로 분리할 수 있습니다.

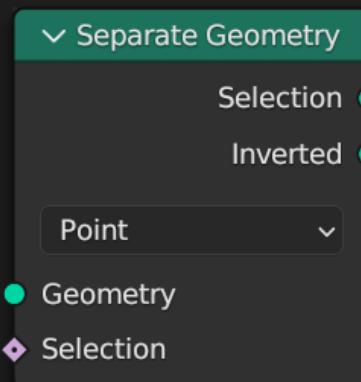


이제부터, 지오메트리의 종류에 따라 단원을 나눠, 차근차근 알아보겠습니다.

Appendix

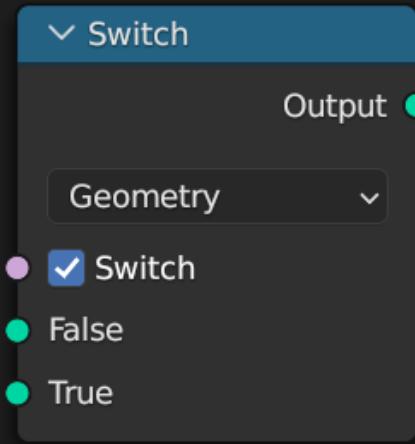


1. 이미지 소켓으로 할 수 있는 것 :
셰이더 노드처럼 Color 정보만 얻는 것이 아니라,
Image Info 노드를 이용하여 이미지의 가로/세로 크기등의 정보를 얻을 수 있습니다.



2. Separate Geometry의 분리기준 :
기본값인 Point로 분리하면, 같은 점을 공유하는 edge, face 중 한쪽이 사라질 수 있습니다.
Edge, face모드로 바꾸어서 확인해 보세요.

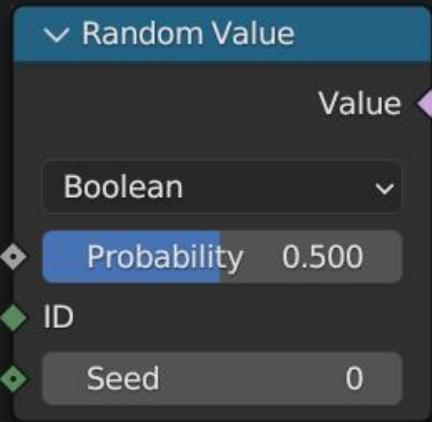
Appendix



Switch를 통해 입력받은 두 값중 하나를 선택하여 출력할 수 있습니다.

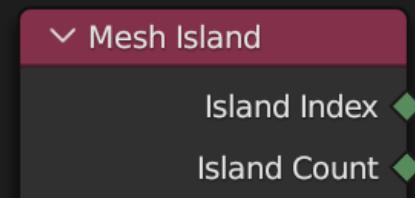
Mix 노드와 비슷하지만, 참/거짓으로 둘 중 하나를 완전히 선택한다는 점,

그리고 모든 데이터타입을 다룰 수 있다는 점이 다릅니다.



3. Random Value는 Float, Vector 말고도 여러가지 다른 모드를 제공합니다.

예를 들어 Integer로 설정하면 랜덤 정수값을 출력하고,
Boolean 으로 설정하면 참 또는 거짓 (1 또는 0) 만 출력합니다.



Mesh Island : 서로 연결되지 않은 지오메트리마다 번호를 매깁니다.
이것을 random value에 연결하면 셰이더 노드의 random per island가 됩니다.
지오메트리 노드에서도 유용하게 쓸 수 있지만, 셰이더 노드로 넘겨 사용하면
훌륭한 random per island 대용이 됩니다.

033강 Instance

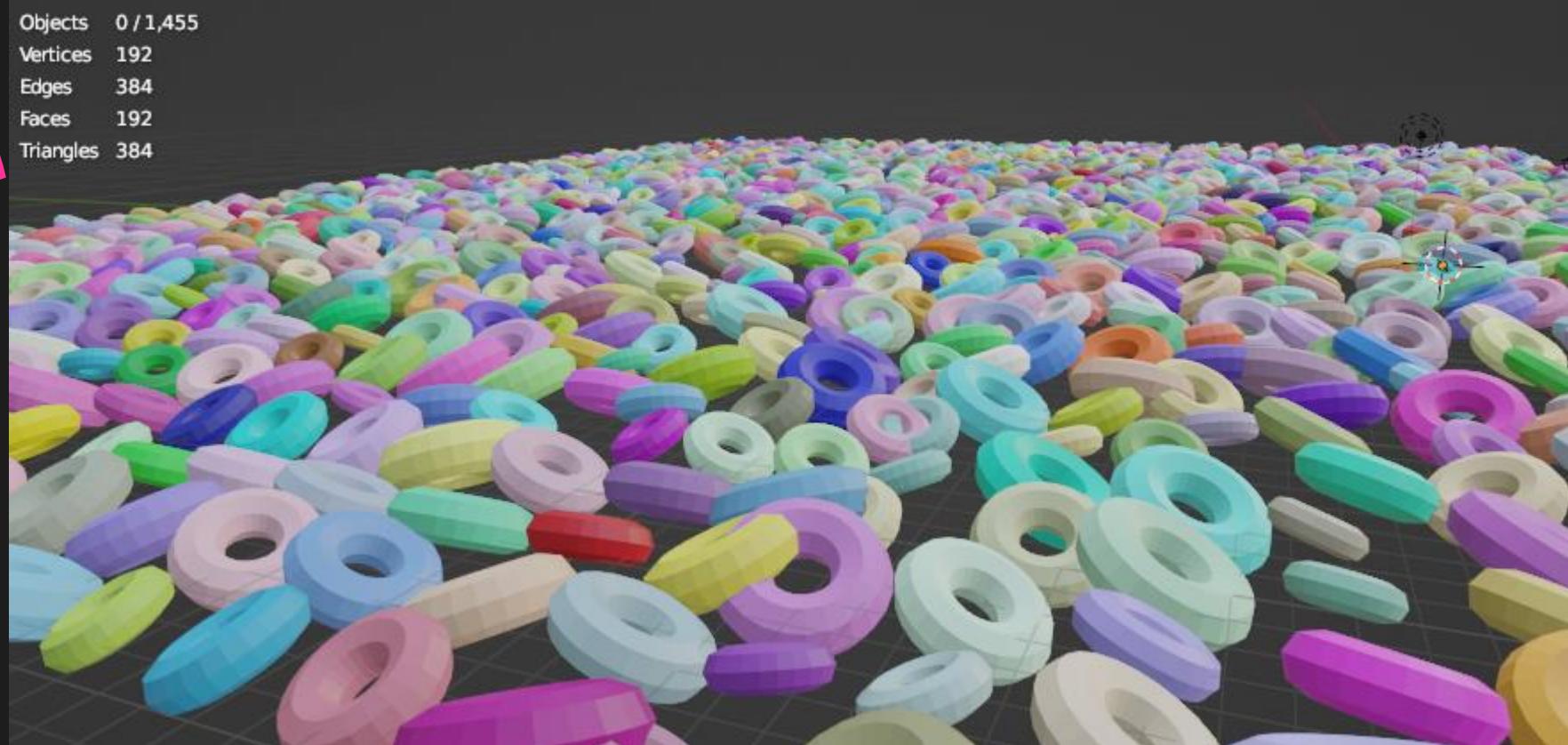
Instance의 정의
인스턴스를 이용한 나무 만들기



Instance

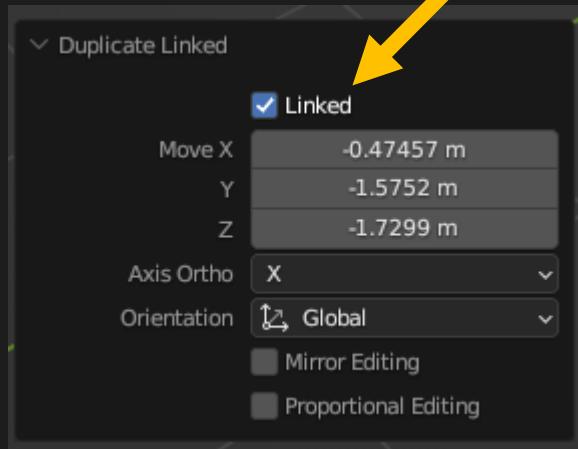
Instance는 같은 지오메트리를 재사용하는 것을 말합니다.
오브젝트 하나를 여러 번 보여주는 개념이기 때문에 메모리와 연산이 획기적으로 줄어듭니다.

Objects	0 / 1,455
Vertices	192
Edges	384
Faces	192
Triangles	384

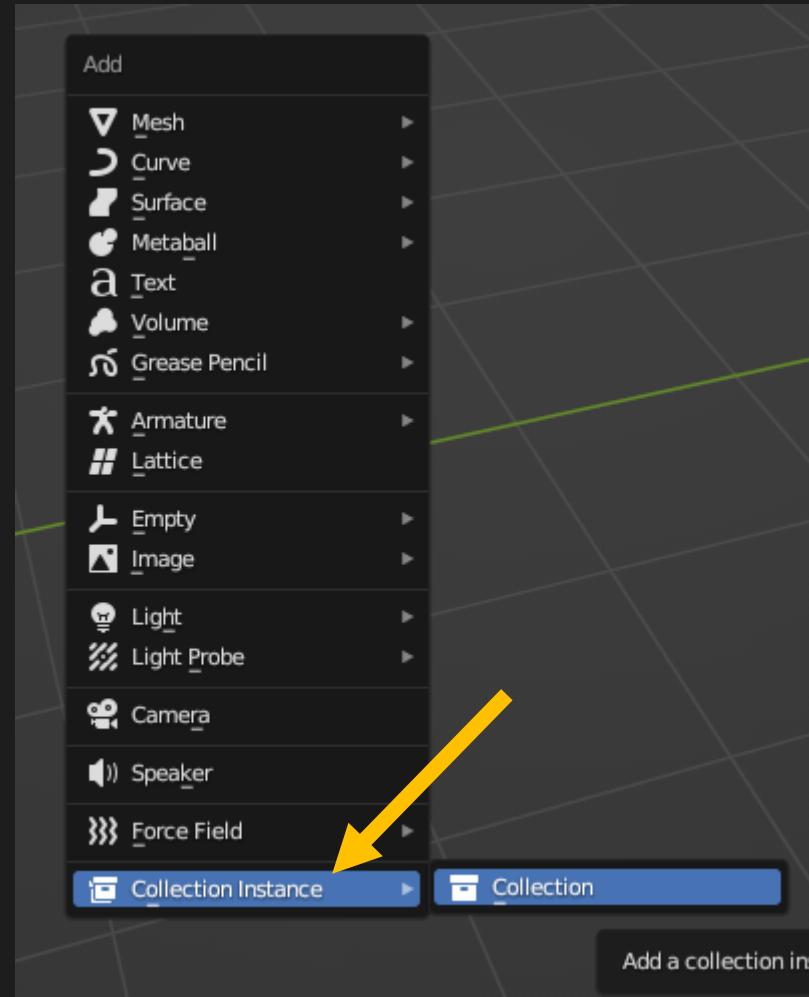


비슷한 기능들

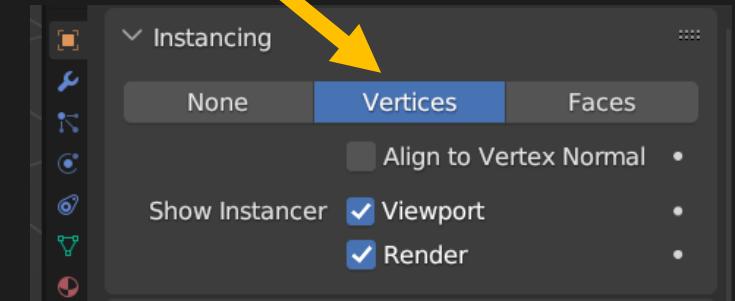
Alt+D로 복제



Collection Instance



Object Instancing



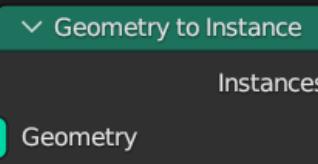
지오메트리 노드의 Instance

지오메트리 노드에서 Instance를 다룰 수 있습니다.
Spreadsheet에서 확인할 수 있으며,
오브젝트 수준의 위치, 회전, 스케일을 컨트롤할 수 있습니다.

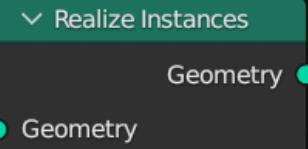
※ 즉, 메쉬 정보에 직접 접근할 수 없습니다. (점, 선, 면)



Name	position			
	0	1	2	3
0 Geometry	0.388	1.355	1.007	0
1 Geometry	1.294	0.901	-0.718	0
2 Geometry	0.487	-0.564	1.564	0
3 Geometry	1.392	-1.018	-0.161	0
4 Geometry	-1.392	1.018	0.161	0
5 Geometry	-0.487	0.564	-1.564	0
6 Geometry	-1.294	-0.901	0.718	0
7 Geometry	-0.388	-1.355	-1.007	0



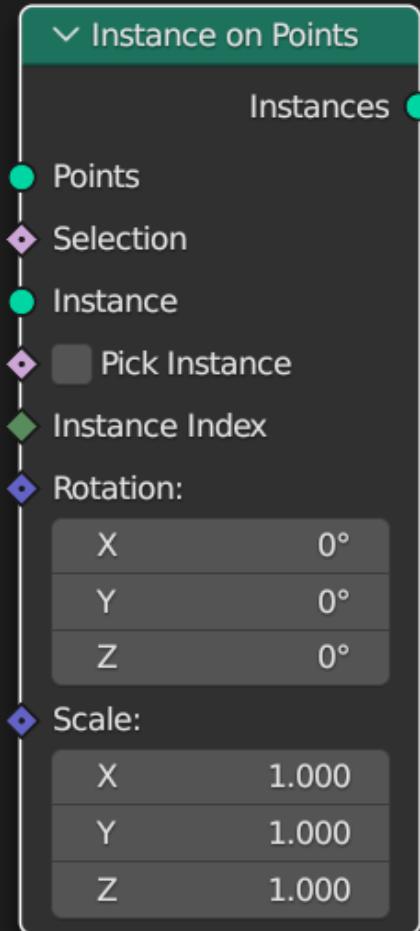
- Geometry to Instance 노드를 통해 인스턴스로 변환할 수 있습니다.



점, 선 면 정보에 접근하려면, 인스턴스를 Realize Instance로 실체화시켜야 합니다.

유명한 노드

Instance on Points

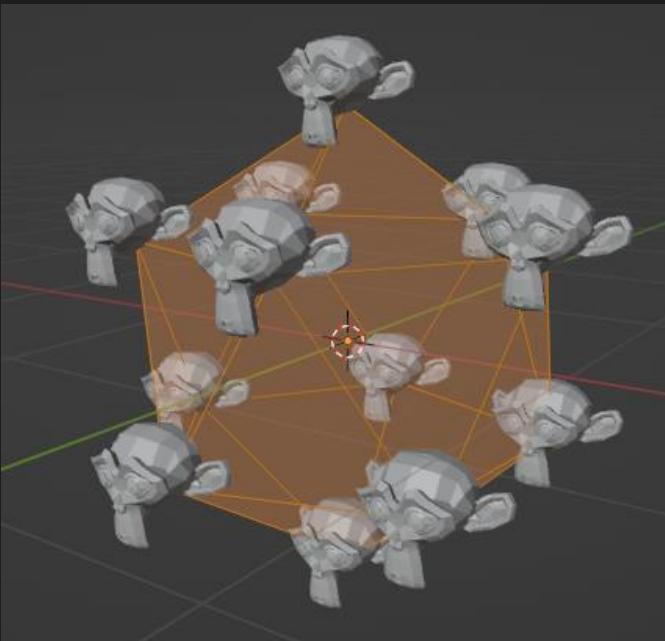


입력받은 각각의 점에 instance를 올려놓습니다.

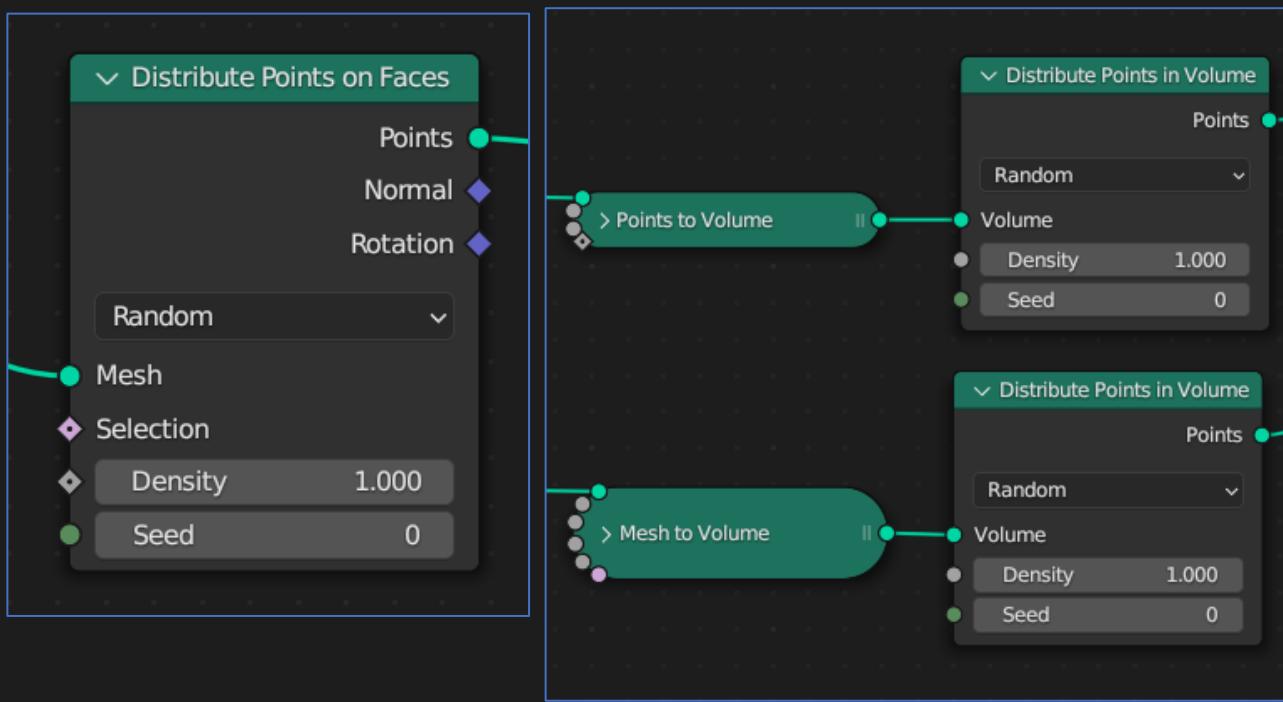
Points, instance는 자동으로 변환됩니다.

그러므로 점이 필요합니다

지오메트리를 그대로 꽂으면
각 점에 인스턴스가 붙습니다.

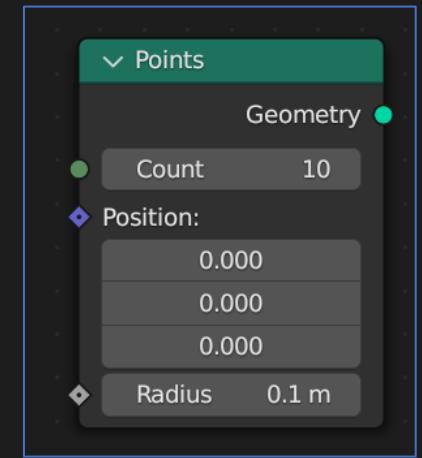


아니면, [Distribute Points on Faces](#)나 [Distribute Points in Volume](#) 노드를 이용하여 표면이나 볼륨에 새로운 점을 생성할 수 있습니다.



※ Distribute Points on Faces는 Face마다 점을 분배합니다.
위치에 따라 밀도를 다르게 하고 싶으면 면이 충분히 나뉘어져 있어야 합니다.

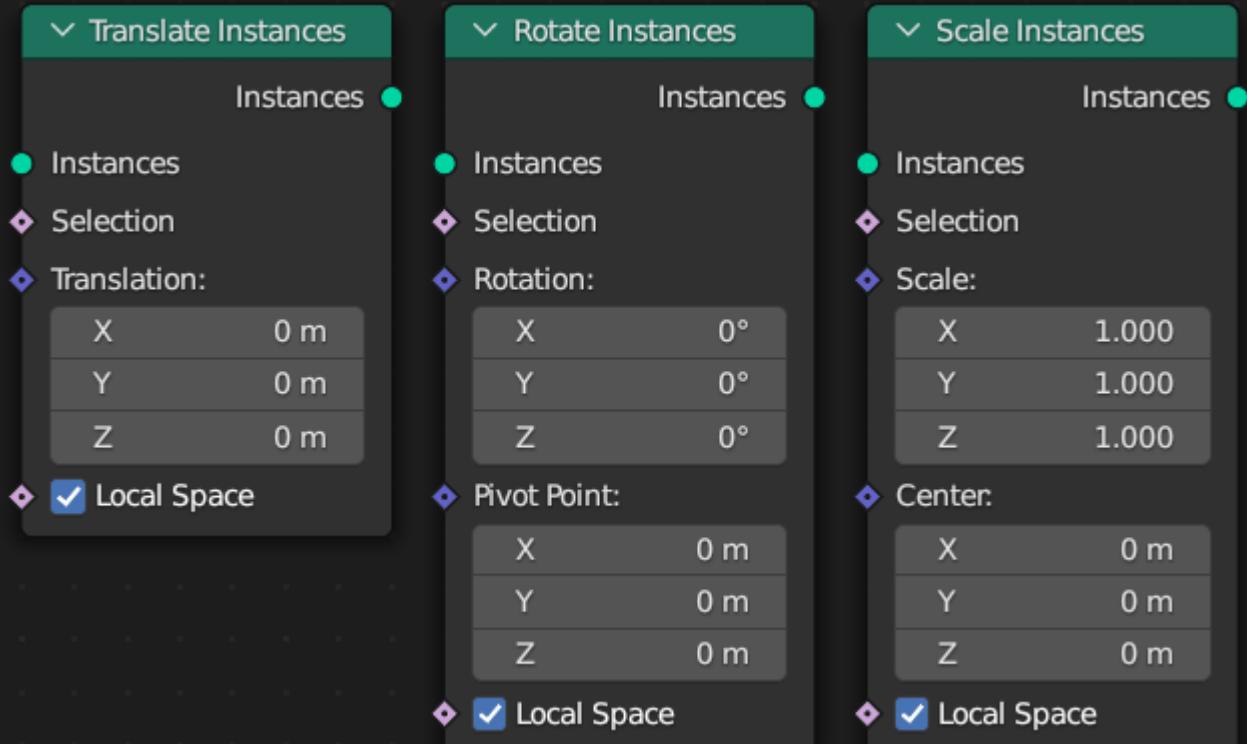
아니면, Points 노드로
포인트를 직접 생성해서
사용할 수도 있습니다.



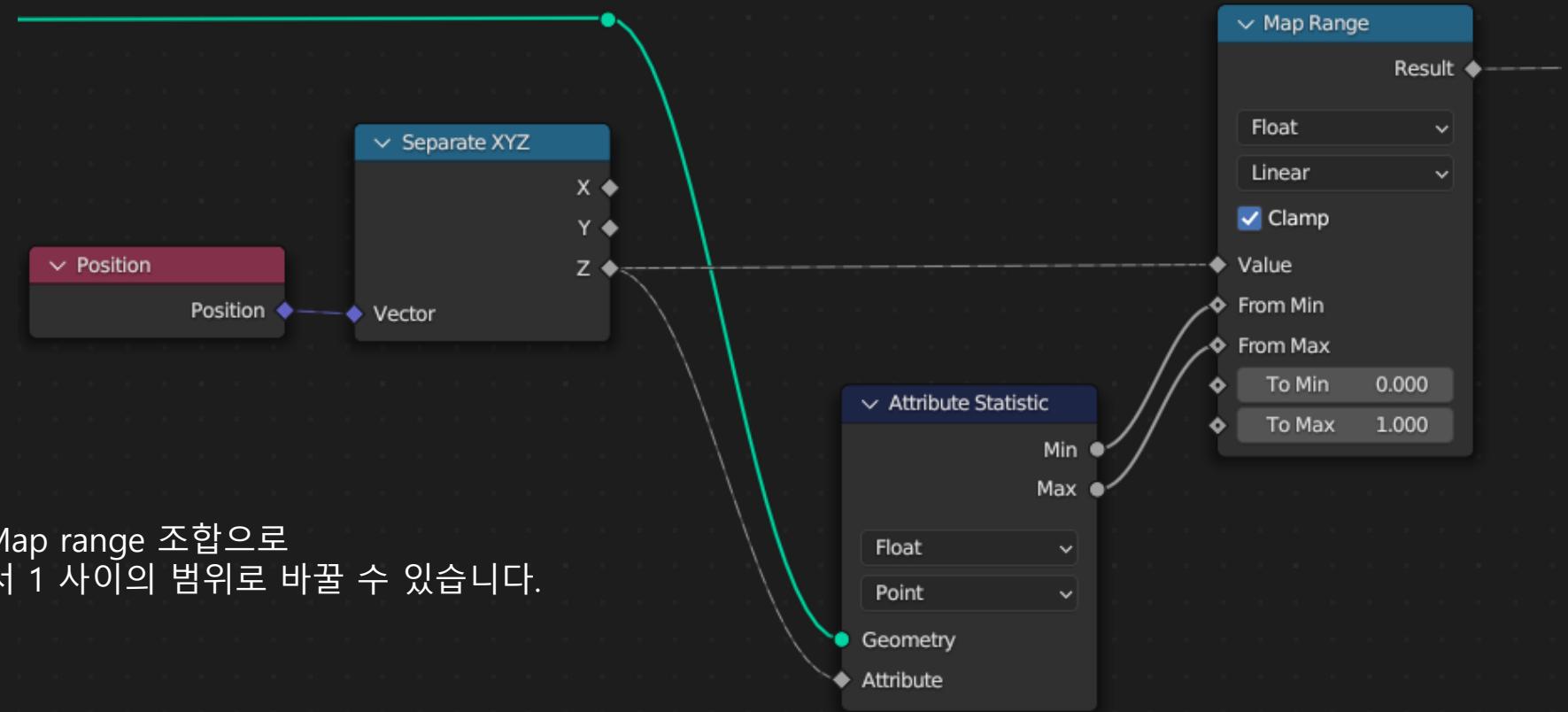
Translate / Rotate / Scale Instance

인스턴스를 이동시키는 노드들입니다.

Set Position으로도 인스턴스를 이동시킬 수 있으나,
회전과 스케일은 불가능하므로 오른쪽 노드를 사용합니다.
특히 Local Space기준으로 움직일 수 있어 유용합니다.



Attribute Statistic & Map Range

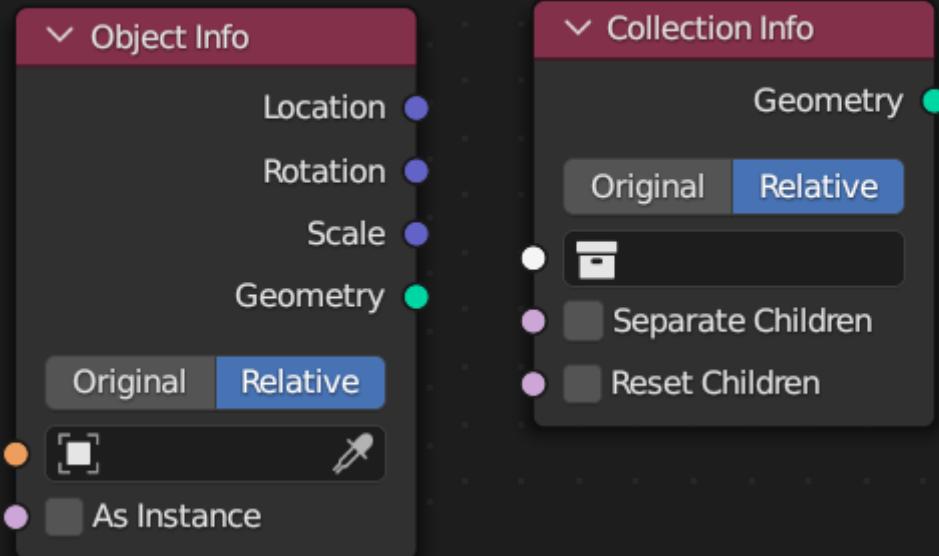


Attribute Statistic과 Map range 조합으로
Attribute의 값을 0에서 1 사이의 범위로 바꿀 수 있습니다.

Object Info / Collection Info

외부에서 오브젝트/컬렉션을 가져옵니다

Original은 오브젝트를 현재 지오메트리의 좌표계로 가져오고,
Relative는 원래의 위치 그대로 유지하여 가져옵니다.

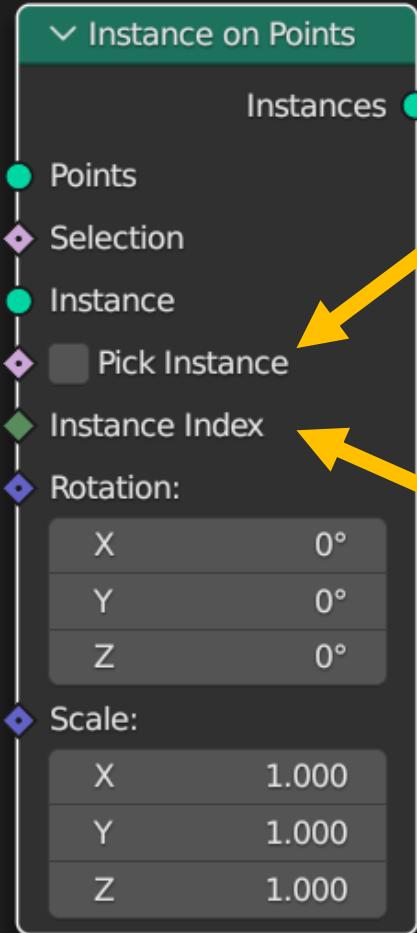


Collection Info 노드는 Collection Instance와 비슷한 개념입니다.
즉, 컬렉션을 [인스턴스](#) 형태로 가져옵니다.

기본적으로 컬렉션 자체를 거대한 하나의 인스턴스로 가져오지만,
Separate Children을 체크하면 컬렉션 안의 오브젝트들을 분리하여
여러 개의 인스턴스로 가져옵니다.

Reset Children : 컬렉션 안의 오브젝트들의 위치를 초기화시킵니다.

Pick instance

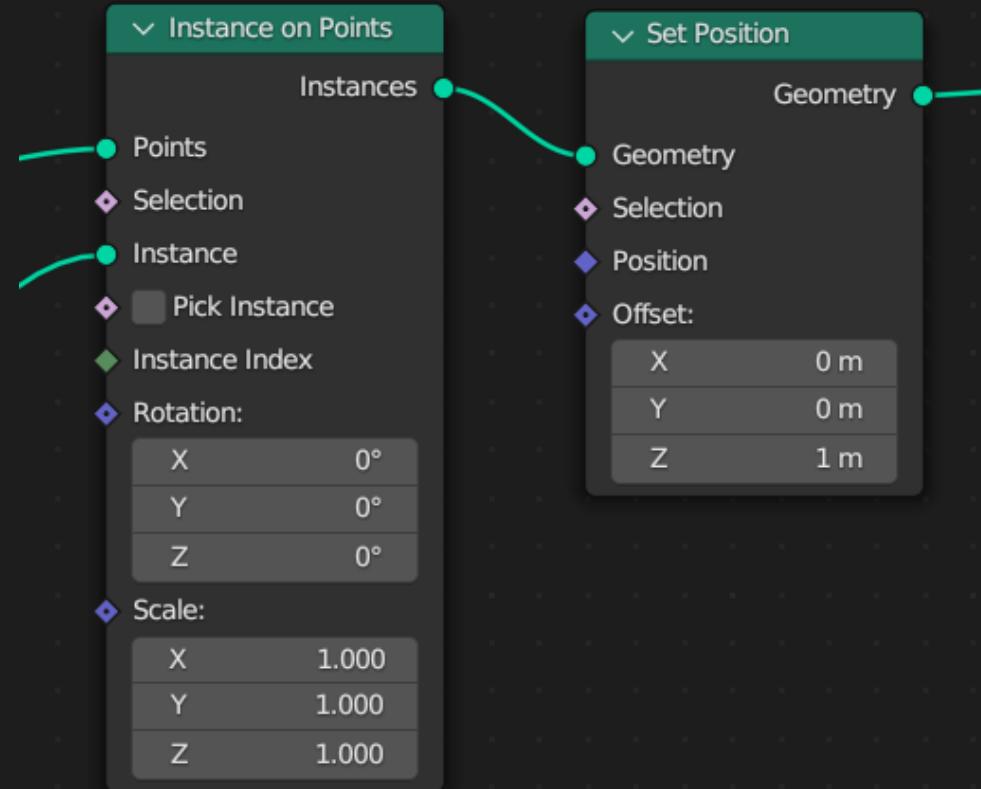


Instance 소켓으로 여러 개의 인스턴스를 한번에 가져올 때,
그중 하나만 선택해서 포인트에 올려놓을 수 있습니다.

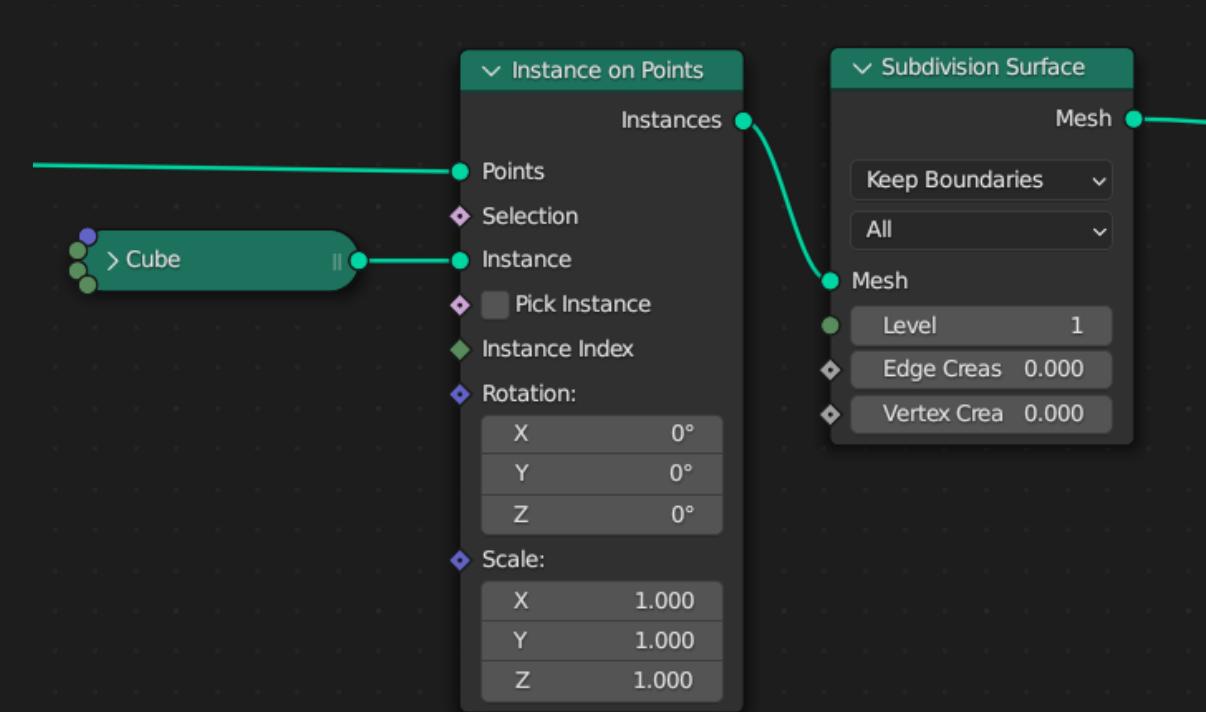
Instance Index로 어느 인스턴스를 선택할 지 고를 수 있습니다.

※작동 원리가 생각보다 복잡합니다. 자세한 사항은 41강 PDF를 참고하세요!

인스턴스에 노드를 연결했을 때 (1)



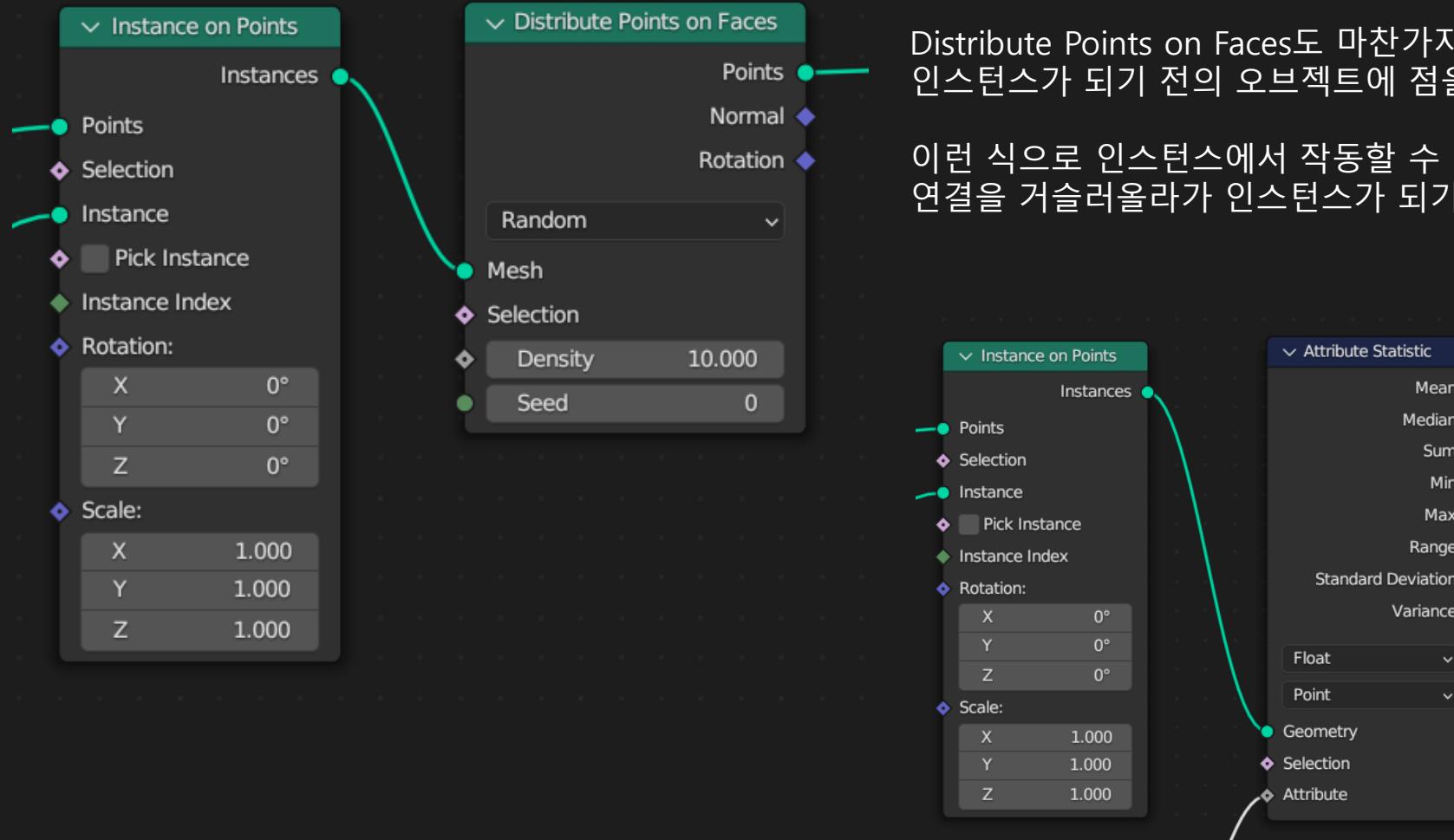
Set Position은 Translate Instance 처럼 인스턴스를 이동시킵니다.
인스턴스의 점, 선, 면에는 접근할 수 없습니다.



인스턴스의 점, 선, 면에는 접근할 수 없으므로
Subdivision Surface는 작동하지 않을 것 같지만
이런 경우 **인스턴스가 되기 전의 메쉬를 변형시킵니다.**

즉 Subdivision Surface는 연결을 거슬러 가서
인스턴스가 되기 전의 Cube에 적용됩니다.

인스턴스에 노드를 연결했을 때 (2)

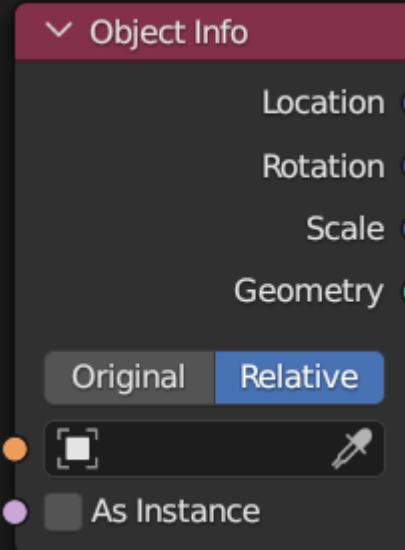


Distribute Points on Faces도 마찬가지로, 인스턴스에는 점을 찍을 수 없으므로 인스턴스가 되기 전의 오브젝트에 점을 찍습니다.

이런 식으로 인스턴스에서 작동할 수 없는 노드가 연결되면, 연결을 거슬러올라가 인스턴스가 되기 전의 오브젝트에 적용되기도 합니다.

하지만 반드시 그렇게 행동하는 것은 아니며 예를 들어 Attribute Statistic은 인스턴스 이전의 오브젝트 정보를 가져올 수 없습니다.

Appendix



Object Info의 Location /Rotation /Scale 를 통해 오브젝트 트랜스폼 값을 받아올 수 있습니다.
그런데, 이 보라색 소켓들의 작동방식은 그 아래의 Geometry 출력과는 작동 방식이 다릅니다.
보라색 소켓과 초록색 소켓의 작동방식을 연결지어 생각하면 머리가 아프니 구분해서 생각해 주세요.
Original일 때 이것은 World의 절대 좌표로 계산되지만, Relative일 때는 현재 지오메트리 기준으로 계산됩니다.
예를 들어, Relative일 때 Location은 '현재 지오메트리로부터 얼마만큼 떨어져 있다' 는 식으로 계산됩니다.
(매우 헷갈리므로, 보통 Rotation /Scale 을 쓸때는 Original로 두고 씁니다..)



Self Object 노드를 이용하여 자기자신의 Location /Rotation /Scale 도 받아올 수 있습니다.
(이 경우 Relative모드는 의미가 없을 것입니다.)

034강 Realize Instance

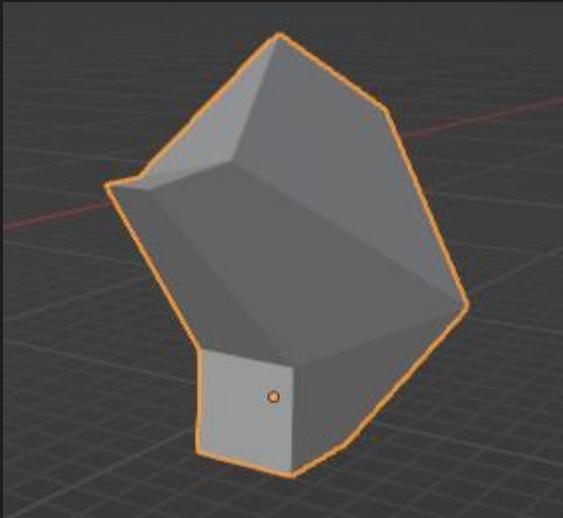
인스턴스를 실체화하여 모델링에 활용하기



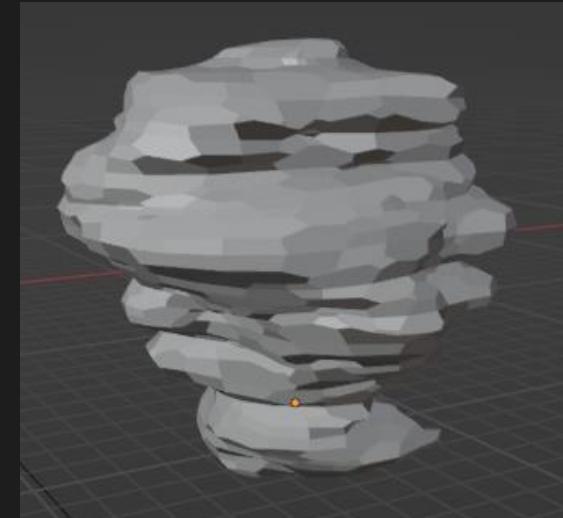
지오메트리 노드의 다른 사용법

여러 개의 인스턴스가 자동으로 생기는 특징을 이용하여, 모델링에 활용해보겠습니다.

Distribute Points in Volume



자동 다듬기

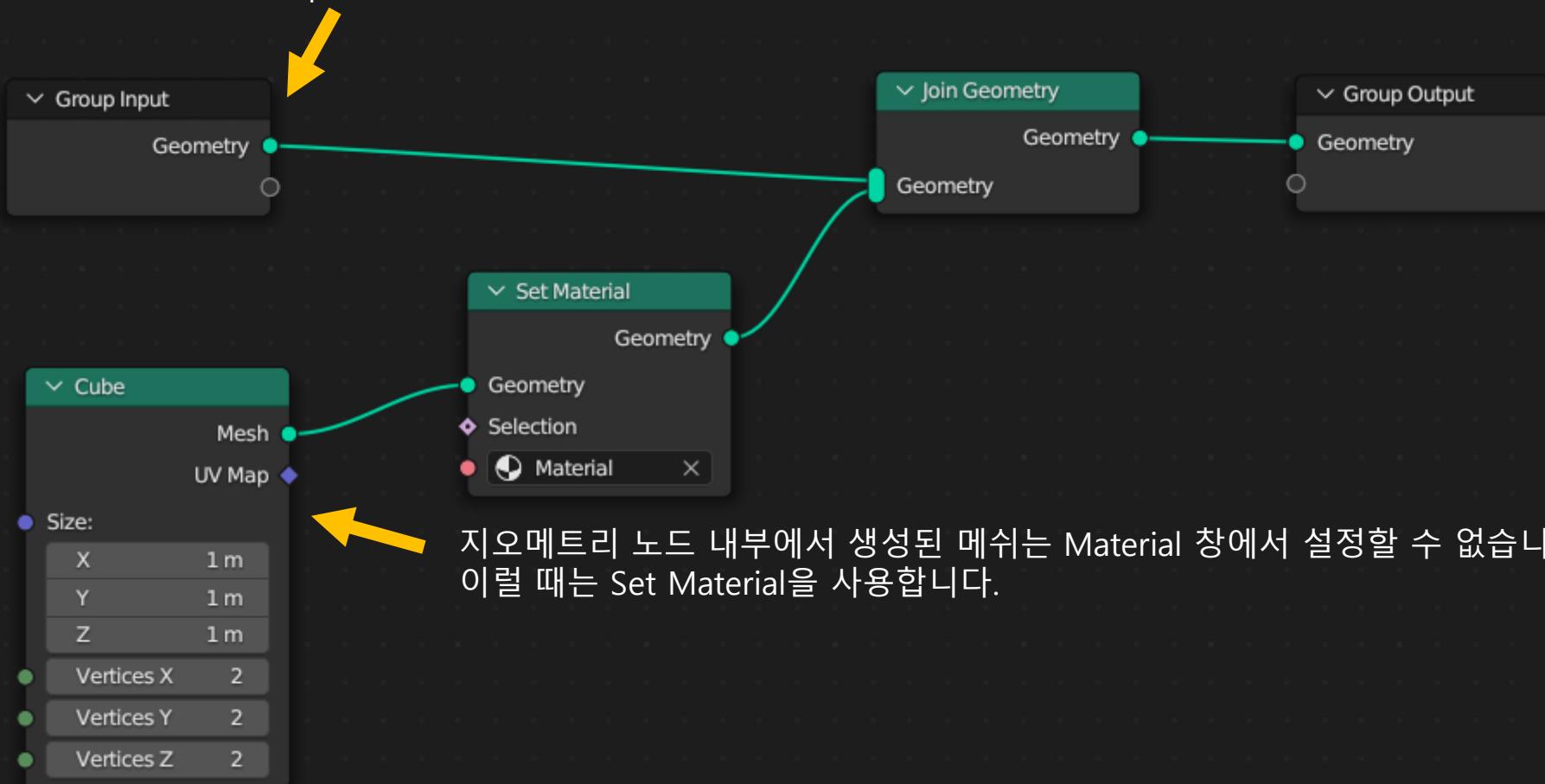


자동 텍스쳐링



지오메트리 노드에서 생성된 Mesh의 Material

Group Input으로 가져온 메쉬는 일반적인 방법
(Material Properties)으로 재질을 설정할 수 있습니다.

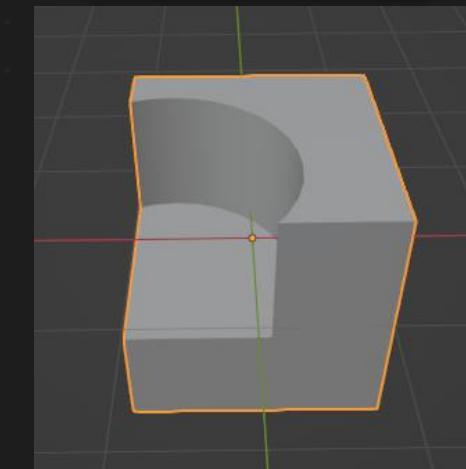
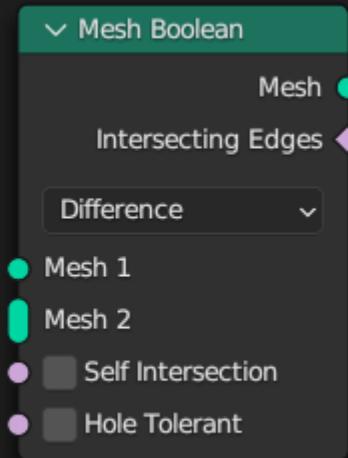
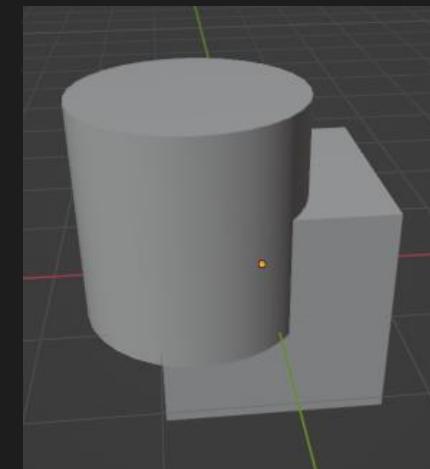
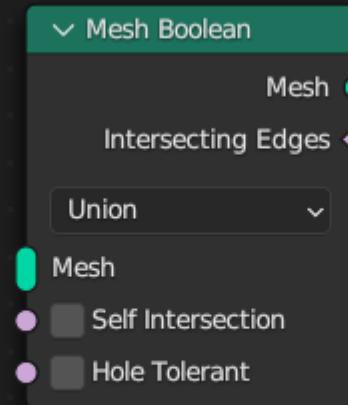
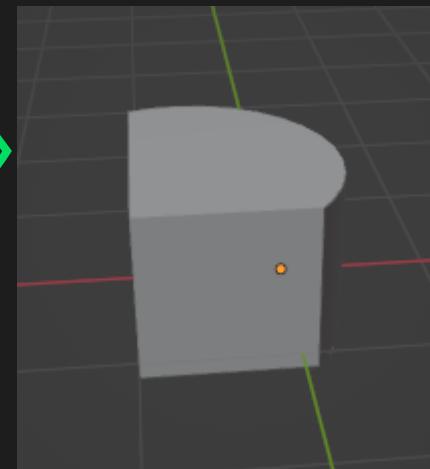
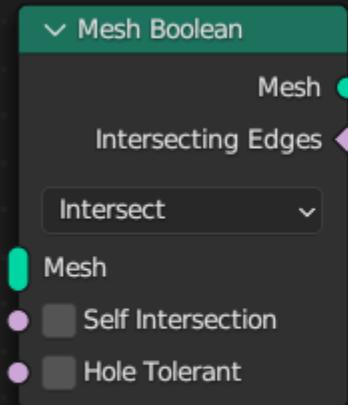
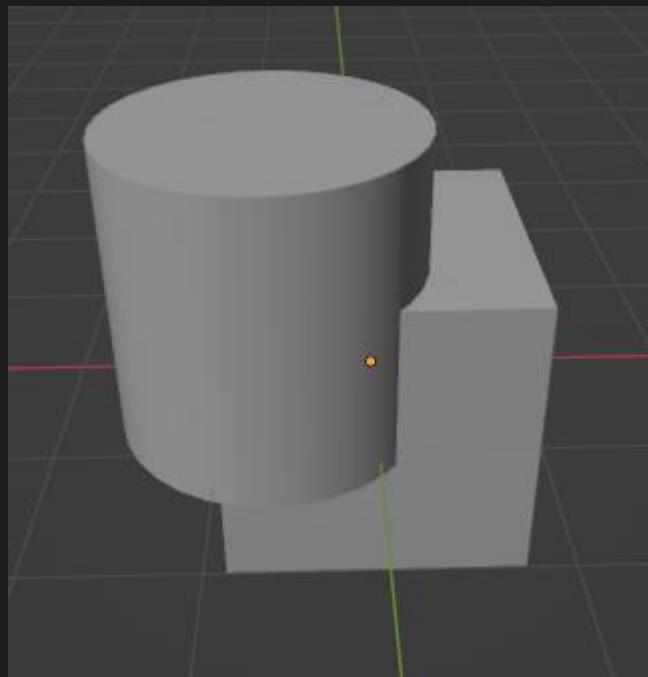


지오메트리 노드 내부에서 생성된 메쉬는 Material 창에서 설정할 수 없습니다.
이럴 때는 Set Material을 사용합니다.

Mesh Boolean Node

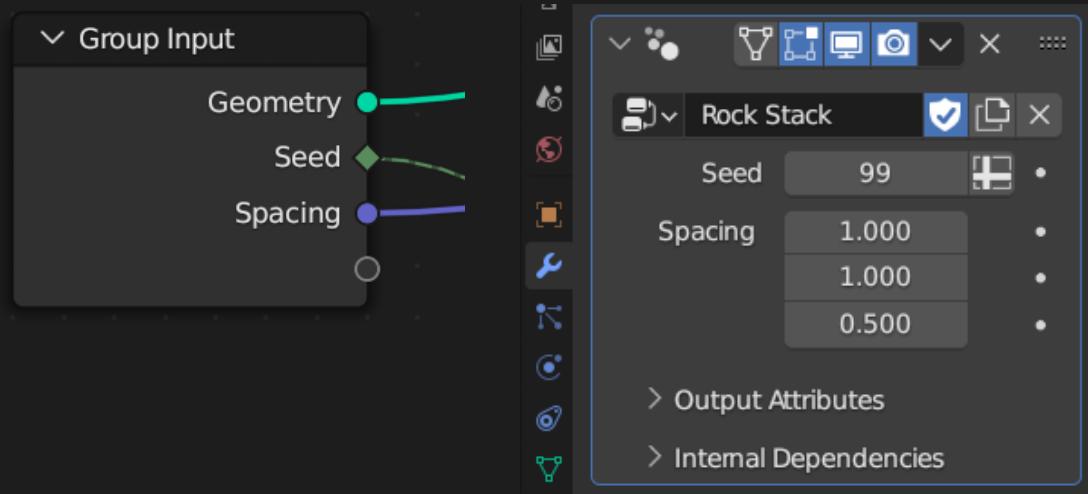
Mesh Boolean은 모델링과 모디파이어의 Boolean기능과 동일하게 작동합니다.

※이 노드는 지금까지 알아본 노드들보다
연산이 매우 무겁습니다. 사용에 주의를 요합니다.

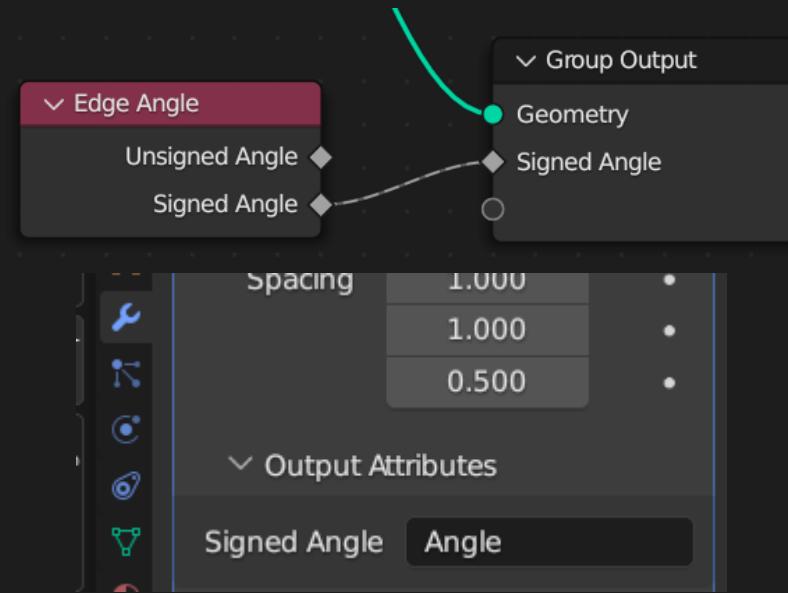


입출력 활용

입력: 소켓을 Group Input에 연결하면,
지오메트리 노드를 여러 개의 오브젝트에 사용할 때
값을 개별적으로 바꿀 수 있습니다.



출력: 소켓을 Group Output에 연결하면,
모디파이어 외부 (세이더 등)에서 사용할 수 있습니다.

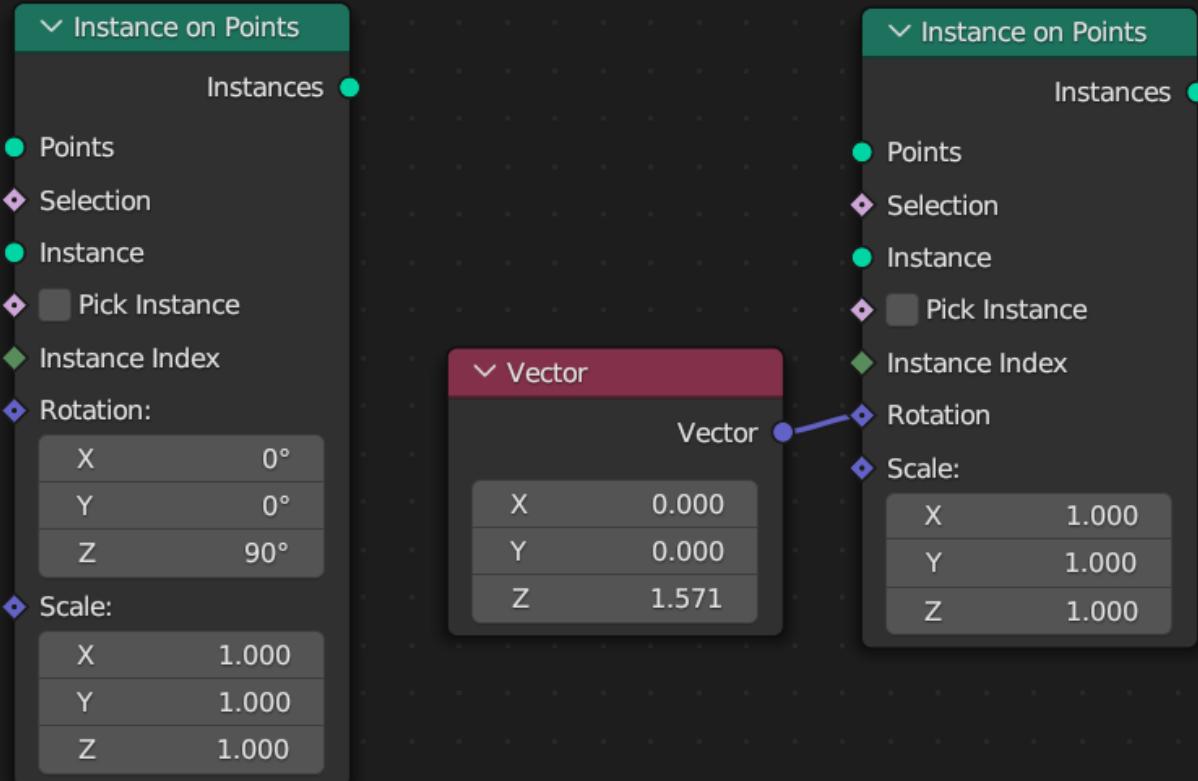


※ 그러나 Group Output으로 빼 경우,
사용할 때마다 매번 변수에 이름을 지정해주어야 합니다.
통일된 이름을 사용하려 한다면 노드 내부에서
Store Named Attribute를 사용하는 것이 편리합니다.

Appendix

라디안 (Radians)에 익숙하지 않은 분들을 위해

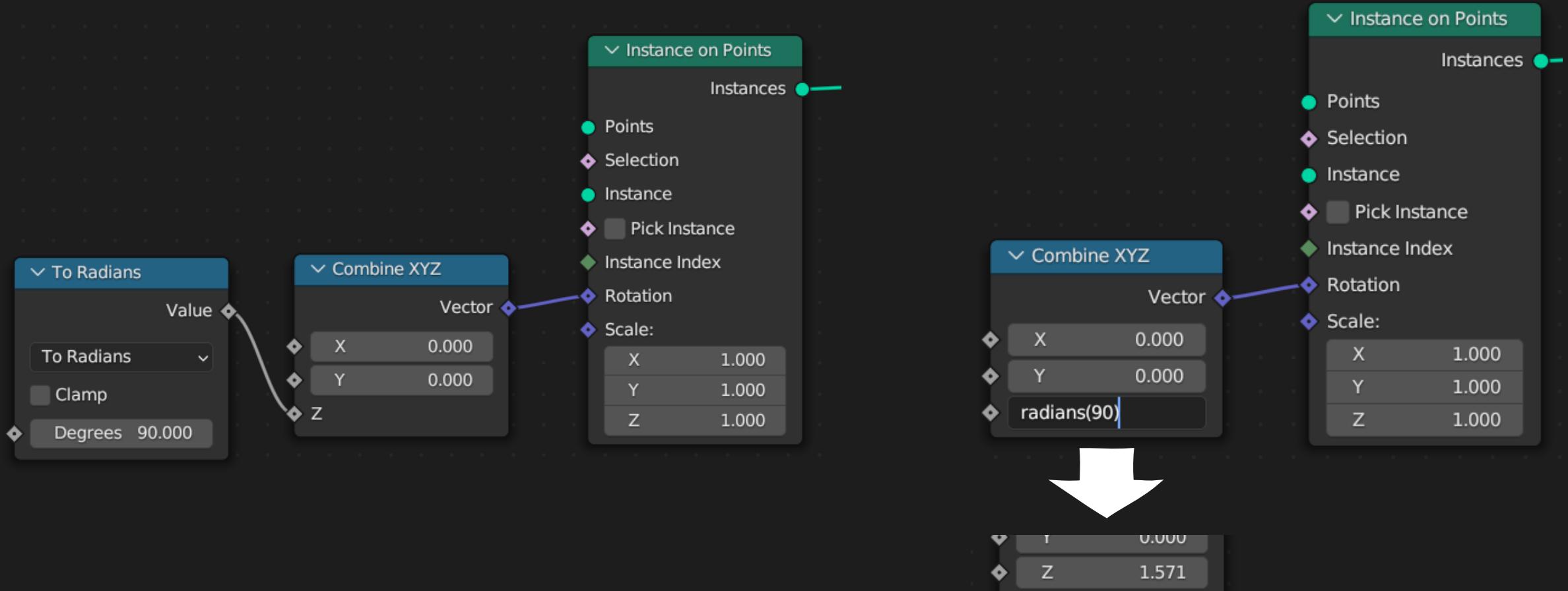
기본적으로 각도는 [육십분법](#)으로 표기되지만, 노드를 연결하기 시작하면 달라집니다.
아래와 같이 각도 표시가 사라지고 일반적인 숫자로 바뀝니다. 이것이 [호도법](#) (라디안각)입니다.
라디안은 180도를 π 로 나타냅니다. 그에 따라 90도는 $\pi/2 = 1.5708\dots$ 이 됩니다.



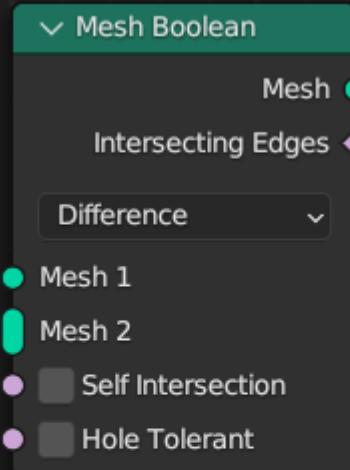
Appendix

라디안 (Radians)에 익숙하지 않은 분들을 위해

사용에 어려움이 있으시면 Math노드의 To radians를 이용하시거나, 파이썬 함수 radians()를 사용하셔도 좋습니다.



Appendix



Mesh boolean의 Self Intersection / Hole Tolerant

Self Intersection : 지오메트리가 자체적으로 겹쳐 있을 때 제대로 작동하지 않으면 체크합니다.
Hole Tolerant : 표면이 닫혀 있지 않고 구멍이 뚫려 있어 제대로 작동하지 않을 때 체크합니다.

체크하면 더 정확한 연산이 나올 수 있지만, 대신 느려집니다.



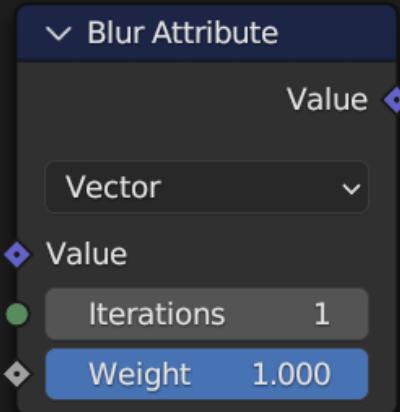
Edge Angle의 Unsigned / Signed

Unsigned는 두 면이 이루는 각도를 계산합니다.

Signed 각은 두 면이 이루는 각을 어떤 한쪽 방향 기준으로 계산합니다.
즉 안쪽으로 굽은 것과 바깥으로 굽은 것이 구분됩니다.

Edge Angle은 두 면이 완전히 평행 있을 때가 0도이고, 완전히 접혔을 때가 180도입니다.
조금 비직관적이므로 유의해주세요.

Appendix



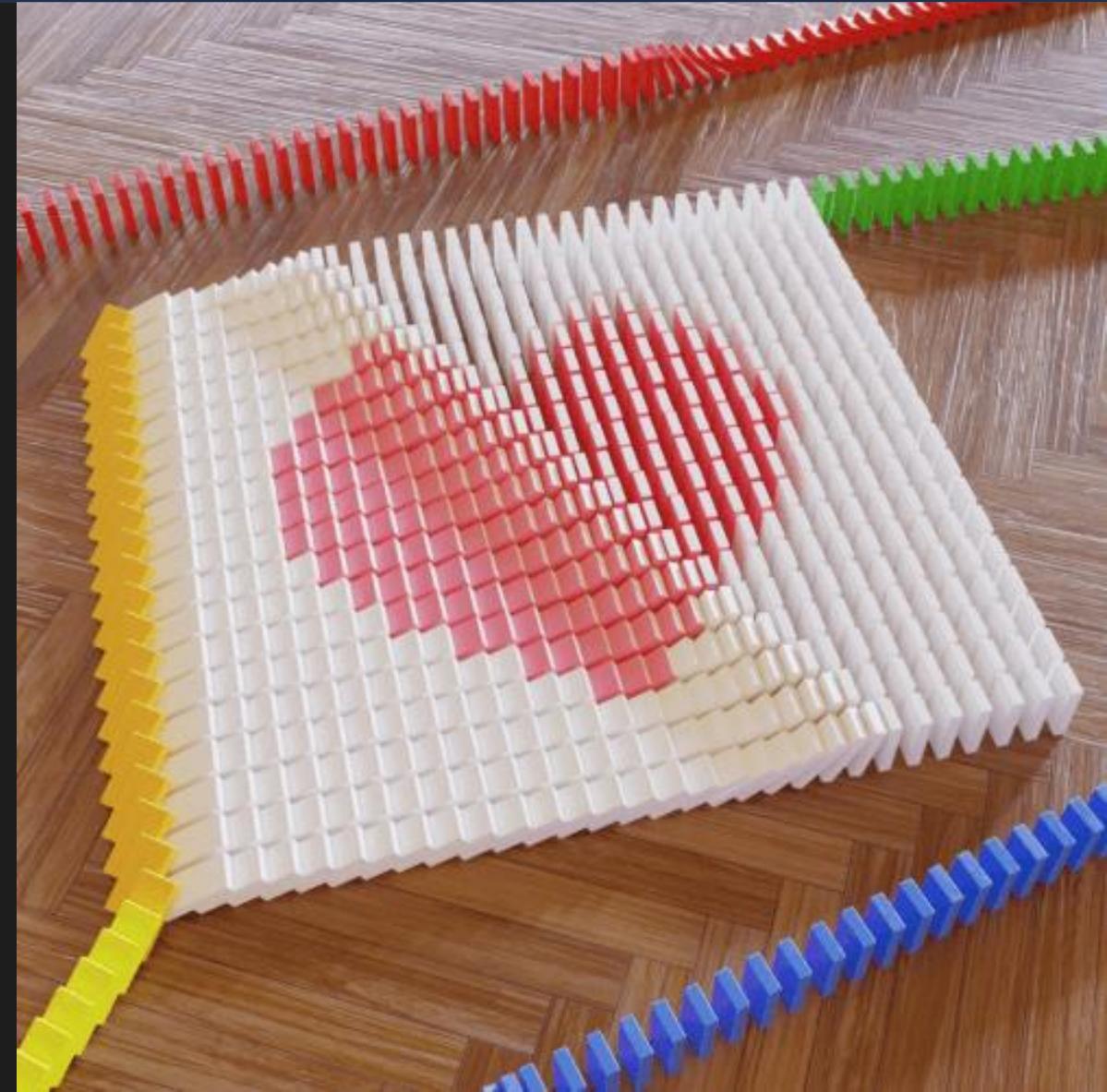
Blur Attribute : 말 그대로 Atribute를 '흐립니다'.

연결된 점, 선, 면을 따라 흐려지므로, 점들이 연결되어 있는 메쉬와 커브에만 사용할 수 있고
점들이 연결되지 않은 경우에는 사용할 수 없습니다.

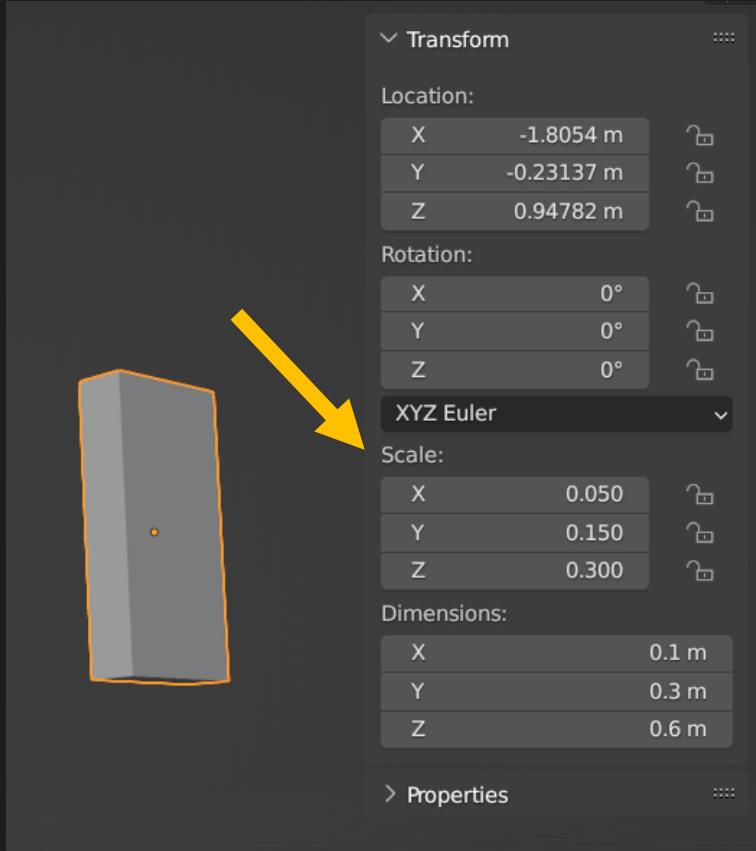
035강 Instance Animation (1)

도미노 애니메이션

Shader Nodes에서 사용했던 노하우를
지오메트리 노드에서 활용하기
- Add와 Multiply



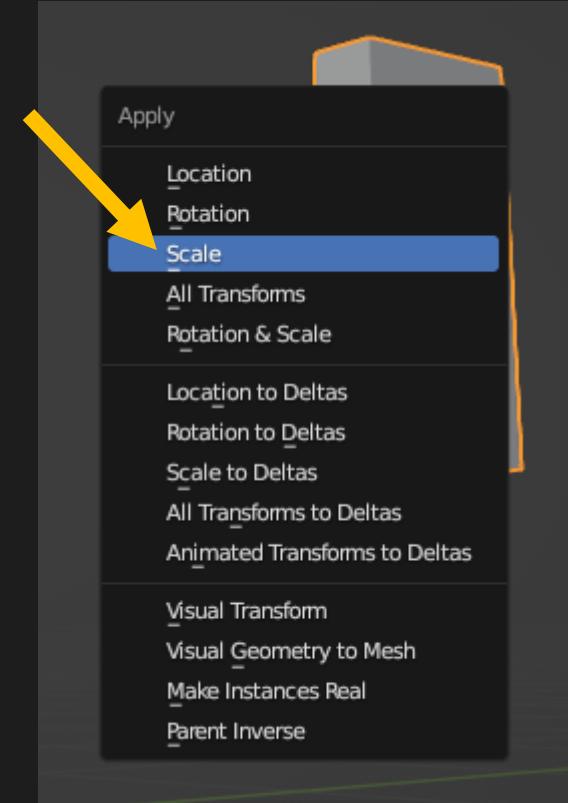
Apply Scale



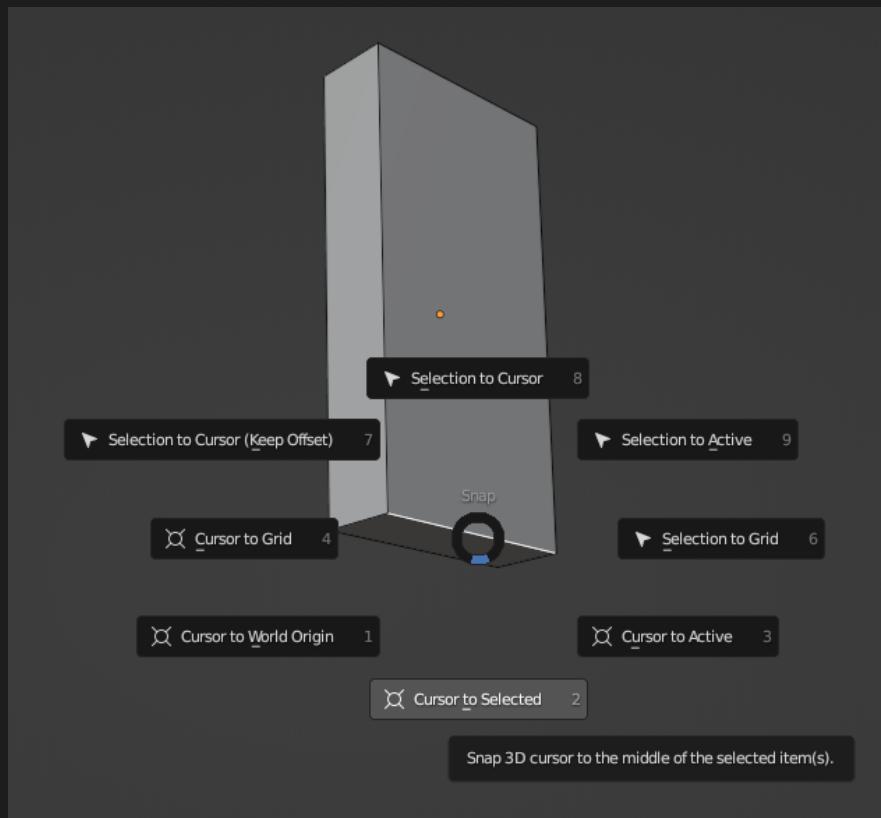
스케일이 1이 아니면 예기치 않은 결과를 얻을 수 있습니다.

Ctrl+A로 Apply transform 창을 열어, **Apply Scale**을 해 주면

현재 상태가 Scale 1로 등록됩니다.



오브젝트 오리진 변경

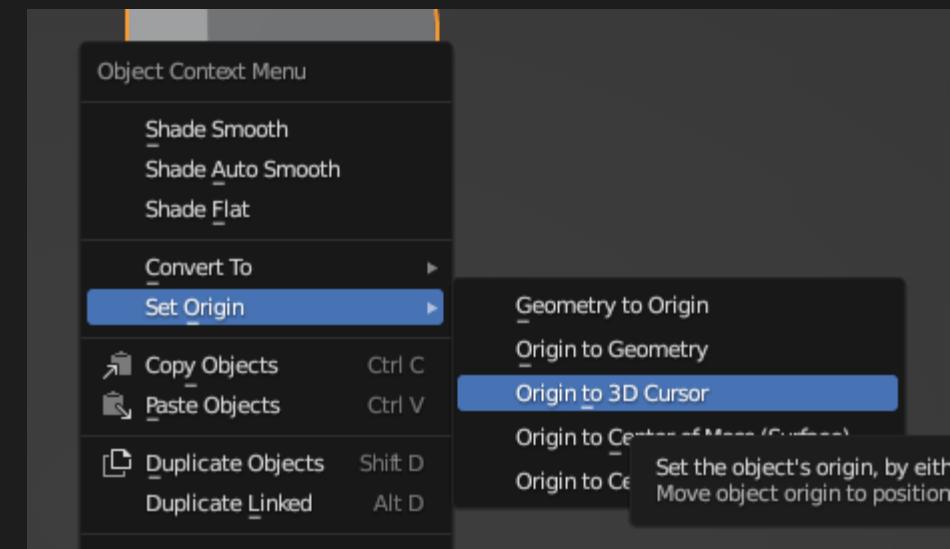


Shift+S의 파이메뉴를 활용합니다.

Edit 모드에서 오리진이 위치할 점, 선, 면을 선택하고

Shift+S ▶ Cursor to Selected로 커서를 갖다놓습니다.

Object 모드로 나와서 마우스 우클릭 ▶ Set Origin ▶ Origin to 3D cursor 를 선택합니다.

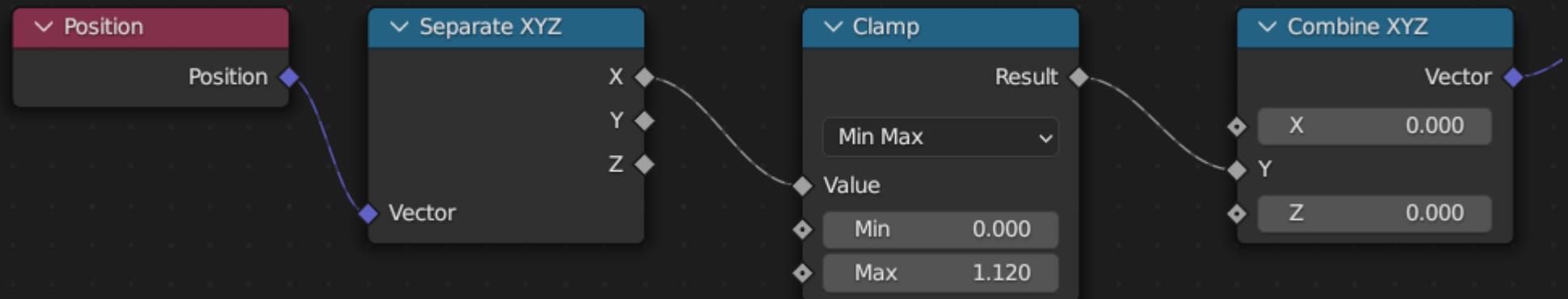
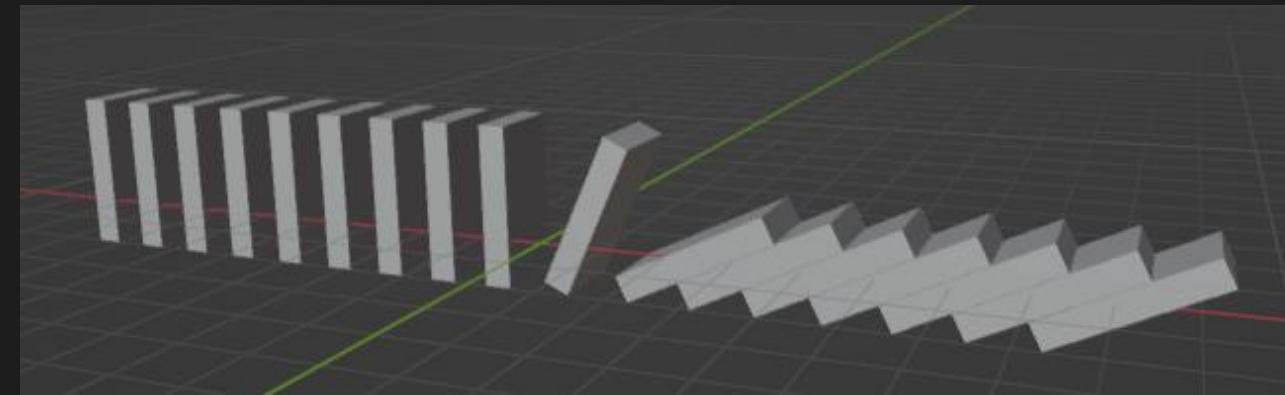


범위 문제

셰이더에선 많은 것들이 0~1 사이에서 컨트롤되기 때문에

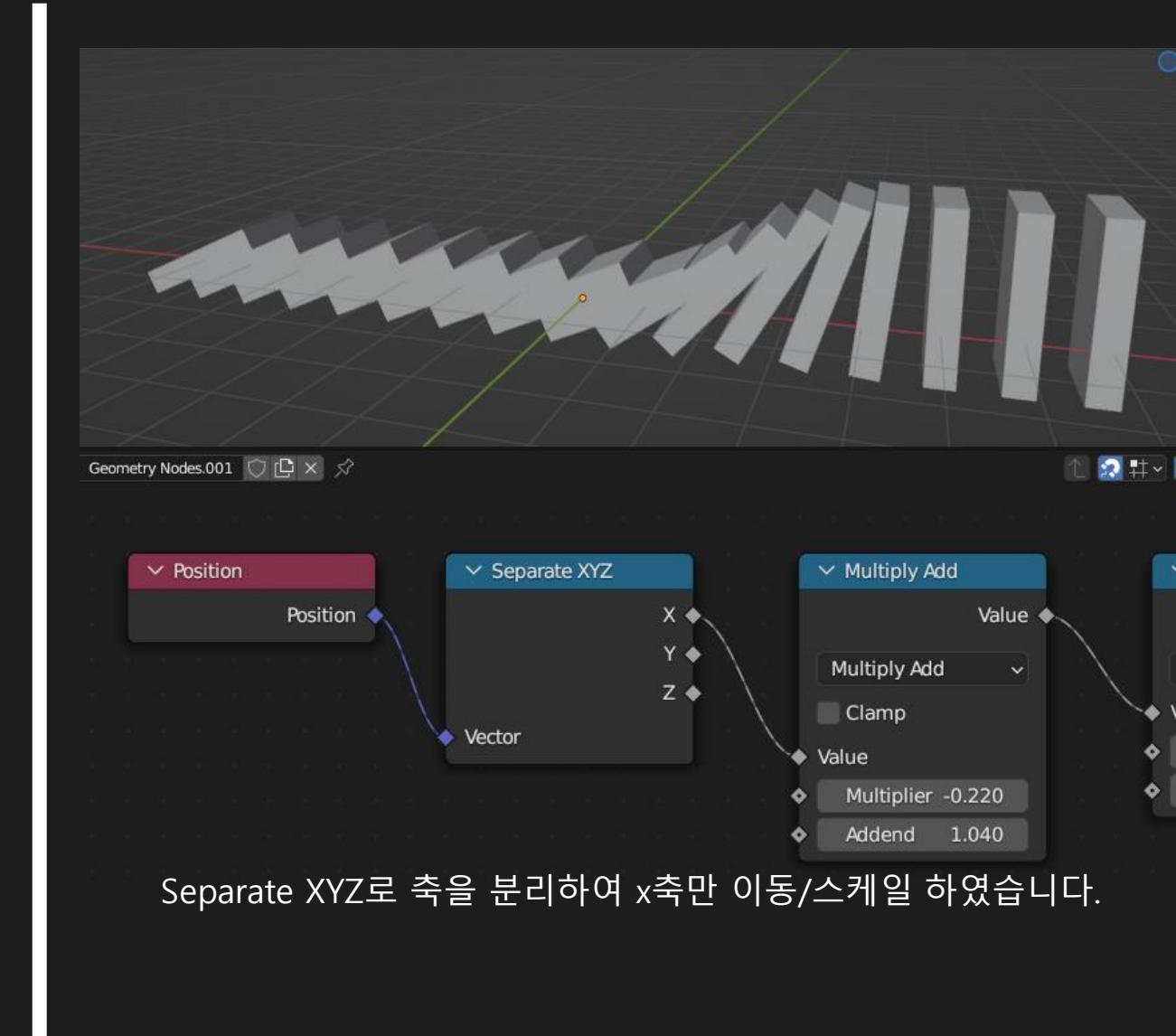
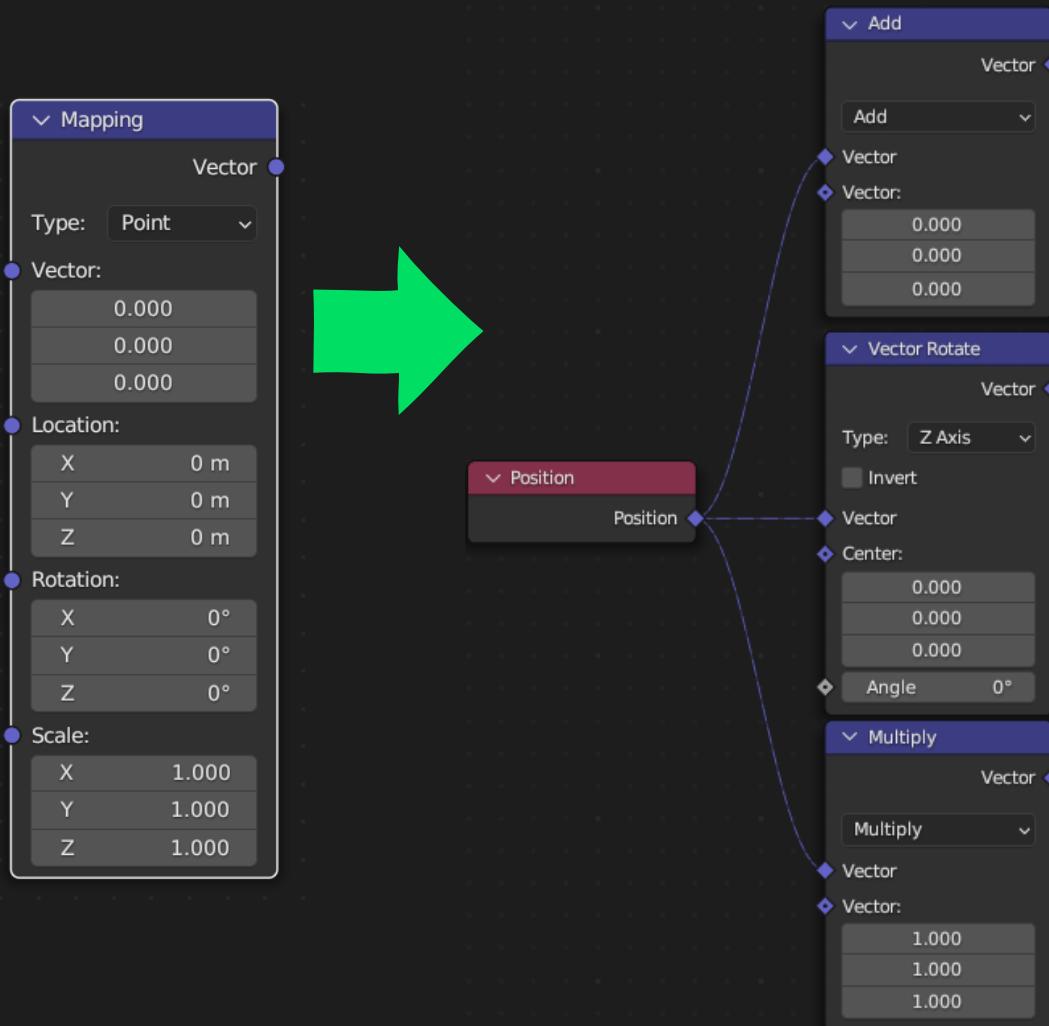
Mix, ColorRamp 등 많은 노드들도 제한된 범위에서 작동합니다.

하지만 지오메트리 노드의 데이터는 더 이상 0~1이라는 범위에
구속되지 않습니다.



벡터의 이동, 회전, 스케일

Position 노드도 셰이더에서와 같은 원리로 이동, 회전, 스케일을 할 수 있습니다.
다만 Mapping 노드는 없으므로, 다른 노드를 사용합니다.

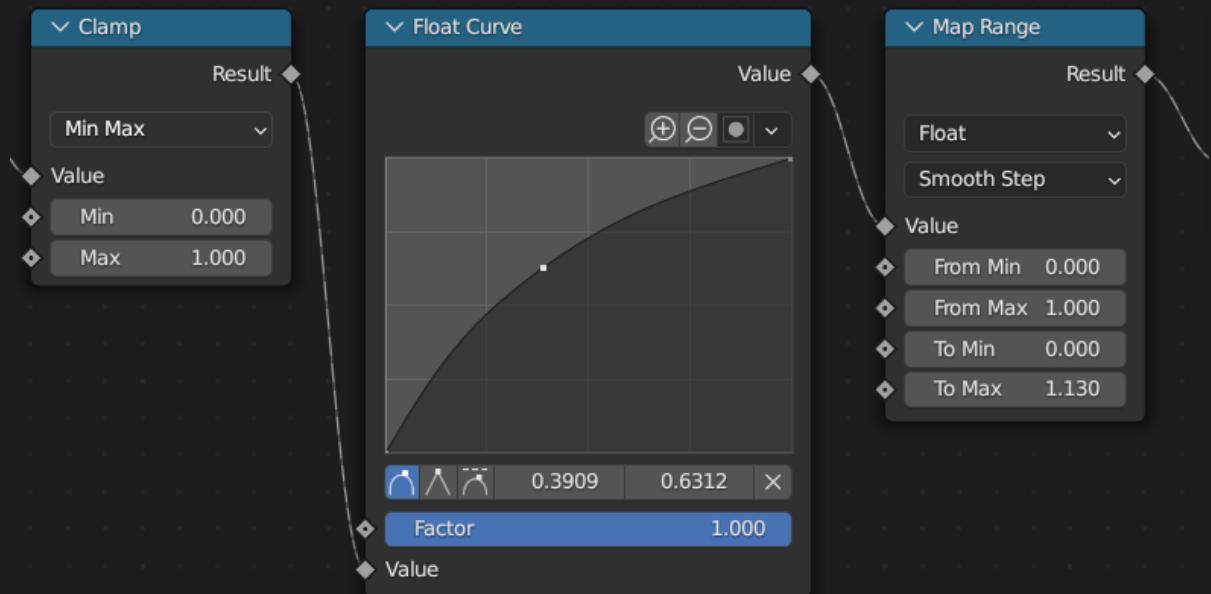
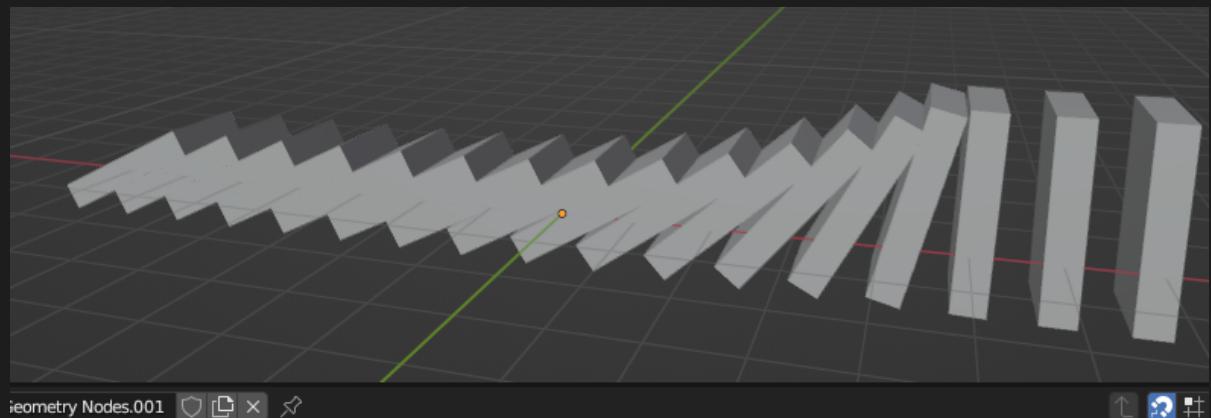


Separate XYZ로 축을 분리하여 x축만 이동/스케일 하였습니다.

노드 활용을 위한 범위 제한

0에서 1 사이에서만 작동하는 노드를 활용하기 위해,

1. 일단 범위를 0에서 1 사이로 제한하고,
2. 나중에 **Map range** 노드로 원하는 범위를 지정하는 식으로 사용할 수 있습니다.



Scene Time



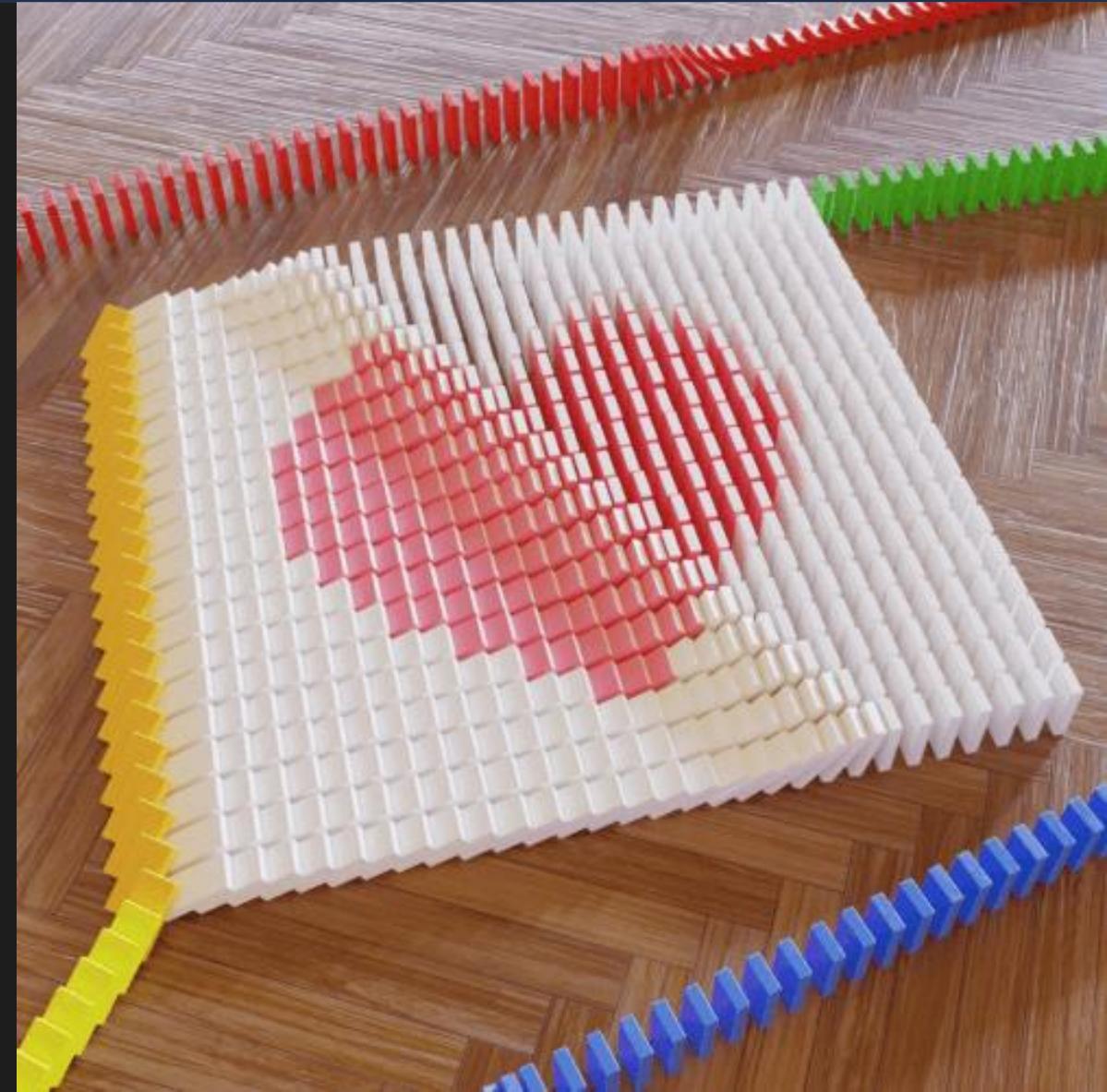
재생시 변하는 시간을 출력합니다.

말 그대로 Second는 초를, Frame은 현재 프레임을 출력합니다.

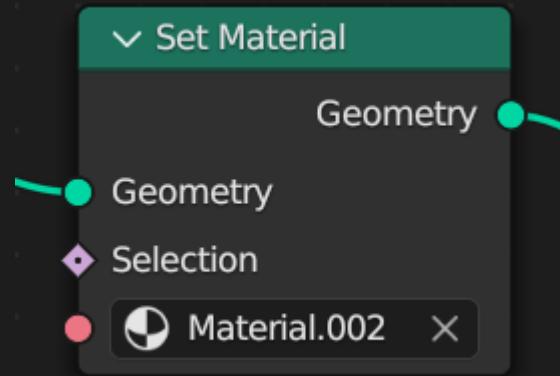
Frame을 사용할 경우 나중에 프레임레이트 변경 시 애니메이션 속도가 변할 수 있음에 유의하세요.

036강 Instance Animation (2)

인스턴스 텍스쳐링



지오메트리 노드와 Material



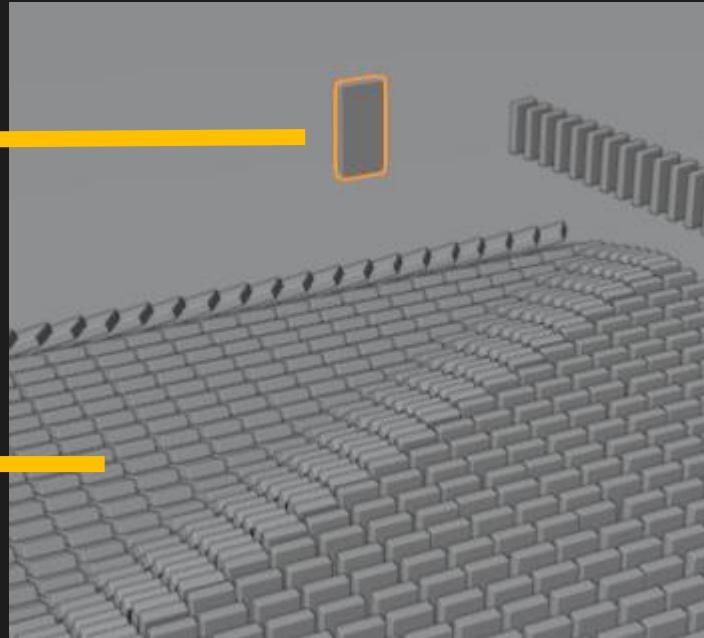
Set Material을 통해 재질을 바꿀 수 있습니다만,
인스턴스 사용시에는 조금 혼란스러울 수 있습니다.

그것은 인스턴스가 지오메트리 노드가 적용되는 오브젝트와 다르기 때문입니다.

인스턴스 원본



인스턴스 오브젝트를
사용하는 오브젝트



Set Material은 어디에, 어떻게 적용될까요?

그리고,

- 인스턴스를 Realize하면?
- 인스턴스 Material에서 인스턴스가 붙어있는 지오메트리의 정보를 받아오려면?

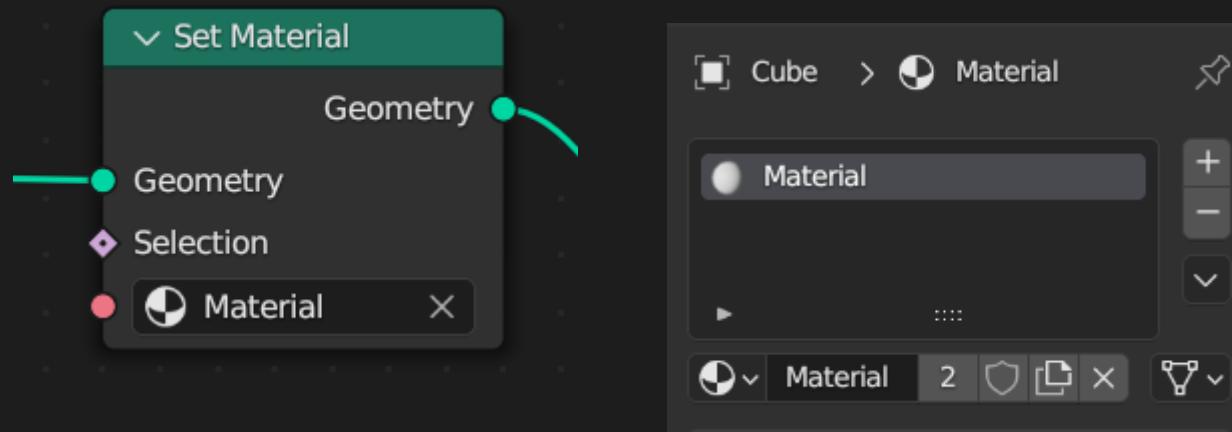
까지 생각하면 굉장히 복잡해집니다.

하나씩 차근차근 살펴봅시다

Set Material과 Material

지오메트리 노드와 Material은 별개입니다. Set Material은 지오메트리의 정보를 전달하지 않습니다.

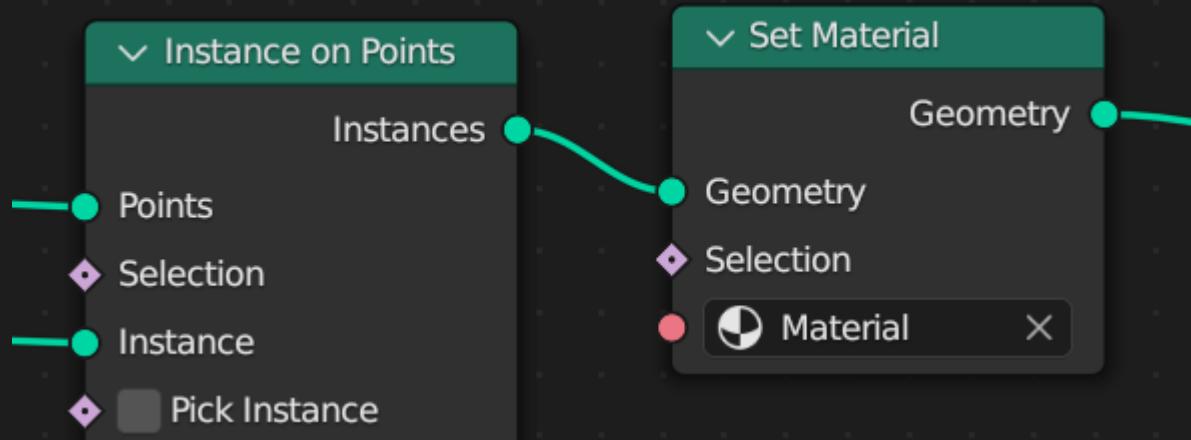
이 둘은 같은 기능을 합니다.



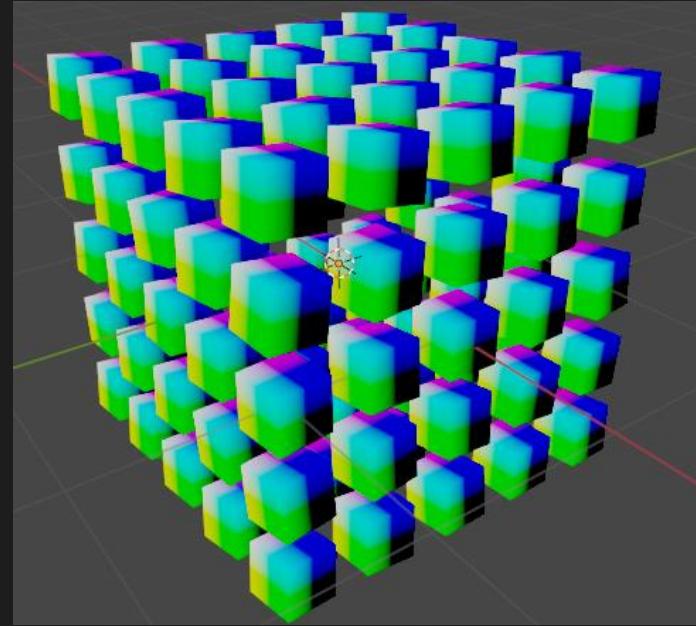
단, Material Properties는
노드 내부에서 생성한 지오메트리의
재질을 설정할 수 없다는 것만 다릅니다.

Instance와 Material

Set Material로 인스턴스의 재질을 바꿀 수 있지만, 지오메트리의 정보를 전달하진 않습니다.



Set Material로 Instance의 재질을 바꿀 수 있습니다.

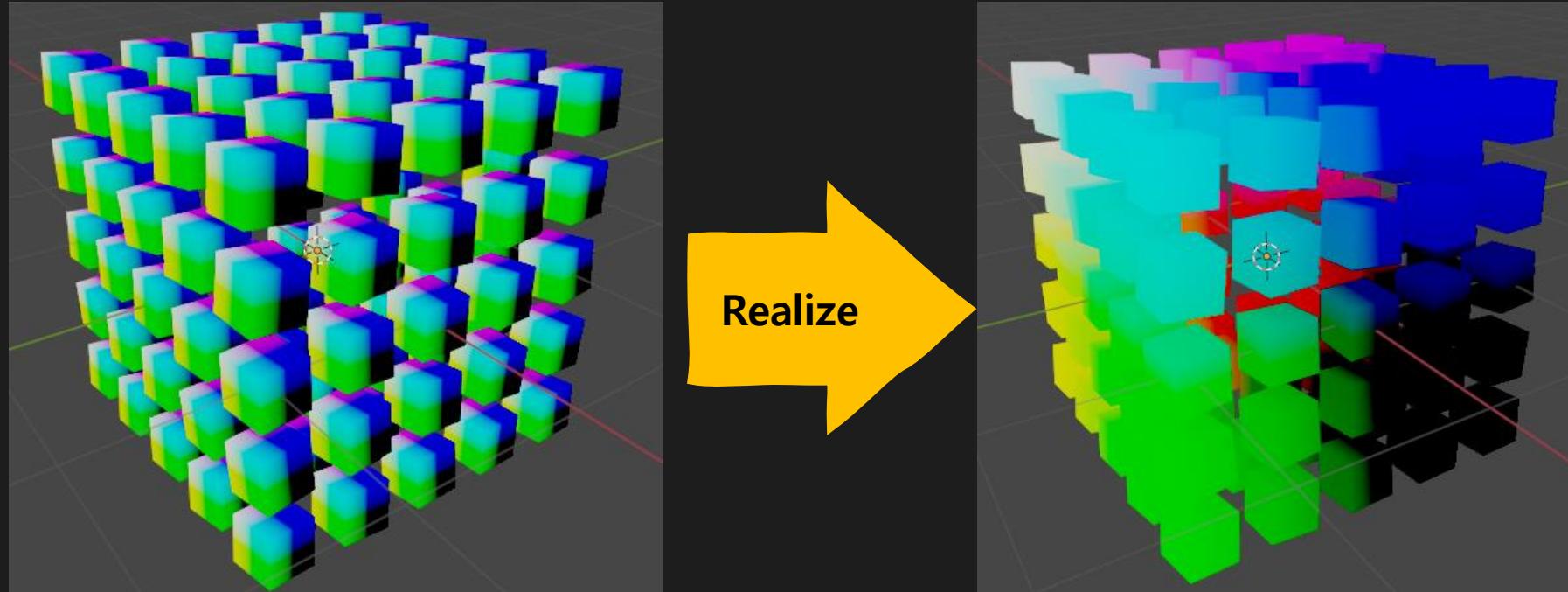


이 과정에서 인스턴스가 붙어 있는 오브젝트의 정보는 전달되지 않습니다.

Realize instance와 Material

인스턴스를 Realize 하면 인스턴스는 지오메트리 노드를 사용하는 오브젝트의 일부가 됩니다.

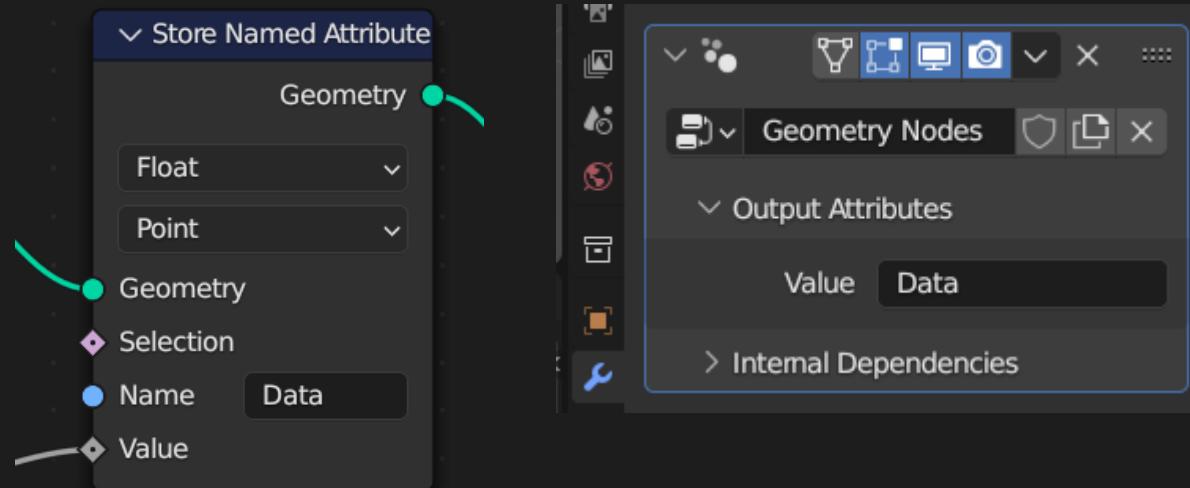
더 이상 인스턴스가 없으므로 Material은 인스턴스가 아니라 인스턴스가 붙어있었던 덩어리에 적용됩니다.



Geometry node 출력과 Material

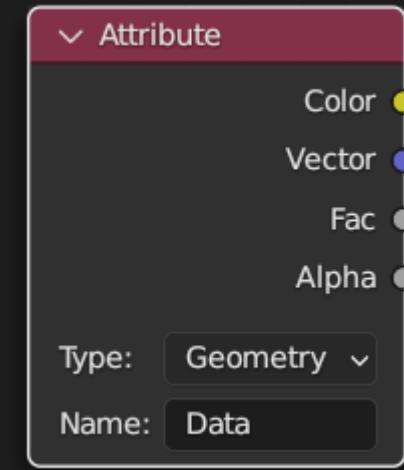
일반적으로는 지오메트리 노드의 데이터를 Material에서 불러올 수 있습니다.

Geometry Nodes



Store Named Attribute 혹은 Group Output에 연결하여 데이터 저장/출력

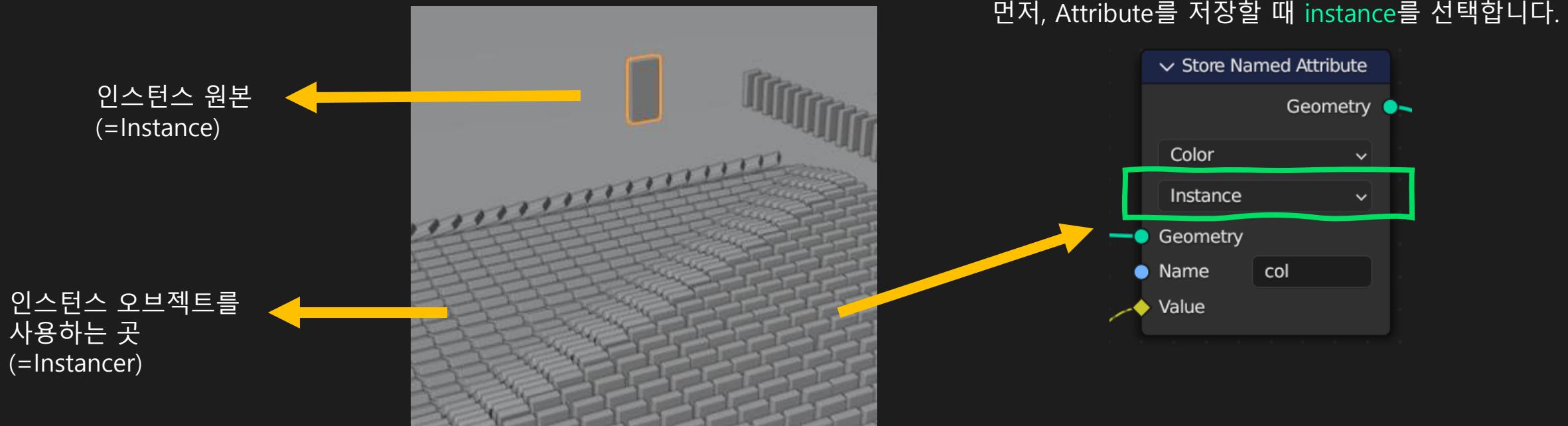
Shader Nodes



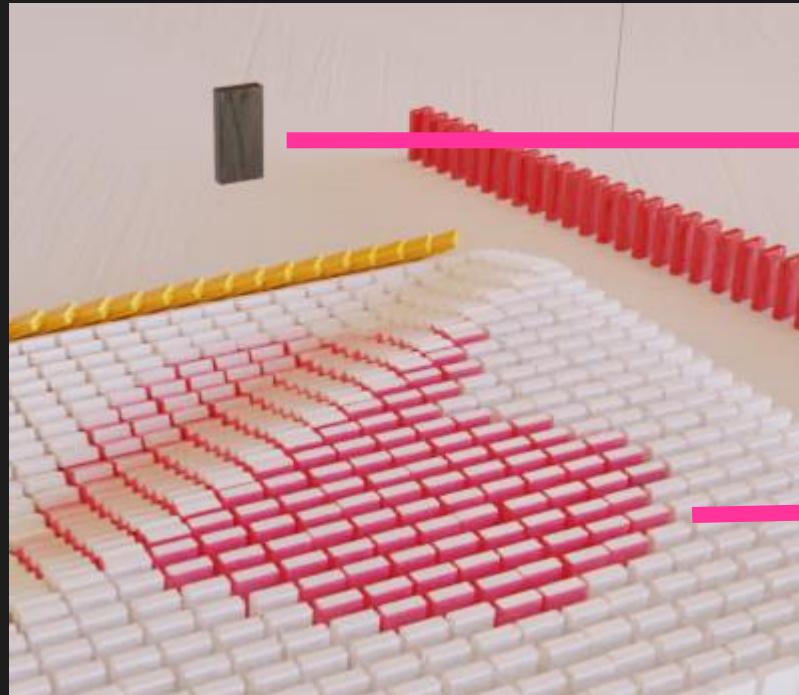
Shader에서 Attribute 노드로 받아옵니다.

Geometry node 출력, Instance, 그리고 Material

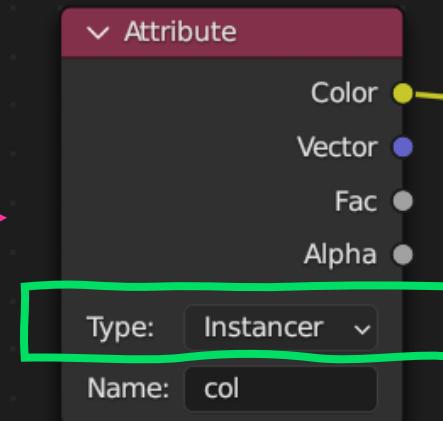
인스턴스를 사용하는 경우 'Instance' 와 'Instancer' 와 별개이기 때문에, 저장과 전송에 유의하여야 합니다.



Geometry node 출력, Instance, 그리고 Material

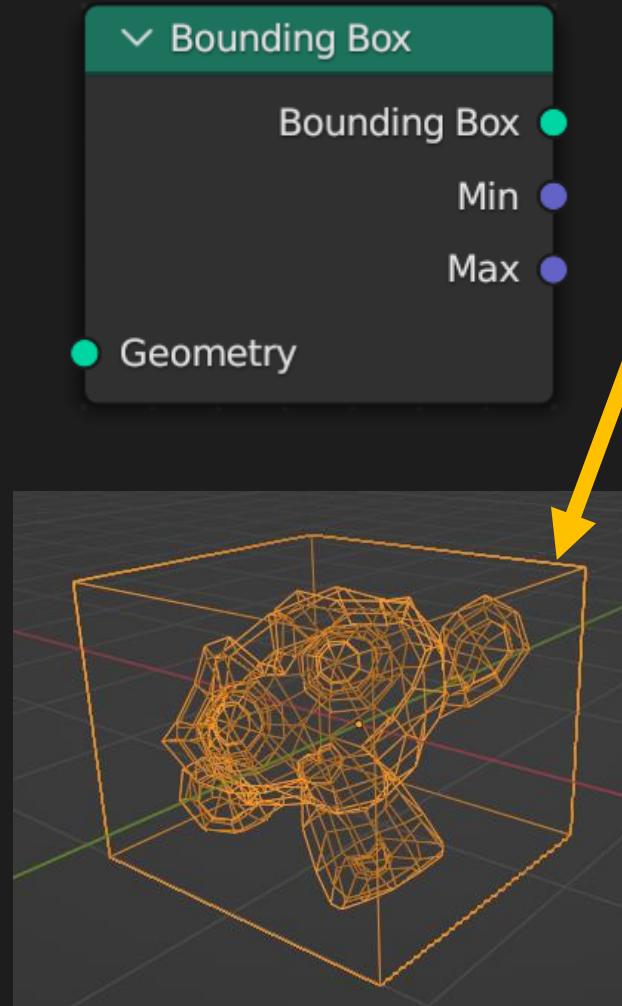


Instance의 Material은 Instancer 와 상관없이 작동하지만...



Attribute의 타입을 'Instancer'로 바꾸면
인스턴스가 붙어있는 오브젝트의 정보를
가져올 수 있습니다.

Appendix



Bounding Box : 지오메트리를 감싸는 박스와 그 최대/최소 좌표를 출력합니다.
Min, Max 는 Attribute Statistic의 Min/Max와 동일합니다만,
Bounding Box는 실제 바운딩박스 지오메트리도 출력합니다.

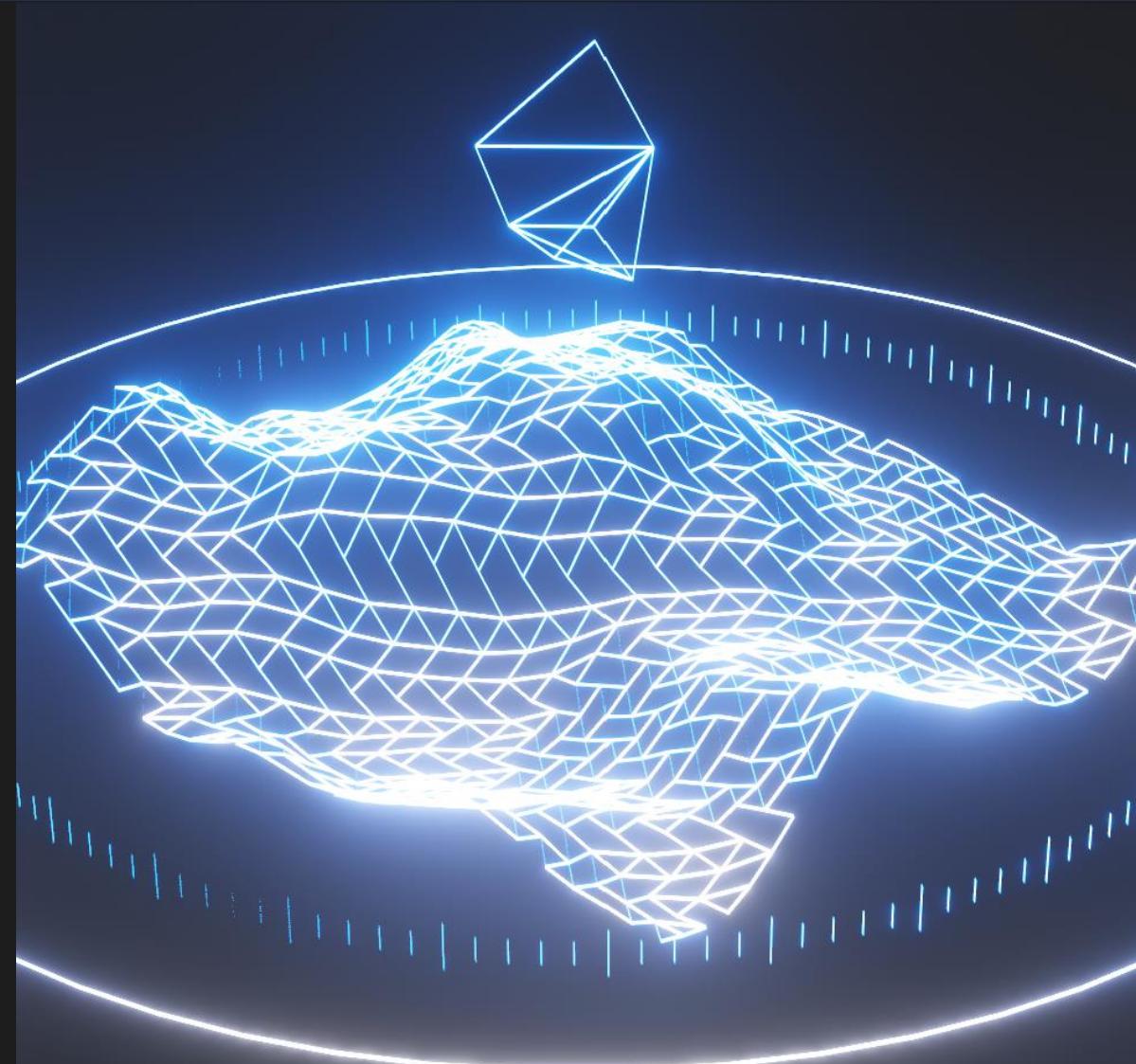
037강 메쉬의 변형 (1)

입력 노드들 :지오메트리의 정보

지오메트리 노드의 연결 방식 정리

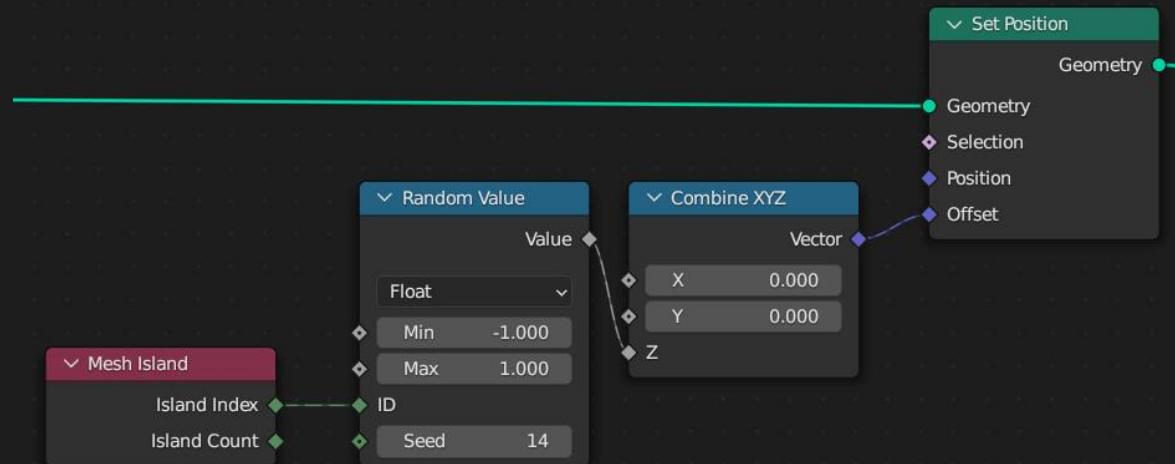
메쉬 변형 노드들

예제 : 홀로그램 HUD

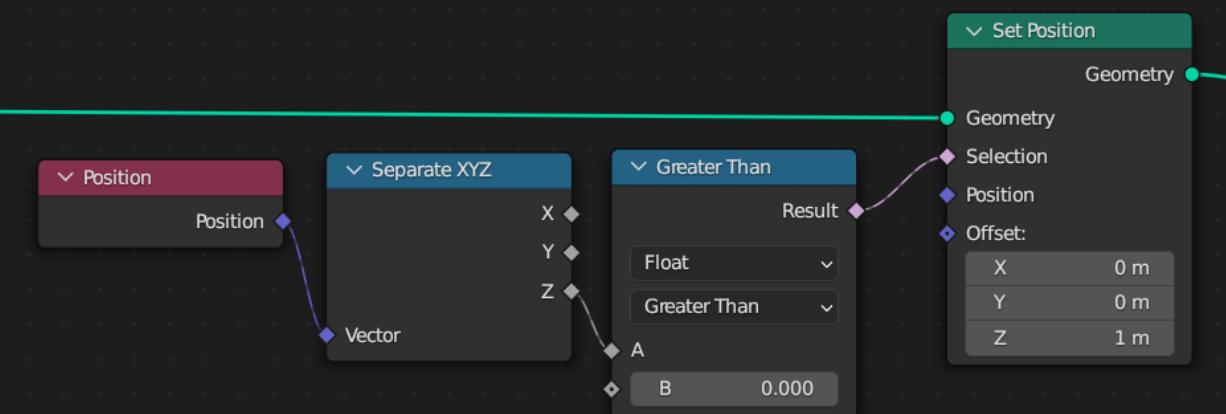


Reminder : 지오메트리 노드의 연결방식

1. Attribute – 컨트롤 – Attribute



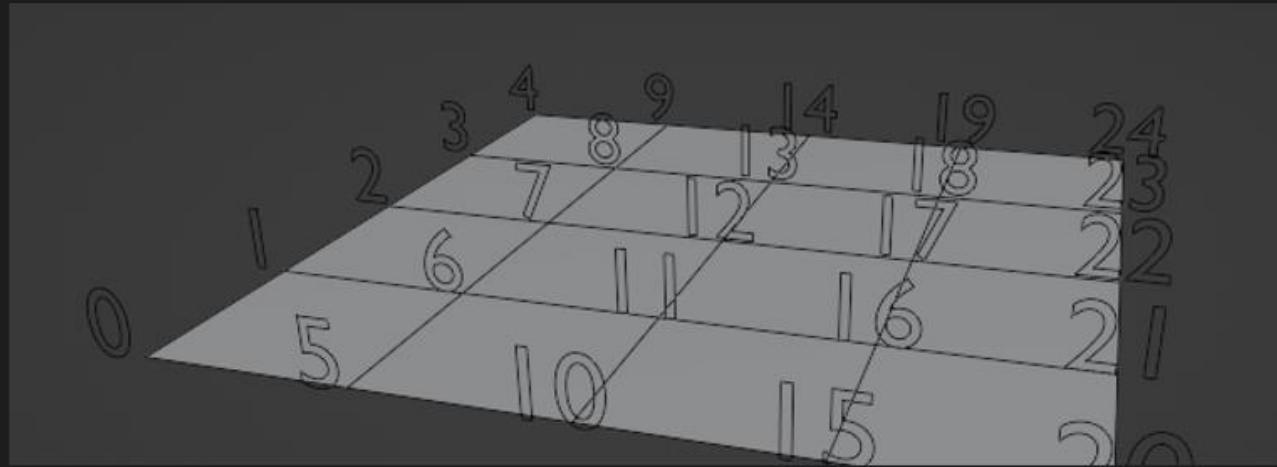
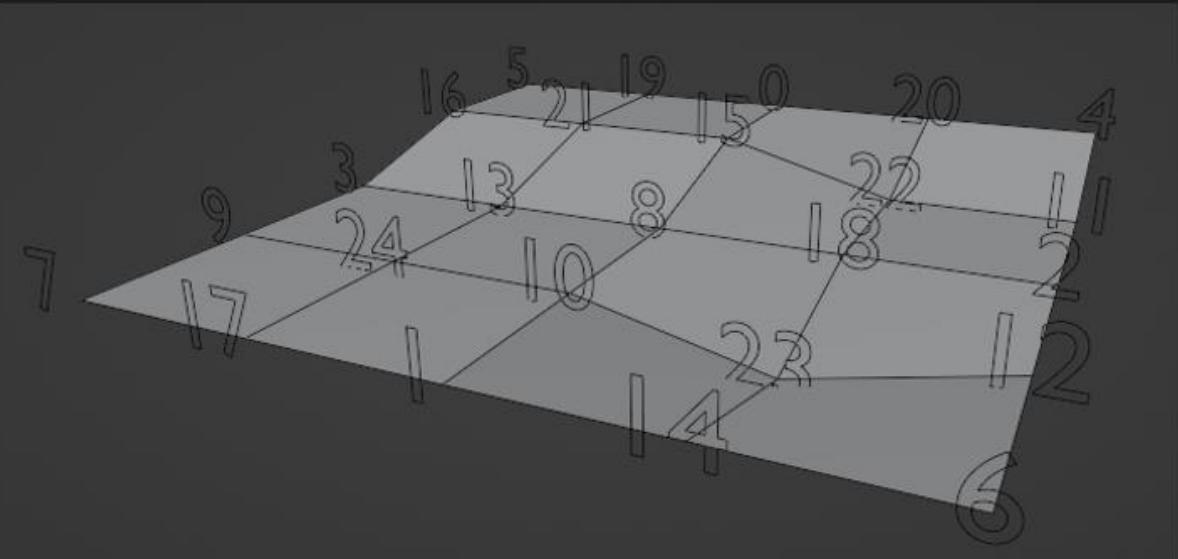
2. Attribute – Compare – Selection



Index

지오메트리의 각 성분은 고유번호를 가지고 있습니다.

모델링시에 이 Index들은 무작위로 섞이게 되지만, 노드를 통해 생성된 지오메트리는 일관된 순서의 Index를 갖습니다.

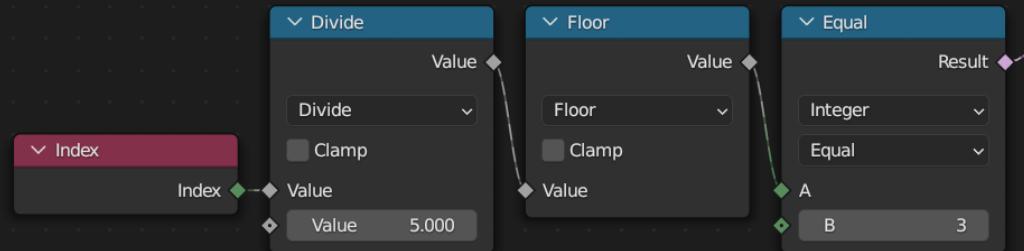


정수의 컨트롤

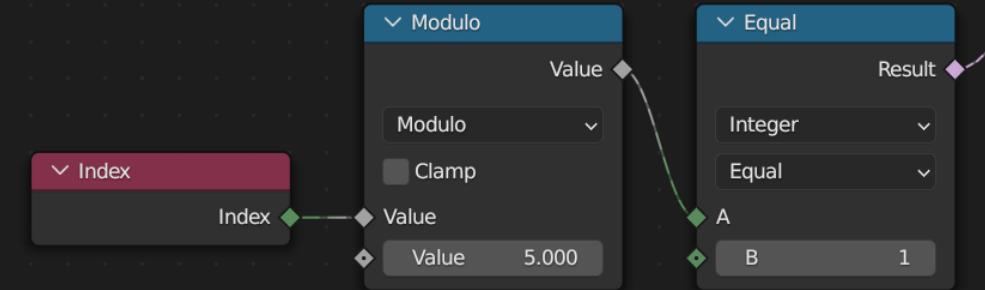
가장 흔히 쓰이는 컨트롤은 나눗셈을 활용하는 것입니다.

나머지

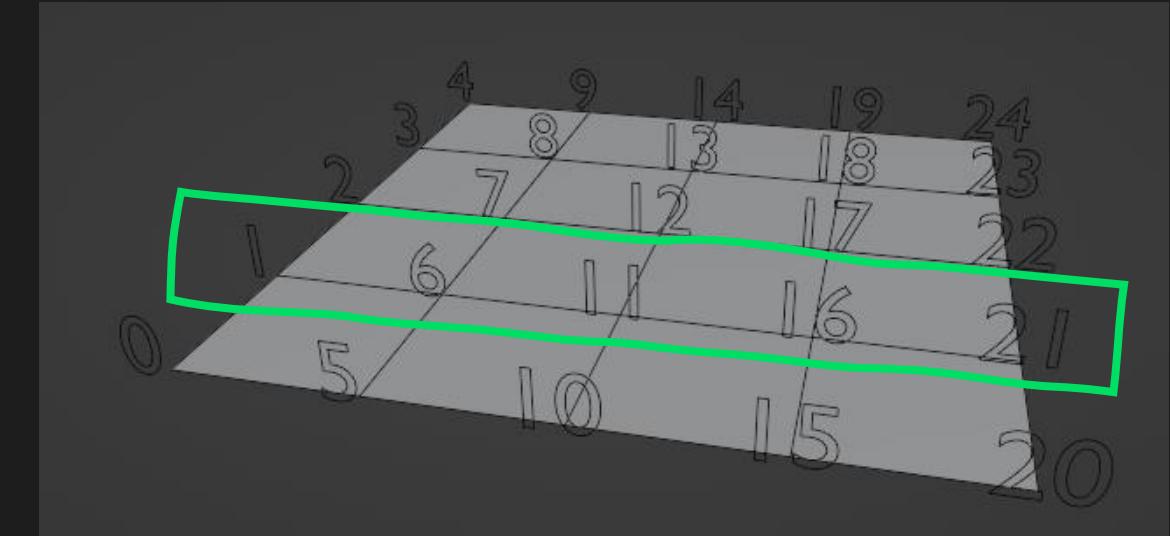
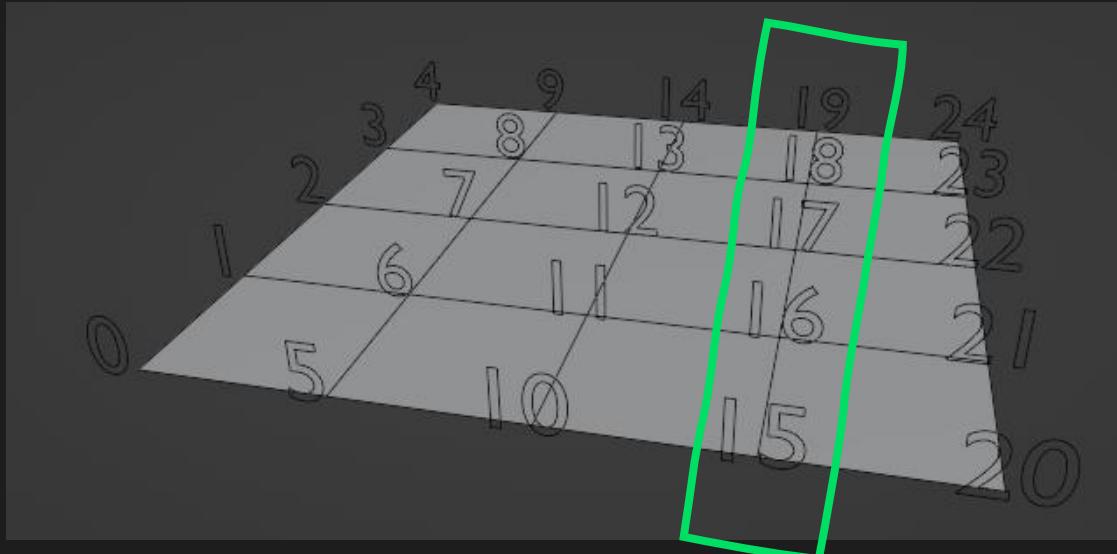
몫



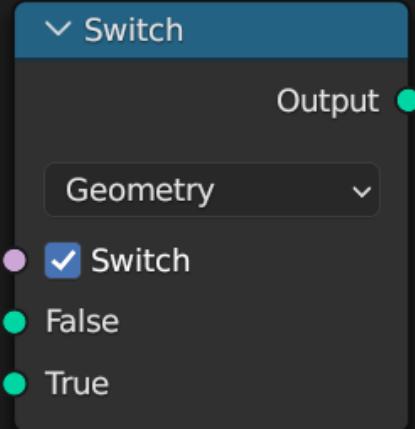
5로 나눈 '몫'이 3 : 15, 16, 17, 18, 19



5로 나눈 '나머지'가 1 : 1, 6, 11, 16, 21

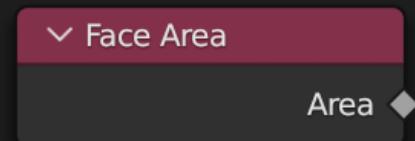


Switch



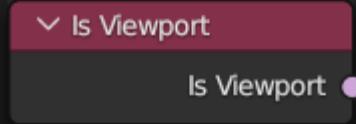
Switch를 통해 입력받은 두 값중 하나를 선택하여 출력할 수 있습니다.

Mix 노드와 비슷하지만, 참/거짓으로 둘 중 하나를 완전히 선택한다는 점,
그리고 더 많은 데이터타입을 다룰 수 있다는 점이 다릅니다.



Face Area : 각각의 면의 넓이를 출력합니다.

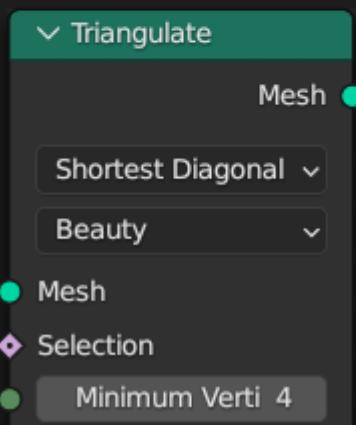
Appendix



Is Viewport : 뷰포트일 때 참, 렌더링시에 거짓을 출력합니다.
뷰포트와 렌더링할 때의 출력이 달라지므로, 렌더링할 때만 특정 작업을 하는 식으로 연결할 수 있습니다.



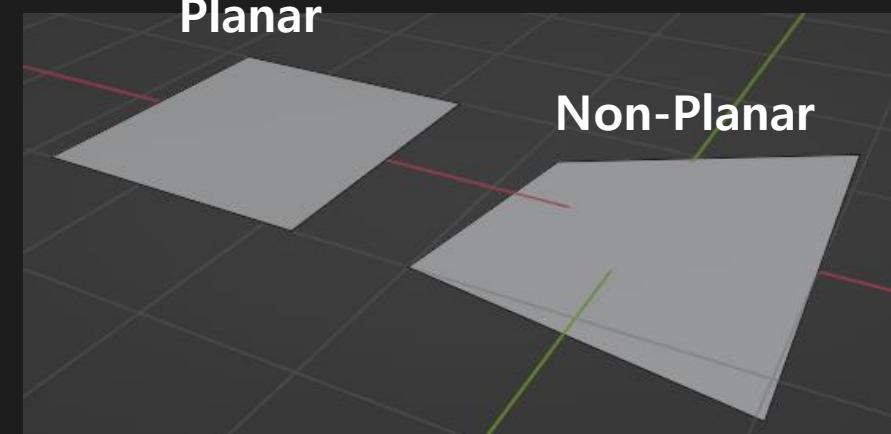
Face is Planar : 다각형이 평면을 이루는지 확인합니다.



Triangulate : 면을 삼각형으로 쪼갭니다.

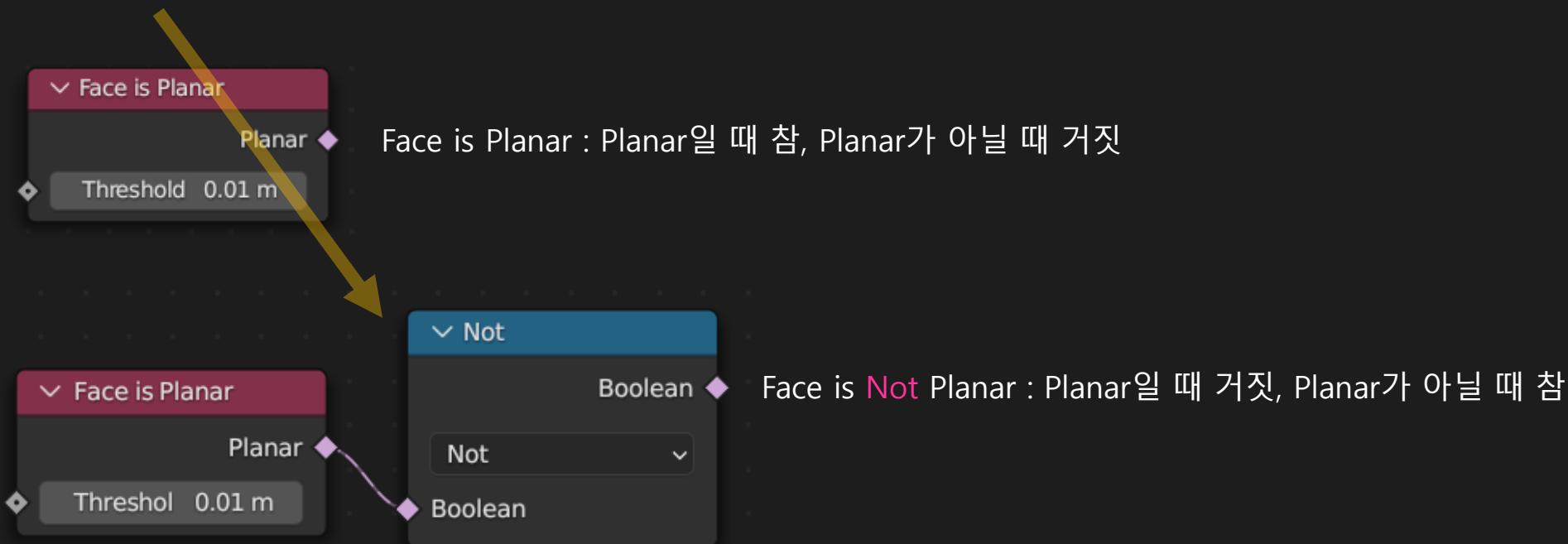
위쪽 선택창은 사각형을, 아래쪽 선택창은 다각형을
쪼개는 방법입니다.

Beauty를 사용하면 가능한 균일한 모양으로 잘라주지만,
더 느립니다.



Appendix

Boolean Math 노드로 불린 연산을 할 수 있습니다. 다음 시간에 자세하게 사용해볼 것입니다.



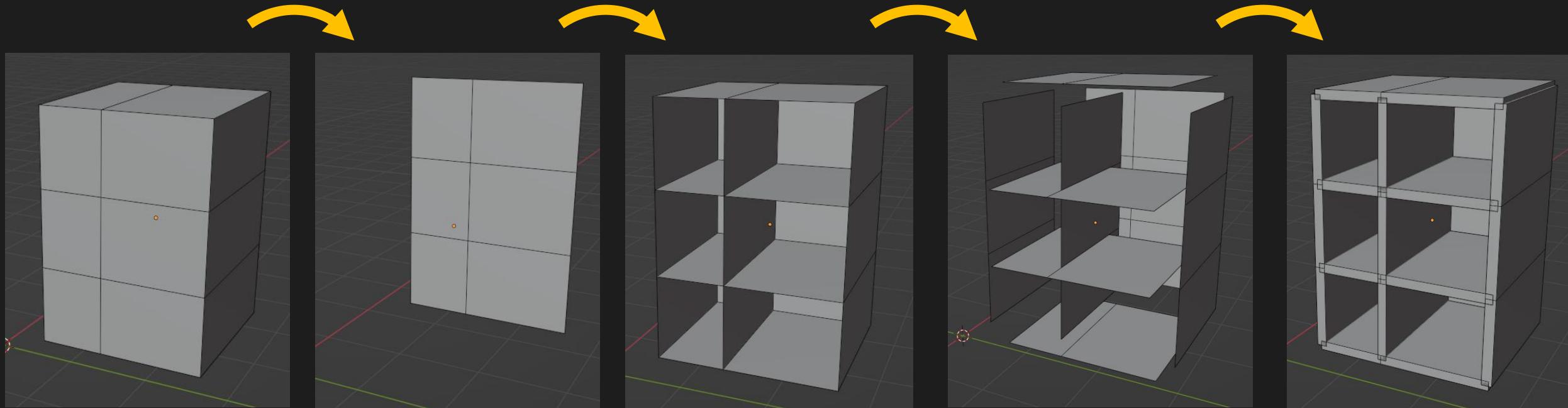
038강 메쉬의 변형 (2)

두 벡터의 비교

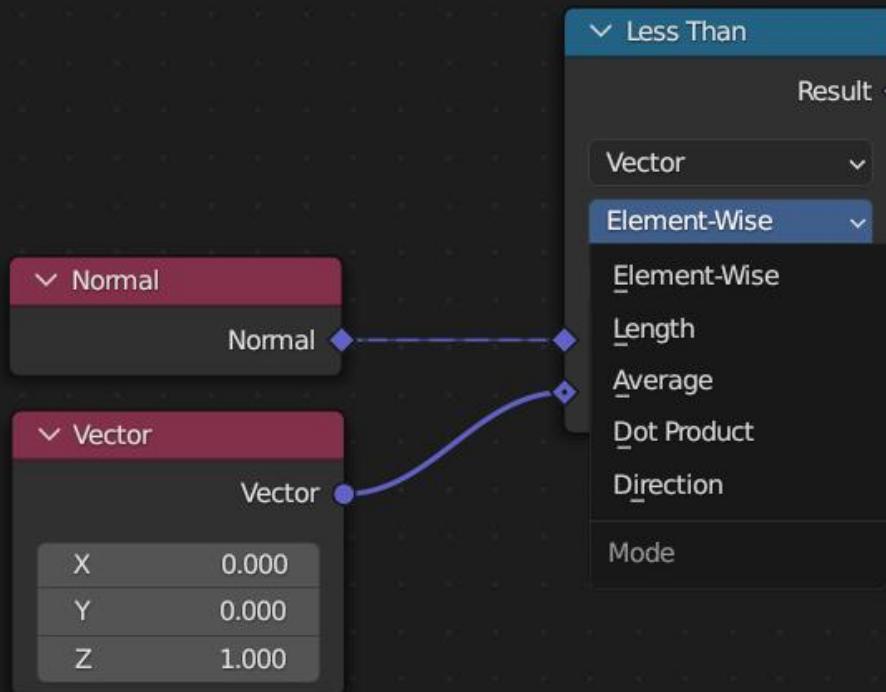
공간박스 메이커 만들기



기본 아이디어



벡터의 비교



Compare 노드는 벡터를 비교하는 여러가지 방식을 제공합니다.

Element-Wise : 입력값을 하나하나 비교합니다. 세 입력값이 모두 만족할때만 참입니다.

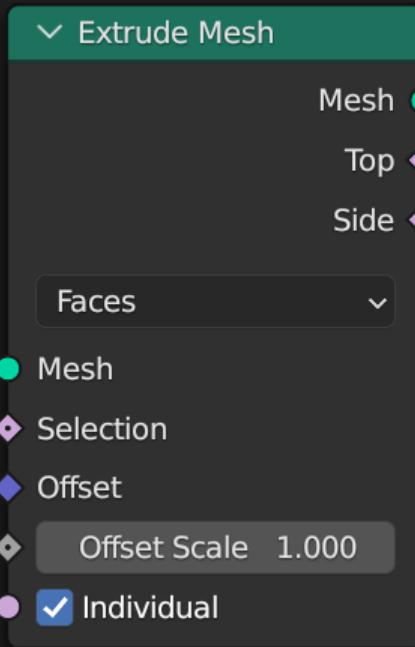
Length : 벡터의 길이 (원점으로부터의 거리) 를 비교합니다.

Average : x,y,z 각 성분을 평균내어 비교합니다.

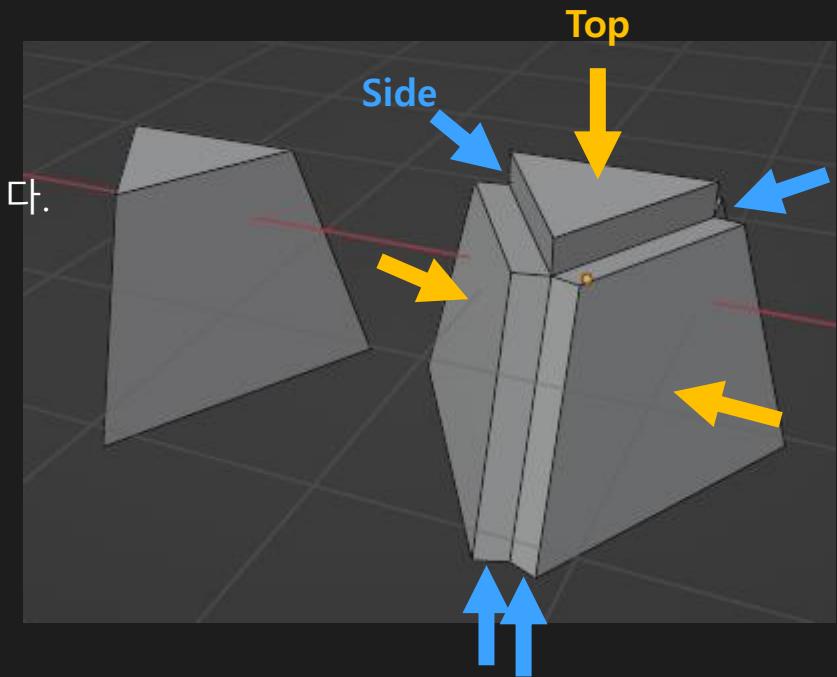
Dot Product : 두 벡터의 내적(dot product)을 계산하여 비교합니다.

Direction : 두 벡터가 이루는 각도를 바탕으로 비교합니다.

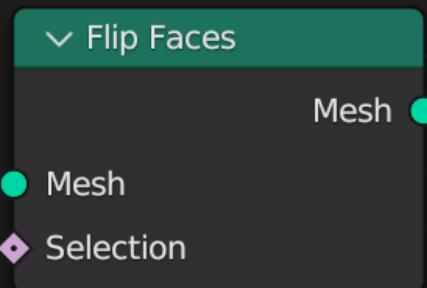
Extrude Mesh



- Offset은 Normal 방향이 기본값입니다.
(면이 없으면 원점으로부터 뻗어나가는 방향이 됩니다.)
- ※아직 Even Extrude 기능은 없습니다. (3.5 기준)
현재 버전에서 even extrude를 사용하고 싶다면 54강을 참고하세요.



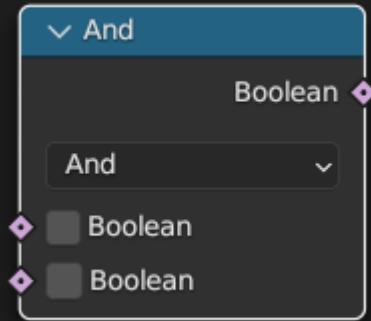
Flip Faces



면의 Normal을 뒤집어줍니다.

모델링에서의 Recalculate Normal (Shift+N, Ctrl+Shift+N) 과 비슷하지만
안팎 자동 계산 기능은 없고, 무조건 뒤집어줍니다.

Boolean Math



불린 연산을 해줍니다.

Operation
<u>Subtract</u>
<u>Imply</u>
<u>Not Equal</u>
<u>Equal</u>
<u>Nor</u>
<u>Not And</u>
<u>Not</u>
<u>Or</u>
<u>And</u>

Not And : And와 Not 연산의 결합. **둘 중 하나라도 거짓일때** 참입니다.

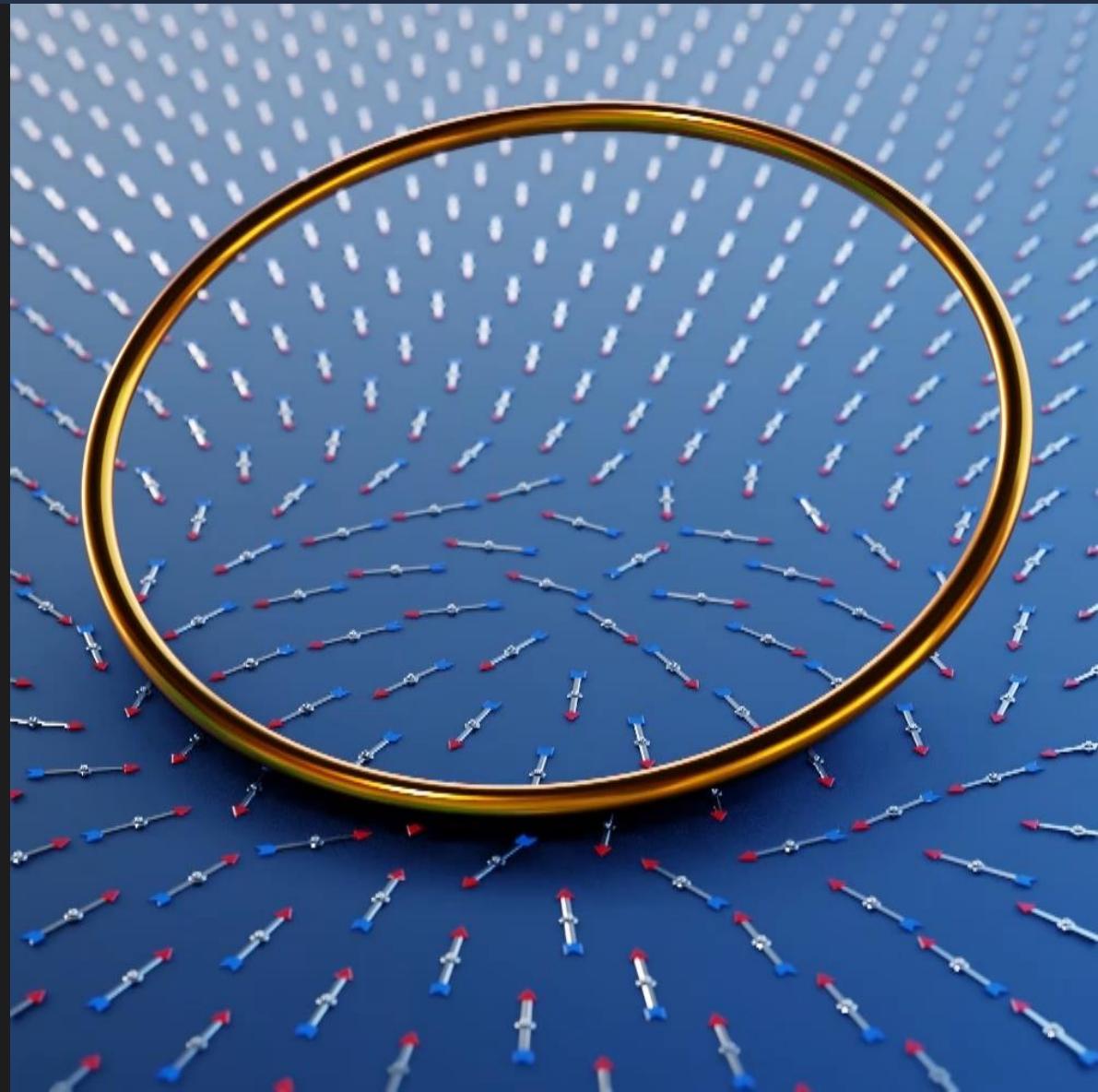
Nor : Or와 Not의 결합. **둘 다 거짓일때** 참입니다.

Imply : $A \rightarrow B$... 잘 쓰이지는 않습니다. (Not A) or B 와 같습니다.

Subtract : $A - B$. A가 참인 것 중 B가 참인 것을 제외합니다.

039강 Attribute의 전송(1)

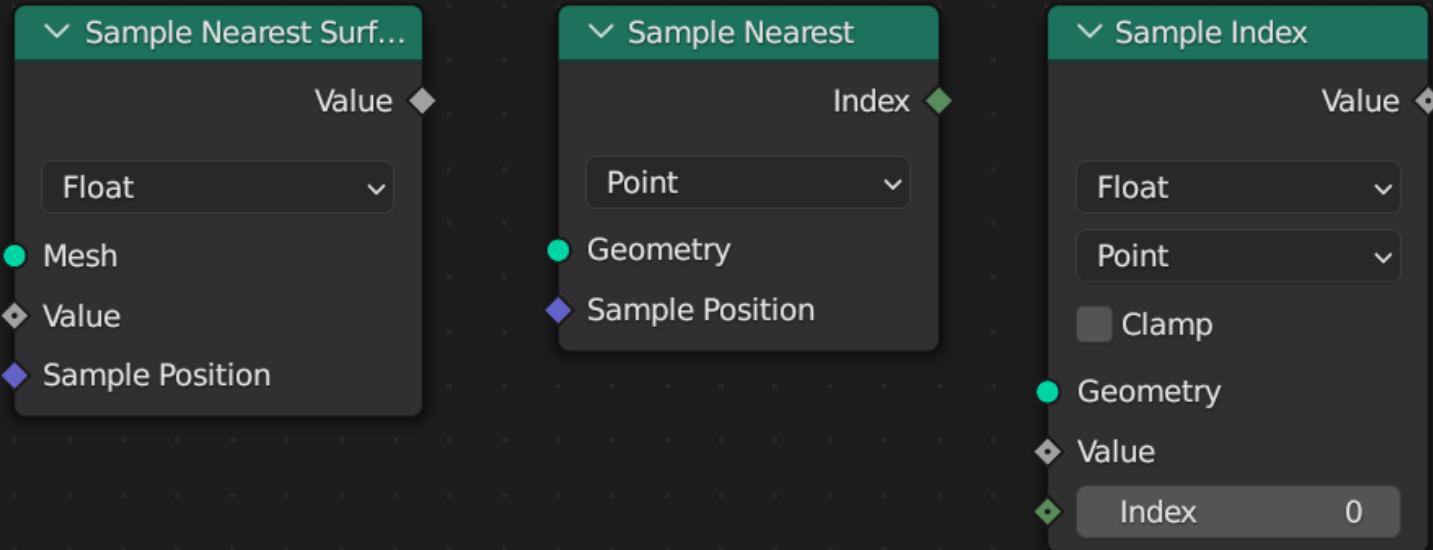
서로 다른 지오메트리 사이에 Attribute를 전달하는 법
Sample Nearest Surface Node
방향을 회전으로 바꾸기



Sample Attribute Series

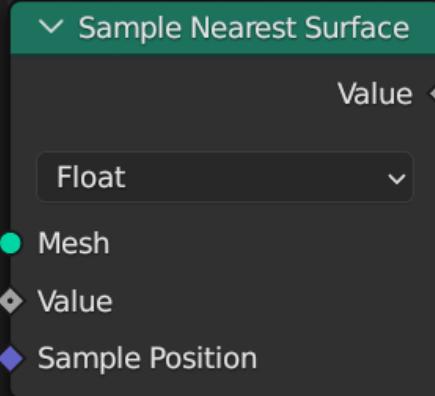
Attribute의 전송

만약 다른 지오메트리에 있는 Attribute를 불러오고 싶으면 어떻게 해야 할까요?



※이전 버전에서는 'Transfer Attribute'라고 불리던 노드들입니다.

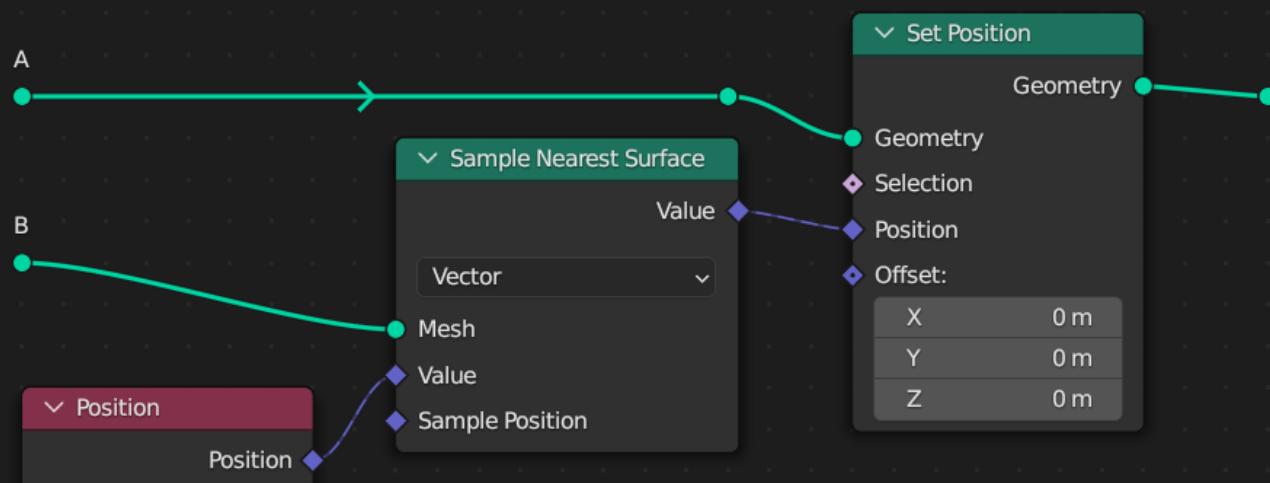
Sample Nearest Surface



Sample Nearest Surface는 타겟에 있는 **가장 가까운** 지점의 attribute를 가져옵니다.

Attribute는 표면의 위치를 바탕으로 자동으로 보간(interpolate)됩니다.

※ 연결예시. B의 위치정보를 A가 받습니다.



Sample Position

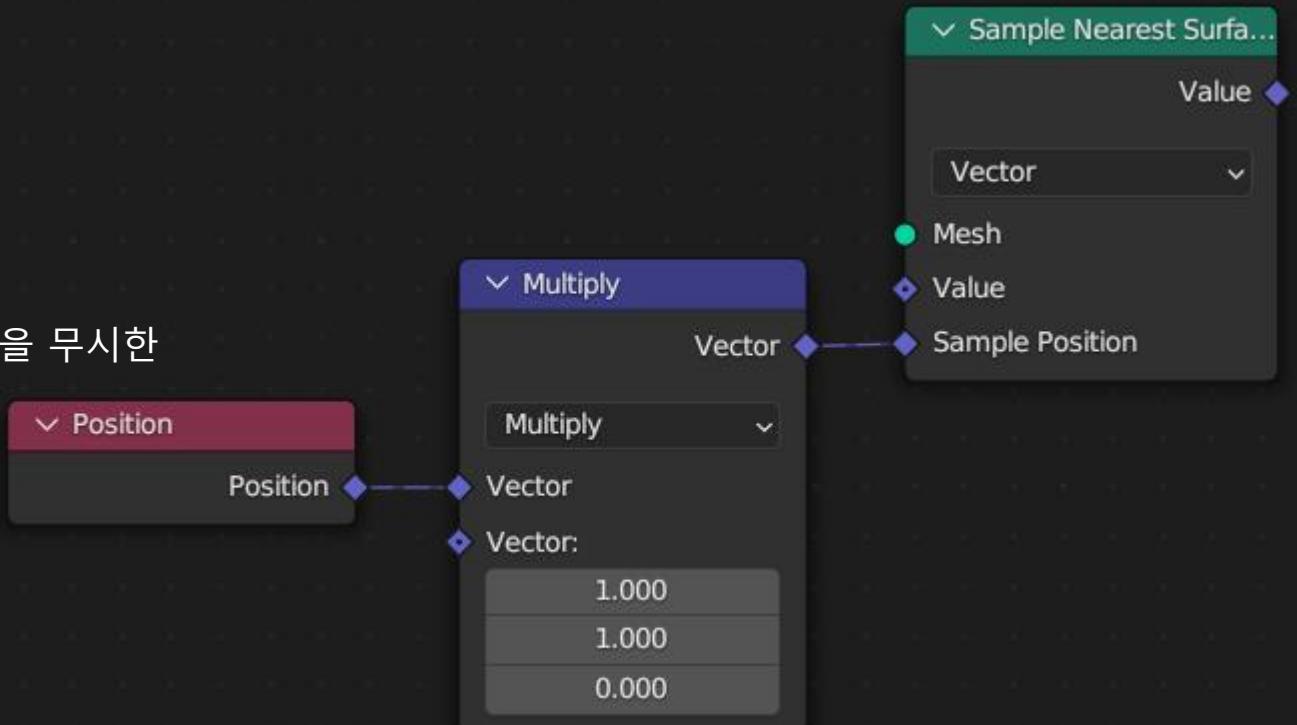
Sample Position은 '가장 가까운' 지점을 찾기 위한 기준위치를 변경합니다.

기본적으로 가장 가까운 위치는 실제 상대의 위치와 실제 나의 위치를 비교할 것입니다.

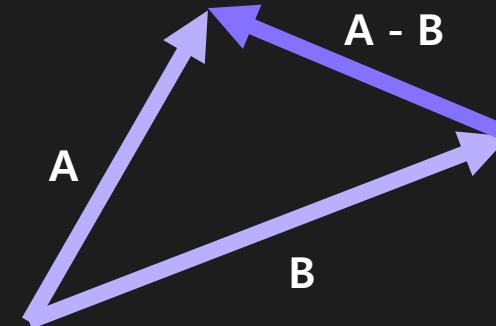
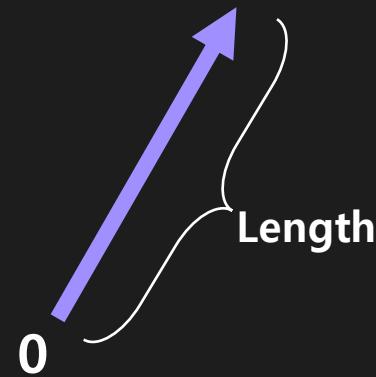
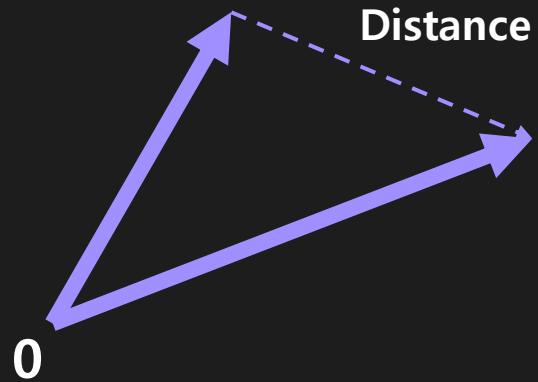
하지만 Sample Position에 다른 값을 입력하면,

자신의 실제 위치가 아니라 Sample Position에 입력한 위치를 바탕으로 가장 가까운 거리를 비교합니다.

예를 들어 우측처럼 연결하면, 자신의 위치가 z축을 무시한
xy평면에 납작하게 붙어있는 것으로 바뀝니다.

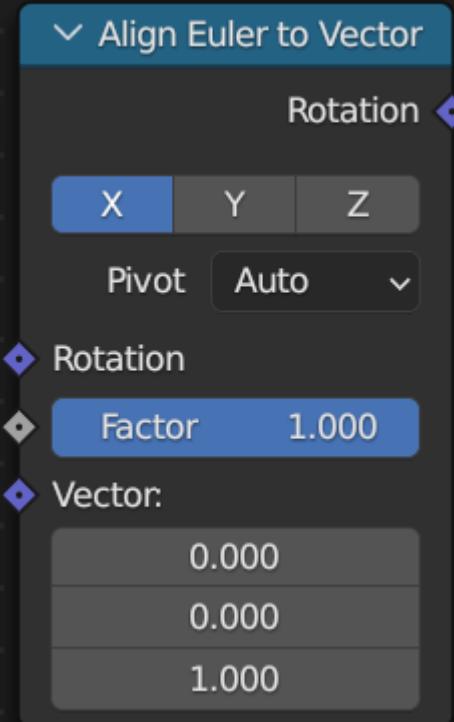


Reminder : 벡터의 길이



A와 B 사이의 Distance는 A-B의 Length와 같습니다.

Align Euler to Vector



방향과 회전값은 둘다 3차원의 정보이지만 의미하는 바가 다릅니다.

Align Euler to Vector는 입력받은 **방향**을 향하도록 하는 **회전값**을 출력합니다.

원하는 방향을 Vector 입력에 꽂으면 됩니다.

X,Y,Z : 방향을 따라갈 축을 선택합니다.

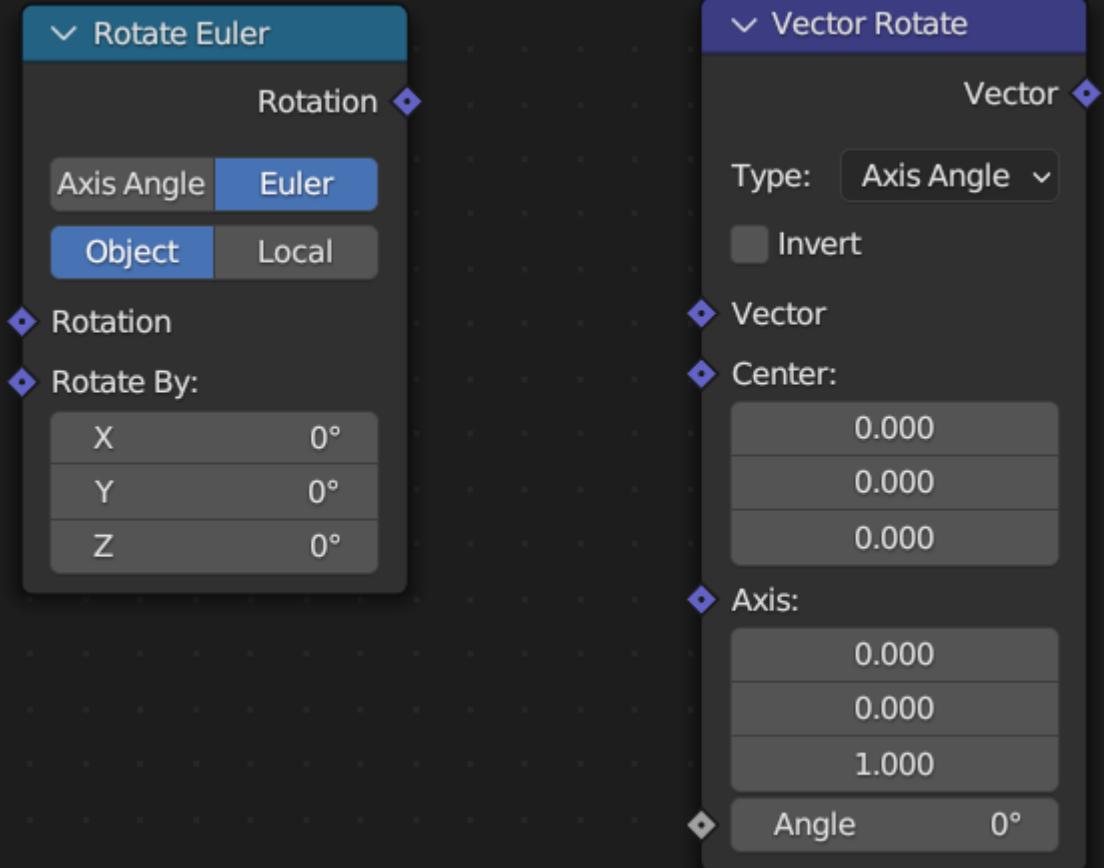
Pivot : 특정 축으로만 돌아가도록 회전을 제한합니다. Auto는 회전을 제한하지 않습니다.

Rotation : 한번 돌린 것을 다시한번 Align Euler to Vector로 돌리려고 한다면 이쪽에 연결합니다.

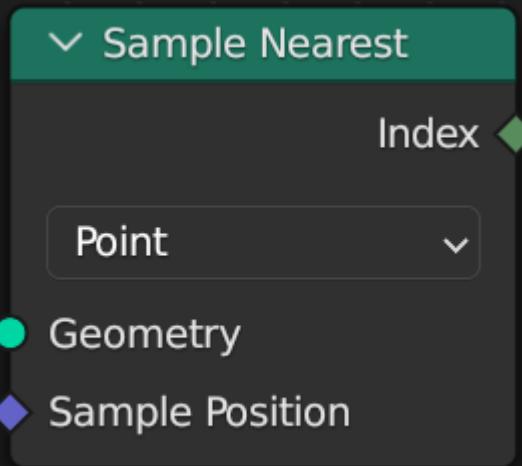
Rotate Euler / Vector Rotate

Rotate Euler는 회전을 받아서 그것을 [한번 더 돌릴 때](#) 사용합니다.
예를 들어 Align Euler to Vector에서 나온 회전값을 받아서
추가적으로 회전시킬 수 있습니다.
출력값은 Rotation입니다.

Vector Rotate는 [벡터를 회전시킬 때](#) 사용합니다.
예를 들어 Position을 Vector Rotate로 회전시킬 수 있습니다.
입력받은 벡터를 회전시키고, 그 회전한 벡터를 출력합니다.



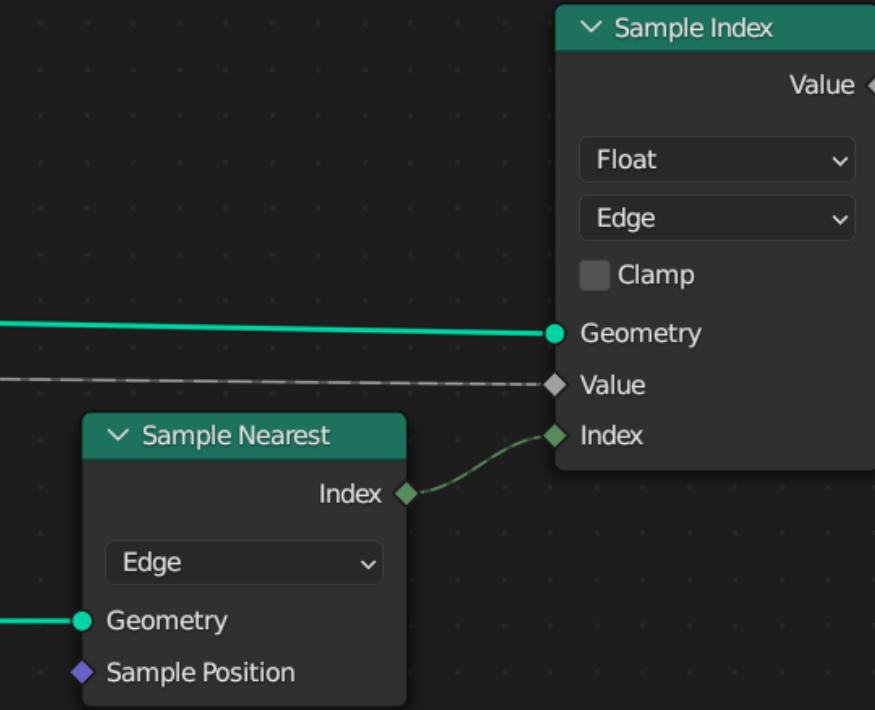
Sample Nearest



Sample Nearest Surface 와 비슷하지만, 가까운 지점의 attribute를 가져오는 게 아니라, 가까운 점, 선, 면의 index를 가져옵니다.

출력값은 타겟의 인덱스이므로 곧바로 사용할 수 없고 Sample Index와 같이 사용합니다.

Sample Nearest – Sample Index



Sample Index는 타겟의 attribute를 입력받은 인덱스에 따라 내보냅니다.

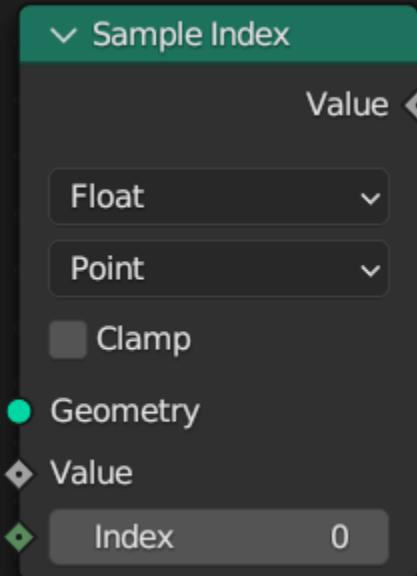
입력값의 주체가 혼란스러울 수 있습니다. 자세한 사항은 다음 시간에 다룹니다.

040강 Attribute의 전송(2)

서로 다른 지오메트리 사이에 Attribute를 전달하는 법
Sample Nearest & Sample Index



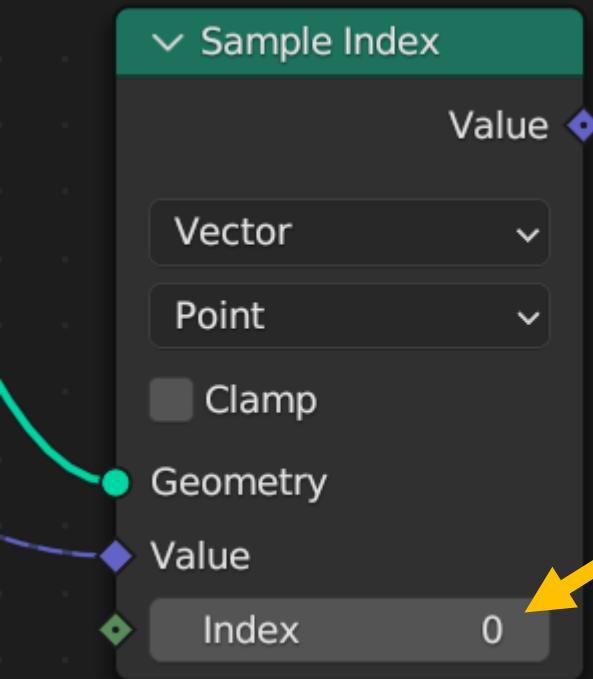
Sample Index



다른 지오메트리의 attribute를, 아래에 입력한 index를 바탕으로 가져옵니다.

구체적으로 어떤 규칙으로 가져오는지 살펴봅시다.

Sample Index 0

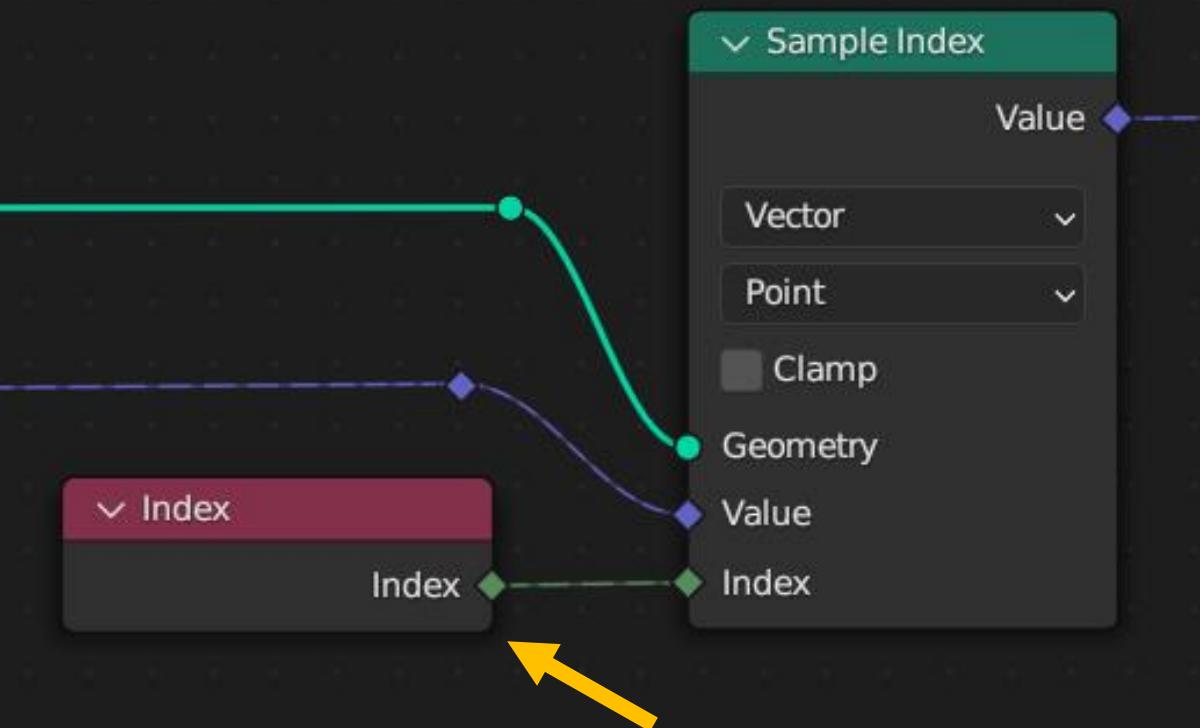


기본값인 0으로 두면, 자신의 모든 점에서 상대방의 지오메트리에 있는 [index 0인 점의 attribute](#)를 가져옵니다.

만약 상대방 지오메트리의 어디에서 읽어도 같은 값인 정보를 가져오려 한다면,
(예를 들어, 위치의 평균값, 커브의 전체 길이...)

어느 점의 정보를 가져와도 되기 때문에, 그냥 0번 점의 정보를 가져올 수 있습니다.

Sample Index – index

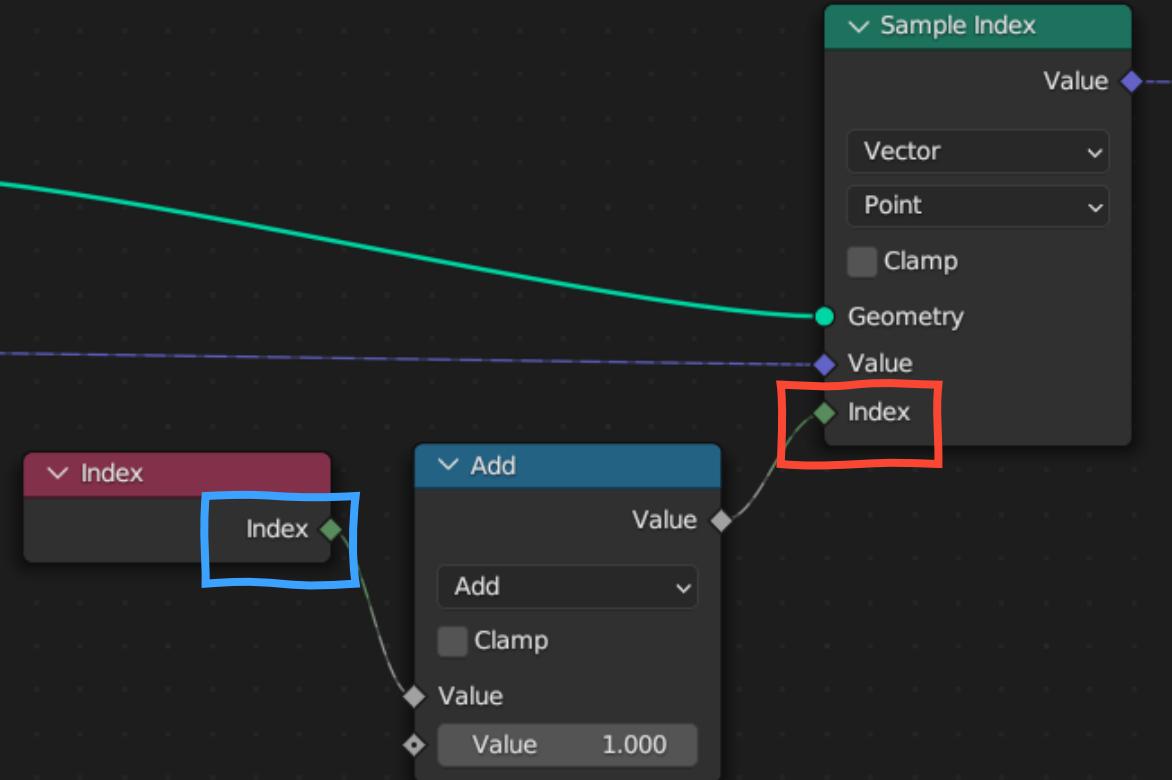


서로 다른 지오메트리의 정보를 같은 index끼리 대응시킵니다.

예를 들어 상대의 3번 점의 attribute를 나 자신의 3번 점에 전달하고,

상대의 5번 점의 attribute를 나 자신의 5번 점에 전달합니다.

Sample Index – $f(index)$



인덱스에 어떤 연산을 해서 꽂으면 어떻게 될까요?

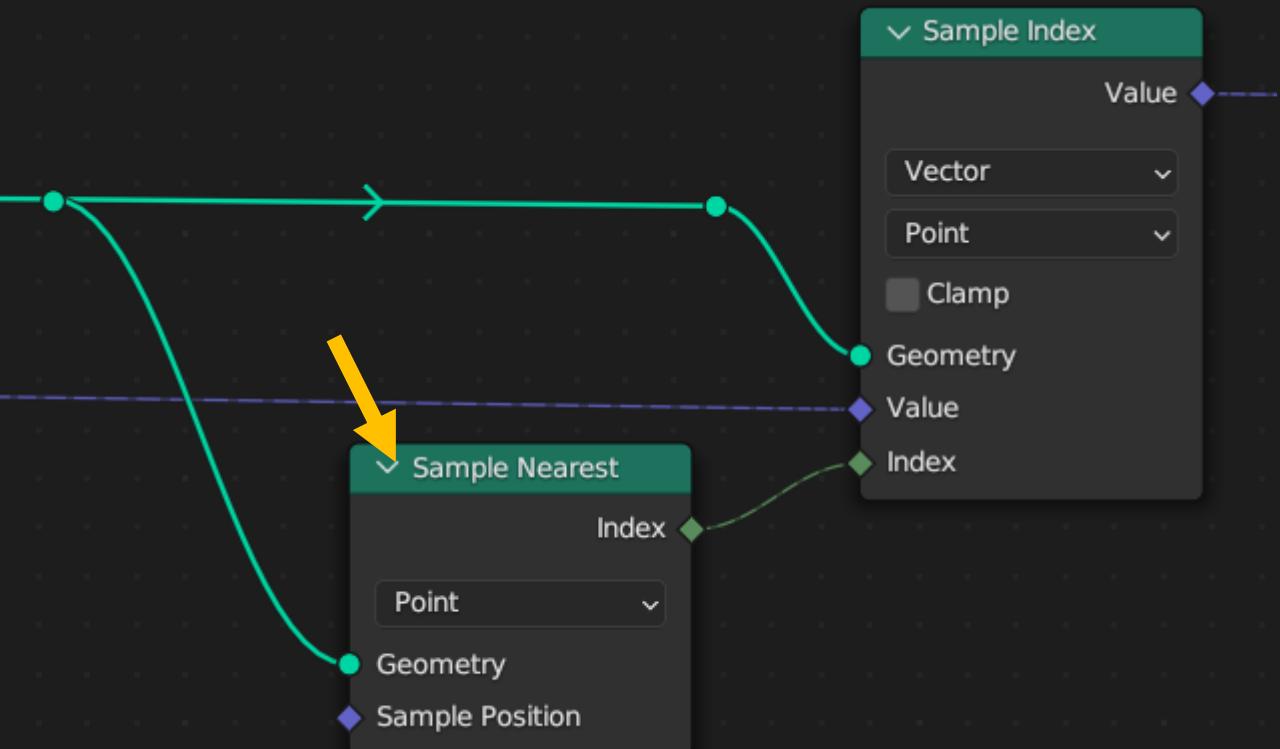
왼쪽의 경우 index에 +1을 해 보았습니다.

이 경우, 상대의 지오메트리의 인덱스가 하나씩 밀려나서 불러옵니다.

상대 지오메트리의 1번 점의 정보가 나의 0번 점에,
2번 점의 정보가 나의 1번 점에 대응됩니다.

“내 번호보다 10이 큰 번호의 정보를 주.”

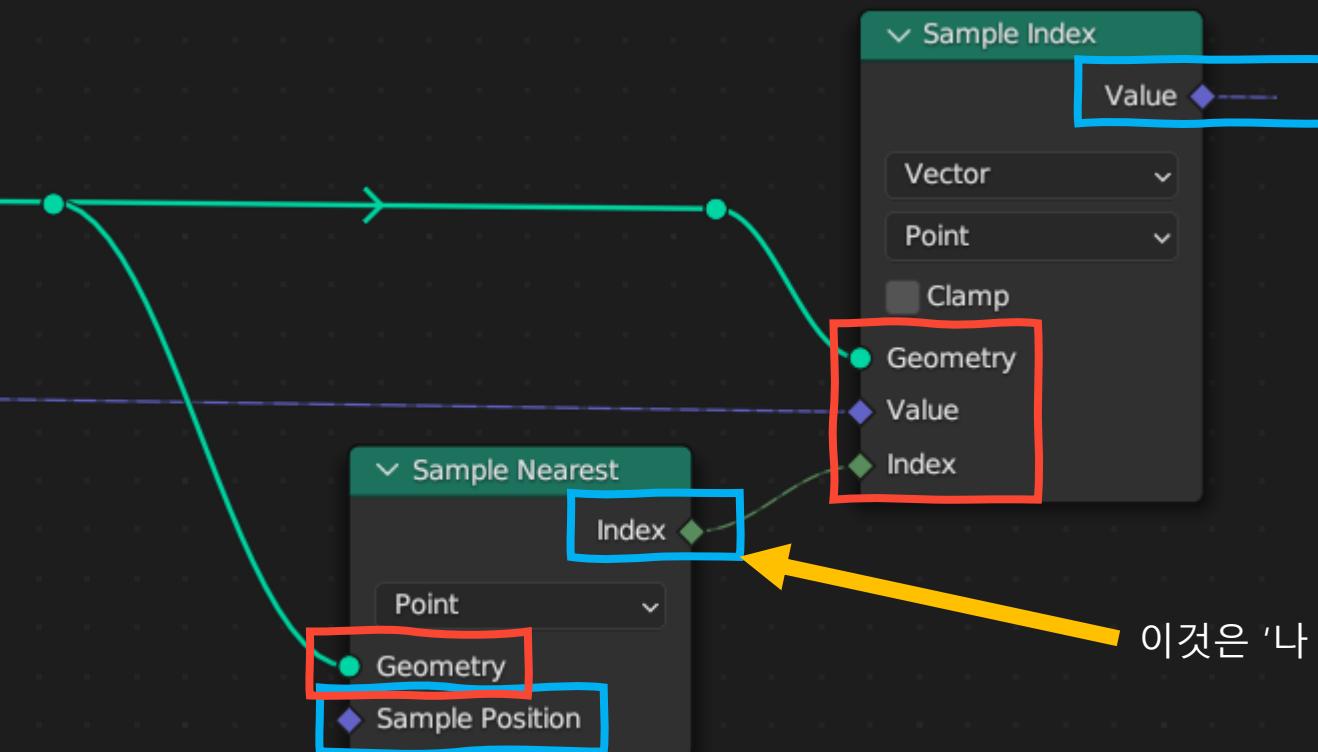
Sample Index – Sample Nearest



Sample Nearest에서 얻은 인덱스 정보를 Sample Index의 Index 입력으로 사용합니다.

이 연결이 정석 사용입니다.

Sample Index – Sample Nearest

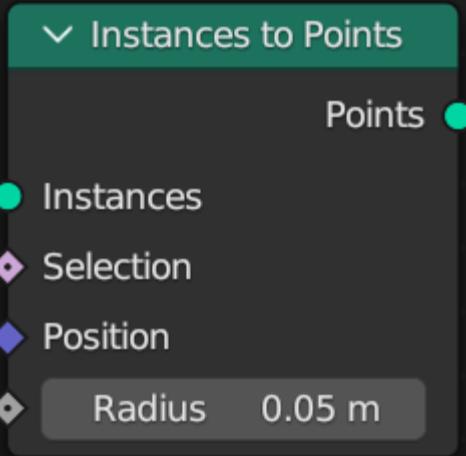


이해를 돋기 위해 원래의 지오메트리의 정보는 파란색,
상대 지오메트리의 정보는 빨간색으로 표시해 보았습니다.

이것은 '나 자신' 이 가지고 있는 상대 지오메트리의 점 번호 정보입니다.

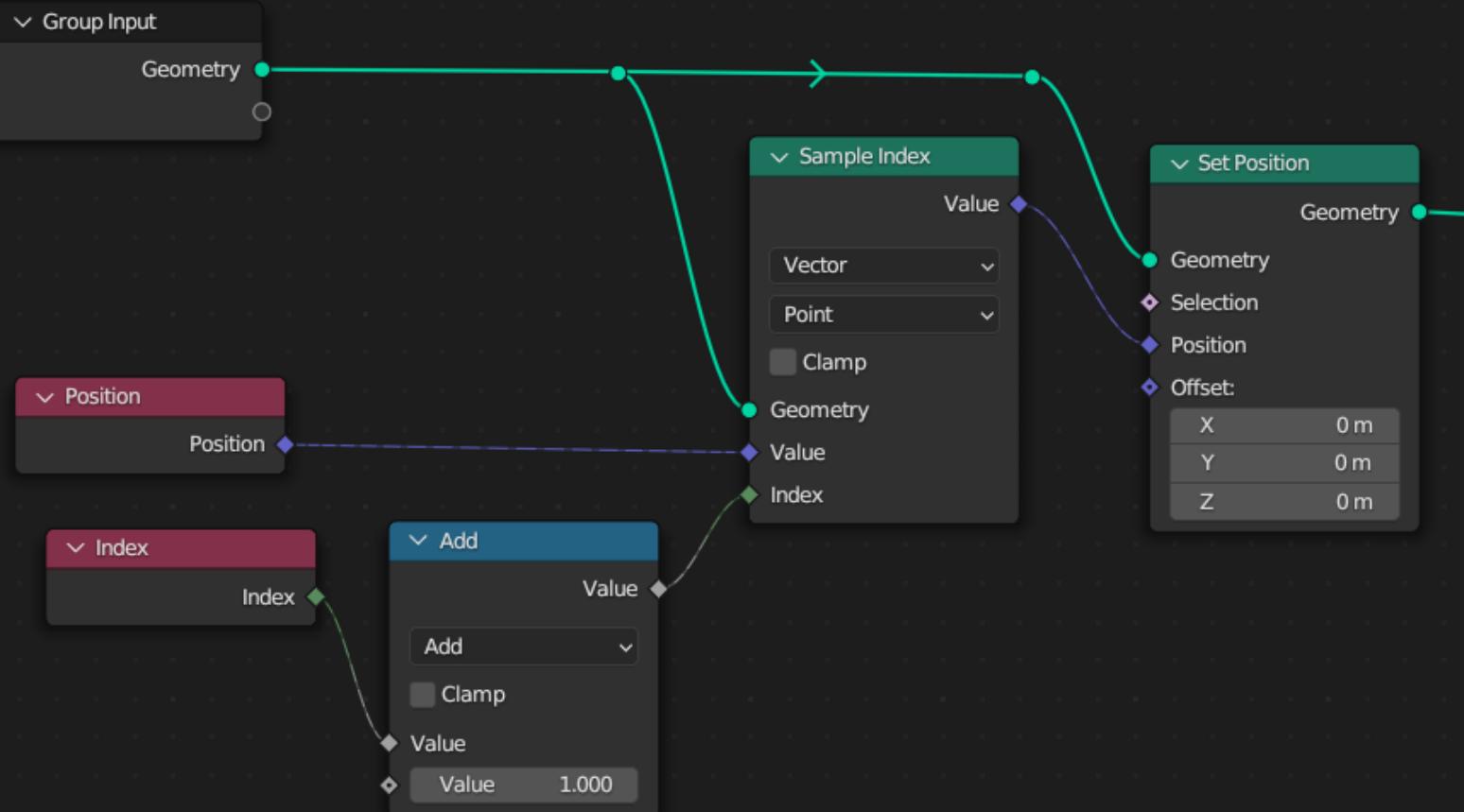
"나와 가장 가까운 너의 점의 정보를 줘.
몇번인지는 내가 알고 있어."

Appendix



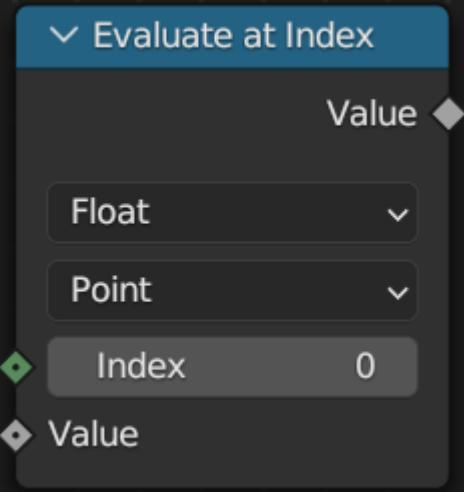
인스턴스는 포인트가 아니므로, Sample Nearest 를 사용할 수 없습니다.
그런 경우 Instances to Points노드로, 인스턴스를 포인트로 변경하여 점을 만들어 사용할 수도 있습니다.

타겟이 자기자신인 Sample Index?



자기 자신의 애트리뷰트 정보를
순서만 바꿔 가져오기 위해,
(예를 들어 자신의 1번 점의 정보를
0번 점에 가져오기 위해)
Sample Index에 [자기자신](#)을 꽂을 수도 있습니다만
이 경우 [Evaluate at Index](#)와 동일하게 작동합니다.

Appendix



Evaluate at Index : 입력한 인덱스에 할당된 attribute를 불러옵니다.

작동방식이 sample index와 흡사하지만, 자기 자신의 정보를 불러온다는 점이 다릅니다.

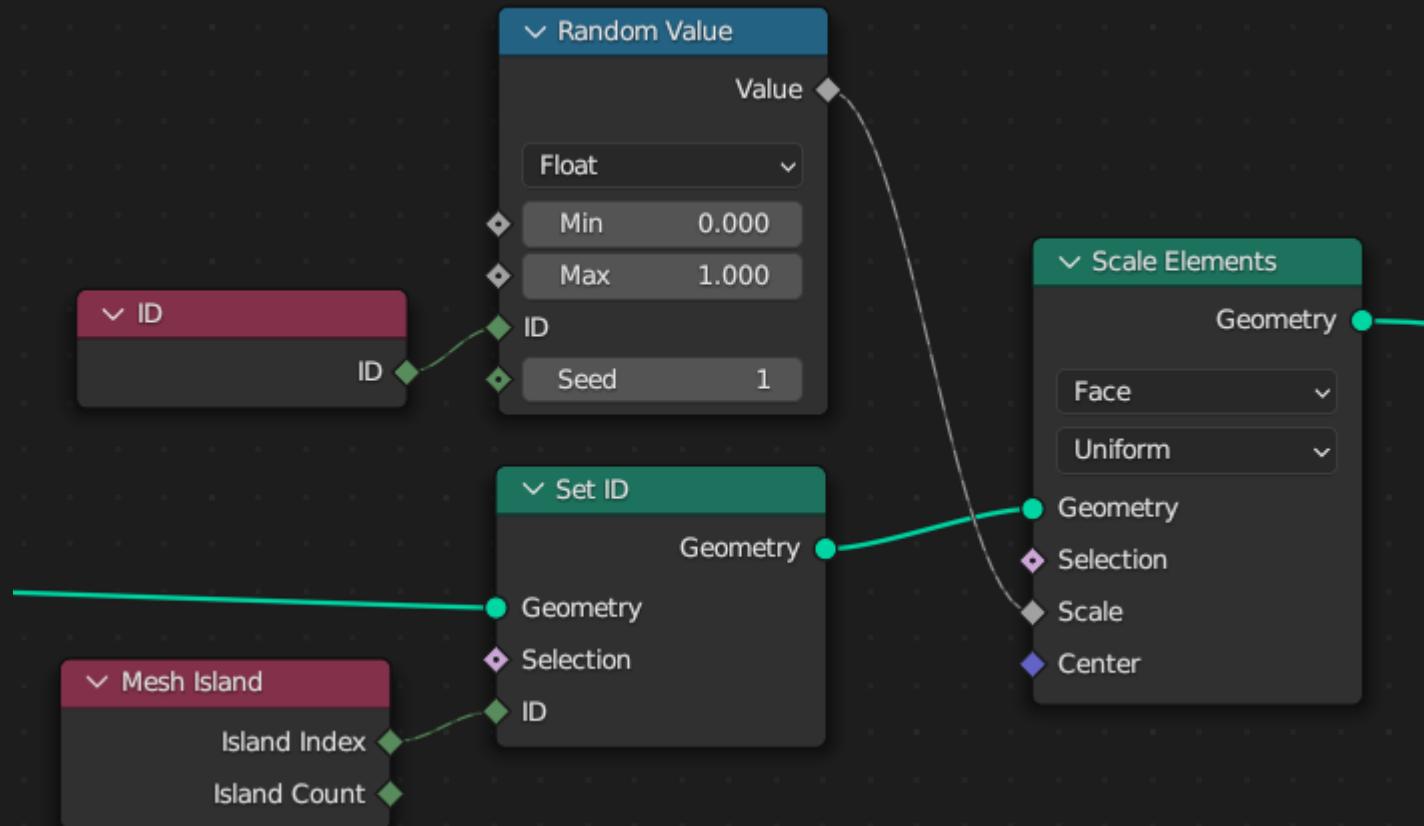
※이 노드의 이름은 3.4 이전에는 *Field at Index* 였습니다.

041강 메쉬의 변형 (3)

지오메트리의 ID에 대한 개념 알아보기

메쉬 변형 노드들 살펴보기

(Dual Mesh, Scale Elements, Convex Hull)



Index와 ID

▼ ID

ID ◆— 지오메트리에는 Index와는 다른 ID라는 개념도 있습니다.

- 이 값은 따로 지정하기 전에는 존재하지 않으며,
존재하지 않는 ID를 불러오려 하면 대신 Index를 불러옵니다.

▼ Set ID

Geometry ●

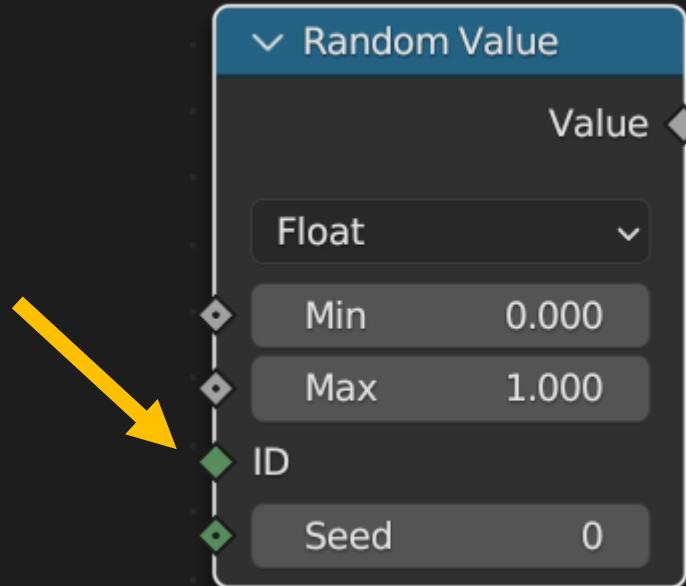
- Set ID로 값을 지정할 수 있습니다.

● Geometry

◆ Selection

◆ ID

Index와 ID

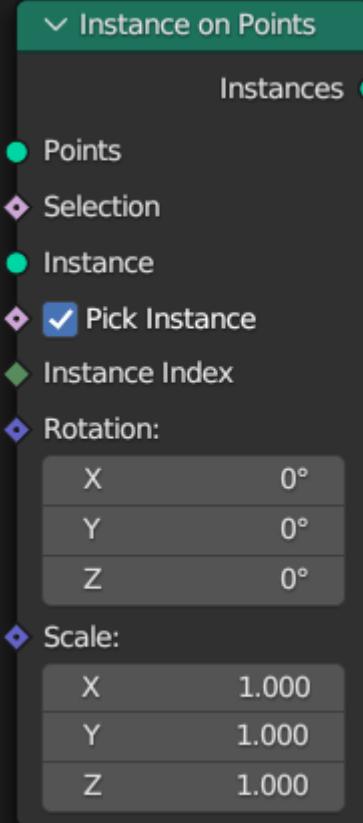


ID는 Random Value를 만들어낼 때 사용됩니다.

ID없이 Random Value를 꺼내면, 아래와 같은 (생각보다 복잡한) 단계가 실행됩니다.

1. Random Value는 ID를 불러오려 시도합니다.
2. ID가 없으므로, 대신 Index를 사용합니다.
3. Random Value는 Index를 바탕으로 랜덤값을 생성합니다.

Index와 ID



ID는 [Instance on Points](#)에서 인스턴스를 고를 때 사용합니다.

ID없이 [Pick Instance](#)로 인스턴스를 고르면, 아래와 같은 (생각보다 복잡한) 단계가 실행됩니다.

1. Instance on Points 는 ID에 따라 인스턴스를 불러오려 시도합니다.

2. ID가 없으므로, 대신 Index를 사용합니다.

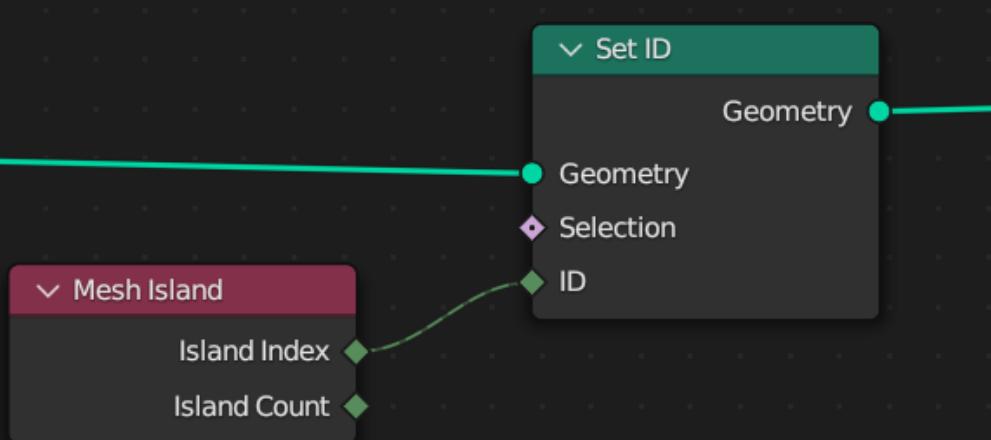
3. Instance on Points는 Index를 바탕으로 인스턴스를 가져옵니다.

그에 따라, 인스턴스가 0,1,2,3... 번 차례대로 불러와집니다.

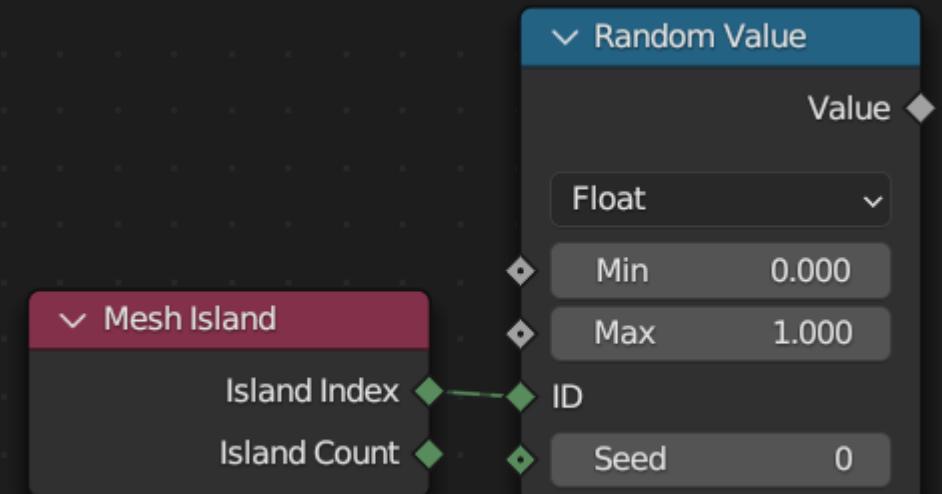
※ Instance on Points는 인덱스에 대응하는 인스턴스가 없을 땐, 순환하여 불러옵니다.
즉, 인스턴스 3개가 입력되었다면, 0,1,2,0,1,2,0,1,2,로 불러옵니다.

Index와 ID

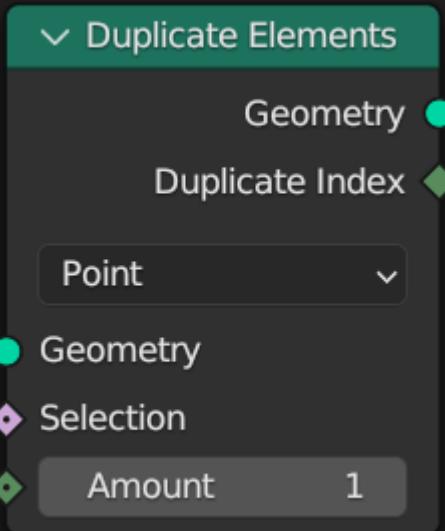
ID를 생성해두면, Random Value와 같은 ID를 사용하는 노드에 자동으로 적용됩니다.



하지만 ID를 생성하지 않고, 필요에 따라 ID 입력을 연결해서 사용하는 것이 더 직관적일 수도 있습니다.



Duplicate Elements

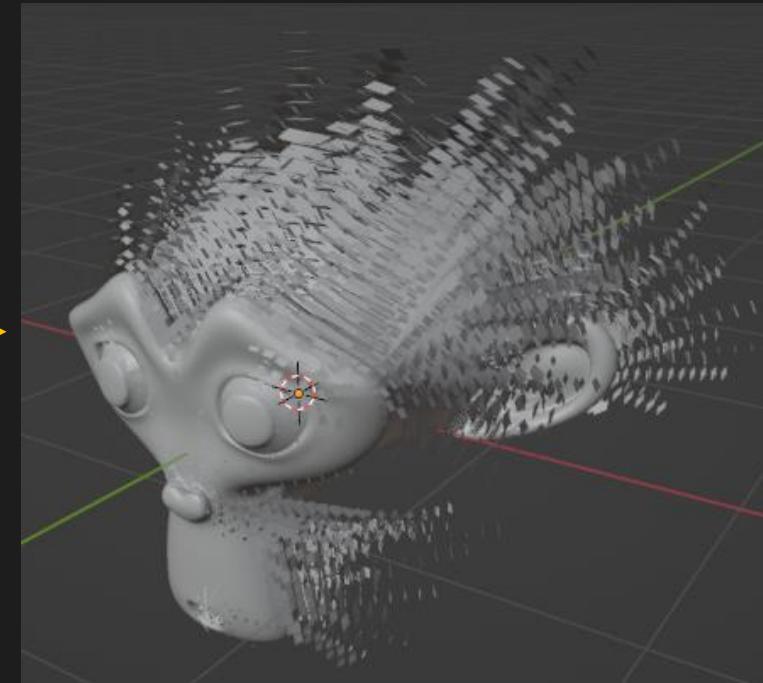
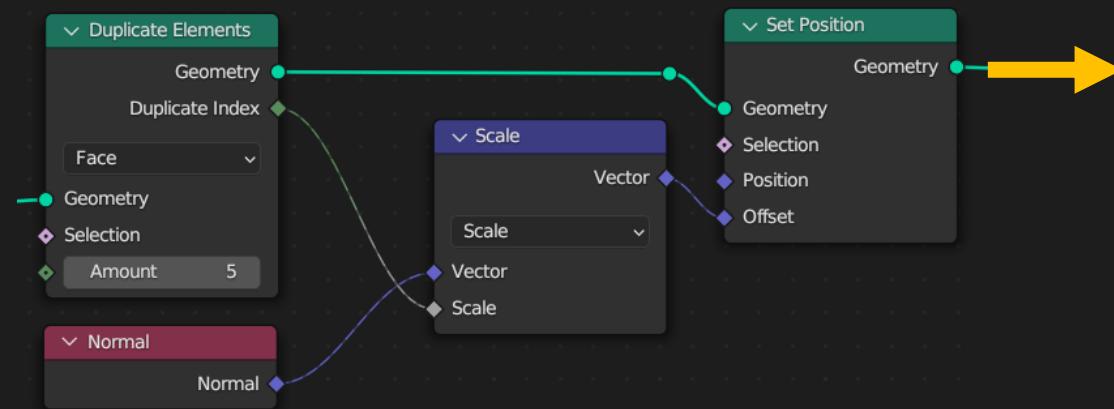


지오메트리 성분을 복제합니다.

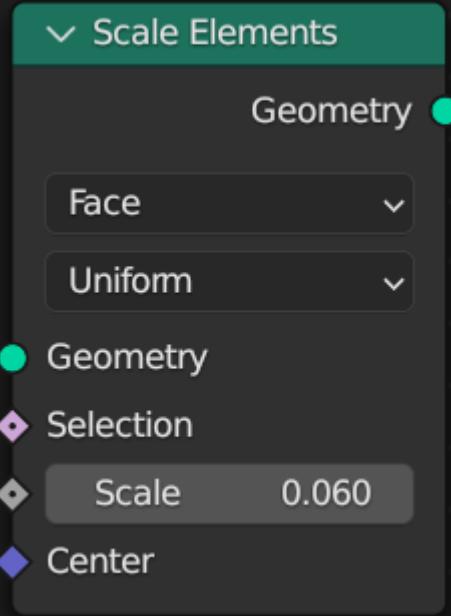
Array (Modifier) 와 비슷할 것 같지만, 전혀 다른 방식으로 작동합니다.

만약 Face를 선택하면, Face가 분리된 채로 복제됩니다!

이렇게 분리되어 복제된 Elements들은 Duplicate Index 로 구분할 수 있습니다.



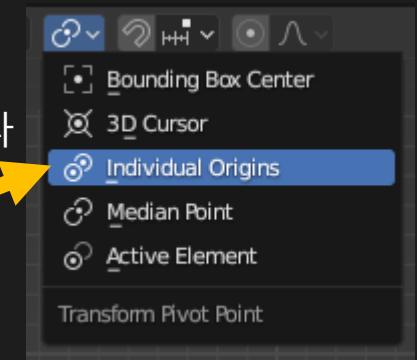
Scale Elements



지오메트리 성분의 크기를 조절합니다.

Transform Geometry에서의 스케일과 다른 점은, 서로 분리되어 있는 island마다
개별적으로 확대/축소된다는 점입니다.

마치 모델링에서, Pivot Point를 Individual로 한 것 같이 작동합니다.



Split Edges

▼ Split Edges

Mesh ●

엣지로 붙어있는 면을 분리시킵니다.

● Mesh

모델링에서의 Rip (단축키 V) 과 비슷한 기능입니다.

◆ Selection

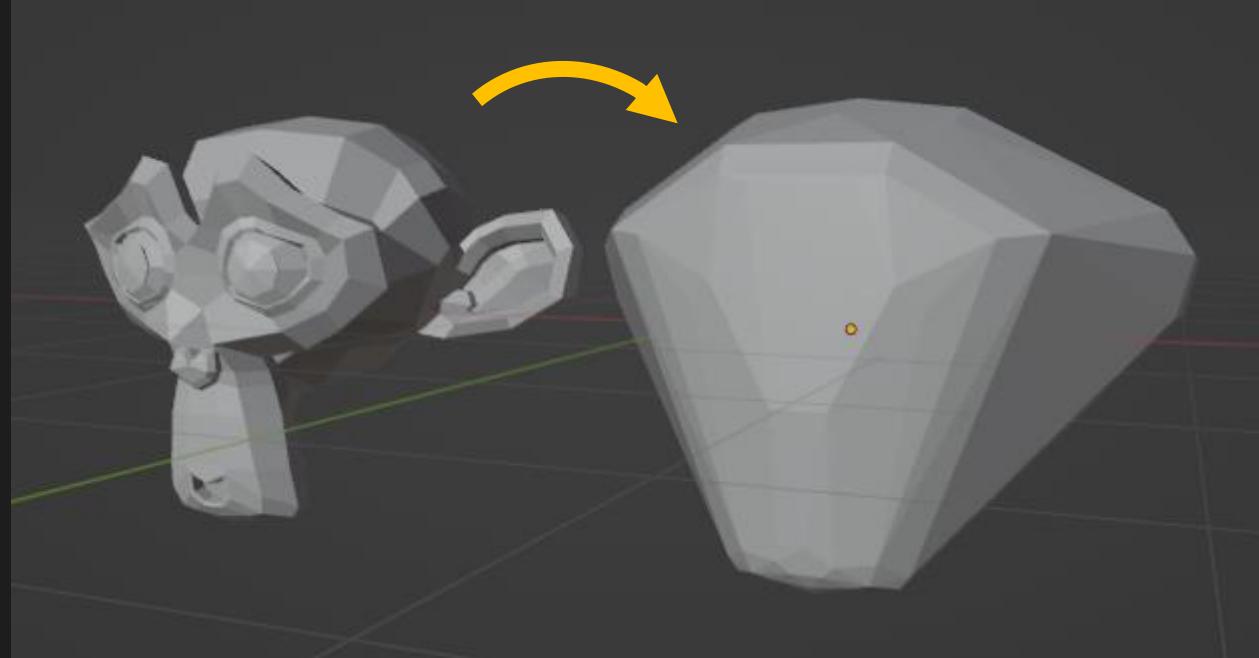
Convex Hull

✓ Convex Hull

Convex Hull

Geometry

지오메트리를 감싸는 볼록다면체를 생성합니다.
반드시 메쉬를 입력할 필요는 없으며, 포인트 클라우드를 통해 생성할 수도 있습니다.



Dual Mesh

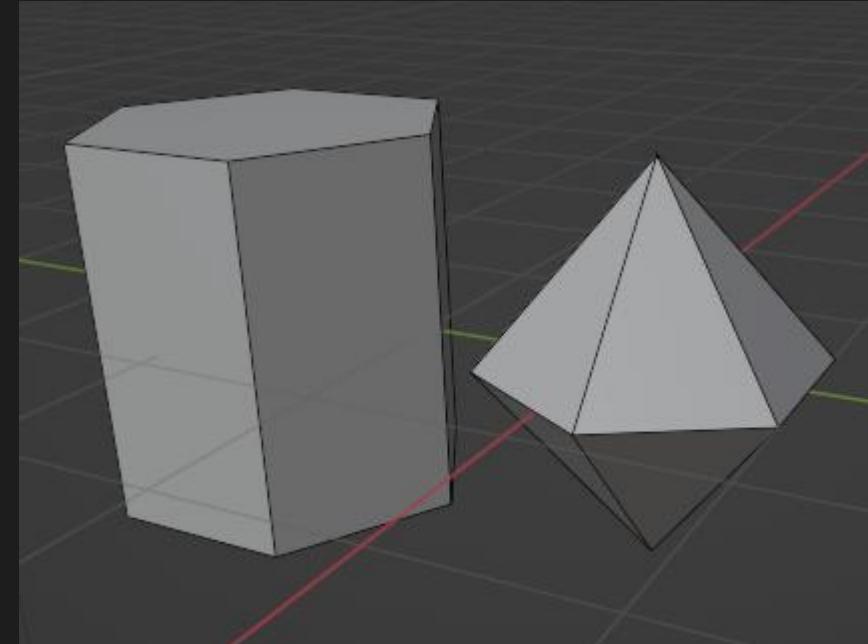


Dual이란 지오메트리의 Face를 Point로 치환하여 재구성하는 것을 말합니다.

변환되면서, 원래의 Face가 가진 Attribute는 Point로 옮겨갑니다.

Convex Hull과 더불어, 새로운 면을 생성하는 노드 중 하나입니다.

Keep Boundaries : 변환할 때, Non-Manifold 경계를 보존합니다.



042강 커브

커브의 정의와 종류

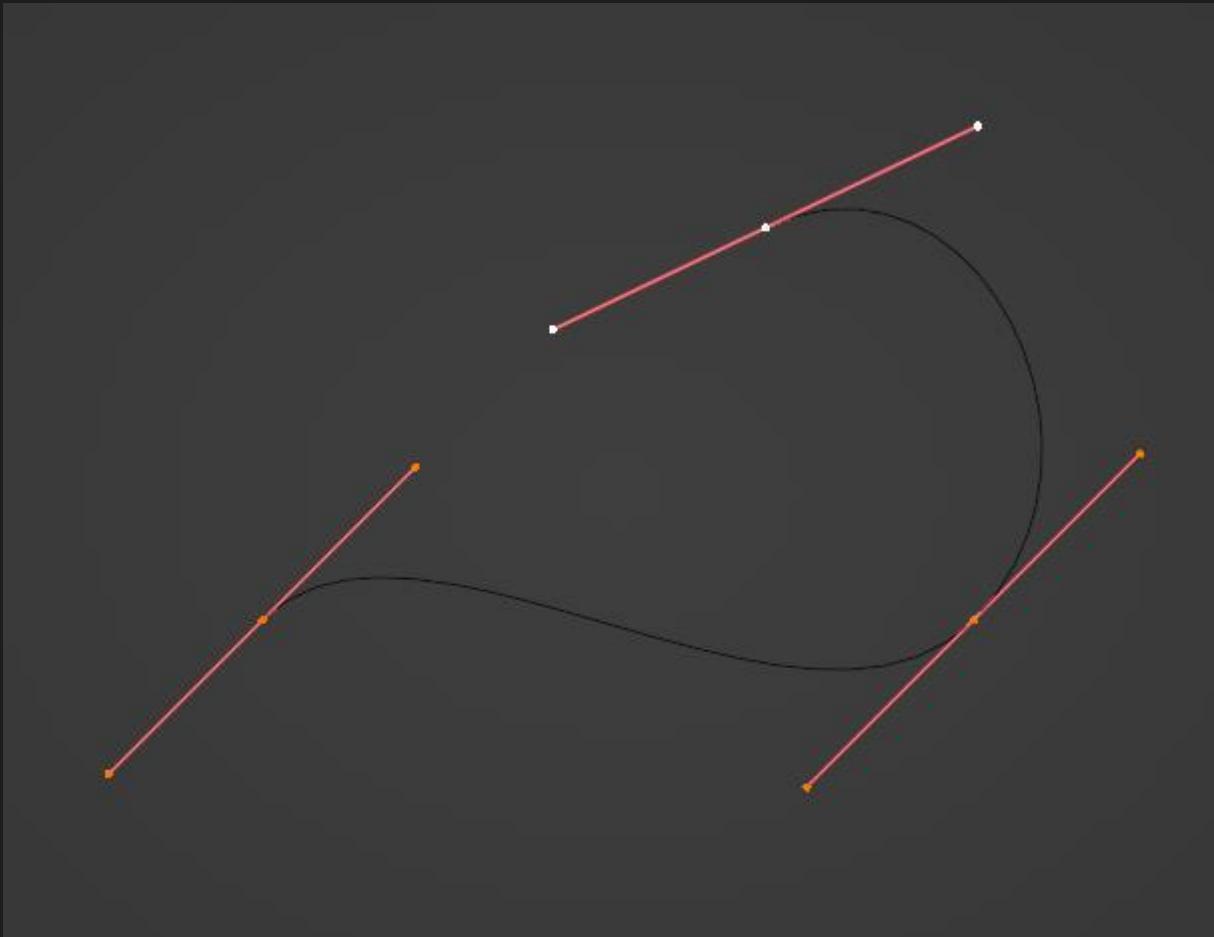
커브 핸들 컨트롤



Curve

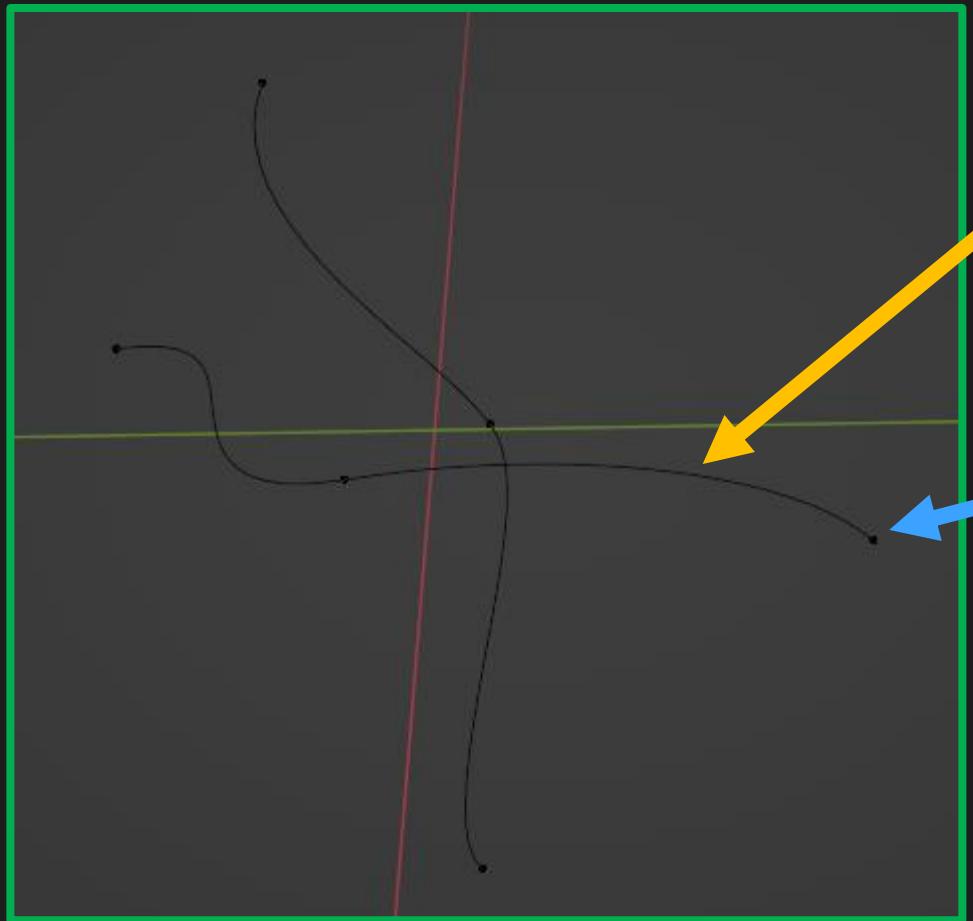
Curve는 Mesh와는 다른 개념의 지오메트리입니다.

단순하게 설명하자면, 커브는 점들을 곡선으로 잇고, 면을 생성하지 않습니다.



Curve의 구성

커브는 Spline과 Control Point로 구성됩니다.



Spline : 각각의 곡선을 Spline이라고 부릅니다.

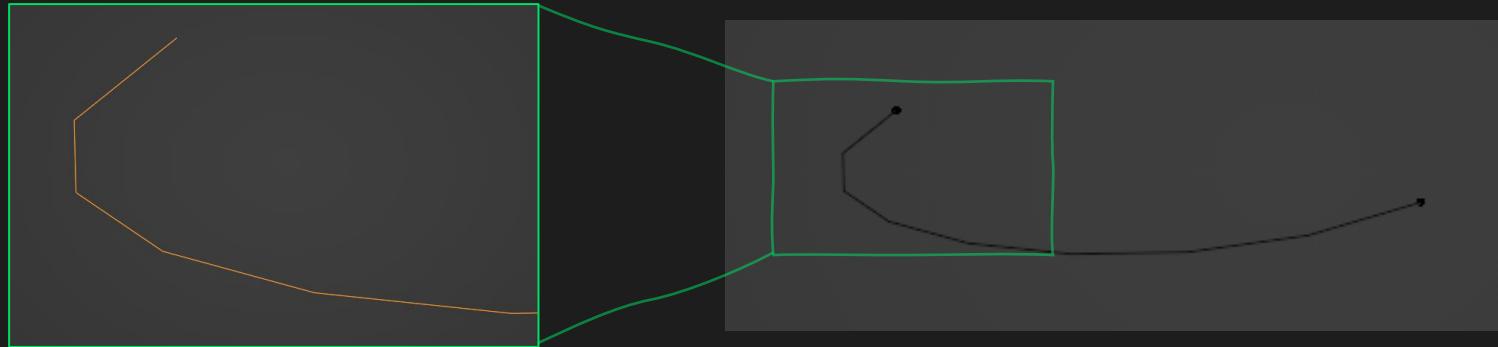
Spline들이 모여 **Curve** 오브젝트를 만듭니다.

Control Point들을 곡선으로 이으면 Spline이 됩니다.

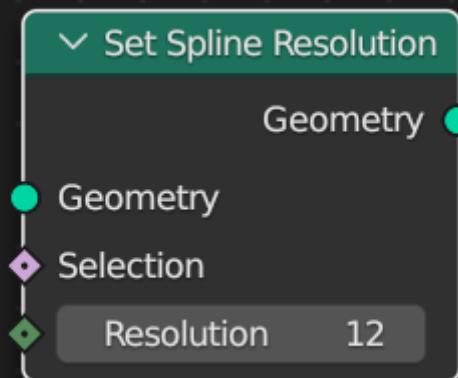
※지오메트리 노드 내에서 Control point는 다른 점들처럼 'point' 취급됩니다.
예컨대 position 입력 노드는 컨트롤 포인트의 위치를 나타내고,
set position으로 컨트롤 포인트의 위치를 변경할 수 있습니다.

Spline Resolution

Curve는 Control Point들을 곡선으로 잇습니다.
그런데 곡선을 자세히 보면, 이들은 또다른 점들을 직선으로 이은 것처럼 보입니다.



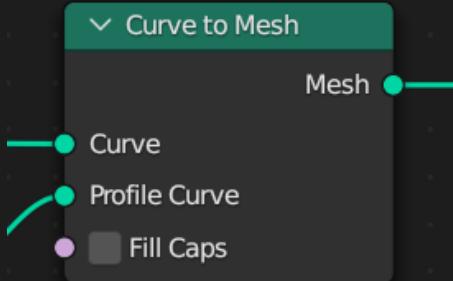
이처럼 커브는 직선을 이어 곡선을 만듭니다. 점들이 많을수록 부드럽게 보이지만, 연산이 오래 걸립니다.
이 점들의 갯수를 **Resolution**이라고 합니다.
이 점들은 컨트롤 포인트가 아니므로 제어할 수 없지만, 커브 타입을 변환시키거나(Resample Curve)
메쉬로 변환하면 (Curve to Mesh) 컨트롤할 수 있습니다.



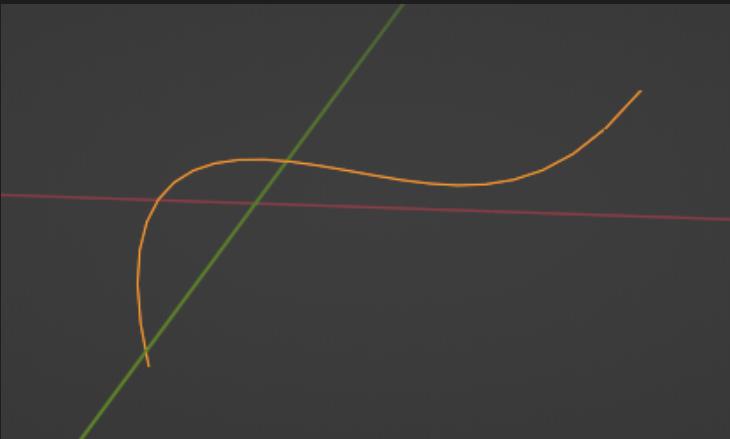
Set Spline Resolution으로 해상도를 컨트롤할 수 있습니다.

커브의 변환

Curve to Mesh



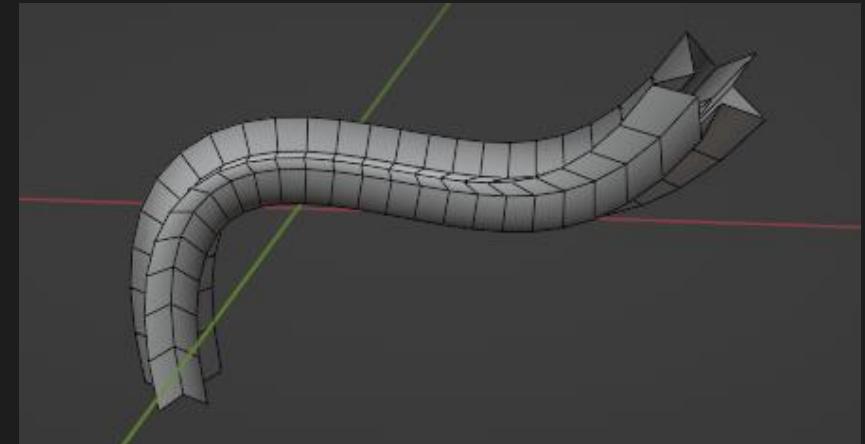
커브는 Edge같이 그 자체로는 렌더링에서 보여지지 않습니다.
실제로 사용하기 위해서는 Curve to Mesh를 통해 커브를 메쉬로 변환해야 합니다.
Profile Curve를 지정하여 어떤 모양으로 변환할지 선택할 수 있습니다.



Curve



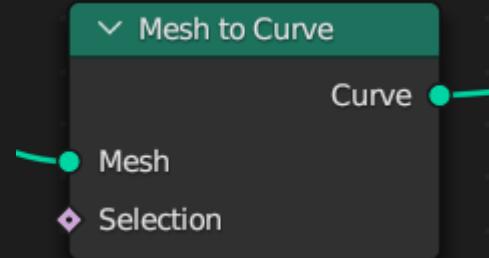
Profile Curve



Curve to Mesh

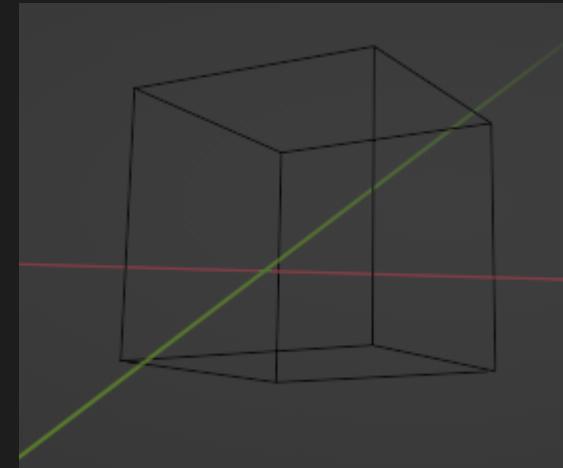
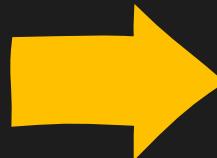
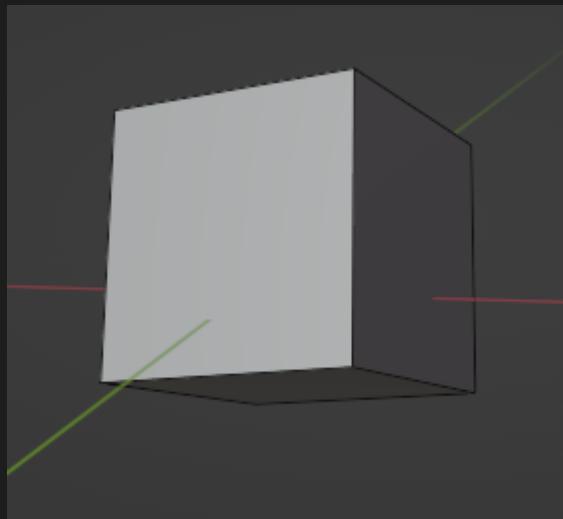
커브의 변환

Mesh to Curve



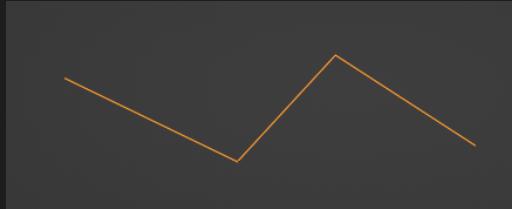
메쉬를 커브로 변환하면, 면은 사라지고 edge들이 임의로 여러 개의 spline을 형성합니다.

경우에 따라, 커브를 바로 사용하는 것보다 이렇게 메쉬를 커브로 변환해 쓰는 쪽이 편할 수도 있습니다.

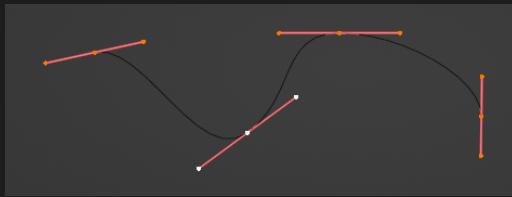


Spline Type

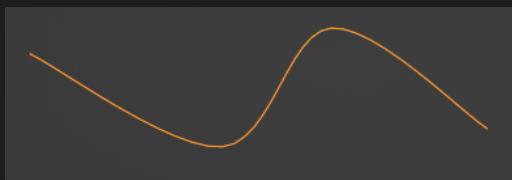
Set Spline Type으로 커브의 종류를 선택할 수 있습니다.



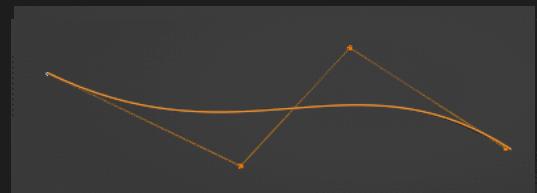
Poly : 포인트를 직선으로 잇습니다.



Bezier : 3차 베지에 곡선으로 만듭니다. 가장 일반적인 타입입니다.

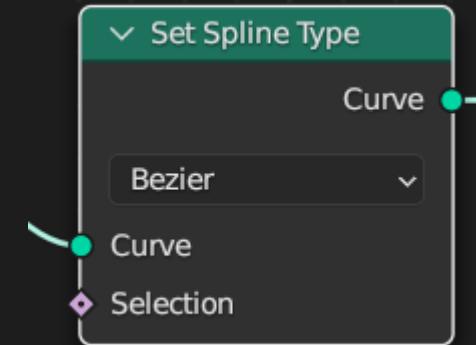


Catmull Rom : 포인트를 자동으로 곡선으로 잇습니다. 추가적인 컨트롤은 불가능합니다.



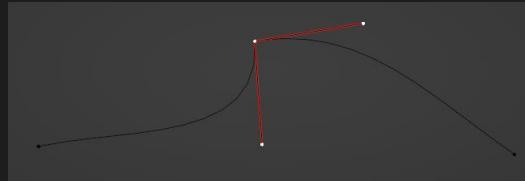
NURBS : 커브를 넙스로 만듭니다.

곡선이 컨트롤 포인트를 지나지 않아 통제가 어렵지만, 아주 부드러운 곡선이 만들어지므로 경우에 따라 유용하게 사용할 수 있습니다.

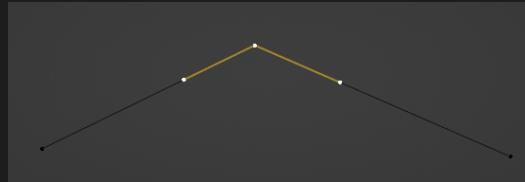
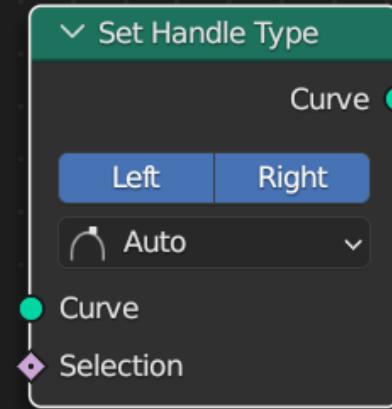


Handle Type (Bezier)

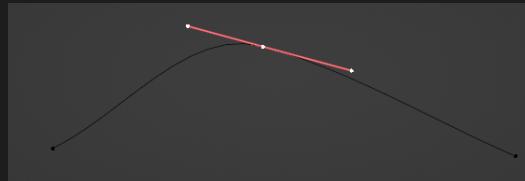
Set Handle Type을 통해 베지에 곡선의 핸들 타입을 고를 수 있습니다.



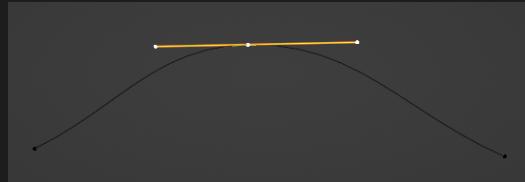
Free : 핸들 움직임에 제약을 주지 않습니다.



Vector : 핸들이 이웃한 점을 곧바로 향합니다.



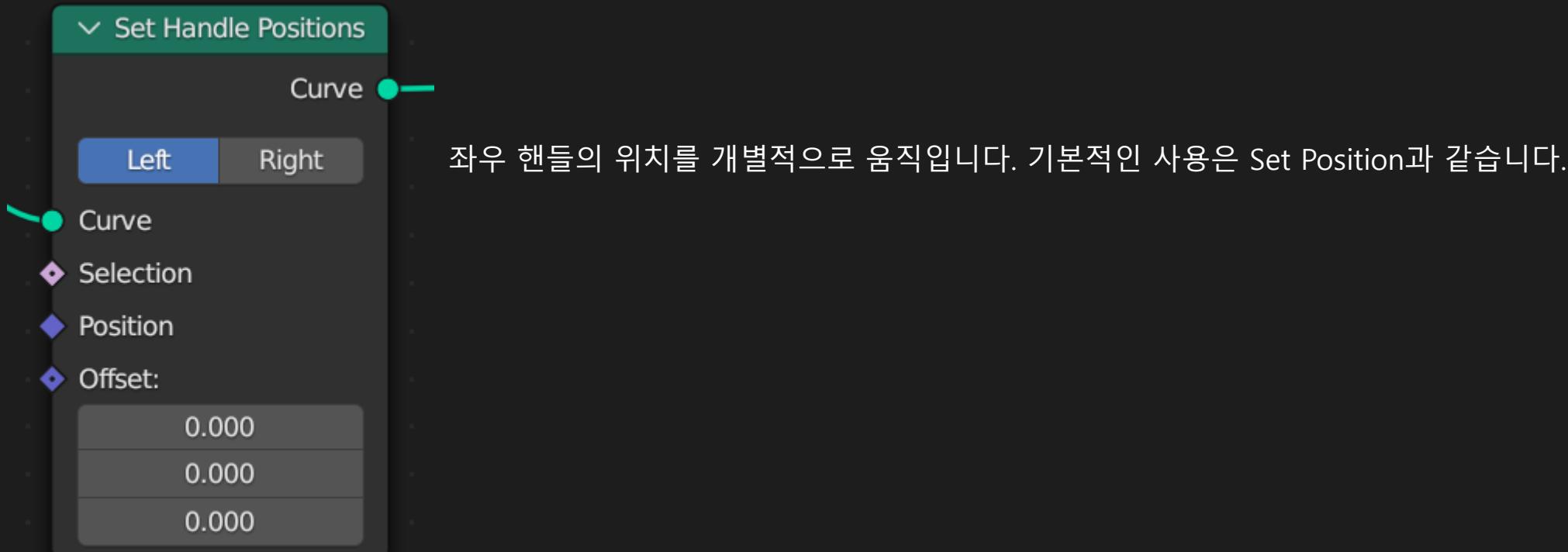
Align : 좌우 핸들이 일직선상에 놓여 부드러운 곡선을 유지합니다.



Auto : 부드러운 곡선이 되도록 핸들의 위치를 자동으로 조절합니다.

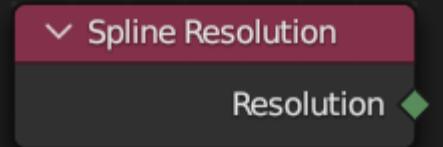
Handle Position

Set Handle Position을 통해 좌우 핸들의 위치를 조정할 수 있습니다.

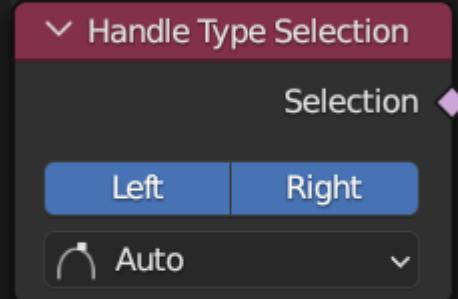


Appendix

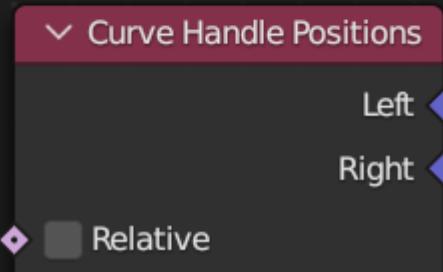
앞에서 알아본 것들에 대응되는 입력 노드들이 존재합니다.



Spline Resolution : 스플라인의 해상도를 출력합니다.



Handle Type Selection : 핸들을 타입에 따라 선택합니다.
Set Handle Position을 사용했다면 핸들 타입이 변했을 수도 있으므로 주의하세요.
(vector → free, auto → align)



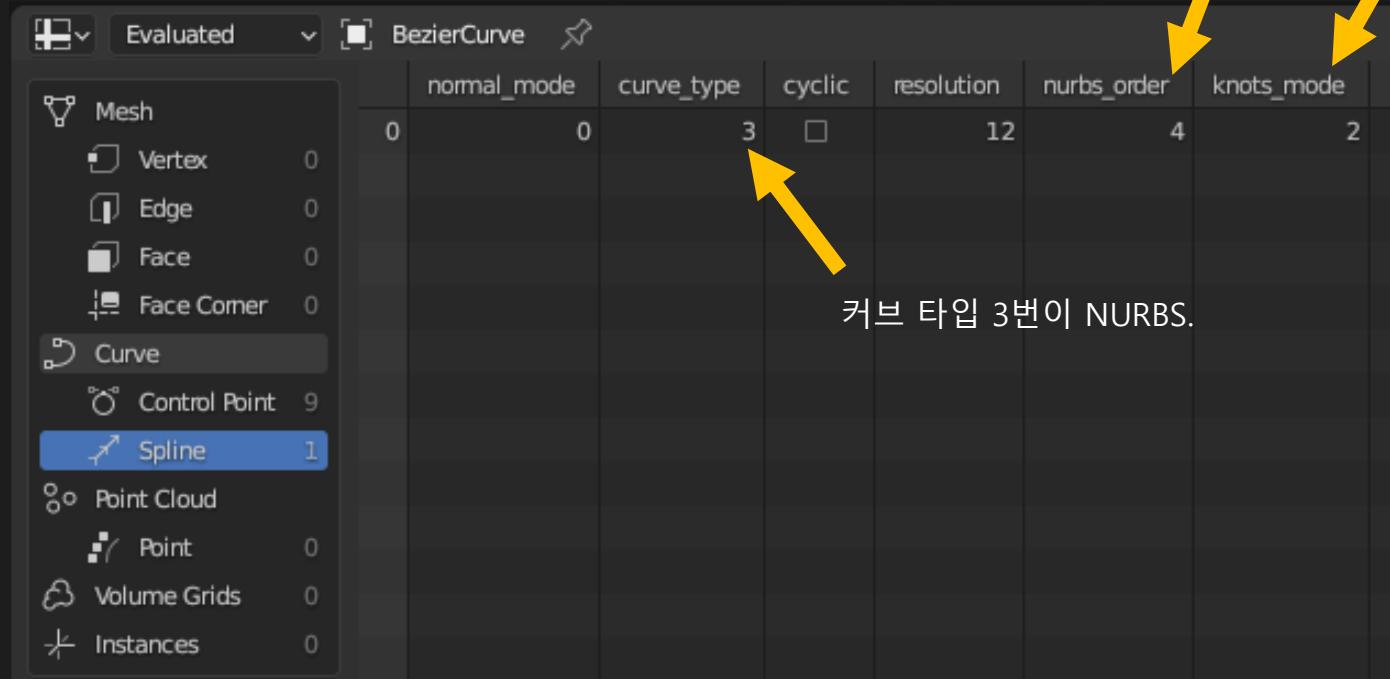
Curve Handle Positions : 좌/우 핸들의 위치.
Relative를 켜면 컨트롤 포인트를 기준으로 위치를 출력합니다.
(= 컨트롤 포인트에서부터 얼마나 떨어져 있는지를 출력합니다.)

Appendix

스프레드 시트에서 NURBS 관련 Attribute를 확인할 수 있습니다.

정보에 접근은 가능하지만, 컨트롤은 할 수 없습니다. (*3.5 기준)

나중에는 유용하게 사용할 수도 있을지도 모릅니다.



	normal_mode	curve_type	cyclic	resolution	nurbs_order	knots_mode
BezierCurve	0	3	<input type="checkbox"/>	12	4	2

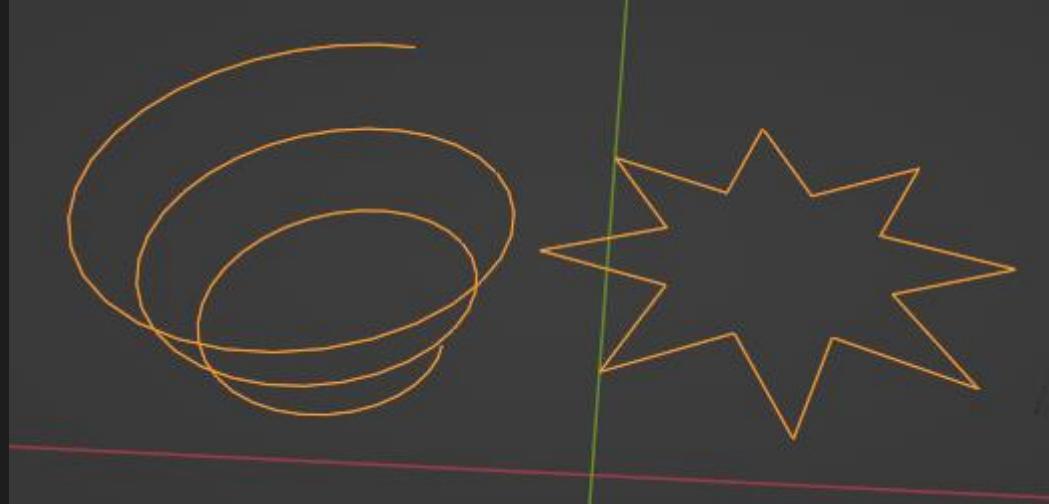
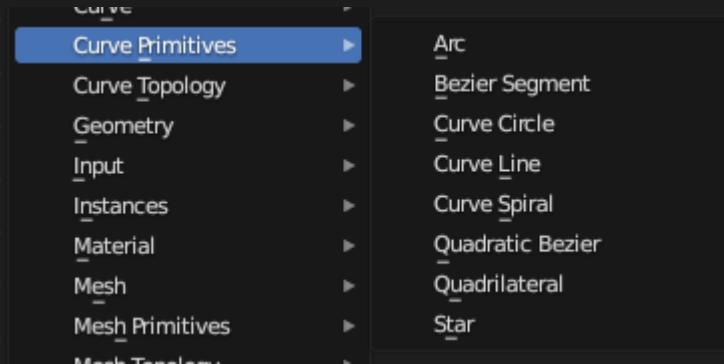
커브 타입 3번이 NURBS.

스프레드 시트에는 이외에도 커브 관련 attribute들이 있으니, 한번 확인해보시는 것도 좋습니다.

Appendix

Curve Primitives

메쉬처럼 커브도 기본도형을 제공합니다.



다만, 이들은 모두 **Poly** 타입으로 제공됨에 유의하세요.

예를 들어 Curve Circle은 4개의 Bezier포인트가 아니고 32개의 Poly 포인트를 가집니다.

심지어 2차 베지에(quadratic Bezier) 마저도 출력은 Poly입니다!

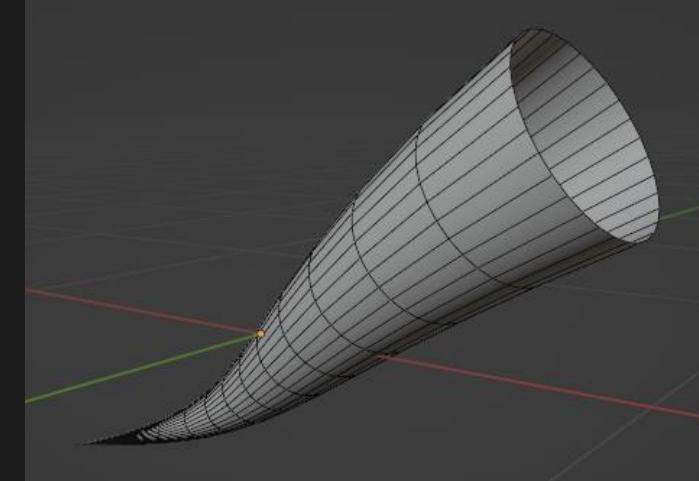
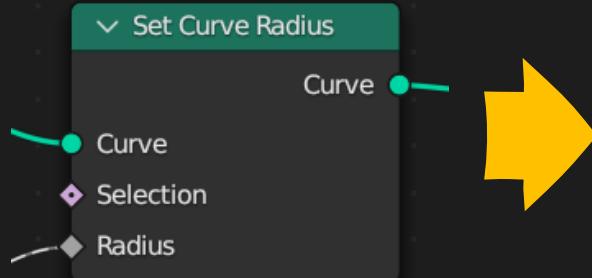
043강 커브의 컨트롤 (1)

커브가 가지는 정보들
Sample Curve를 이용하여 목걸이 만들기

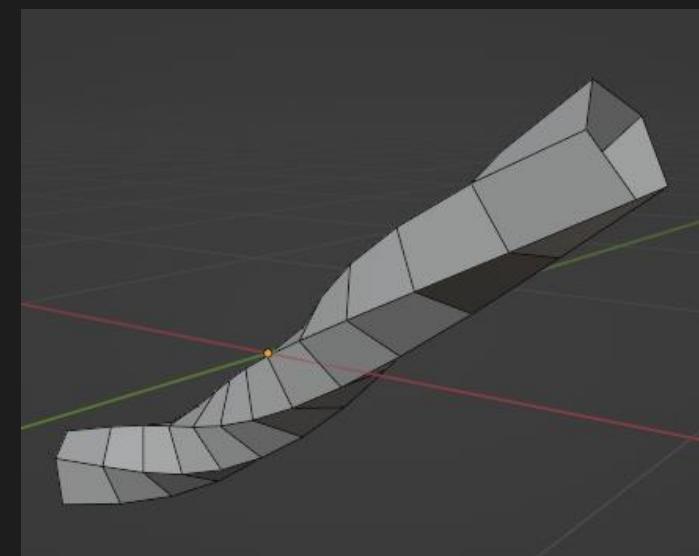
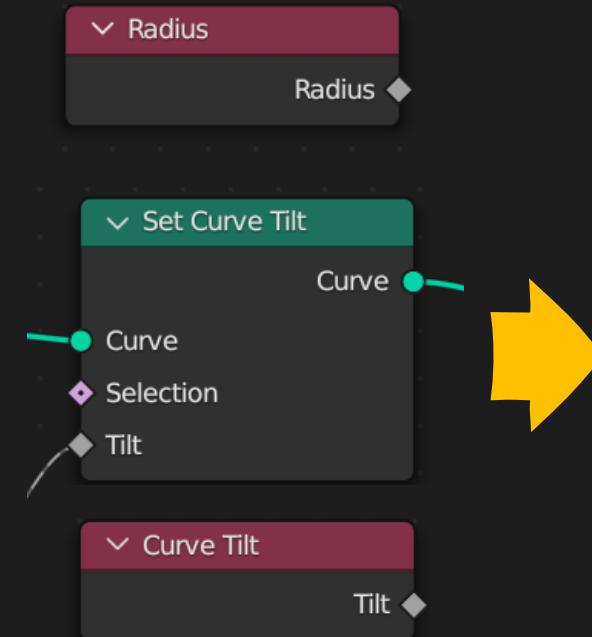


Radius and Tilt

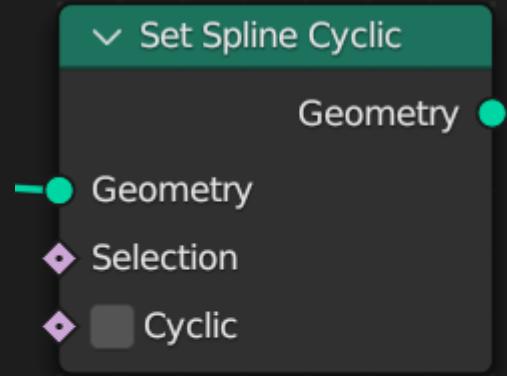
Set Curve Radius, Set Curve Tilt



커브의 Mesh 변환시에 사용될,
반지름과 회전값을 읽고 쓸수 있습니다.

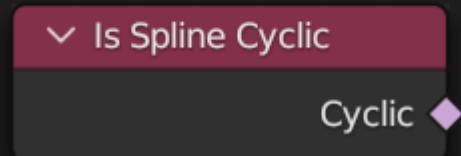
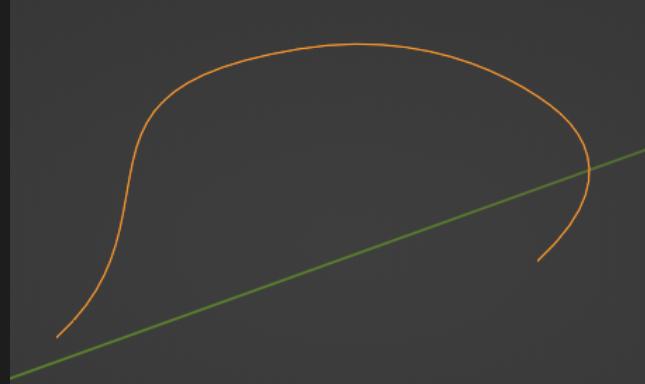


Cyclic



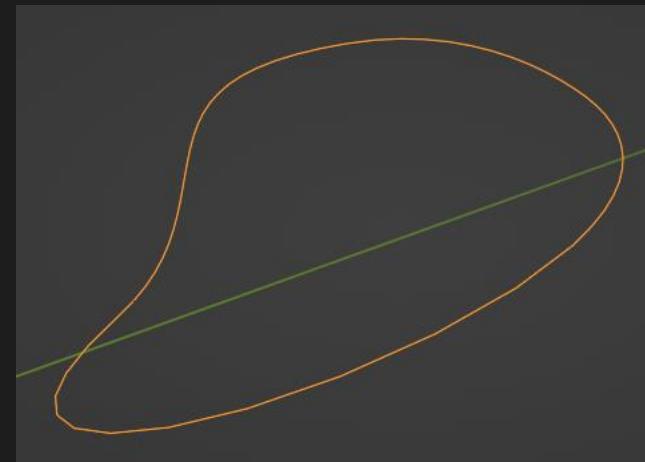
Set Spline Cyclic

Spline의 마지막 점과 끝점을 이어서 고리를 만들 것인지 결정합니다.



Is Spline Cyclic

Spline이 Cyclic인지 아닌지를 판별합니다.



Curve 위의 정보

Spline을 시작점부터 끝점까지 이동하면서, **Factor**와 **Length**를 체크합니다.

Spline Parameter

▼ Spline Parameter

Factor ◆

Length ◇

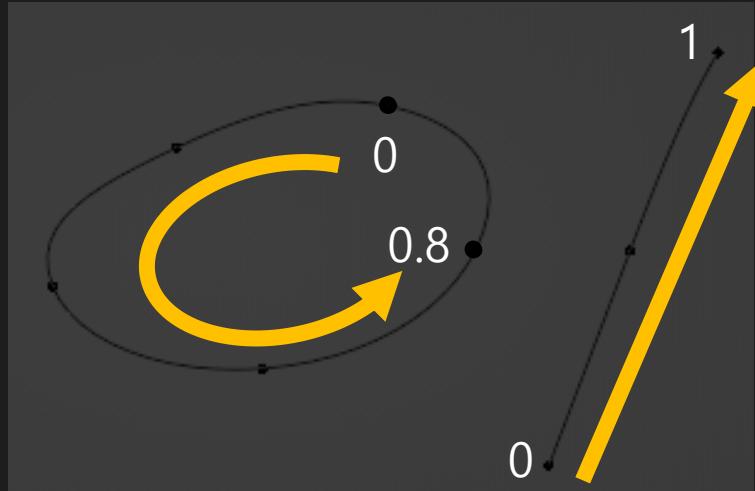
Index ◇

Factor : Spline의 시작점부터 끝점까지를 0에서 1로 두고, 얼마나 지나갔는지를 계산합니다.

Length : 시작점부터 끝점 방향으로, 얼마나 지나갔는지 거리를 계산합니다.

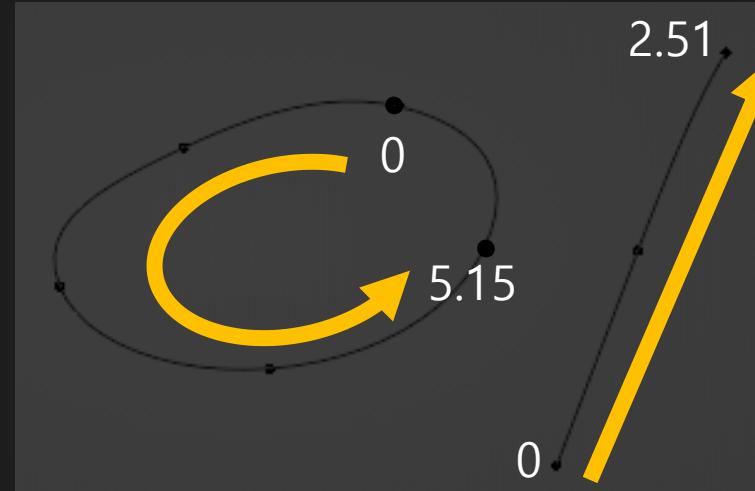
Index : 스플라인의 시작점부터 차례대로 번호를 매깁니다.

Factor



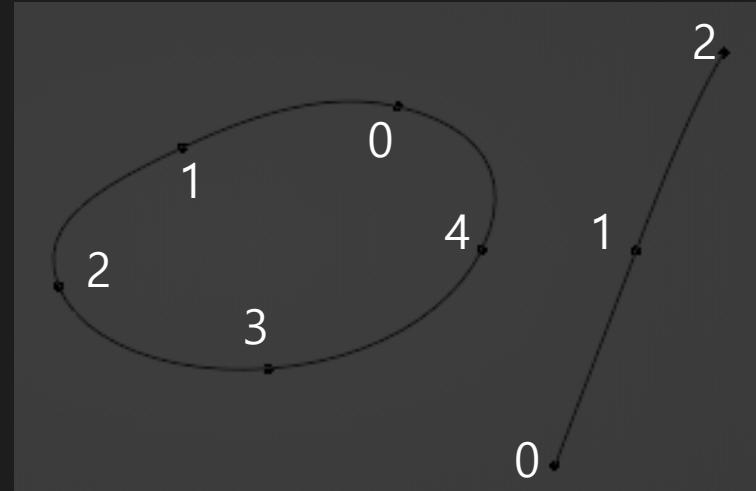
Cyclic의 경우 끝점의 팩터값이 1이 아닙니다.
(한바퀴를 돌아야 1이 되기 때문에.)

Length



Cyclic의 경우 끝점의 Length값이 전체 스플라인 길이가 아닙니다.

Index



Curve 위의 정보

Spline Length

▼ Spline Length

- Length ◆ Length : 각각의 스플라인의 길이
- Point Count ◆ Point Count : 각각의 스플라인의 총 컨트롤 포인트

Curve Length

▼ Curve Length

- Length ● 입력받은 커브의 전체 길이를 출력합니다.
※ 이 출력값은 Single Value이기 때문에,
자신이 아닌 지오메트리에도 연결할 수 있습니다.

Endpoint Selection

▼ Endpoint Selection

- Selection ◆ Spline의 시작점과 끝점을 선택합니다.
Size를 늘려 시작점/끝점 **근처**의 점도 선택할 수 있습니다.



Tangent

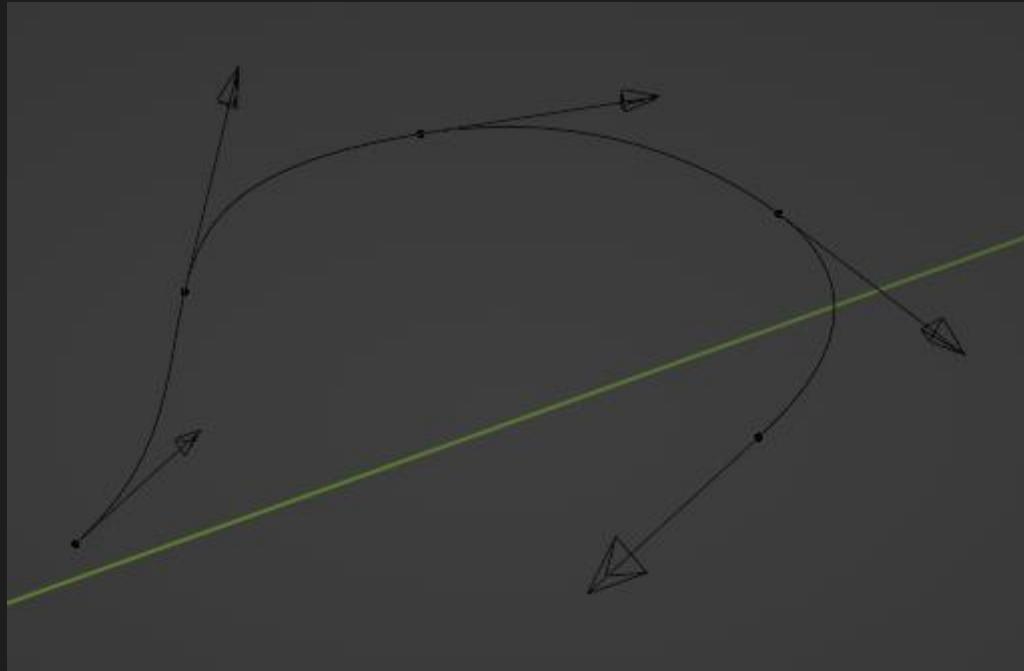
Curve Tangent

Curve Tangent

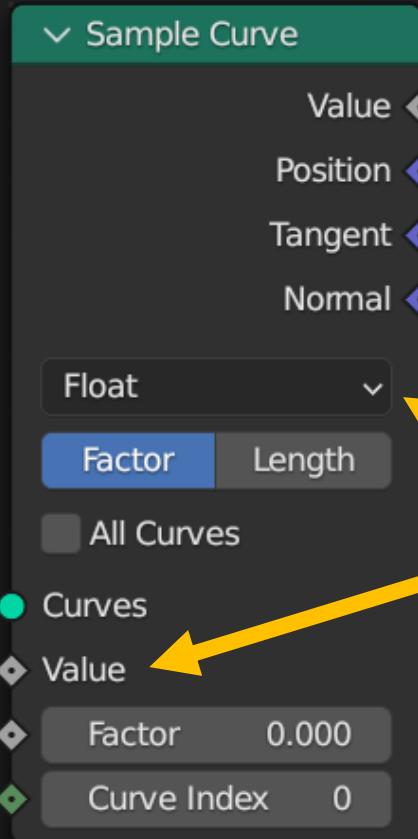
Tangent ◆

커브 위의 점선의 방향벡터를 출력합니다.

방향을 회전으로 변환해주는 [Alien Euler to Vector](#) 와 같이 쓰면 좋습니다.



Sample Curve



커브 위의 정보들은 연속적으로 존재하지만, 앞서 확인한 입력 노드들로는 각 컨트롤 포인트 위의 값만 확인할 수 있습니다.

Sample Curve는 Factor나 Length를 이용하여 커브 위의 정보를 연속적으로 확인할 수 있습니다.

Curve의 특정 Attribute를 샘플링할 수 있습니다. (옵션)

● Curves

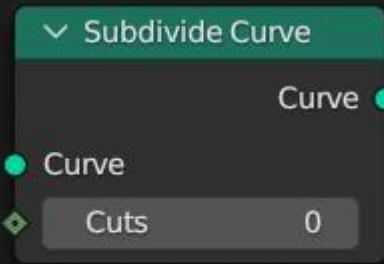
◆ Value

Factor 0.000

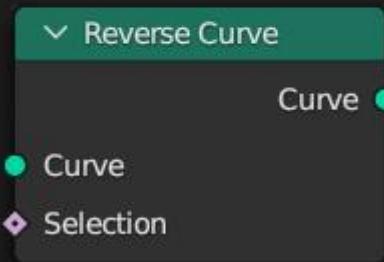
Curve Index 0

Curve Index : 어느 Spline을 샘플링할지 선택합니다.

Appendix



Subdivide Curve : 컨트롤 포인트 사이를 몇 개로 쪼갤지 정할 수 있습니다.



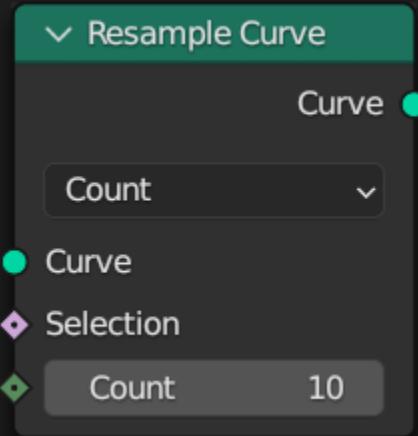
Reverse Curve : 커브의 시작과 끝 순서를 뒤집습니다.
즉 Factor나 Length의 순서가 뒤바뀝니다.

044강 커브의 컨트롤 (2)

커브를 변형시키는 노드들
마법진 애니메이션



Resample Curve



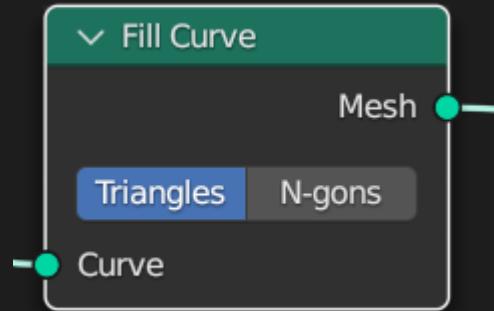
Spline 위의 컨트롤 포인트 개수를 조절합니다.
이 과정에서 커브는 Poly로 변화합니다.

Count : Spline 위의 총 포인트 수를 지정하여 resample

Length : 포인트 사이의 거리를 지정하여 resample

Evaluated : 베지에 커브 등의 Resolution을 그대로 활용합니다.

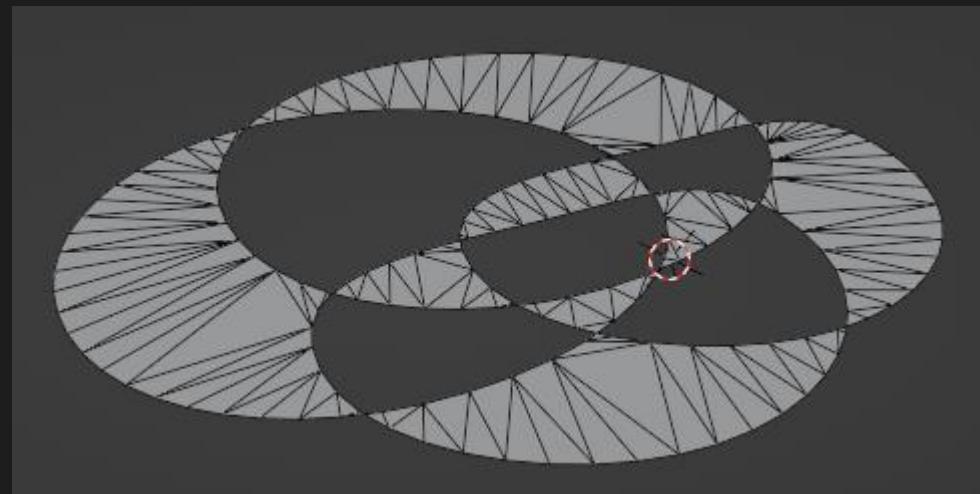
Fill Curve



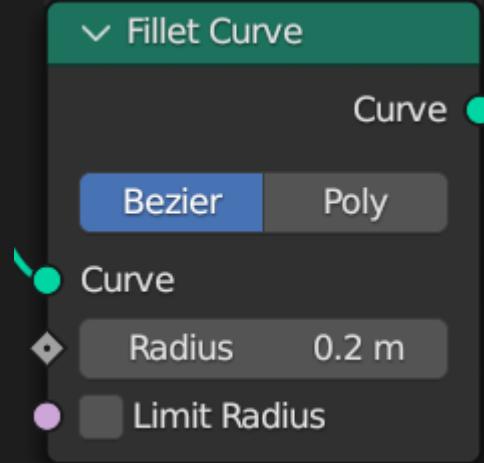
커브 내부에 면을 채워넣어 Mesh로 변환합니다.

이 과정에서 커브는 자동으로 2차원으로 변환되어, z축 정보는 사라집니다.

※커브가 이중으로 겹치는 부분은 채워지지 않습니다. 한편, 3중으로 겹치는 부분은 다시 채워집니다..



Fillet Curve

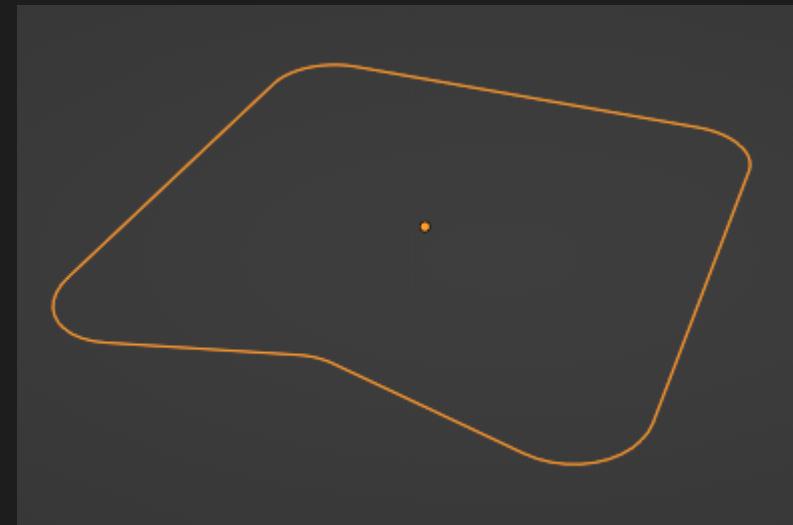
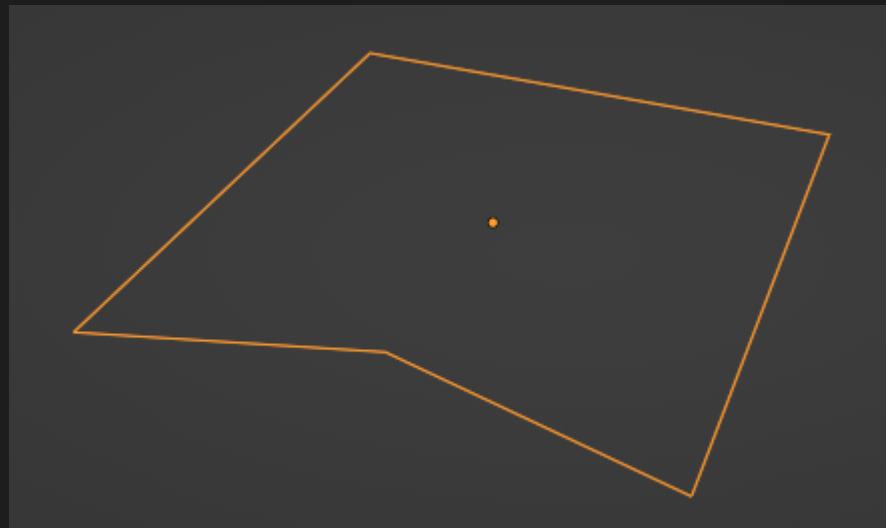


Fillet Curve는 각 컨트롤 포인트를 둥글립니다.

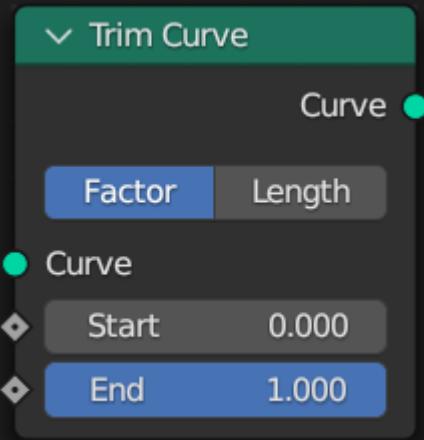
이는 모든 컨트롤 포인트에 적용되므로, Resample Curve 후에 적용할 때는 예기치 않은 결과가 나타날 수 있습니다.

Bezier / Poly : 어느 방식으로 둥글릴지를 선택합니다.

Bezier 모드는 입력받은 커브가 베지에일 때만 정상작동합니다.



Trim Curve



Trim Curve는 각 Spline의 시작점이나 끝점 부분을 자를 수 있습니다.



045강 커브의 UV

커브 방향을 따라 UV를 만드는 법

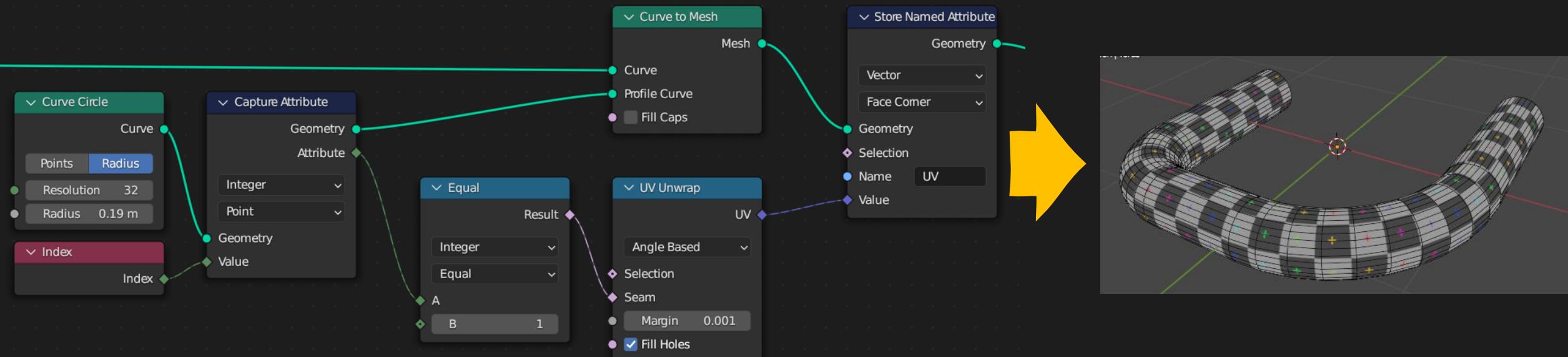
지오메트리 노드로 밧줄을 만고 UV를 따라 텍스쳐링하기



커브의 UV

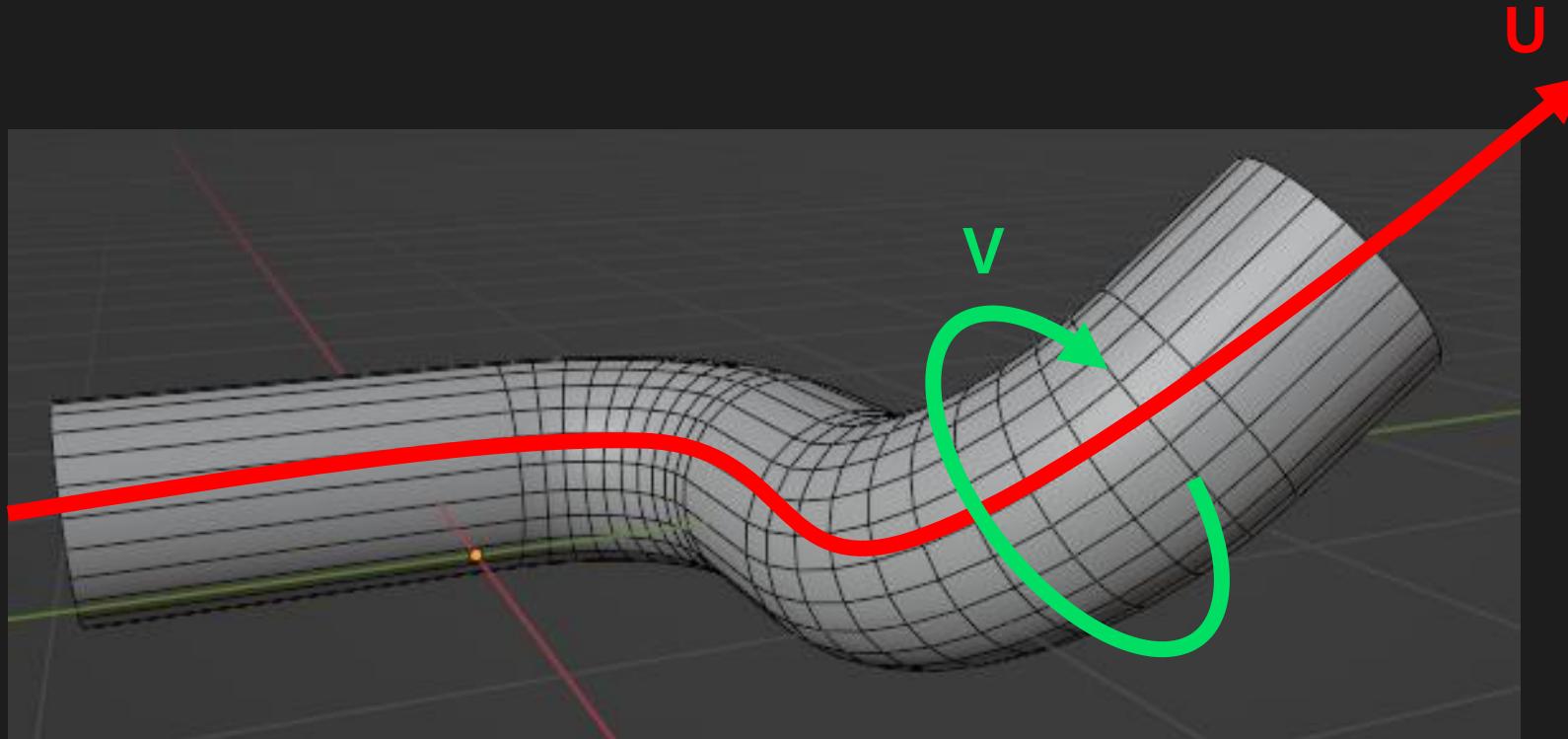
Curve to Mesh를 사용할 때 UV는 생성되지 않습니다.

UV Unwrap 노드가 있지만 조금 무겁고,
또한 커브 형태를 완전히 따라가게 만들기 힘듭니다.



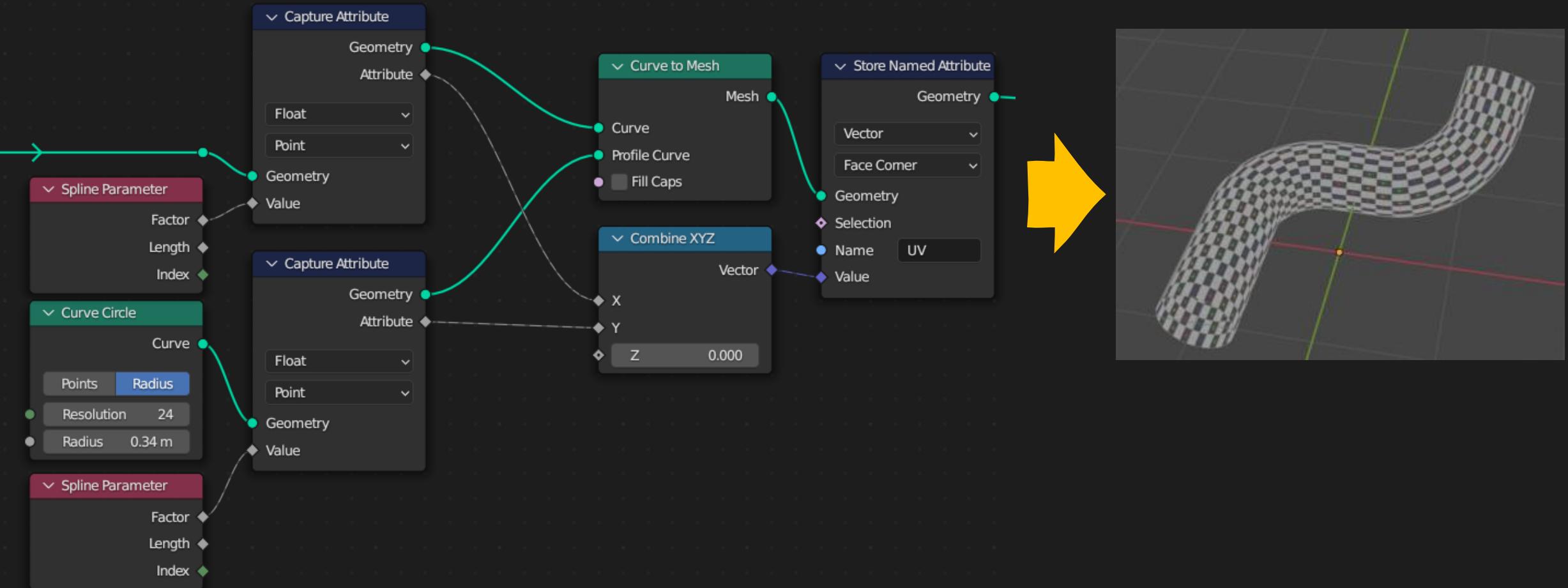
UV의 방향

이상적인 커브의 UV는 커브의 두 방향을 따라야 할 것입니다.

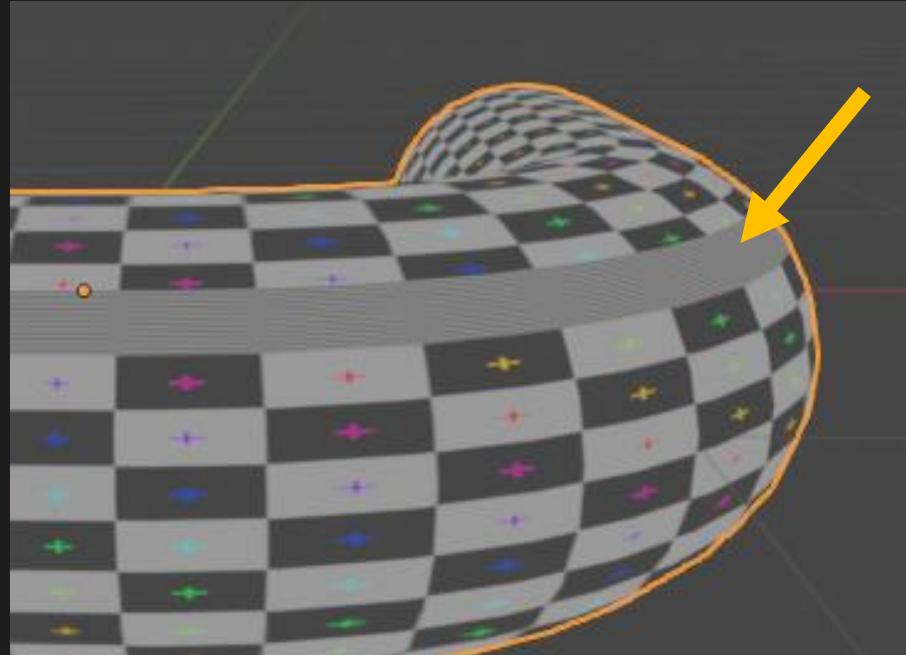


UV의 방향

커브와 Profile Curve의 Factor를 U,V로 사용하면 잘 작동하는 것처럼 보입니다...



도메인 문제

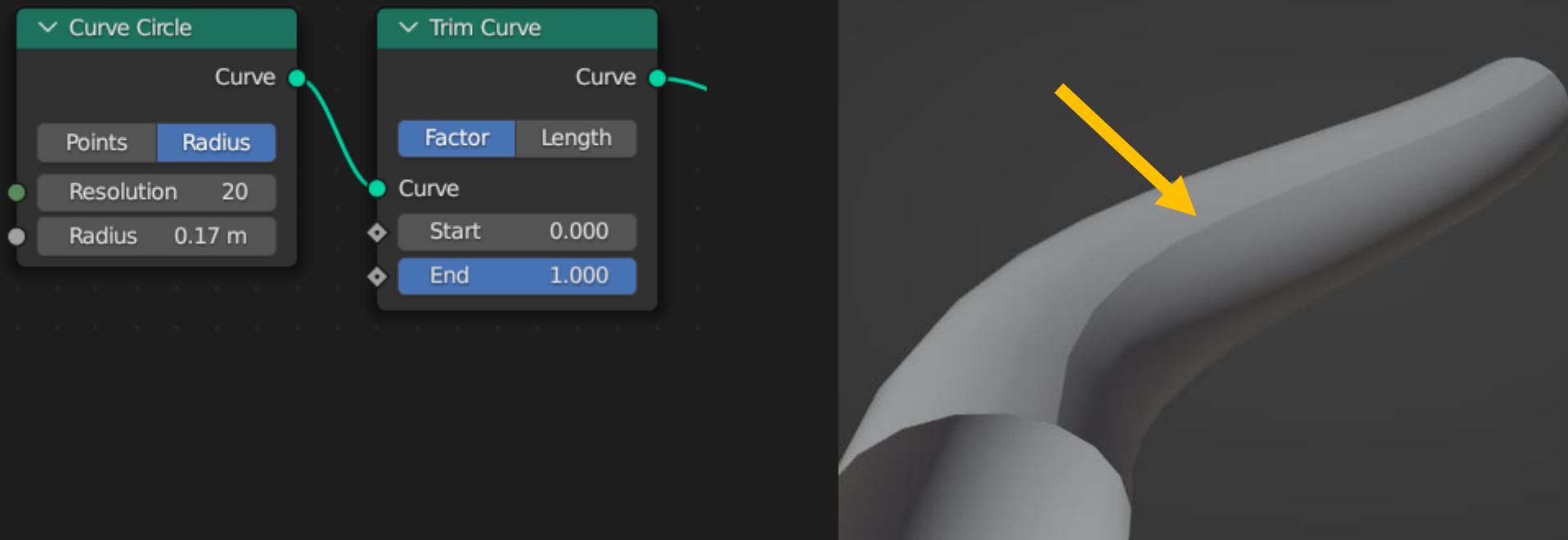


그러나 자세히 살펴보면 왜곡된 부분이 생기는 것을 볼 수 있습니다.

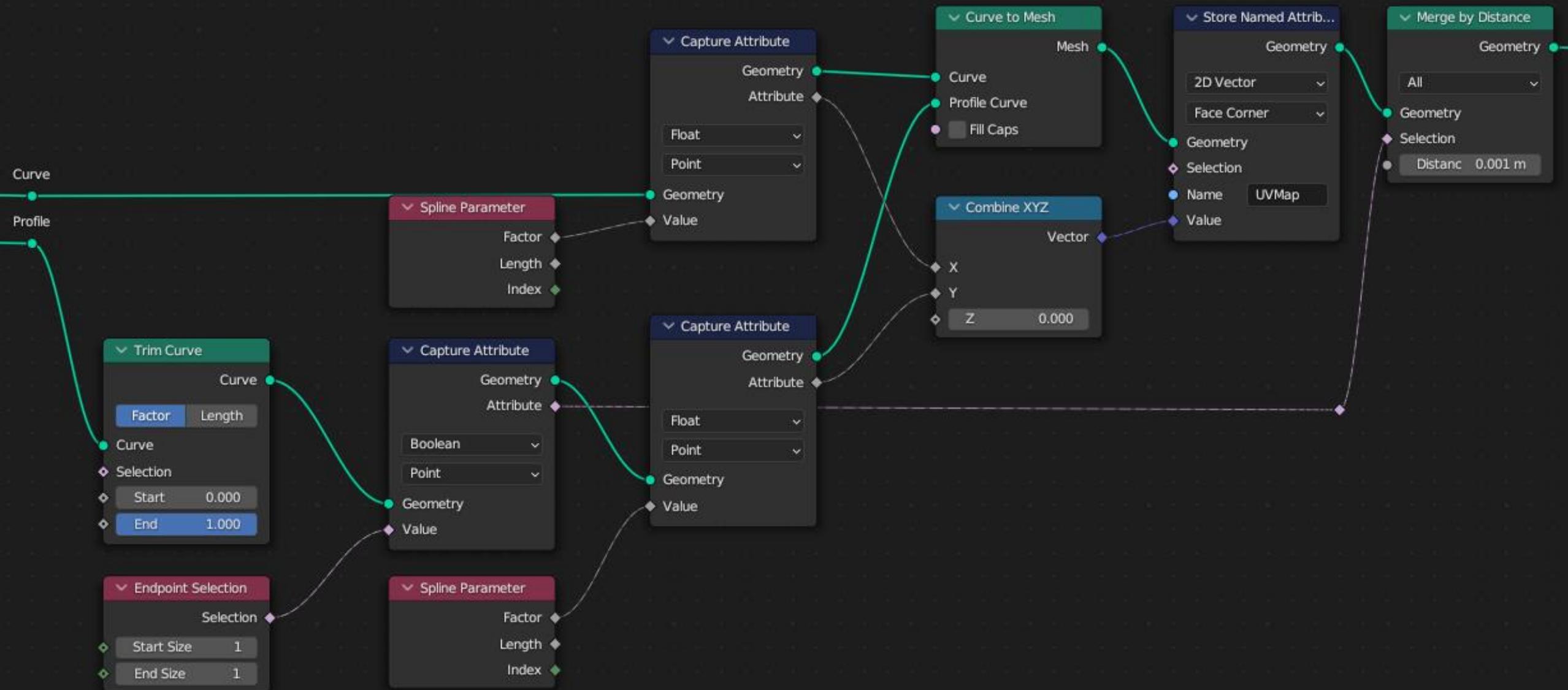
이것은 Profile Curve의 Factor가 시작점과 끝점의 연결부위에서
자동으로 Interpolate 되기 때문입니다.

도메인 문제

Trim Curve 가 Cyclic이 아니게 끝점을 분리하면서도 끝부분의 선분을 제거하지 않으므로,
현재 상황에 유용하게 사용할 수 있습니다.
분리된 시작점과 끝점은 맨 마지막에 Merge by Distance로 합치면 됩니다.

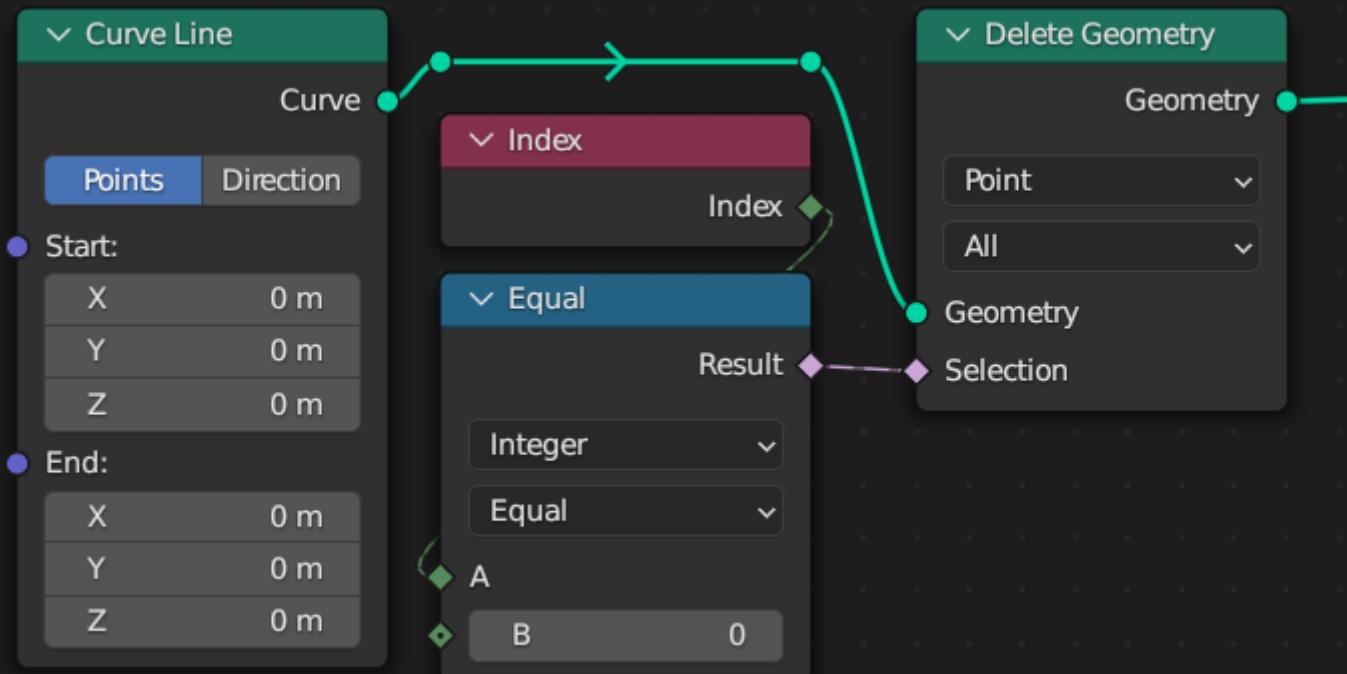


UV Maker



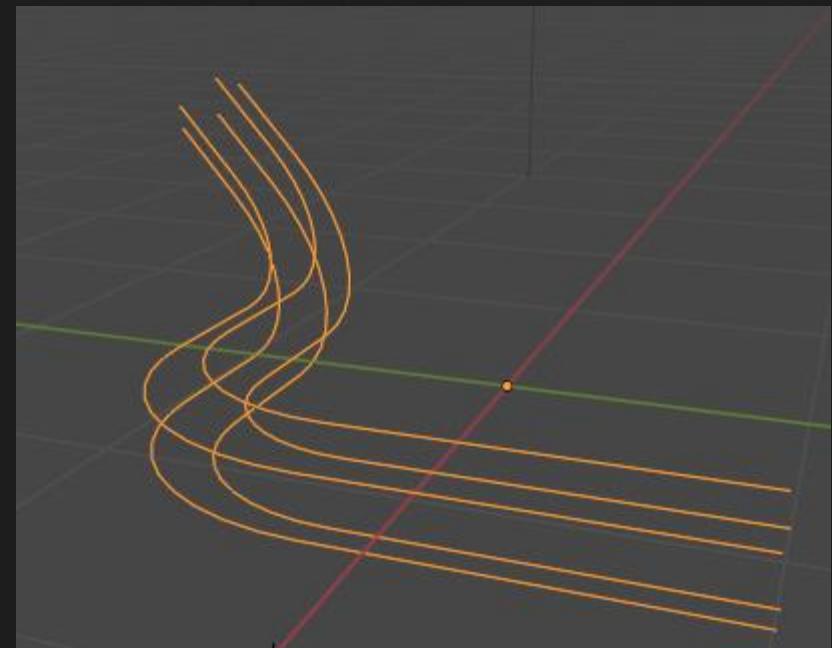
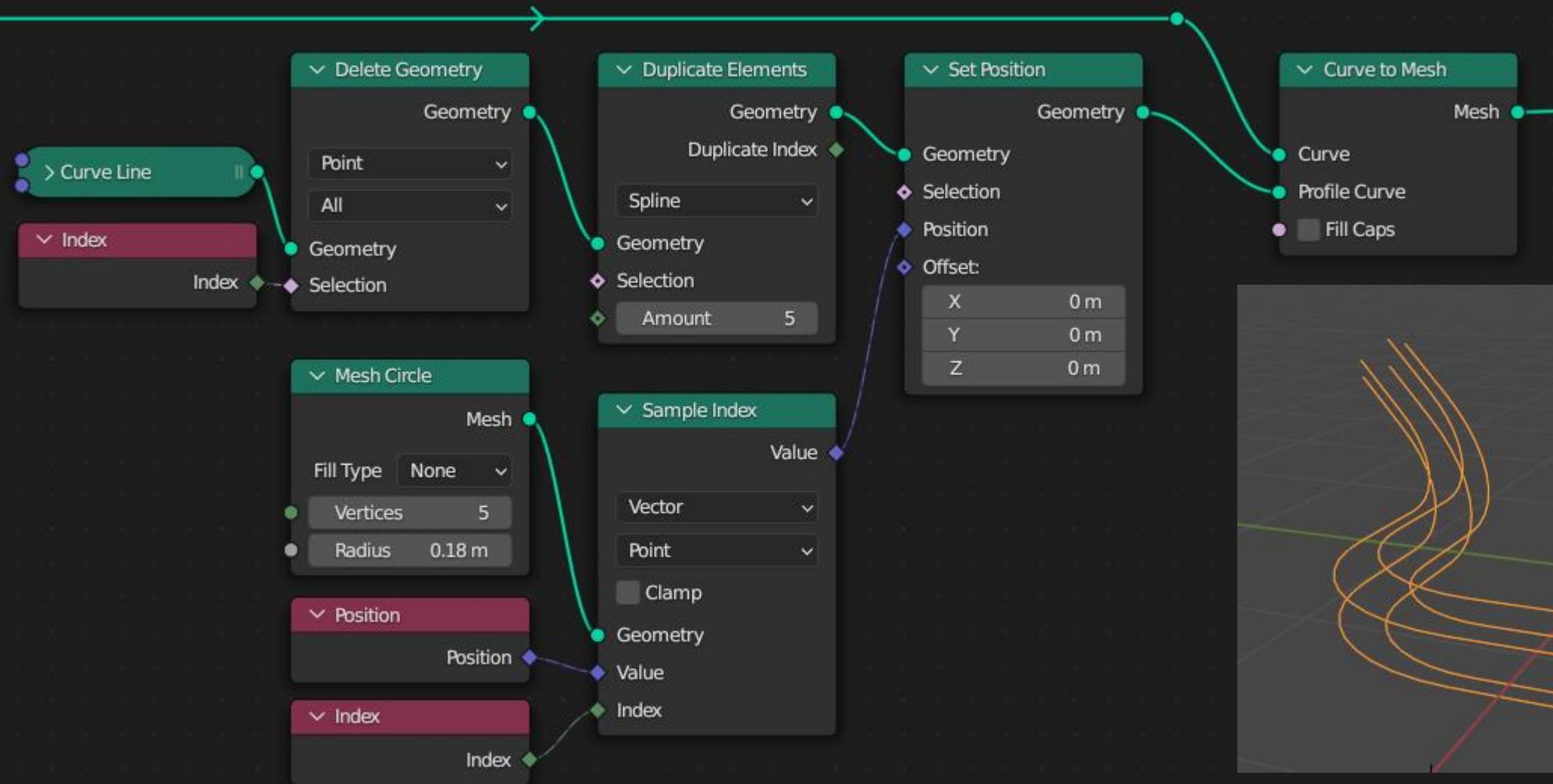
Appendix

점을 커브로 변환하면 아무것도 생성되지 않습니다.
하지만 Curve Line에서 한 점을 제거하면 점 하나만 있는 커브가 생성됩니다.



Appendix

Profile Curve로 사용하면....



047강 Strings (1) - 3D 타이포그래피

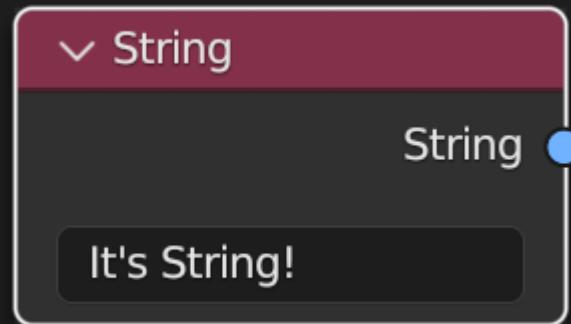
String 데이터타입을 사용하는 법
String을 커브로 만들어 3D 타이포그래피 만들기



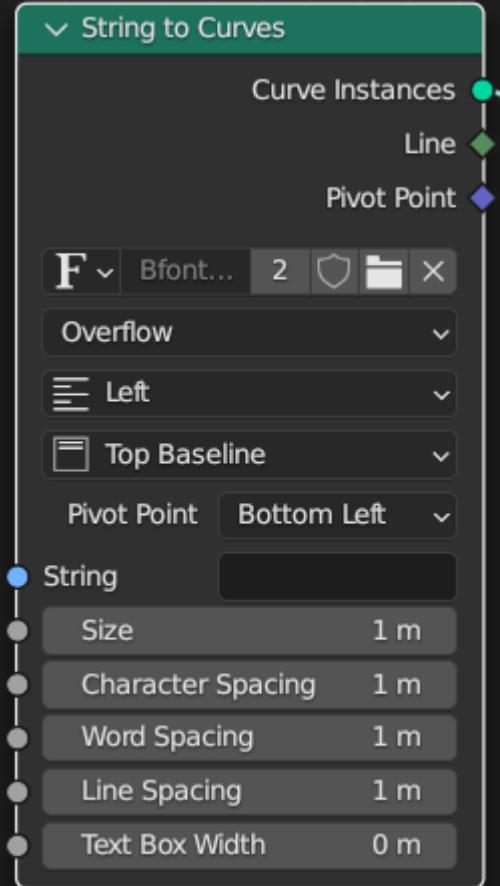
Strings

String은 문장을 다루는 지오메트리 노드의 데이터타입 중 하나입니다.

일반적으로 Named Attribute 등을 불러올 때 같은 제한적인 용도로만 사용되지만,
커브로 내보낼 수 있기 때문에 또다른 용도로도 사용 가능합니다.



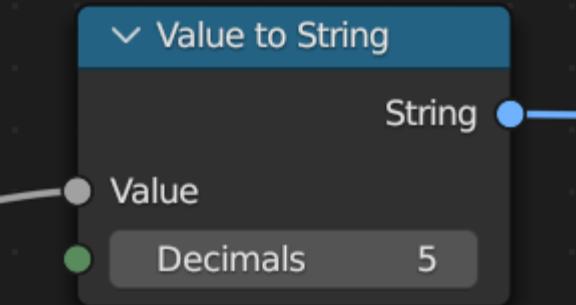
String to Curves



String을 Curve Instance로 변환해줍니다.

String to Curves는 String을 여러 개의 커브 인스턴스로 출력합니다.
지오메트리에 접근하기 위해서는 나중에 Realize Instance를 해 주어야 합니다.

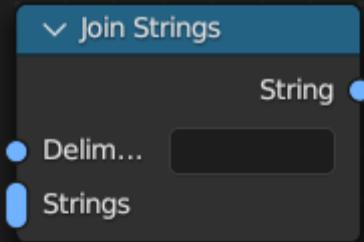
Value to String



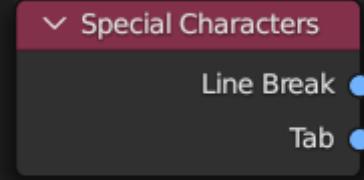
Float 형식은 String으로 바로 사용할 수 없고 Value to String으로 변환하여야 합니다.

안타깝지만 단일값만 사용 가능하므로, 다이아몬드 소켓은 사용할 수 없습니다.

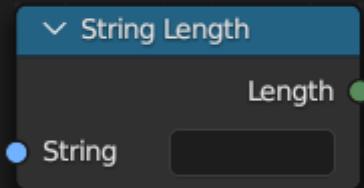
String 컨트롤



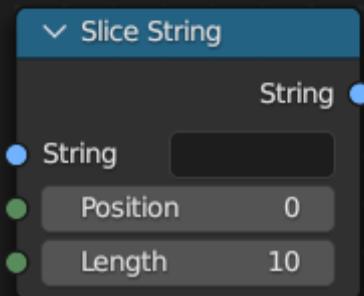
Join Strings : Join Geometry와 비슷하게 String을 차례대로 결합합니다.
Delimiter의 문자는 각 String의 사이에 추가됩니다.



Special Characters : 줄바꿈을 하거나, 글자 간격을 크게 띄우는 기능입니다.
String취급이므로 Join Strings와 함께 사용할 수 있습니다.



String Length : 글자수 계산.

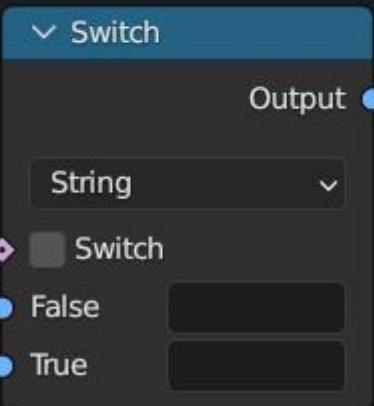


Slice String : 문자의 특정 부분만 남깁니다.

String 컨트롤



Replace String : 말 그대로 찾아 바꾸기.

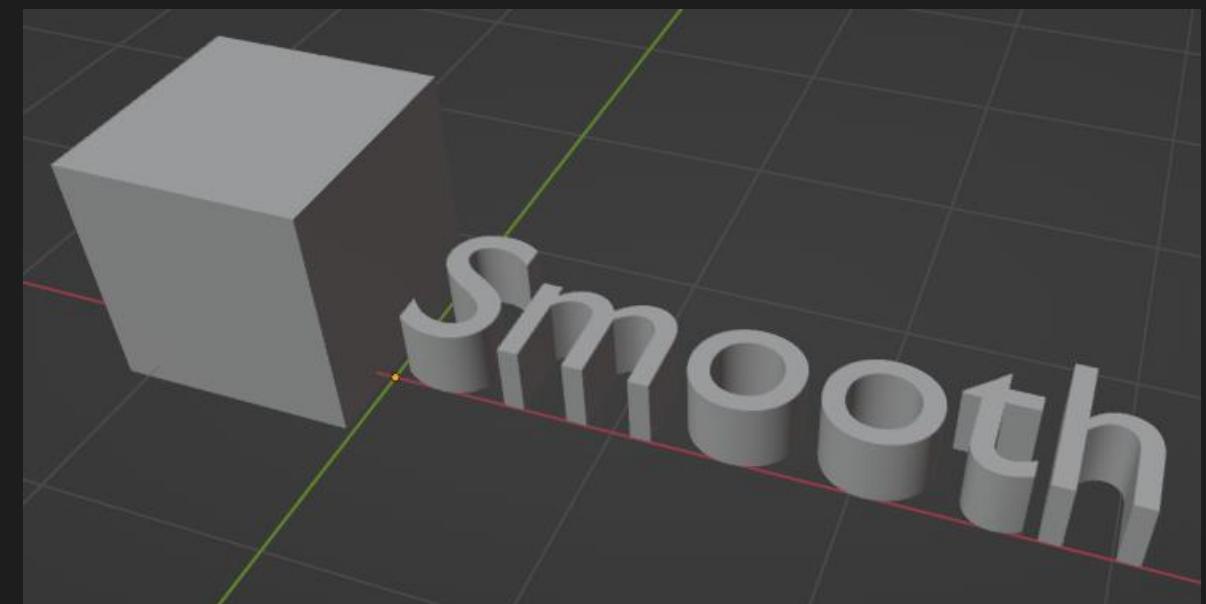
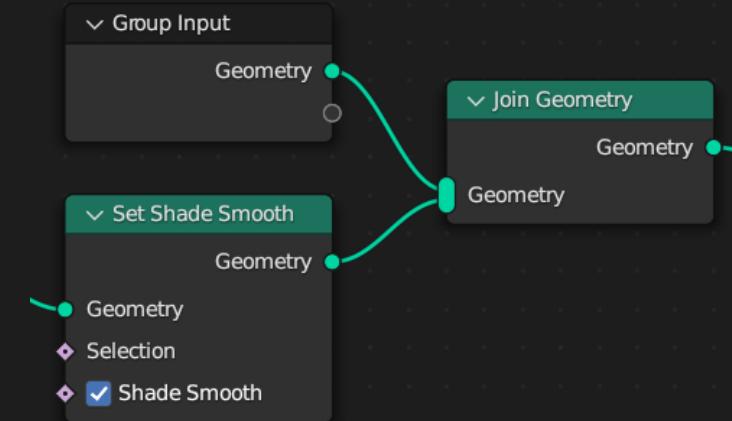
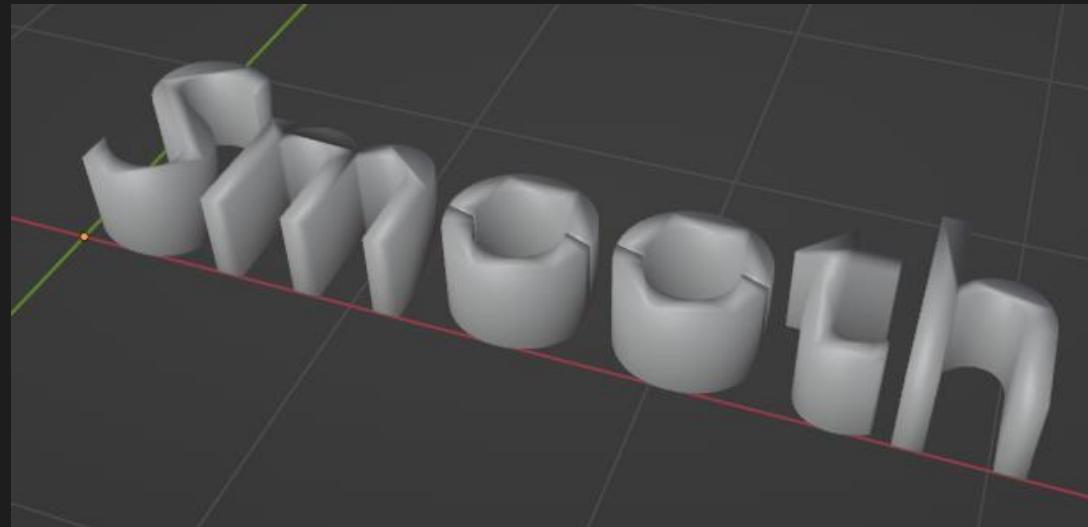
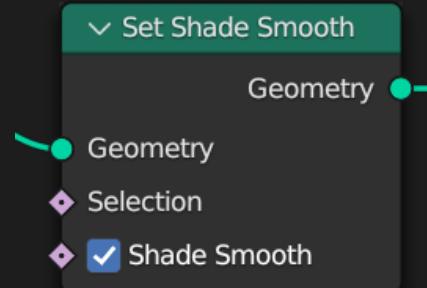


Switch에서 String도 사용할 수 있습니다.

Shade Auto Smooth?

3.5 기준으로 지오메트리 노드는 Auto Smooth를 사용할 수 없습니다.

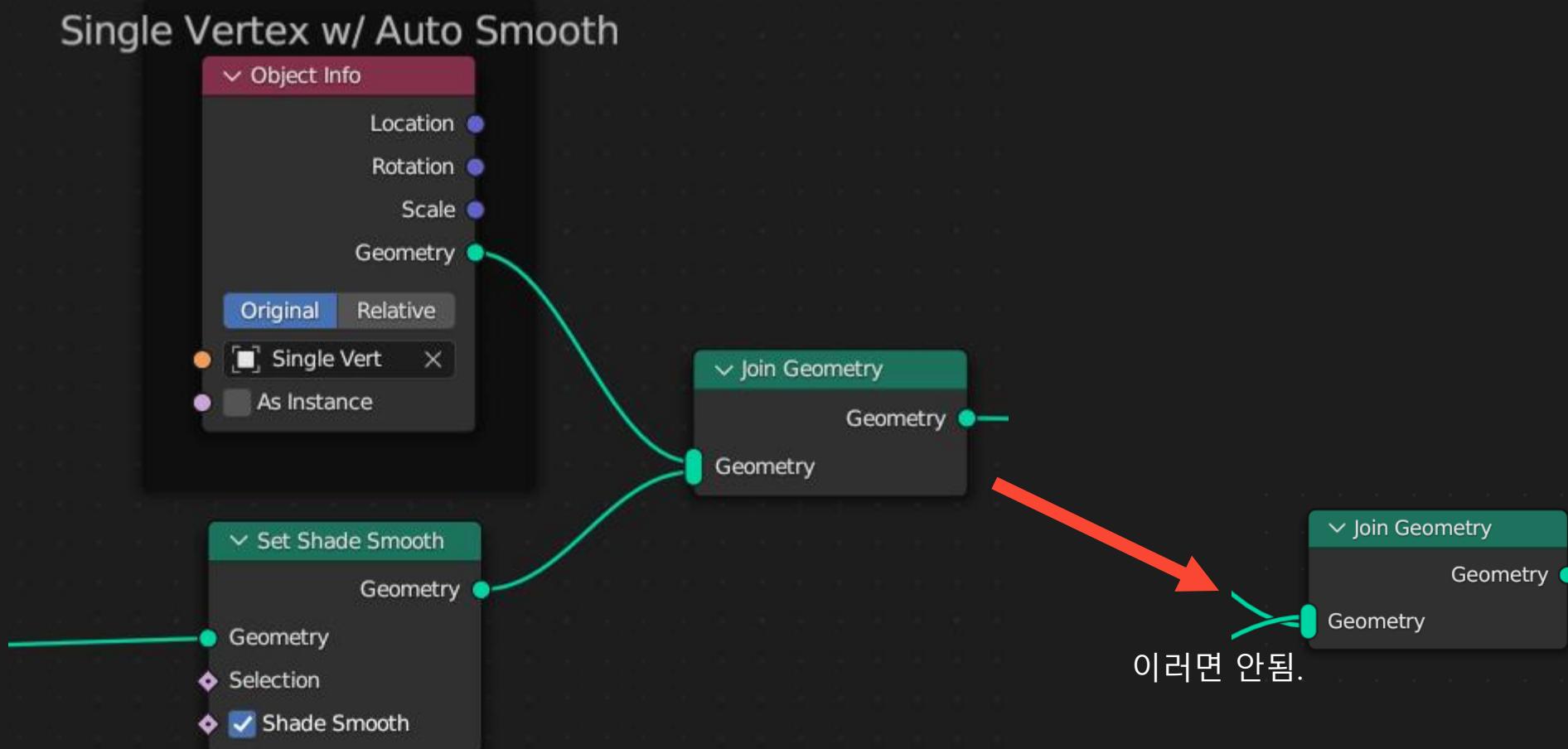
오로지 Auto Smooth인 메쉬가 결합되어있는 상태여야만 사용 가능합니다...



Shade Auto Smooth?

Auto Smooth의 작동기준은 'Join Geometry의 최상위에서 Auto Smooth를 쓸 것' 이라서,
Auto Smooth를 캔들 수 있는 지오메트리를 사용해야만 합니다.

이후 버전에서는 해결되길 바라면서, 일단 아래와 같이, **점 하나만 있는 auto smooth 오브젝트**를 사용하는 우회법을 써봅시다.



048-049강 Strings (2) - 2D 타이포그래피

String을 순차적으로 움직이는 애니메이션

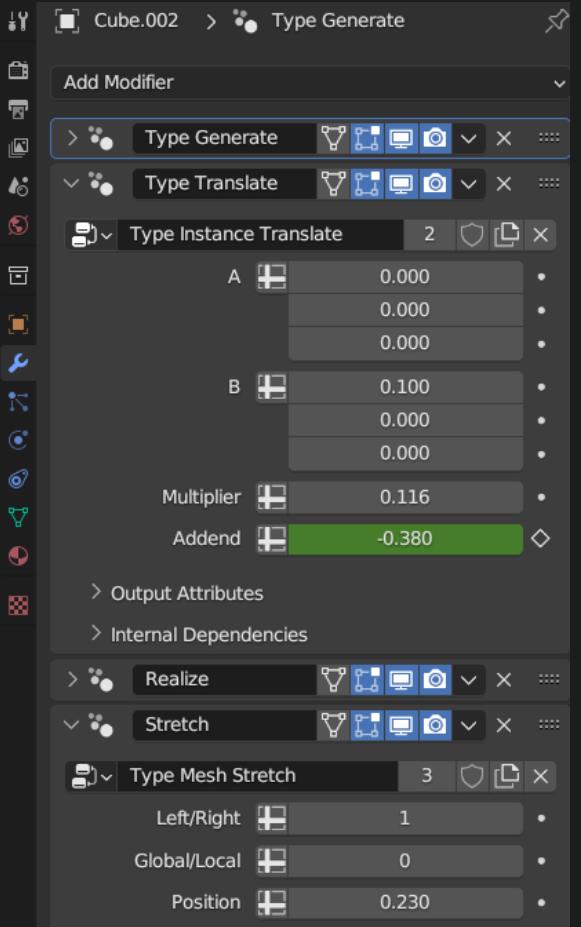
키네틱 타이포그래피를 위한 노드그룹 작성

지오메트리 노드를 스택으로 쌓아 활용하는 법

CREATE
YOUR
OWN
PATH

Kinetic Typography

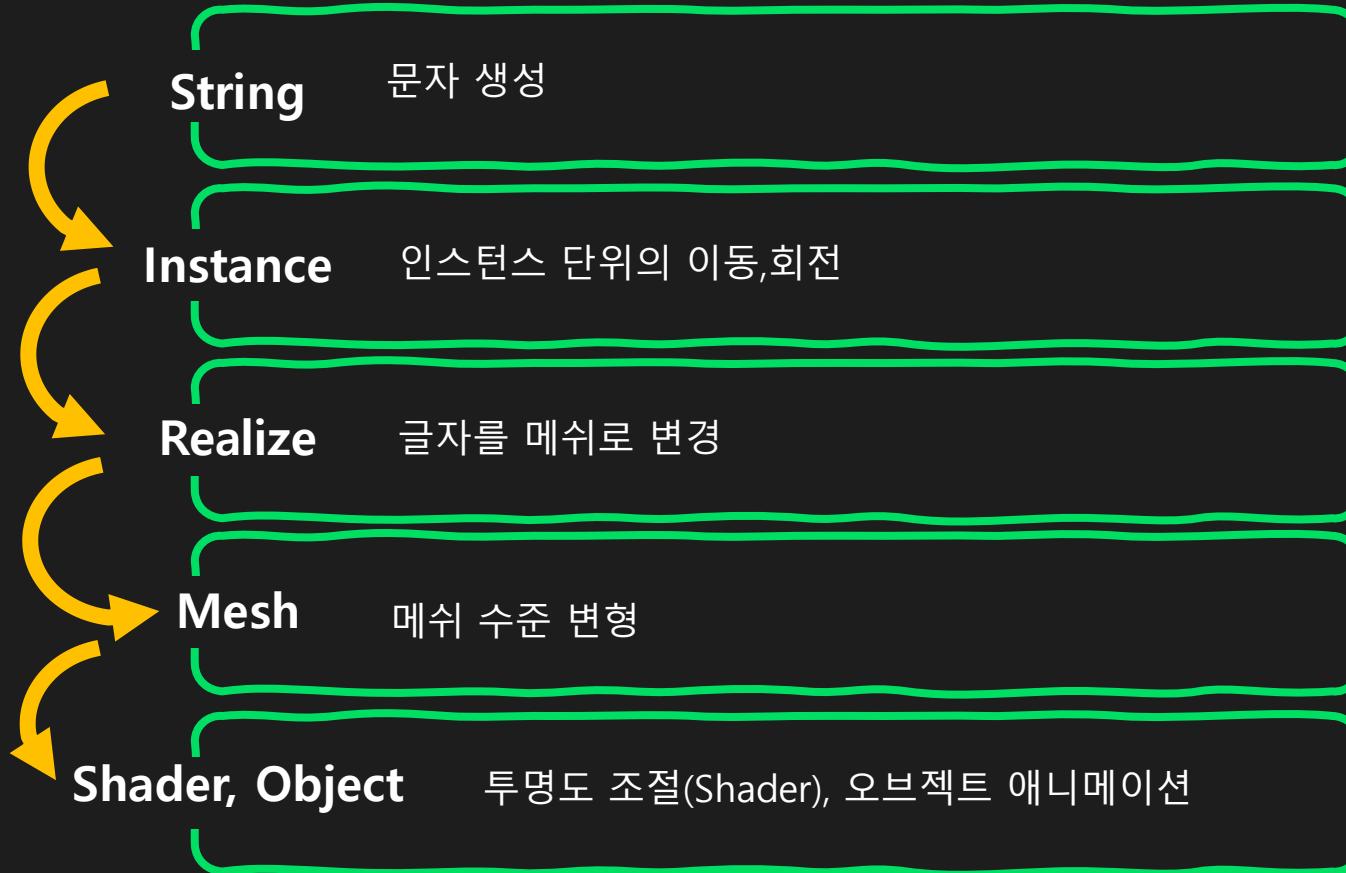
지오메트리 노드트리를 매번 처음부터 만들 수도 있겠지만,
반복 작업을 분석하여 미리 몇 개의 노드그룹으로 만들어 두고,
그것들을 조합하여 사용한다면 더 편리합니다.



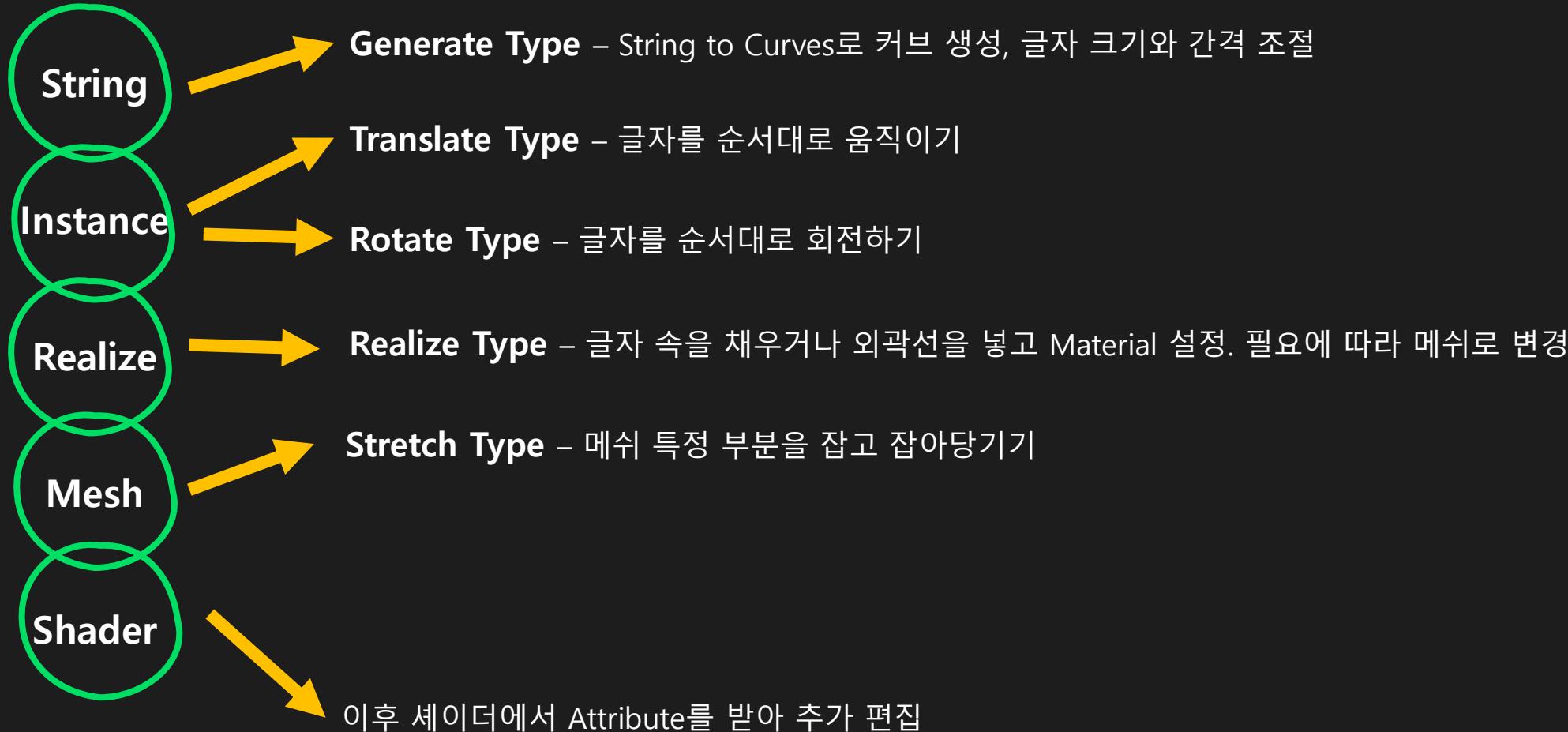
이와 같이 모디파이어 스택에
노드그룹을 쌓아 만들어 봅시다.



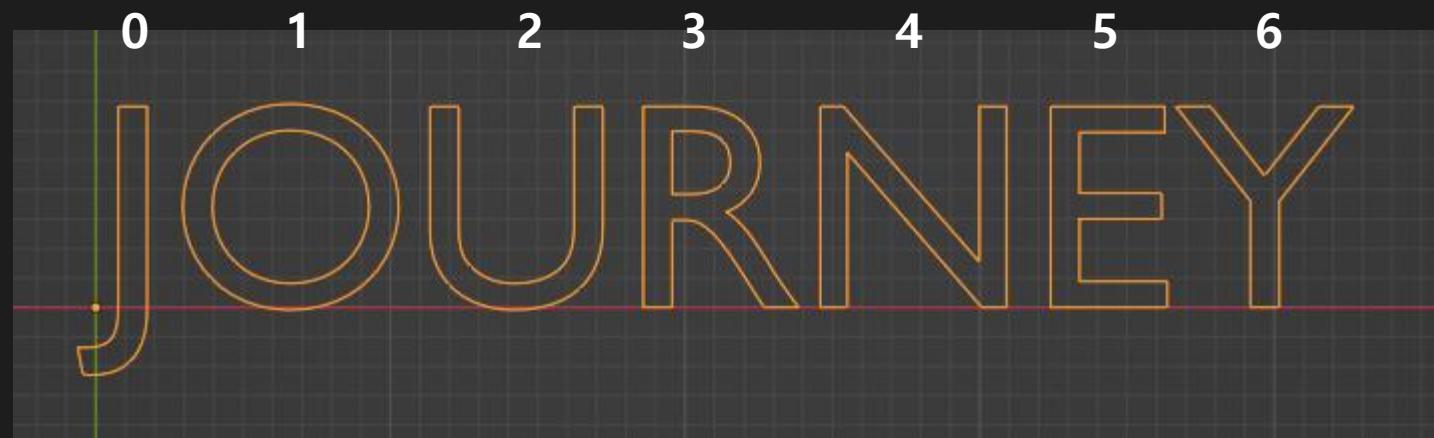
애니메이션 단계별 분류



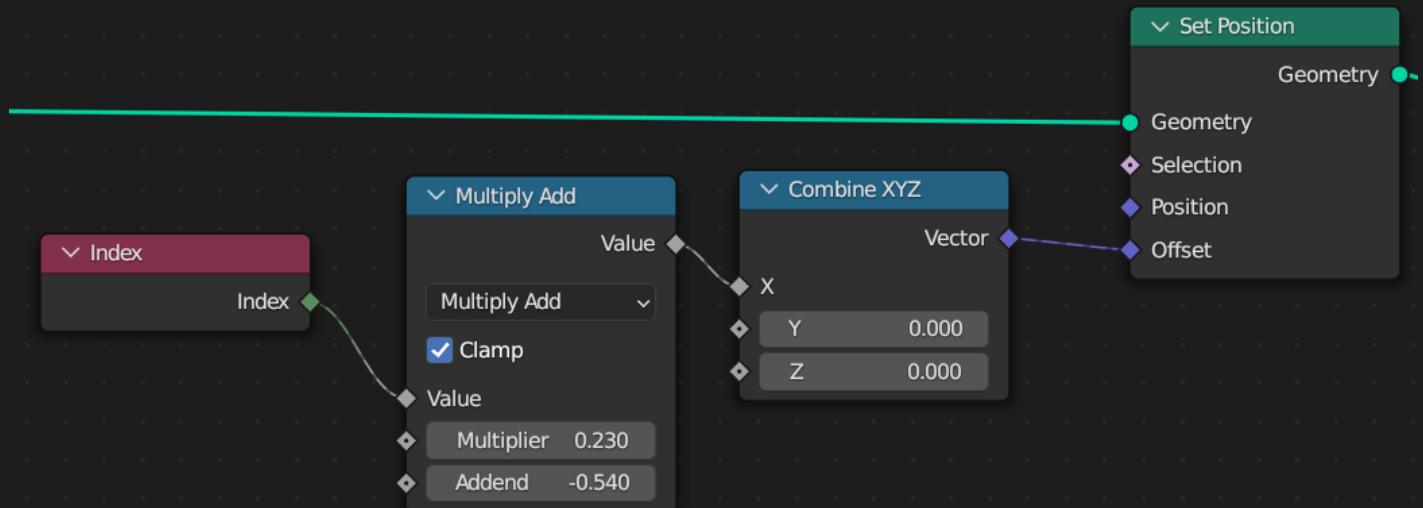
노드그룹 계획



Instance Translate/Rotation

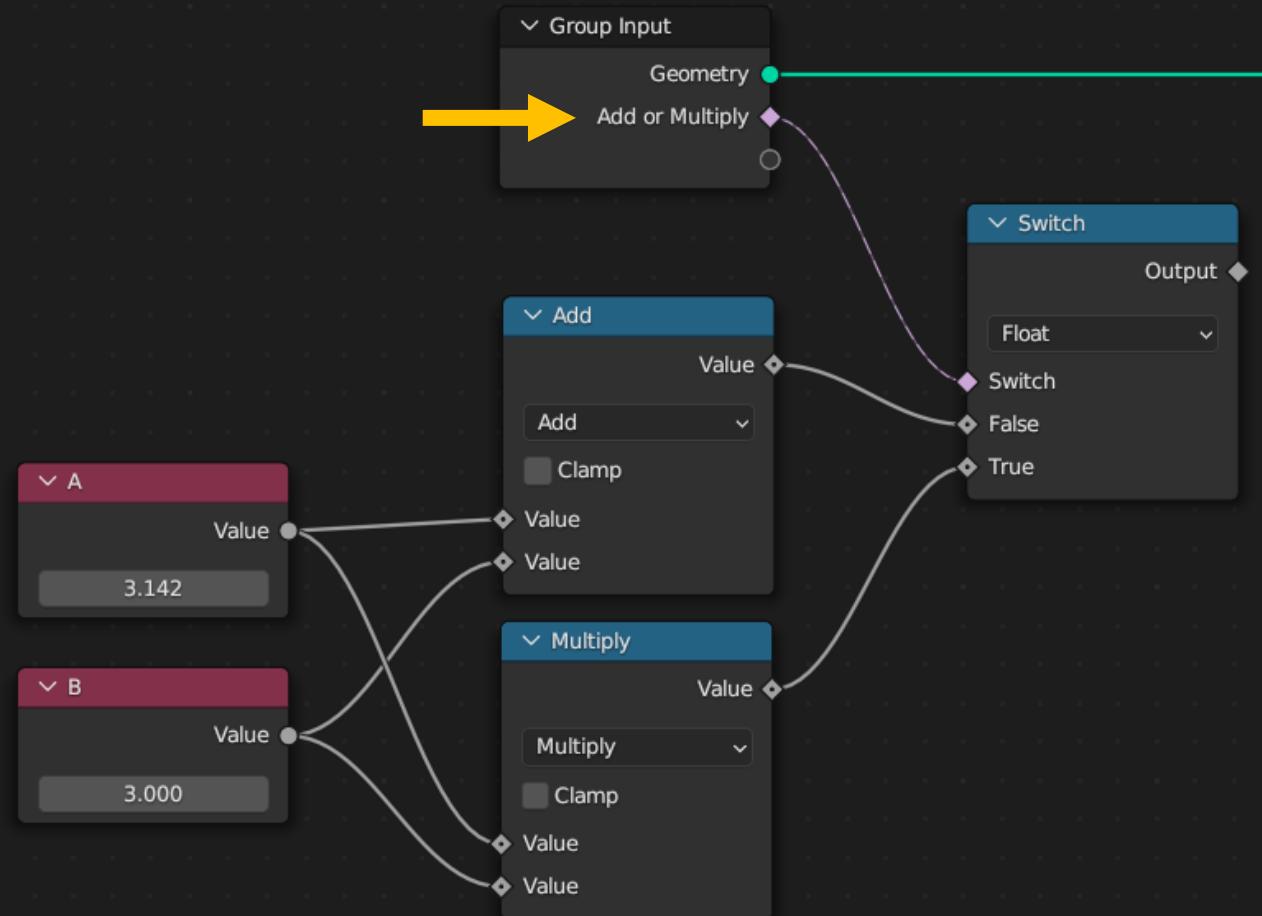
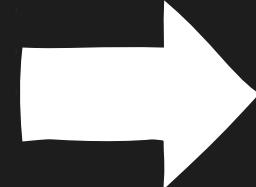
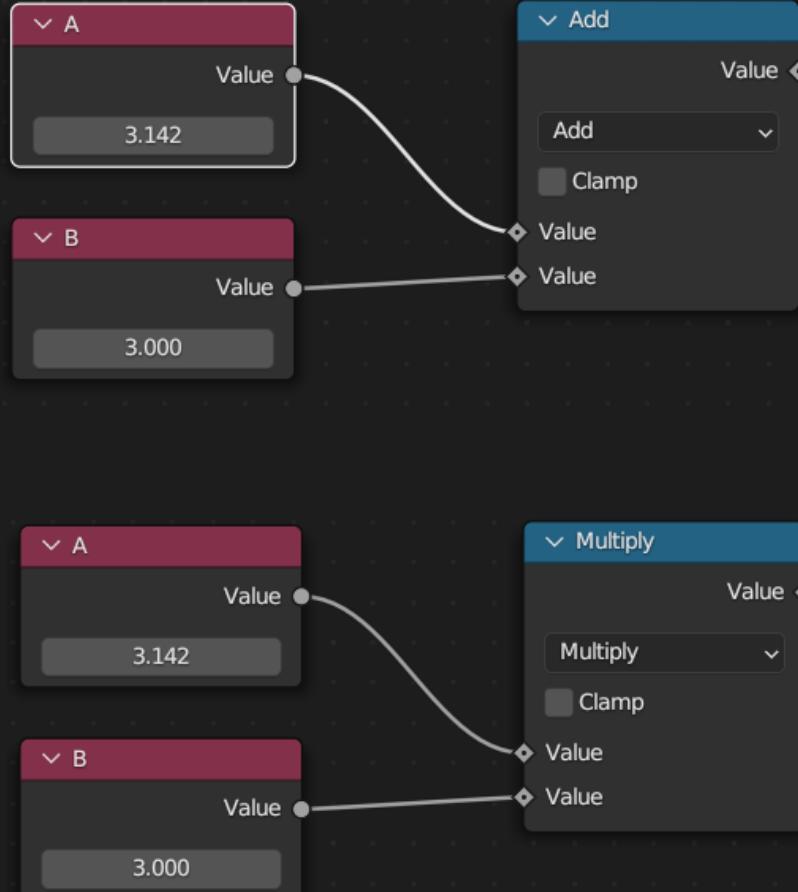


String to Curves가 만드는 인덱스를 이용하여 문자를 순서대로 움직일 수 있습니다.



Group Input에 연결할 수 없는 경우

소켓이 없는 경우 (예를 들어, Math 노드의 모드 등)
Switch 노드를 활용하면 Group Input에 꽂을 수 있습니다.



3개 이상을 switch로 선택하려면

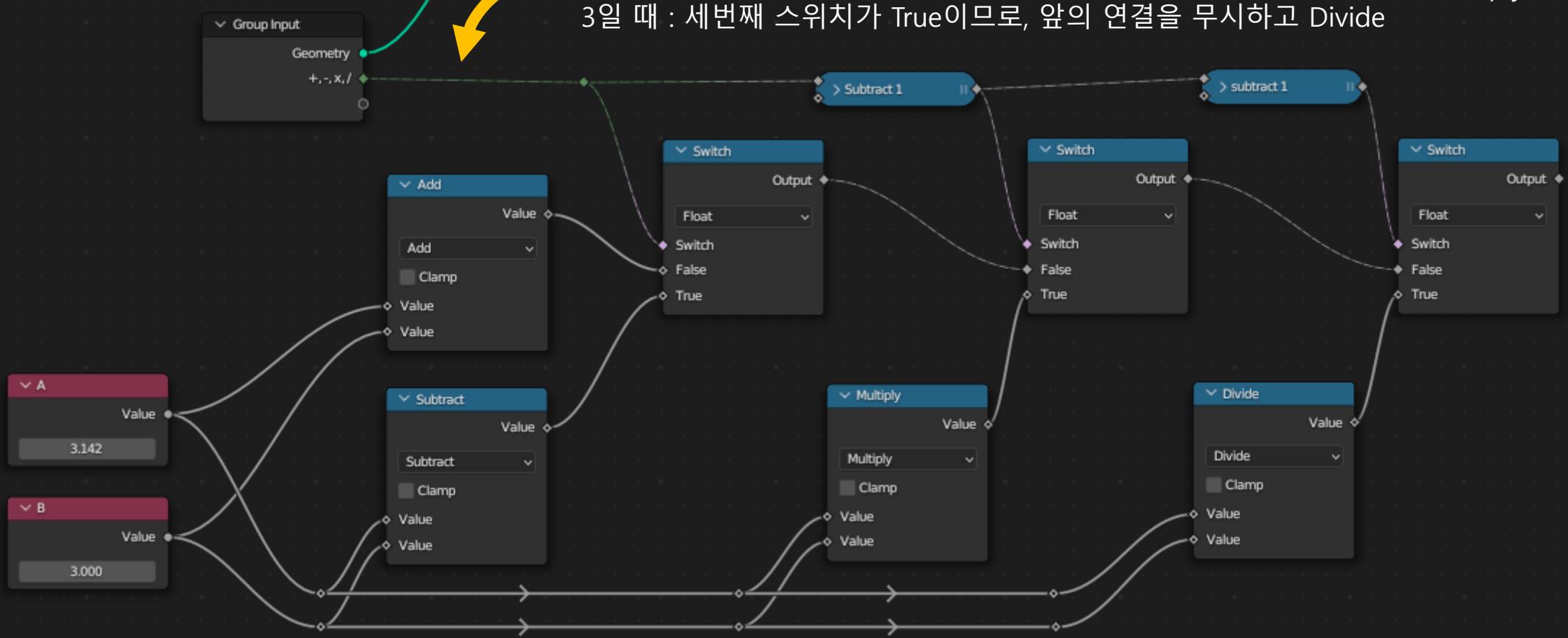
Switch를 연속으로 연결하고, 소켓에 정수를 연결합니다.

0일 때 : 모든 스위치가 False 이므로 Add가 됩니다.

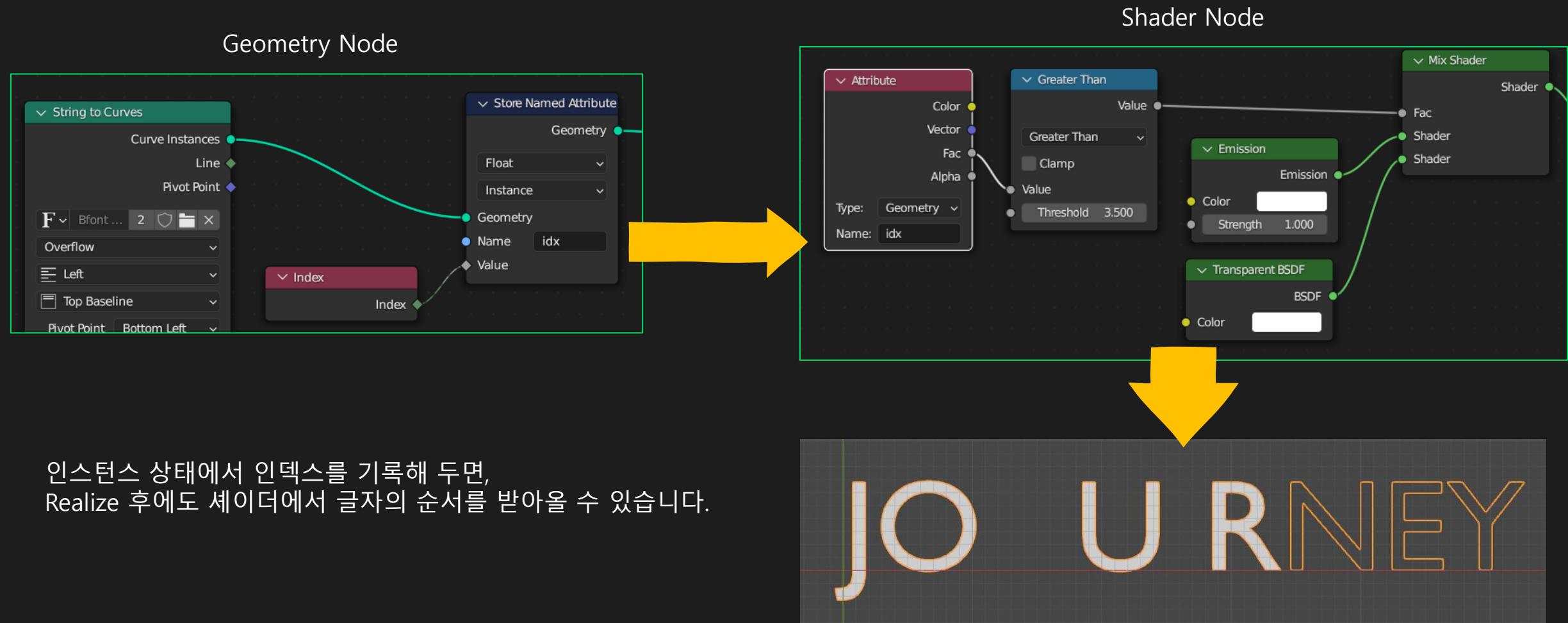
1일 때 : 첫번째 스위치만 True이므로, Subtract가 됩니다.

2일 때 : 두번째 스위치가 True이므로, 첫번째 스위치의 값을 무시하고 Multiply

3일 때 : 세번째 스위치가 True이므로, 앞의 연결을 무시하고 Divide



셰이더 연동

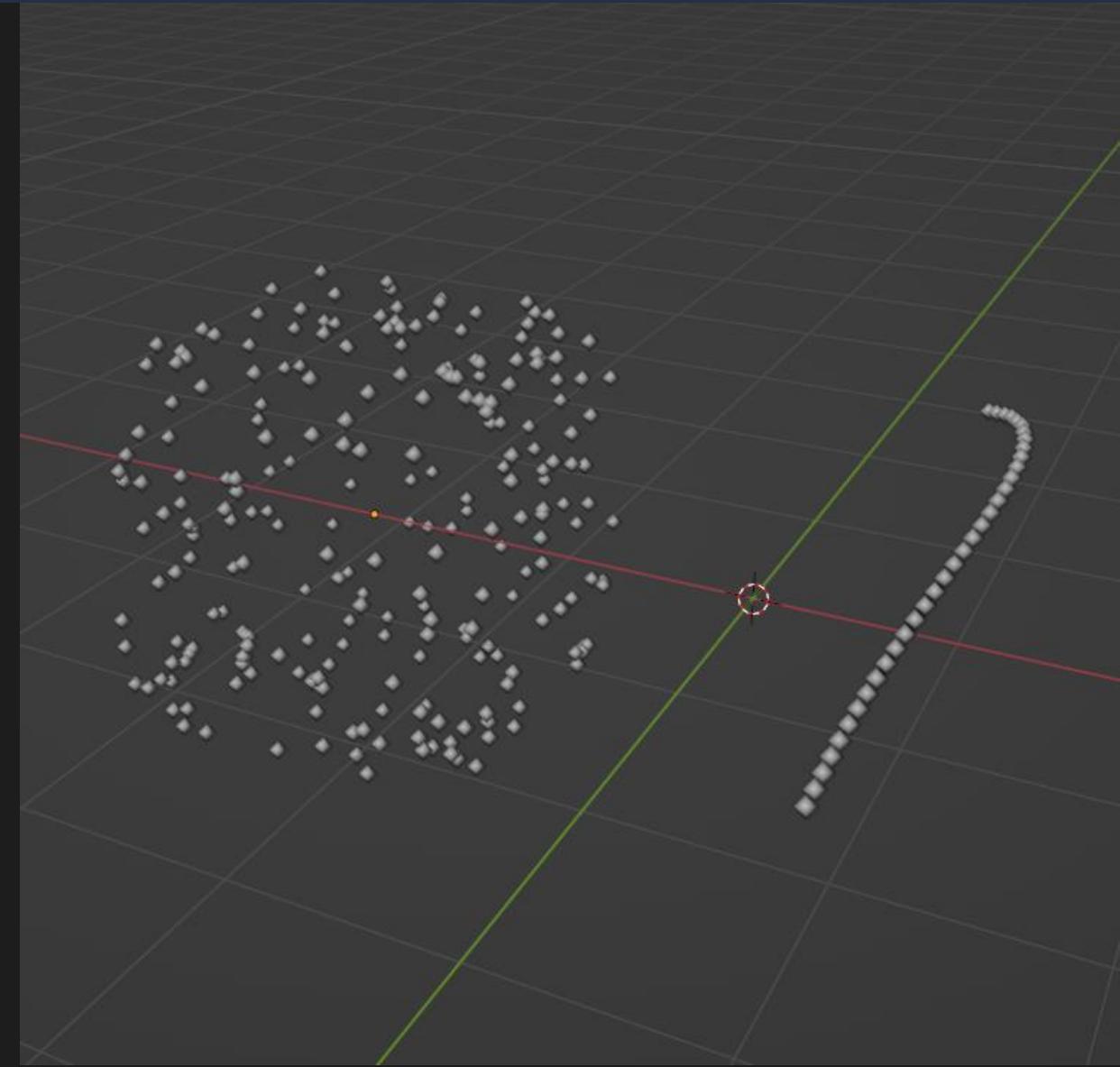


050강 Point Cloud

포인트클라우드의 개념

포인트 애니메이션

포인트클라우드가 사용되는 분야 살펴보기

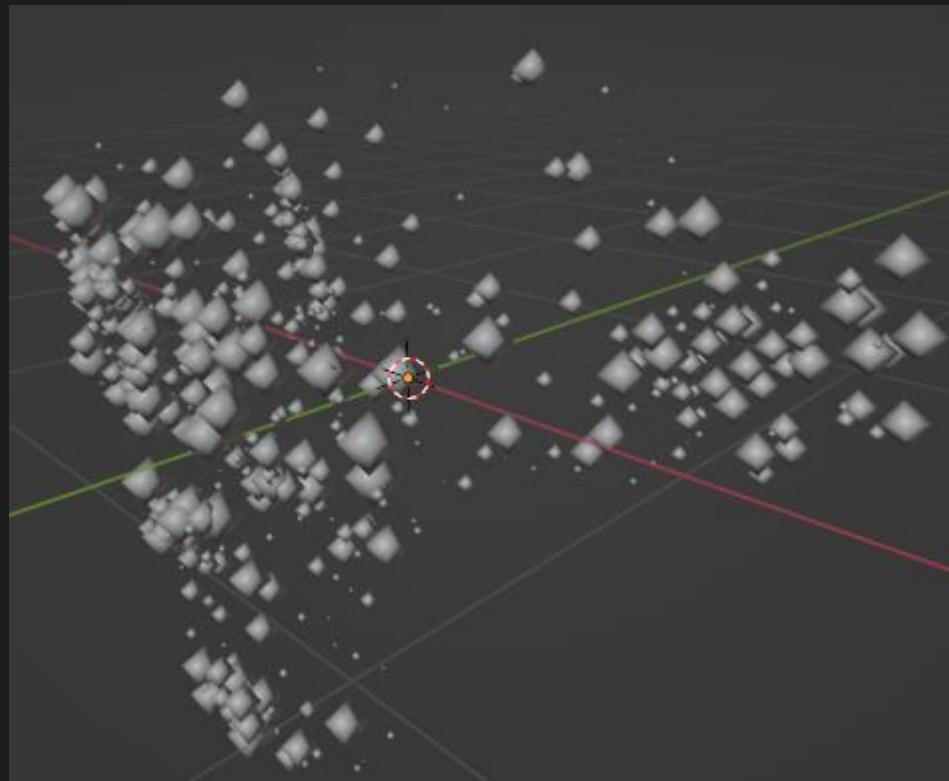


Point Cloud

개념

Point Cloud는 포인트 정보만 담는 지오메트리로, Vertex와는 다른 개념입니다. 뷰포트에서 마름모 모양으로 표시됩니다.

Point Cloud는 Radius 정보를 담을 수 있습니다.



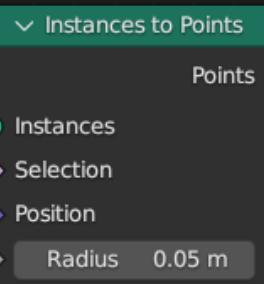
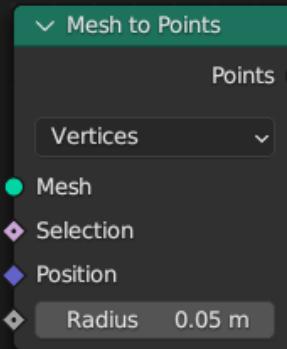
		position	radius
Face Corner	0	-0.352 -0.617 -0.023	0.050
Curve	1	0.352 -0.617 -0.023	0.050
Control Point	2	-0.547 -0.578 0.055	0.050
Spline	3	0.547 -0.578 0.055	0.050
Point Cloud	4	-0.500 -0.688 0.094	0.050
Point	507	0.500 -0.688 0.094	0.050
Volume Grids	6	-0.438 -0.766 0.164	0.050
Instances	7	0.438 -0.766 0.164	0.050
	8	0.352 -0.719 0.031	0.050
	9	-0.352 -0.719 0.031	0.050
	10	0.352 -0.781 0.133	0.050
	11	-0.352 -0.781 0.133	0.050

Point Cloud

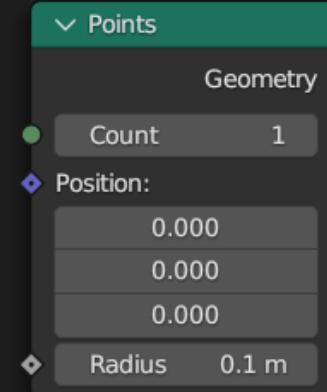
Point Cloud는 vertex와는 다른 개념이지만, 컨트롤은 다른 타입의 포인트 (vertex, control point)와 동일합니다. 예컨대, Capture Attribute에서 Point는 vertex, control point, point cloud를 모두 포함하는 개념입니다.

Mesh to Points : 점, 선, 또는 면을 포인트로 바꿉니다.

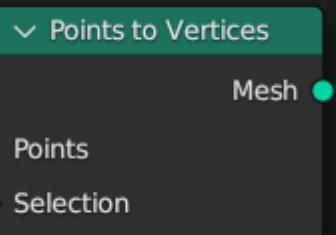
Instances to Points : 인스턴스의 위치를 포인트로 바꿉니다.



Points 노드로
포인트를 직접 생성할 수도 있습니다.

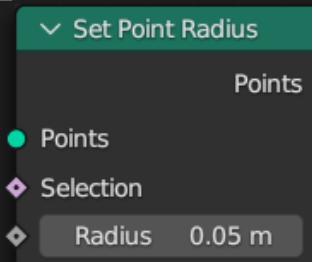
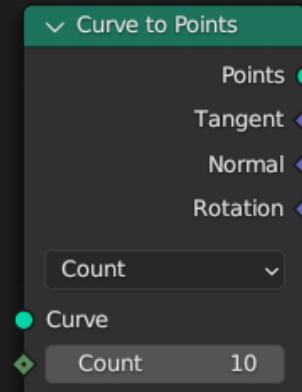


Point to Vertices로
포인트에서 메쉬로의 변환도 가능합니다.



Curve to Points

Curve to Points는 컨트롤 포인트와 별개로
점 개수를 지정할 수 있어 편리합니다.



Set Point Radius는 포인트의 반지름을 조절합니다.

Point Cloud

렌더링

포인트클라우드는 Eevee에서 렌더링할 수 없습니다. (3.5기준)

하지만 Cycles에서는 완전한 구형으로 보이게 됩니다.
구체 파티클을 만들고 싶다면 pointcloud를 사용하는 것이
다른 방식보다 훨씬 가볍습니다.

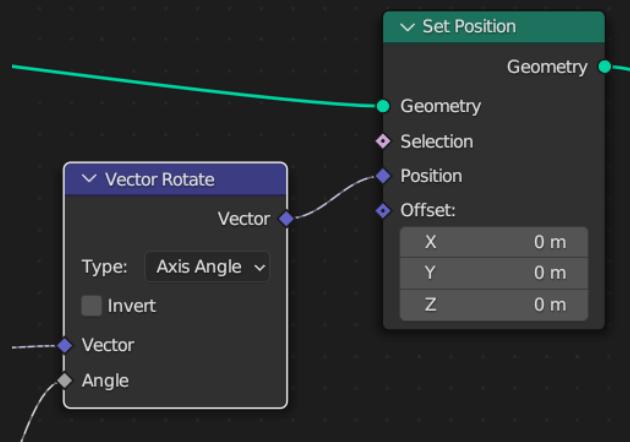


- Position ● **셰이더**에서 포인트의 정보를 불러오는 Point Info 노드가 있습니다.
- Radius ● Object Info 노드처럼, 포인트 각각의 위치, 그리고 각각의 랜덤값을 만들어 줍니다.
- Random ●

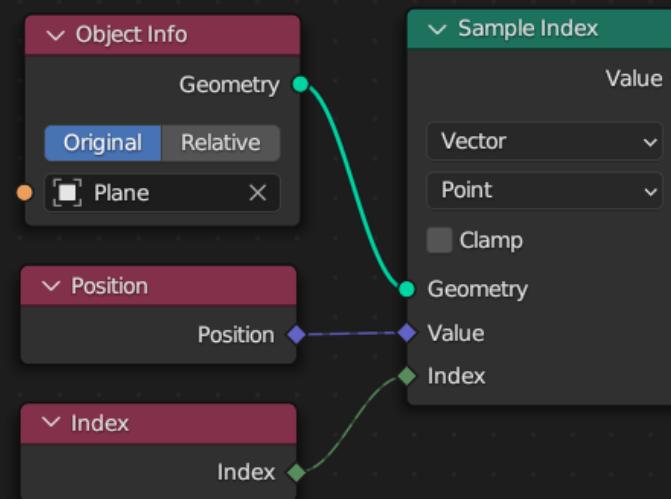
포인트 클라우드 애니메이션

파티클 시뮬레이션처럼 물리법칙을 따르게 할 순 없지만, 구체적으로 위치를 지정할 수 있습니다.

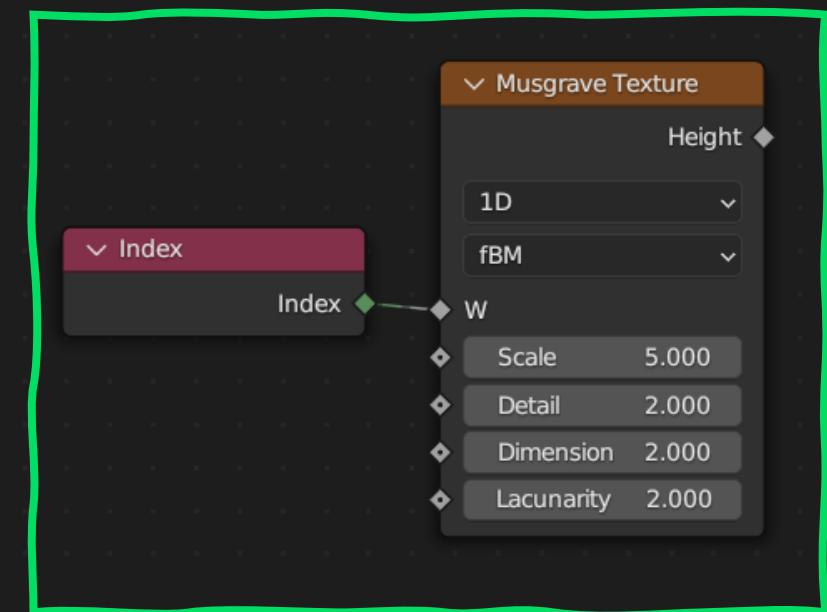
직접 위치 지정



Sample index



Index를 시드로 하는 Texture



Point Cloud

포인트 클라우드의 본래 용도는 주로 3D 스캐닝 기술을 사용하여 수집된 정보로, 실세계의 물체나 환경을 캡처할 때 사용합니다.

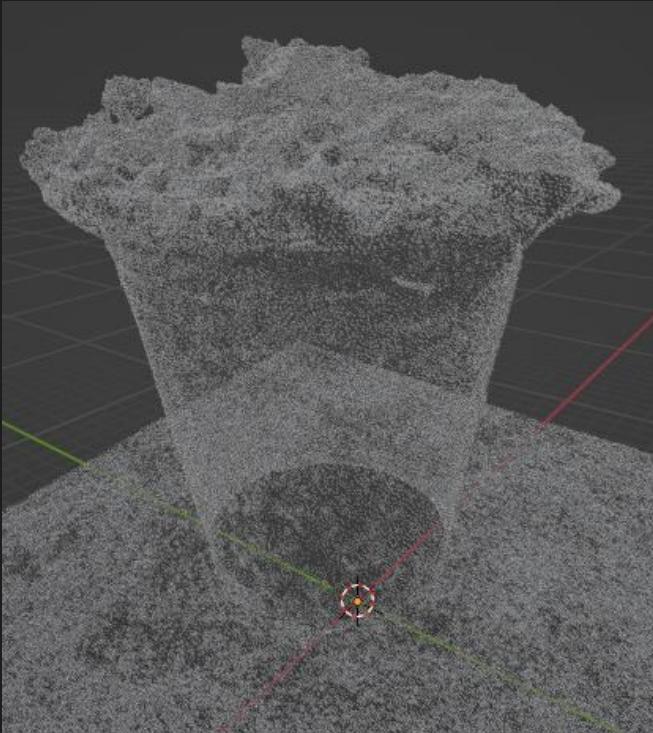


사진 정보를 이용하여 3D 모델을 만드는 것을 photogrammetry라고 합니다.

3D 스캔 프로그램들

Reality Capture (PC)

연산에 PC자원을 사용. 매우 전문적인 컨트롤이 가능하지만, 반대로 말하면 변수에 따라 퀄리티에 기복이 있는 편입니다.
프로그램은 무료로 받을 수 있지만, 3D 모델을 내보낼 때 결제가 필요합니다. (1회당 0.x \$ ~1.x \$)

RealityScan (iOS)

찍음과 동시에 서버에서 연산이 이루어지는 방식입니다.
무료지만 Sketchfab에 업로드되어야 다운로드 가능합니다.

PolyCam (iOS, Android)

움직임을 감지하여 자동으로 셔터를 눌러줍니다. 아이폰 기종에 따라 LiDAR 센서도 활용 가능합니다.
유료 구독방식이며, 구독중에는 홈페이지를 통하여 휴대폰 카메라 말고도 다른 이미지로 3D스캔을 할 수 있습니다.

3D 스캔-친화적인 오브젝트

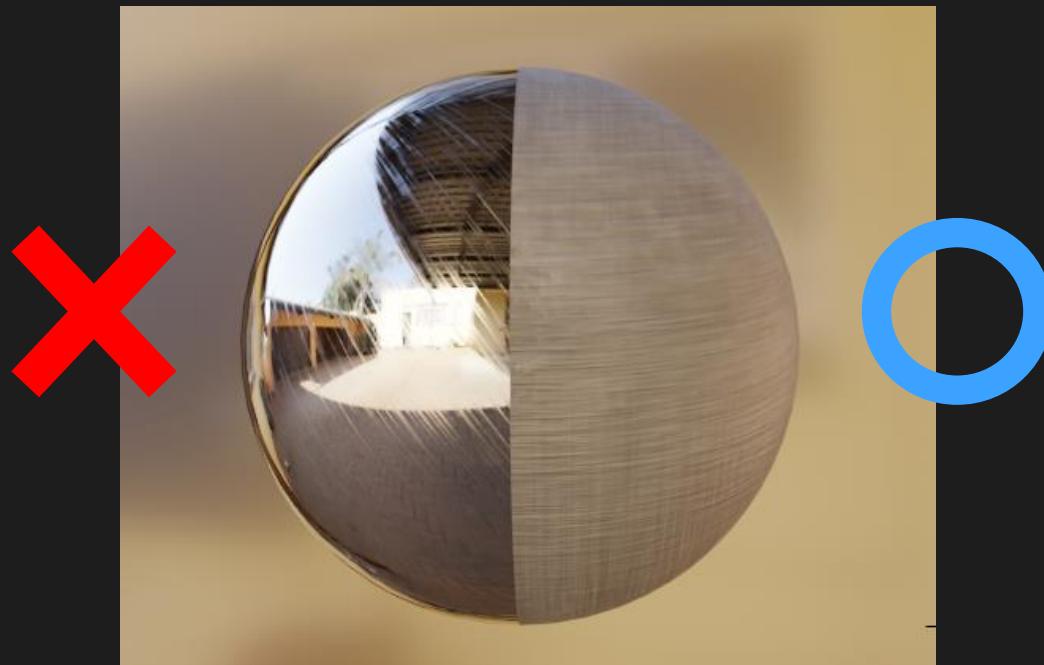
반짝이지 않는 것

3D스캔의 원리는 서로 다른 사진에서 같은 색을 가지는 포인트를 감지하는 것입니다.

투명한 유리나, 매끈한 금속은 각도에 따라 색이 바뀌므로 스캔할 수 없습니다.

텍스쳐가 완전한 단색이어서 특징잡을 만한 포인트가 없을때도 스캔이 잘 되지 않습니다.

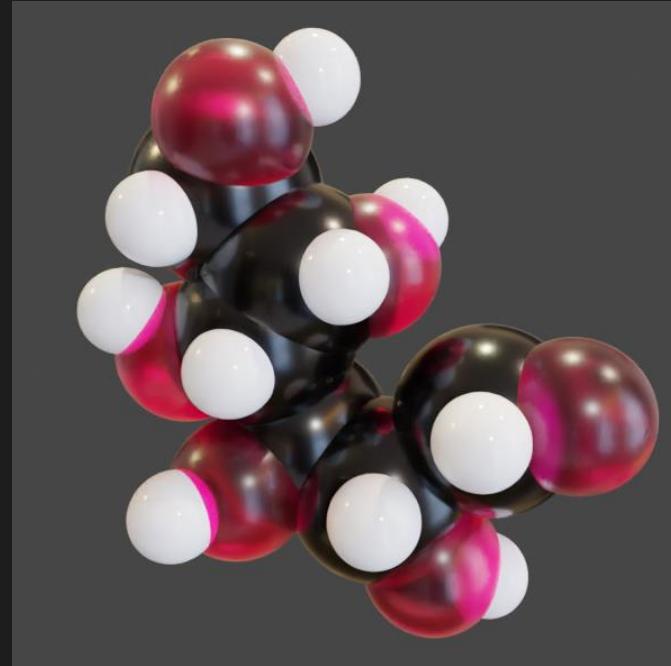
이런 재질들은 스캔 알고리즘에서 어느정도 보정해 주긴 하지만 일반적으로, 무늬가 뚜렷한 거친 불투명 재질만 스캔이 가능합니다.



Point Cloud

본래 용도와 상관없이, '점'의 데이터를 다루는 분야라면 포인트 클라우드를 사용할 수도 있습니다.

<https://www.youtube.com/watch?v=adhTmwYwOiA>



051강 Volume (1)

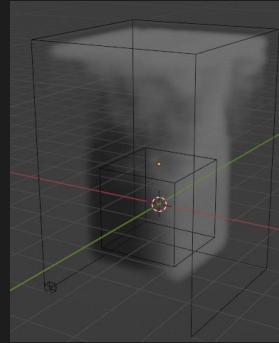
Volume의 종류
Volume Cube를 통한 메타볼 생성



볼륨의 종류와 지오메트리 노드의 볼륨

1. 볼륨 정보+셰이더

기존 방식은 메쉬에서 볼륨 정보를 사용하는 식으로 작동합니다.
플루이드 시뮬레이션이 이 방식을 사용합니다.



2. 새로운 볼륨 타입

최근 새로 생긴, 볼륨을 전문적으로 다루는 오브젝트 타입입니다.



2-B. 지오메트리 노드의 볼륨

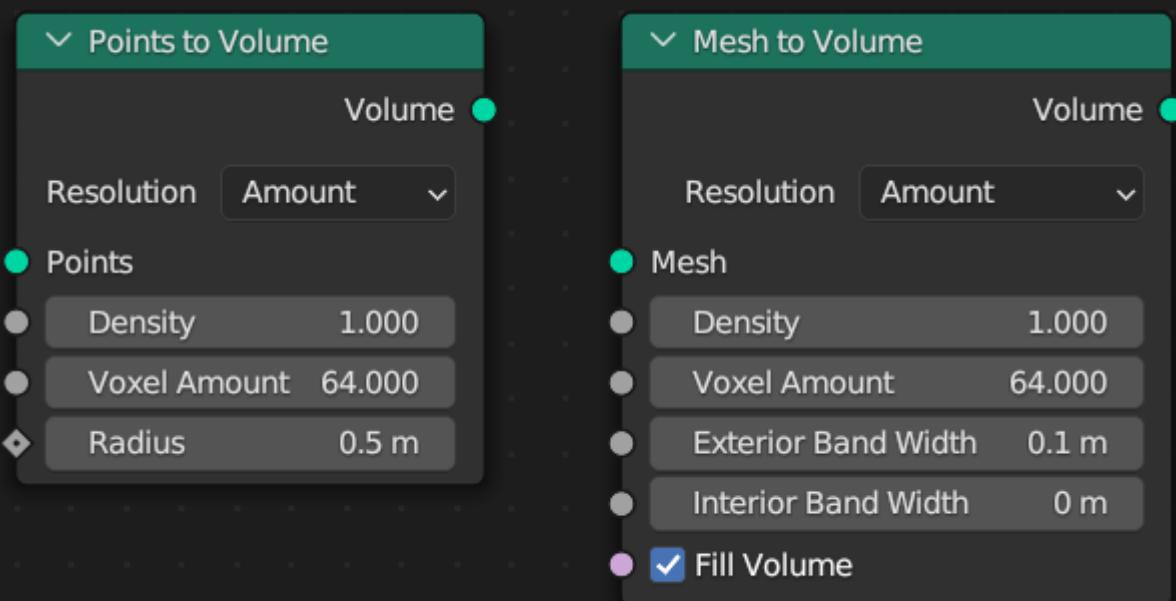
지오메트리 노드에서도 볼륨을 다룰 수 있습니다.
2 형식의 볼륨만 인식하며,
내부에서 생성될 때도 2 형식처럼 생성됩니다.

1의 셰이더 볼륨과는 제대로 상호작용하지 못합니다. (3.5기준)

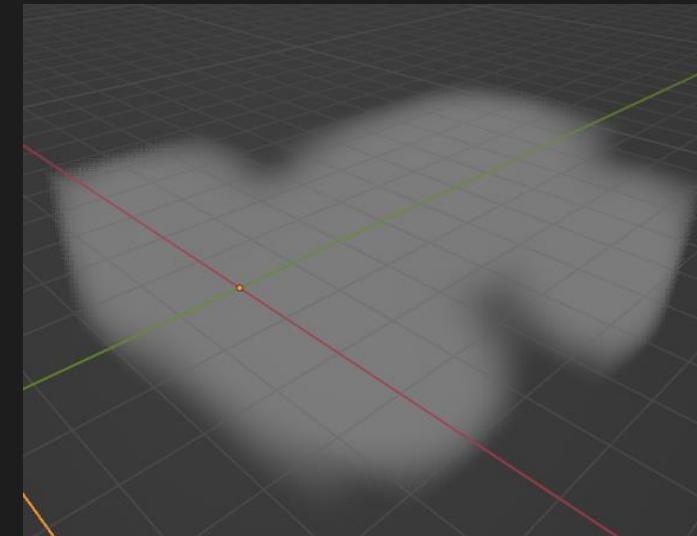
	Grid Name	Data Type	Class
Mesh	0 density	Float	Fog Volume
☐ Vertex	0		
☐ Edge	0		
☐ Face	0		
☒ Face Corner	0		
Curve			
○ Control Point	0		
↗ Spline	0		
Point Cloud			
■ Point	0		
Volume Grids	1		
Instances	0		

Geometry node의 볼륨

지오메트리 노드의 볼륨은, mesh를 volume으로, volume을 mesh로 변환할 수 있어, 볼륨을 메쉬 생성에 사용할 수 있습니다.



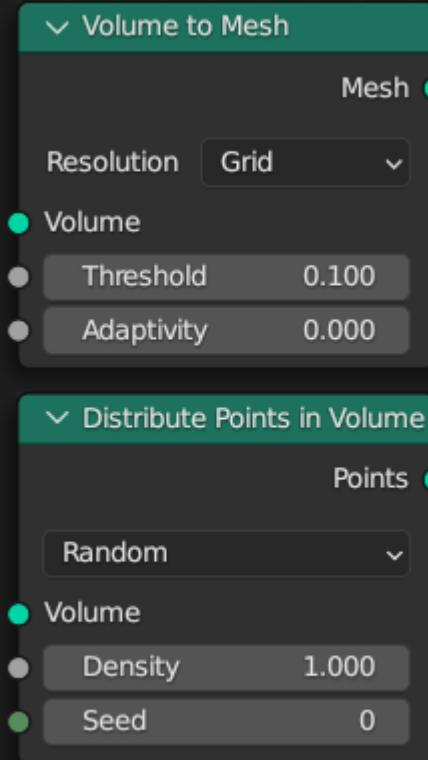
Points to Volume, Mesh to Volume : 포인트나 메쉬를 볼륨으로 바꿉니다.
포인트나 메쉬 근처의 공간을 입력한 밀도로 만들어 줍니다.



볼륨을 형성할 때는 Voxel이라는 개념을 사용합니다.
이미지의 Pixel처럼 공간을 상자 형태로 나누어 볼륨을 계산합니다.

복셀 수가 많아질수록 연산이 증가함에 유의하세요.

볼륨의 사용



Volume to Mesh : 볼륨을 메쉬로 변환합니다.

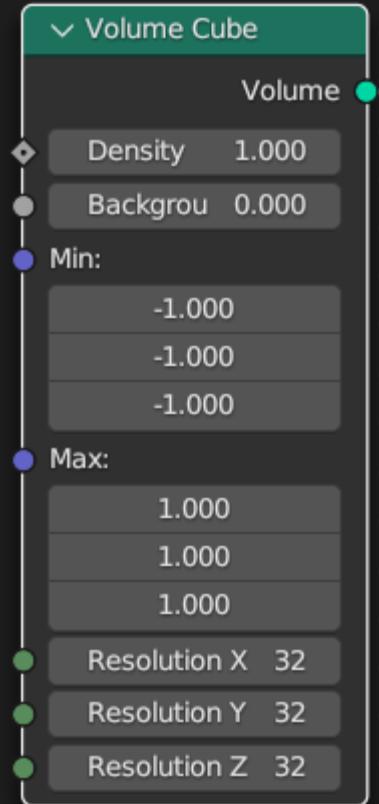
Threshold를 기준으로, 더 높은 밀도를 가진 쪽을 메쉬의 안쪽으로 계산합니다.

볼륨을 메쉬로 만들 때, 볼륨이 가진 복셀을 그대로 사용 하는 것이 grid 모드입니다.
이것을 Amount나 Size로 바꾸면 볼륨을 생성할 때처럼 새로 복셀을 계산합니다.

Adaptivity : 생성된 메쉬에서 불필요한 점들을 없애 단순화시켜줍니다.
성능이 별로 좋지는 않습니다.

Distribute Points in Volume : 34강에서 알아보았듯이, 볼륨 내부에 포인트를 분포시킵니다.

Volume Cube

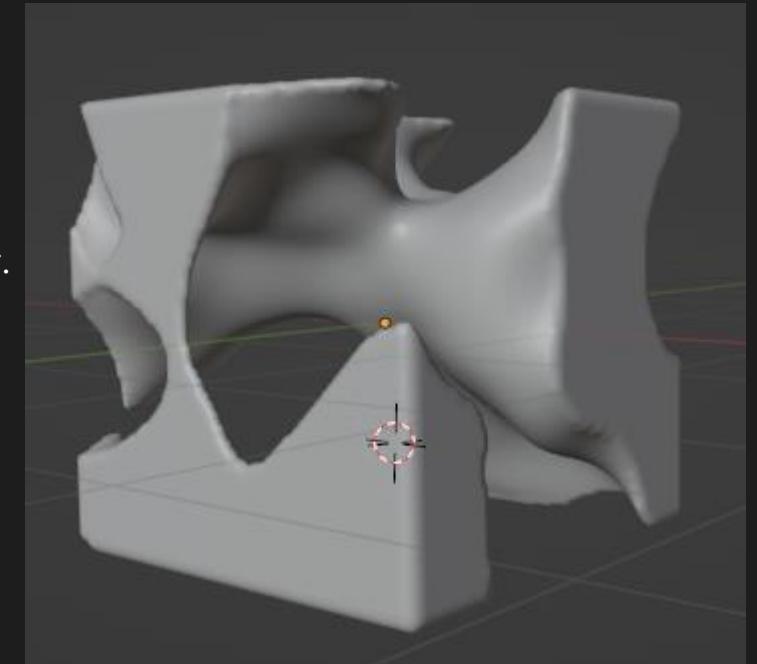


밀도(Density) 를 바탕으로 볼륨을 생성합니다.

29강 셰이더 파트에서 3차원 텍스쳐를 통해 볼륨의 밀도를 조절했던 것과 같은 방식이지만,
이것은 지오메트리 노드이기 때문에 나중에 Mesh로 변환할 수 있습니다!

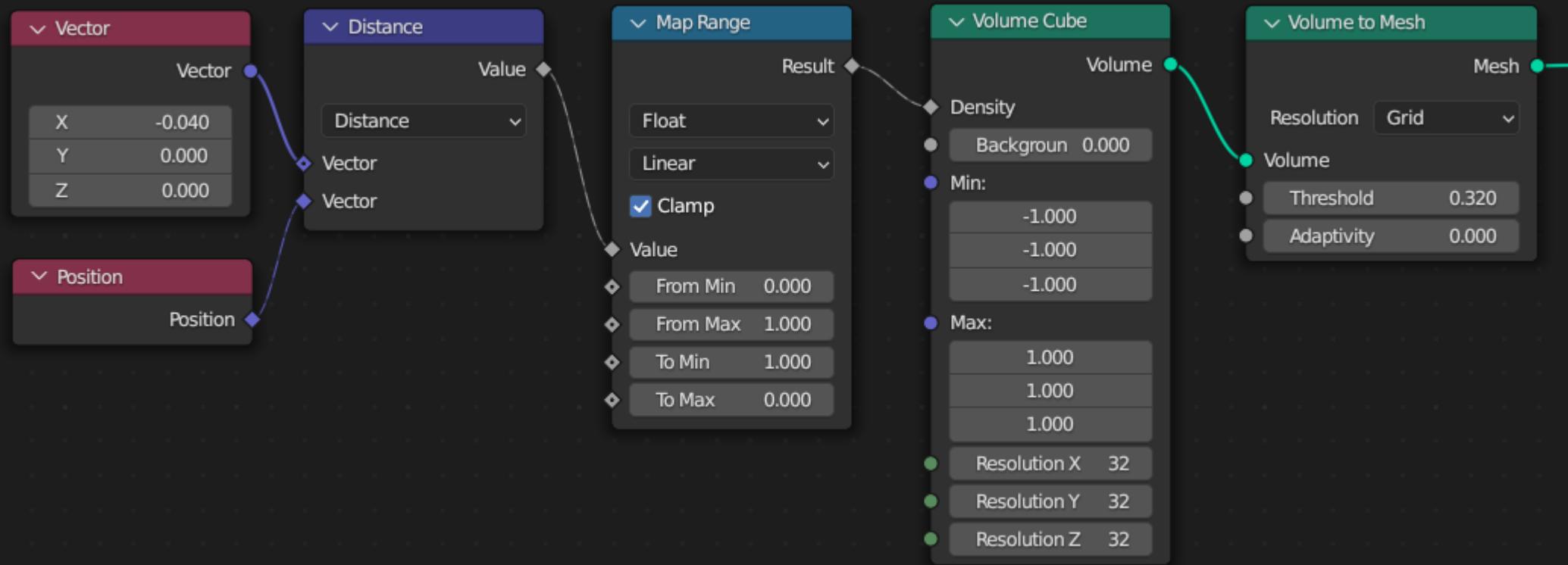
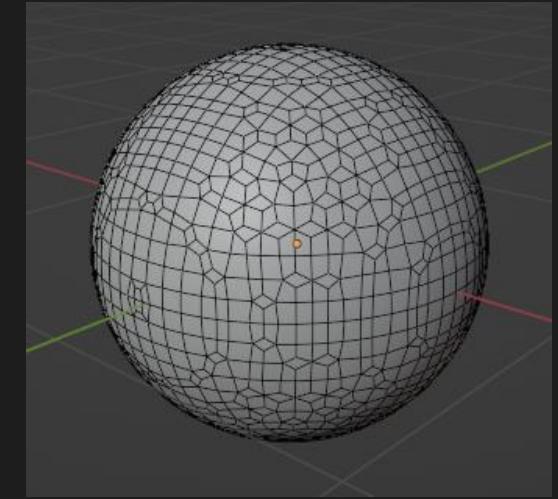
Position

Volume Cube에 연결된 Position 노드는 복셀의 좌표를 의미합니다.



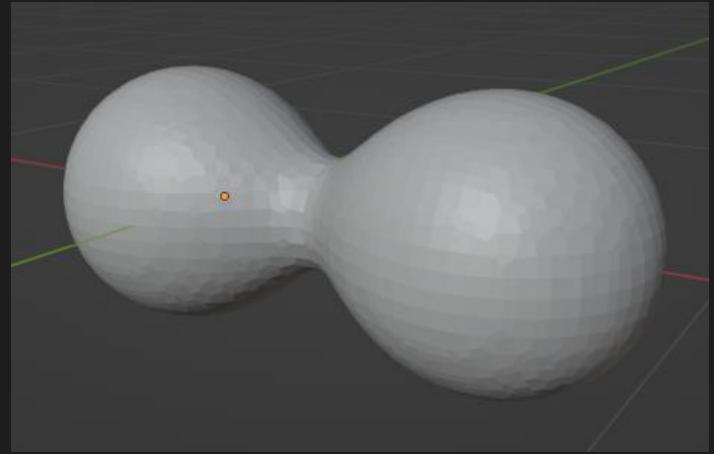
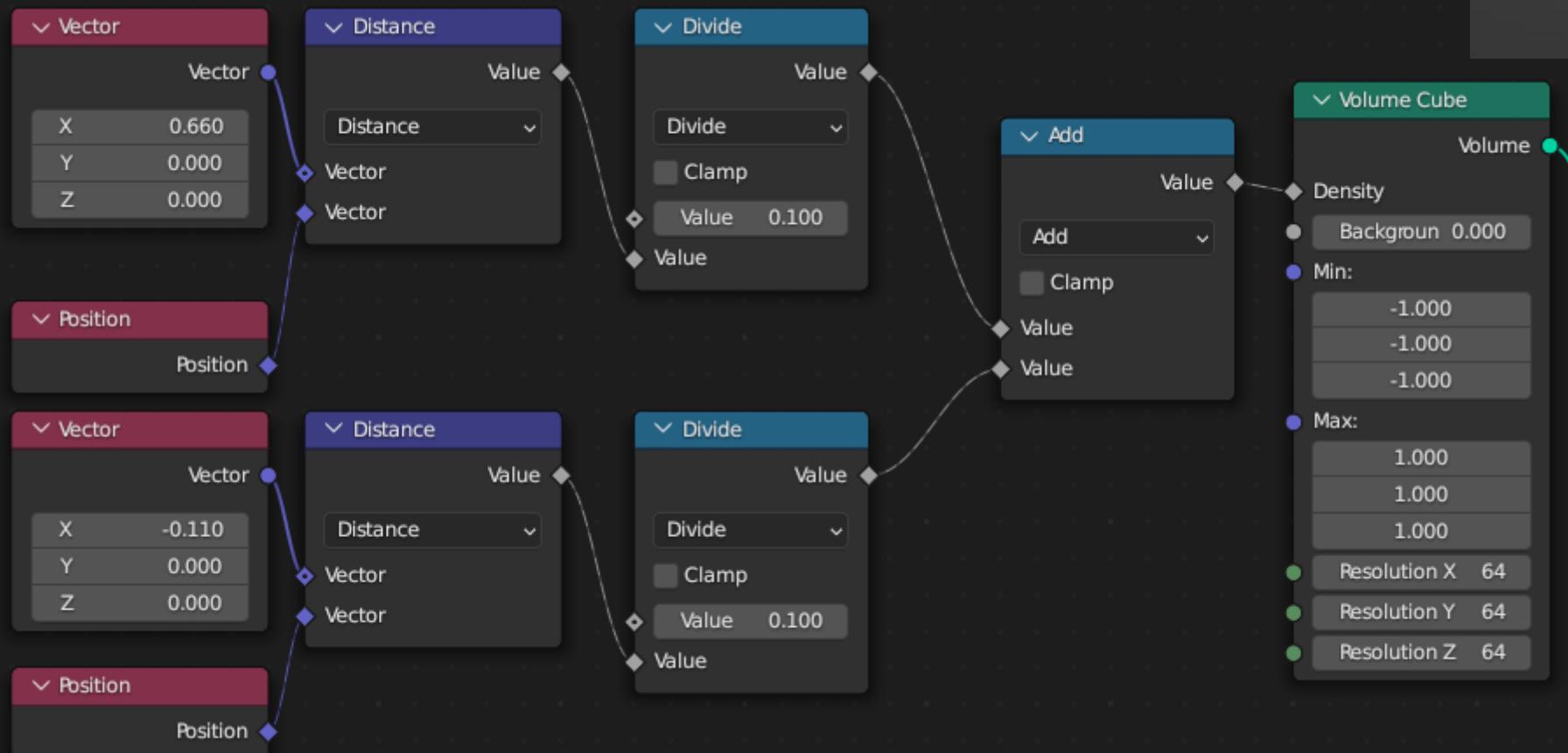
Metaball 입문

밀도 함수를 만들어 메쉬를 생성할 수 있습니다. 아래는 특정 점으로부터의 거리를 이용하는 방법입니다.



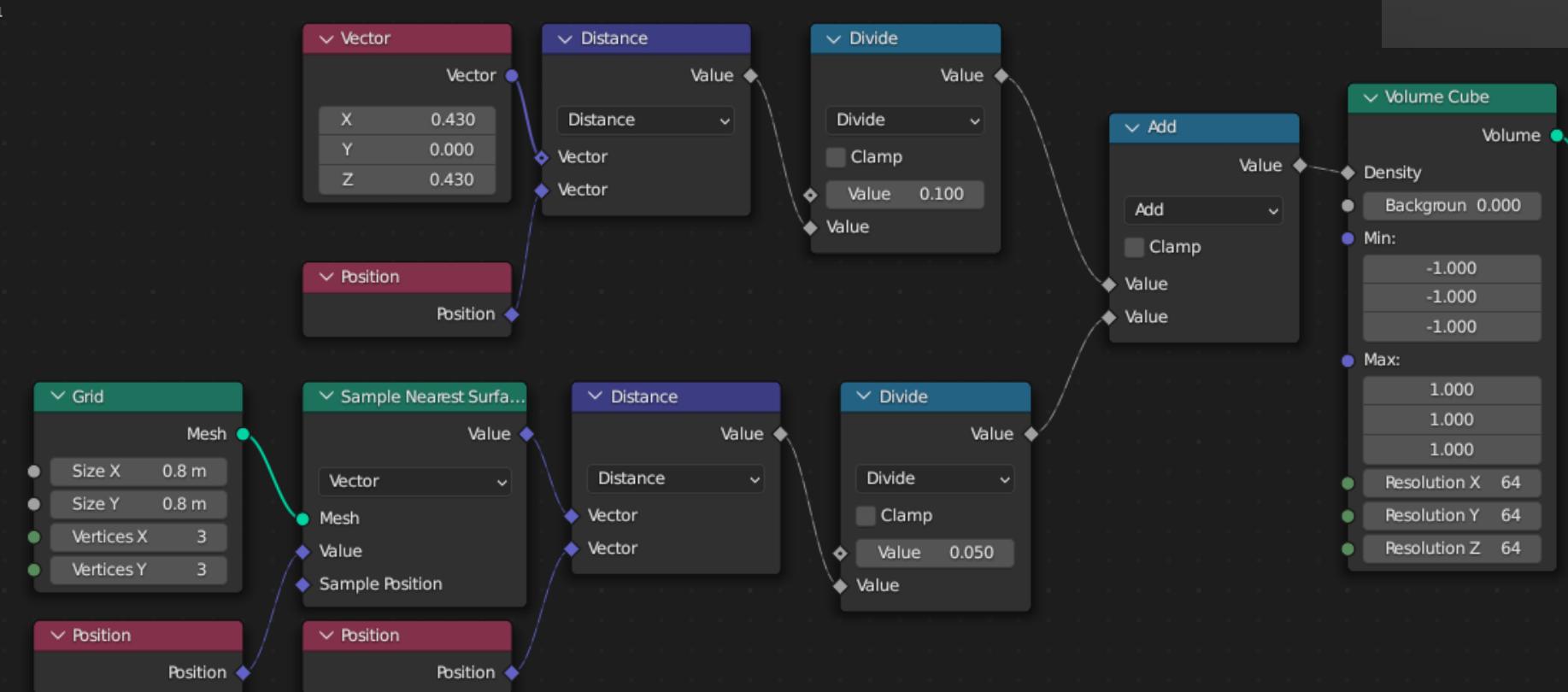
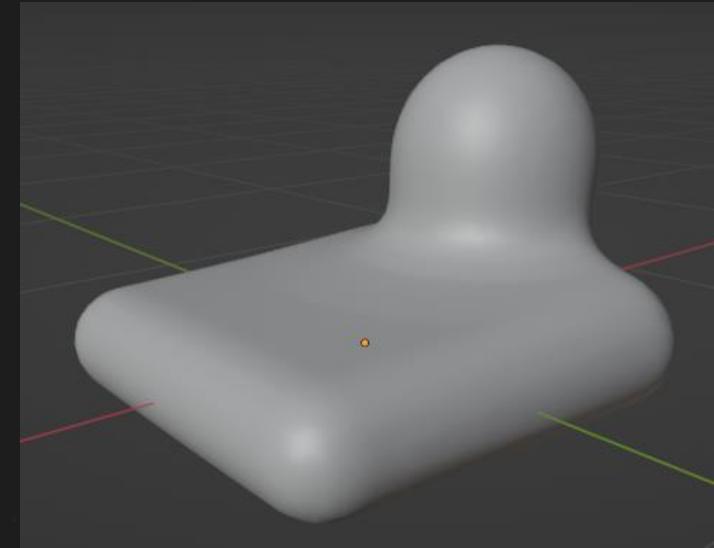
두 개의 메타볼

메타볼 함수를 **거리의 역수**로 정의한 뒤, 이들을 합한 값을 기준으로 밀도를 정의하여 메쉬를 생성합니다.



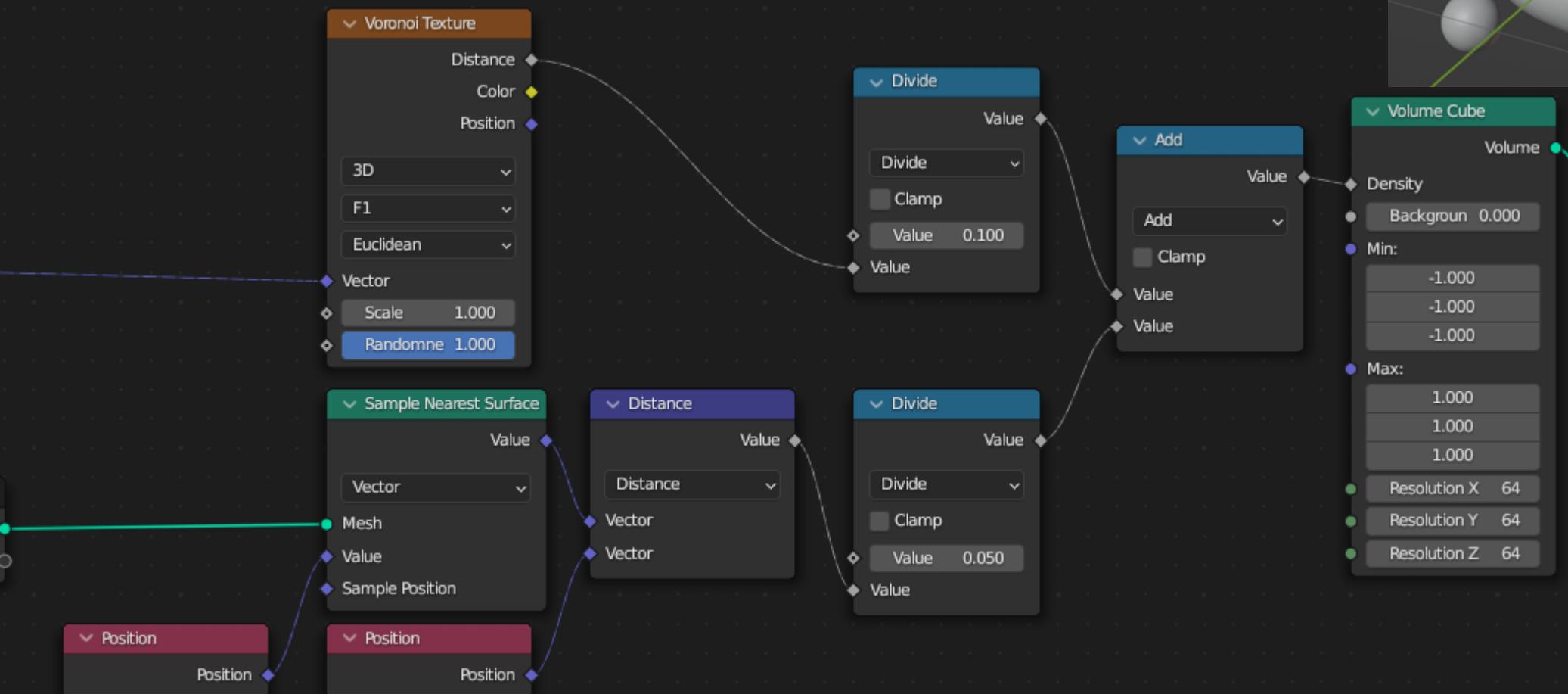
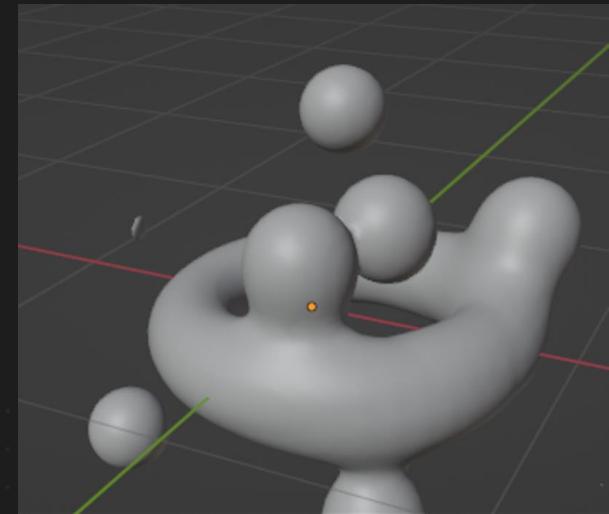
표면의 메타볼

Sample Nearest Surface를 사용하면 표면으로부터의 거리를 잴 수 있습니다.
앞에서와 동일한 메타볼 함수를 통해 형태를 유기적으로 합칠 수 있습니다.



Voronoi

Voronoi는 임의로 흩뿌린 점으로부터의 거리를 나타내기 때문에,
이 거리에 역수를 취하면 메타볼 함수에 쓸 수 있습니다.
보로노이 내부 점들끼리의 상호작용은 일어나지 않지만, 외부와의 상호작용은 가능합니다.



052강 Volume (2)

Volume을 연산하는 법
Mesh Volume을 이용한 개비온 담장 만들기

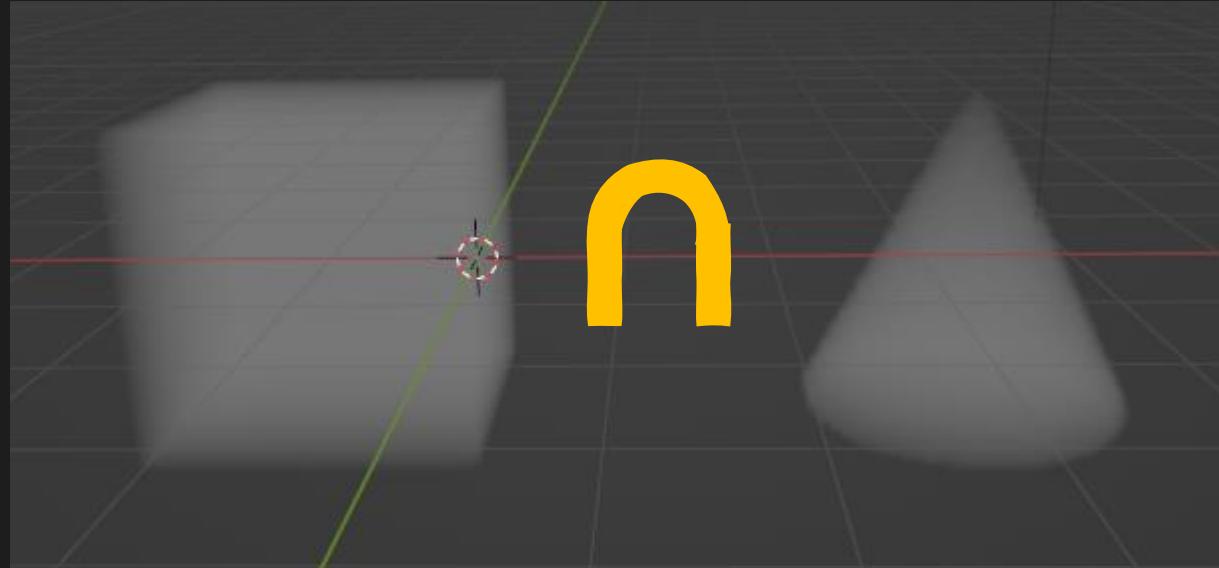


볼륨의 연산?

여러 '볼륨' 을 합치거나 겹치는 부분을 찾는 등의 컨트롤이 가능할까요?

안타깝지만 그것은 불가능합니다. (3.5기준)

하지만 mesh to volume을 이용하지 않고, 메쉬를 [밀도 함수](#)로 만들어서 Volume Cube 를 이용한다면 컨트롤할 수 있습니다.

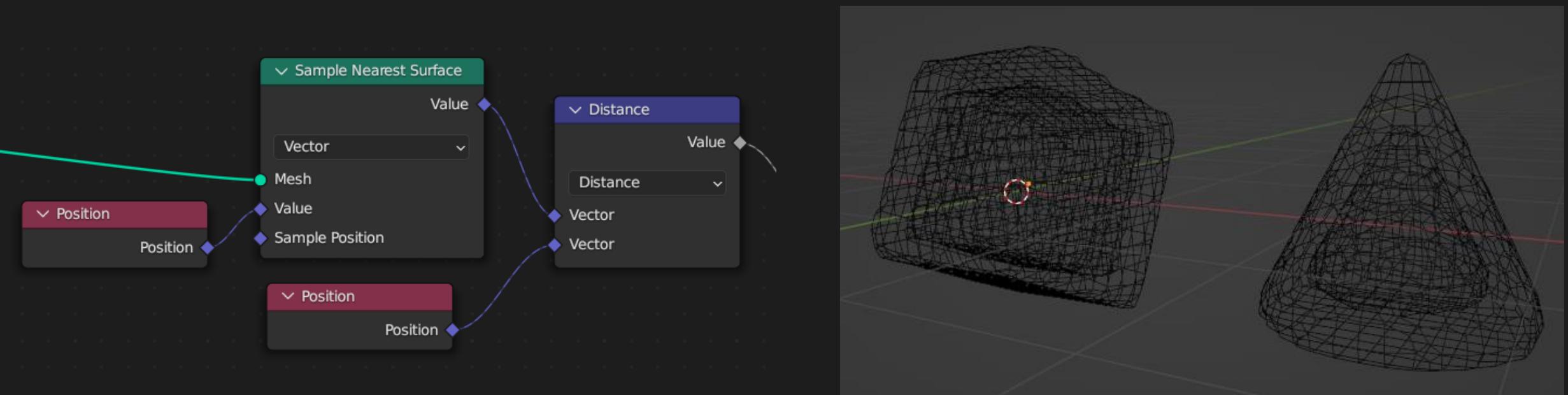


?

내부와 외부

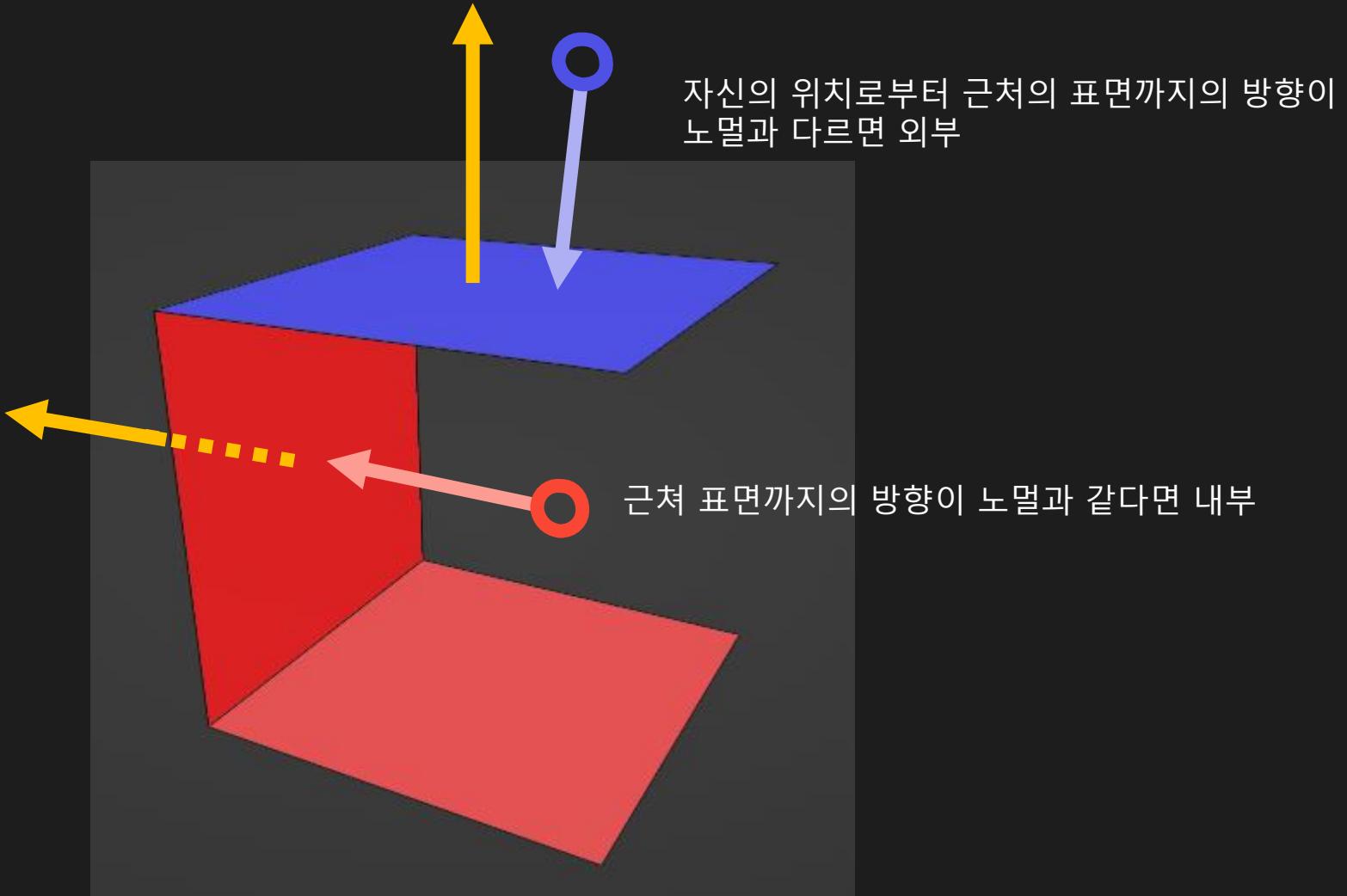
이전 시간의 Sample Nearest Surface를 이용한 방식은 Mesh를 Volume으로 만드는 데 한계가 있습니다.
말 그대로 표면으로부터의 거리를 재기 때문입니다.

볼륨을 만들기 위해서는 현재 위치가 물체의 내부인지 외부인지 판단할 수 있어야 합니다.



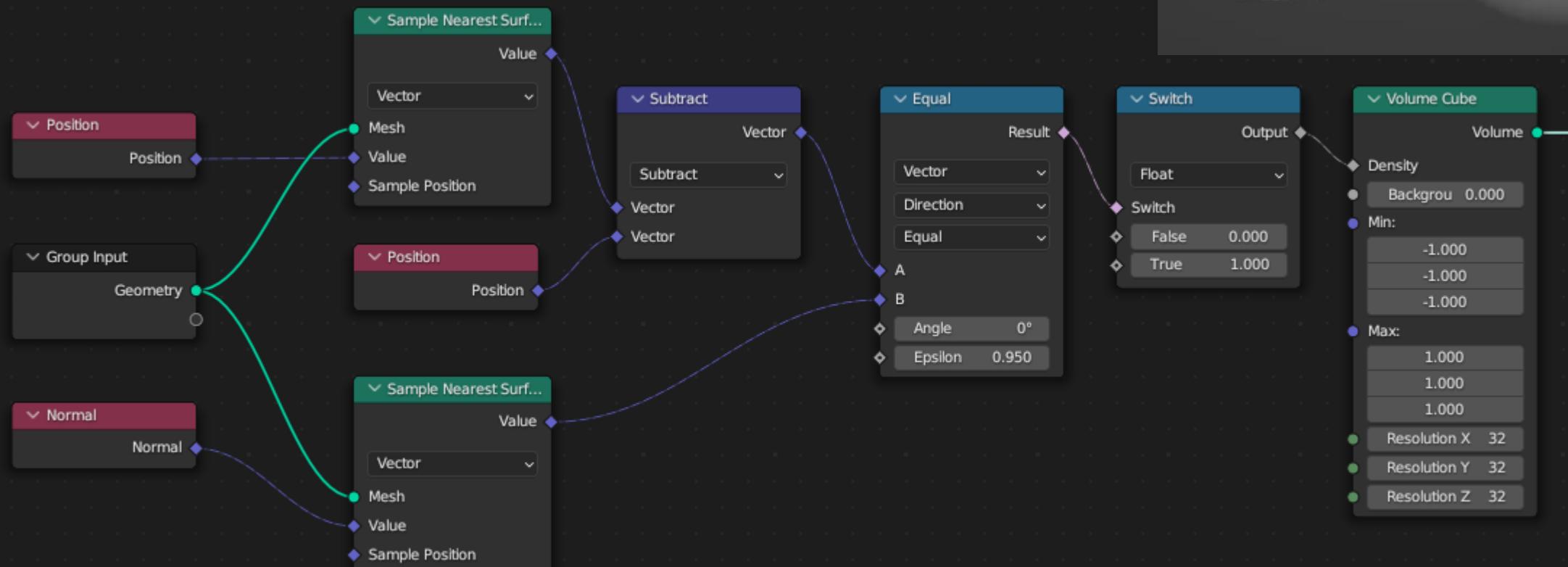
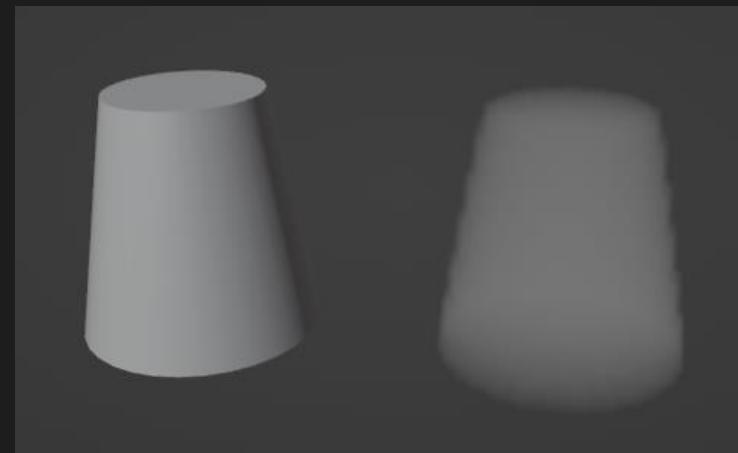
내부와 외부

Normal을 이용하여 내부/외부를 파악할 수 있습니다.



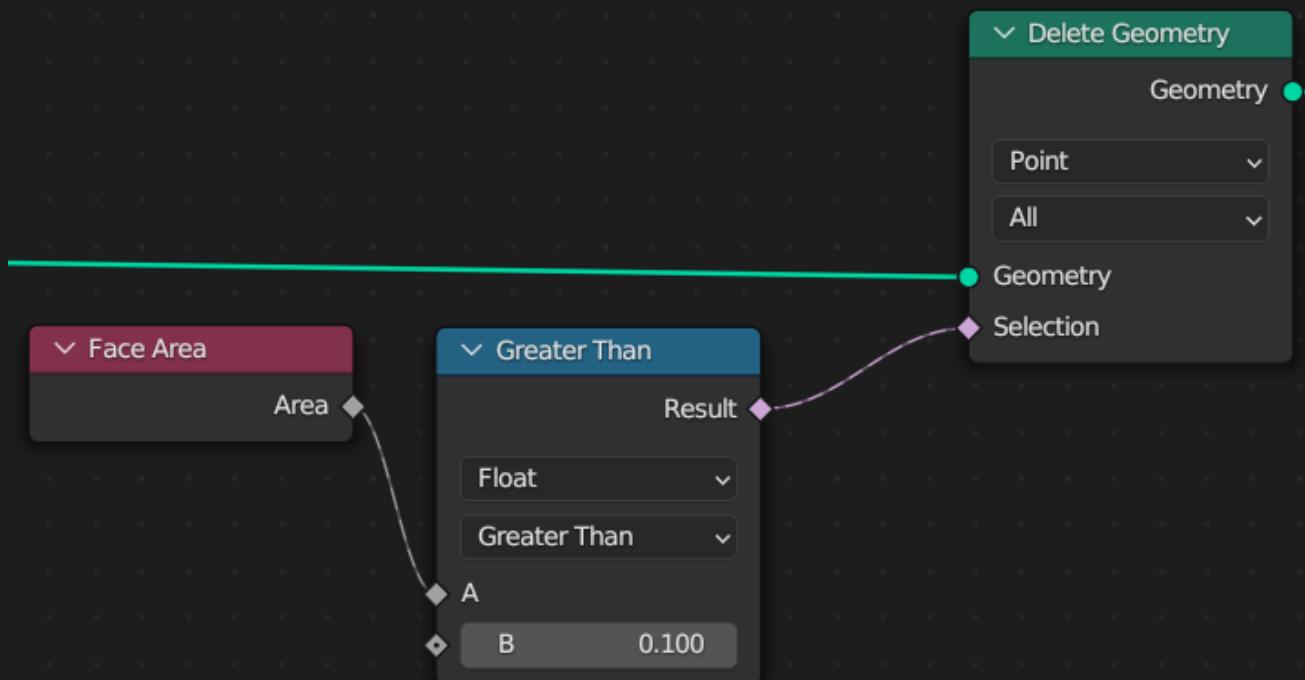
내부와 외부

아래와 같이 Mesh to Volume을 재구성할 수 있었습니다. 하지만 이 경우엔 밀도 함수를 얻었으므로, 이제 밀도 상태에서 다른 것들과 연산할 수 있습니다!



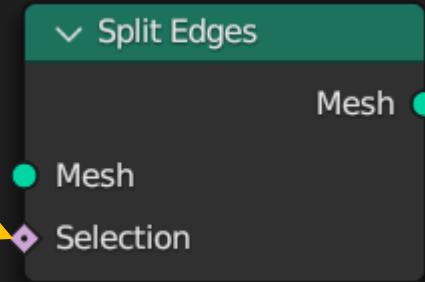
053강 도메인 문제

Attribute domain 문제 : 소켓의 점, 선, 면
Laplacian smoothing



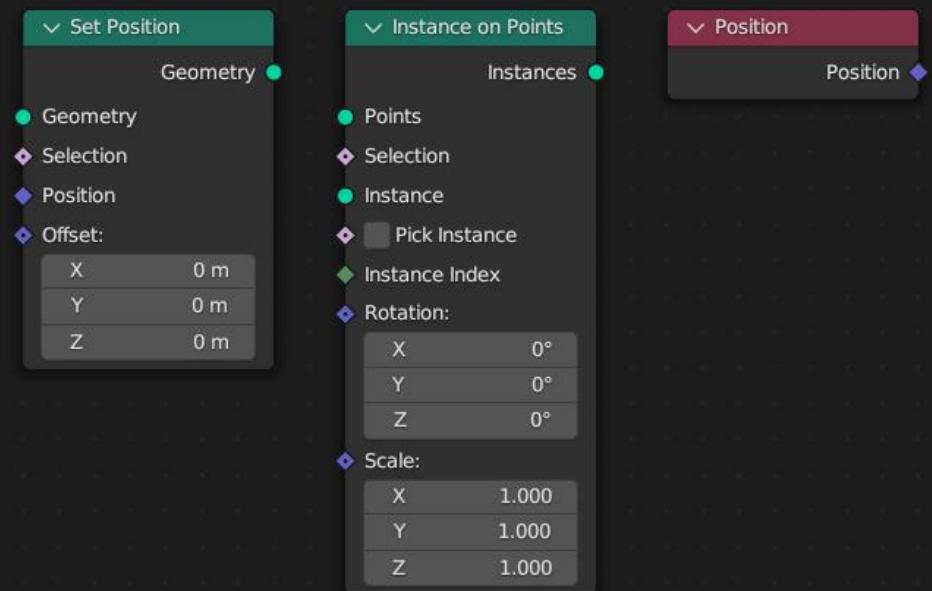
도메인 문제

일부 노드는 특정 도메인을 사용해야 제대로 작동합니다.
이럴때는 도메인을 신중하게 생각해야 합니다.



대부분은 Point입니다

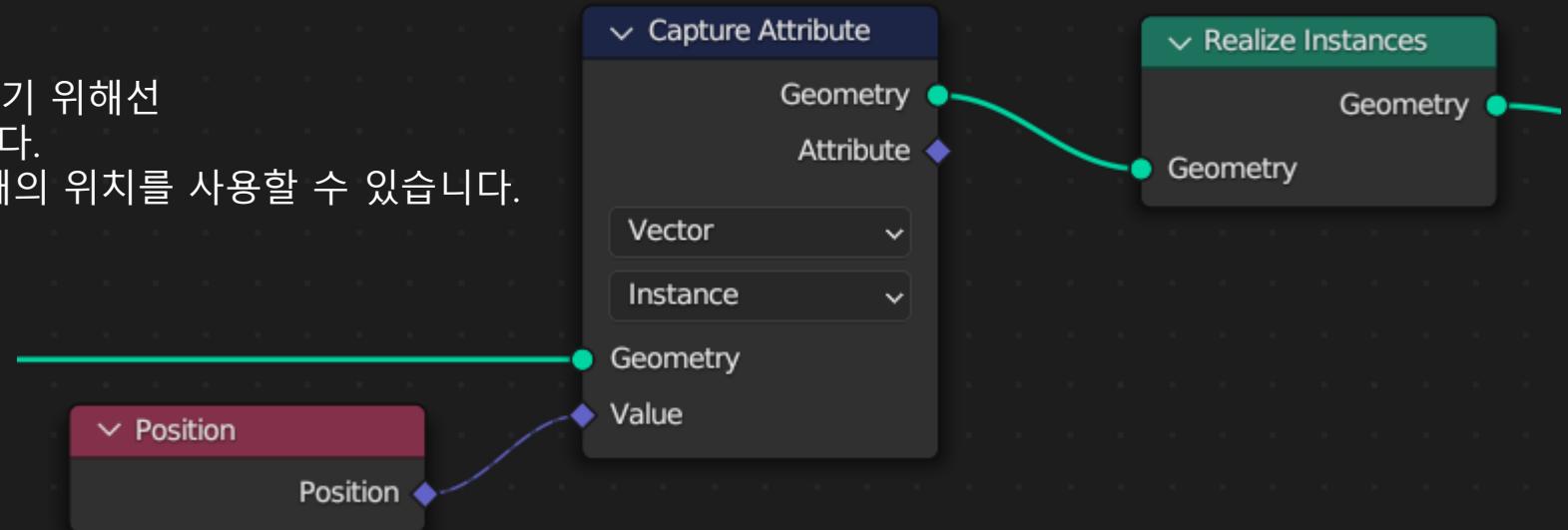
지금까지 도메인을 생각할 필요가 없었던 것은,
사용했던 노드 대부분이 포인트를 기준으로 작동했기 때문입니다.



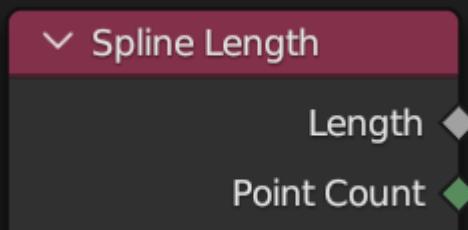
Point가 아닌 것

Instance

Realize 후에도 인스턴스의 위치를 사용하기 위해선
인스턴스 타입으로 위치를 캡쳐해야 합니다.
이렇게 하면 Realize 뒤에도 인스턴스일 때의 위치를 사용할 수 있습니다.



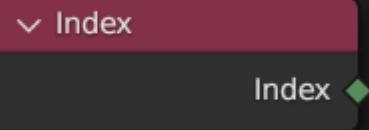
Spline



Spline Length 노드는 스플라인 도메인의 데이터입니다.
스플라인 정보를 컨트롤 포인트에서 읽을 때는 도메인이 달라지지만 별로 문제가 없습니다.

Point가 아닌 것

Index



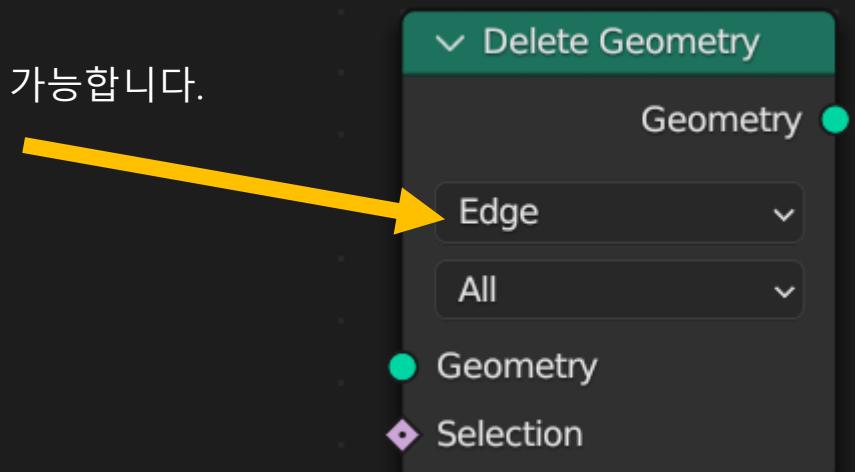
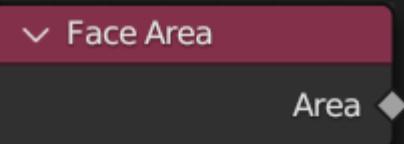
인덱스는 점, 선, 면 모든 도메인이 가지고 있습니다.
따라서, 연결상태에 따라 불러오는 도메인이 바뀔 수 있습니다.

Edge, Face 관련 노드



도메인이 Point가 아닌 노드들은 이름으로 쉽게 확인 가능합니다.
일부 노드는 노드 내에서 도메인을 고르기도 합니다.

자세한 목록은 마지막 페이지를 참고하세요.

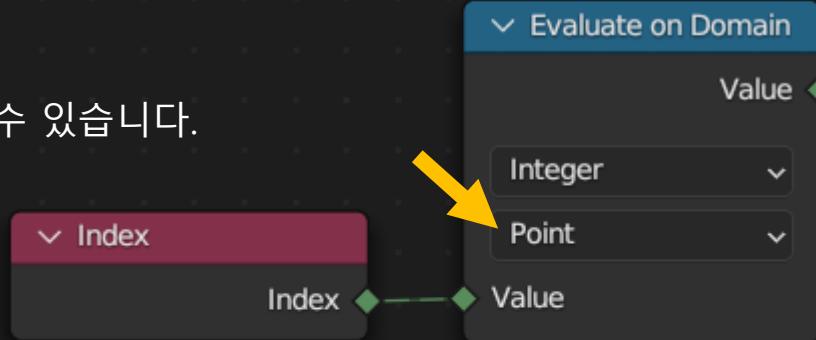


Evaluate on Domain

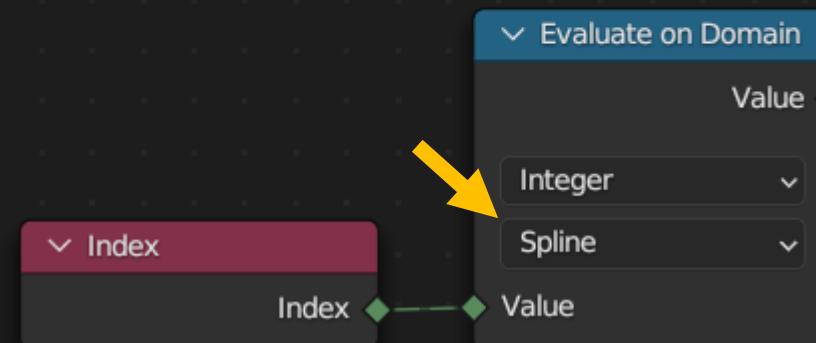
※ 이 노드의 이전 버전 이름은 Interpolate Domain 이었습니다.

연결된 정보가 어떤 도메인의 값인지 설정할 수 있습니다.

이것은 '컨트롤 포인트의 인덱스' 가 나오고,

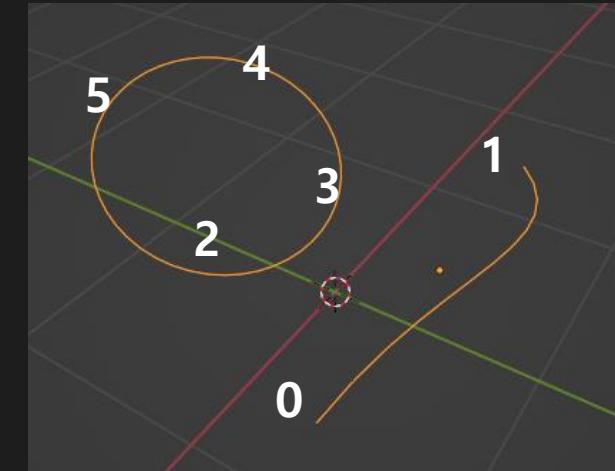


이것은 '스플라인의 인덱스' 가 나옵니다.



이처럼 Index를 Evaluate on Domain에 연결하면, 인덱스는 모든 도메인에 존재하기 때문에, 각 도메인의 인덱스 정보를 불러 오지만, 만약 Point의 정보를 Edge의 값으로 불러온다거나 하면, 문제가 달라집니다.

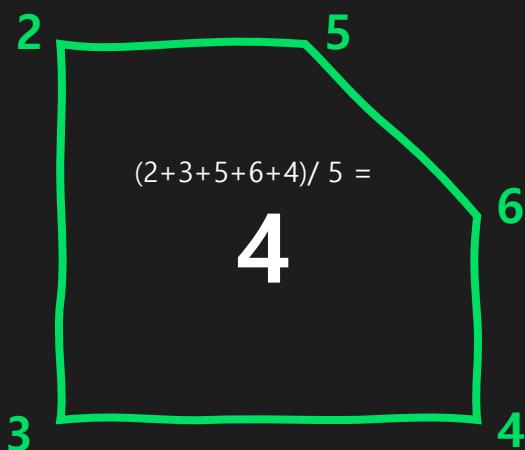
그런 경우 Attribute는 Interpolation되어 불러와집니다. 자세한 이야기는 다음 페이지를 참고하세요.



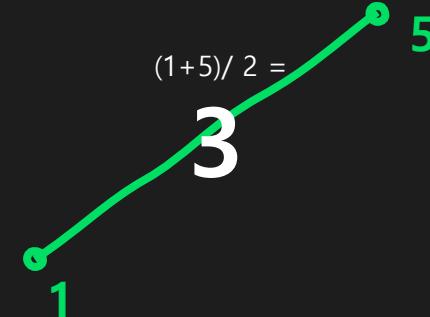
Interpolation

도메인이 변경될때는 interpolation, 쉽게 말해서 '평균'을 내서 가져옵니다. 예를 들어, Face의 Position은 Face를 이루는 점들의 위치의 평균이 됩니다.

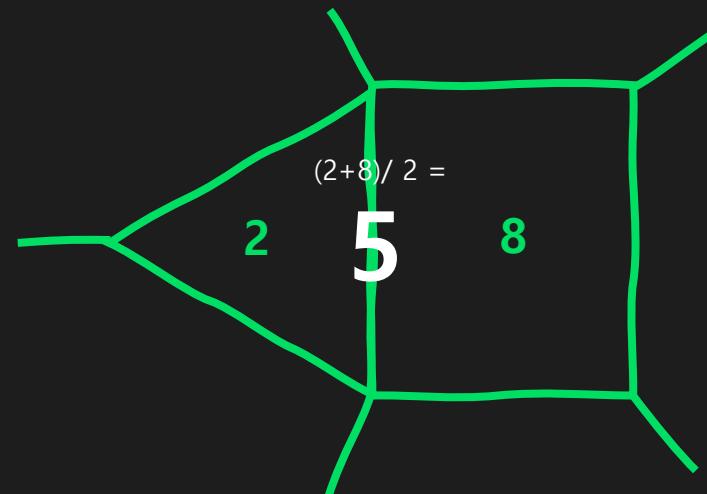
Point ► Face



Point ➤ Edge



Face ➤ Edge

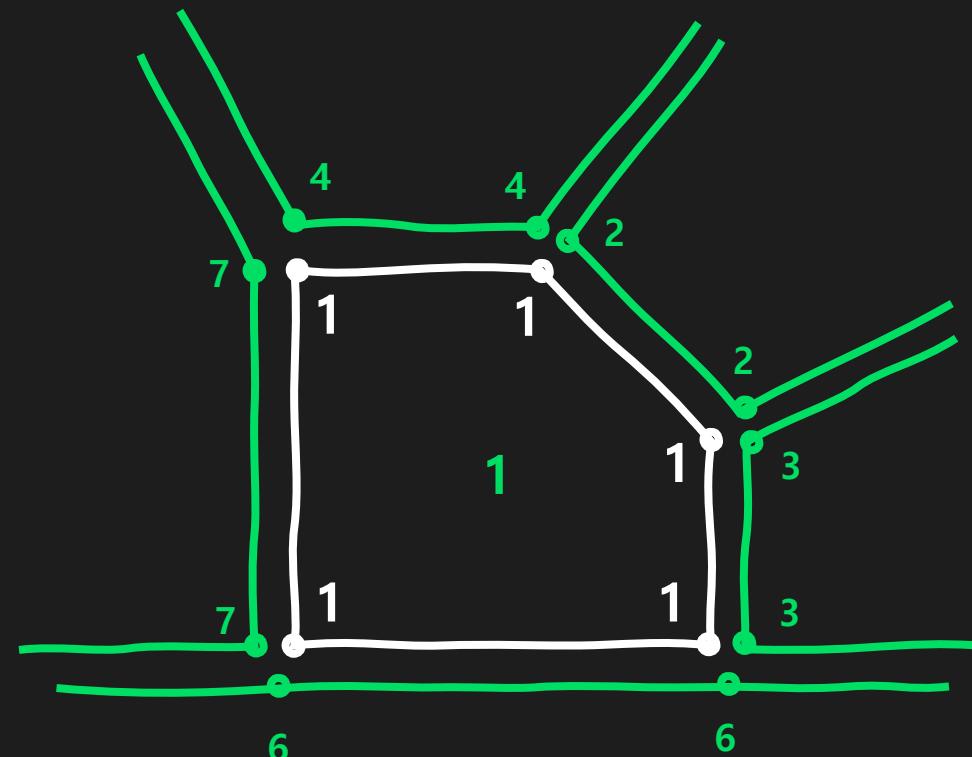
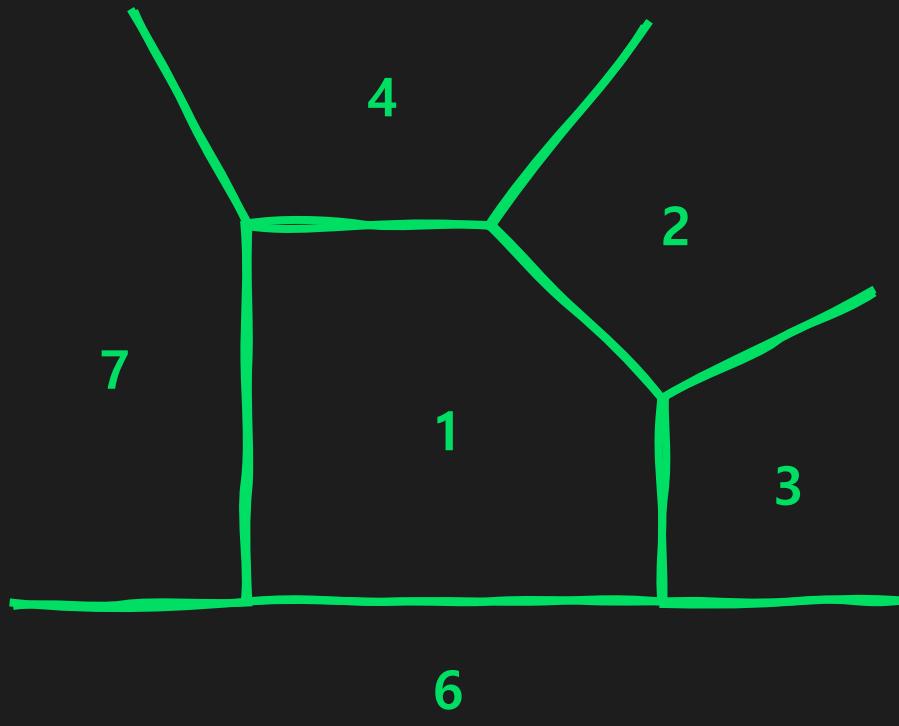


이외의 다른 조합들 (Edge to Face, Face to Point 등...) 도 동일하게 작동합니다.

Face Corner

Face Corner는 Face를 따로따로 분리시켰을 때 그것을 둘러싼 점들을 가리킵니다.
그 정의 때문에, Face ▶ Face Corner로 바꿀 때는 자신이 속한 Face만 가지고 계산합니다.

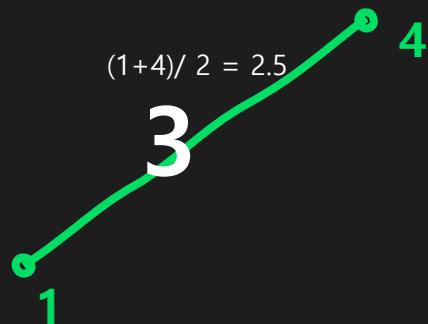
Face ▶ Face Corner



데이터타입 문제

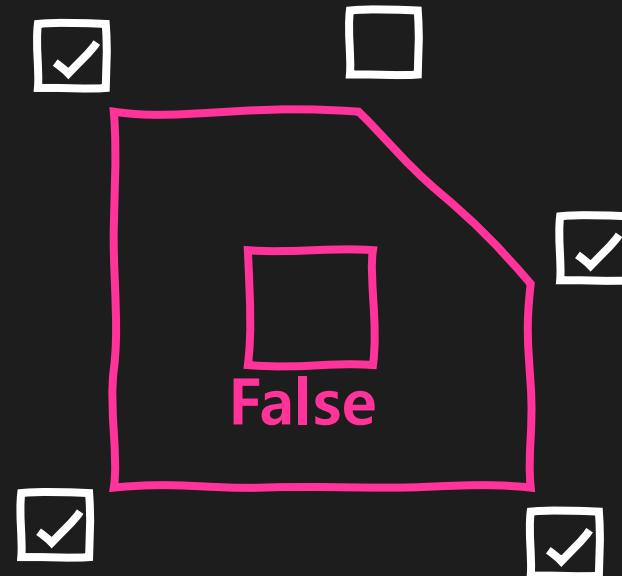
실수나 벡터 타입은 '평균' 을 내지만, 정수와 불린은 조금 다릅니다.

Point ▶ Edge



정수는 **반올림**으로 계산됩니다.

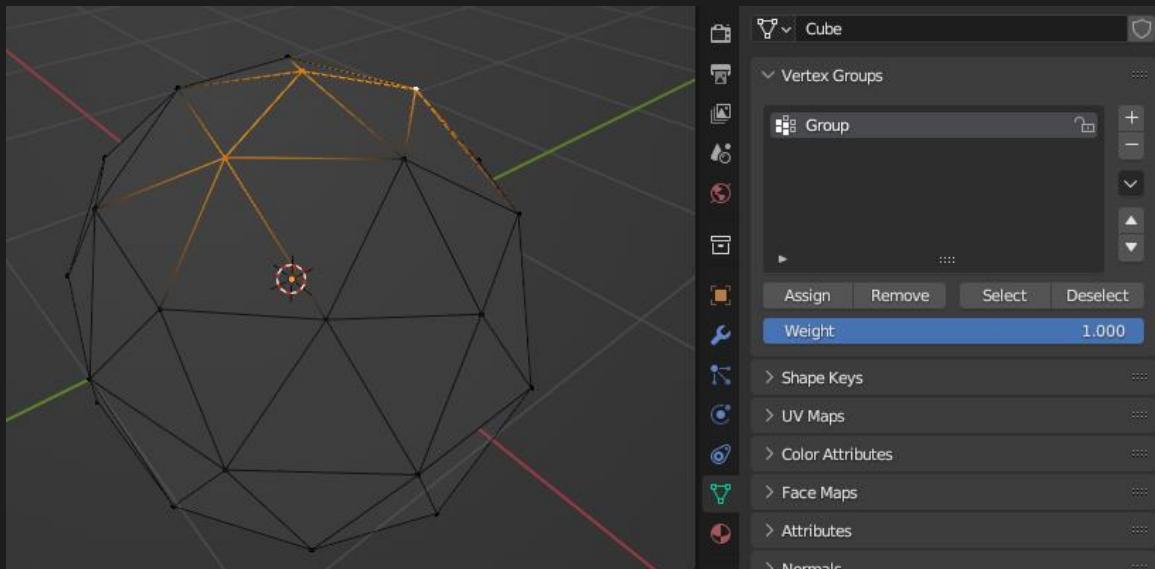
Point ▶ Face



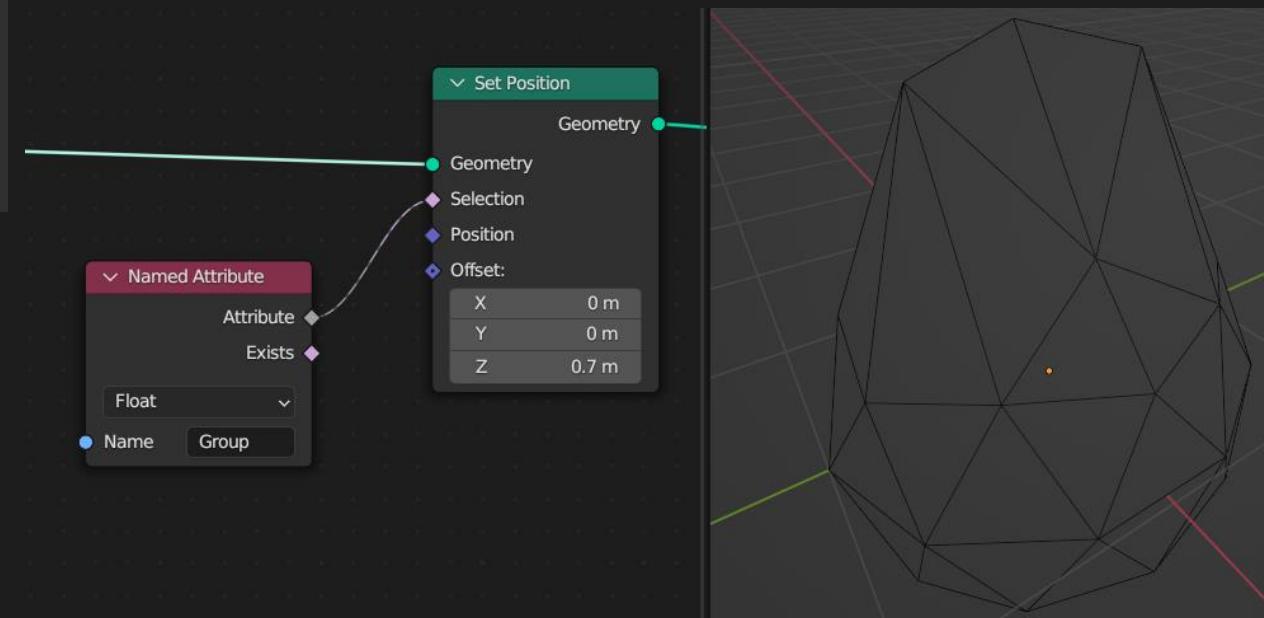
불린은 **모두 참일때만 참**입니다.

수동으로 Edge를 선택하려면

Reminder : Vertex Group

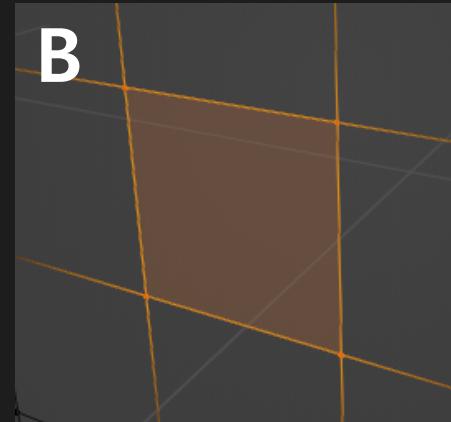
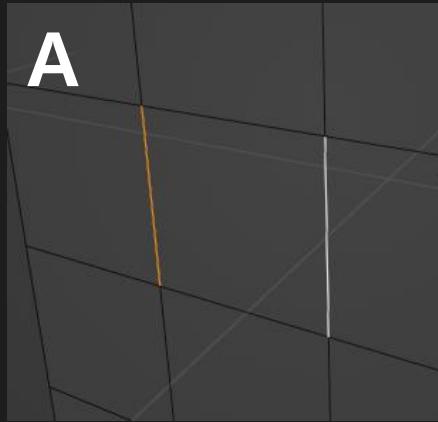


Vertex Group을 통해 수동으로 점을 선택할 수 있습니다.
이후 named Attribute 를 통해 Group을 노드로 가져오실 수 있습니다.



수동으로 Edge를 선택하려면

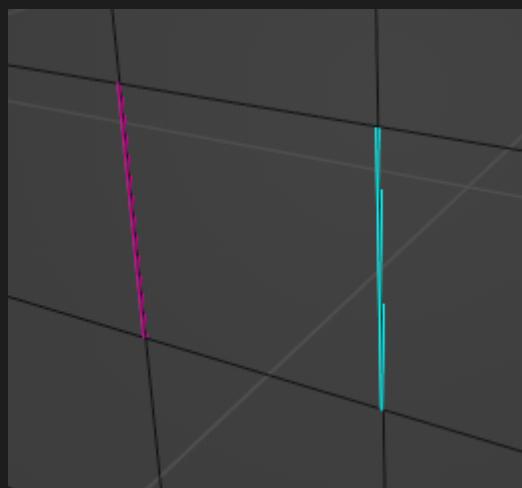
Vertex 선택을 Edge로 변경 시의 문제



버텍스 그룹으로는 A처럼 엣지를 선택할 수 없습니다.

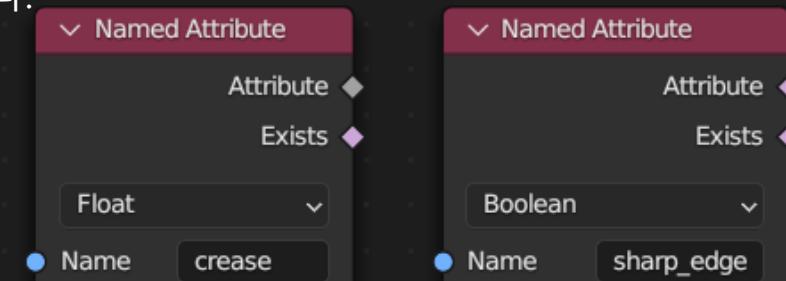
엣지를 이루는 4개의 점을 선택하면, 앞에서 설명한 방식으로 점의 정보가 Edge로 Interpolate되어, B처럼 4개의 엣지가 선택됩니다!

엣지를 수동으로 선택하여 지오메트리 노드에서 사용하려면



3.5버전 기준으로 지오메트리 노드에서 인식 가능한 Edge Attribute는 두 가지입니다.
[Mark sharp](#) (엣지 우클릭 후 Mark sharp) 혹은
[Edge Crease](#) (Shift+E - 마우스 좌우 이동 후 클릭으로 결정)
같은 것들을 사용하면, 수동으로 엣지를 선택할 수 있습니다.
다만 이것들은 다른 기능을 겸하기 때문에,
경우에 따라 사용이 곤란할 수도 있습니다.

이후 기능이 추가될 예정이라고 하니 기다려 봅시다.



Interpolation을 활용할 수 있을까요?

Laplacian Smoothing

▼ Edge Vertices

Vertex Index 1 

Vertex Index 2 

Position 1 

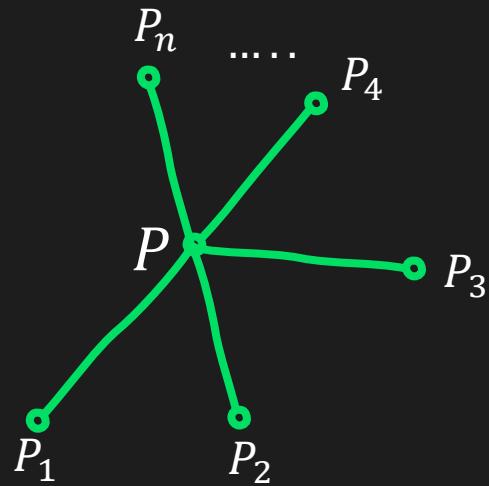
Position 2 

Edge Vertices 노드는 엣지 양 끝점의 인덱스와 포지션을 출력합니다.

물론 도메인이 Edge이므로, Point에서 불러오려 한다면 주변 점들의 위치를 평균낸 값이 됩니다.

※ Edge Vertices를 포함한 토플로지 노드들은 57,58강에서 자세히 다룰 예정입니다.

Laplacian Smoothing



Laplacian Smoothing은 자신의 이웃 점들의 위치의 평균을 가져오는 것입니다.

즉,

$$P_{new} = (P_1 + P_2 + P_3 + \dots + P_n) / n$$

Position 1, 2를 P에서 뺄려오면

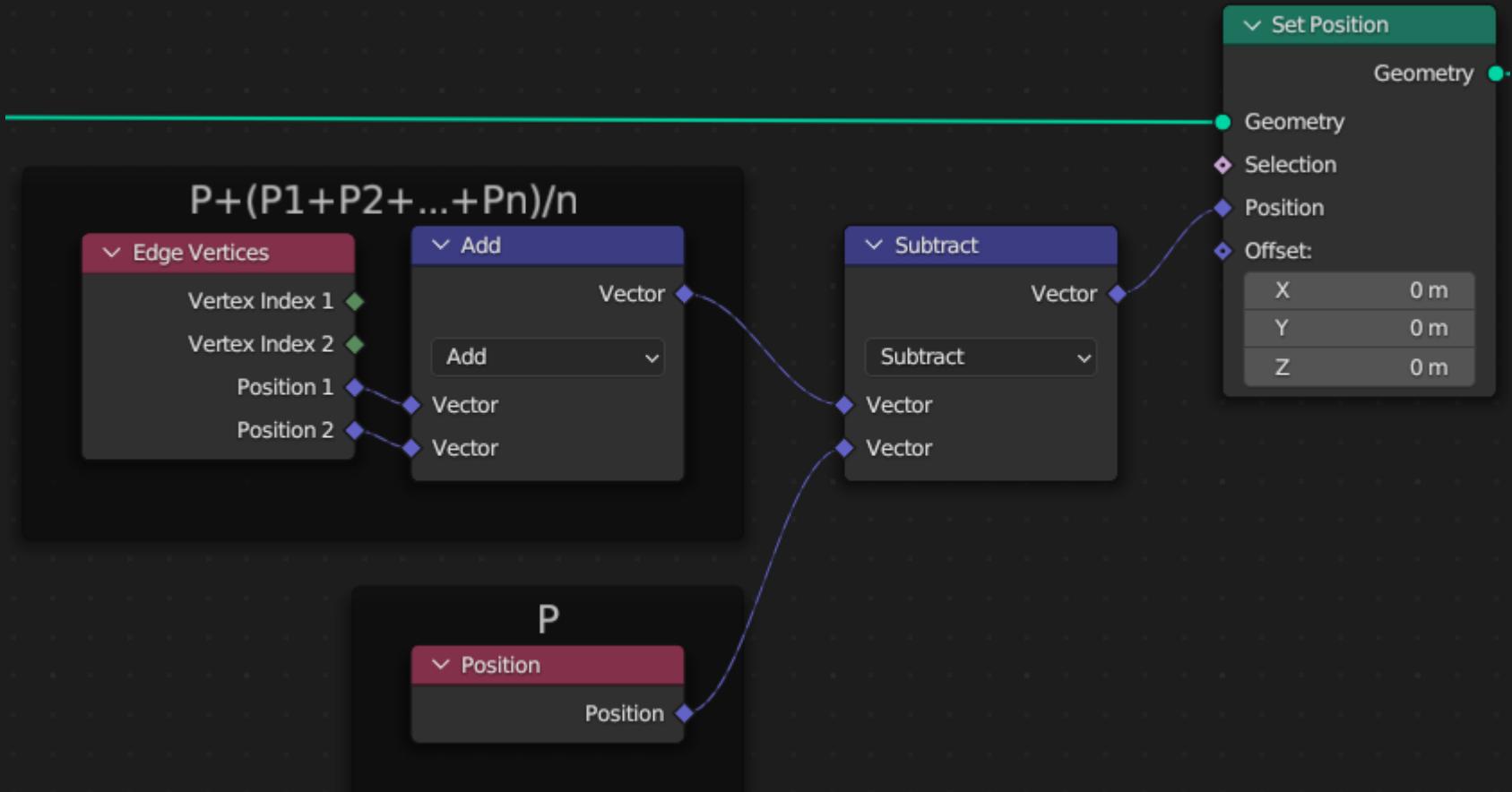
$$\begin{aligned} Position_1 + Position_2 &= \{(P + P_1) + (P + P_2) + (P + P_3) + \dots + (P + P_n)\} / n \\ &= (nP + P_1 + P_2 + P_3 + \dots + P_n) / n \\ &= P + (P_1 + P_2 + P_3 + \dots + P_n) / n \end{aligned}$$

따라서 $Position_1, Position_2$ 의 합에서 자기 자신의 위치 P 를 빼주면 Laplacian Smoothing이 됩니다!

Laplacian Smoothing

아래와 같이 Edge Vertices의 합에서 Position을 빼 줍니다.

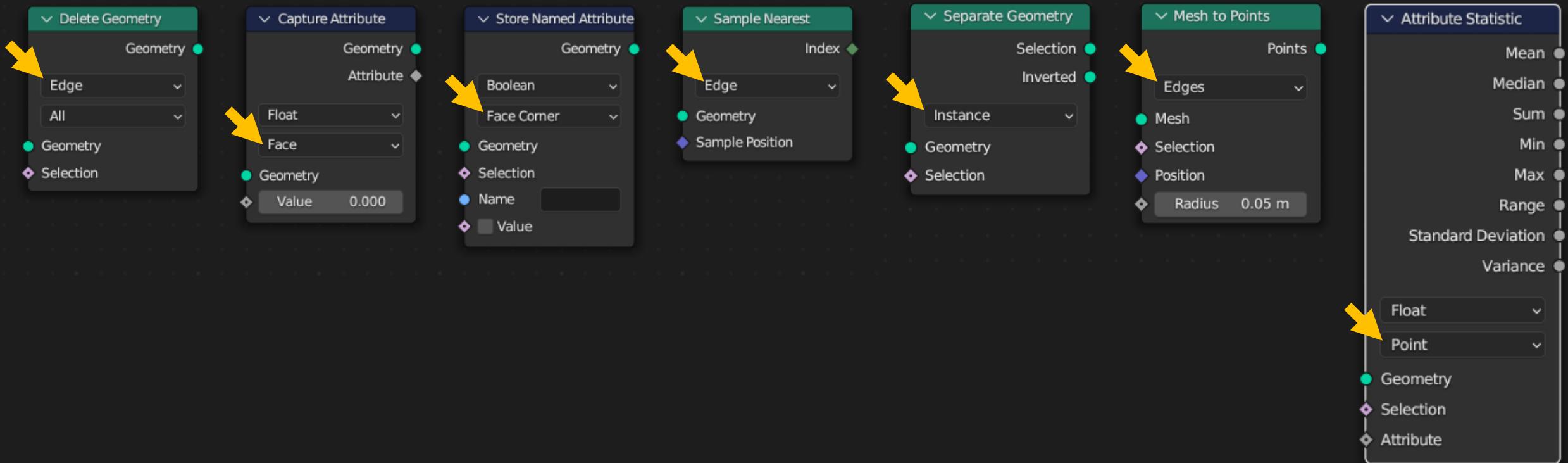
※인터넷의 많은 강좌에서 Position 1,2를 합한 뒤 평균을 내기 위해 2로 나누는데, 정확하지 않은 계산입니다.



Appendix

아래의 노드들은 도메인을 선택할 수 있는 것들입니다.

이런 노드들은 Evaluate on Domain을 사용하지 않아도 도메인을 자동으로 바꾸어 계산합니다.

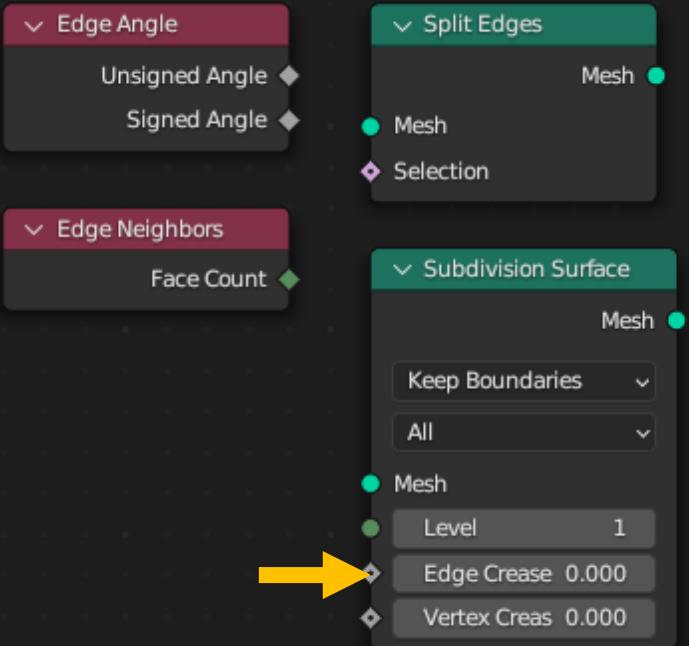


Appendix

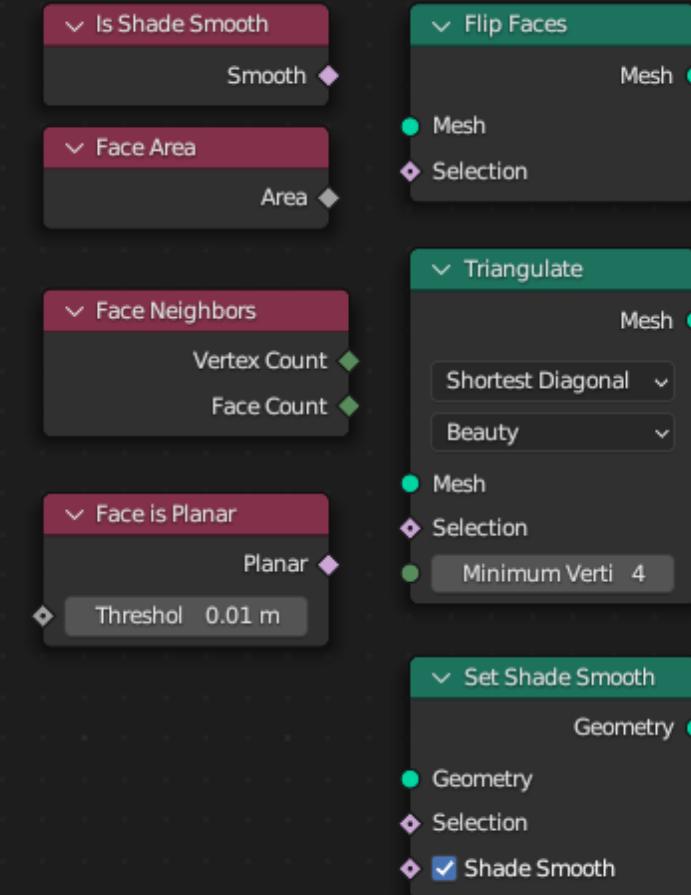
아래쪽은 Point 도메인이 아닌 노드들입니다.

※여기에는 토플로지 노드들은 빠져 있습니다. 지금 거기까지 생각하면 상당히 어려워지므로, 토플로지 노드는 조금 뒤에 알아보겠습니다.

Edge

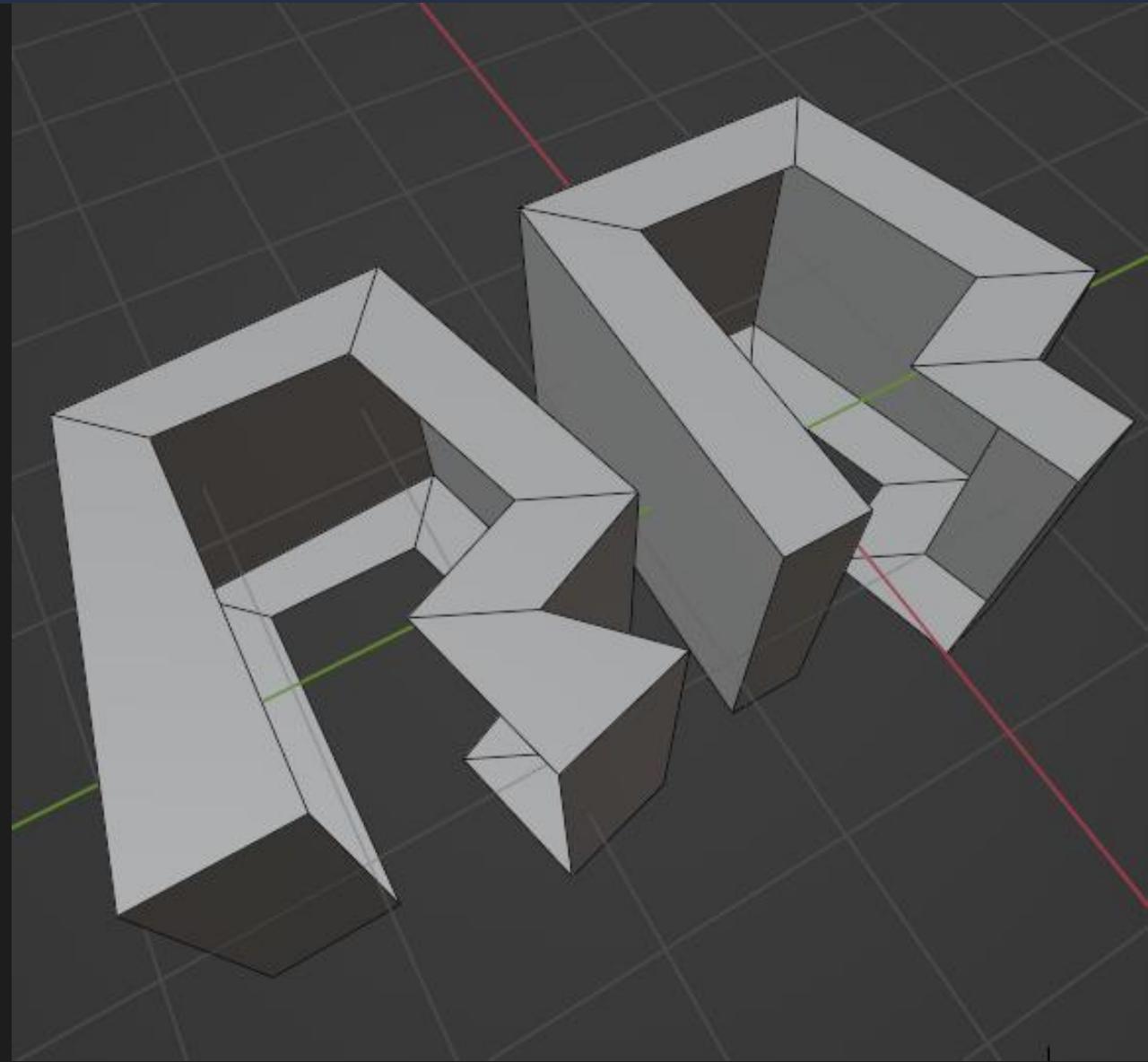


Face



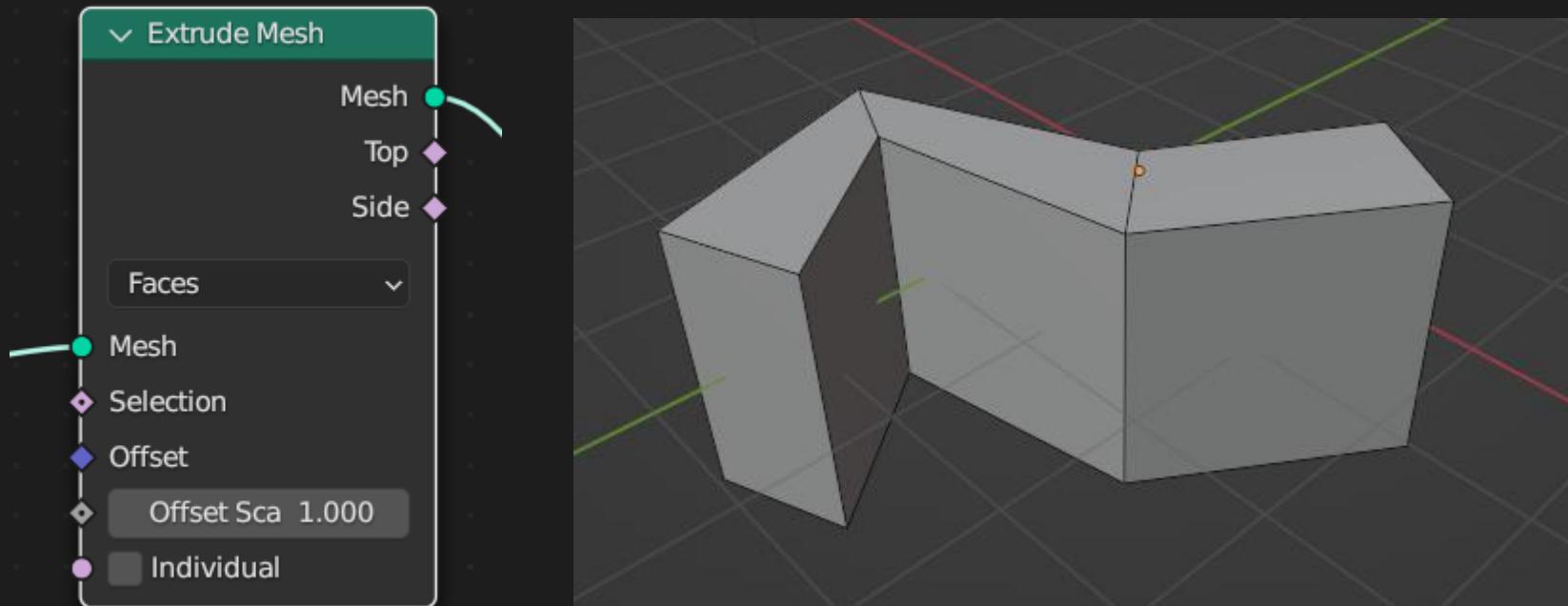
054강 도메인 문제 (2)

Normal의 도메인
Even extrude 를 구현하는 법



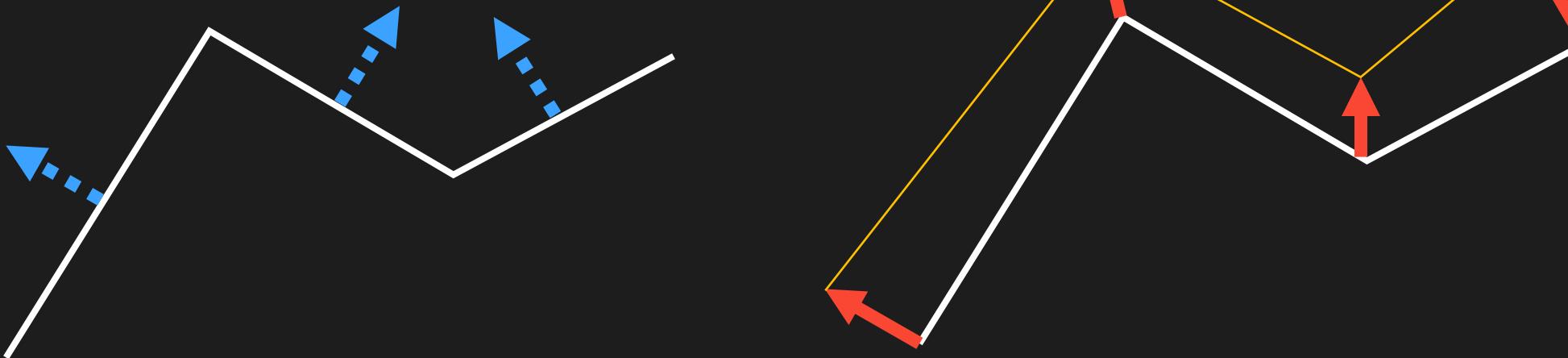
Extrude Mesh의 문제점

Extrude Mesh는 모델링에서의 Extrude처럼 작동하지만, Offset even옵션 (이하 Even extrude) 를 제공하지 않습니다.



이런 일이 왜 발생할까요?

Normal은 Face에 존재하지만, 실제로 움직이는 건 Point이기 때문입니다. 지난 시간에 알아본 도메인 문제입니다.



Point와 Edge의 Normal

그런데, 점, 선에 Normal을 연결하면 나오는 값은 Face의 Normal을 평균낸 것이 아닙니다.

Point와 Edge는 독자적인 Normal값을 가지고 있습니다.

대체로 Face의 노멀을 평균낸 것과 비슷하지만, 완전히 같지 않습니다.

이러한 성질 때문에.. 블렌더에서 Normal이 가질 수 있는 상태는 9가지가 되어버립니다..!

Face Normal

Point Normal

Edge Normal

Face Normal →
Edge Interpolation

Face Normal →
Point Interpolation

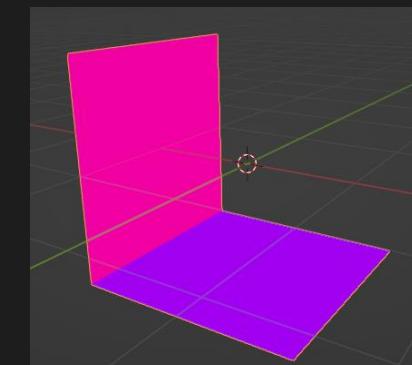
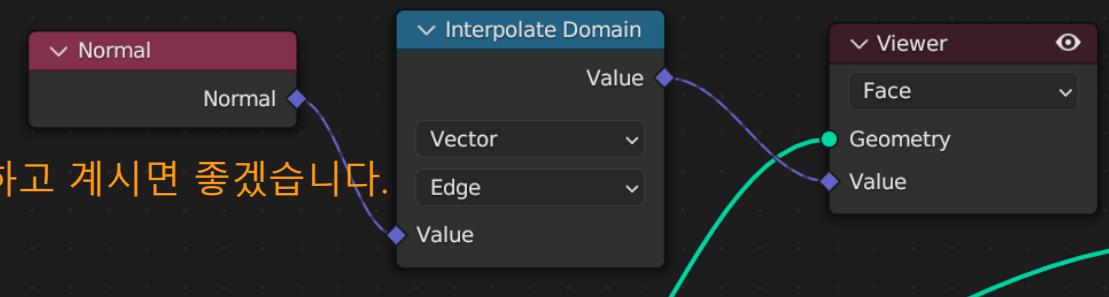
Edge Normal →
Point Interpolation

Edge Normal →
Face Interpolation

point Normal →
Face Interpolation

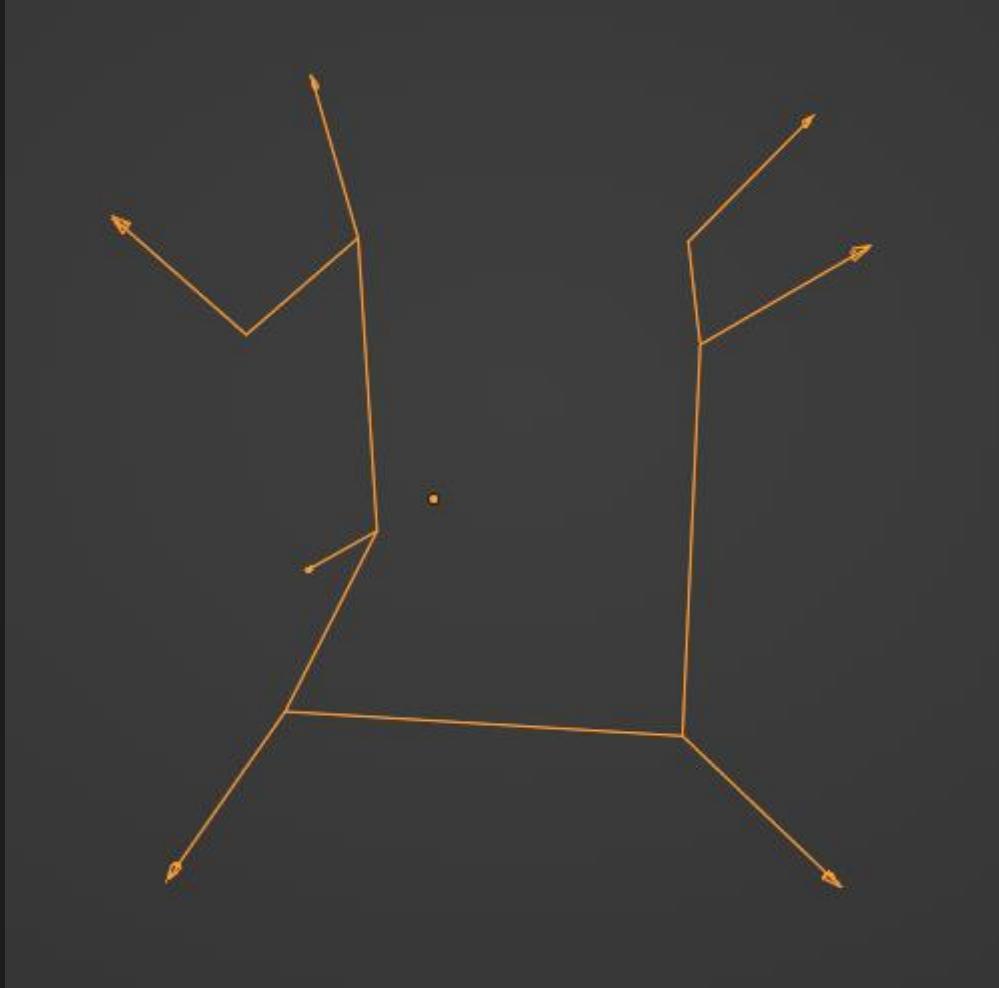
point Normal →
Edge Interpolation

이것들은 대부분 사용할 일이 없지만,
Normal은 여러가지 상태가 될 수 있다는 것을 인지하고 계시면 좋겠습니다.



Face가 없을 때의 Normal

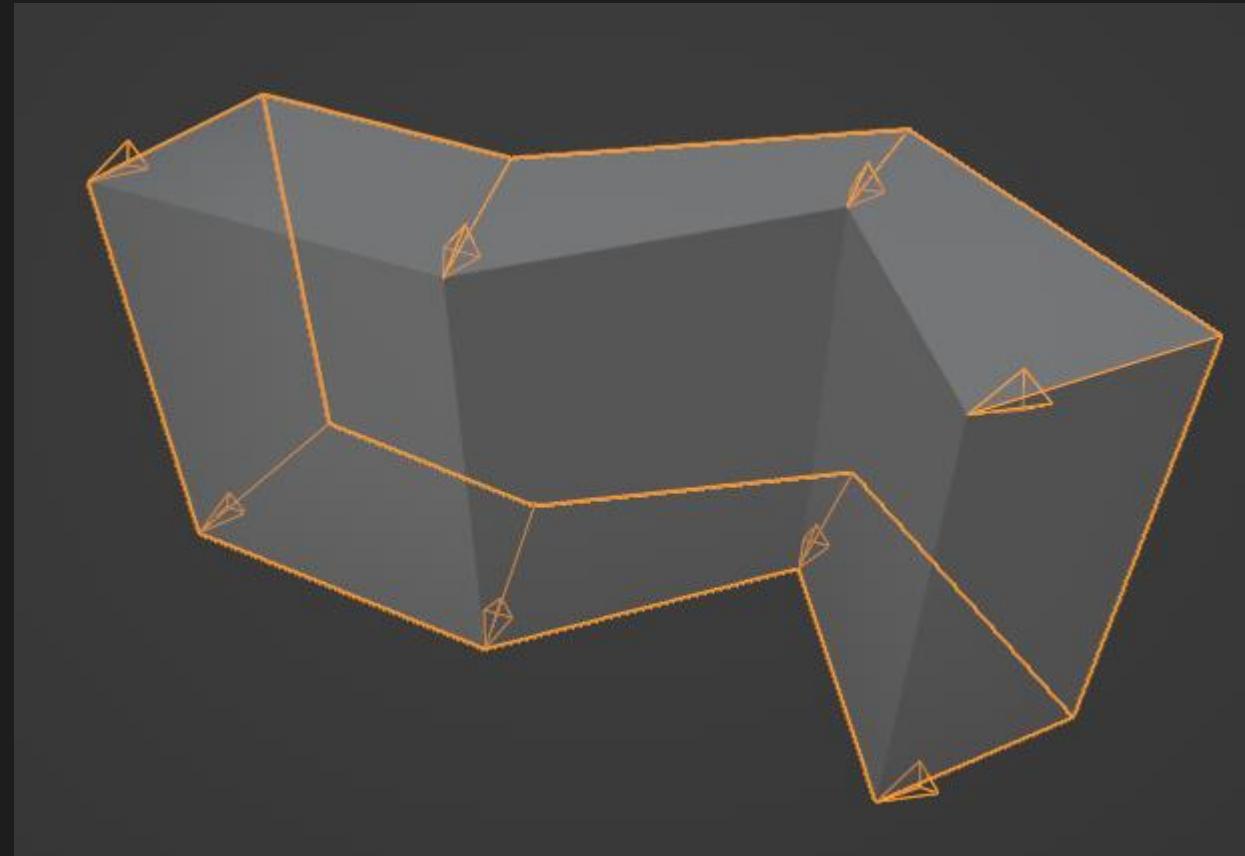
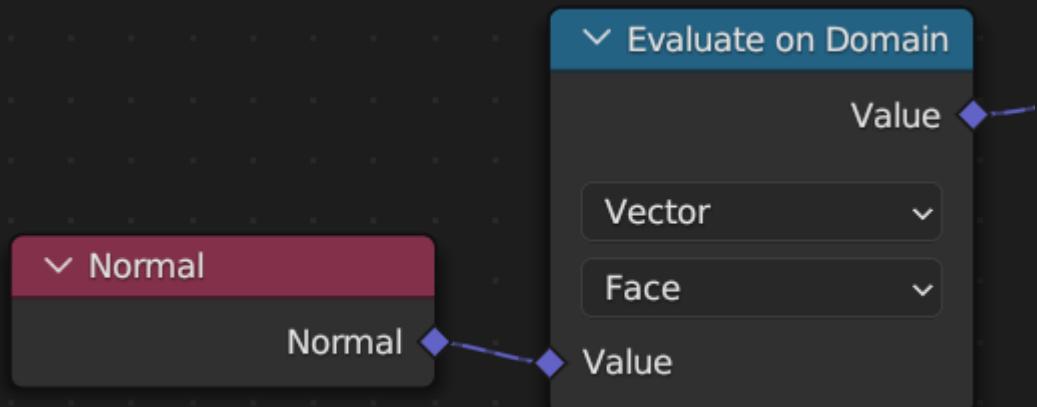
Face가 없으면 Edge와 Point는 Origin을 기준으로 뻗어나가는 방향을 Normal값으로 가집니다.
이상한 기준이지만, 값이 아예 없는것보단 나을 것이므로..



Even Extrude의 작동 원리

Extrude Mesh는 Face Normal을 가져와 Point를 움직입니다.

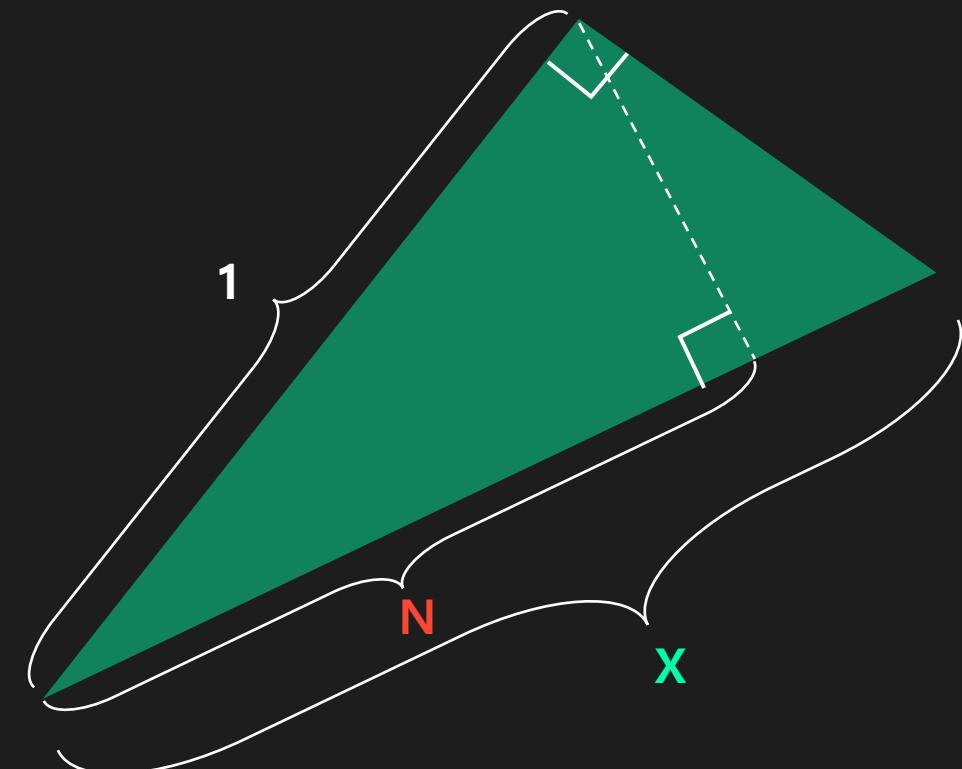
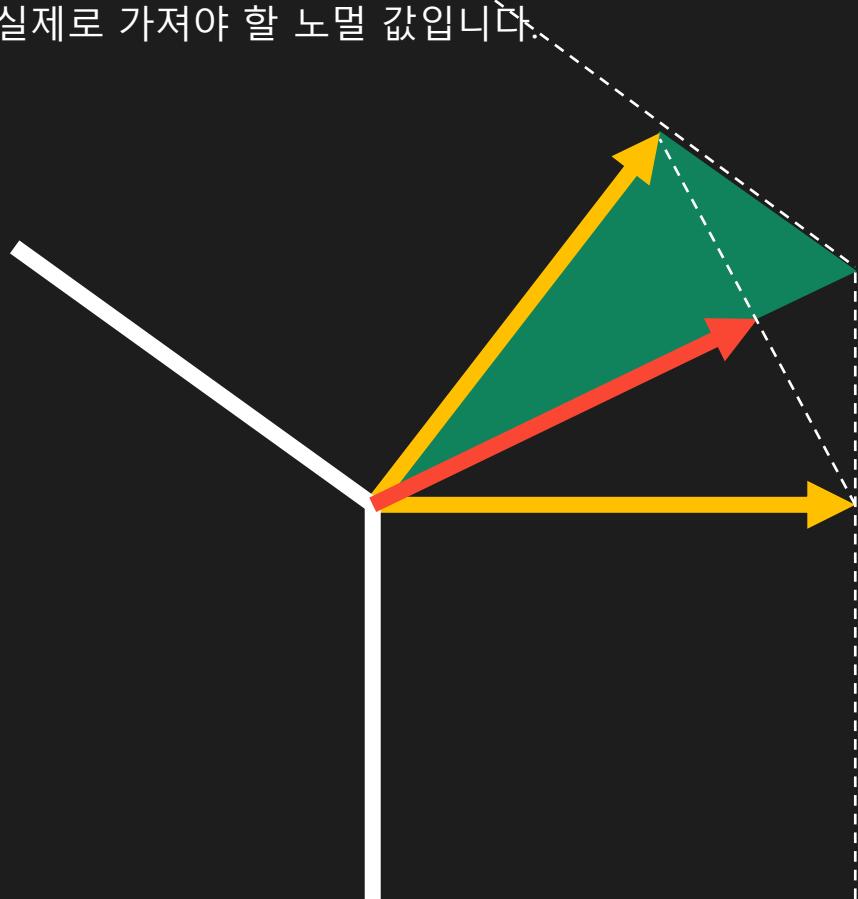
즉, 앞에서 봤던 Normal의 상태 중 Face Normal의 Point Interpolation상태입니다.



실제로 가져야 할 값은?

3차원에서 생각하면 어려우므로 2차원일 때만 살펴봅시다.

아래 그림에서, **N**이 현재의 인터플레이션 (노멀의 평균값)이고,
X가 실제로 가져야 할 노멀 값입니다.

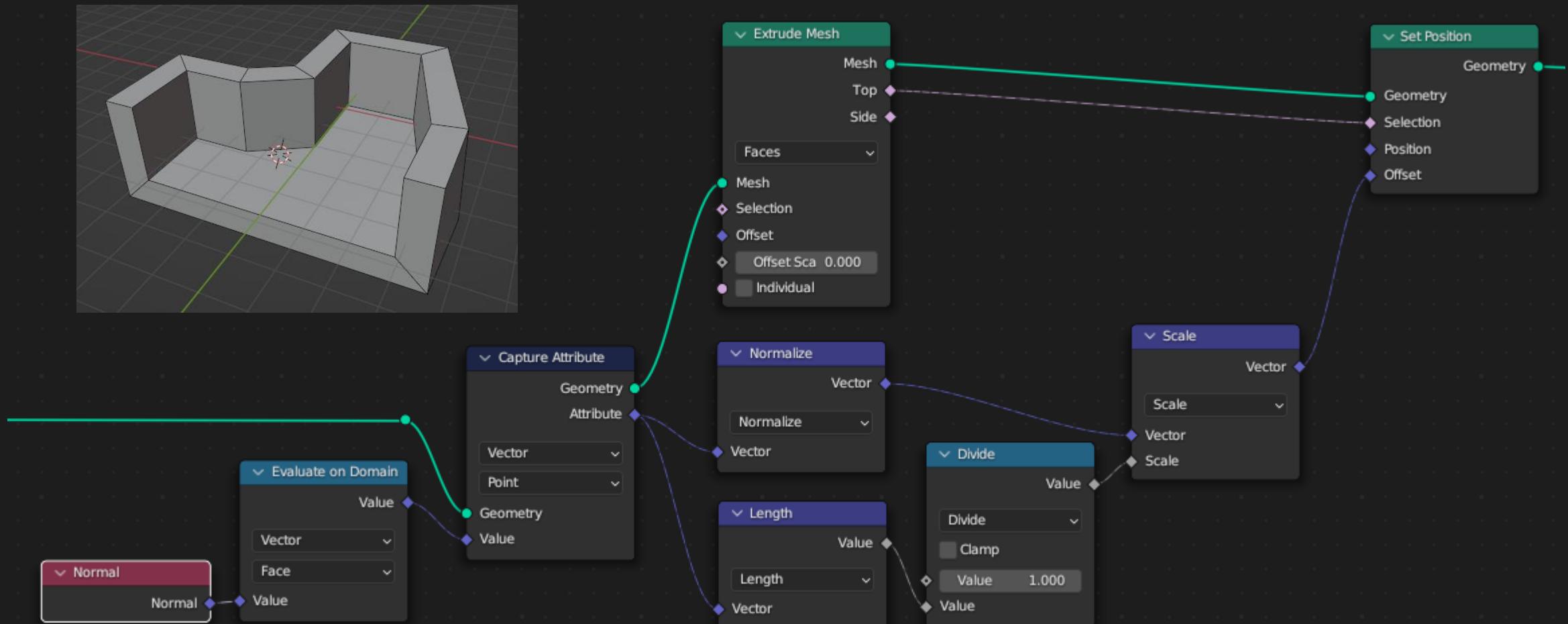


$$1 \times 1 = N \times X$$

$$X = 1/N$$

Even Extrude 구현

Normal을 앞에서 알아본 대로 바꿈으로써, Even Extrude를 구현하였습니다.
2D 기하학에 근거한 것이므로 3차원 지오메트리에서는 제대로 작동하지 않을 수 있습니다.
그런 경우엔 Separate Geometry로 분리해 보는 것도 좋습니다.



055강 Attribute의 전송 (3) : Proximity & Raycast

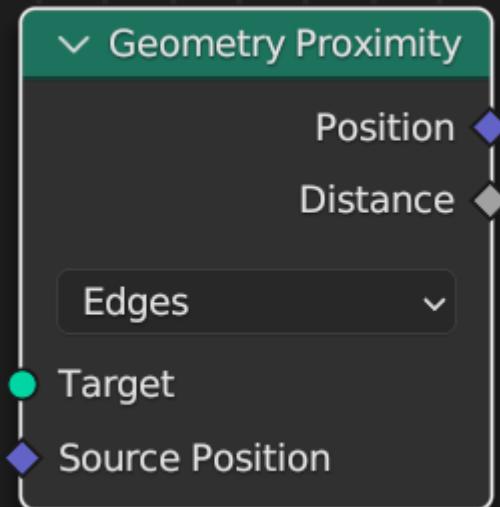
Geometry Proximity와 Raycast를 사용하는 법
Raycast를 이용하여 애니메이션을 만들기 : 녹아내리는 물체



Geometry Proximity

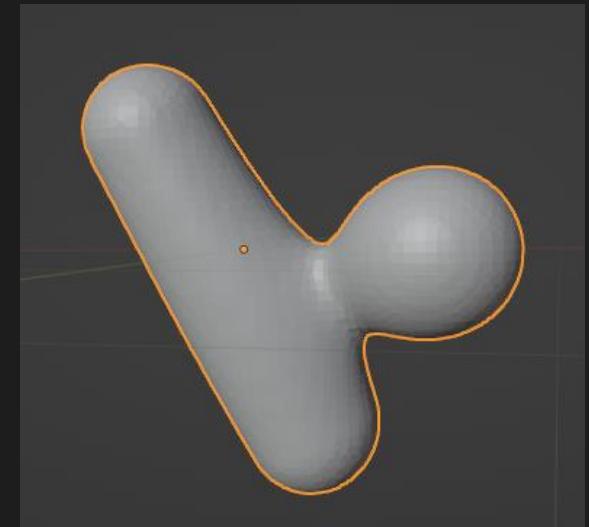
Geometry Proximity는 두 지오메트리 사이의 거리를 재는 데 특화된 노드입니다.

일부 기능은 Sample Nearest, Sample Nearest Surface와 중복되지만, 노드트리를 더 간결하게 만들 수 있습니다.



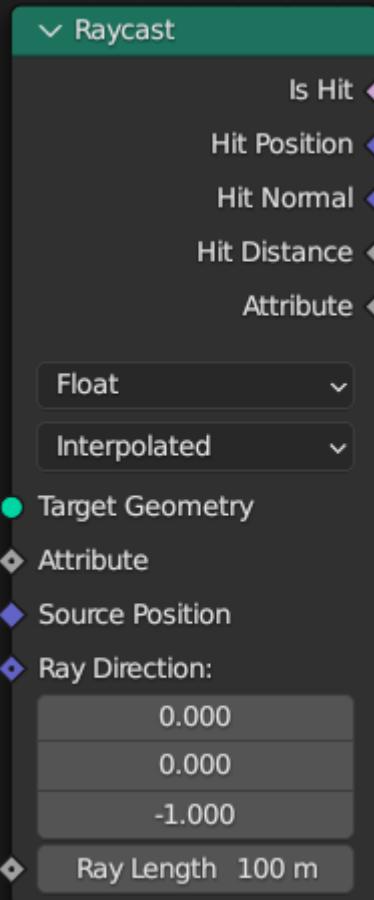
Edge모드는 Edge 위의 가장 가까운 거리를 잡니다.
Sample Nearest로는 불가능했던 기능입니다.

51강에서 Sample nearest 대신 Geometry Proximity를 쓰면
Edge를 이용한 메타볼도 만들 수 있습니다.



Raycast

Raycast는 가장 헷갈리는 노드 중 하나일 것입니다. 차근차근 알아봅시다.



Is Hit

Hit Position

Hit Normal

Hit Distance

Attribute

Float

Interpolated

● Target Geometry

◆ Attribute

◆ Source Position

◆ Ray Direction:

0.000

0.000

-1.000

◆ Ray Length 100 m

Raycast는 **현재의 지오메트리 위치**에서 ▶ **타겟 지오메트리**로 광선을 쏘아보냅니다.

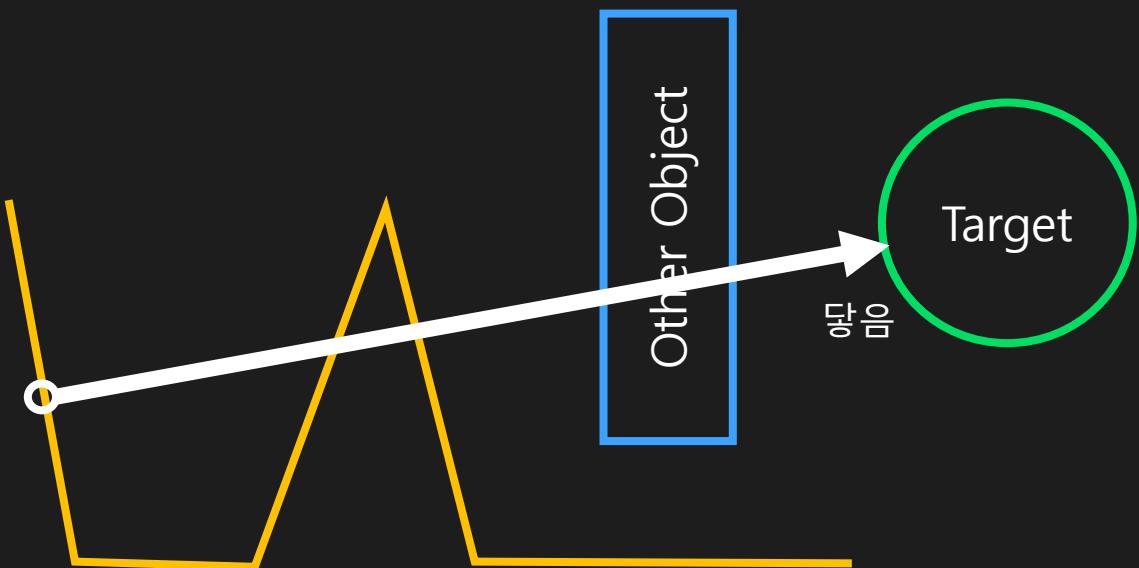
쏘아보낸 광선이 지오메트리에 닿는지 아닌지 / 닿은 부분의 위치와 노멀, /
닿은 부분의 특정 Attribute를 가져올 수 있습니다.

광선을 쏘는 방향은 Ray Direction으로 정합니다. 기본값은 (0,0,-1)로, 연직 아래 방향입니다.

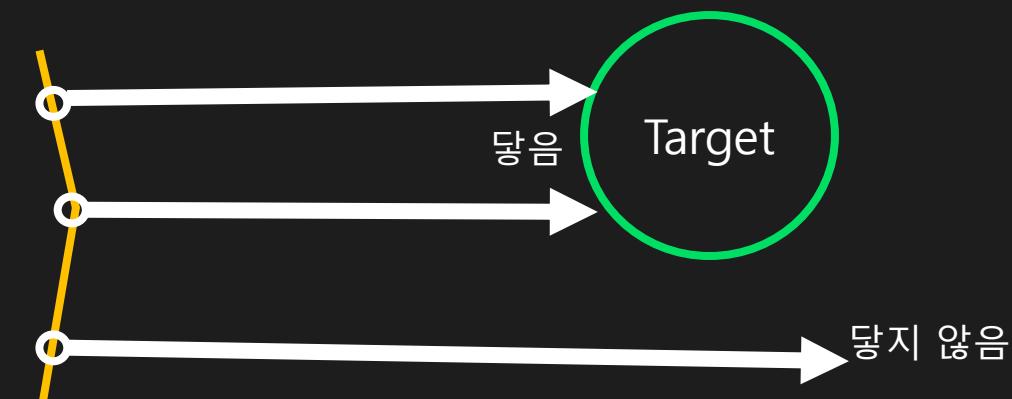
Ray Length는 탐지 범위를 조절합니다.

'닿는다' 는 개념

Raycast가 쏘아보낸 광선은 자기 자신 혹은 제 3의 오브젝트와는 상호작용하지 않습니다.
모두 뚫고 지나가서 광선의 경로에 타겟 지오메트리가 닿는지만 체크합니다.



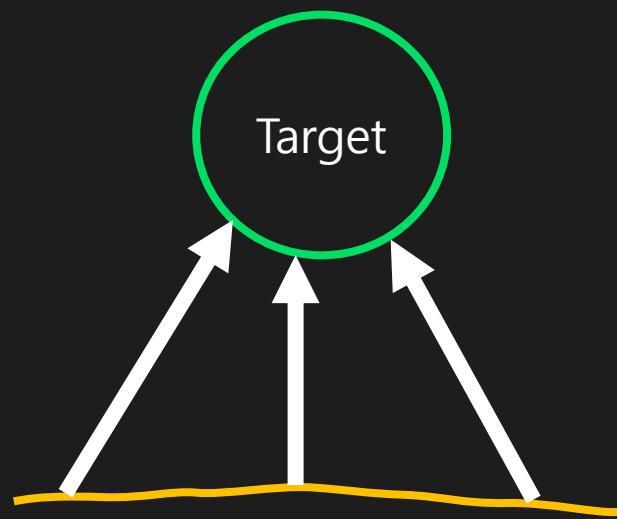
다른 오브젝트 혹은 자기 자신에 가려지는 것은
생각하지 않습니다.



광선이 지나가는 경로에
타겟 지오메트리가 있는지를 체크합니다.

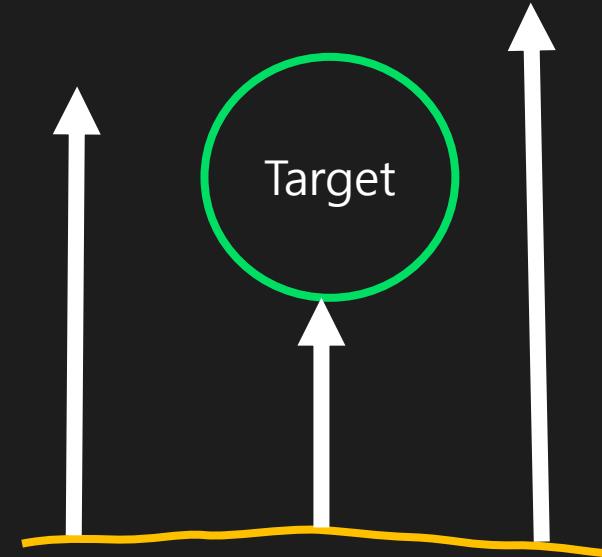
Sample Nearest와의 차이점

Sample Nearest (surface)



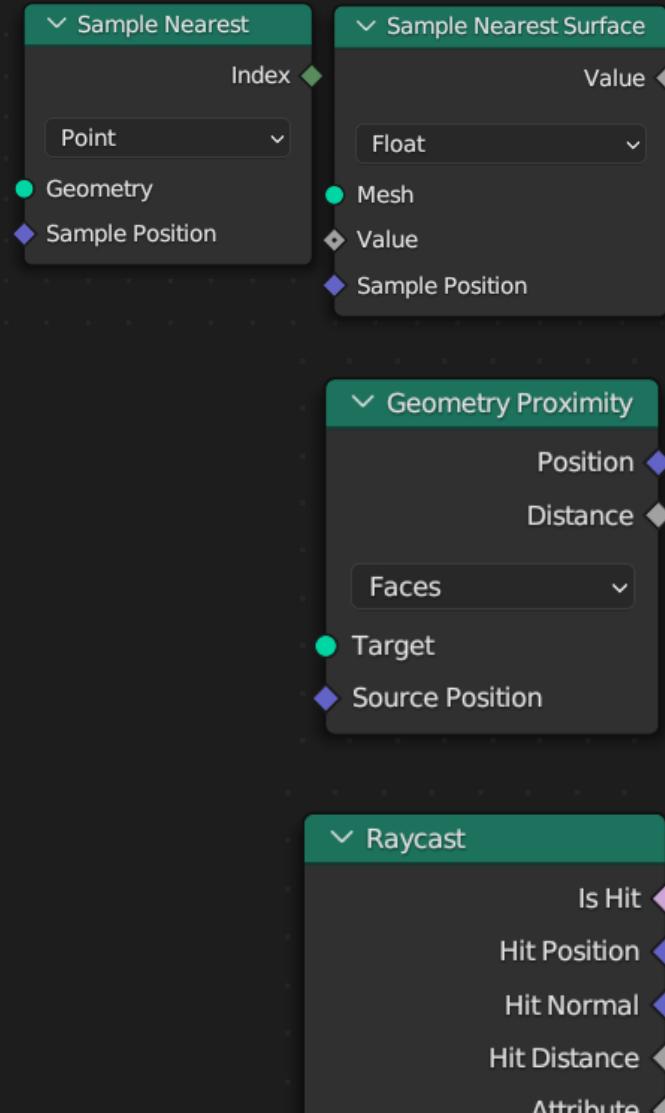
Sample Nearest는 항상 가장 가까운 지점의 값을 가져옵니다.
따라서 타겟을 향하는 방향은 임의로 결정되며,
어느 점에서나 값이 존재합니다.

Raycast (Ray direction (0,0,1))



Raycast는 먼저 방향을 결정하므로, 타겟에 닿을 수도 있고
닿지 않을 수도 있습니다.
따라서 닿지 않은 점에서는 거리값이 존재하지 않습니다.
닿지 않는 경우 is Hit 이 False가 되고 Hit Position은 0이 됩니다.

Raycast vs Sample Nearest vs Proximity

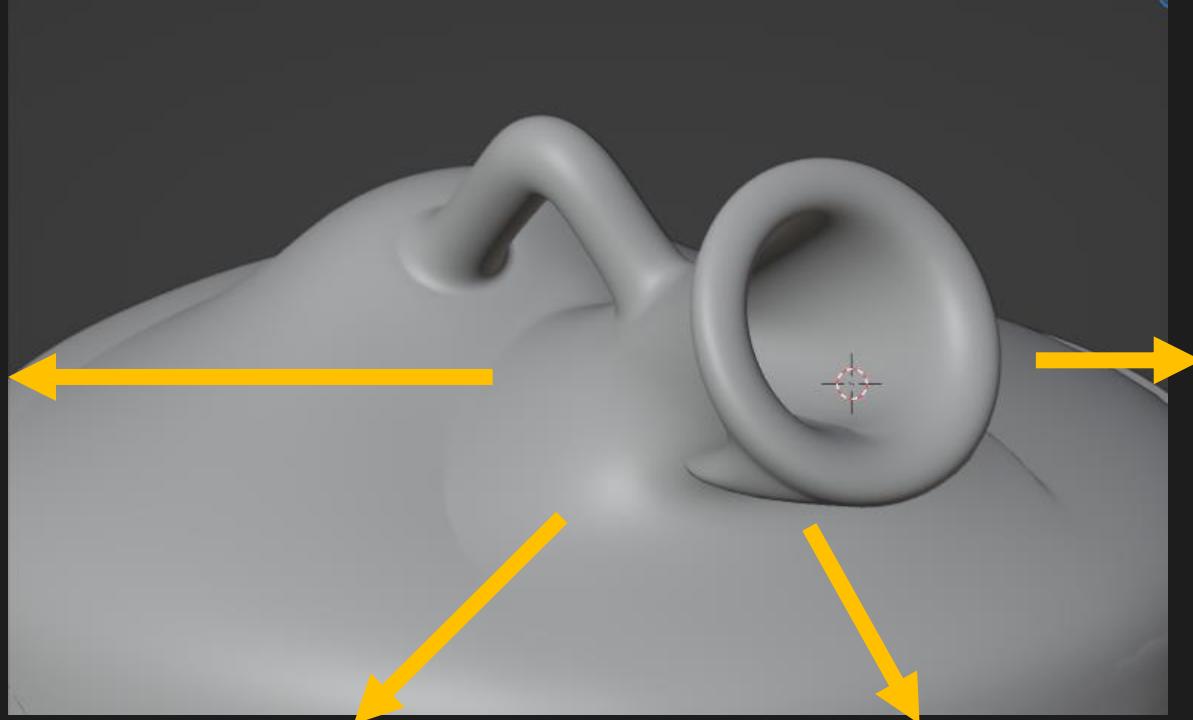


Sample Nearest (surface) : 가장 가까운 지점의 Attribute를 가져옵니다.

Geometry Proximity : 가장 가까운 위치를 찾는 데 특화되어 있습니다.

Raycast : 타겟 지오메트리와의 관계가 '방향'과 관련있을 때 사용합니다.

'녹아내리는 효과'



물체가 녹아 바닥에 닿으면 사방으로 퍼질 것입니다.

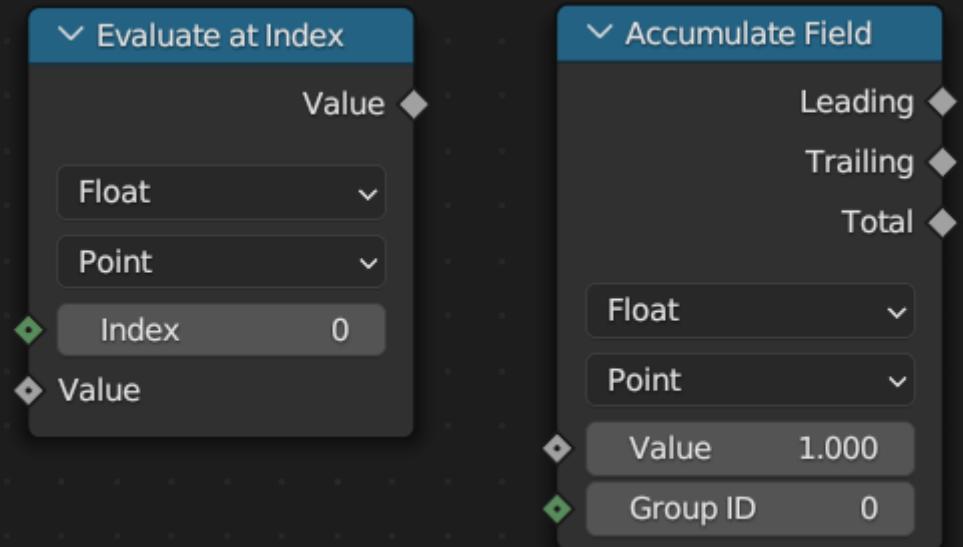
'바닥에 닿는다'는 나중에 생각하고 '사방으로 퍼진다'만 생각해봅시다.

즉 Z축 위치에 따라 자신의 노멀 방향으로 뻗어나가게 해 봅시다.

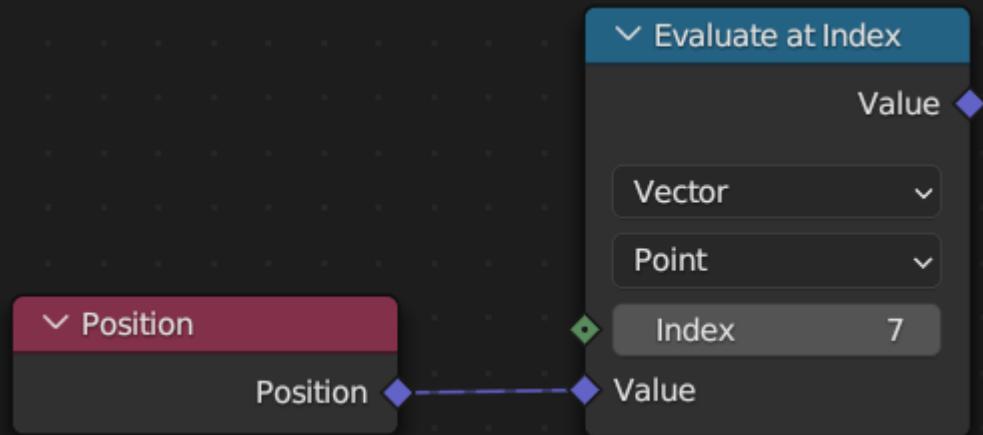
056강 정보의 변경

Accumulate Field

Evaluate at Index

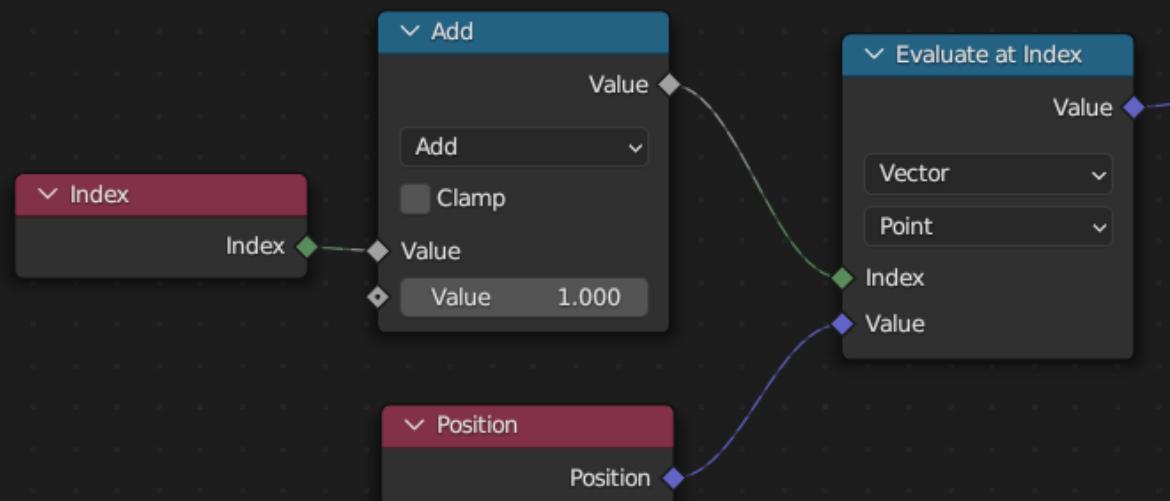


Evaluate at Index



Evaluate at Index 노드는 특정 인덱스의 정보를 가져옵니다.

오른쪽처럼 연결하면, 모든 점에서 7번 점의 위치정보를 가져옵니다.



오른쪽처럼 연결하면 자기 자신보다 1 큰 번호의 위치를 가져옵니다.

즉 0번은 1번의 정보를,
1번은 2번의 정보를.
2번은 3번, 을 가져옵니다.

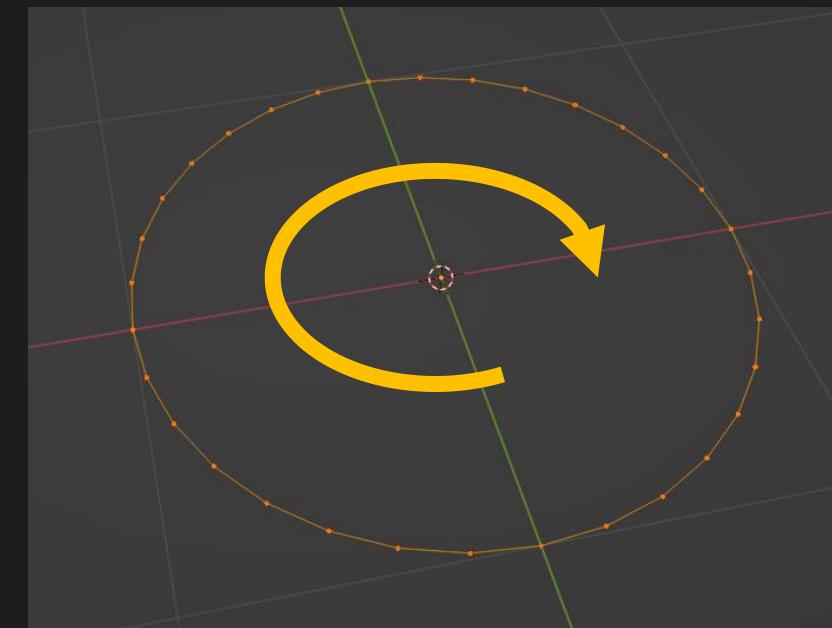
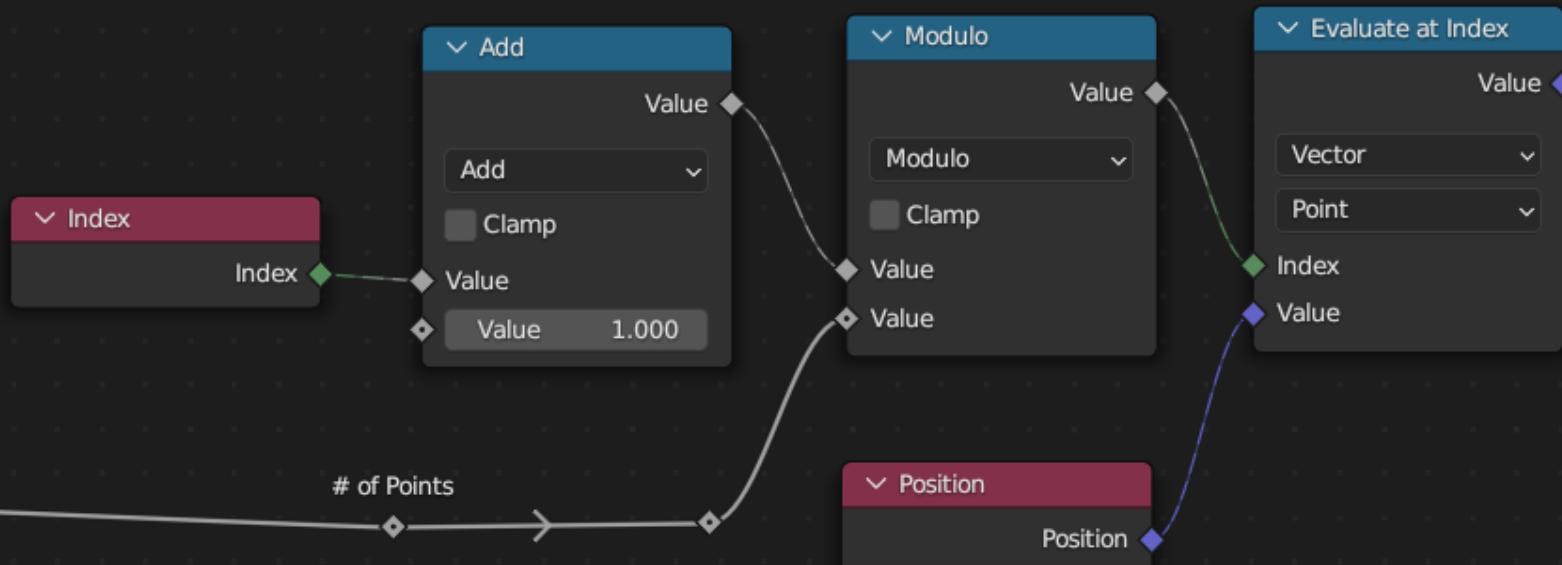
※이 노드의 이름은 이전에는 **Field at Index**였습니다.

인덱스 순환

인덱스를 하나씩 건너뛰어서 불러오면, 맨 마지막 번호의 점은 존재하지 않는 인덱스를 받게 됩니다.
이 마지막 점이 맨 처음 점 (0번)을 받게 하려면, Modulo를 사용하면 됩니다.

아래와 같이 연결하면, 전체 점 개수가 n 개일 때,
마지막 점 ($n-1$ 번 점)은

- ▶ 1을 더해서 n 이 되고
- ▶ 이를 n 으로 나눈 나머지 (modulo n) 는 0이므로,
- ▶ $n-1$ 번 점이 0번 점에 대응되게 됩니다.



Accumulate Field

▼ Accumulate Field

- Leading
- Trailing
- Total

Vector

Point

◆ Value:

1.000
1.000
1.000

◆ Group ID 0

Accumulate Field 노드는 입력한 Attribute의 누적값을 계산합니다.

단순히 Attribute의 합을 구하기 위해서라면, Attribute Statistic 노드를 사용할 수 있습니다.

Accumulate Field는 이것과는 두 가지가 다릅니다.

1. Accumulate Field는 더하는 과정의 값을 얻을 수 있습니다.
2. Group ID를 이용해서 그룹별로 더할 수 있습니다.

▼ Attribute Statistic

- Mean
- Median
- Sum
- Min
- Max
- Range
- Standard Deviation
- Variance

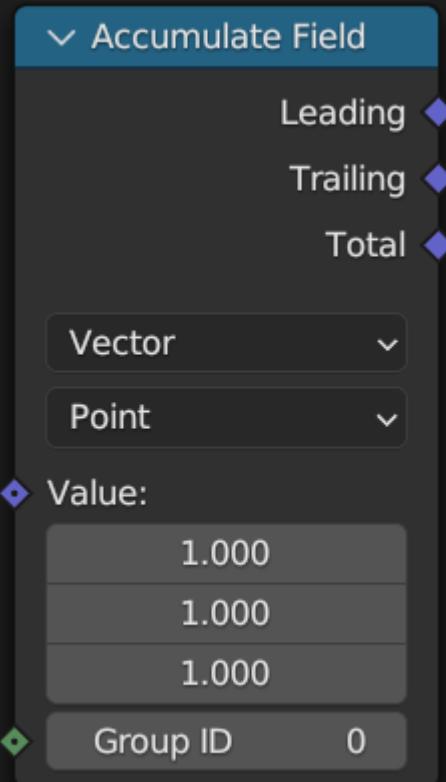
Vector

Point

- Geometry
- ◆ Selection
- ◆ Attribute



Leading, Trailing



Accumulate Field는 인덱스 순서에 따라 Attribute를 차례대로 더합니다.

Index	Value	Leading	Trailing
0	A	A	0
1	B	A+B	A
2	C	A+B+C	A+B
3	D	A+B+C+D	A+B+C
4	E	A+B+C+D+E	A+B+C+D
...

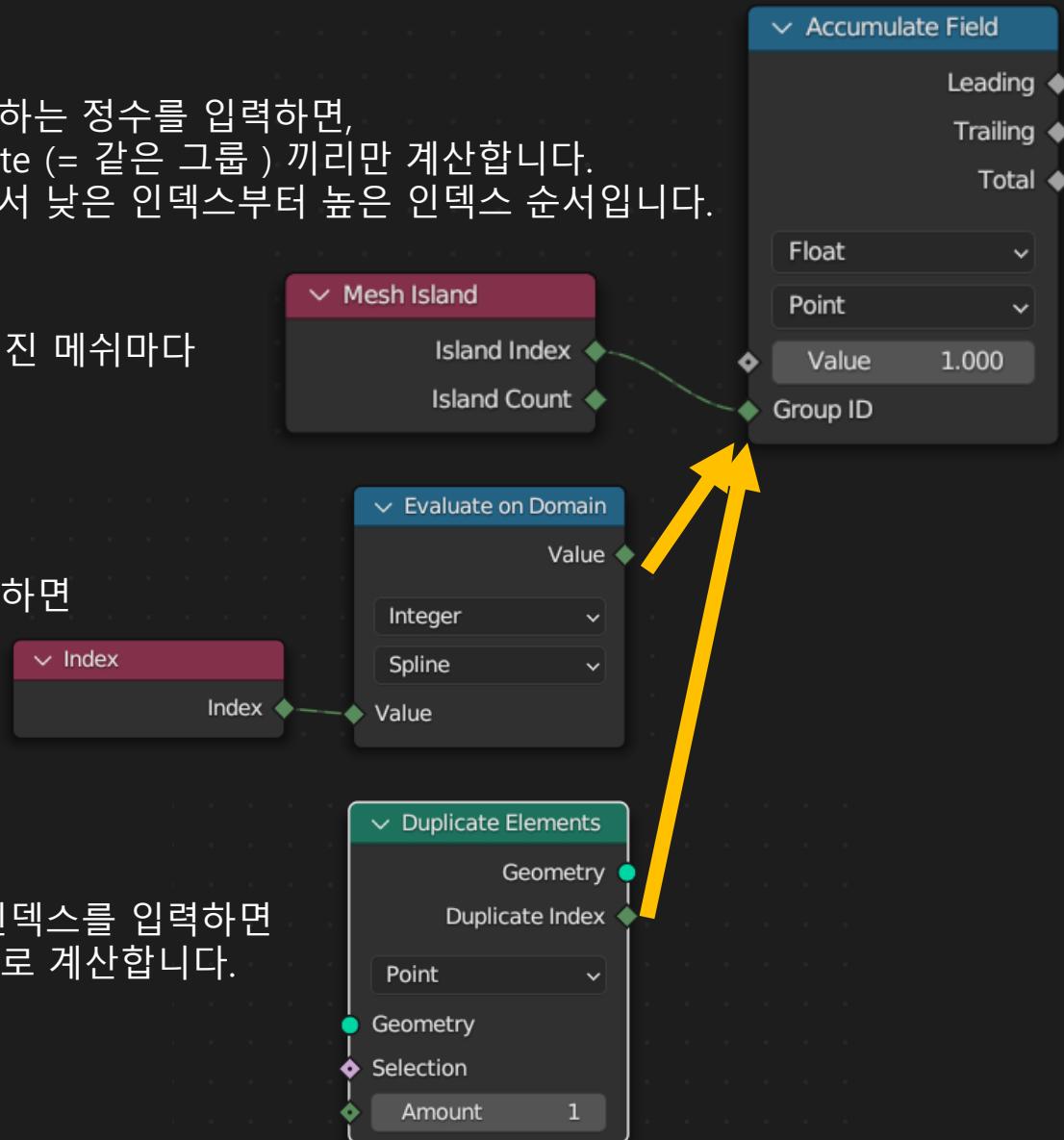
그룹 분리

Group ID에 그룹을 구분하는 정수를 입력하면,
같은 정수를 갖는 Attribute (= 같은 그룹) 끼리만 계산합니다.
더하는 순서는 그룹 내에서 낮은 인덱스부터 높은 인덱스 순서입니다.

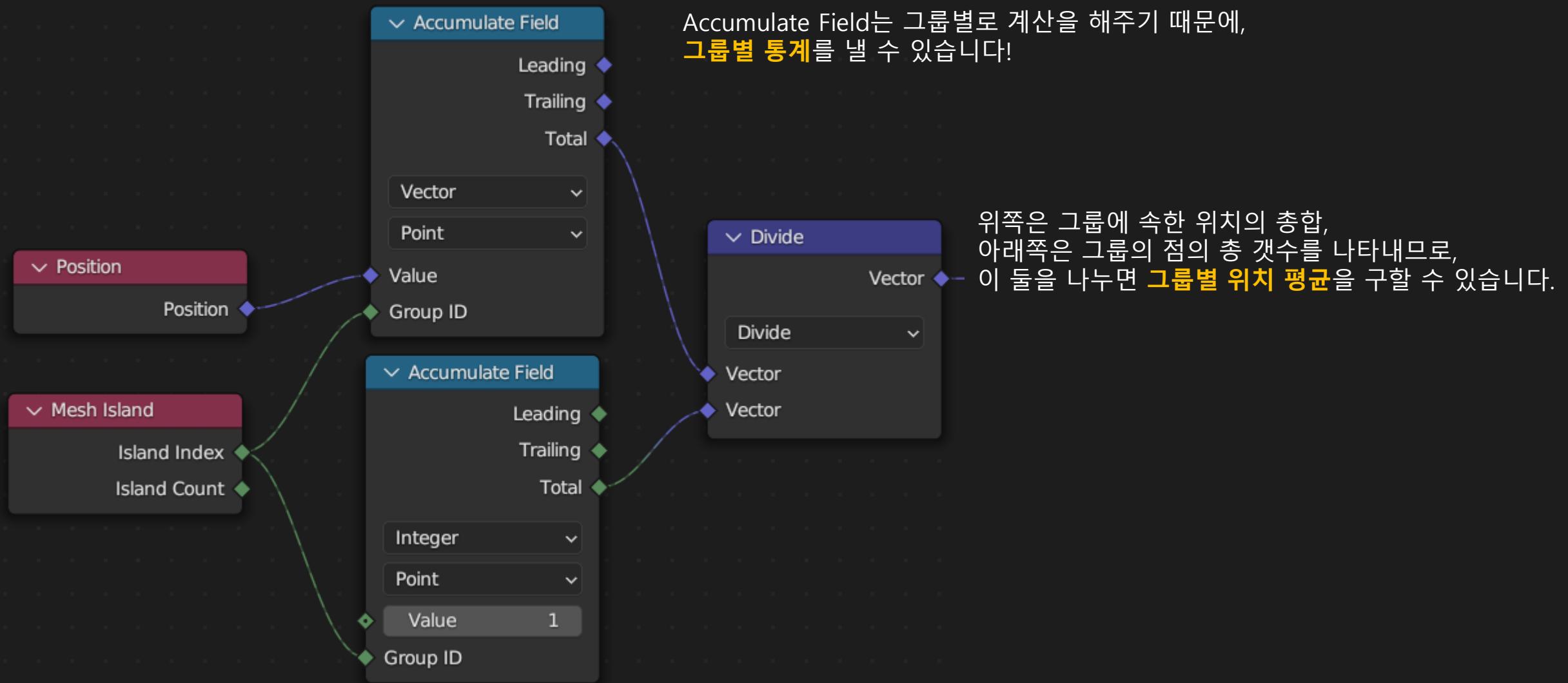
Mesh Island를 입력하면 떨어진 메쉬마다
따로따로 계산합니다.

스플라인의 인덱스를 입력하면
스플라인마다 계산합니다.

Duplicate Elements의 인덱스를 입력하면
복제된 Elements마다 따로 계산합니다.



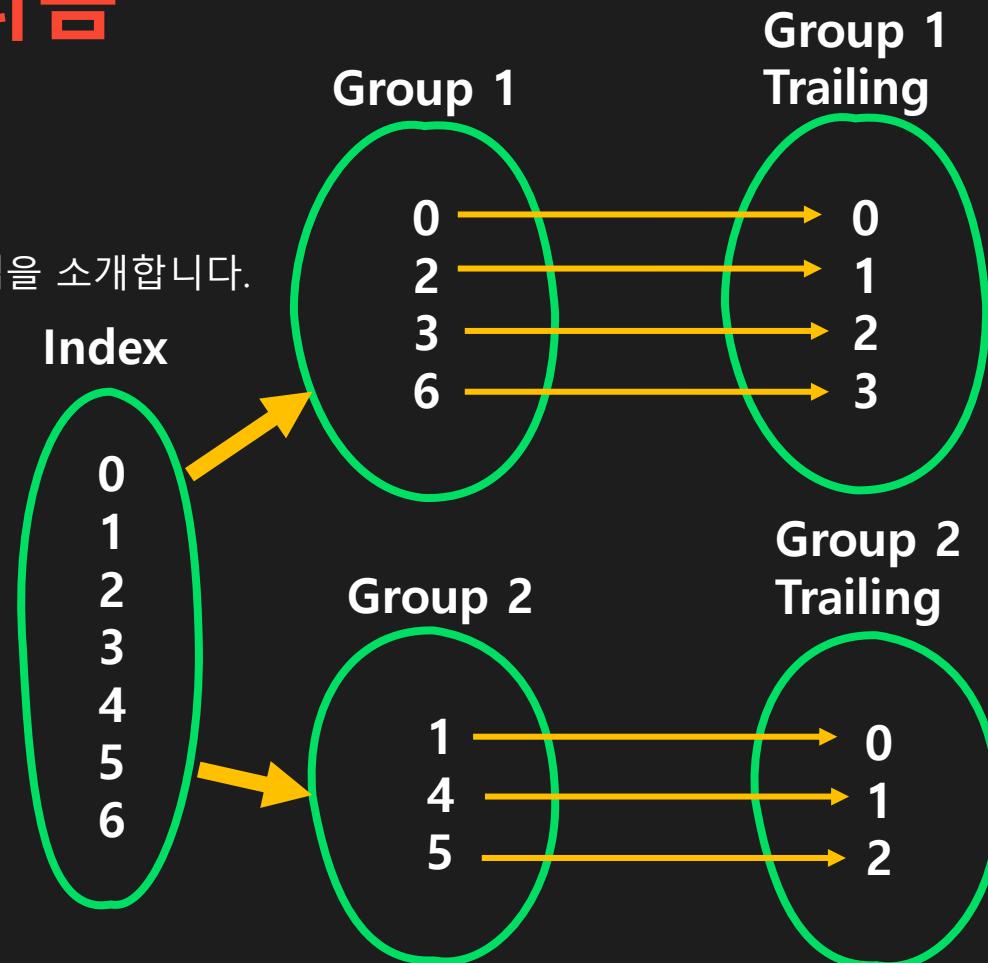
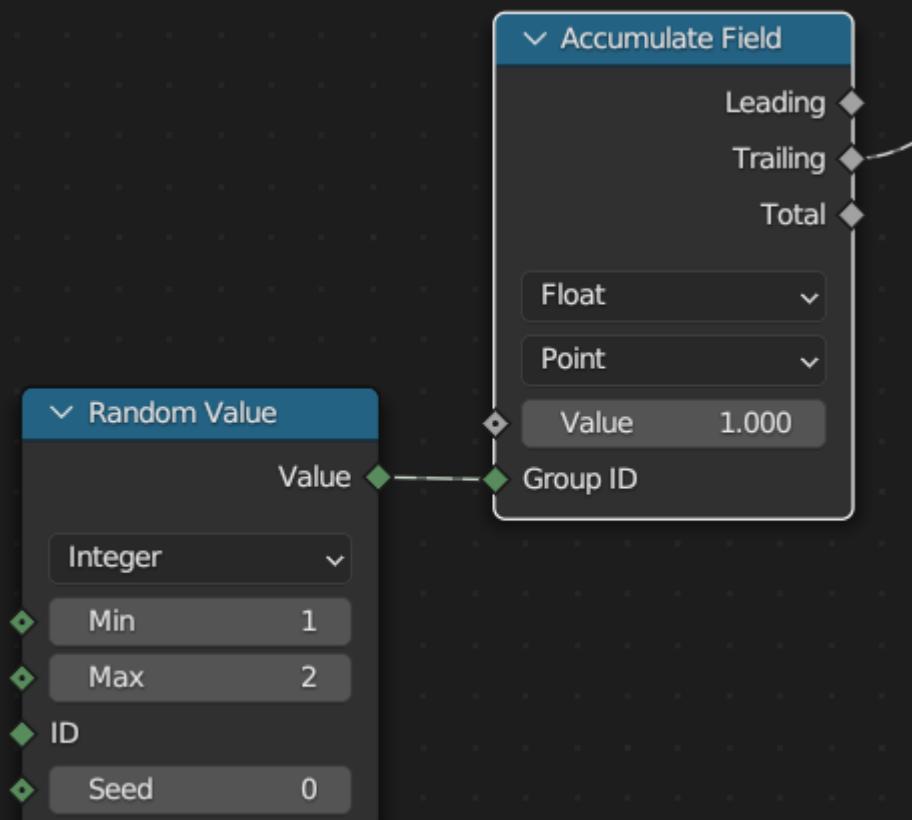
그룹 분리를 이용한 그룹별 정보의 합과 평균



랜덤 그룹을 이용한 섞기 알고리즘

Random Value로는 인덱스를 온전히 섞을 수 없습니다.
중복되는 숫자가 나오기 때문입니다.

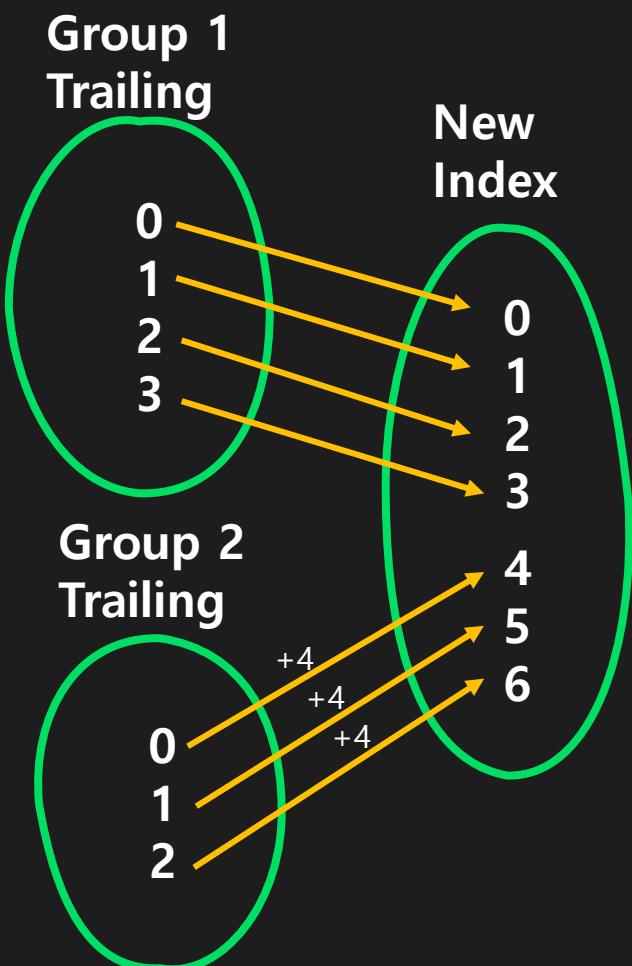
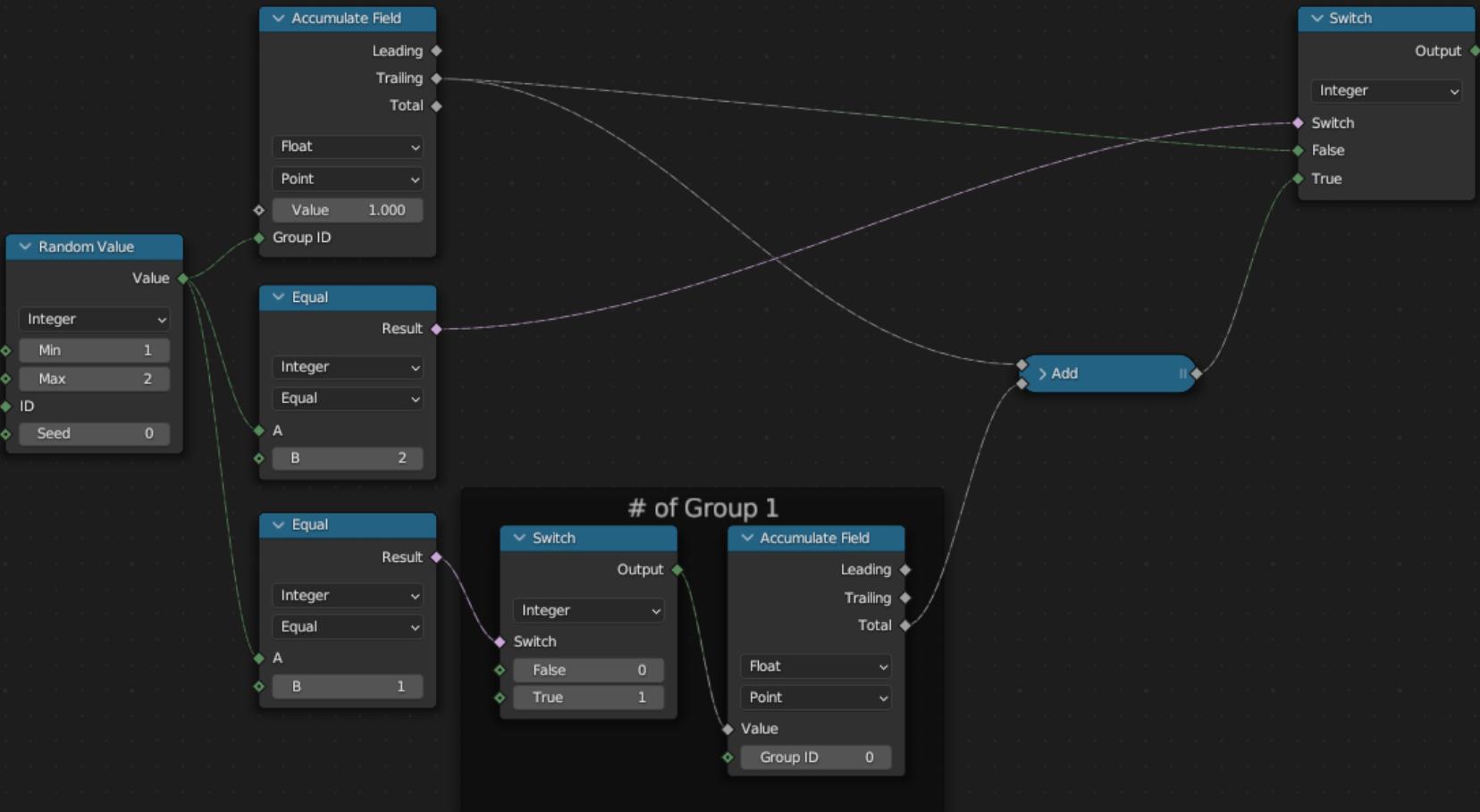
여기서는 Accumulate Field를 이용하여 인덱스를 중복없이 섞는 방법을 소개합니다.
완벽한 섞기 알고리즘은 아니지만 충분히 사용 가능합니다.



오른쪽처럼, 점을 임의로 두 그룹으로 나누어 1을 더하면,
각각의 점이 얻는 Trailing 값이 마치 순서가 섞인 인덱스처럼 보입니다.

랜덤 그룹을 이용한 섞기 알고리즘

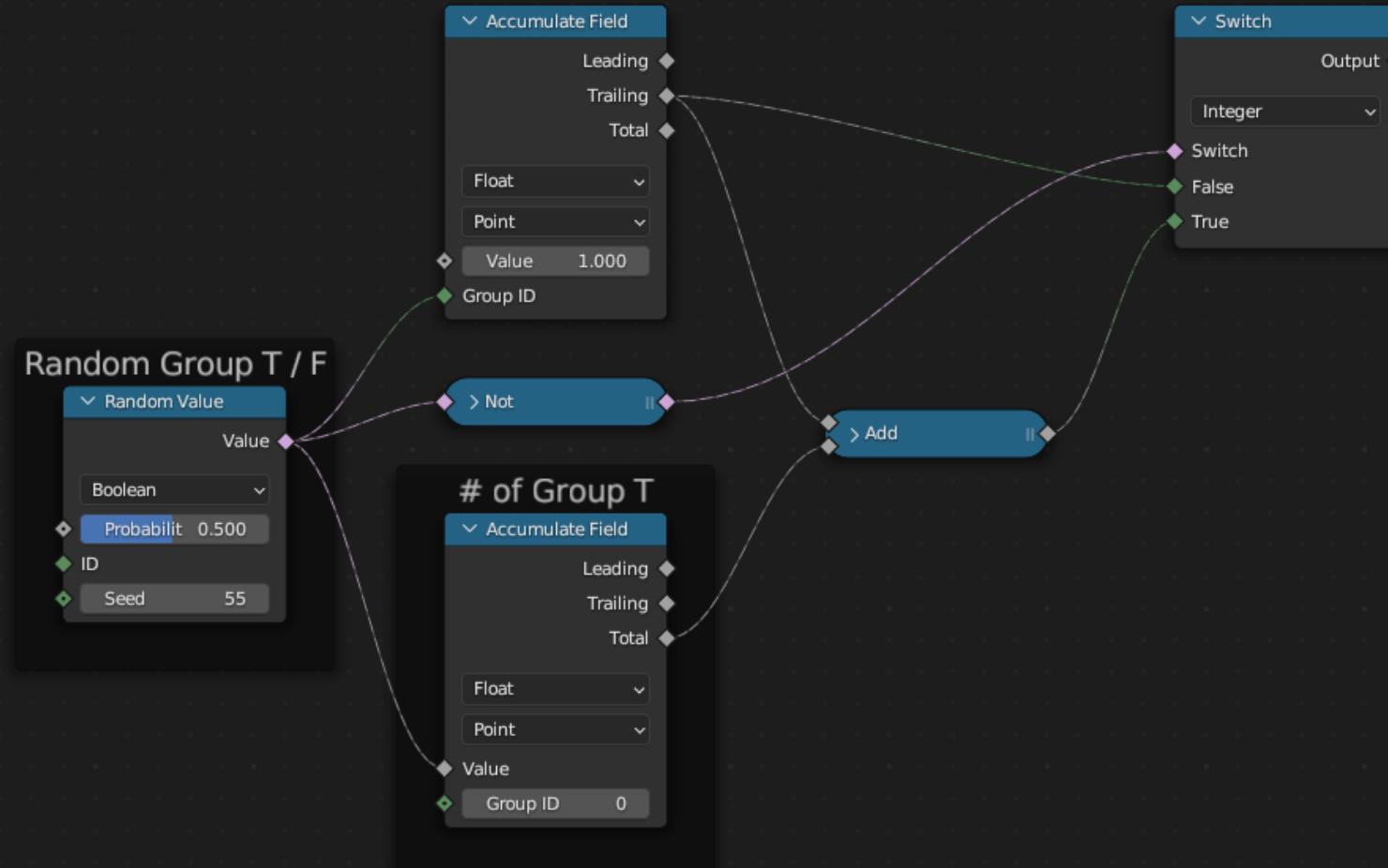
그 다음에, 그룹 2에 그룹 1의 개수를 더해주면 0번부터 6번까지의 모든 숫자가 나옵니다!



랜덤 그룹을 이용한 섞기 알고리즘

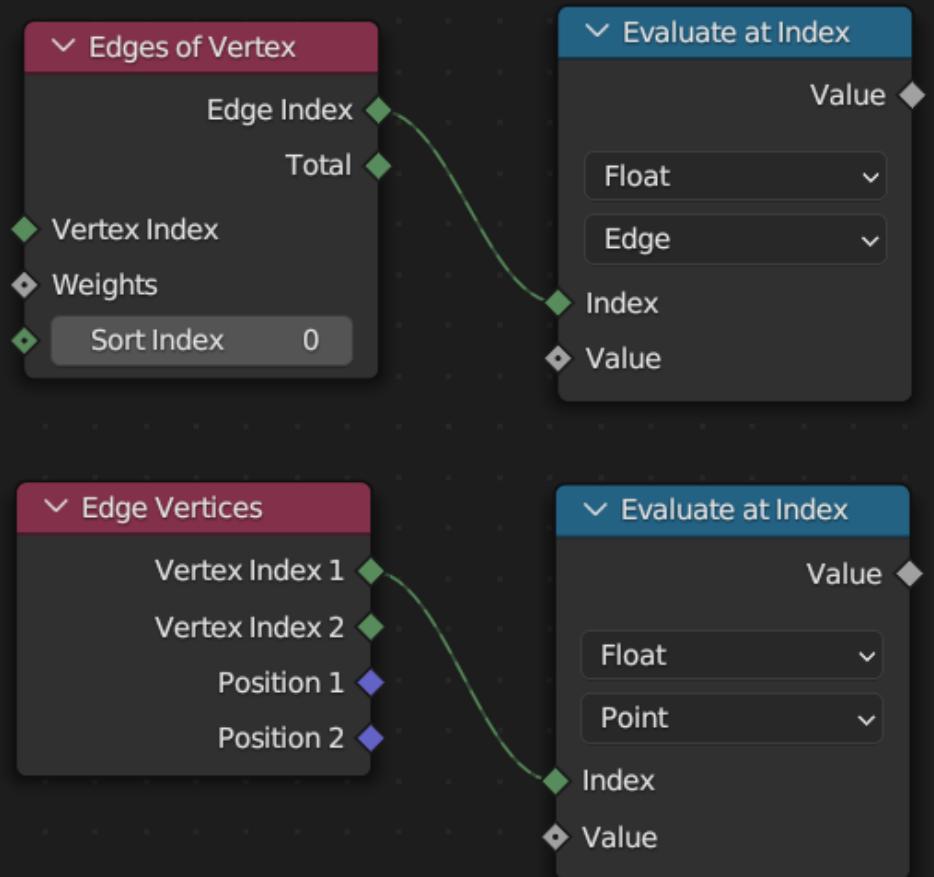
불린을 이용하여 단순화하면 이렇게 됩니다.

이렇게 보면 상당히 난해하지만 기본 골자는 앞과 동일합니다.

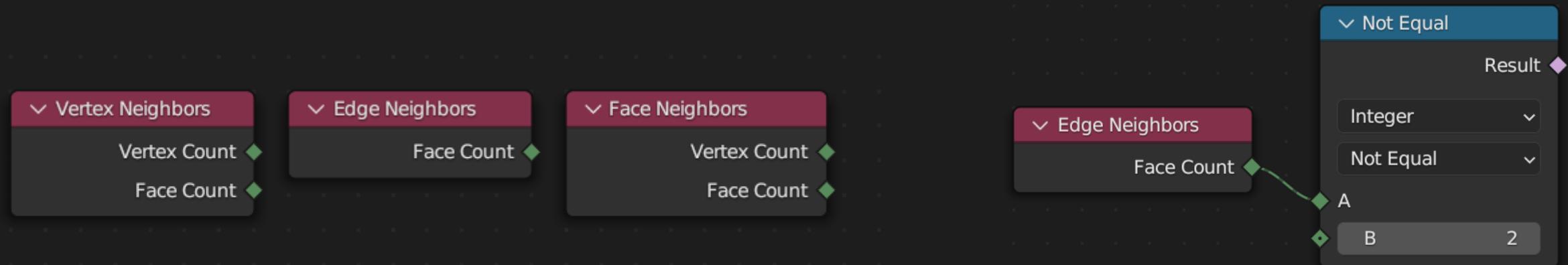


057강 Topology (1)

지오메트리를 따라 자신의 주변 점/선/면에 접근하는 법



Reminder : Neighbors



Vertex / Edge / Face Neighbors 노드는
점, 선, 면에 연결된 이웃이 '몇 개인지' 출력하고,
그 이웃의 정보를 얻을 수는 없습니다.

물론 그것도 활용 가능합니다.
일반적으로 엣지에 붙은 면의 개수는 2개이므로,
2개가 아닌 엣지를 찾으면 지오메트리의 끝부분을 찾을 수 있습니다.

Reminder : 지오메트리 노드의 연결방식

지오메트리 강의 앞부분에서 이야기드린 것처럼, 대개는

- Attribute를 변환하여 사용
- Attribute를 해석해서 선택

의 두 가지 연결 방식을 사용합니다.

Evaluate at Index를 이용하면, 인덱스를 통해서 Attribute를 얻는 새로운 연결 방식을 사용할 수 있습니다!

Attribute ▶ 계산 ▶ Attribute

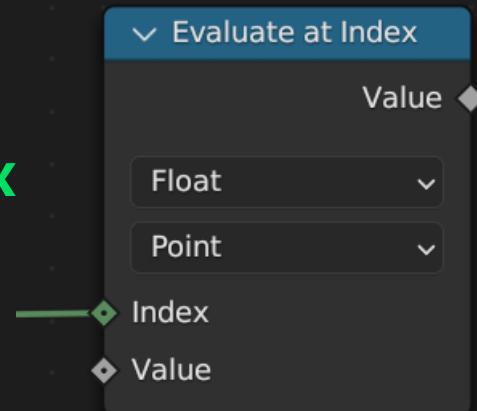
Position ▶ +1 ▶ 이동한 Position..

Attribute ▶ Compare ▶ Select

Position ▶ Greater than 1 ▶ Select ...

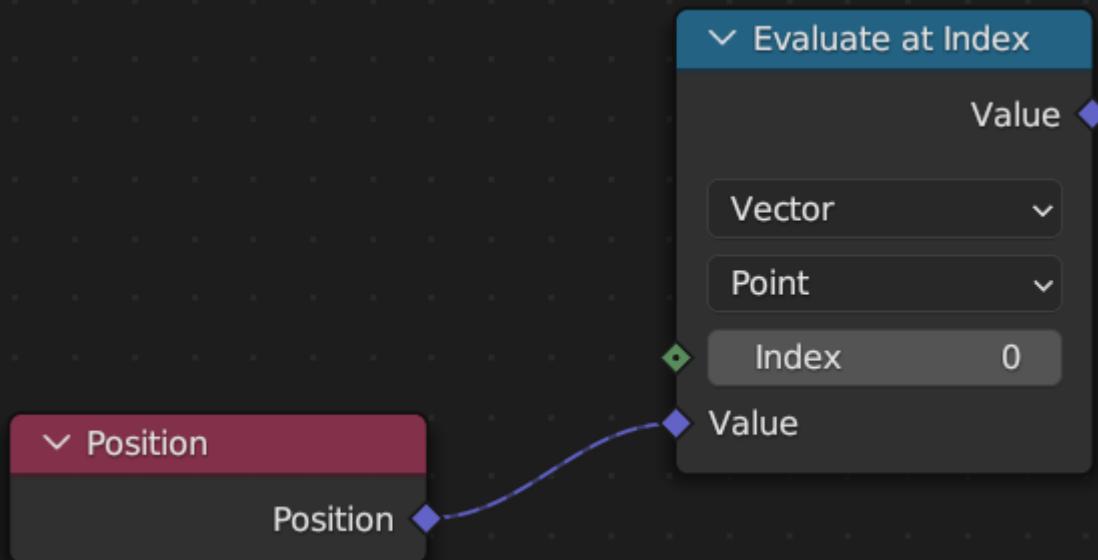
NEW!

Index ▶ Evaluate at Index



다른 부분의 정보에 접근하는 방법

Evaluate at Index를 통하여 다른 인덱스의 정보를 가져올 수 있습니다.
아래와 같이 연결하면 '0번 점의 위치' 가 됩니다.



만약, Index를 알 수 있다면, 지오메트리의 정보 접근이 훨씬 자유로워집니다.

Topology Nodes

여기, 자신 주변의 인덱스를 가져오는 노드들이 있습니다.

Vertex

Edge

Face

Face
Corner

Vertex

Edge

Face

Face
Corner

	<p>Edge Vertices</p> <ul style="list-style-type: none">Vertex Index 1Vertex Index 2Position 1Position 2		<p>Vertex of Corner</p> <ul style="list-style-type: none">Vertex IndexCorner Index
<p>Edges of Vertex</p> <ul style="list-style-type: none">Edge IndexTotalVertex IndexWeightsSort Index 0			<p>Edges of Corner</p> <ul style="list-style-type: none">Next Edge IndexPrevious Edge IndexCorner Index
			<p>Face of Corner</p> <ul style="list-style-type: none">Face IndexIndex in FaceCorner Index
<p>Corners of Vertex</p> <ul style="list-style-type: none">Corner IndexTotalVertex IndexWeightsSort Index 0		<p>Corners of Face</p> <ul style="list-style-type: none">Corner IndexTotalFace IndexWeightsSort Index 0	<p>Offset Corner in Face</p> <ul style="list-style-type: none">Corner IndexCorner IndexOffset 0

Topology Nodes



토플로지 관련 노드들은 사용법을 확실히 숙지하지 않으면 연결이 어렵습니다.

point 도메인이 아닌 것들이 섞여 있으므로, 각 소켓이 속하는 도메인을 확실히 해야 합니다.

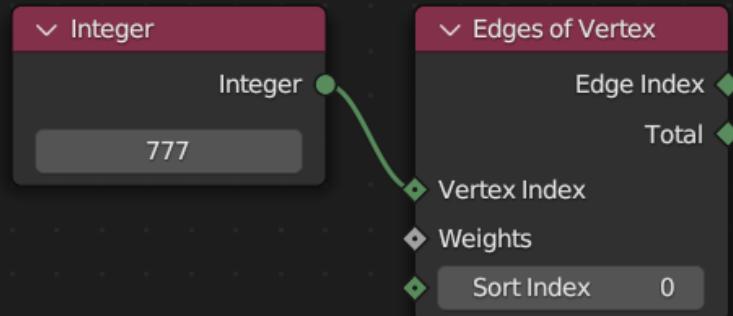
예컨대 Edges of Vertex는 **Vertex**를 받아 **Edge**의 정보를 내보냅니다.

※ 토플로지 노드들은 Index를 출력하기 때문에, 독자적으로는 사용하기 어렵습니다. 대부분 Evaluate at Index와 같이 사용합니다.

Weights는 출력값과 관련됩니다. Edges of Vertex의 Weight는 Vertex가 아니라 Edge의 가중치입니다.
기본적으로 Weight가 작은 값부터 출력됩니다.

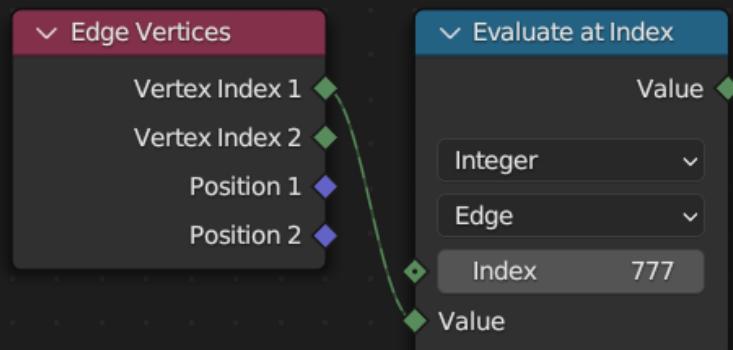
Index 입력을 별도로 하지 않은 경우 기본 인덱스가 꽂혀 있는 것처럼 계산됩니다.

Edge Vertices에 엣지 바꾸기



토플로지 노드들은 인덱스 입력이 있어서, 자신의 이웃의 정보 뿐만 아니라 인덱스를 특정해서 가져올 수 있습니다.

예를 들어, 오른쪽처럼 연결하면 777번 점에 연결된 엣지 번호를 얻을 수 있습니다.

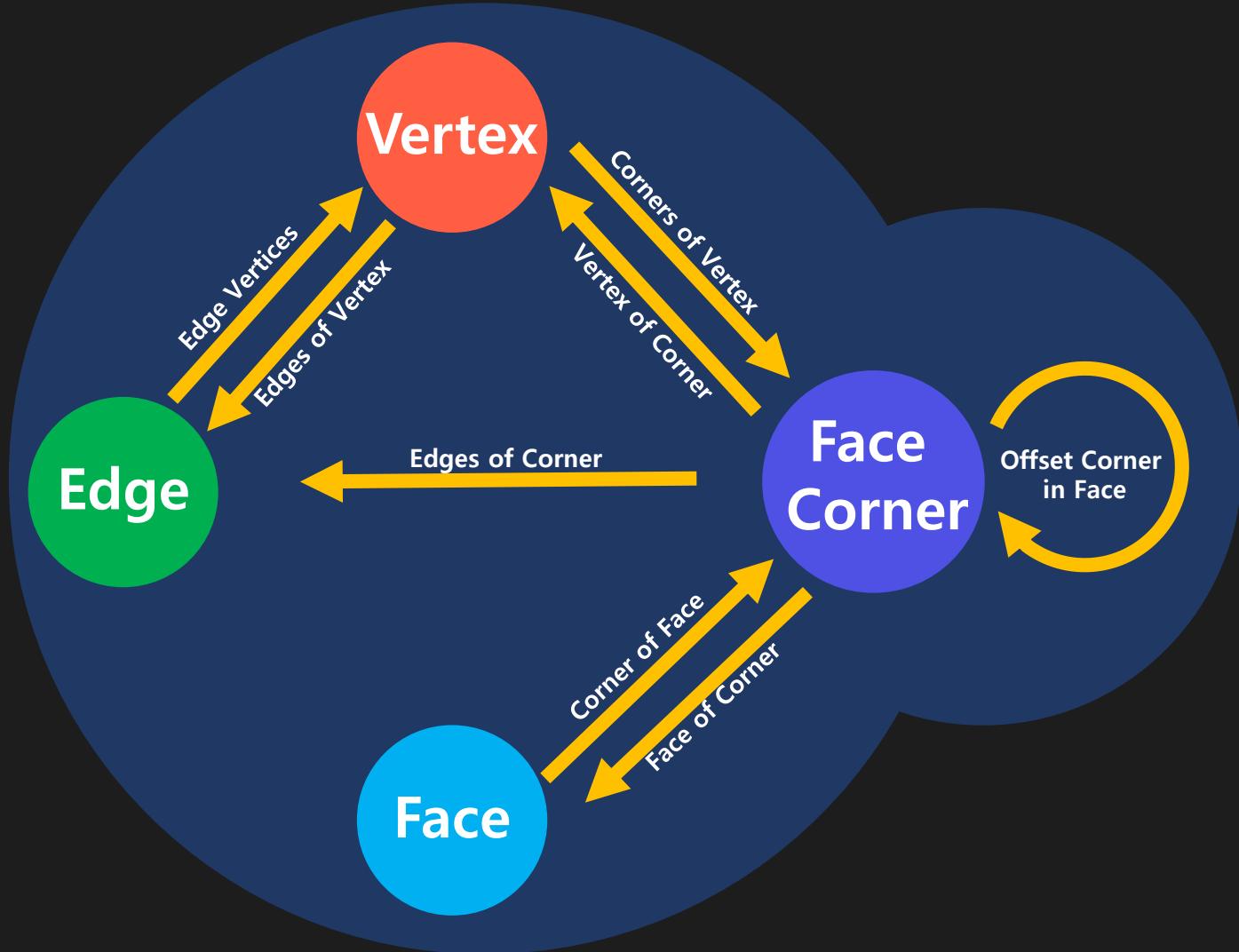


Edge Vertices는 엣지 인덱스를 입력하는 소켓이 없습니다.

하지만 Evaluate at Index를 이동하면 다른 토플로지 노드처럼 특정 엣지에 연결된 점 번호를 얻을 수 있습니다.

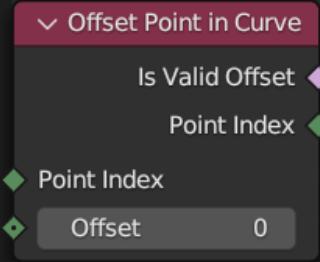
오른쪽처럼 연결하면 777번 엣지 한쪽 끝의 점 인덱스를 얻습니다.

점-선-면-페이스코너 변환표 (v3.5)

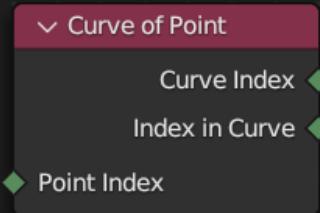


Topology Nodes

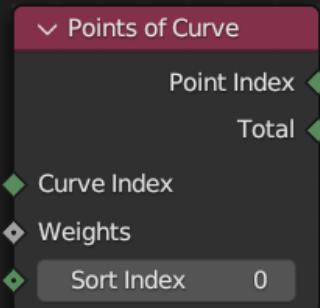
Curve를 위한 토폴로지 노드들도 있습니다. 커브는 2차원이므로 비교적 이해와 사용이 쉽습니다.



Offset Point in Curve는 사실상 Index에 덧셈을 하는 것과 같습니다만, 마지막 점을 인식해서 존재하지 않는 점에 대응하는걸 막아주고, Is Valid Offset으로 그 인덱스가 실제로 커브에 있는지 확인할 수 있습니다.

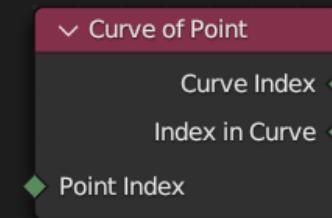
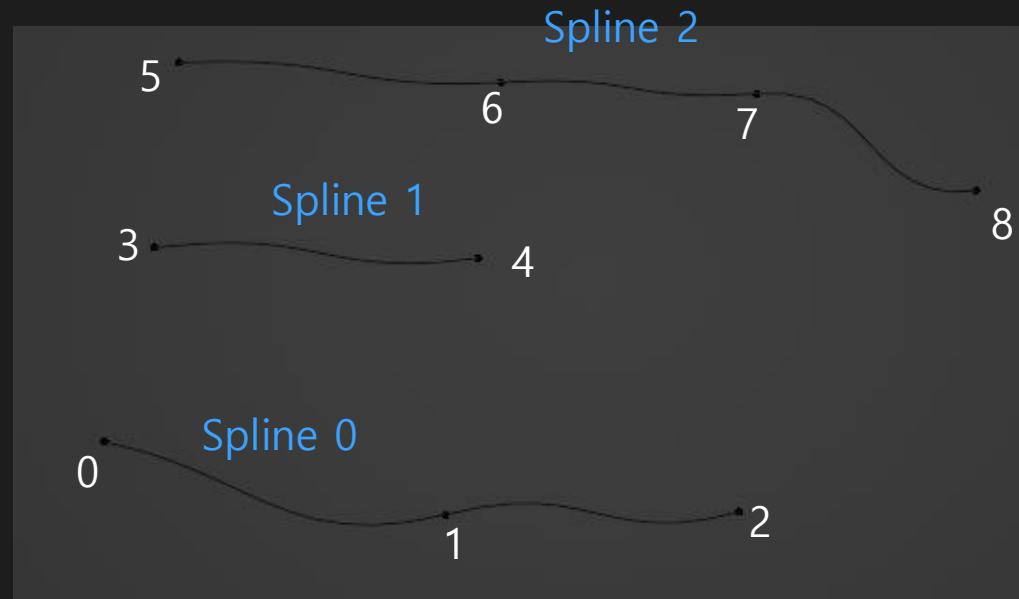


Curve of Point :
Curve Index는 입력받은 포인트가 속한 스플라인의 인덱스입니다.
Index In Curve는 현재 스플라인 기준으로, 입력받은 포인트가 몇 번째인지를 셹니다.

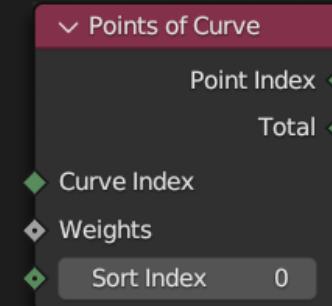


Points of Curve :
Point Index는 스플라인 내부 포인트의 인덱스를 찾습니다.
Sort Index를 통해 몇 번째 포인트를 찾을 지 고를 수 있고, 인덱스가 자동으로 반복되기 때문에 유용합니다.

연습문제 1



Point Index가 4일때
Curve Index = ?
Index in Curve = ?



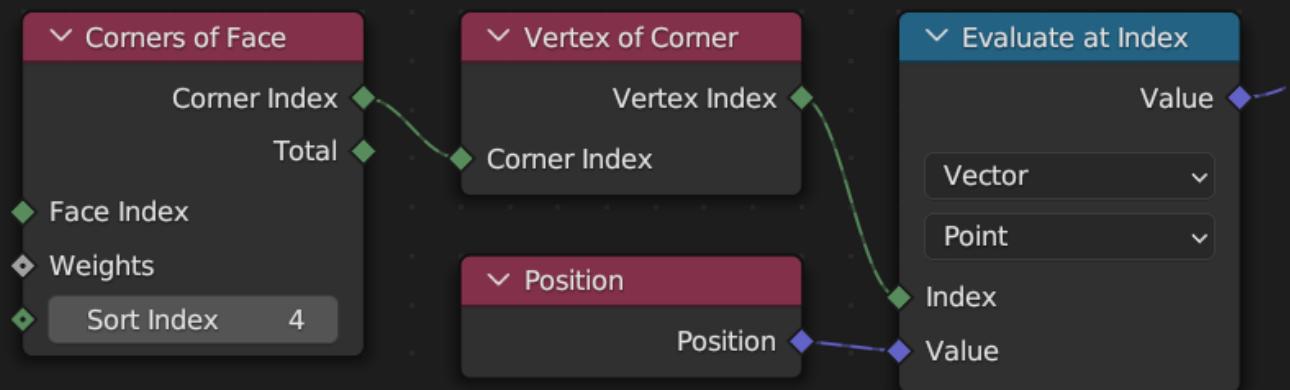
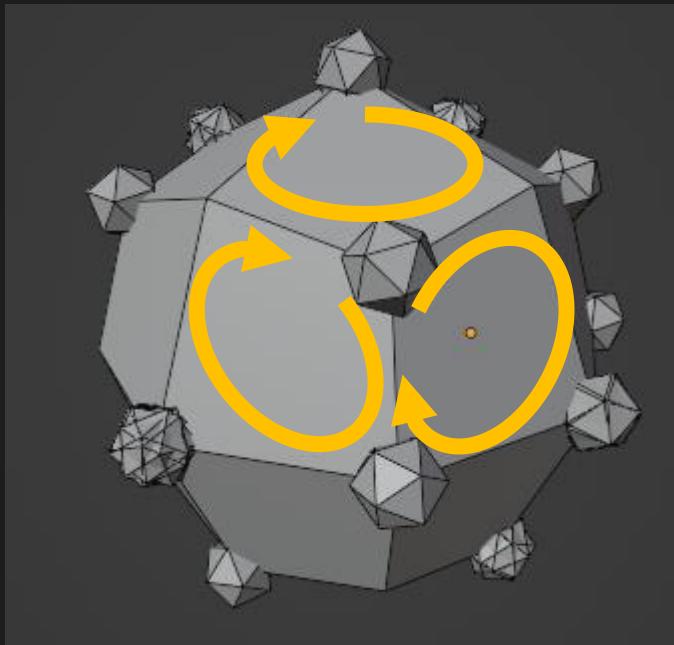
Curve Index가 2
Sort Index가 1일 때

Point Index = ?

정답 : 차례대로 1,1,6

연습문제 2

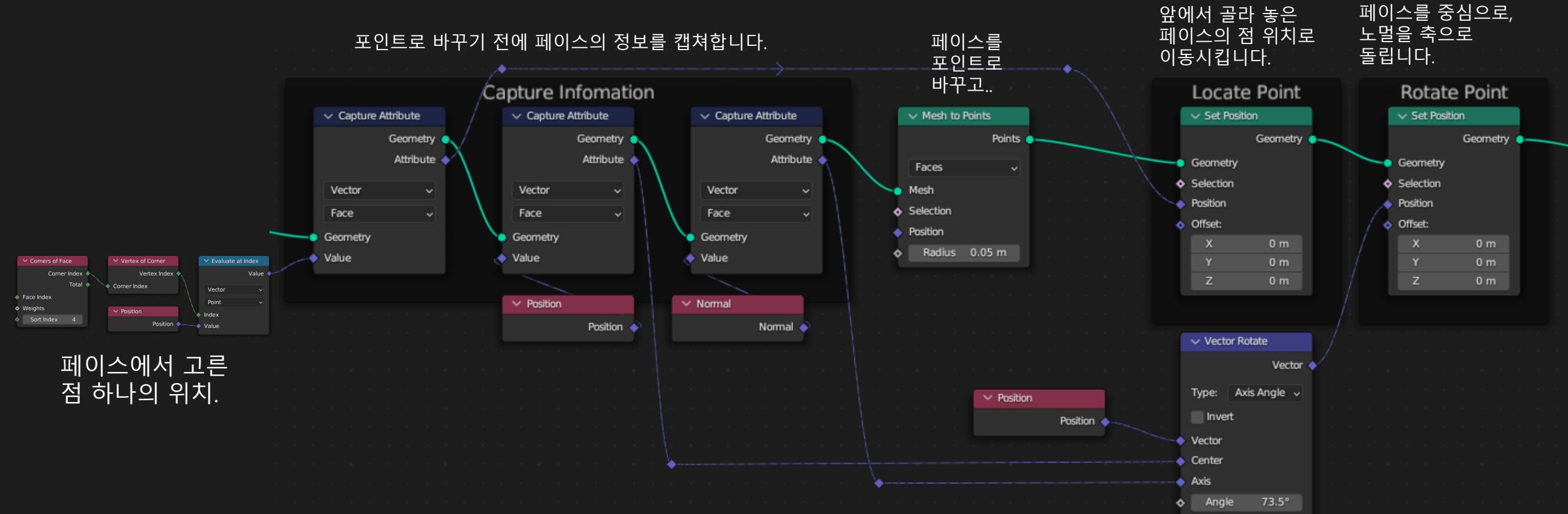
Face마다 점 하나씩을 골라서 Face를 중심으로 빙글빙글 돌려주세요.



'페이스에서 고른 점 하나의 위치'

연습문제 2 정답

Evaluate on Domain과 Capture Attribute의 도메인을 주의하세요!

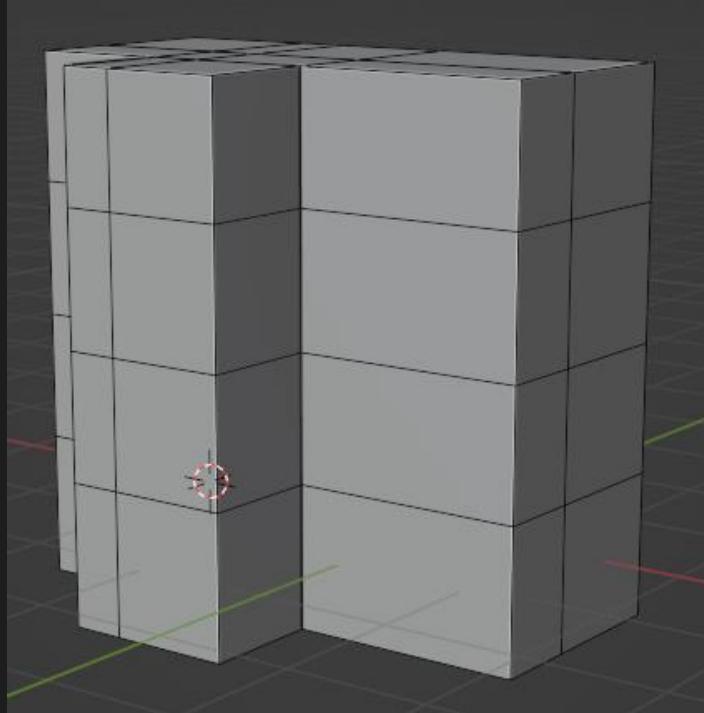


058강 Topology (2)

자기 주변의 점/선/면에 접근하는 법
토폴로지 정보를 통해 자동 생성 건축물 만들기



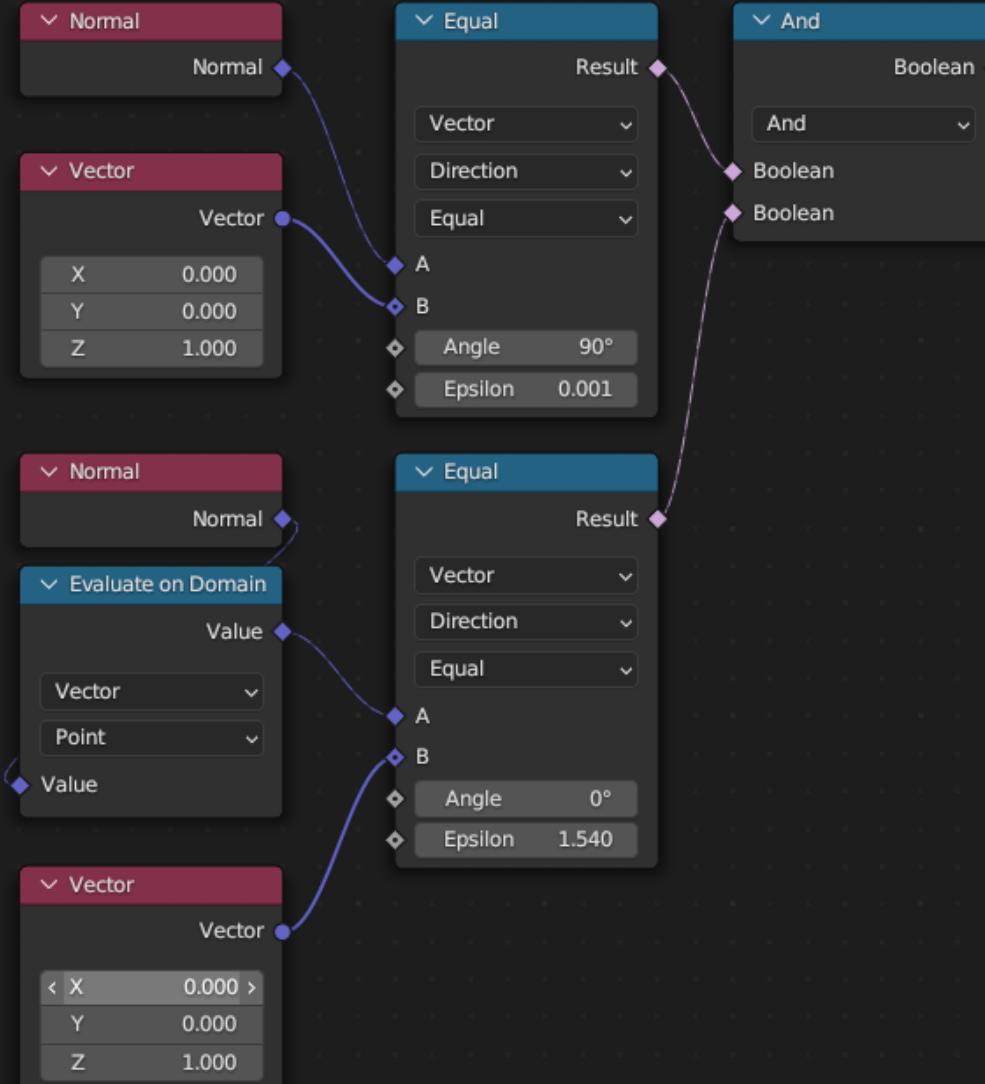
뼈대는 만들어 놨으니까 나머지 부탁드려요!



1. 각 면마다 창문 붙여주세요. 폭에 따라 크기 조절해주세요.
2. 각 층 사이에 몰딩 붙여주세요.
3. 맨 위층 두껍게 해주세요.



맨 위층을 선택하기



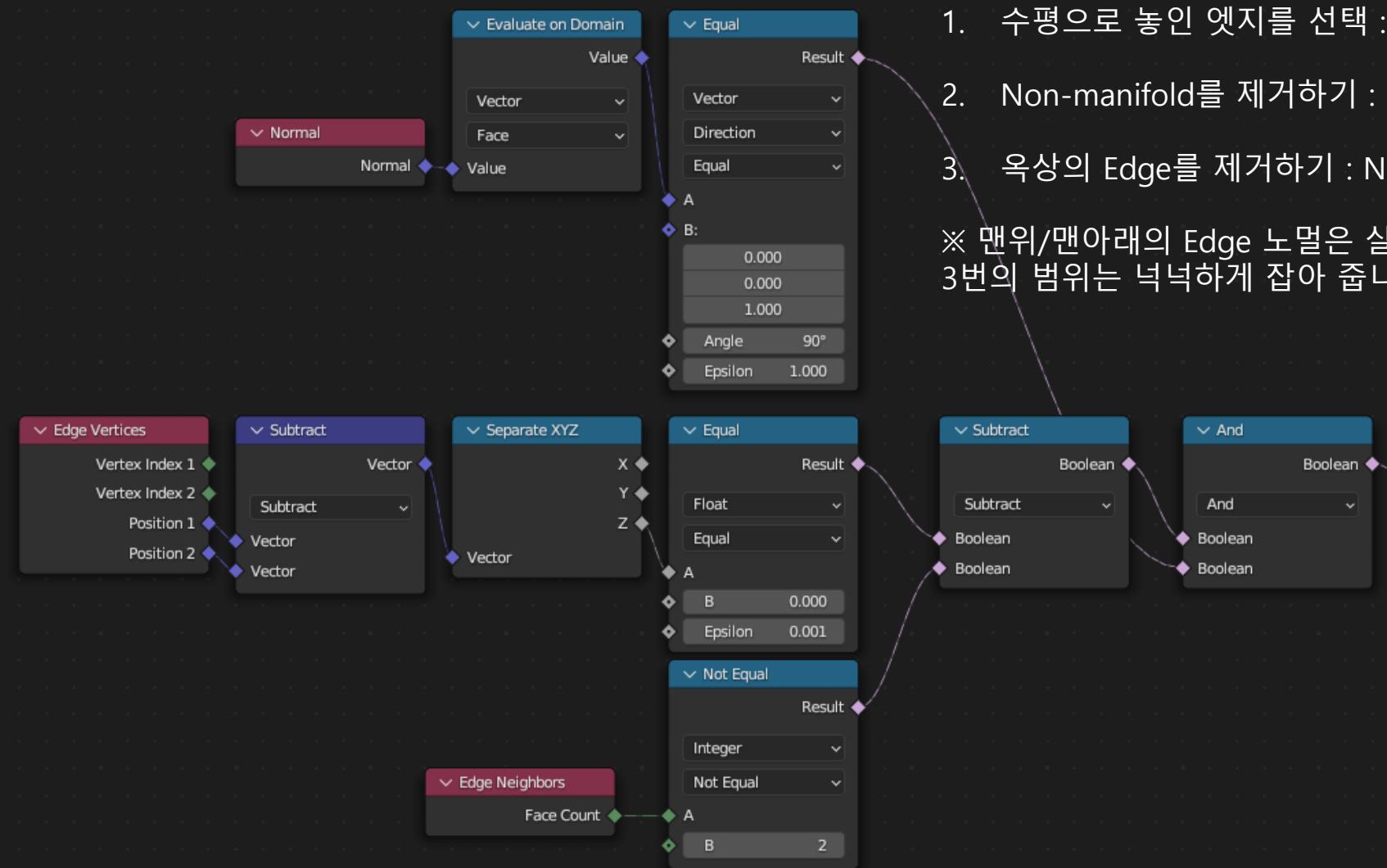
토폴로지가 복잡하지 않다는 전제 하에 진행합니다.

즉, 맨 위층을 나타내는 면은 옥상쪽과 붙어 있고, 그 사이에 루프컷이 없다는 등의 가정이 필요합니다.

Face Normal이 옆을 향해 있으면서,
Point Normal은 살짝 위쪽인 **면**을 선택합니다.

※Even Extrude를 위해 54강의 노드를 사용합니다.

몰딩을 불일 옆지 선택하기

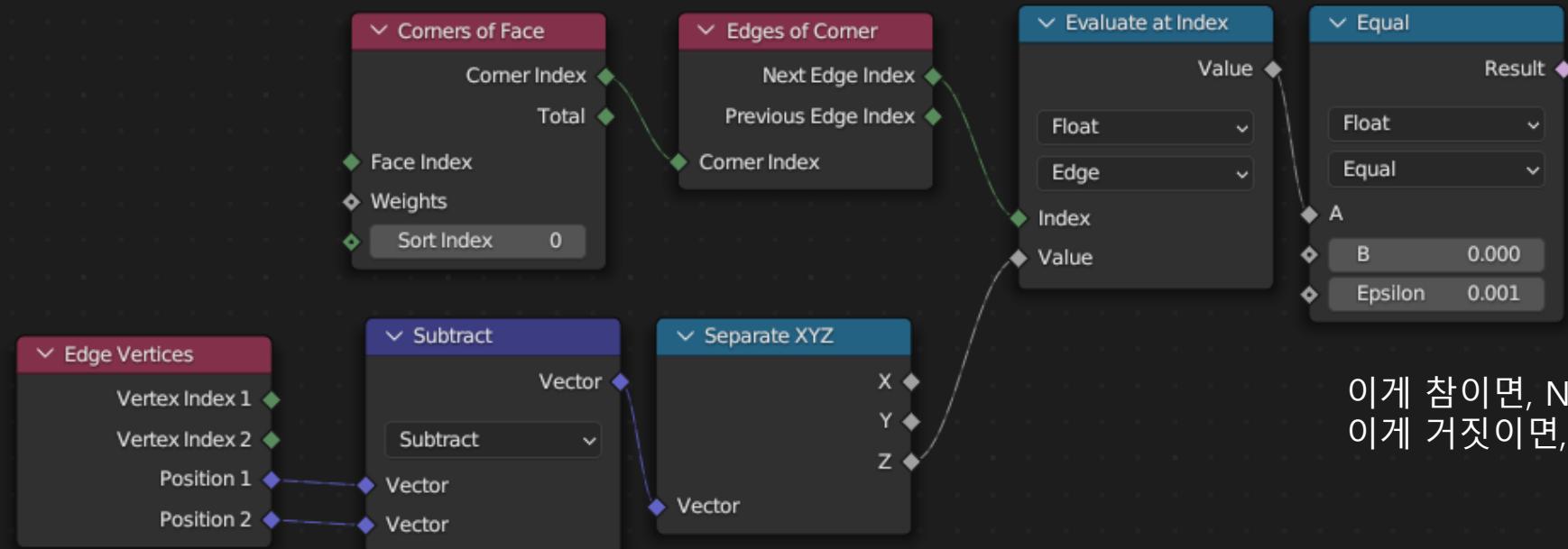


- 수평으로 놓인 옆지를 선택 : Edge 양 끝의 높이 차이가 없는 것
 - Non-manifold를 제거하기 : 최상층을 Extrude할 때 생긴 옆지를 제거
 - 옥상의 Edge를 제거하기 : Normal이 수평을 향해 있는 것
- ※ 맨위/맨아래의 Edge 노멀은 살짝 위/아래를 향하고 있으므로, 3번의 범위는 넉넉하게 잡아 줍니다.

면의 폭을 재기 (1)

Face에서 Corner 하나를 고른 뒤, Corner에 연결된 Edge중 하나를 선택합니다.

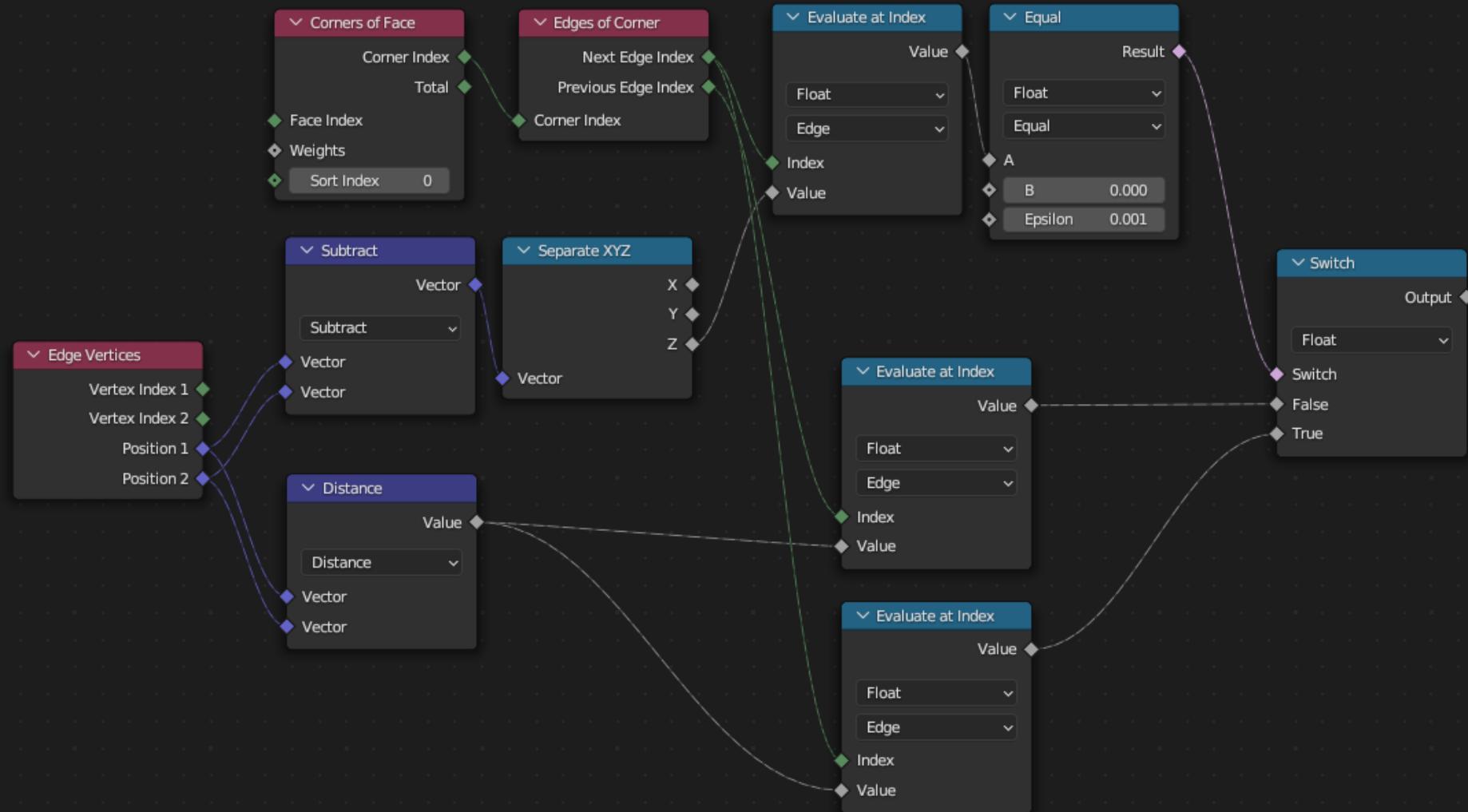
선택한 Edge가 양 끝의 높이차이가 없다면, 수평으로 놓였다는 것이고, 만약 그렇지 않다면 선택하지 않은 edge가 수평일 것입니다.



이게 참이면, Next Edge Index가 수평.
이게 거짓이면, Previous Edge Index가 수평.

면의 폭을 재기 (2)

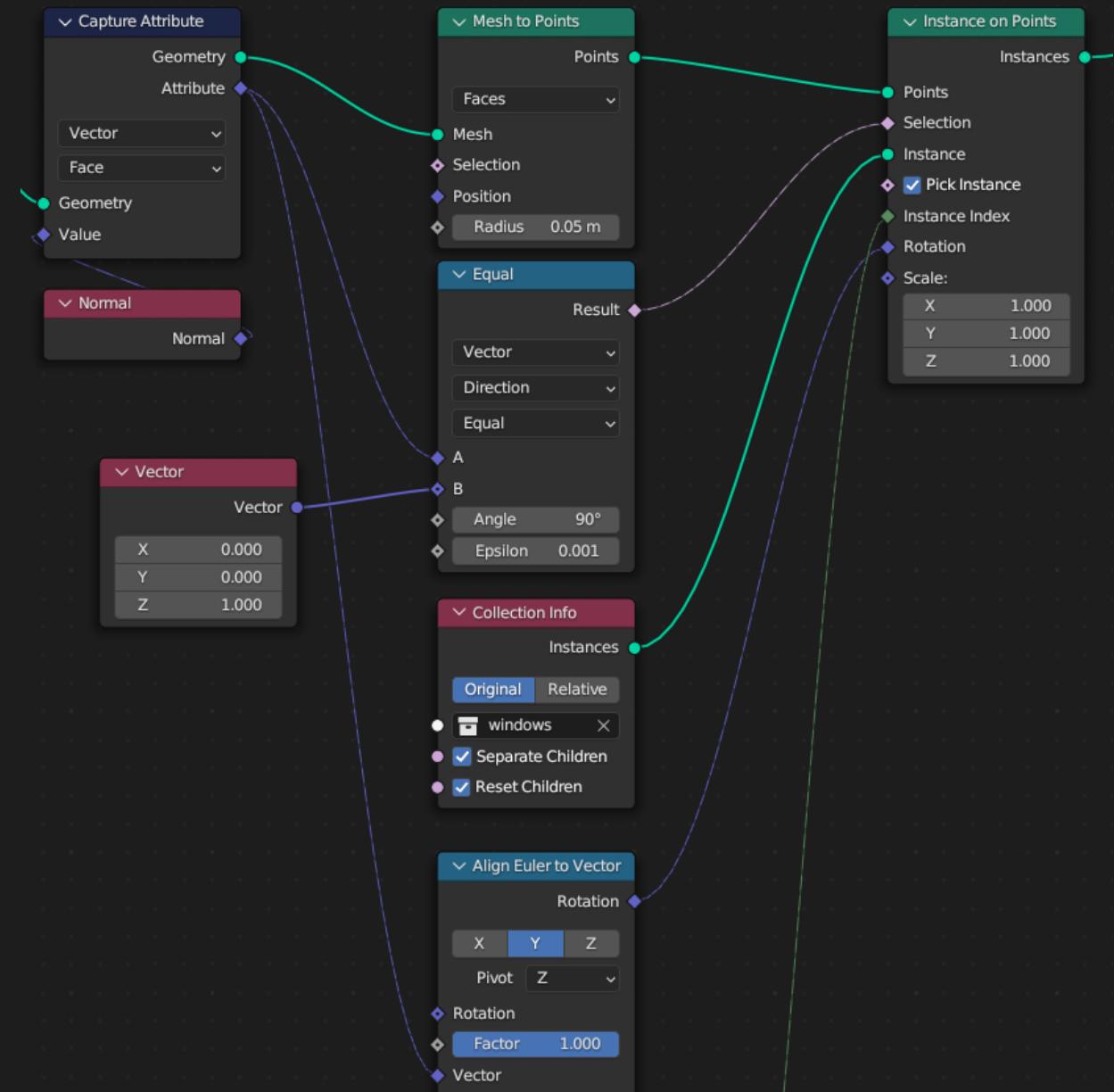
면에 붙은 Edge 중, 수평으로 놓인 쪽의 길이를 내보냅니다 :



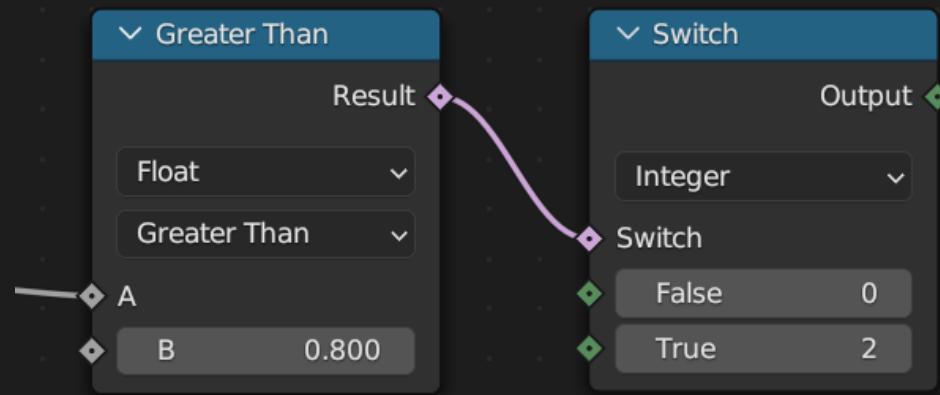
면에 오브젝트를 붙이기

Mesh to Points로 면을 점으로 바꿔, 창문을 붙입니다.
Normal을 이용해 옆으로 놓인 면만 선택하고,
창문을 올바른 방향으로 회전시킵니다.

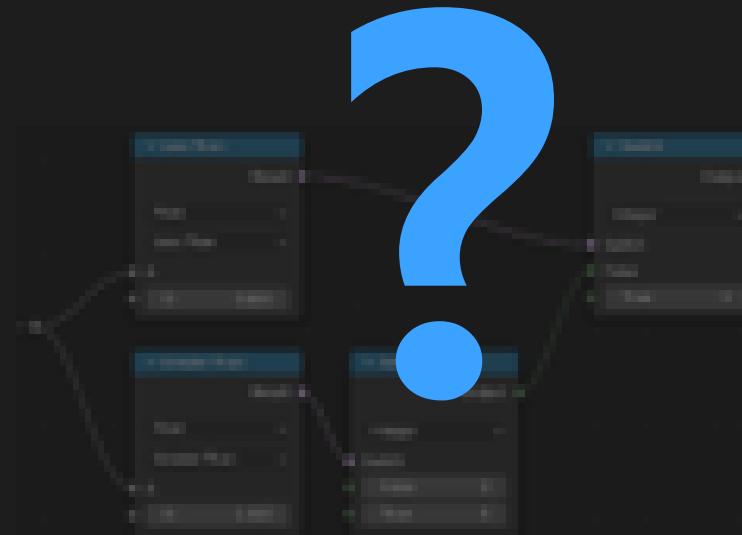
※점으로 바꾸기 전에 Normal을 캡쳐해 놓아야 합니다.



다중 Switch



0.8보다 크면 2번
작으면 0번



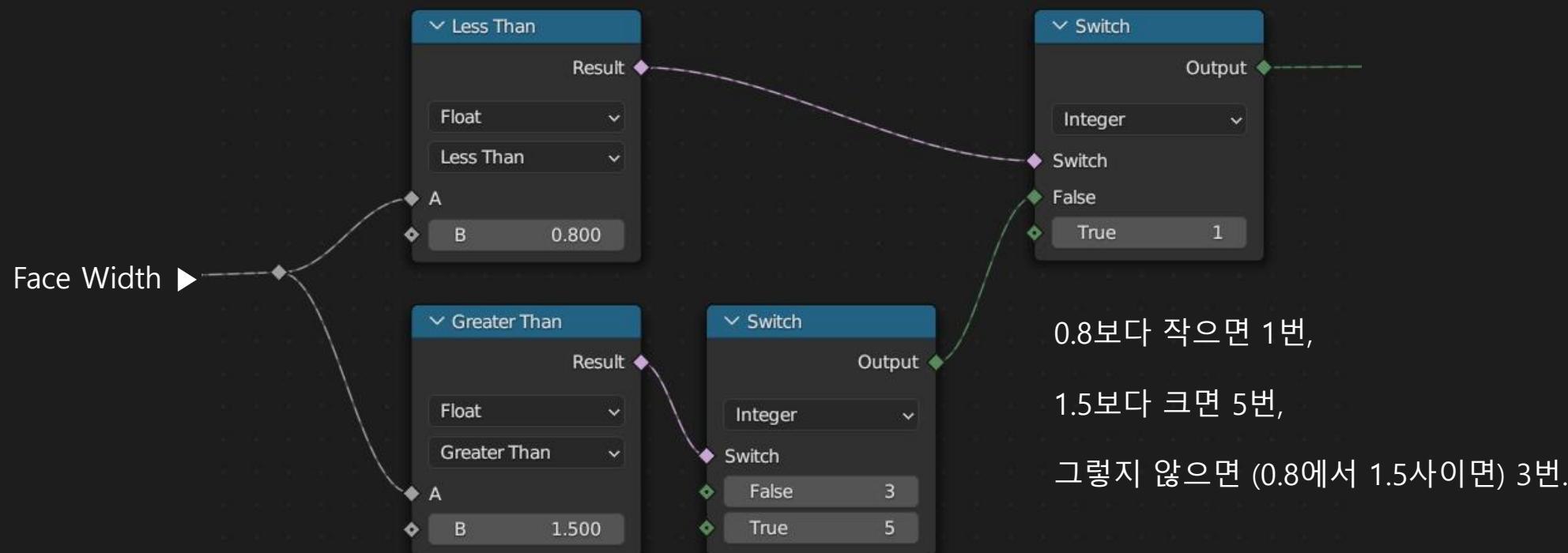
0.8보다 작으면 0번
0.8에서 1.5 사이는 2번
1.5보다 크면 4번

다중 Switch

페이스 폭에 따라 인스턴스의 인덱스를 고릅니다.

먼저 Compare와 Switch를 한번씩 사용하여 작은 창문을 선택한 다음에,
조건을 만족하지 않은 경우에 한하여 Compare와 Switch를 다시 한번 사용하면,
다중 Switch를 만들 수 있습니다.

노드트리를 한번에 보면 헷갈릴 수 있으므로, 한 단계씩 생각하는 것이 포인트입니다.

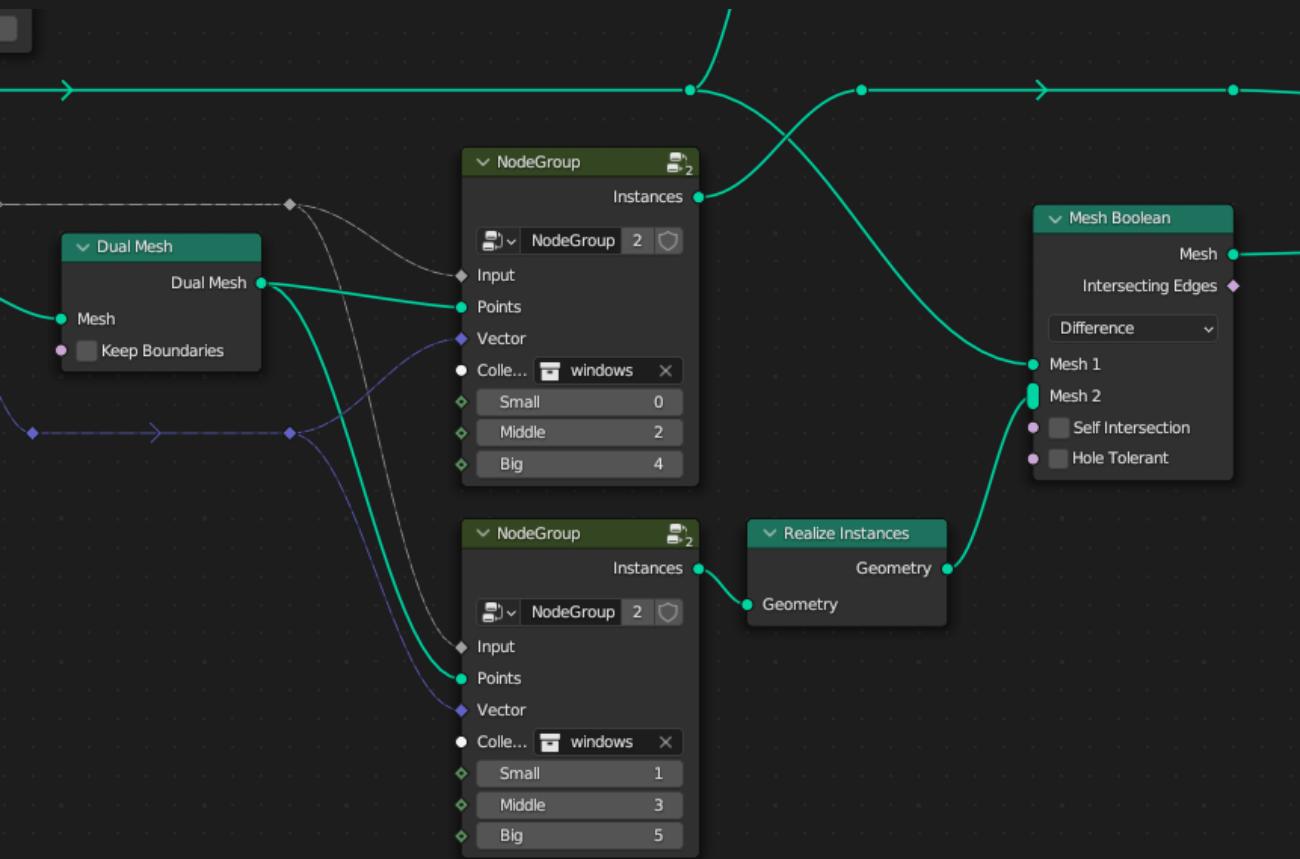


정리

창문을 뚫을 Cutter와 창문 오브젝트를 따로 따로 붙여야 합니다.

노드 연결이 매우 혼란스러워지므로, 노드그룹으로 묶어 단순화시키면 알아보기 쉽습니다.

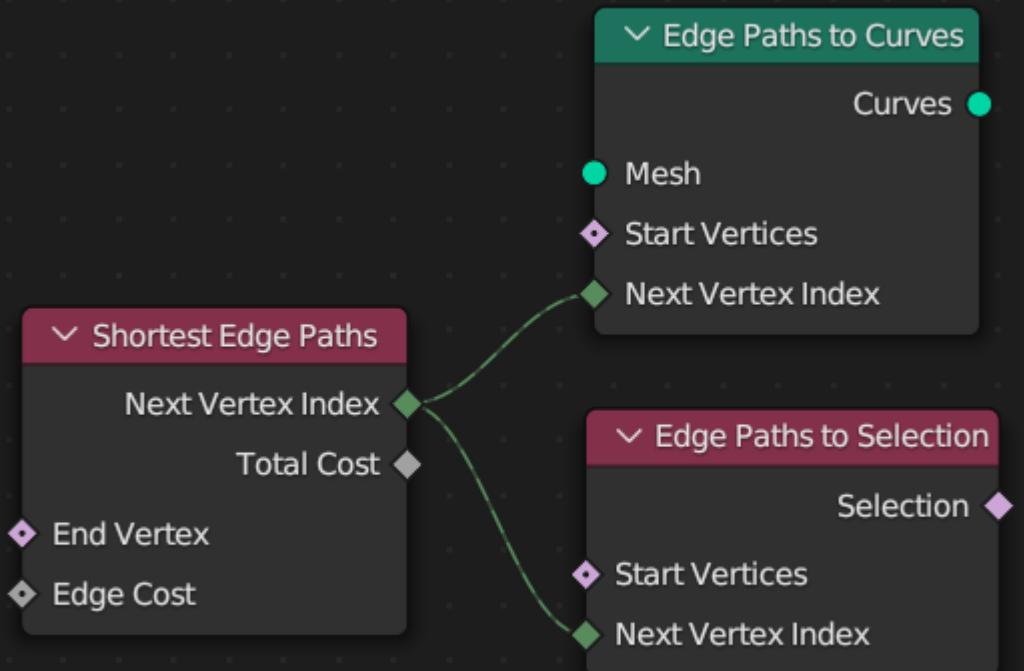
추후 수정을 위해선 각 소켓의 이름을 적절히 정해주는 편이 좋습니다.



059강 Shortest Edge Path : 최단경로

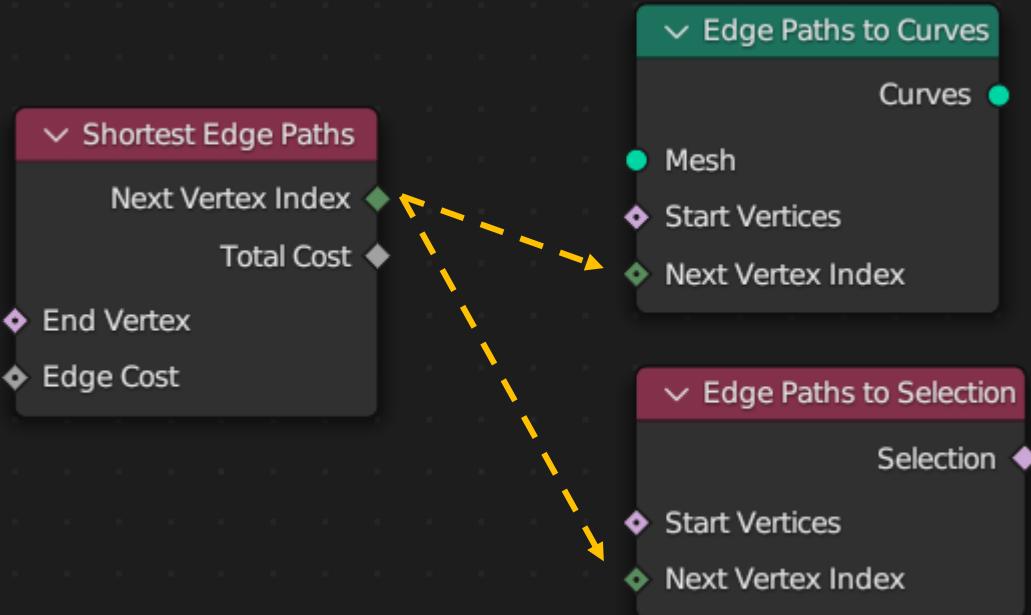
Shortest Edge Path 노드의 개념과 작동 원리

예제 : 번개 효과



Edge Path Nodes

Shortest Edge Paths, Edge Paths to Selection, Edge Paths to Curves는 **최단경로**를 만드는 노드입니다.
시작점으로부터 끝점까지의 가장 빠른 경로를 찾아서 선택하거나, 직접 커브를 만듭니다.



기본 규칙

▼ Shortest Edge Paths

- Next Vertex Index
- Total Cost

◆ End Vertex

◆ Edge Cost

시작점, 끝점

Shortest Edge Path는 각각의 점에서 End Vertex까지 가는 최단경로를 계산합니다.

기본적으로 '모든 점'에서 End Vertex까지의 경로를 찾기 때문에 시작점이 없어도 작동합니다.

이후에 Edge Paths to Curves / Edge Paths to Selection을 통해 이 중에서 특정 경로만을 찾습니다.

▼ Edge Paths to Curves

- Curves

● Mesh

◆ Start Vertices

◆ Next Vertex Index

경로를 찾는 기준

최단경로의 기준은 **Edge Cost**를 기준으로 합니다.

Cost의 기본값은 엣지마다 모두 같으므로,

기본값으로는 '지나가는 엣지 개수가 가장 적도록' 움직입니다.

▼ Edge Paths to Selection

- Selection

◆ Start Vertices

◆ Next Vertex Index

만약 실제 거리를 이용하고 싶으면, Edge Vertices를 Cost에 연결합니다.

▼ Edge Vertices

- Vertex Index 1
- Vertex Index 2
- Position 1
- Position 2

▼ Distance

- Value
- Distance
- Vector
- Vector

The diagram shows four blue diamond-shaped nodes labeled 'Vertex Index 1', 'Vertex Index 2', 'Position 1', and 'Position 2'. Two arrows point from 'Position 1' and 'Position 2' to a blue diamond node labeled 'Vector'. Another arrow points from this 'Vector' node to a dropdown menu labeled 'Distance' in a box labeled 'Value'. A third arrow points from the 'Distance' box to a blue diamond node labeled 'Vector'.

기본 규칙

▼ Shortest Edge Paths

Next Vertex Index ◆

Total Cost ◆

◆ End Vertex

◆ Edge Cost

Next Vertex Index의 값

Shortest Edge Paths가 만드는 Next Vertex Index는 말 그대로 '다음 점'의 인덱스를 출력합니다.

만약 경로가 1-3-2-6-4 라면, 3번 점의 Next Vertex Index 값은 2가 되는 식입니다.

▼ Edge Paths to Curves

Curves ●

● Mesh

◆ Start Vertices

◆ Next Vertex Index

Total Cost

Total Cost는 각각의 점에서 끝점까지 도달하는데 드는 총 비용입니다.

Shortest Edge Paths는 모든 점에 대해 계산하기 때문에, 모든 점에서의 Total Cost를 계산합니다.

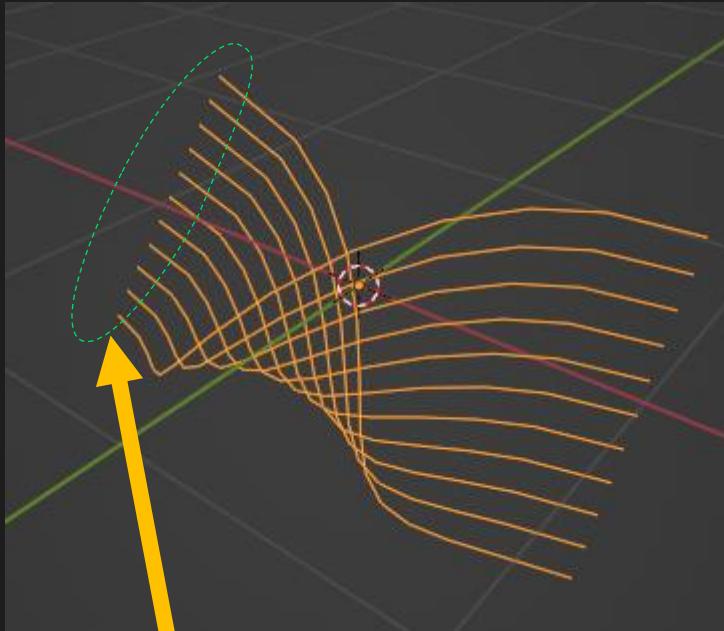
▼ Edge Paths to Selection

Selection ◆

◆ Start Vertices

◆ Next Vertex Index

기본 사용법



Named Attribute

Attribute: **Exists**

Float

Name: **Group**

Shortest Edge Paths

Next Vertex Index

Total Cost

End Vertex

Edge Cost

Edge Paths to Selection

Selection

Start Vertices

Next Vertex Index

Not

Boolean

Not

Boolean

Delete Geometry

Geometry

Edge

All

Geometry

Selection

Shortest Edge Paths로 계산한 결과를, Edge Paths to Curves나 Edge Paths to Selection에 연결시켜 완성합니다.

Selection은 오로지 엣지를 선택만 해 주고, Curves는 직접 엣지를 연결해 커브들을 만들어줍니다.

지나가는 엣지 수를 최소한으로 하는 노드의 성질을 이용하면, 복잡한 토플로지에서도 한쪽 방향의 엣지만 선택할 수 있습니다.

060강 Attribute의 전송 (4) : Sample UV

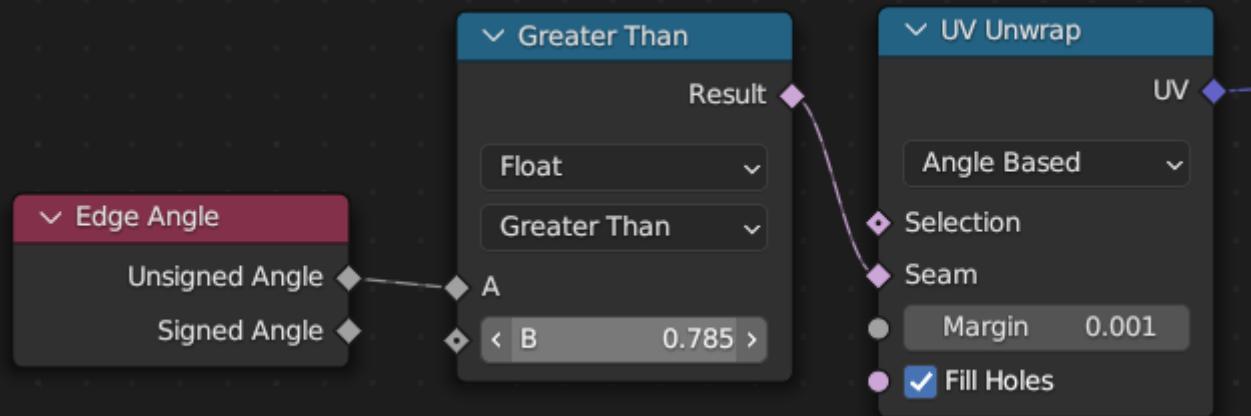
지오메트리 노드를 통한 UV Wnwrap

Sample UV Surface를 통해 지오메트리 간 정보를 전송하기



UV의 생성

UV Unwrap Node

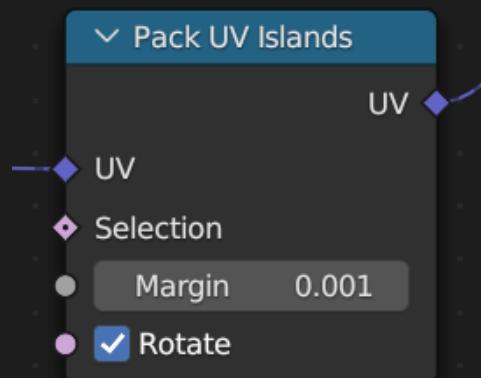


UV Unwrap 노드는 연결된 지오메트리의 UV를 생성합니다.

왼쪽처럼 엣지의 각도를 꽂으면

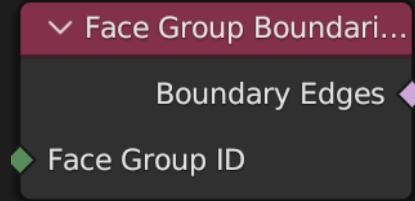
면이 크게 꺾이는 부분을 잘라 줍니다.

Pack UV Islands Node

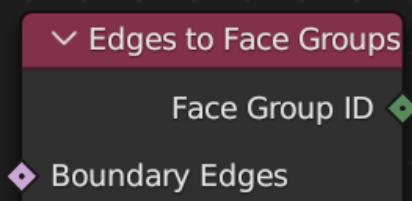
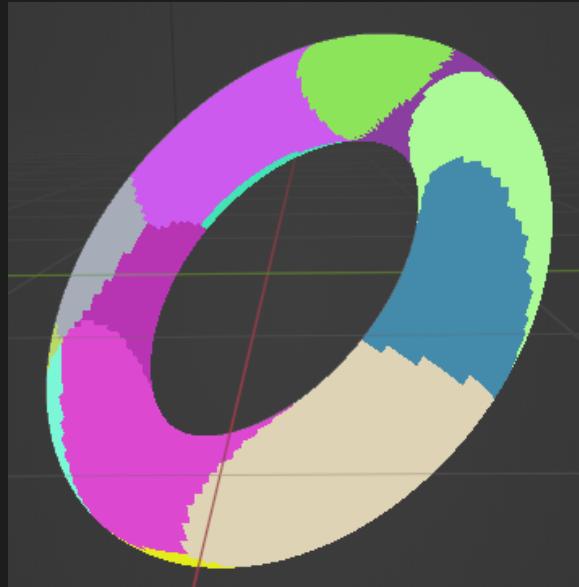


UV의 크기와 위치를 조절하여 UV의 간격을 겹치지 않게 적절히 조절합니다.

Face Group Boundaries



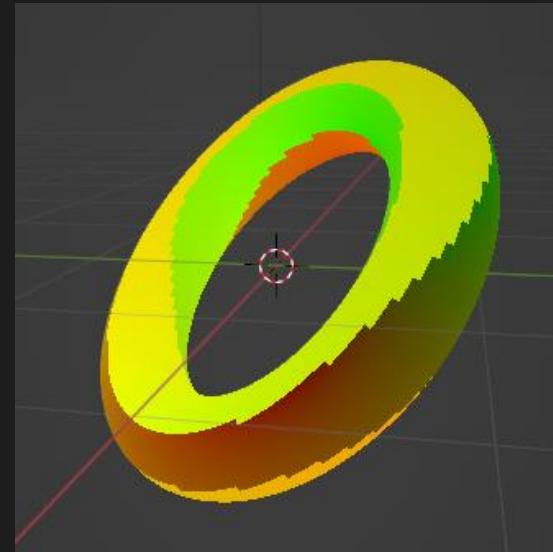
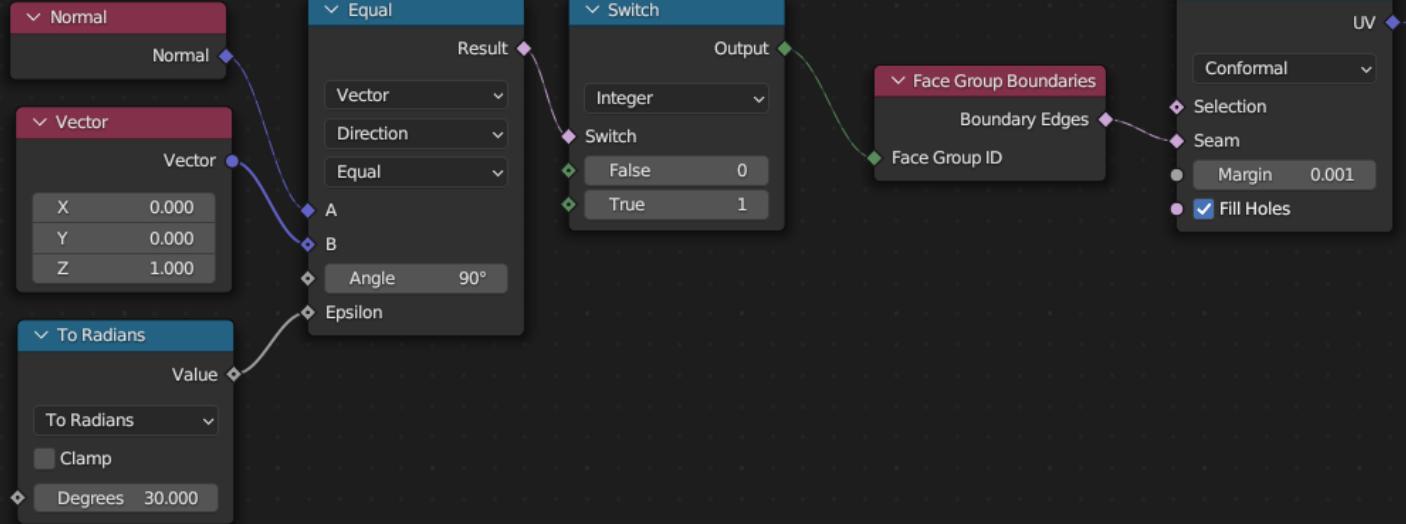
어떤 기준으로 면이 구분되었을 때,
그 경계 (엣지)를 선택해줍니다.



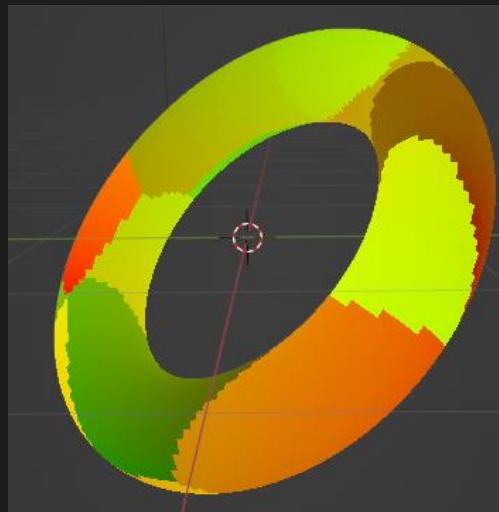
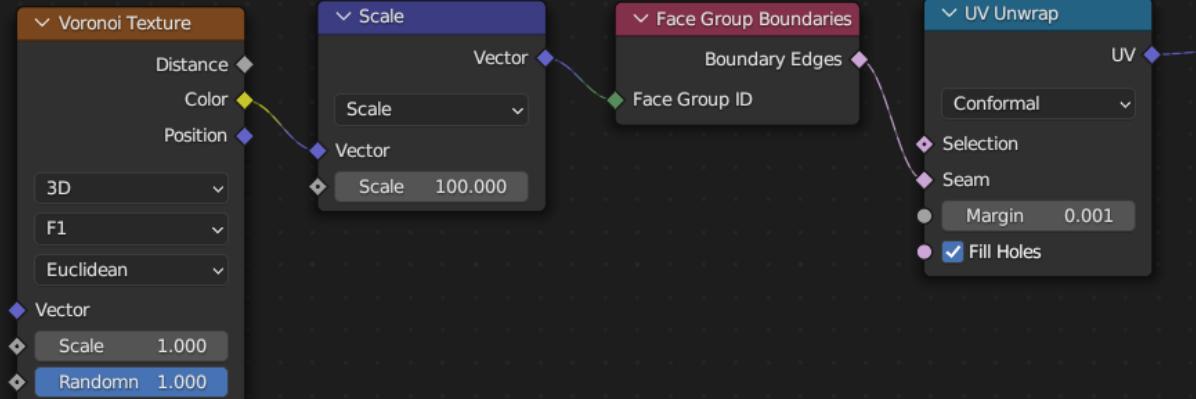
Edges to Face Groups : 거꾸로,
엣지 경계를 바탕으로 면을 구분해 줍니다.

Face Group Boundaries의 UV사용

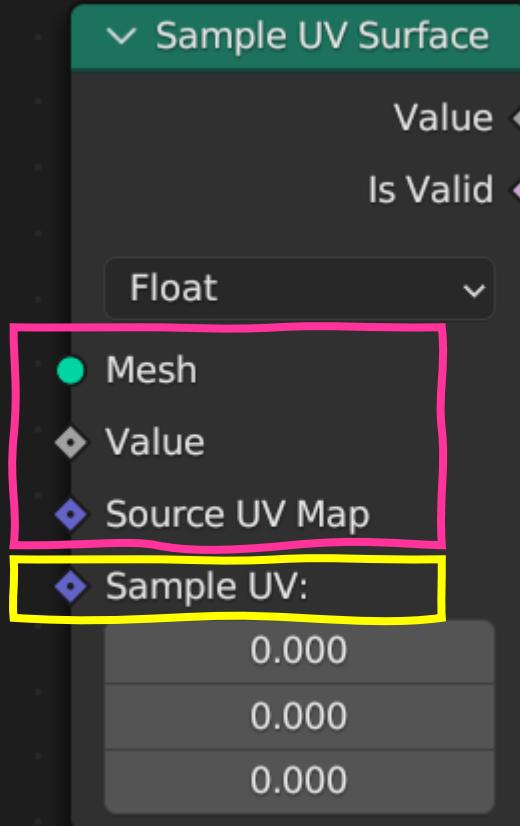
노멀 기준



텍스쳐를 이용하는 방법



Sample UV Surface



Sample UV Surface는 UV를 바탕으로 Attribute를 구하는 노드입니다.

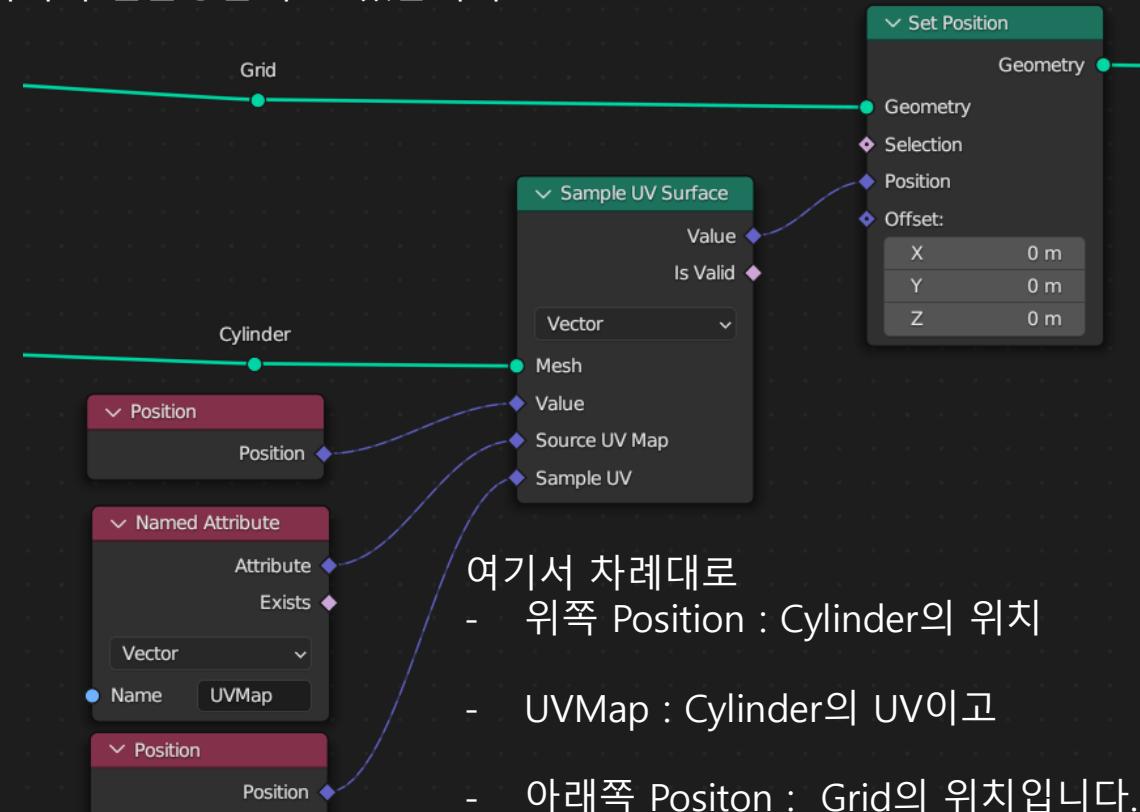
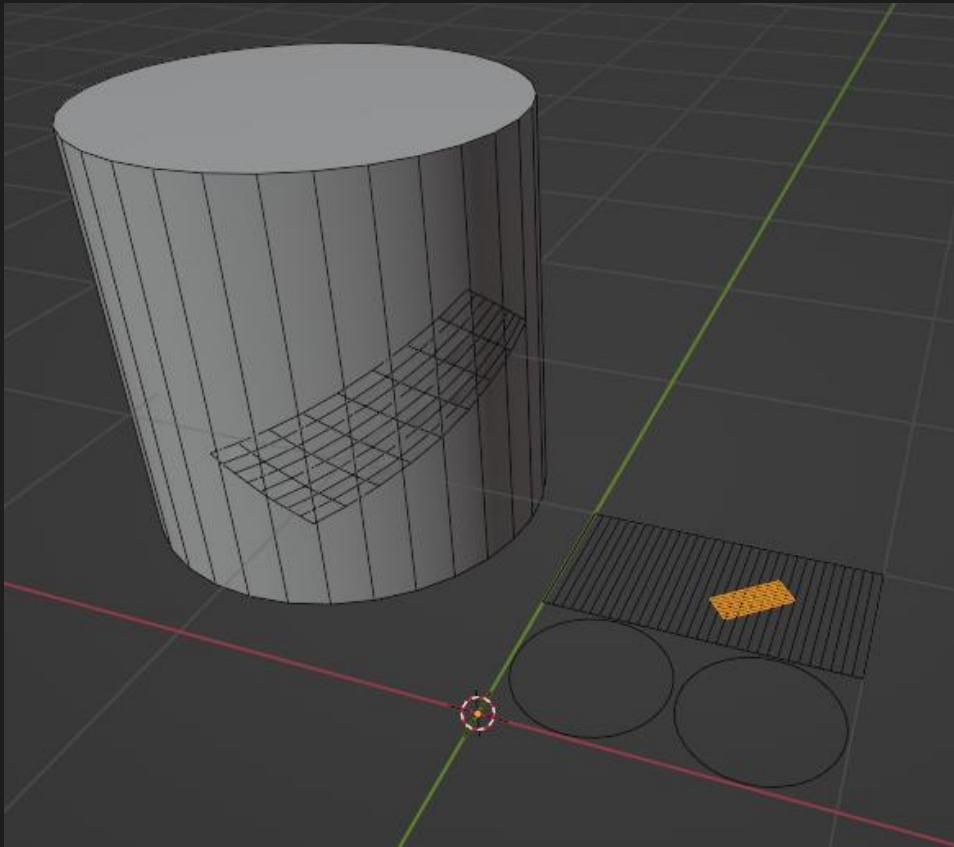
기본 작동원리

각각의 점에 대하여,

1. 각각의 점이 가지고 있는 Sample UV 값을 찾고,
2. 그 값과 일치하는 Source UV Map을 기준으로, 해당 UV좌표를 갖는 표면을 찾고,
3. 그 표면 위의 Value (Attribute)를 내보냅니다.

Sample UV = Position

UV가 속하는 0~1 사이의 좌표에 직접 메쉬를 위치시켜서 샘플링할 수도 있습니다.

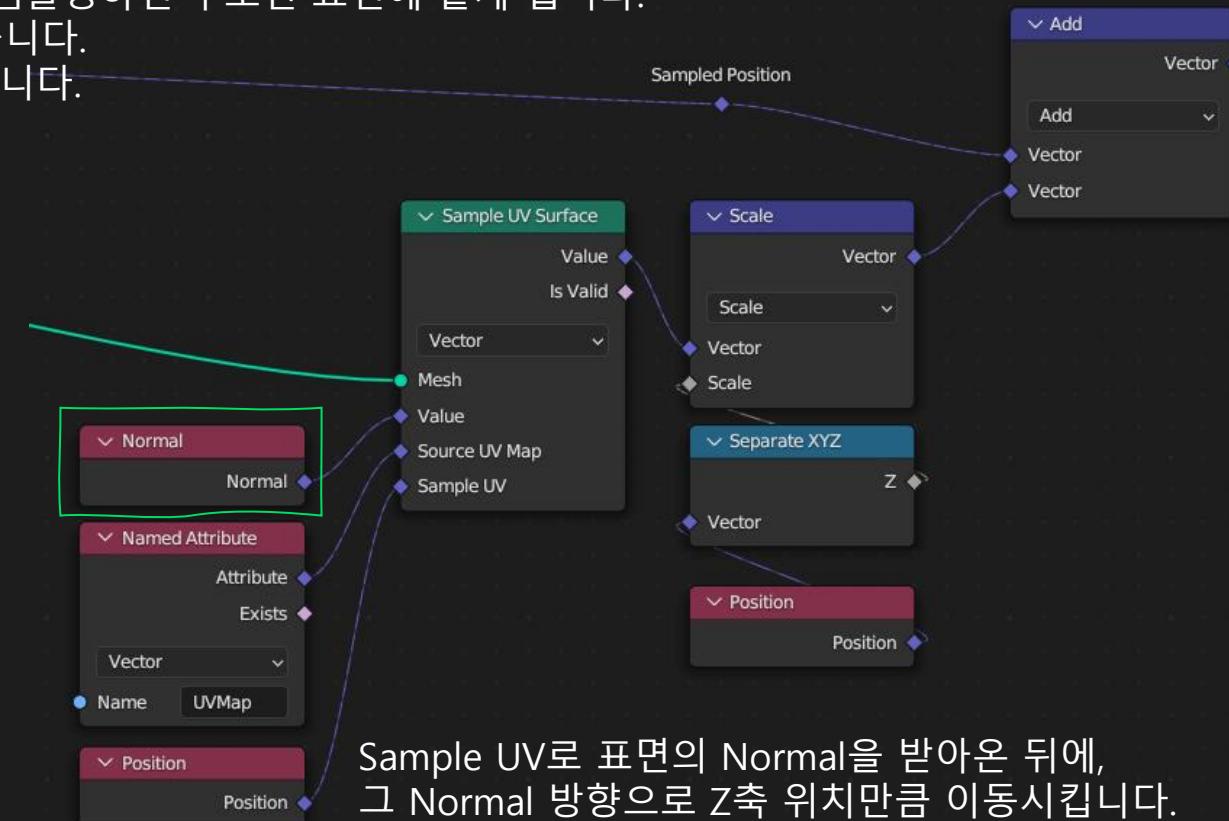
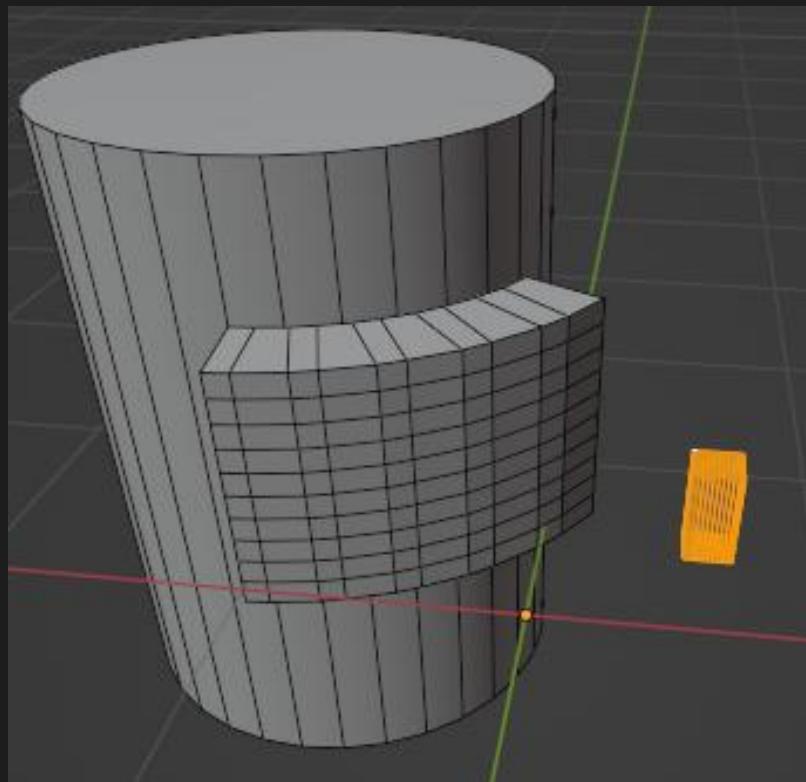


여기서 차례대로

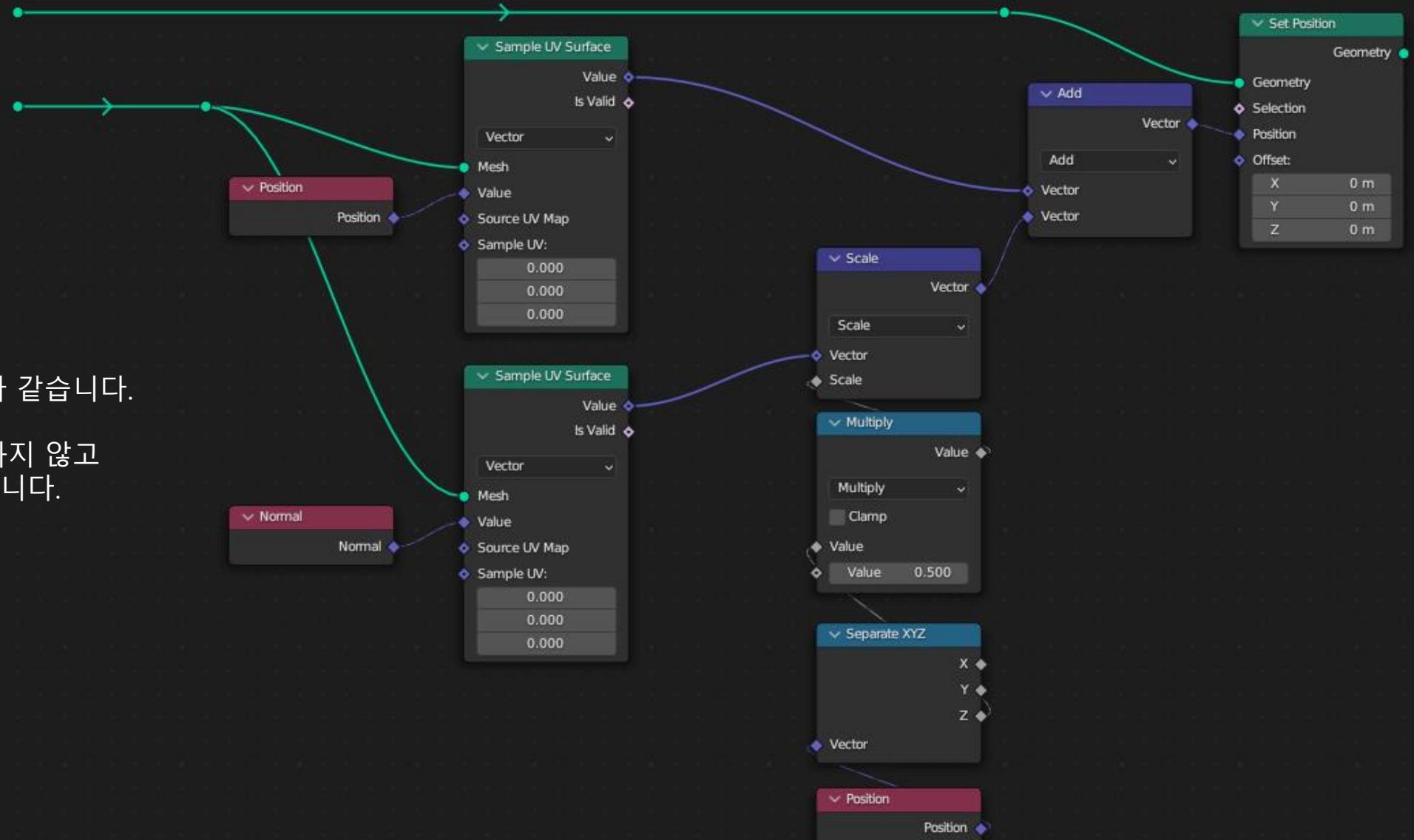
- 위쪽 Position : Cylinder의 위치
- UVMap : Cylinder의 UV이고
- 아래쪽 Position : Grid의 위치입니다.

높이의 복원

UV는 2차원이므로, 샘플링 시 Z좌표는 무시됩니다. 따라서 위치를 샘플링하면 무조건 표면에 붙게 됩니다.
붙은 표면의 Normal 방향으로 이동시키면 높이를 복원시킬 수 있습니다.
다만, 그러기 위해서는 Sample UV Surface를 다시한번 사용해야 합니다.



높이의 복원



전체적인 노드 구조는 다음과 같습니다.

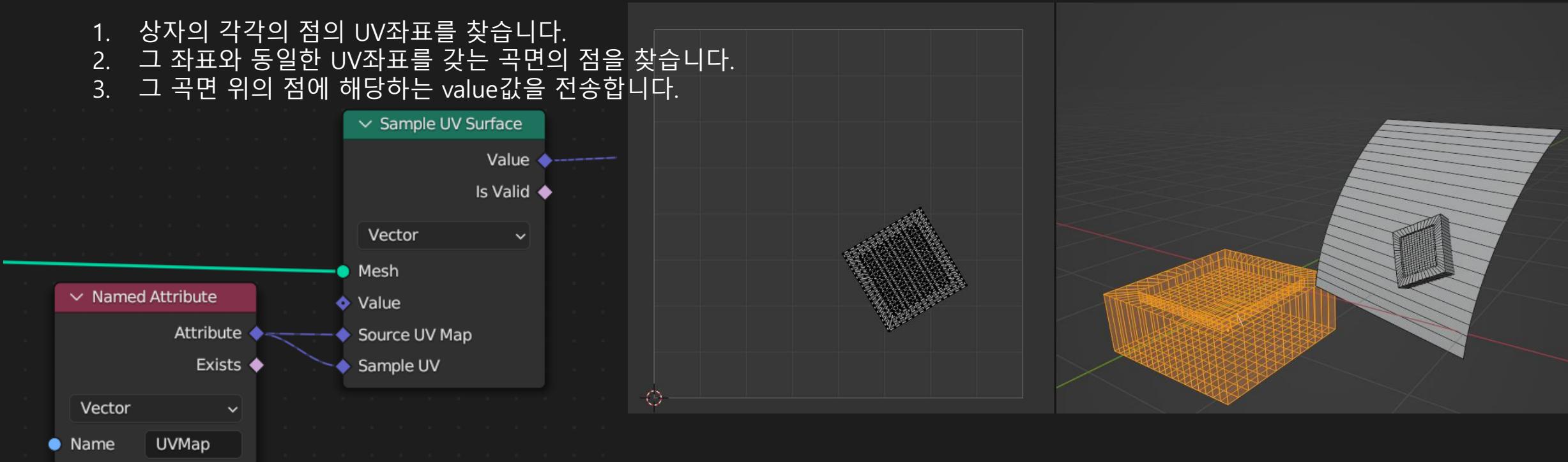
※Normal 부분을 Add로 더하지 않고
Offset에 끌어도 같은 연결입니다.

Sample UV = 나의 UV

샘플링에 자신의 UV를 사용할 수 있습니다.

아래와 같이 상자를 곡면에 붙이면, 다음과 같은 계산 과정이 일어납니다.

1. 상자의 각각의 점의 UV좌표를 찾습니다.
2. 그 좌표와 동일한 UV좌표를 갖는 곡면의 점을 찾습니다.
3. 그 곡면 위의 점에 해당하는 value값을 전송합니다.



UV는 지오메트리가 바뀌어도 변하지 않기 때문에, 안정적으로 Attribute를 전송할 수 있습니다.

061강 Curves(1)

Hair를 만들어주는 Curves 오브젝트의 개념

Hair 노드그룹 사용법



학습목표

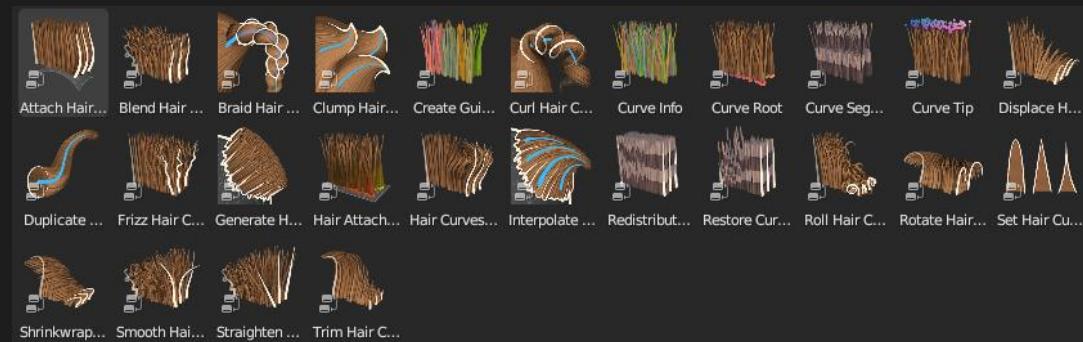
Curves란?

- Curves와 Curve의 다른점?
- Curves가 필요한 때?
- Curves의 생성 방법?

Deform Curves on Surface 란?

- Deform Curves on Surface가 하는 일은?
- surface_uv_coordinate 란?
- Armature Deform을 따라가려면

헤어 노드그룹의 사용법은?



새로운 헤어 기능

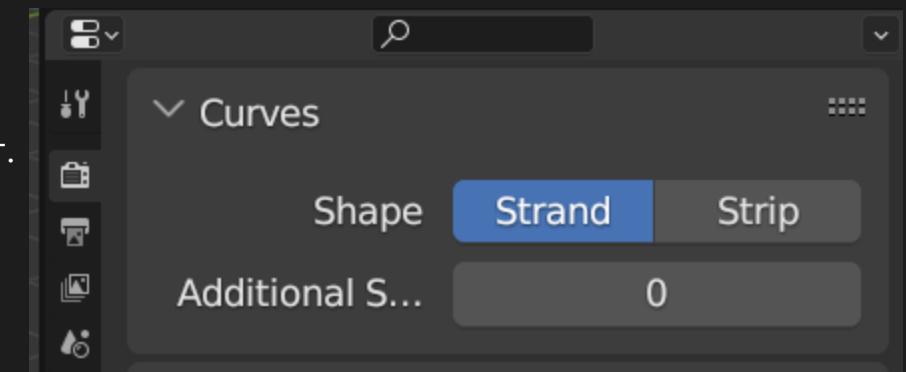
기존에 있던 헤어 파티클이 아닌 새로운 형식의 헤어 커브가 추가되었고,
3.5 버전으로 업그레이드되면서 이를 컨트롤 할 수 있는 노드그룹이 추가되었습니다.

Curve 's'



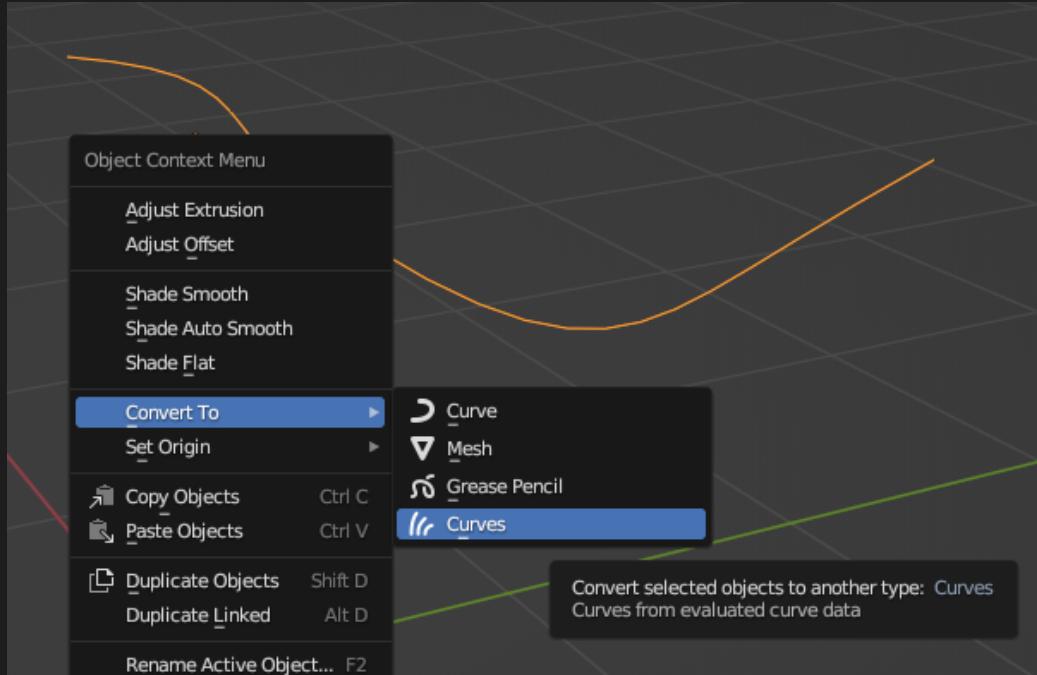
Curves 오브젝트는 여러 개의 커브를 다룰 때 사용하는 새로운 타입의 오브젝트입니다.
Curves에 속한 커브 지오메트리는 - 많은 커브를 렌더링하기 위해 설계된 -
단순화된 형태로 보여집니다.

※기존의 헤어 파티클을 오브젝트로 만든 형태라고 생각하시면 됩니다.

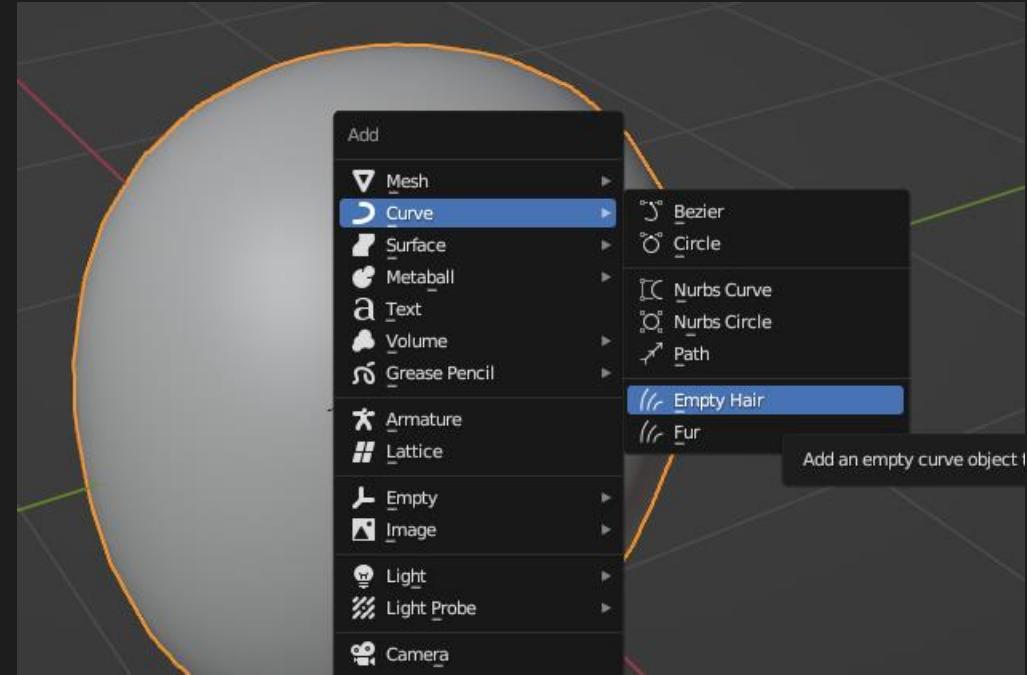


렌더 설정에서 Curves가 어떻게 보여질지 설정할 수 있습니다.

Curves의 생성



Curve를 우클릭해서 Curves타입으로 변환할 수 있습니다.
이 경우 기존 커브에 있던 모디파이어는
모두 적용되어 사라집니다..

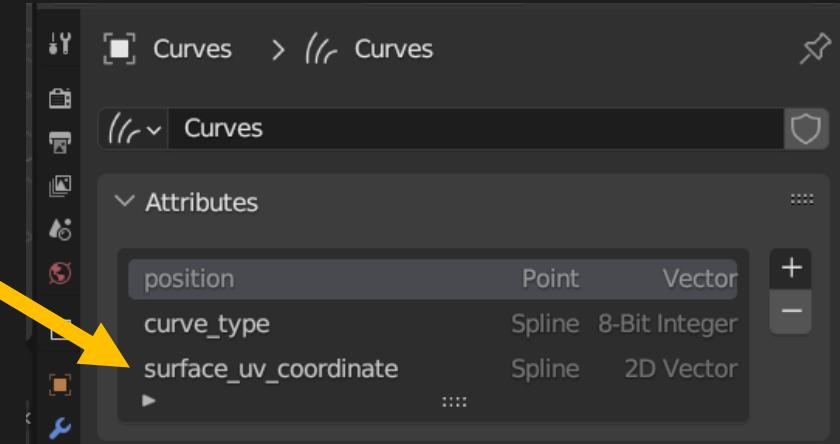


혹은 Curves가 붙은 오브젝트를 선택하고 Shift+A로
Curve – Empty hair 혹은 fur를 선택합니다.

Curves Properties

Curves 오브젝트는 단독으로도 사용할 수 있지만, 헤어를 만드는 데 특화된 여러가지 기능을 제공합니다.

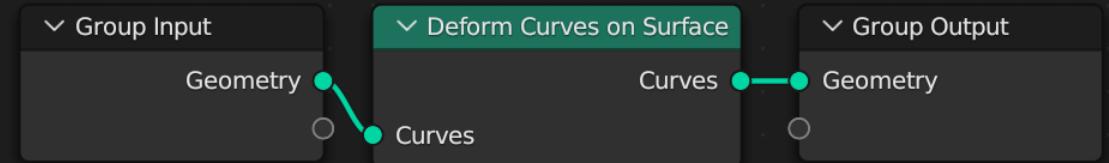
- Shift+A로 Curves 생성시 자동으로 오브젝트와 parenting이 이루어집니다.
- Sculpt 모드를 이용하여 Parent에 커브를 심고 다음을 수 있습니다.
이렇게 심어진 커브는 **surface_uv_coordinate**로 현재 위치의 UV를 기록합니다.



Deform Curves on Surface

Curves 오브젝트를 만들면, 자동으로 Deform Curves on Surface라는 노드가 연결된 지오메트리 노드를 생성합니다.

이 노드는 Parent가 변형될 때 (예컨대 Armature deform) 그 표면을 따라가게 해 주는 역할을 합니다.



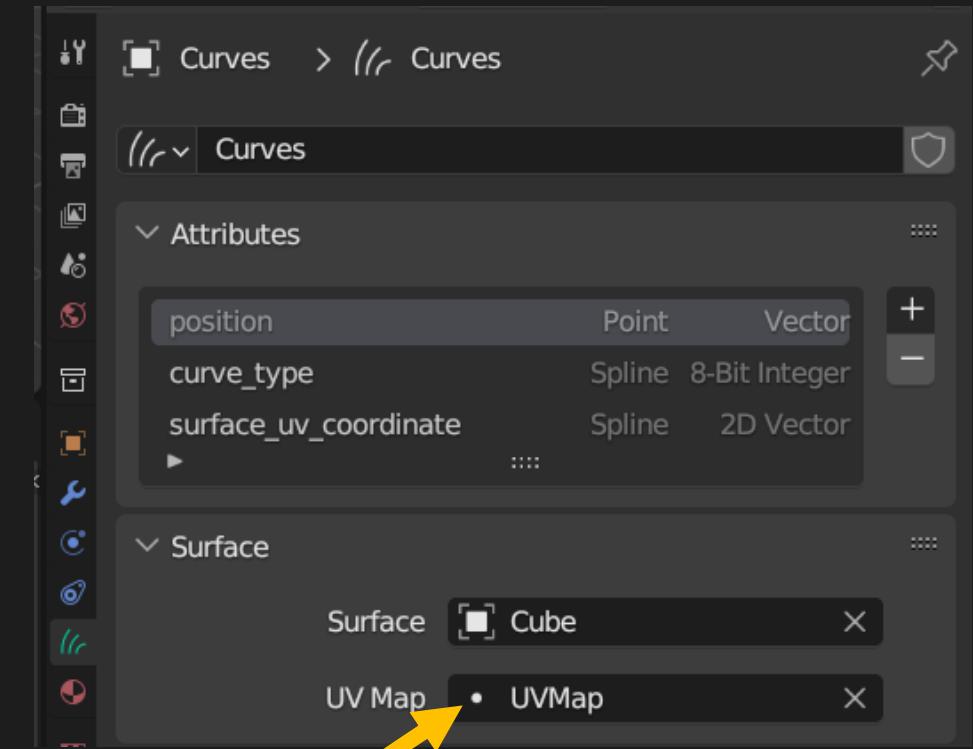
Deform Curves on Surface 노드는 **surface_uv_coordinate** 를 읽어 작동합니다.

UV를 이용한 위치 추적

헤어가 심어질 때 surface_uv_coordinate에 표면의 UV를 기록합니다.
Deform Curves on Surface는 표면이 변형될 때 이 기록된 UV를 바탕으로 위치를 바꾸게 됩니다.

따라서 Curves에는 Parent의 UV Map이 등록되어야 하며,
이 UV는 중복되는 면이 없어야 합니다.

※위치를 샘플링하는 용도이므로 예쁘게 펴져 있을 필요는 없습니다.



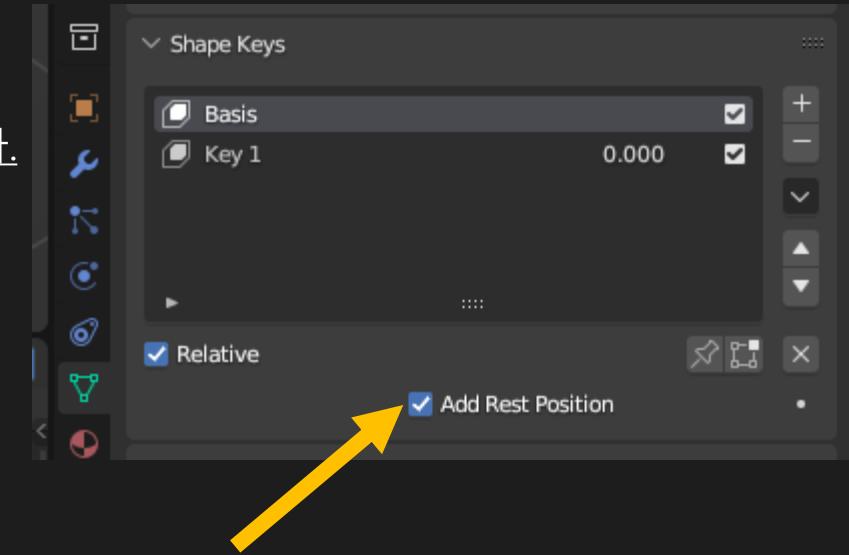
Rest Position

변형 전의 위치를 기억하기 위해서, Rest Position이라는 개념이 추가되었습니다.

Mesh Properties의 Shape Keys에서 설정하지만, 실제 작동은 Shape Key가 없어도 됩니다.

Add Rest Position을 체크하면, 어떤 식으로든 **변형되기 전의 포인트 위치**를 'Rest Position'이라는 이름의 Attribute에 저장합니다.

헤어 노드그룹에서, 표면의 위치를 찾기 위해 자주 활용하는 변수입니다.
예컨대, Armature Deform을 하기 전의 표면 위치를 사용하기 위해서,
표면의 Rest Position을 불러와 사용할 때가 많습니다.



헤어 노드그룹

Curves 오브젝트를 다루기 위한 노드그룹이 제공됩니다.

이들은 커브의 자동 생성부터, 조금 다듬는 정도 ~ 큰 형태의 변형까지 다양한 범위에서 사용할 수 있습니다.



헤어 노드그룹 - Generation



커브를 생성합니다.

Generate Hair Curves : 입력한 표면에 새로운 커브를 생성합니다. (현재 만들어진 커브는 무시합니다.)

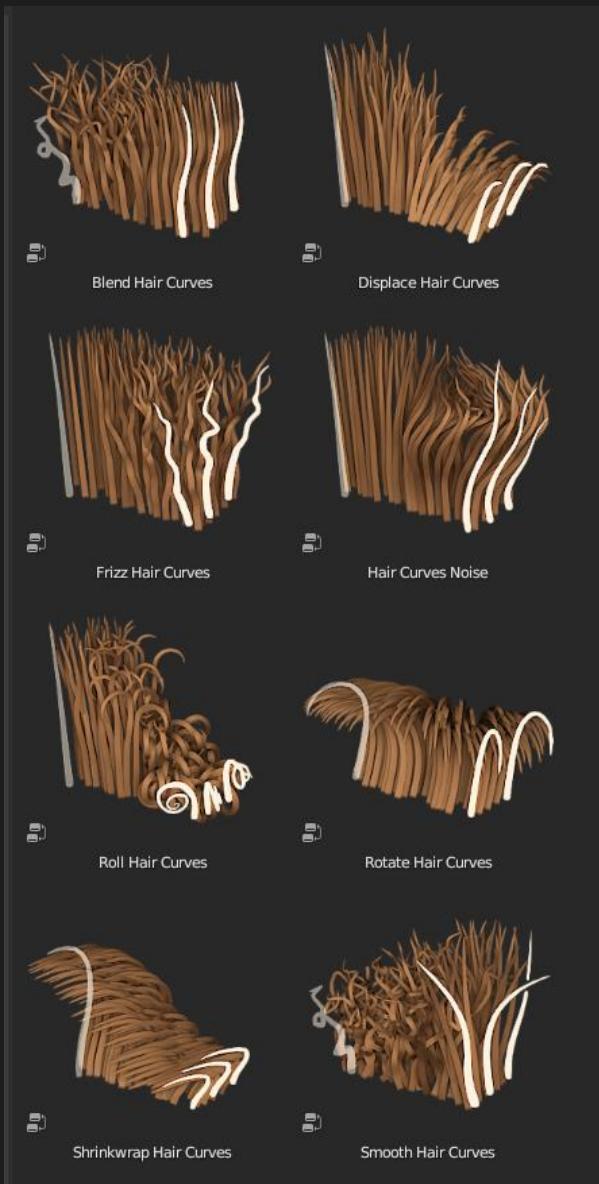


Interpolate Hair Curves : 현재 만들어진 커브를 가이드로 삼아, 표면에 새로 커브를 생성합니다.

Duplicate Hair Curves : 현재 커브를 여러 개로 복제합니다.
렌더링 할 때 머리숱을 더 많아보이게 하기 위해 사용할 수 있습니다.

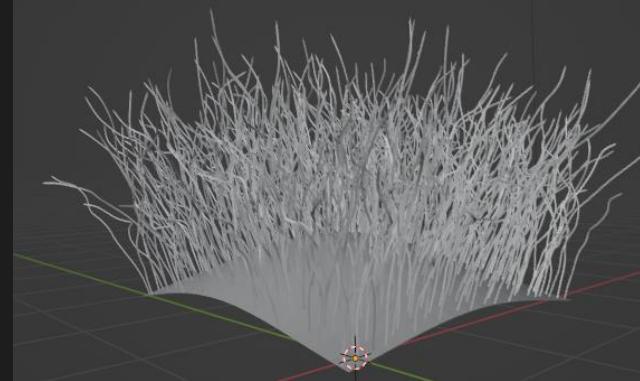
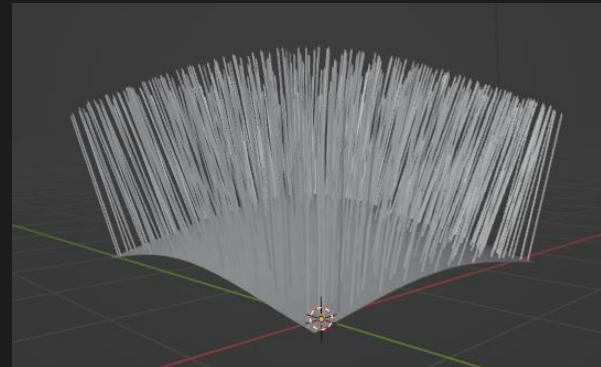


헤어 노드그룹 - Deformation, Write

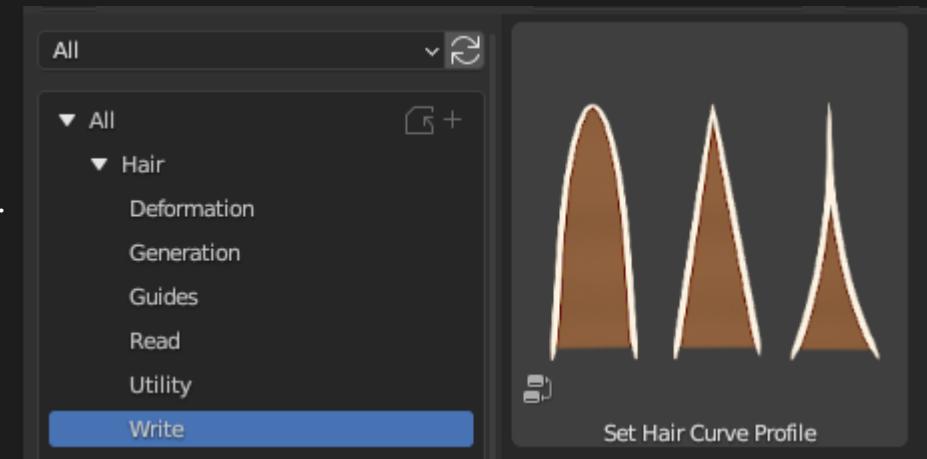


개별적인 커브의 변형에 사용됩니다.

엉키거나 펼 수 있고, 길이에 맞추어 자를 수 있습니다.



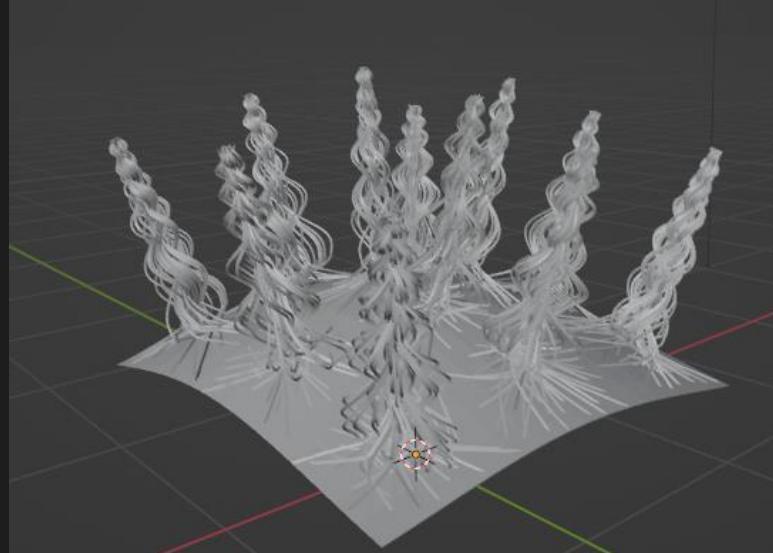
커브의 두께는 Set Hair Curve Profile로 조절합니다.



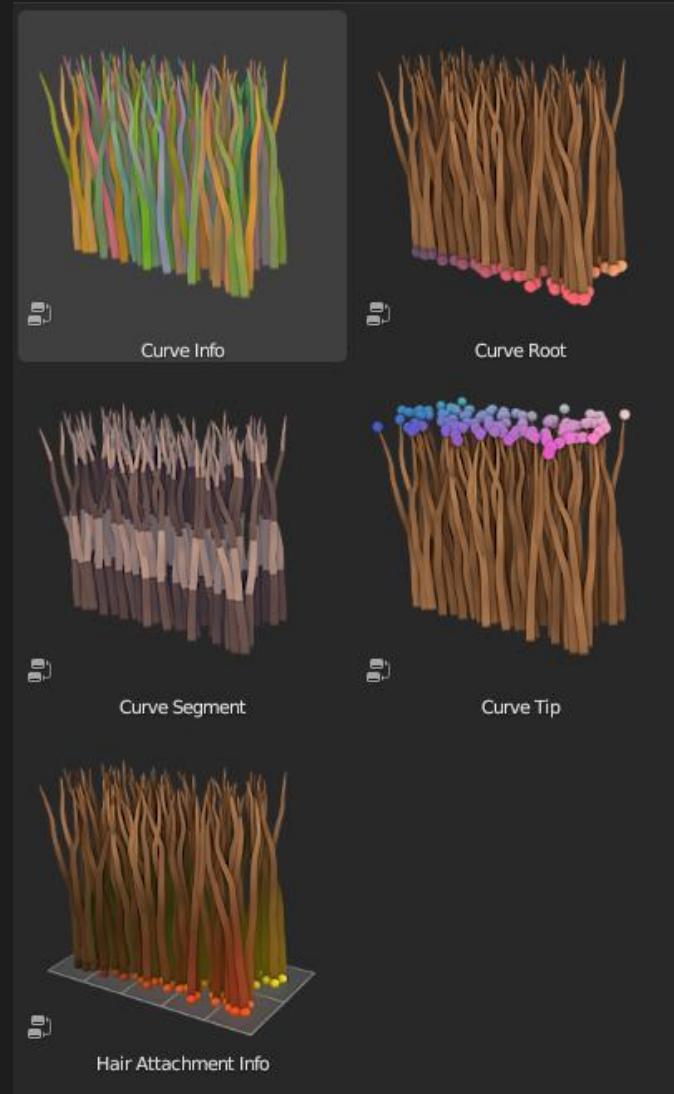
헤어 노드그룹 - Guides



여러 개의 커브를 뎅어리 단위로 변형시킵니다.



헤어 노드그룹 - Read



커브의 정보를 받기 위한 Input 노드들입니다.
독자적으로 사용할 수 없으며 노드 에디터에서 사용해야 합니다.

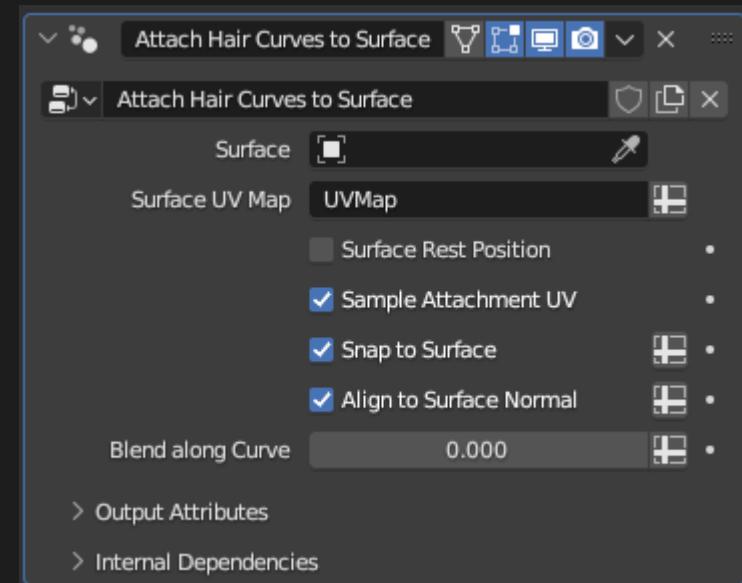
헤어 노드그룹 - Utility



Attach Hair Curves to Surface : surface_uv_coordinate를 만들어줍니다 .

Sculpt로 생성되지 않은 커브는 표면의 UV정보가 없습니다.
이런 경우에 이것을 통해 뿌리와 가까운 표면을 찾아 UV와 대응시킬 수 있습니다.

외부에서 커브를 가져올 때 유용하게 사용할 수 있습니다.



Sample Attachment UV : surface_uv_coordinate를 만들어줍니다 .

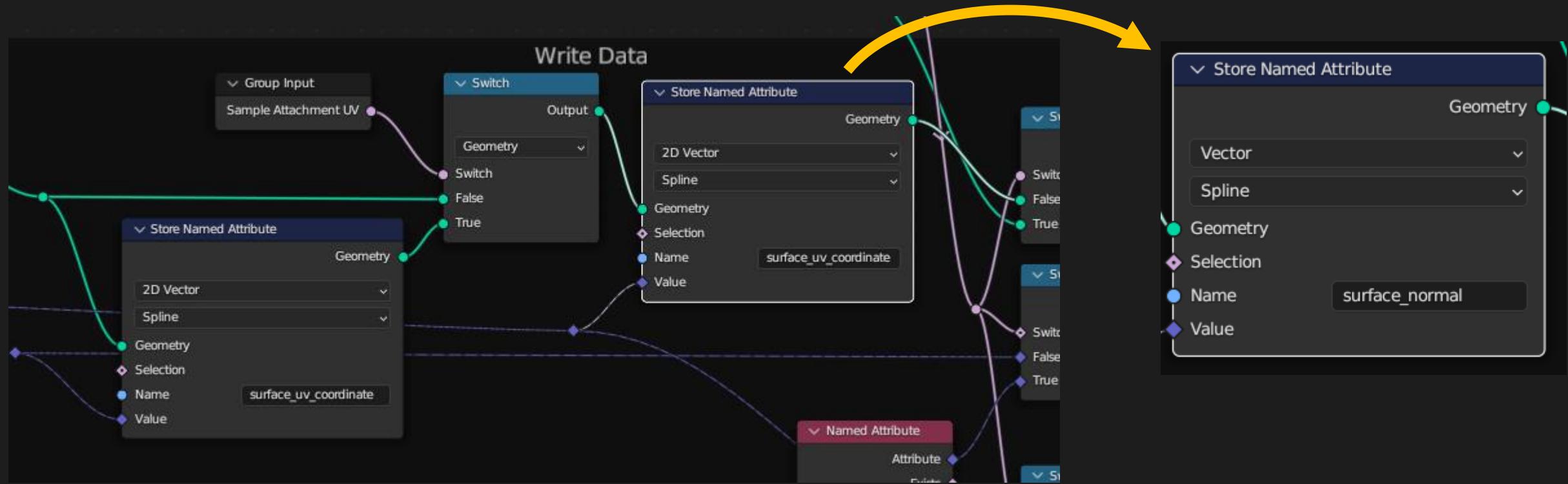
Snap to Surface : 커브의 뿌리 부분을 가까운 표면에 붙여 줍니다.

Align to Surface Normal : 커브를 가까운 표면의 노멀에 정렬시킵니다.

오류

Attach Hair Curves to Surface 노드그룹에 오류가 있습니다! (블렌더 3.5.1)

노드 뒤쪽의 두개의 Store named Attribute중, 두번째 이름을 아래와 같이 surface_normal로 바꿔주셔야 정상적으로 작동합니다.
문제가 없다면 그대로 두시면 됩니다.



Curves란?

- Curves와 Curve의 다른점 : Curves는 헤어 파티클을 생성하는데 특화되어 있는 커브 타입입니다.
기본적으로 커브이지만 커브의 몇가지 기능이 빠져있는 대신 (Cyclic, Resolution, Type)
대신 자체적으로 두께가 있고, 헤어를 만들기 위한 특수 기능을 제공합니다.
- Curves가 필요한 때 : 머리카락을 만들때 / 그 외에도 여러 개의 가는 커브를 만들 때 사용할 수 있습니다.
- Curves의 생성 방법 : 커브 오브젝트를 우클릭 – convert to Curves 하시거나,
mesh 오브젝트를 하나 고르시고 Shift-A –Curves에서 표면에 붙은 헤어를 생성합니다.

Deform Curves on Surface ?

- Deform Curves on Surface가 하는 일 : 표면의 변형 (armature deform, Shape key...)을 따라서 커브를 이동시켜 줍니다.
기본 원리는 Sample UV Surface와 비슷하기 때문에, 표면의 UV좌표와
그것에 대응하는 커브의 좌표가 필요합니다.
- surface_uv_coordinate : 각각의 커브가 표면의 UV와 대응하는 좌표값입니다. 이것이 있어야 Deform Curves on Surface가 작동합니다.
- Armature Deform을 따라가려면 : 기본적으로 커브가 몸통에 Parenting되어있어야 하고,
Deform Curves on Surface가 작동해야 합니다. 그러기 위해서는
 - 몸통의 Data Properties에서 Rest Position이 체크되어 있어야 하고,
 - Curves Properties에서 Surface / UV Map입력이 되어 있어야 하고,
 - surface_uv_coordinate 가 존재해야 합니다.

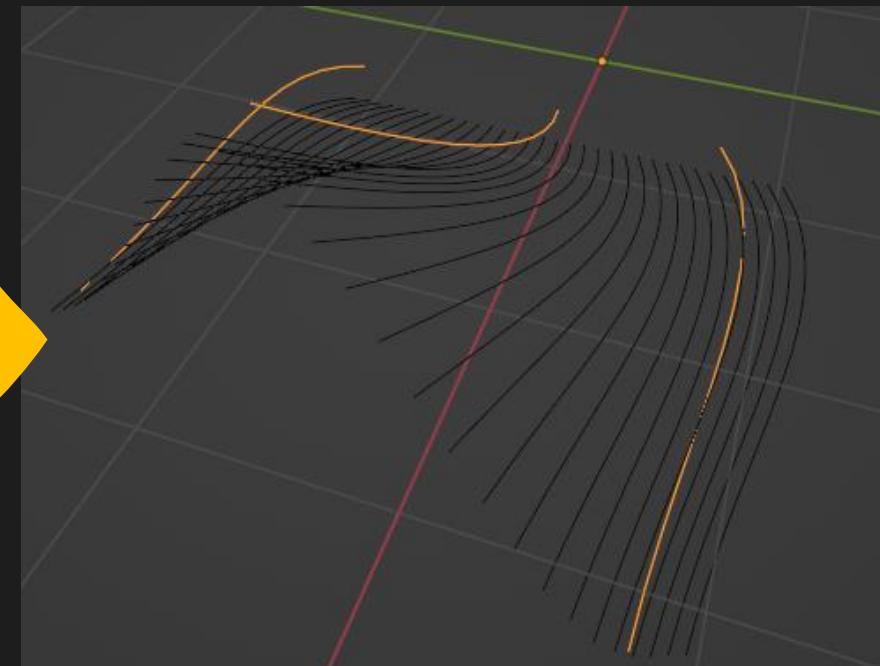
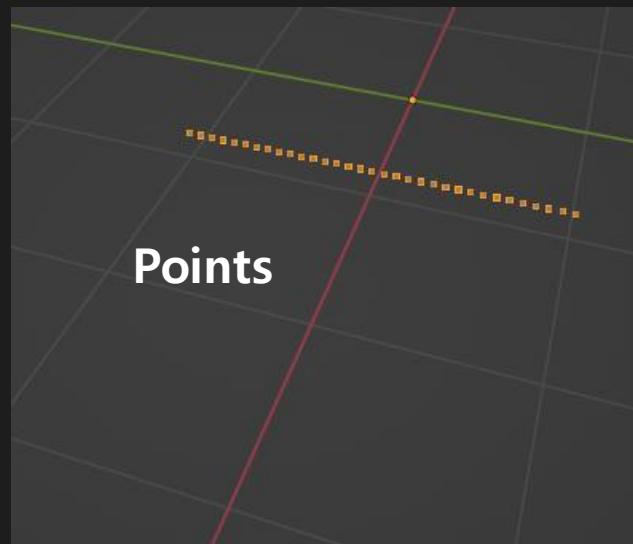
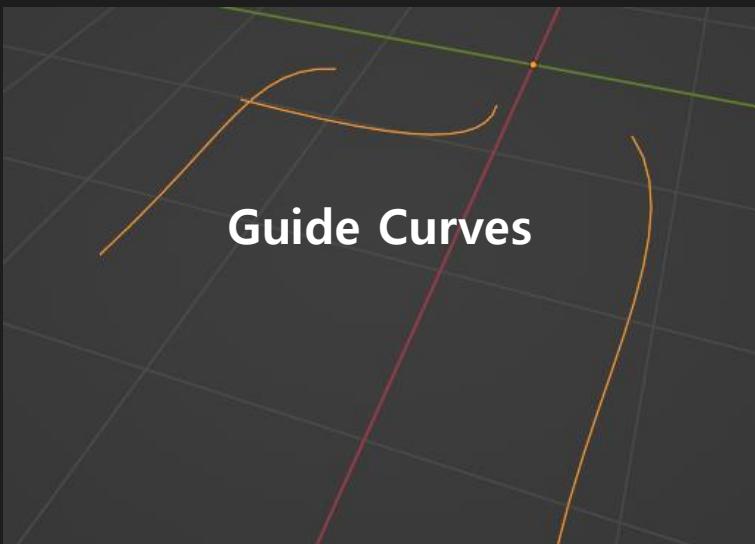
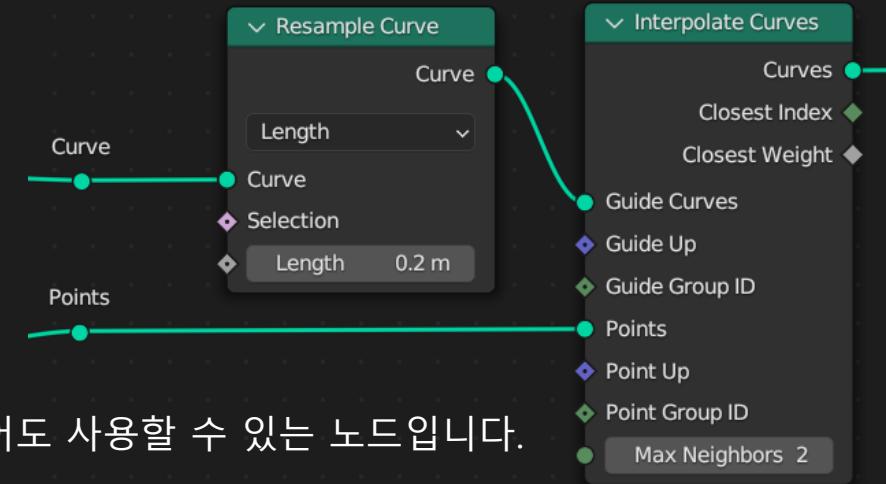
Appendix

Interpolate Curves

Interpolate Curves 노드는 입력받은 커브를 가이드로 하여
입력받은 점 위에 커브들을 생성합니다.

(Interpolate Hair Curves 노드그룹은 이것을 이용하기 편하게 다듬은 것입니다.)

Curves 자체는 단지 여러 개의 스플라인일 뿐이므로, 헤어 Curves가 아니어도 사용할 수 있는 노드입니다.



062강 Curves(2)

Curves를 통하여 지오메트리 노드로 헤어를 생성하기

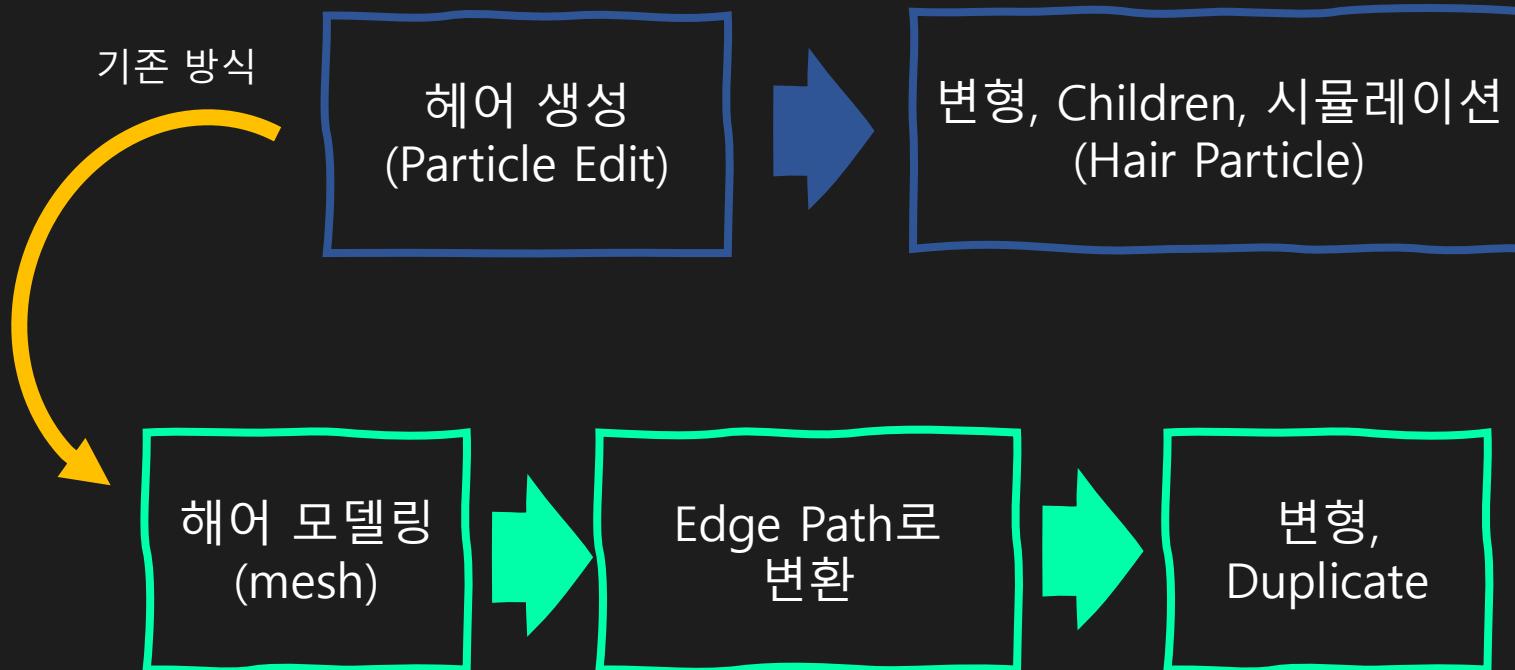


Curves를 위한 지오메트리 노드

전통적인 방식의 헤어 생성법은 Sculpt (구 Particle Edit) 입니다.

하지만 지오메트리 노드를 이용하여 새로운 제작단계를 고안할 수 있습니다.

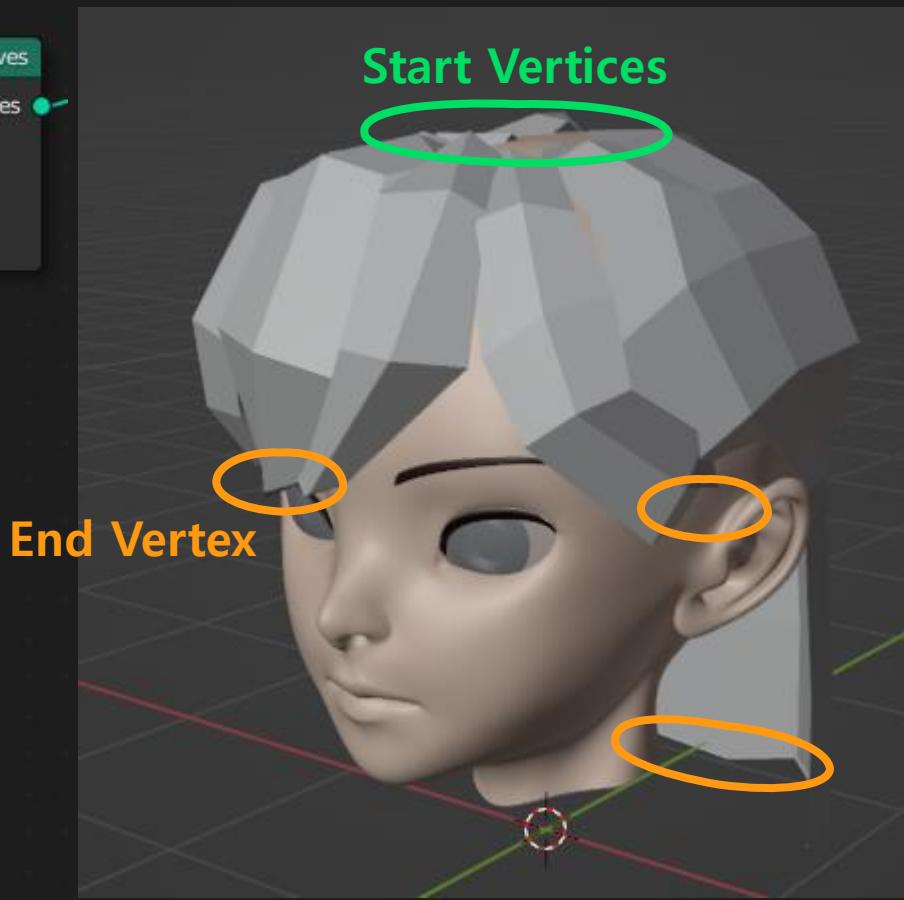
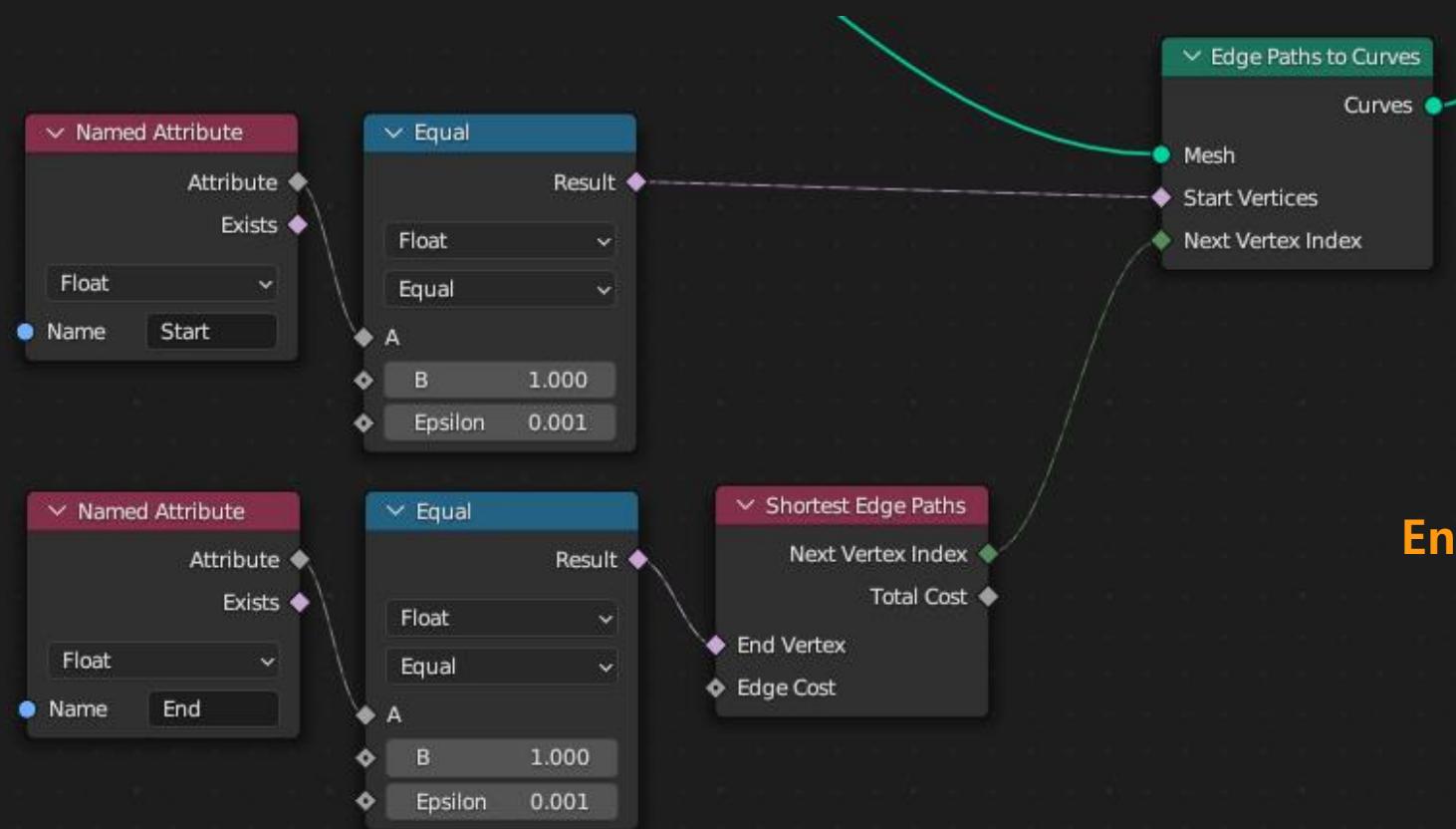
여기서는 Sculpt를 우회하는 방법을 소개합니다.



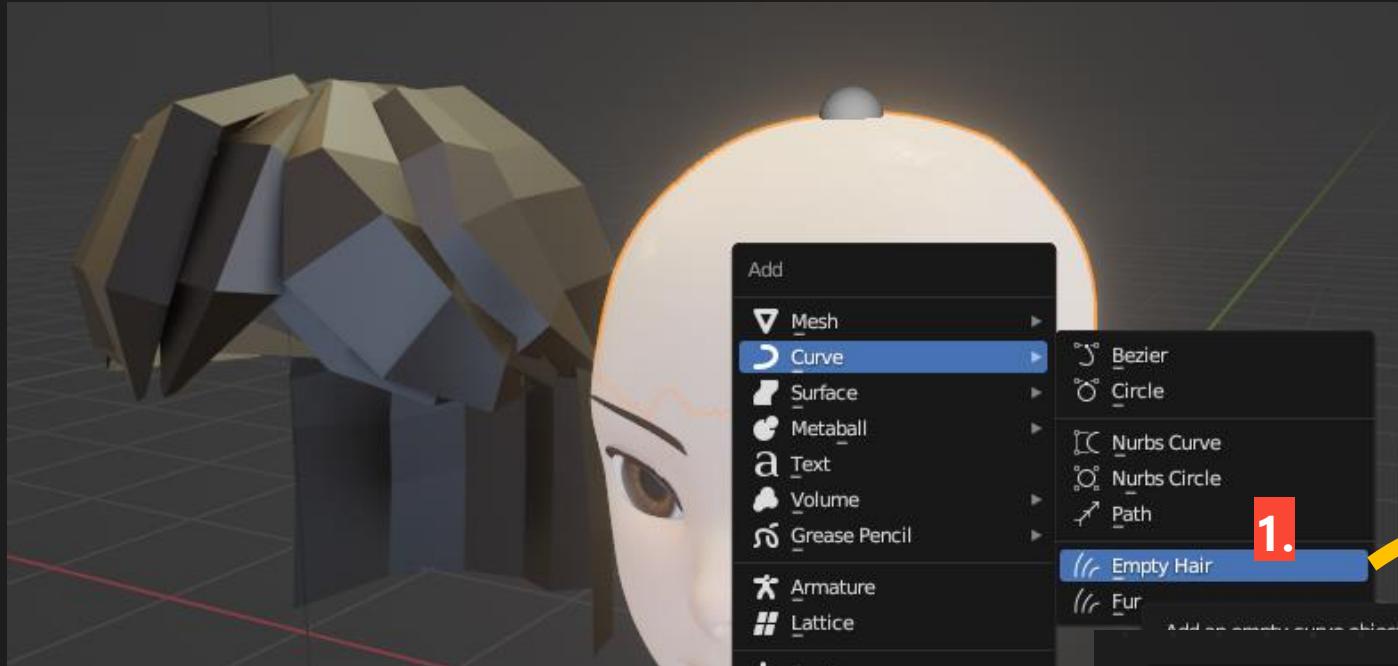
메쉬를 통한 커브 생성

Edge Paths to Curves를 이용하여 한쪽 방향 옛지만 선택해 커브를 생성할 수 있습니다.

Start Vertices와 End Vertex를 **Vertex Group**으로 지정하여 불러옵니다.



Armature Deform을 따라가려면 (1)



고정된 모델을 만들땐 상관없지만,

캐릭터를 움직이기 위해선 머리카락이
Armature Deform을 따라가야 합니다.

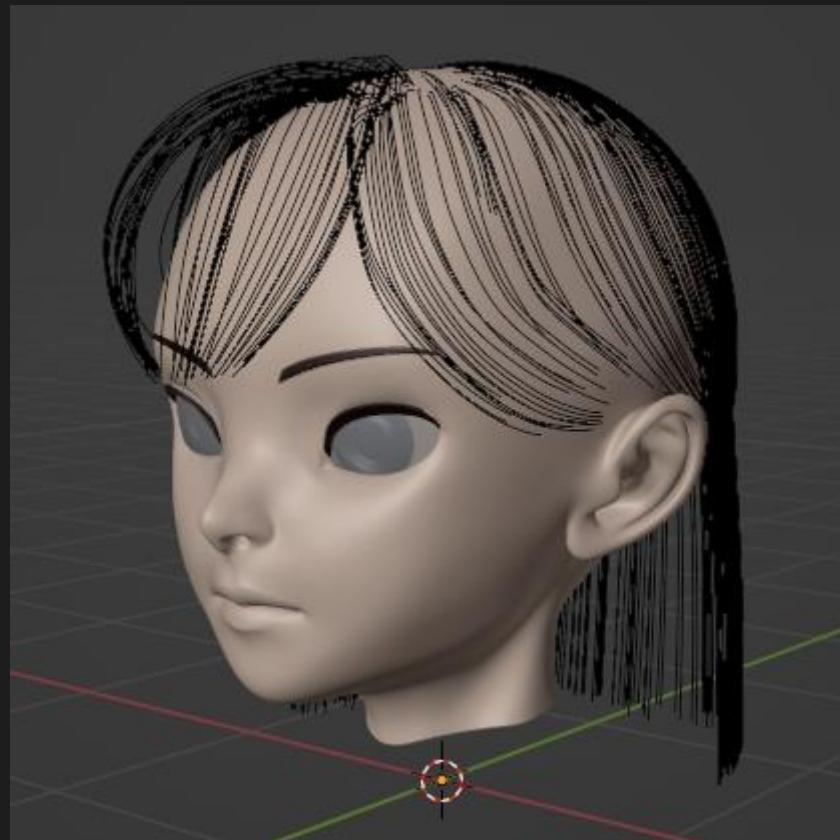
그러기 위해선 메쉬를 그냥 커브로 바꾸는 게 아니고,

1. Empty Hair를 만들고
2. 비어있는 Empty Hair에서 메쉬를 불러온 뒤,
3. 그것으로 커브를 만듭니다.

Armature Deform을 따라가려면 (2)

4 .Attach Hair Curves to Surface를 사용해서 surface_uv_coordinate를 만듭니다.

5. 이후, Deform Curves on Surface 노드를 사용하면 Armature deform을 따라갈 수 있습니다.



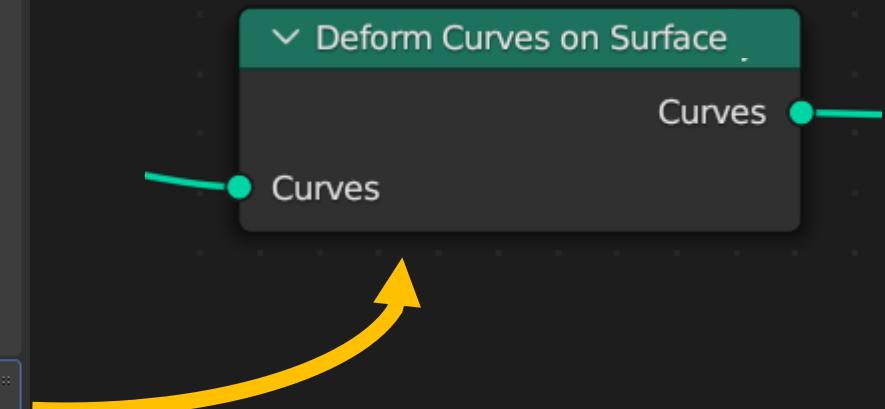
~3.

4.

5.

The screenshots show the Geometry Nodes Editor interface with three nodes:

- Mesh to Curves**:
 - Output Attributes
 - Internal Dependencies
- Attach Hair Curves to Surface**:
 - Surface: Hair Region
 - Surface UV Map: UVMap
 - Surface Rest Position (checked)
 - Sample Attachment UV (checked)
 - Snap to Surface (checked)
 - Align to Surface Normal (unchecked)
 - Blend along Curve: 0.000
 - Output Attributes
 - Internal Dependencies
- Surface Deform**:
 - Output Attributes
 - Internal Dependencies



다듬기

Duplicate Hair Curves 등, 제공된 노드그룹을 이용하여 커브를 변형시킵니다.



Shaders

Curves는 실제 머리카락만큼 가늘지 않으면 어색하게 보이기 쉽습니다.
하지만 실제로 사람의 머리카락 개수만큼 커브를 심을 수는 없으므로
두껍고 적은 양으로도 자연스럽게 보이게 해주는 셰이더 노드가 존재합니다.



Principled Hair BSDF (Cycles Only)

머리카락을 표현하기 위해 특화된 셰이더입니다.
사이클에서만 작동합니다.

Curves Info (구 Hair Info)

헤어 지오메트리의 정보를 불러옵니다.
Tangent Normal은 지오메트리의 노멀이 아니라 커브의 노멀을 불러옵니다.
이것을 노멀 대신 사용하면, 한올 한올의 굴곡은 무시하므로,
Principled Hair를 쓸 수 없는 EEVEE에서 자연스러운 머리 재질을 만드는데
큰 도움이 됩니다.

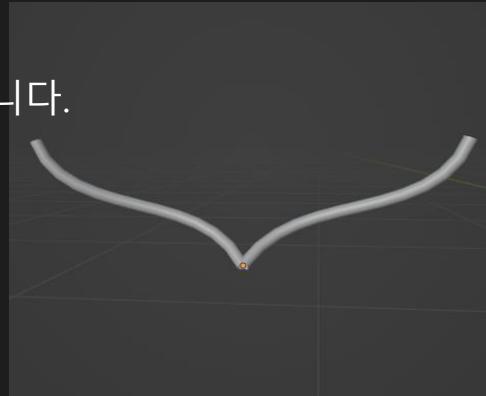
063강 Curves(3)

Curves를 통하여 깃털을 만들기

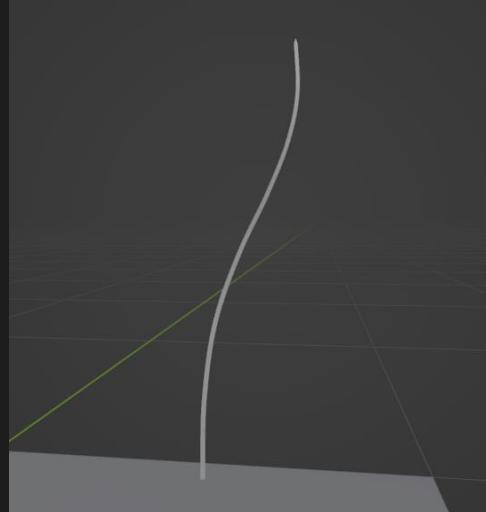


Feather

깃털은 커브에 또다른 커브가 붙어있는 형태입니다.
Instance on Points를 통하여 만들 수 있습니다.



Instance



Points



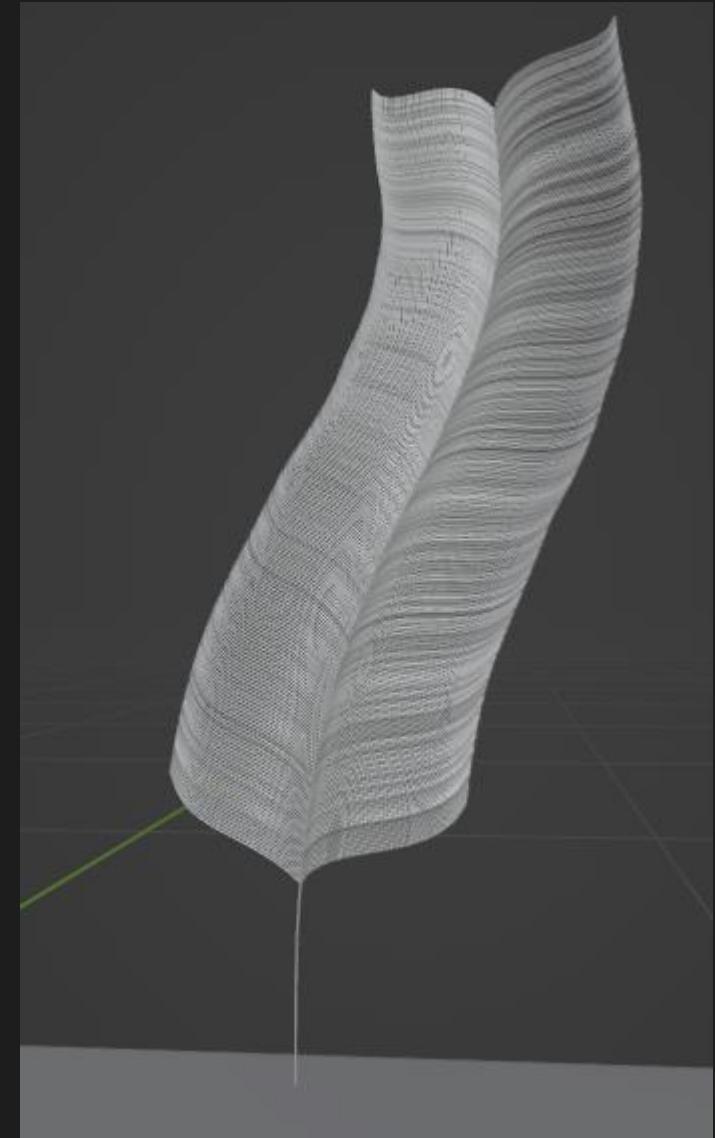
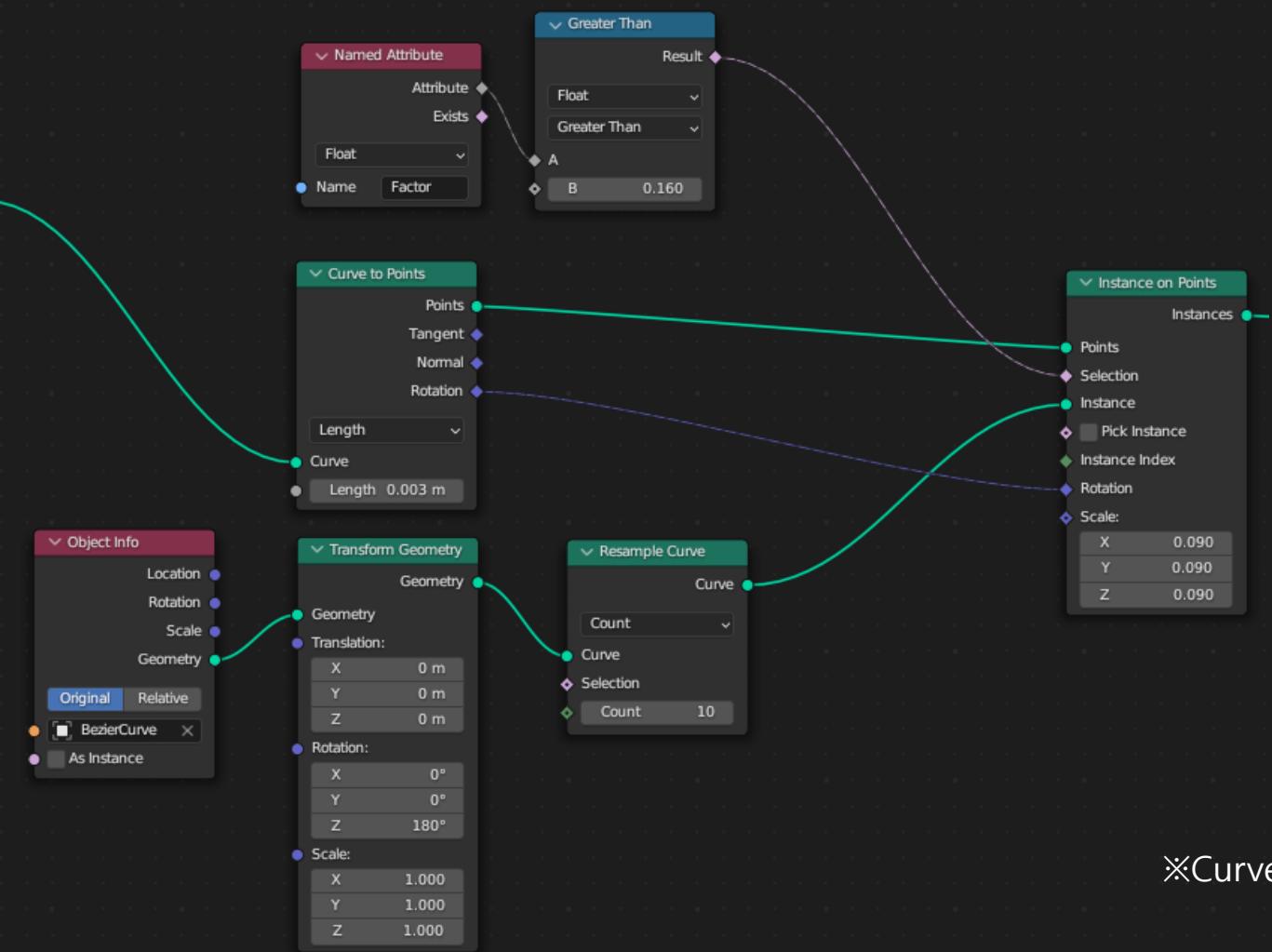
Instance on Points

Instance on points

베지에 커브로 깃털 한 올을 만들어 커브에 붙입니다.

Curve to Points를 이용하면 점들의 간격과 깃털의 회전값을 쉽게 얻을 수 있습니다.

다만 Spline Parameter가 전달되지 않으므로 Factor 등의 정보를 이전에 저장해 두어야 합니다.



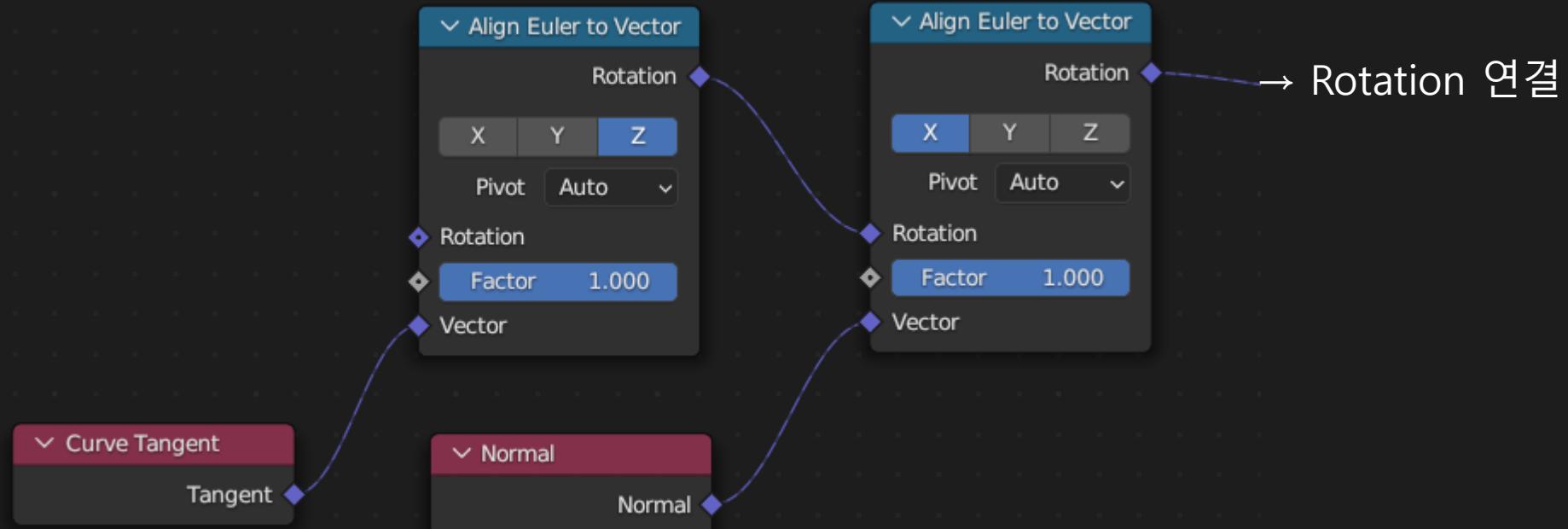
※Curves는 인스턴스를 제대로 표현하지 못하므로 나중에 Realize해 줍니다.

Curve to Points를 사용하지 않는 경우(Optional)

Curve to Points를 사용하지 않는 경우 Rotation을 직접 만들어야 합니다.

깃털은 커브의 두개의 축, Tangent와 Normal에 모두 정렬되어야 합니다.

[Align Euler to Vector](#)를 두 번 사용하여 회전값을 얻을 수 있습니다.

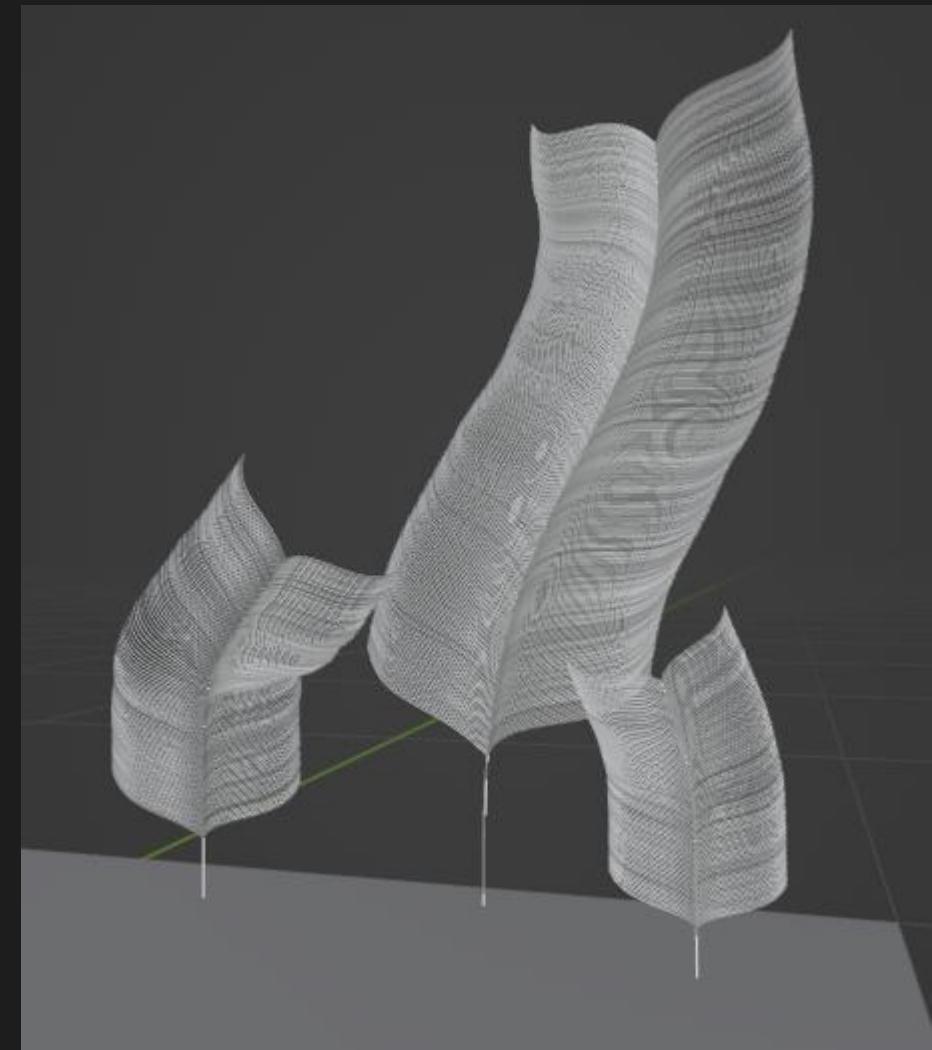
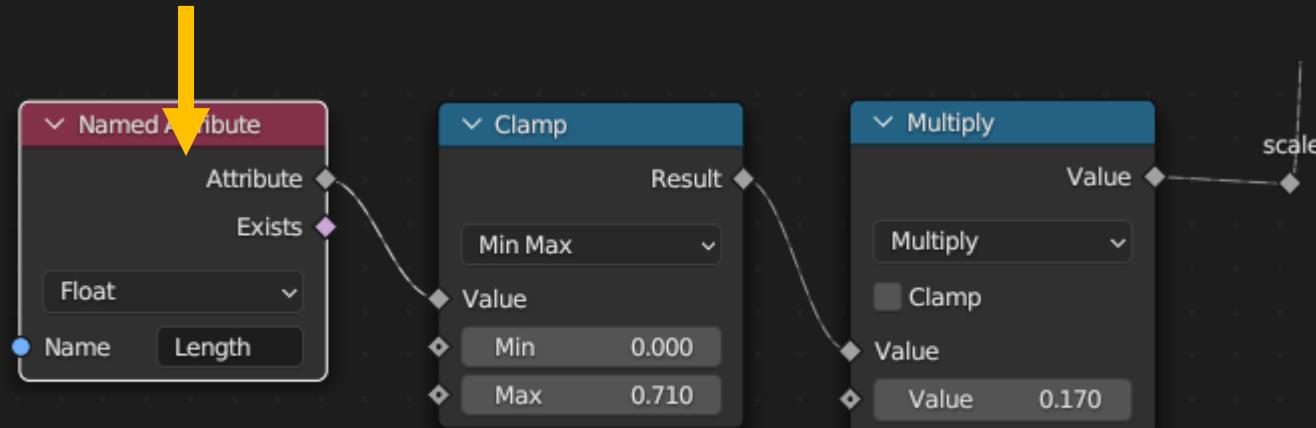


깃털 크기

원본 커브의 길이에 따라 크기를 조절합니다.

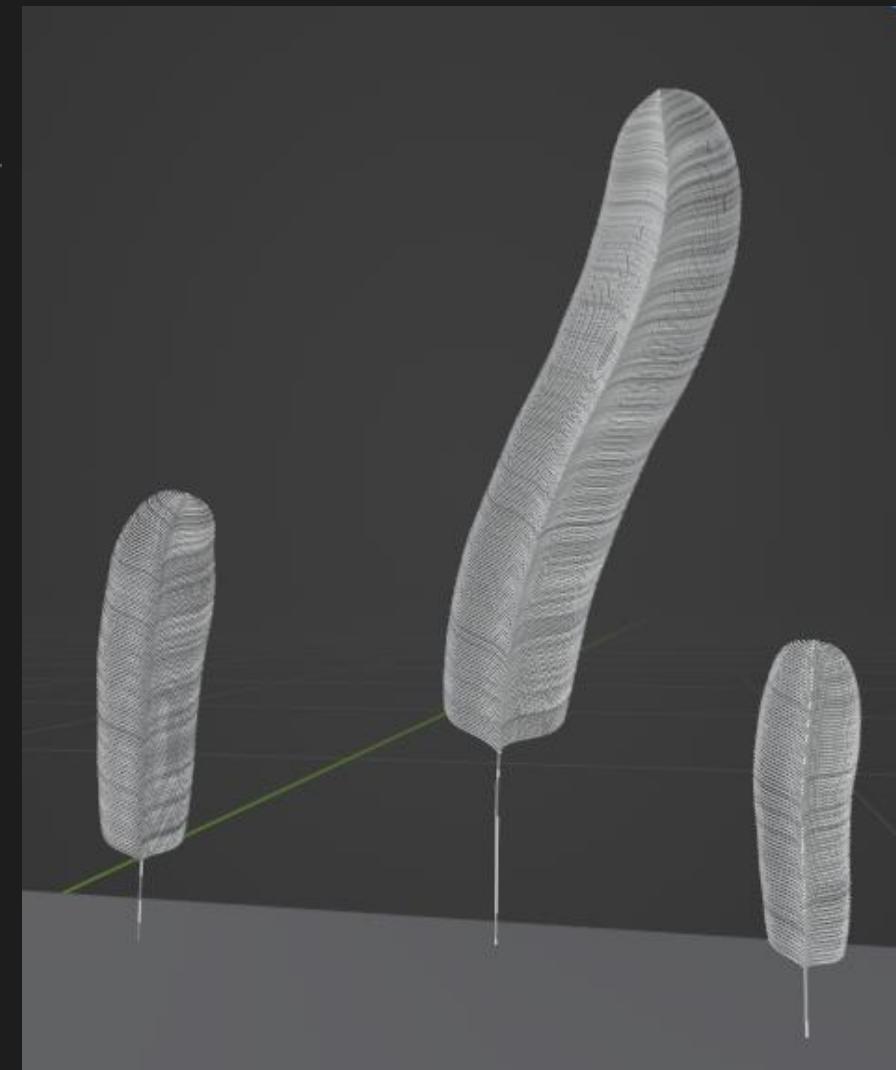
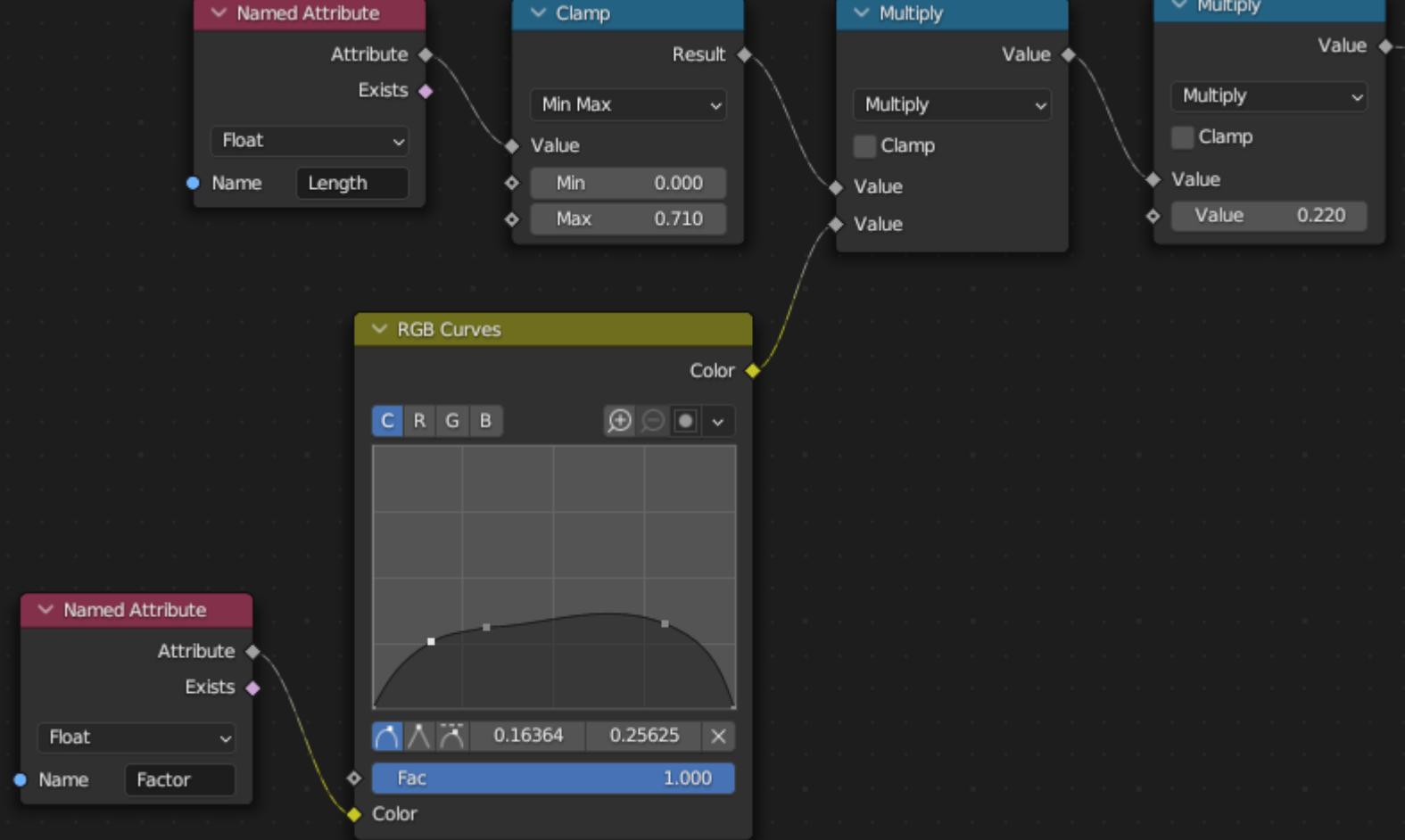
이 때, 너무 커지지 않게 Clamp로 크기 상한을 정할 수 있습니다.

Spline Length 는
Curve to Points 이전에 저장해둡니다.



깃털 모양

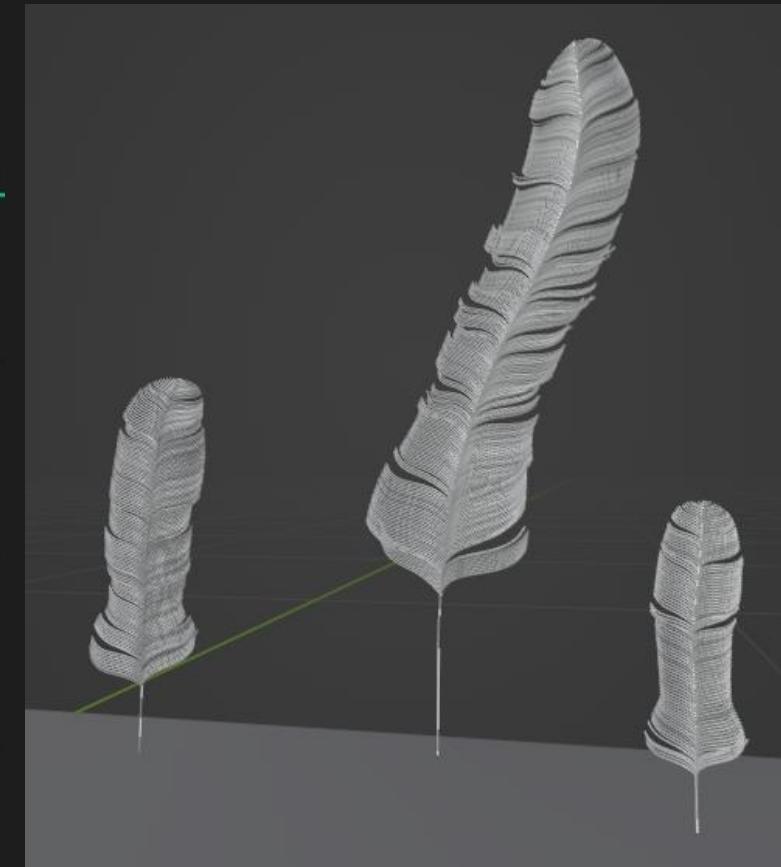
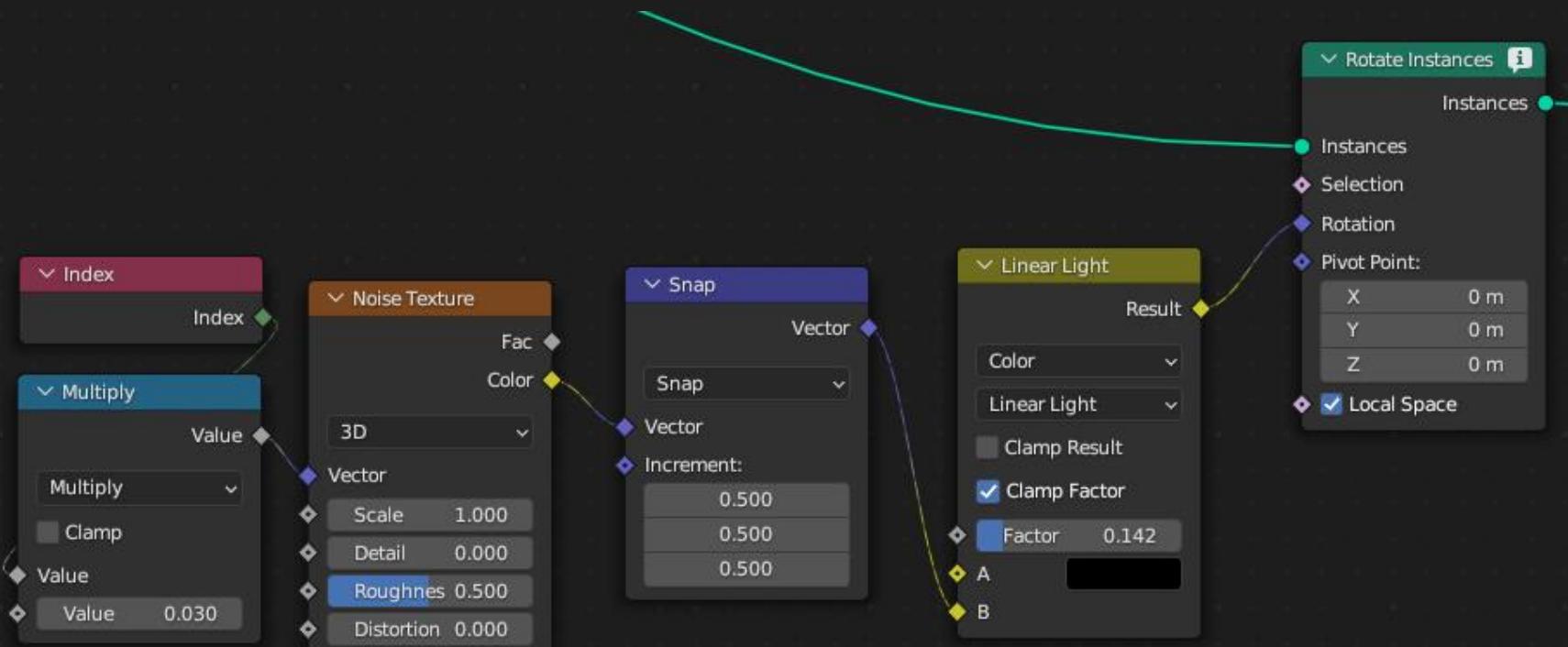
원본 커브의 Factor를 따라 스케일을 추가로 조절하여 최종적인 깃털 모양을 만듭니다.



디테일 표현

깃털의 갈라진 부분을 표현해봅시다.

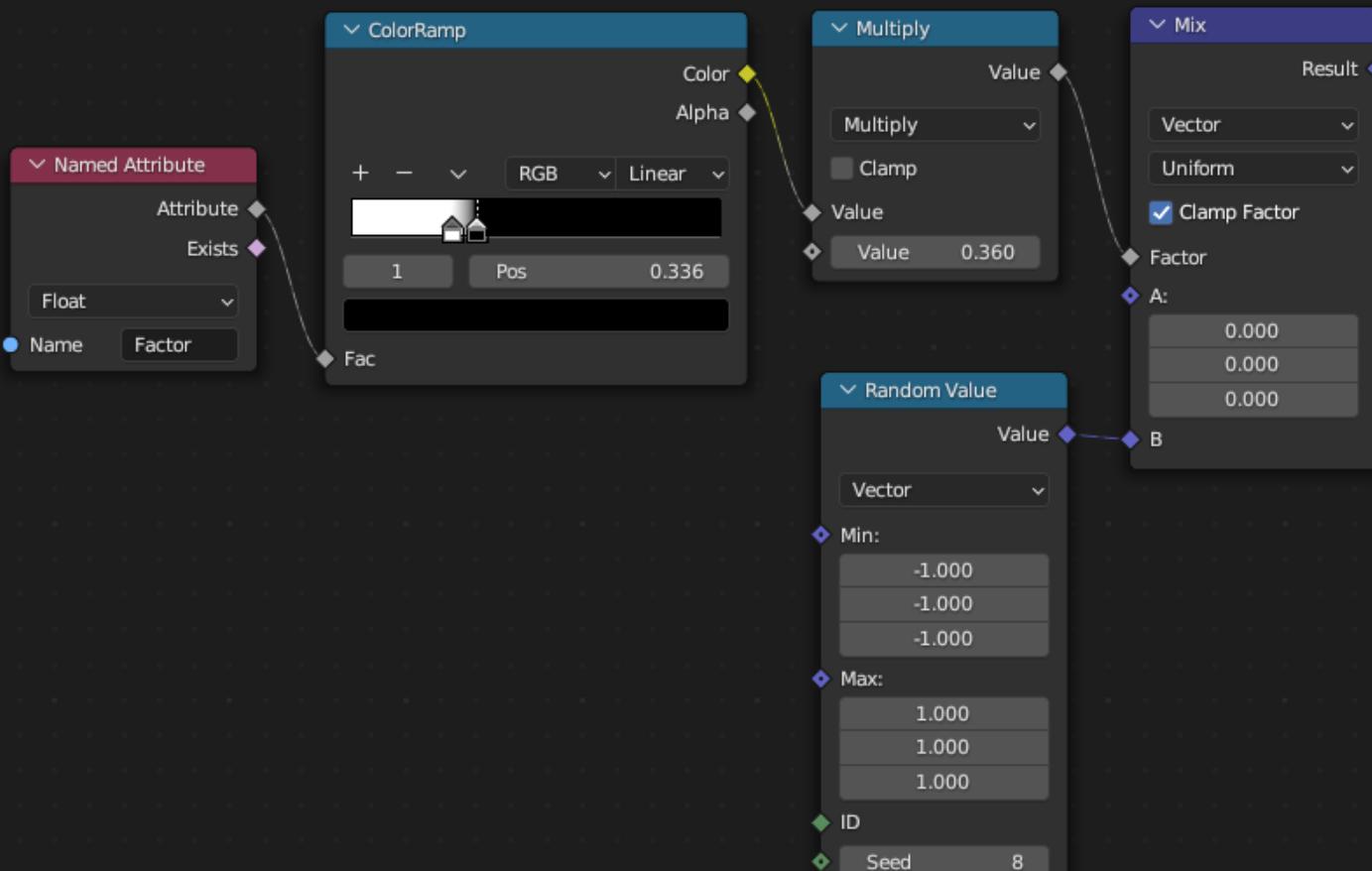
Curve의 index는 항상 정렬되어 있으므로, 노이즈의 좌표로 활용하면 원본 커브를 따라 노이즈를 생성할 수 있습니다.
이 상태에서 Snap으로 비슷한 값을 묶으면, 깃털의 뎅어리짐과 갈라짐을 표현할 수 있습니다.



디테일 표현(2)

Rotate Instances를 한번 더 사용하여, 깃털의 형클어짐을 표현합니다.

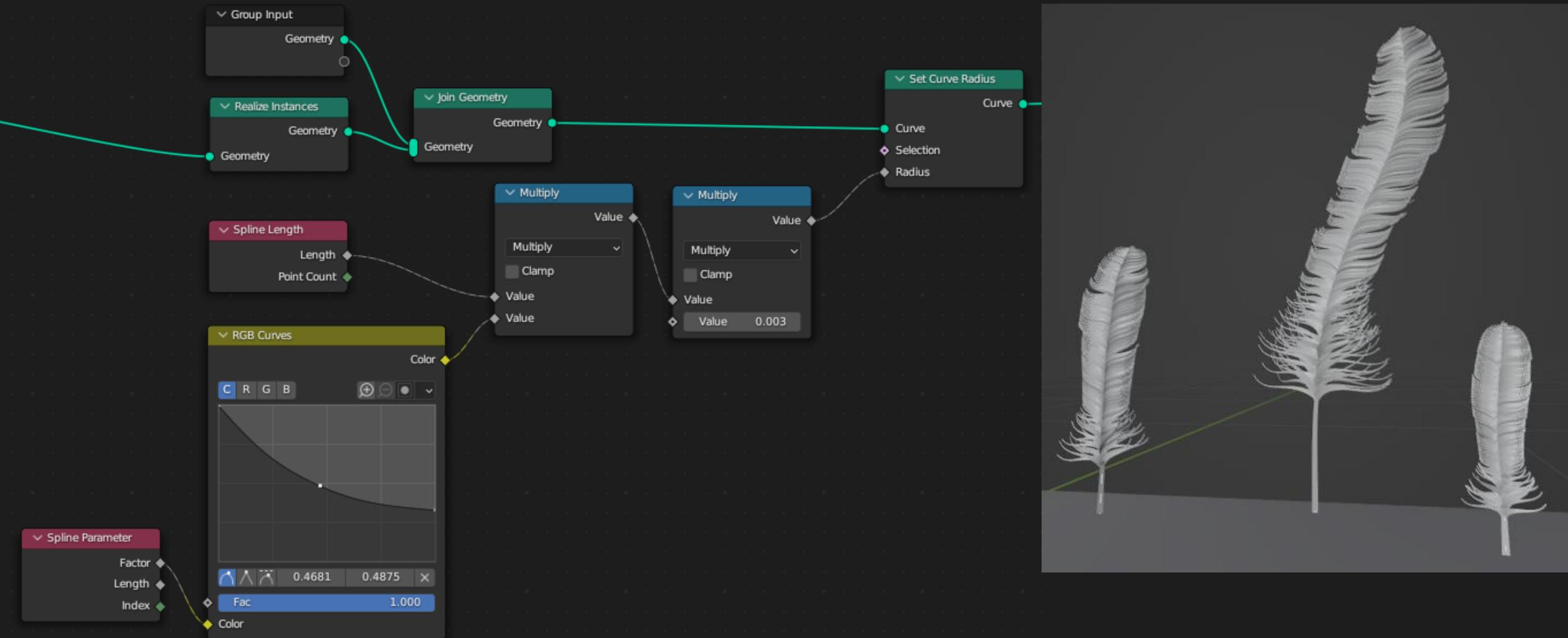
이 때는 붙어있는 깃털 한올한올 사이의 관계가 없으므로, 노이즈가 아니라 그냥 랜덤을 사용합니다.



The image shows a node editor interface with a complex network of nodes. On the left, there is a 'Named Attribute' node with 'Attribute' set to 'Exists' and 'Type' set to 'Float'. It has two outputs: one to a 'ColorRamp' node and another to a 'Random Value' node. The 'ColorRamp' node has 'Color' and 'Alpha' outputs. The 'Alpha' output goes to a 'Multiply' node, which then feeds into a 'Mix' node. The 'Mix' node's 'Result' output is labeled '→ Rotate Instance'. The 'Random Value' node has 'Value' and 'B' outputs. The 'Value' output goes to the 'Multiply' node, and the 'B' output goes to the 'Mix' node. The 'Mix' node also has a 'Factor' input. The 'Mix' node is connected to a 'Vector' input of a 'Rotate Instances' node. The 'Rotate Instances' node is applied to a feather mesh in the 3D view on the right, where three feathers are shown with their stems aligned along a green line.

두께 표현

Realize 후, 원본 커브와 합친 뒤 두께를 조절해줍니다.
*Eevee와 Cycles의 두께가 미묘하게 다르므로 유의하세요.



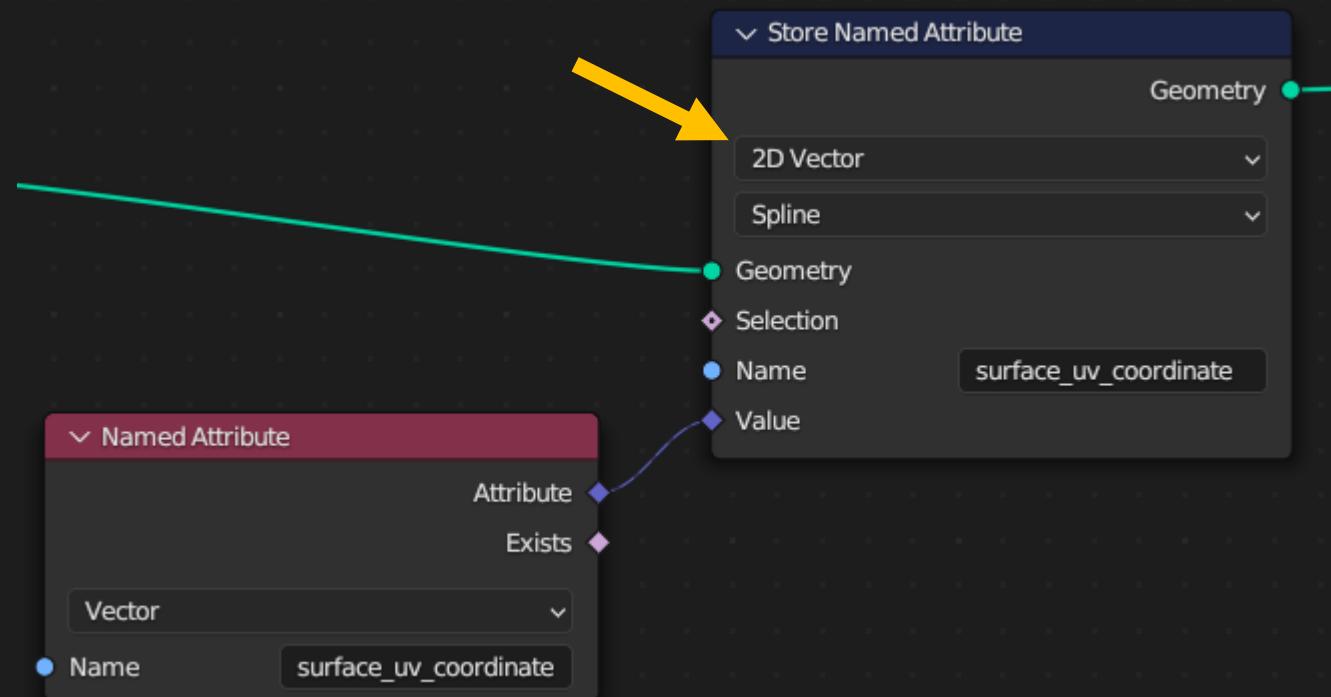
변형을 따라가기 위해

움직이지 않는 모델이라면 상관없지만, Armature deform 등 변형이 일어났을 때 커브가 따라가기 위해서는 surface_uv_coordinate가 제대로 입력되어 있어야 합니다.

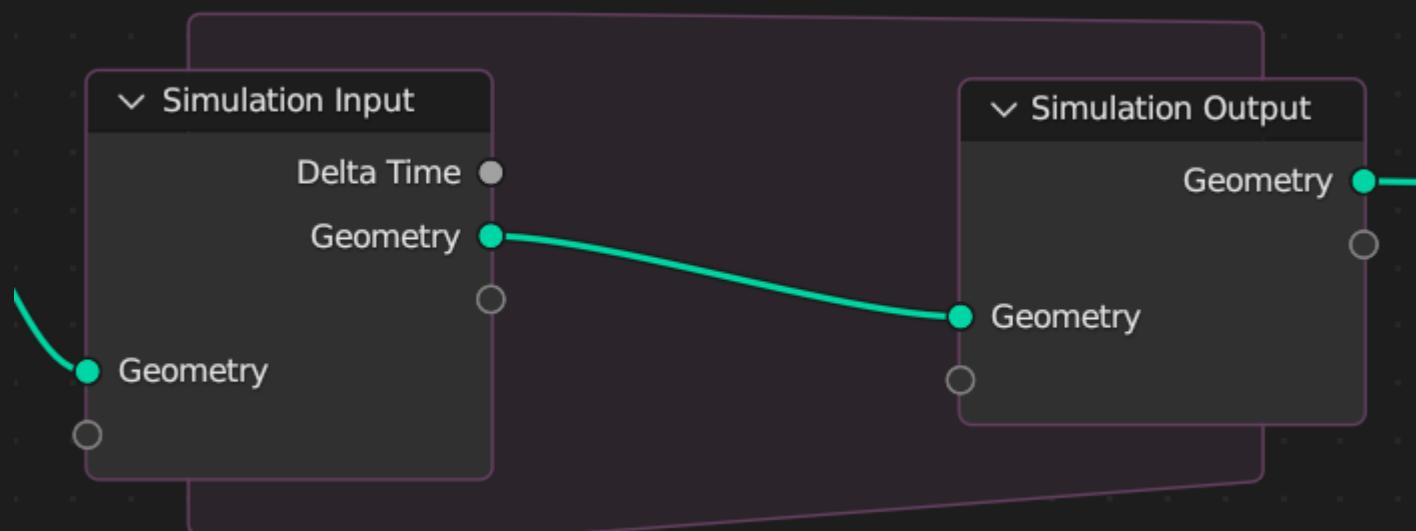
Instance가 원본 커브의 surface_uv_coordinate를 받아오지만, 도메인이 control point로 잘못되어 이동합니다.

이 문제는 도메인을 Spline으로 교체하는 것으로 간단히 해결됩니다.

이후에 deform curves on surface를 사용하면 변형을 따라가게 됩니다.



064강 Simulation Node 입문 (v3.6~)

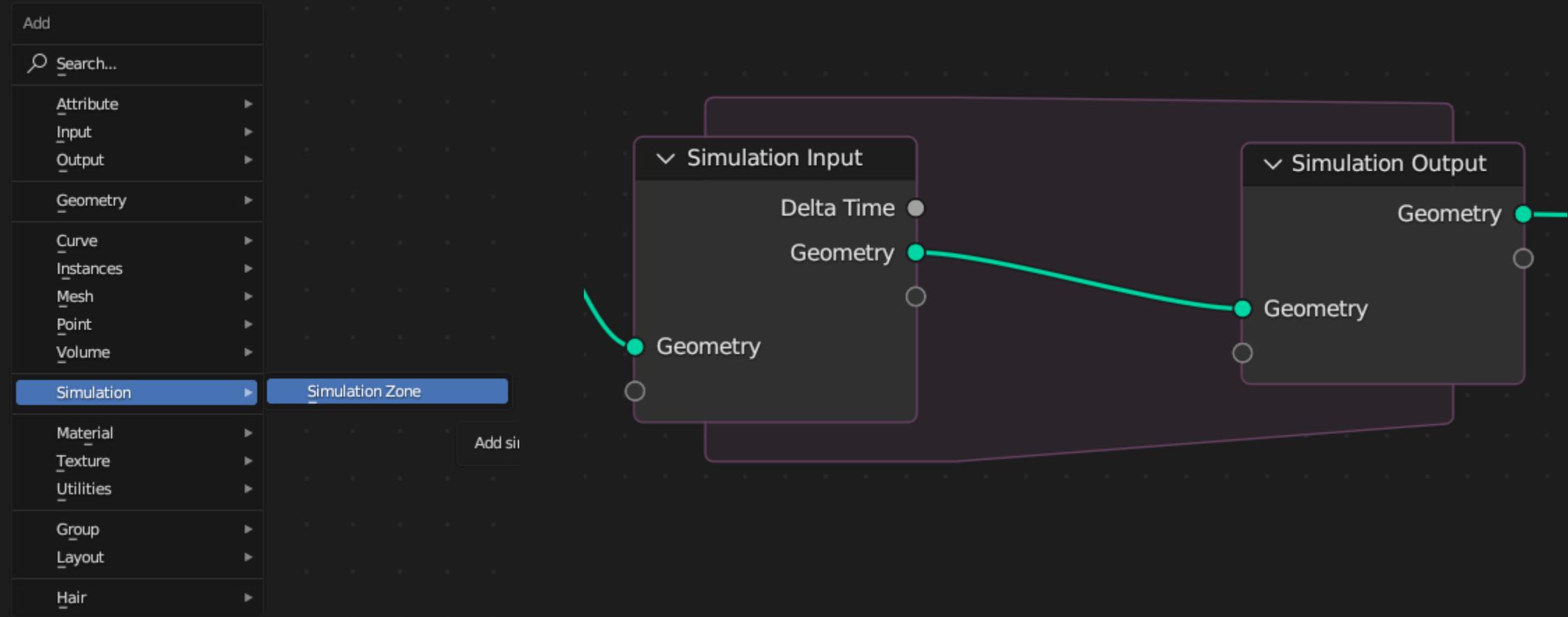


Simulation Node

블렌더 3.6에서 시뮬레이션 노드가 추가되었습니다.

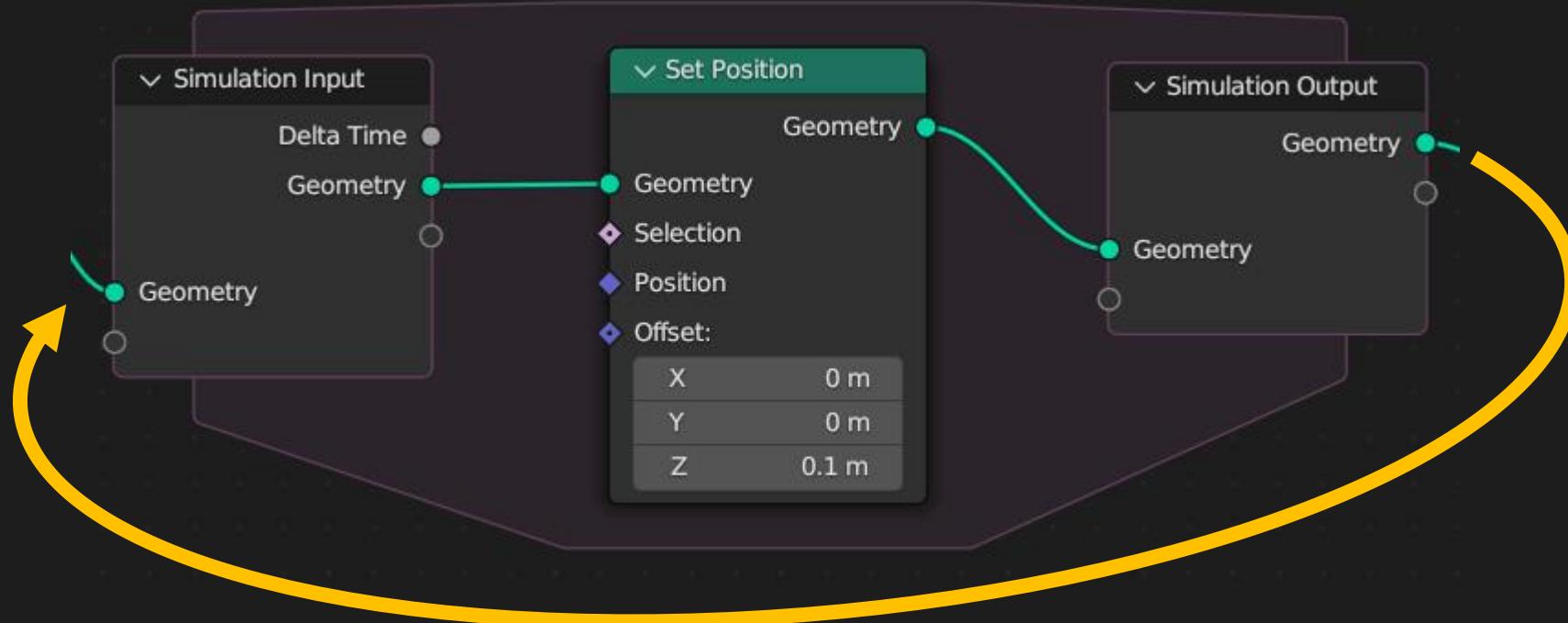
Simulation Zone을 추가하면 Simulation Input과 Simulation Output 노드를 생성합니다.

분리된 두 노드로 보이지만 둘이 연결되어 있어야 작동하며 임의로 분리시키는 것은 권장되지 않습니다.



Simulation?

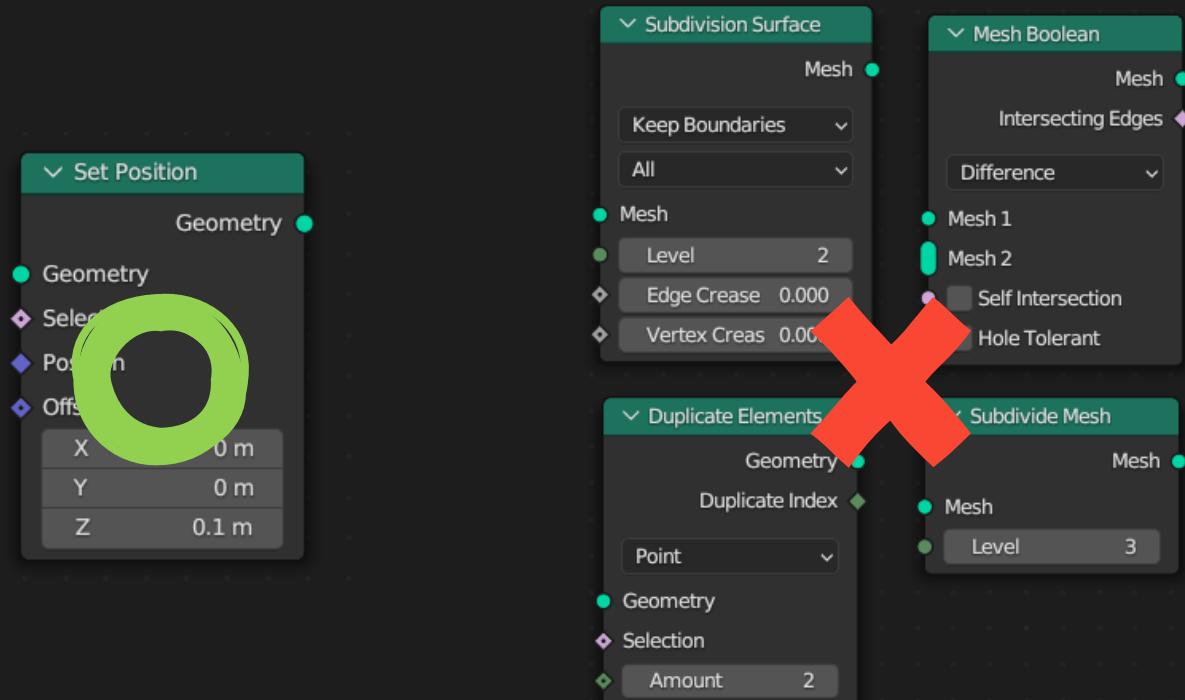
Simulation Node는 반투명 박스 내부로 표시된 Simulation Zone 내부의 연산을 반복합니다.
정확히는, 첫 프레임에서만 초기상태의 지오메트리를 받아오고,
그 다음 프레임부터는 Simulation Output의 결과가 Simulation Input으로 다시 들어갑니다!



사용시 유의사항

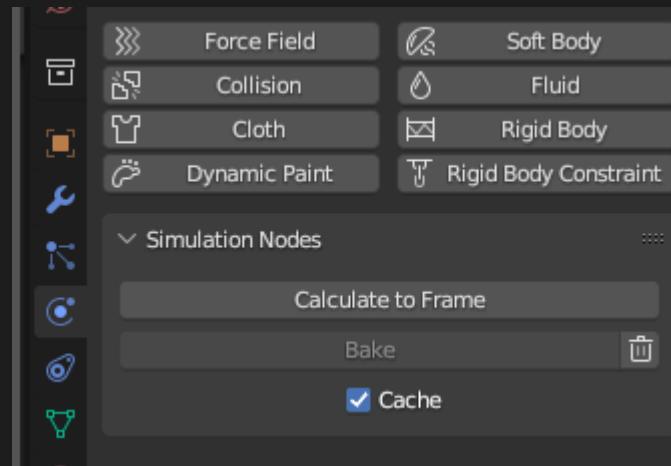
연산이 매 프레임마다 반복되므로, 시뮬레이션 존 내부에 함부로 노드를 끌으면 곤란해질 수 있습니다.

Subdivide / Duplicate : 매 프레임마다 지오메트리가 배수로 증가하므로,
말 그대로 기하급수적으로 증가하여 강제종료될 수 있습니다.
Mesh Boolean같은 연산이 무거운 노드도 곤란합니다.



데이터 저장 (3.6b)

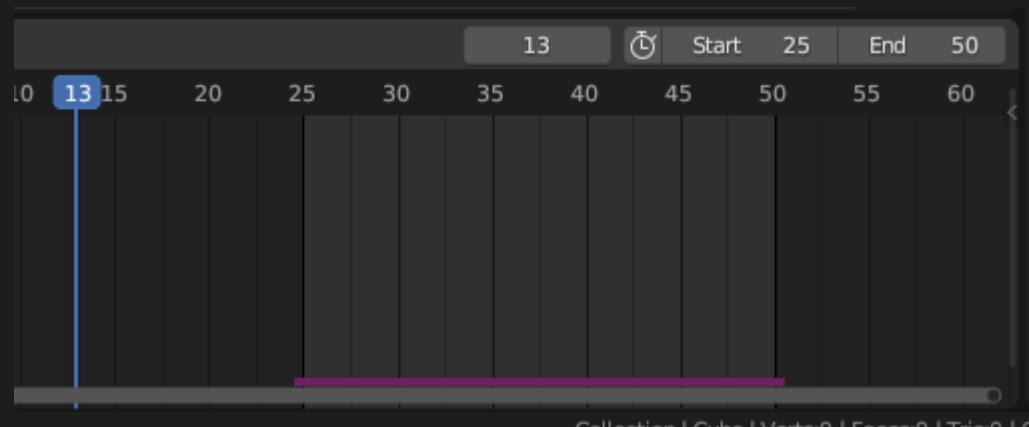
시뮬레이션 노드 데이터도 다른 시뮬레이션처럼 Physics Properties 탭에서 데이터를 저장합니다.



Calculate to Frame : 1프레임부터 현재 프레임까지 시뮬레이션 데이터를 생성합니다.

Bake : 블렌더 파일이 저장된 위치에 시뮬레이션 데이터를 저장합니다.
저장되는 프레임은 타임라인에서 설정된 프레임이며,
저장되지 않은 프레임 바깥에서는 작동하지 않습니다.

Cache : 시뮬레이션 데이터를 지속적으로 저장합니다. Bake를 위해서는 켜 있어야 합니다.

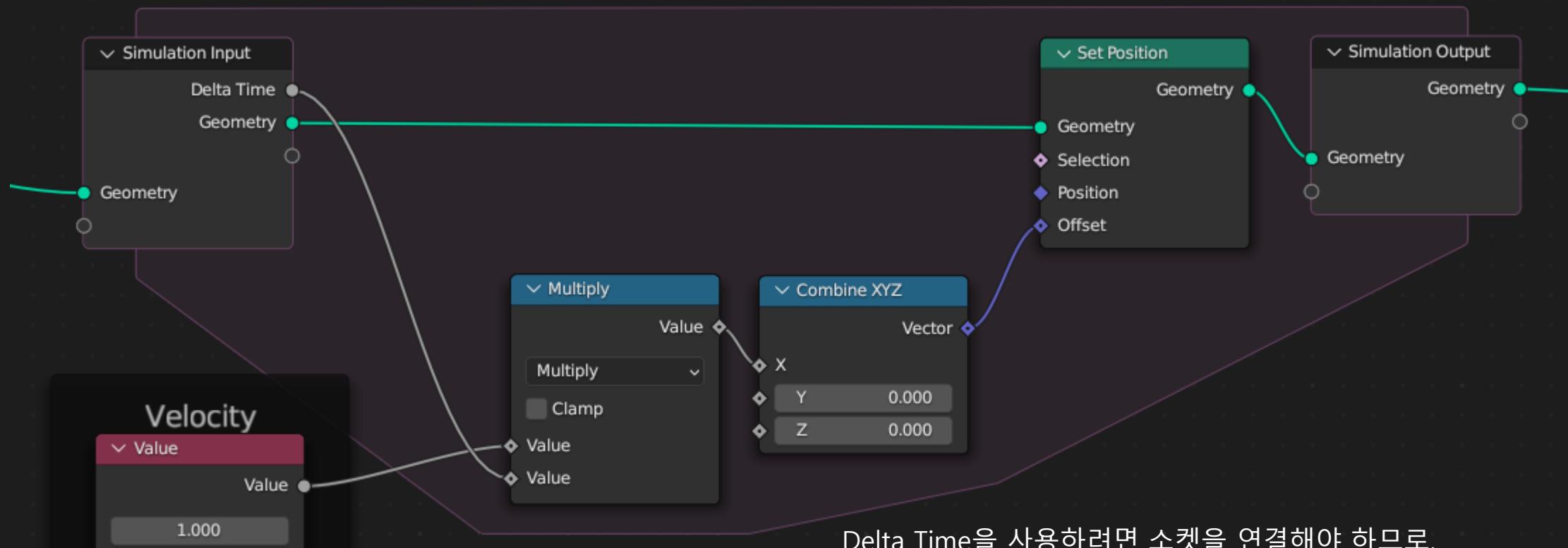


타임라인에서 시작과 끝을 제한한 뒤 Bake하면,
특정 부분에서만 시뮬레이션을 할 수 있습니다.
이 경우 바깥 (1~24프레임, 51프레임~)에서는 시뮬레이션 직전/직후 상태가
유지되는 것이 아니라 전혀 보이지 않을 수 있습니다.

시뮬레이션 노드 연습문제

1. 매초마다 1미터씩 이동하세요. (= 1 m/s로 이동하세요.)

Delta Time은 매 연산(=매 프레임)마다 지나간 시간을 잡니다.
이동거리 = 속도 x 시간 이므로, Delta Time에 1m/s를 곱하면 됩니다.

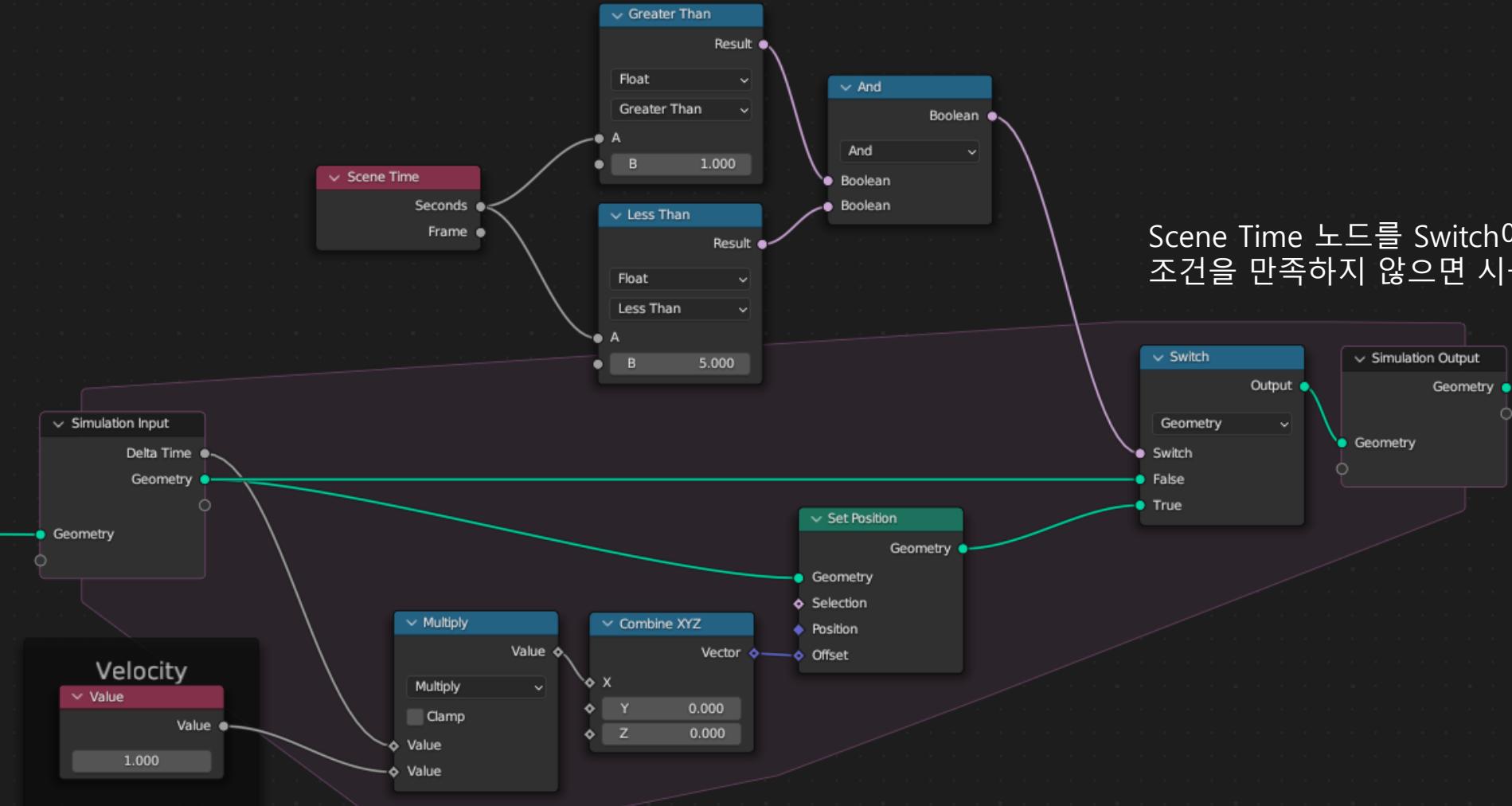


Delta Time을 사용하려면 소켓을 연결해야 하므로,
그냥 1/Frame 값을 Math노드나 Value 노드로 만들어서
사용하는게 편리할 수 있습니다.

시뮬레이션 노드 연습문제

2-A. 20프레임부터 40프레임 사이에서만 이동하세요.

2-B. 1초부터 5초 사이에서만 이동하세요.

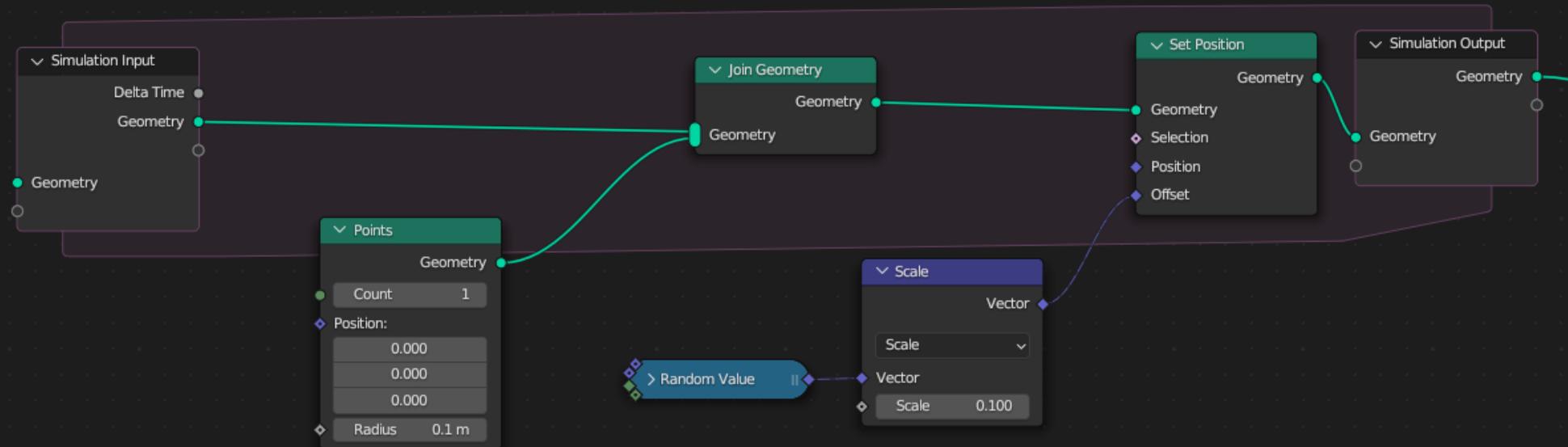


Scene Time 노드를 Switch에 연결하여,
조건을 만족하지 않으면 시뮬레이션을 패스하도록 합니다.

시뮬레이션 노드 연습문제

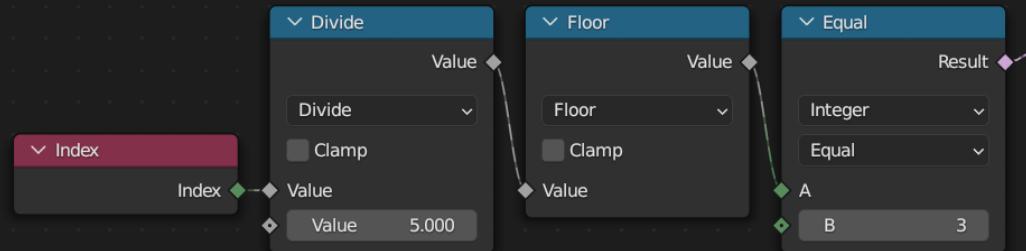
3. 랜덤 방향으로 움직이는 포인트를 매 프레임마다 만드세요.

포인트를 Simulation Input에 입력하지 않고
시뮬레이션 중간에 꽂으면, 매 연산마다 포인트가 추가됩니다.

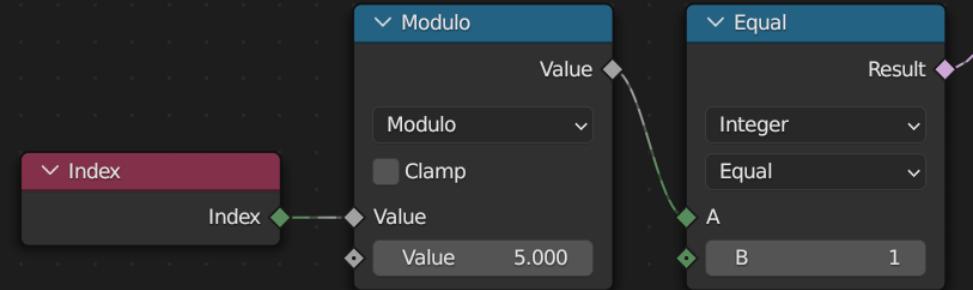


Reminder : 정수의 컨트롤

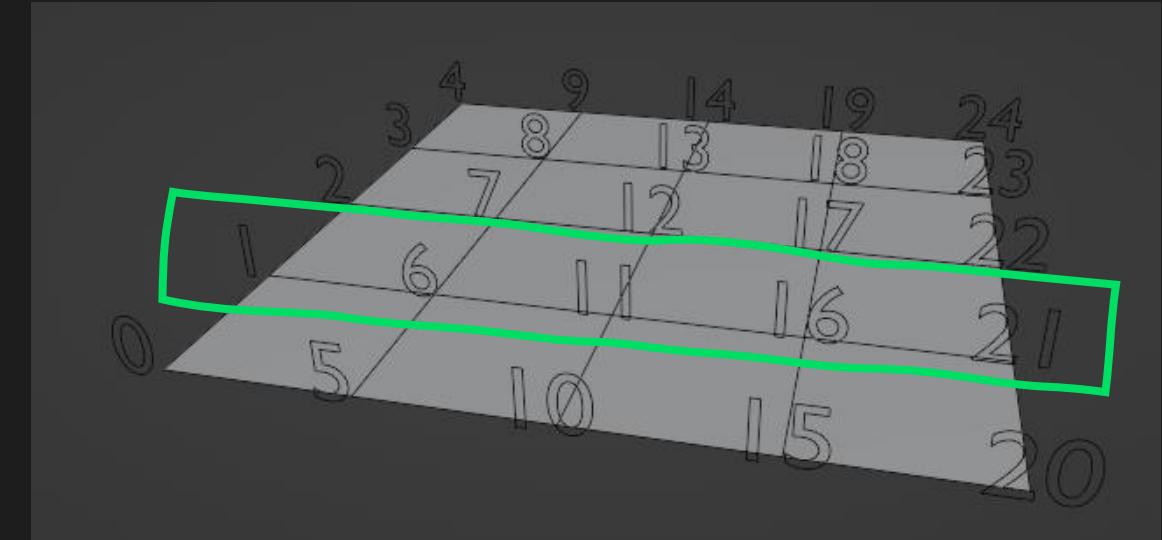
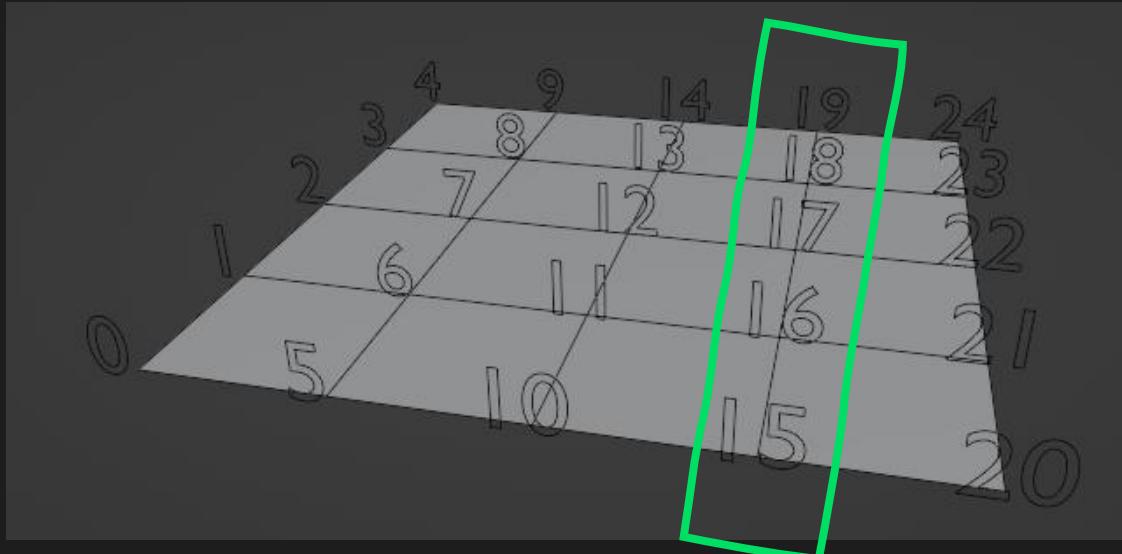
몫



5로 나눈 몫이 3 : 15, 16, 17, 18, 19



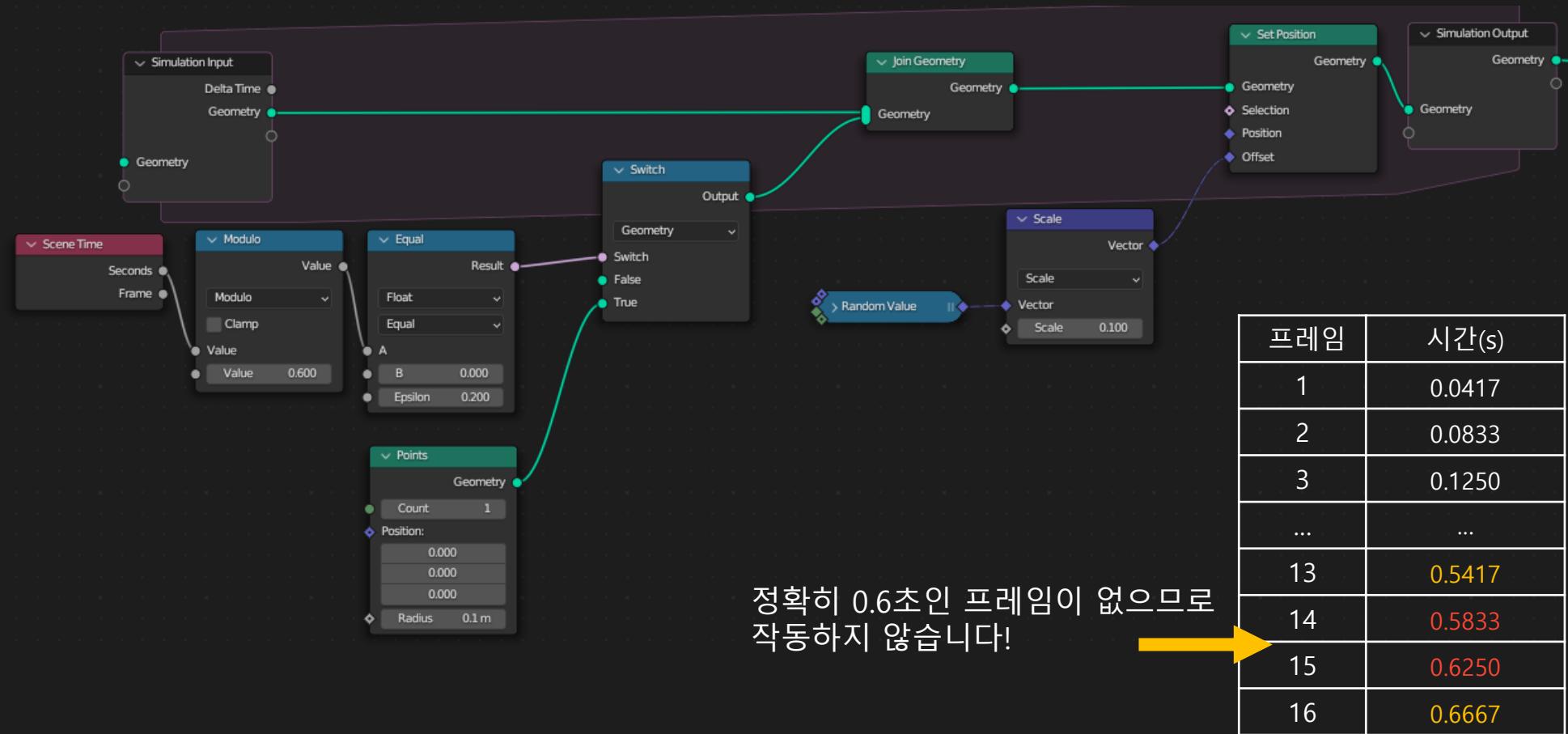
5로 나눈 나머지가 1 : 1, 6, 11, 16, 21



시뮬레이션 노드 연습문제

4. 랜덤 방향으로 움직이는 포인트를 0.6초마다 만드세요.

Modulo로 0.6마다 지오메트리를 스위치하면 될 것 같지만 정상적으로 작동하지 않습니다.(오답)

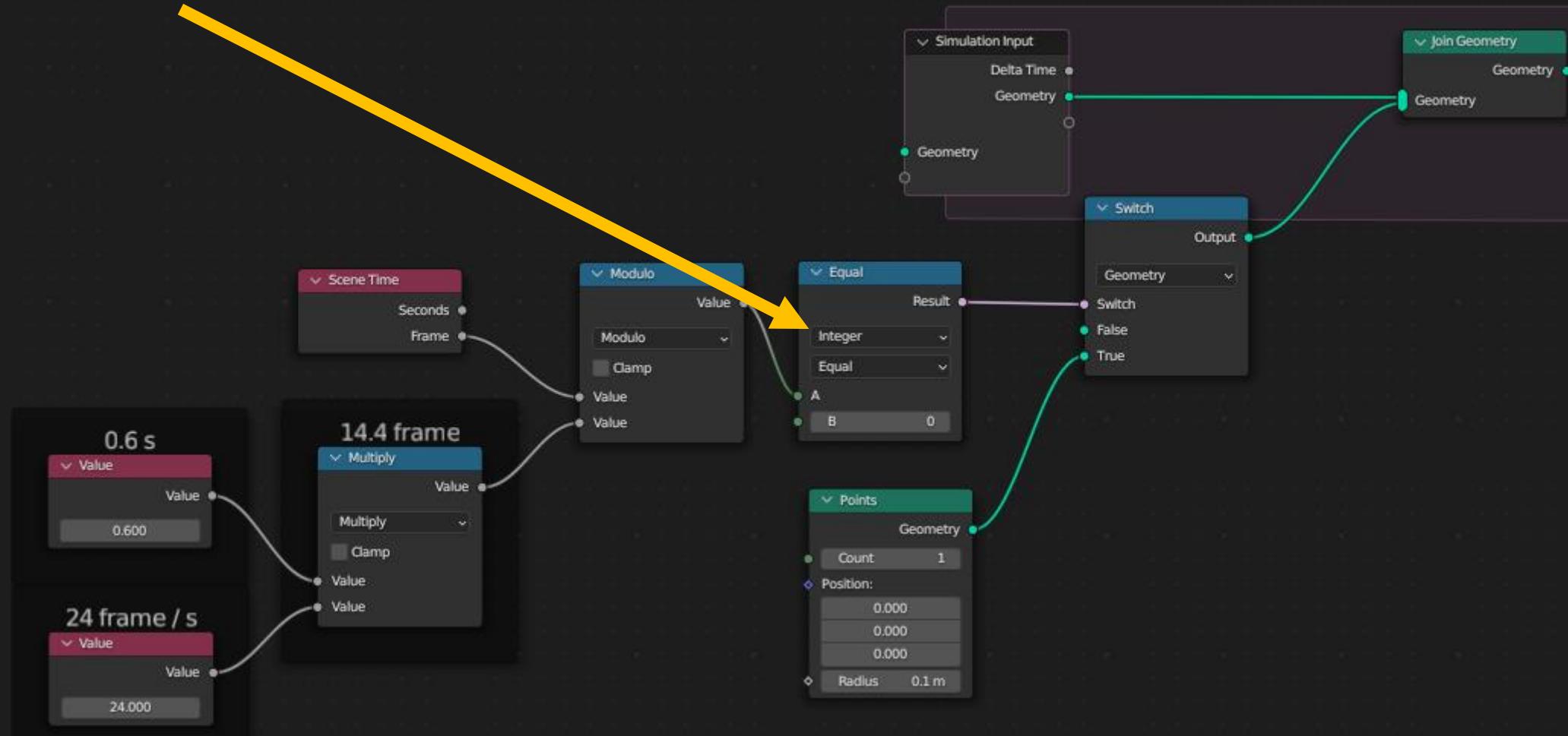


시뮬레이션 노드 연습문제

4. 랜덤 방향으로 움직이는 포인트를 0.6초마다 만드세요.

0.6초를 프레임으로 변환하여 계산합니다.

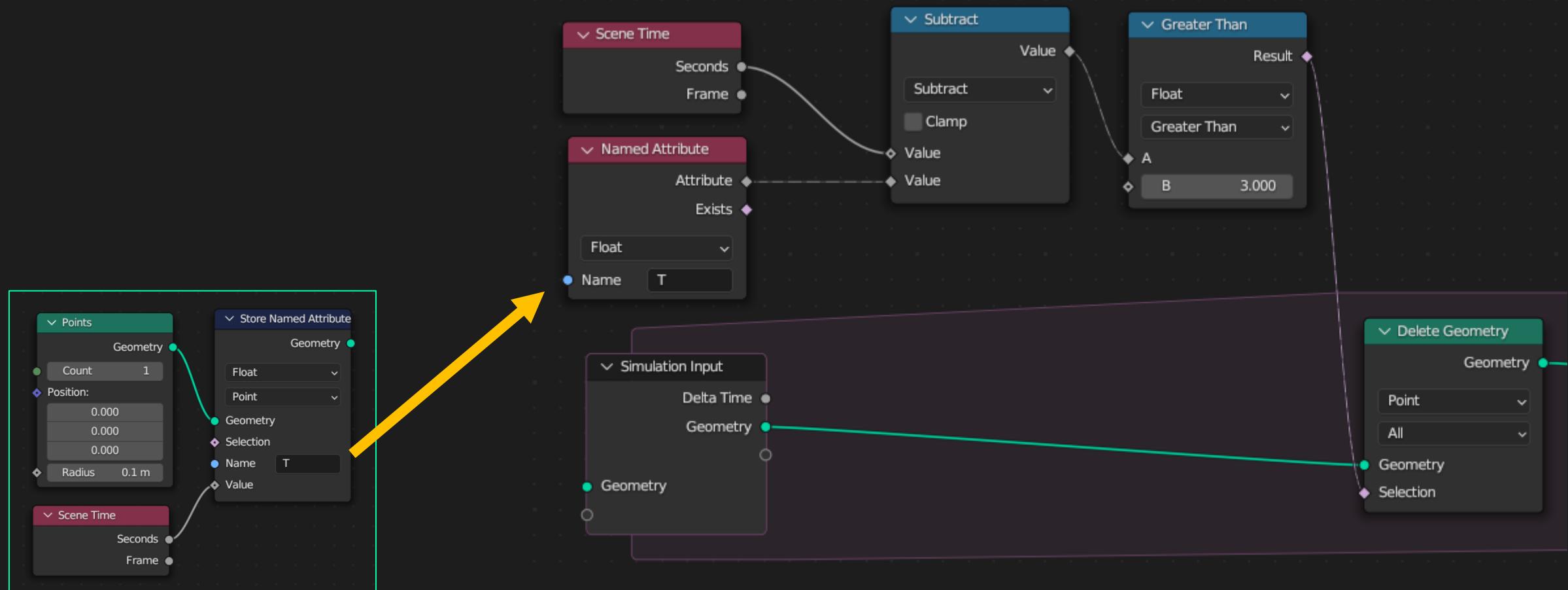
Compare 노드를 정수모드로 사용하면, 정확히 0이 아니어도 그 근처의 프레임이 선택됩니다.



시뮬레이션 노드 연습문제

5-A. 4에서, 모든 포인트는 3초가 지나면 사라지게 하세요.

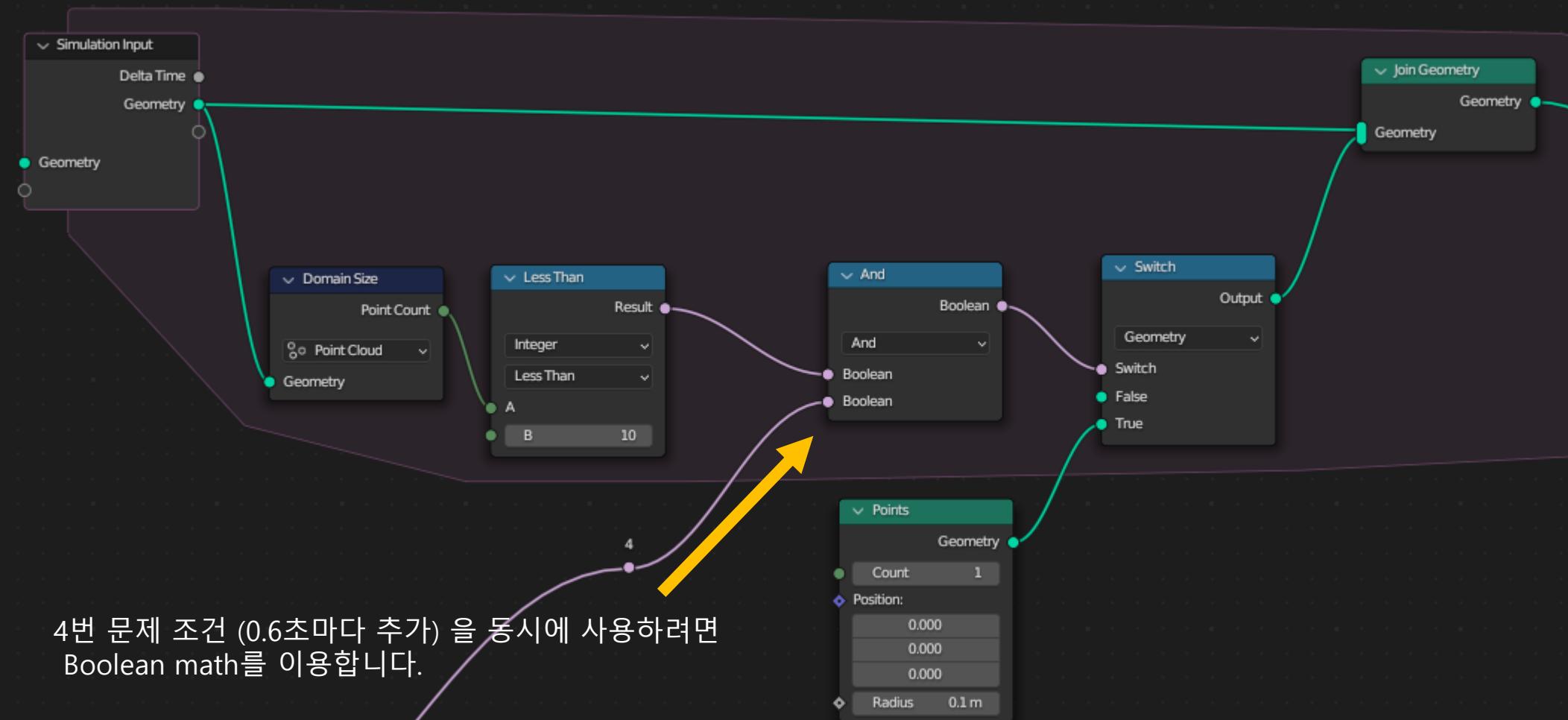
포인트를 조인할 때 Store Named Attribute로 시간을 저장해둔 뒤, 매 프레임마다 그 시간과 현재 시간을 비교합니다.
포인트 개수가 줄어들면 인덱스도 바뀌므로, 인덱스에 따라 랜덤을 만드는 Random Value 사용에 유의해야 합니다!



시뮬레이션 노드 연습문제

5-B. 4에서, 포인트 개수가 10개를 넘어가지 않게 하세요.

Domain Size를 이용하여 조인할 때마다 총 개수를 확인합니다.



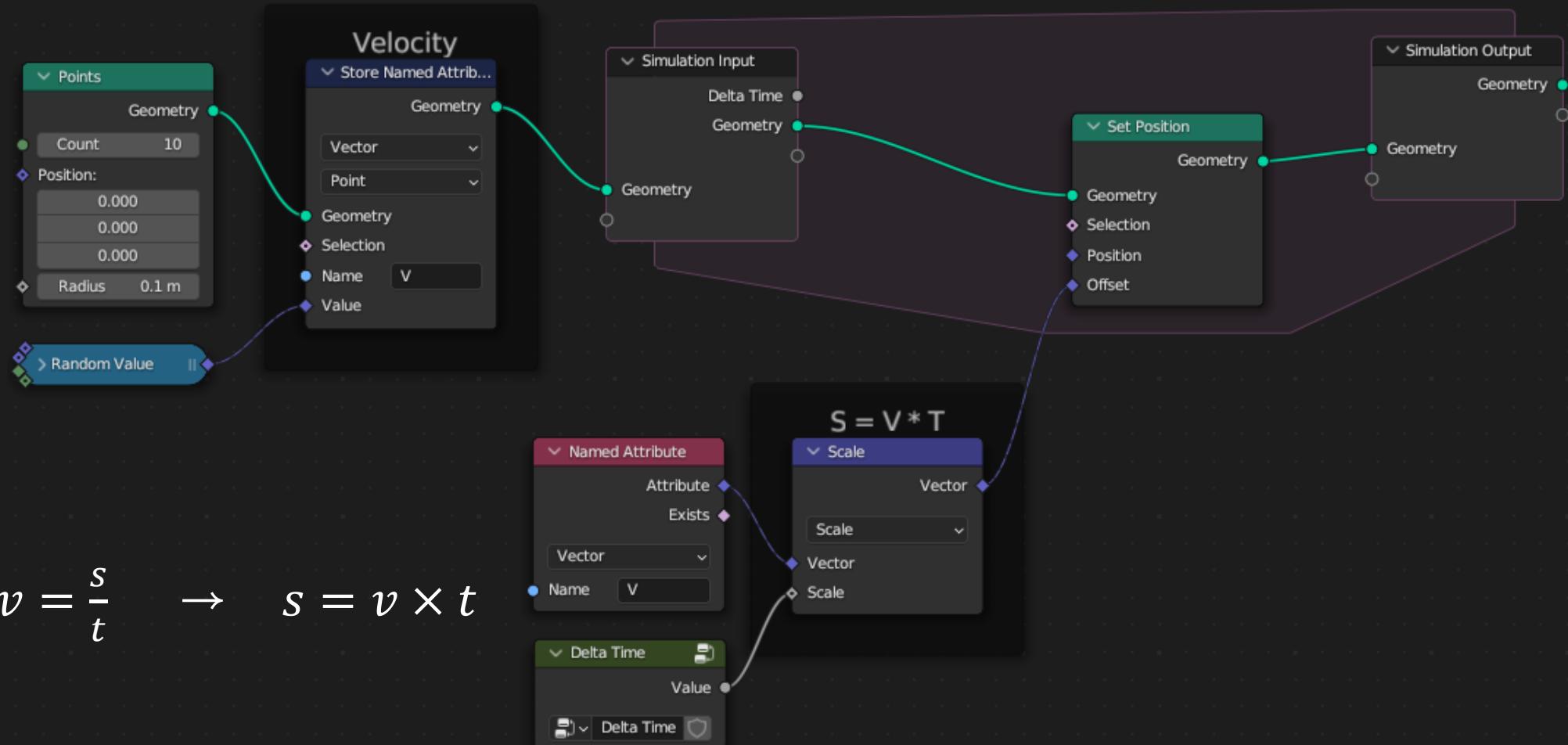
065강 Simulation Node – 속도와 가속도 (v3.6~)

기본적인 운동 표현



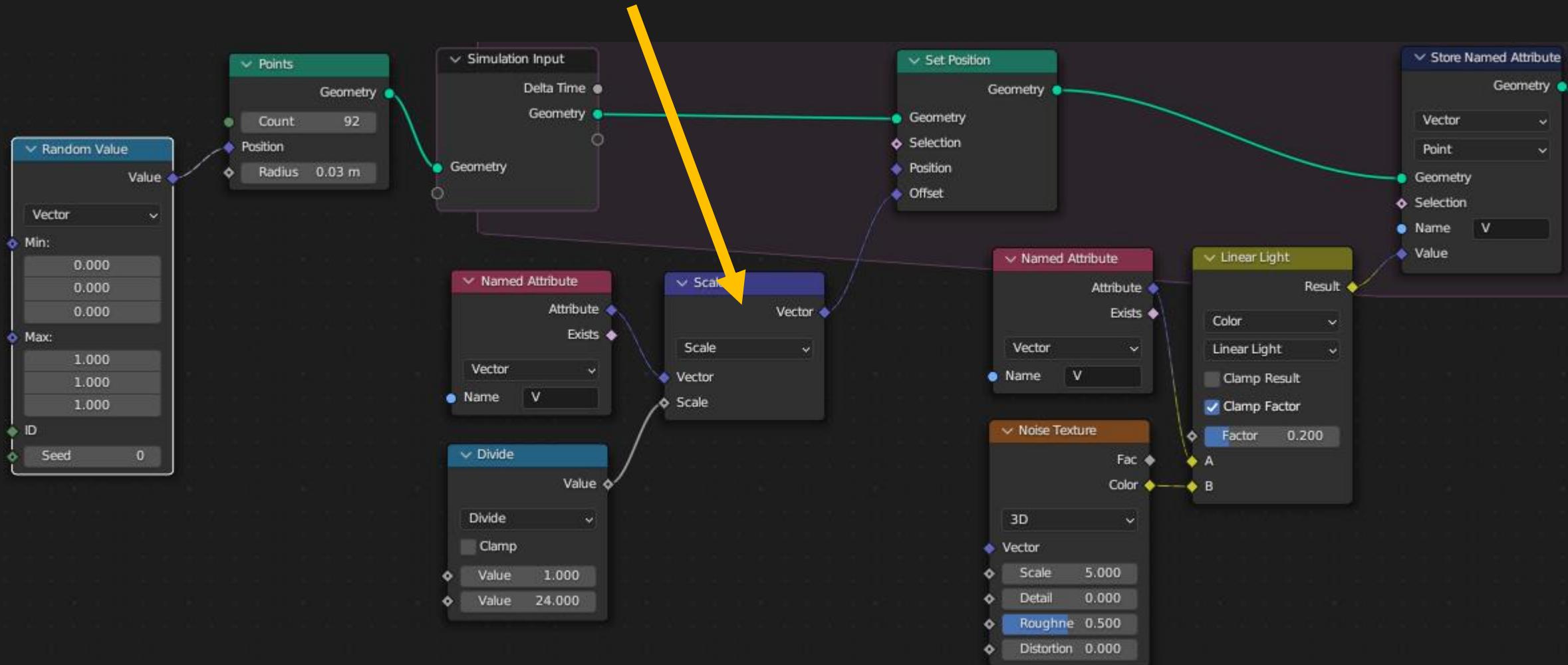
등속도 운동

지난 시간의 1번 문제가 등속도 운동입니다. 벡터 형식으로 바꾸면 아래와 같습니다.
(delta time은 1프레임 당 움직인 시간, 1/24초입니다.)



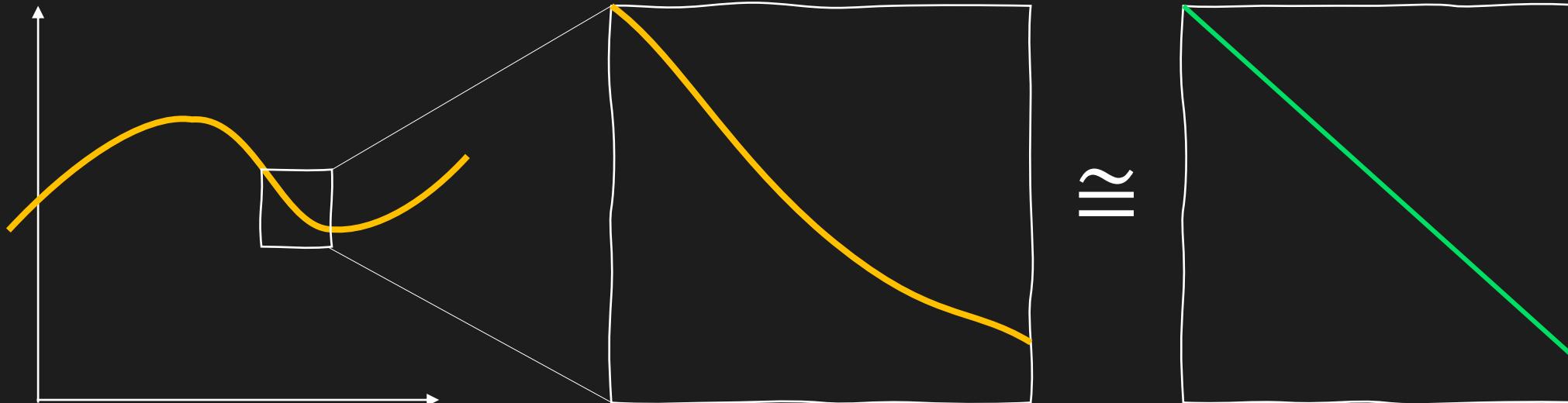
속도가 변한다면?

만약 시시각각 속도가 변할 때도, $s = vt$ 를 사용할 수 있을까요?



짧은 시간의 속도 변화

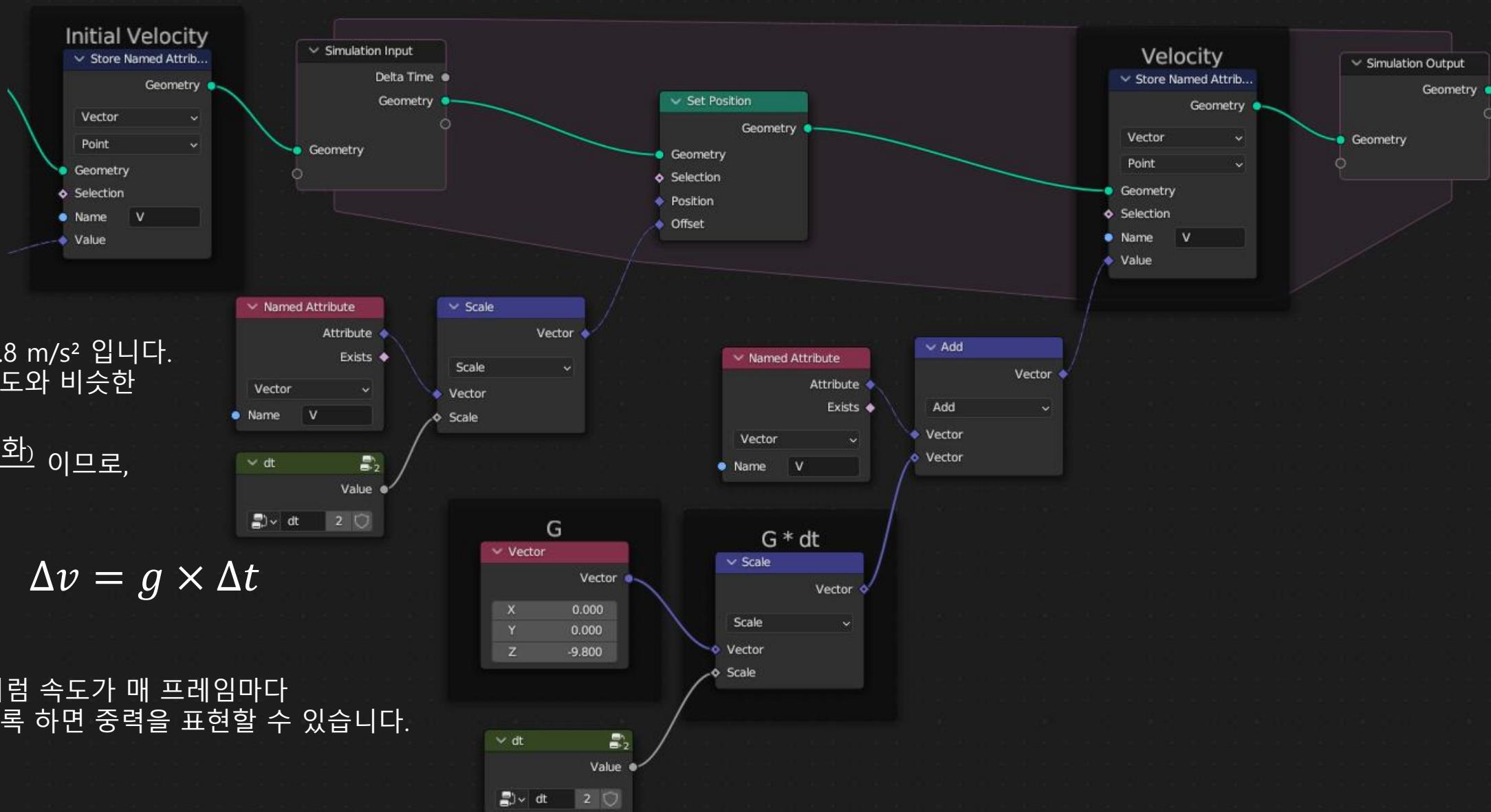
시뮬레이션 노드는 매 프레임마다 움직임을 계산합니다. 즉 0.042초 사이의 속도와 위치 변화를 탐지합니다.
이렇게 짧은 시간에서는 일정한 속도로 이동한다고 생각해도 계산에 큰 차이가 없습니다.
즉 속도가 변해도 노드 연결에서 $s = vt$ 를 사용해도 됩니다.



$$v = \frac{\Delta s}{\Delta t} \rightarrow \Delta s = v \times \Delta t$$

물론 완전히 정확한 계산은 아니며, 급격한 변화가 있는 경우 오차가 커질 수 있습니다.

중력 가속도



가속도 법칙

뉴턴의 제2법칙은 힘과 가속도의 관계를 나타냅니다.
힘 F 와 질량 m , 가속도 a 에 대하여

$$F = ma$$

이므로,

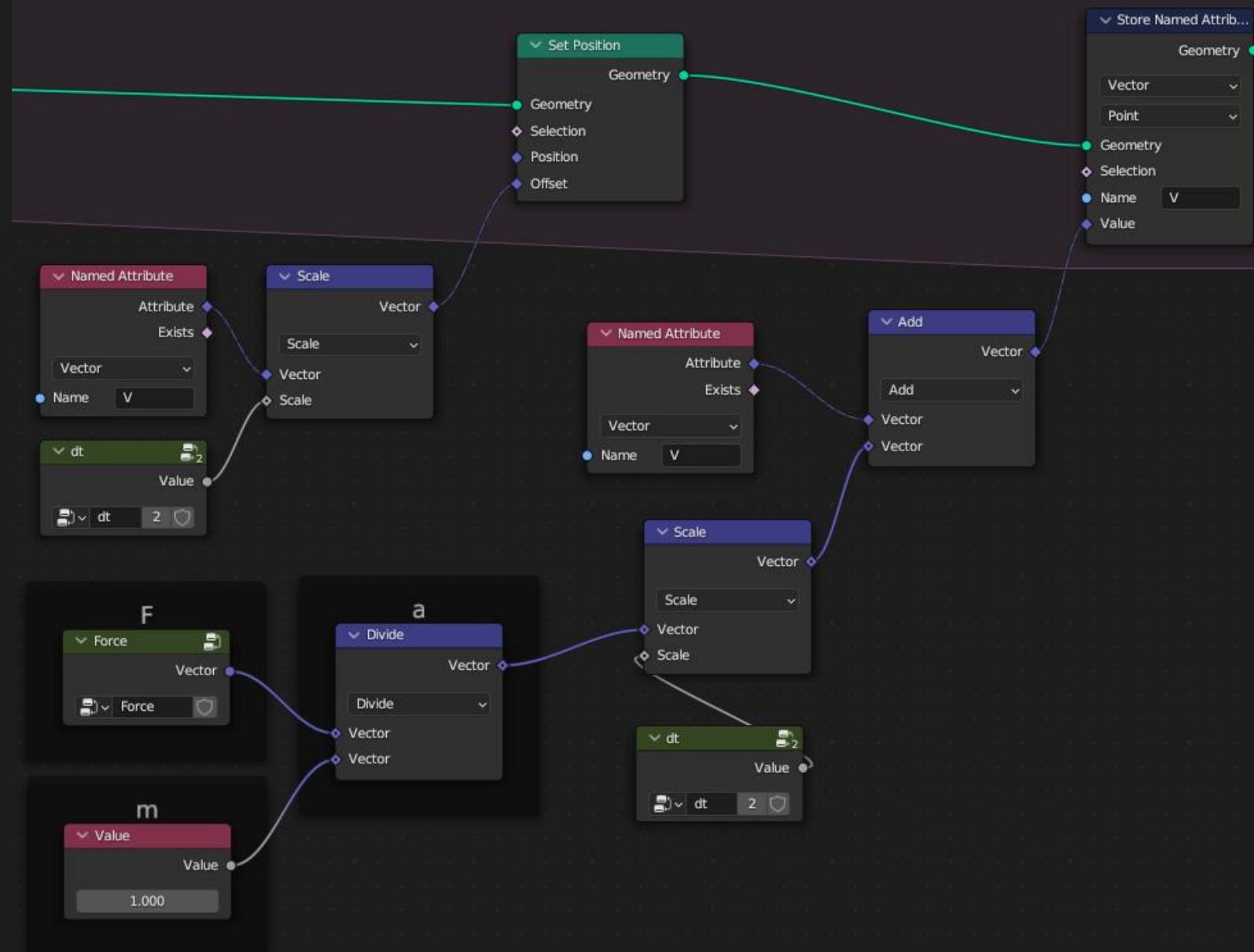
힘에서 질량을 나눈다면 가속도를 구할 수 있습니다.

즉 앞에서의 연결에서, 가속도 부분을 오른쪽처럼

약간 수정한다면 어떤 운동이라도 표현할 수 있는

연결이 만들어집니다. 여기서 힘 부분만 바꾸어

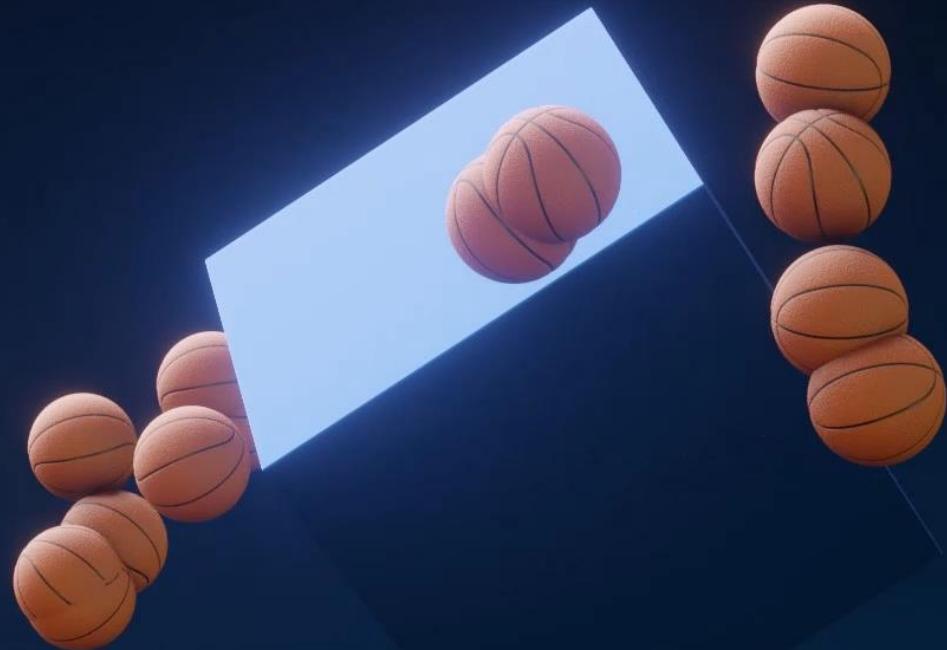
여러가지 움직임을 만들 수 있습니다.



066강 Simulation Node – 중력 (v3.6~)

충돌

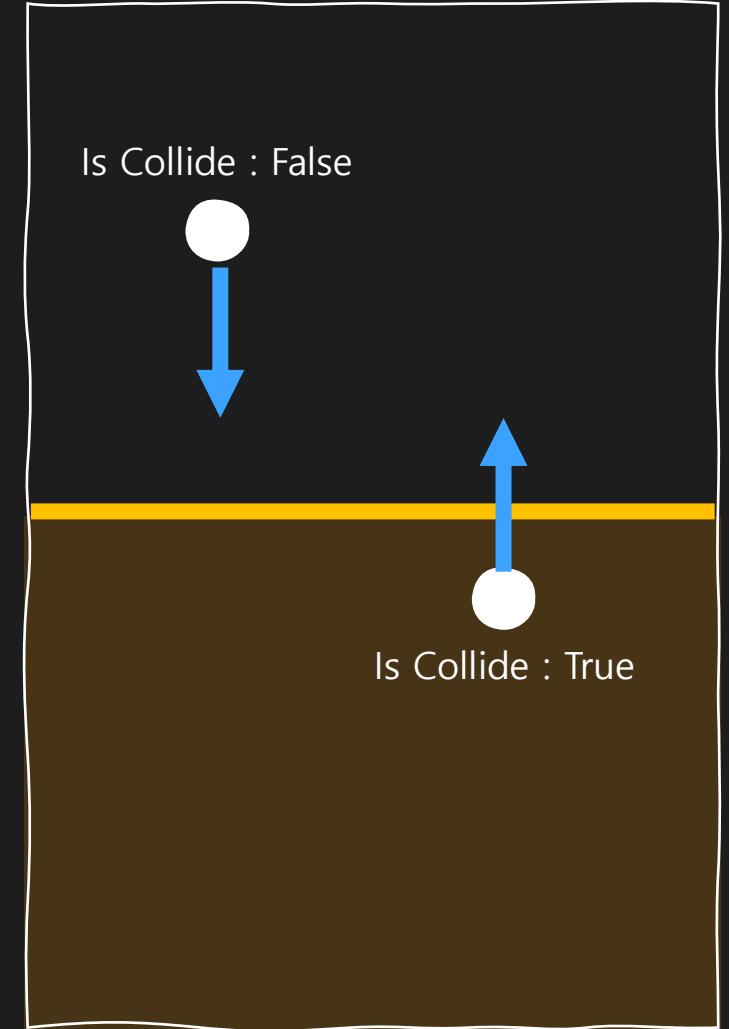
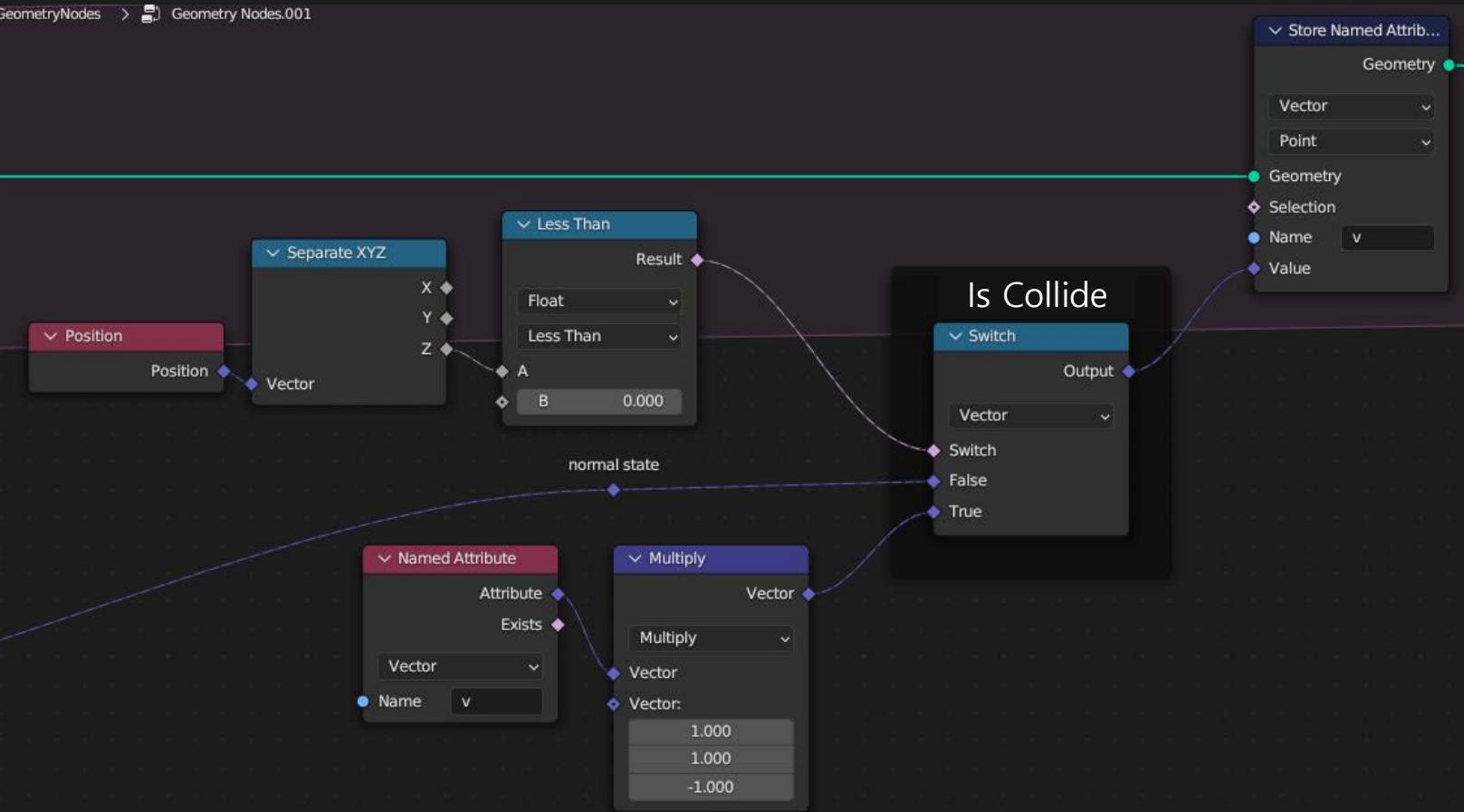
중력



충돌(1)

평평한 지면 ($z=0$)에 충돌하면 튕어오르게 하기 위해서,

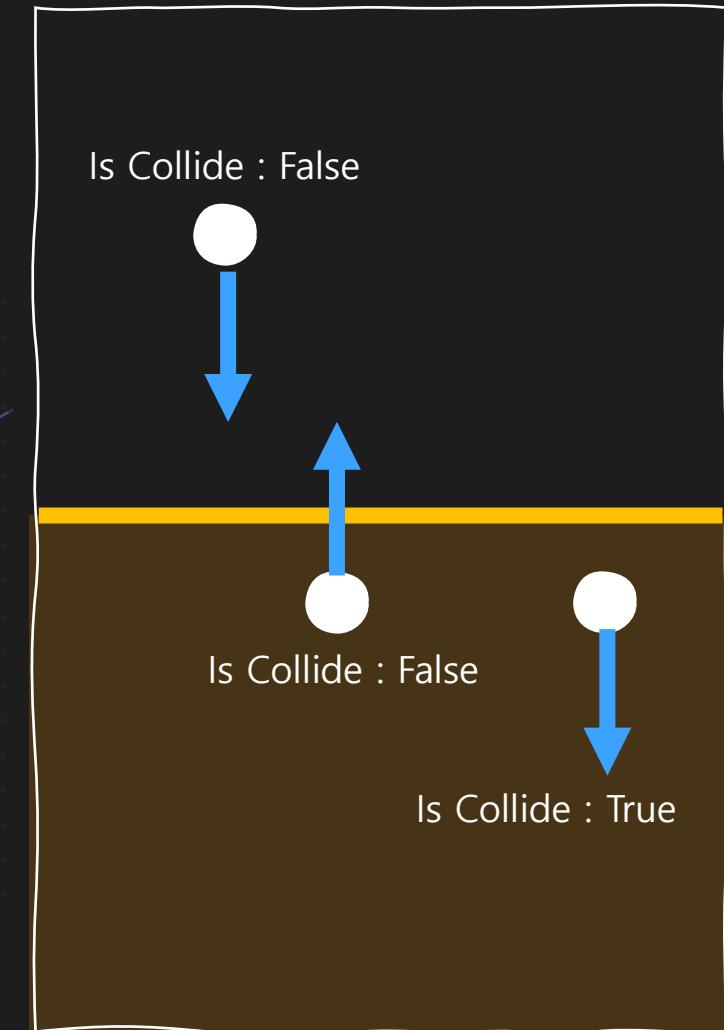
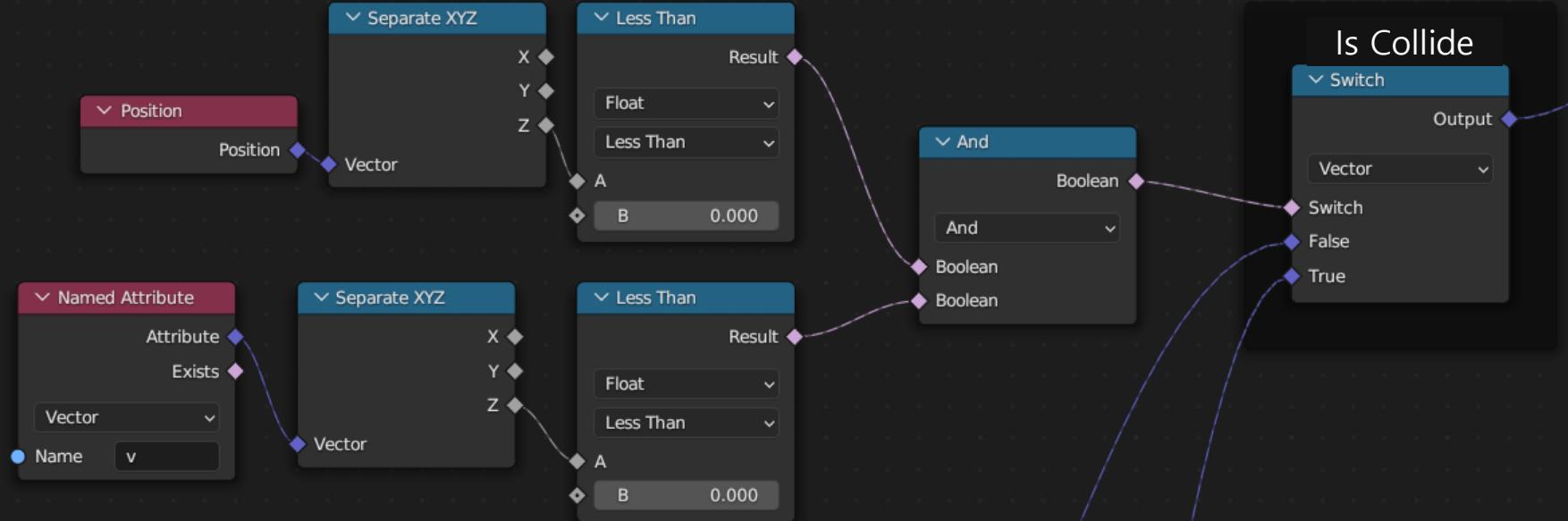
조건 : 위치가 $z=0$ 보다 아래 있으면
행동 : 속도의 z 축 방향을 뒤집는다
고 해 봅시다.



충돌(2)

그러나 앞처럼 연결하면, 이미 충돌 판정이 나서 위쪽으로 방향이 바뀐 경우에도
다시 충돌 판정이 날 수 있습니다.

조건을 : 위치가 $z=0$ 보다 아래인데, 속도 방향도 아래일 때
로 바꿉니다.



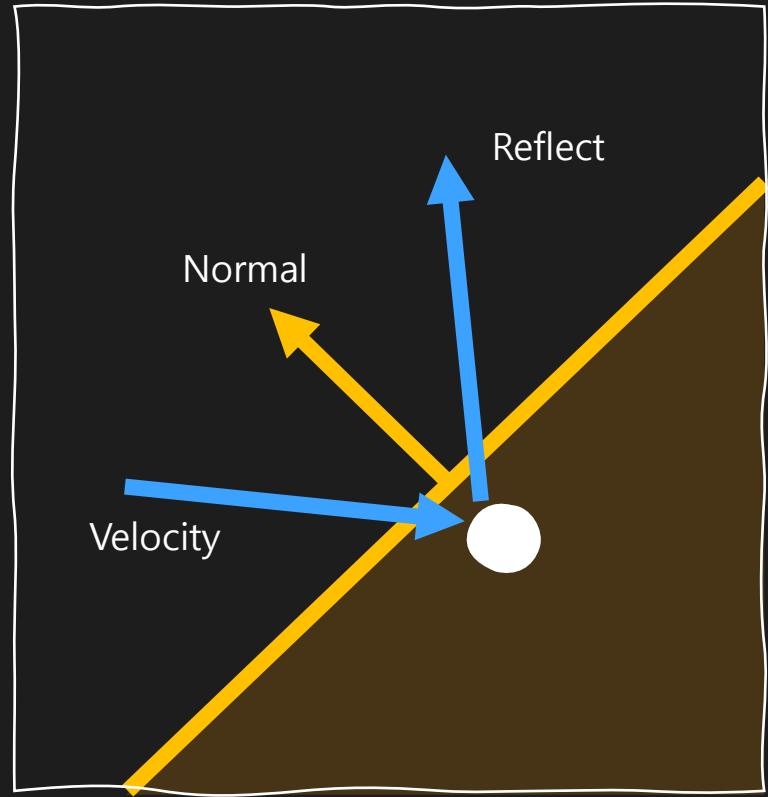
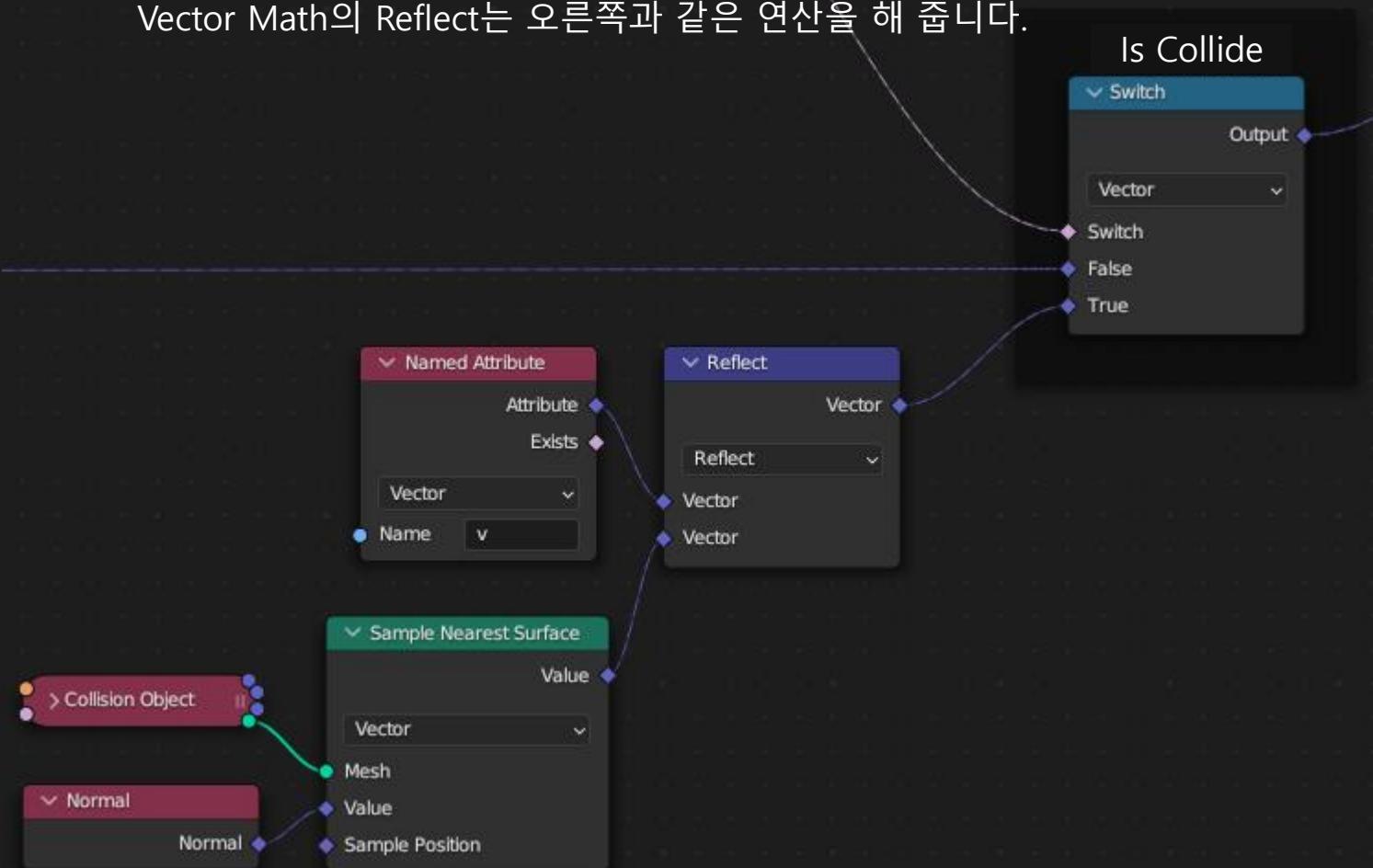
그 외에도 여러 방법이 있을 수 있습니다.

예를 들어 '복구중' 변수를 추가하여 충돌 판정이 나면 z 축 위로 올라올 때까지 켜두어서,
'복구중' 이 켜 있으면 충돌 판정을 계산하지 않는다거나 할 수 있습니다만, 노드가 많이 길어집니다.

표면의 충돌 (1)

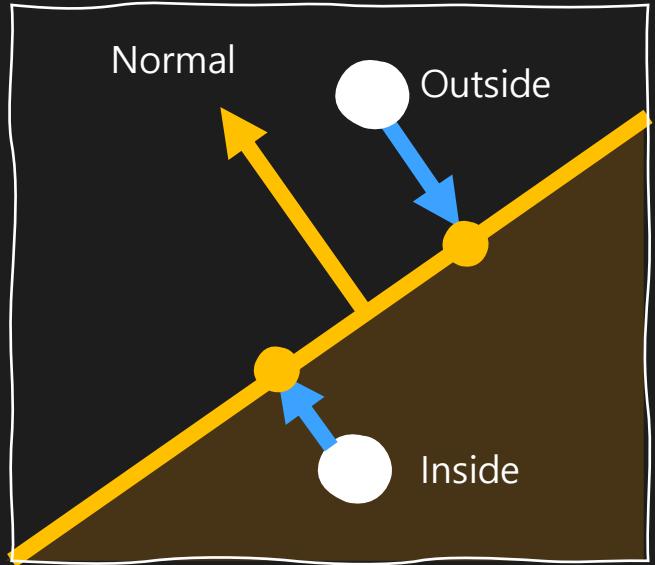
임의의 표면에 충돌하면, 표면의 Normal 방향과 Vector Math를 이용하여 속도를 계산할 수 있습니다.

Vector Math의 Reflect는 오른쪽과 같은 연산을 해 줍니다.

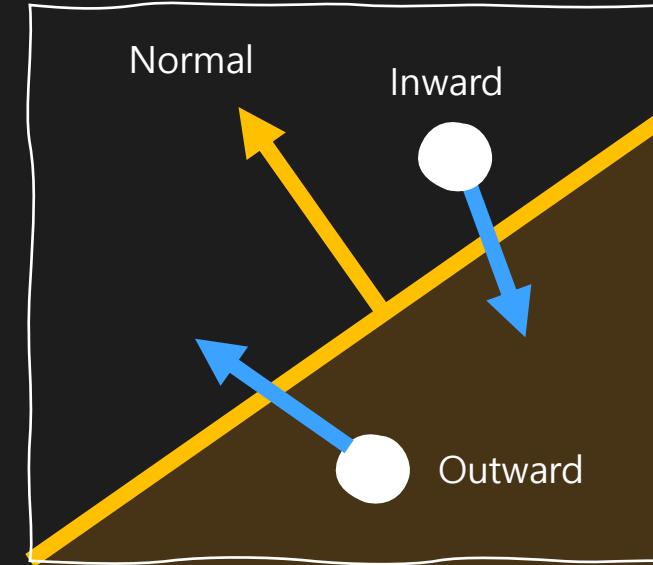


표면의 충돌 (2)

충돌 판정은 물체의 '내부에 있는지 외부에 있는지'
그리고 내부를 향하는지 외부를 향하는지로 판정합니다.



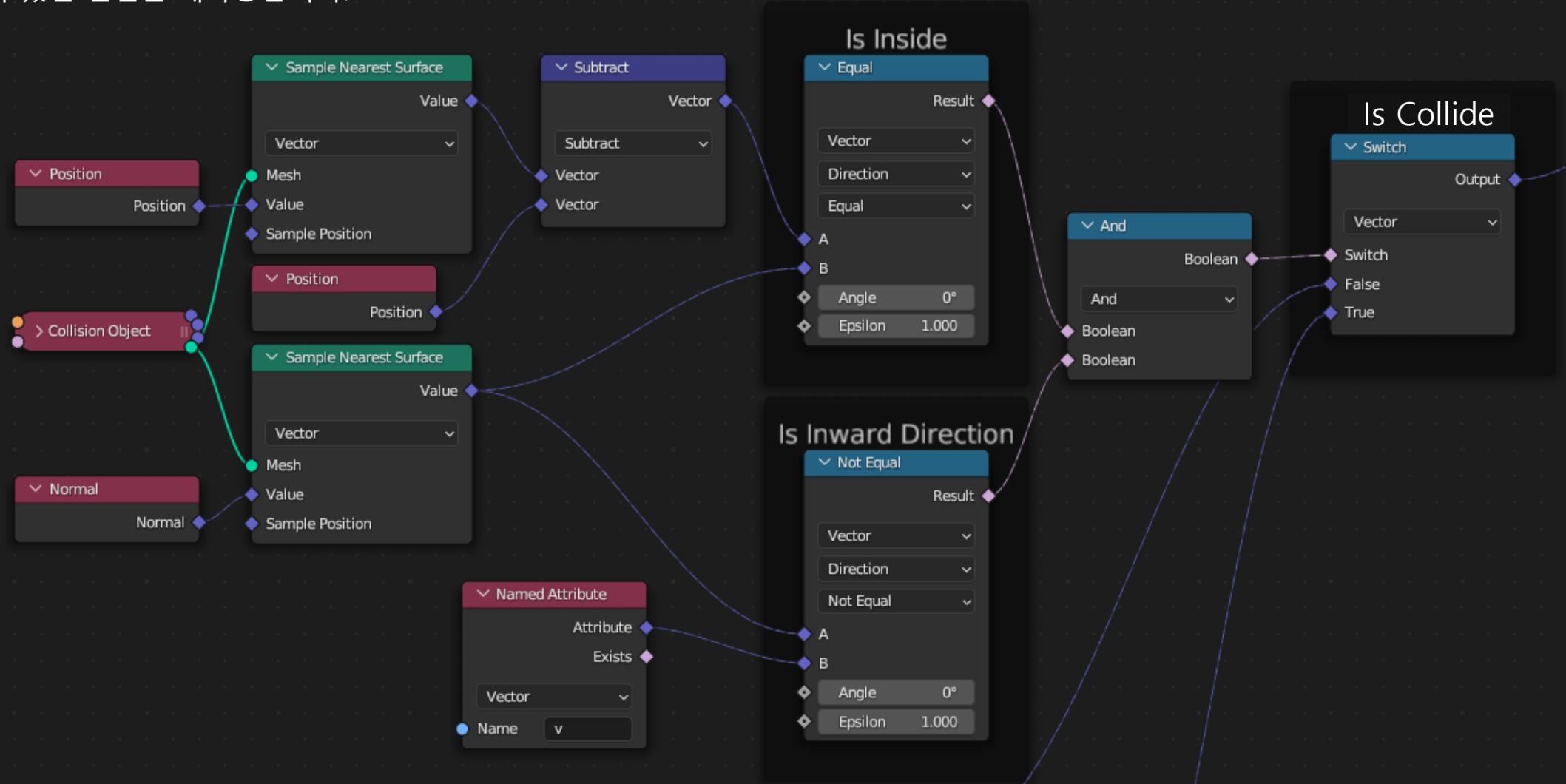
가장 가까운 표면으로의 방향이
Normal과 같다면 내부, 반대라면 외부입니다.



속도의 방향이 Normal과 같다면 바깥쪽 방향,
다르다면 안쪽 방향입니다.

표면의 충돌 (3)

52강에서 했던 연결을 재사용합니다.



중력

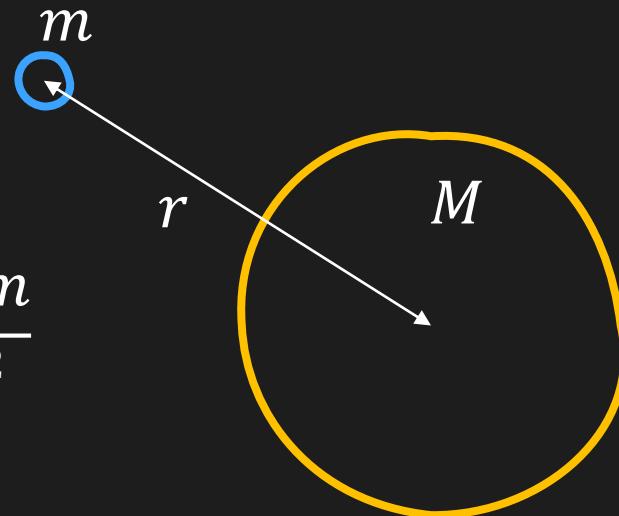
중력은 질량이 있는 물체끼리 서로 끌어당기는 힘입니다. 공식은 오른쪽과 같습니다.

중력은 두 물체의 거리 제곱에 반비례하며, 질량에 비례합니다.

한쪽 물체가 충분히 크고 무겁다면,

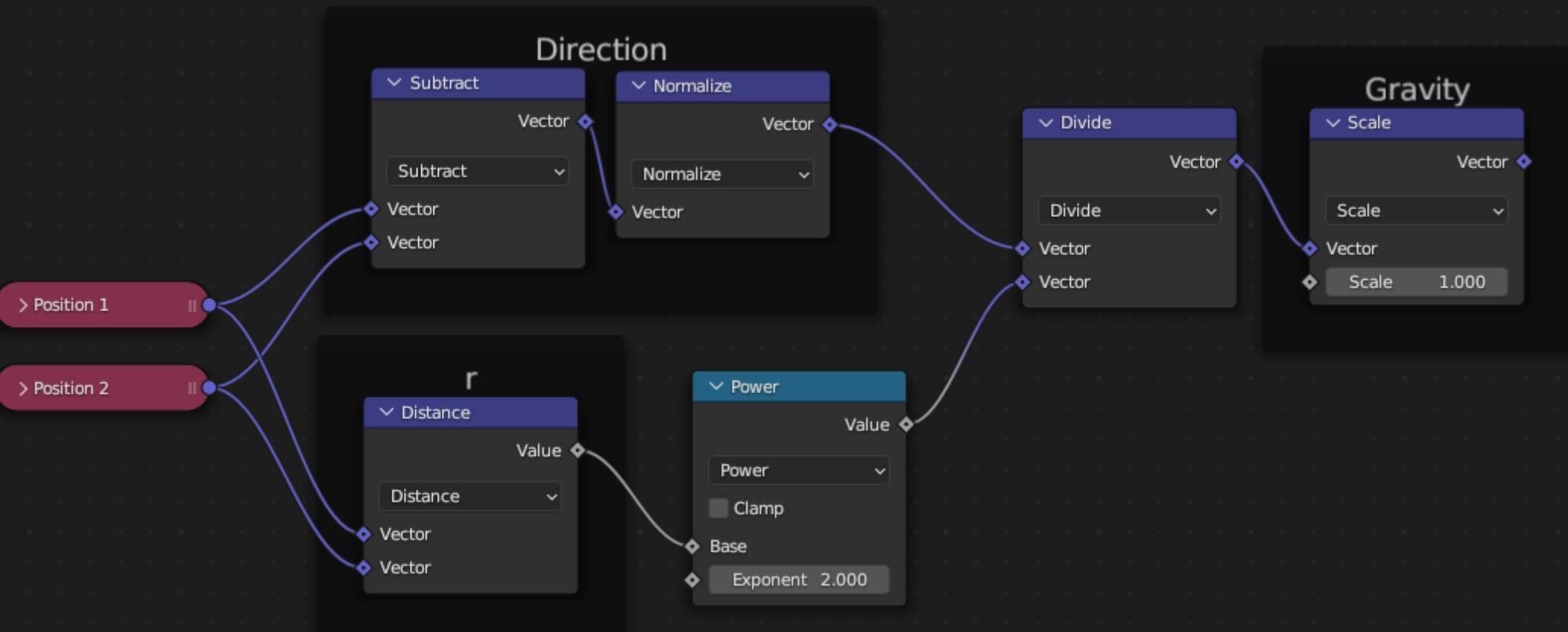
큰 쪽은 고정시켜 놓고 작은 쪽의 움직임만 계산할 수 있습니다.

$$F = G \frac{Mm}{r^2}$$



중력

노드로 구현하면 이렇게 됩니다. 거리 제곱으로 나눈 뒤, 변하지 않는 상수들 (질량, 중력상수 등..) 을 곱하면 됩니다.
65강의 노드 트리에 연결하여 사용할 수 있습니다.



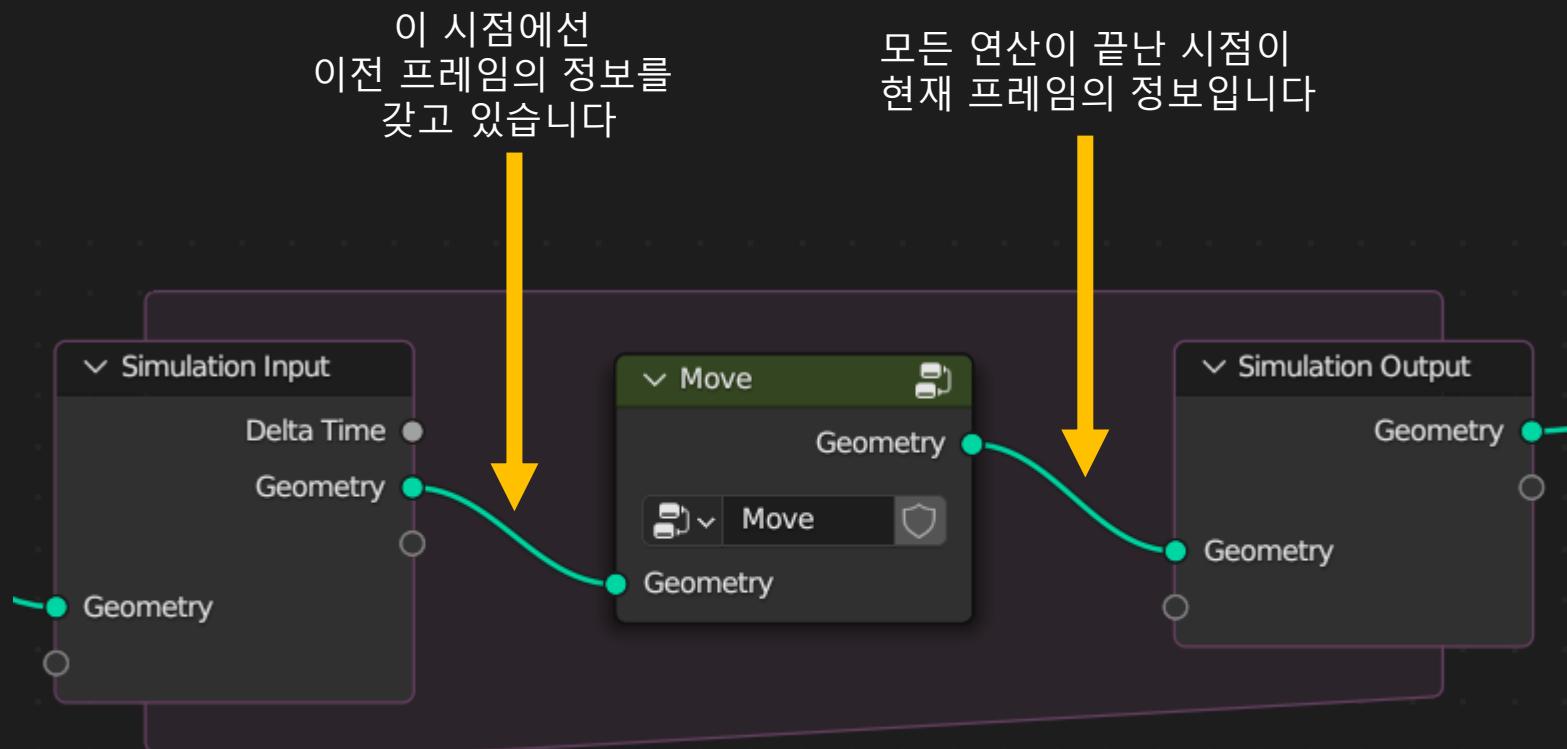
067강 Simulation Node – 잔상 (v3.6~)

이전 프레임의 정보를 가져오는 법



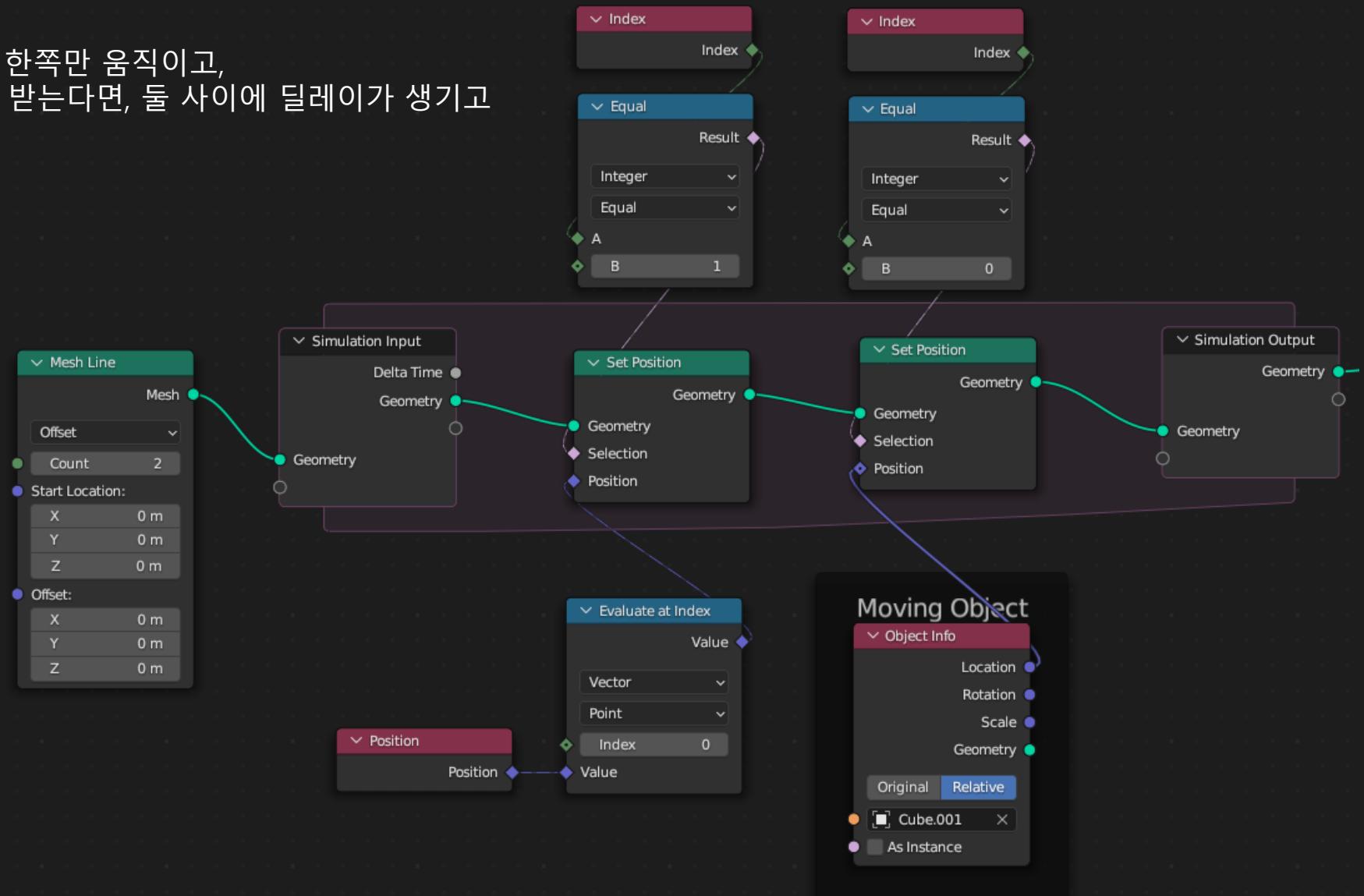
이전 프레임의 정보

블렌더에서는 이전 프레임의 정보를 현재 프레임에서 얻는 것이 힘듭니다.
지오메트리 노드도 마찬가지지만, 시뮬레이션 노드는 다릅니다.



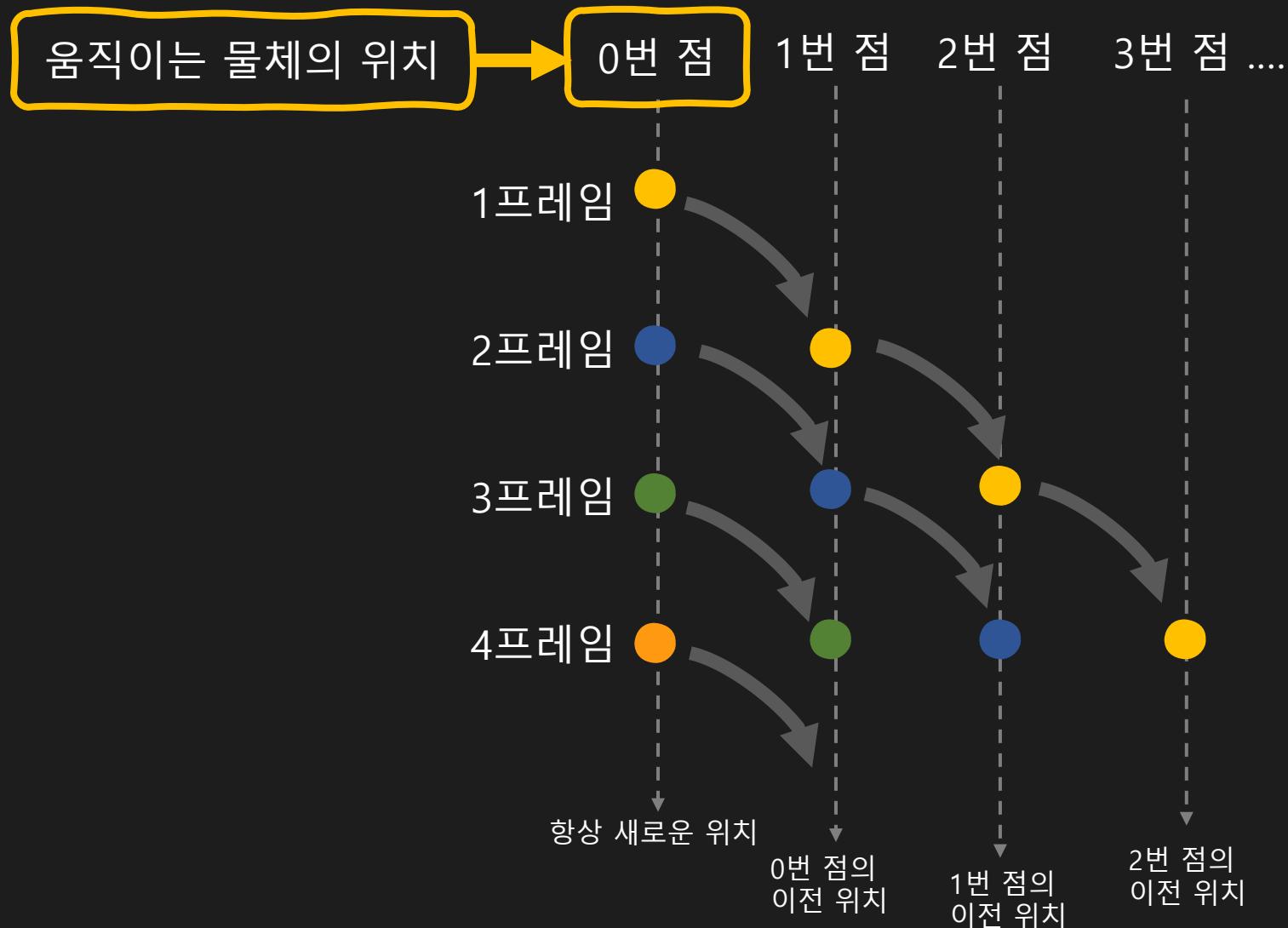
잔상(1)

움직이는 오브젝트의 위치를 받아, 한쪽만 움직이고,
다른 한쪽은 움직이기 전의 위치를 받는다면, 둘 사이에 딜레이가 생기고
잔상이 됩니다.



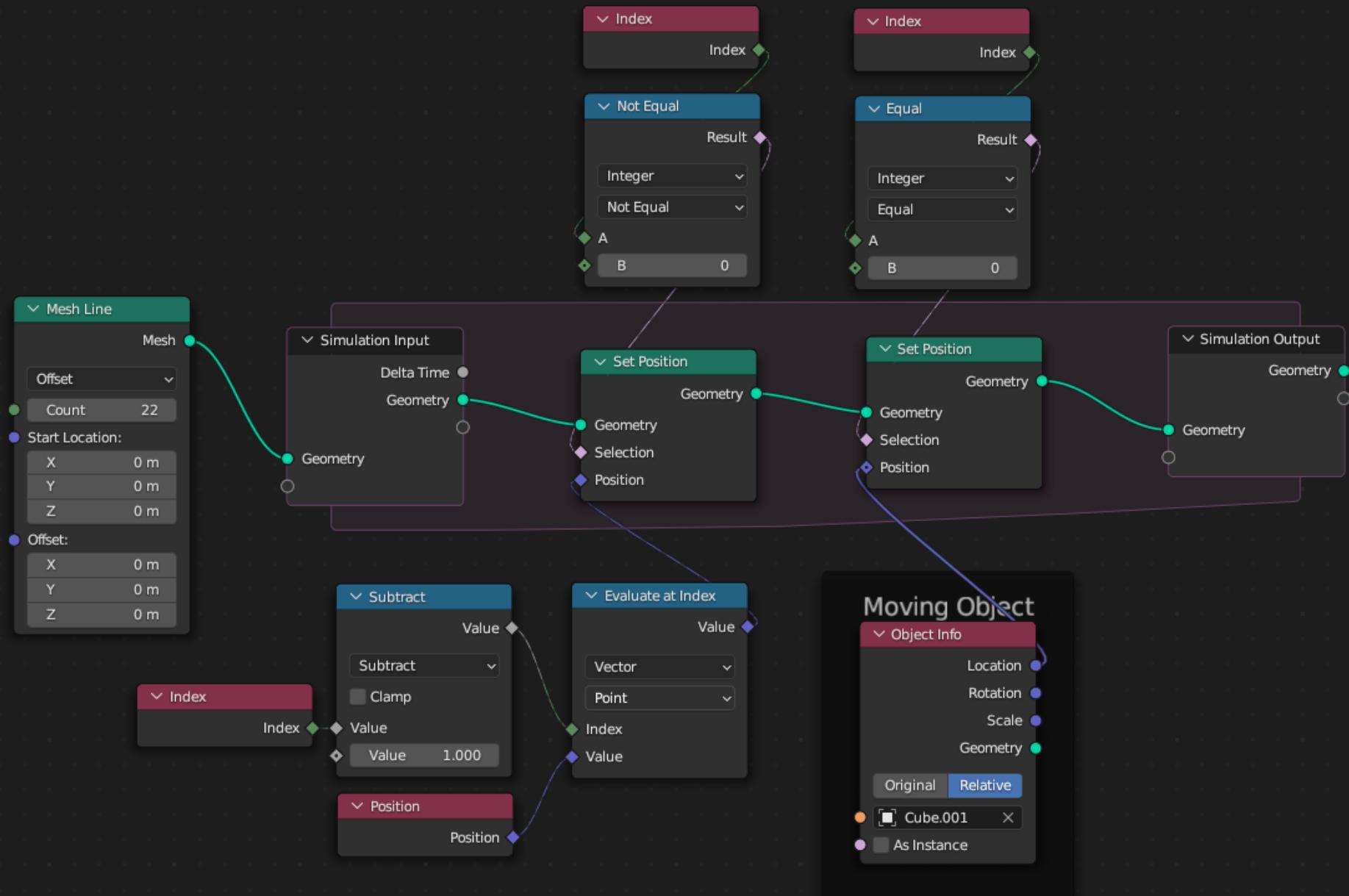
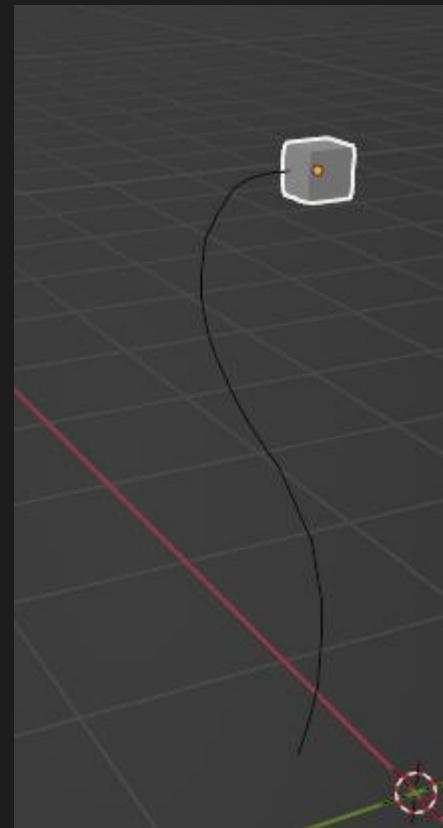
잔상(2)

같은 원리로, 맨 처음 포인트에만 정보를 전달한 뒤,
나머지는 자신보다 **한칸 앞번호의 위치**를 받아오면,
길게 꼬리를 물고 움직이게 됩니다.



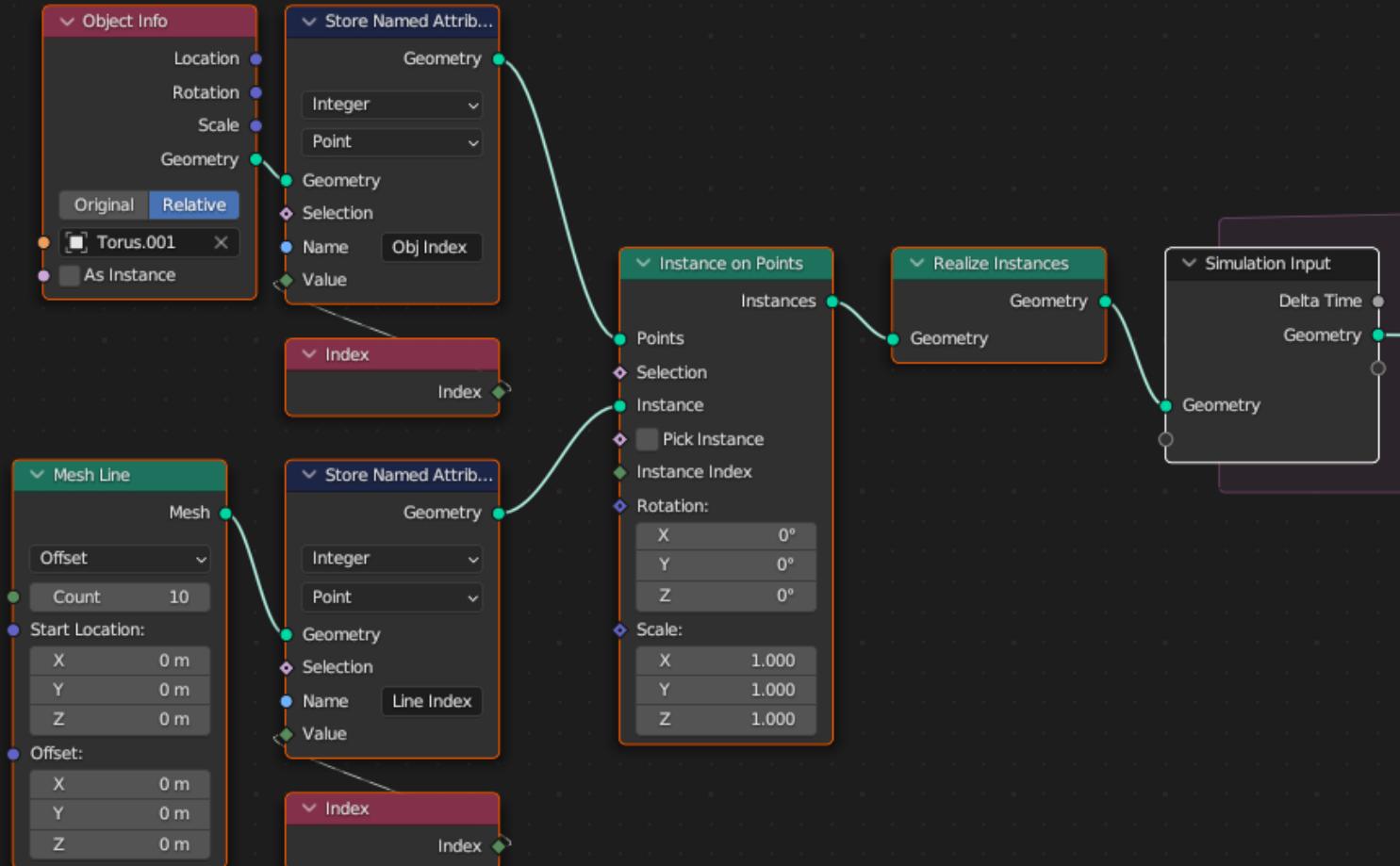
잔상(3)

실제 연결은 다음과 같습니다.



여러 점의 잔상

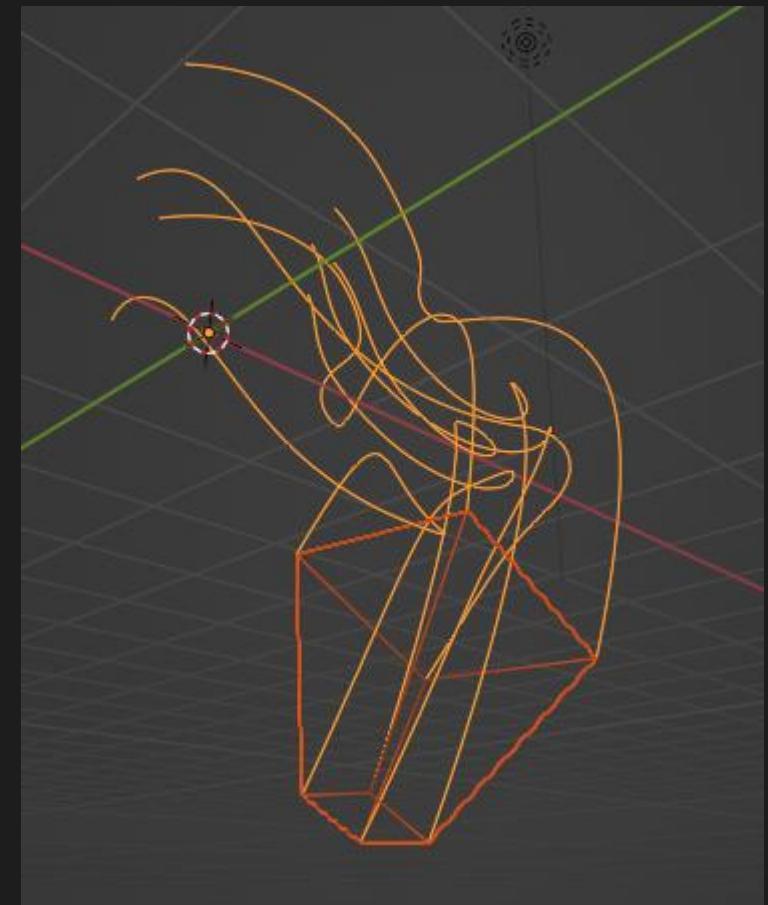
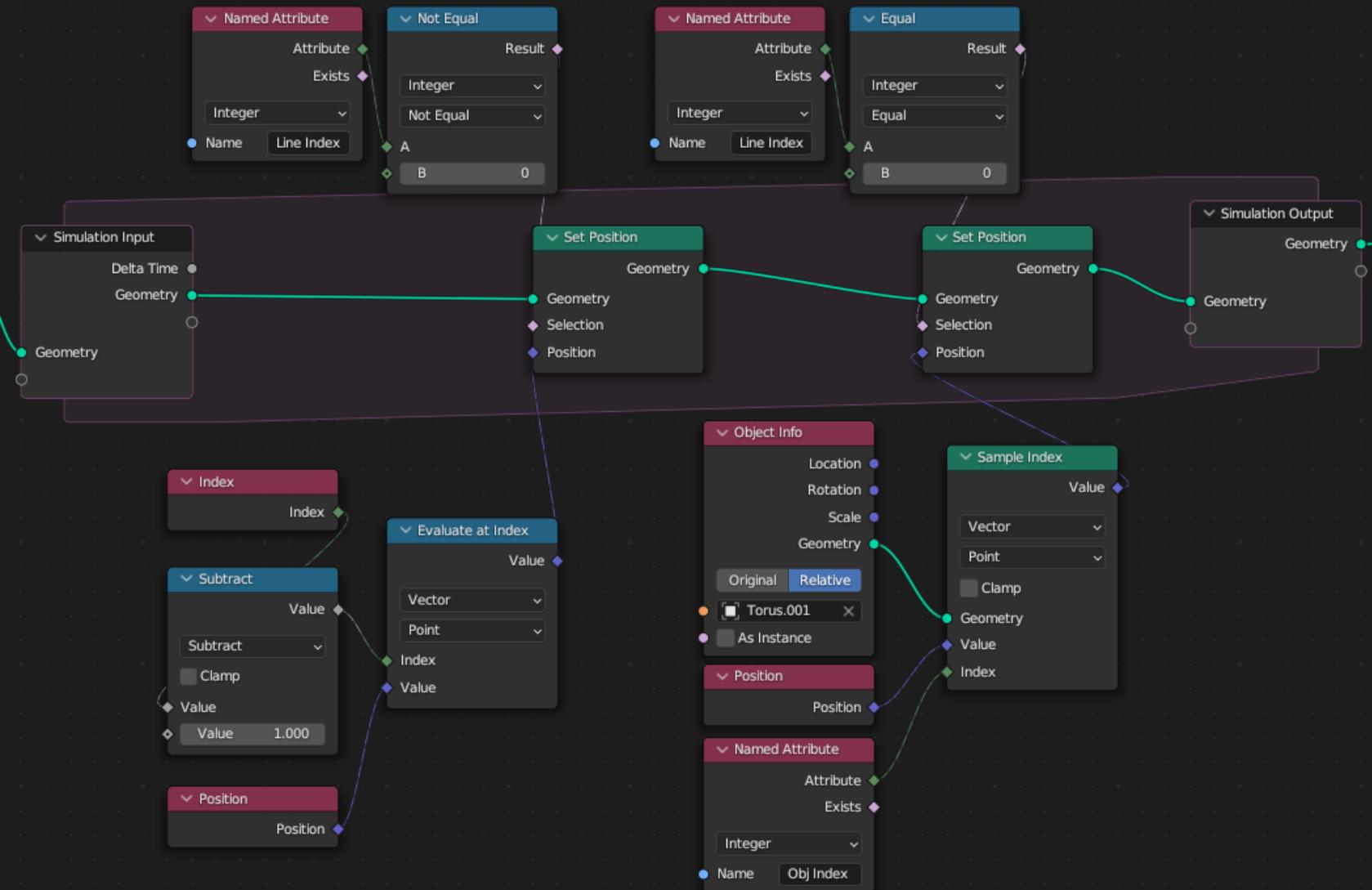
여러 개의 위치를 추적하고 싶으면 선을 여러 개 만들어야 합니다.



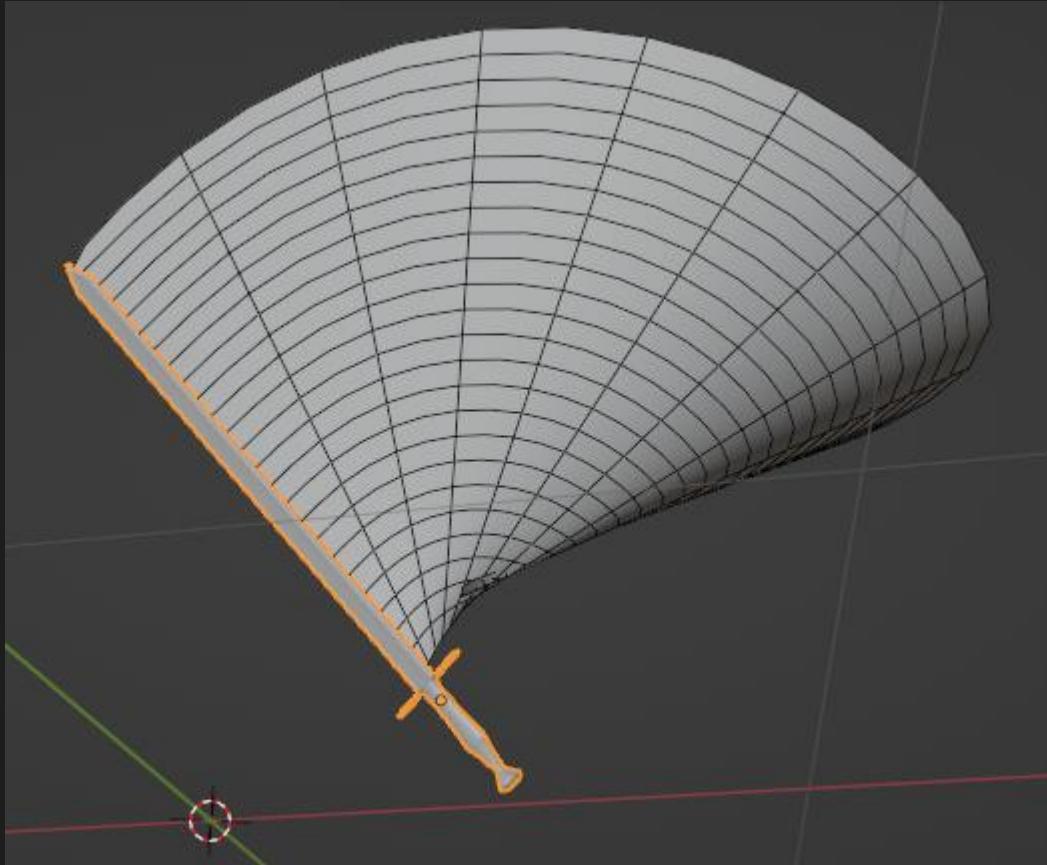
각각의 위치에 선을 심고, 인덱스를 저장합니다.
이 인덱스는 Realize 뒤에도 유지됩니다.

여러 점의 잔상(2)

원래 선의 인덱스를 바탕으로 움직임을 나눠 계산합니다.



선의 잔상(1)

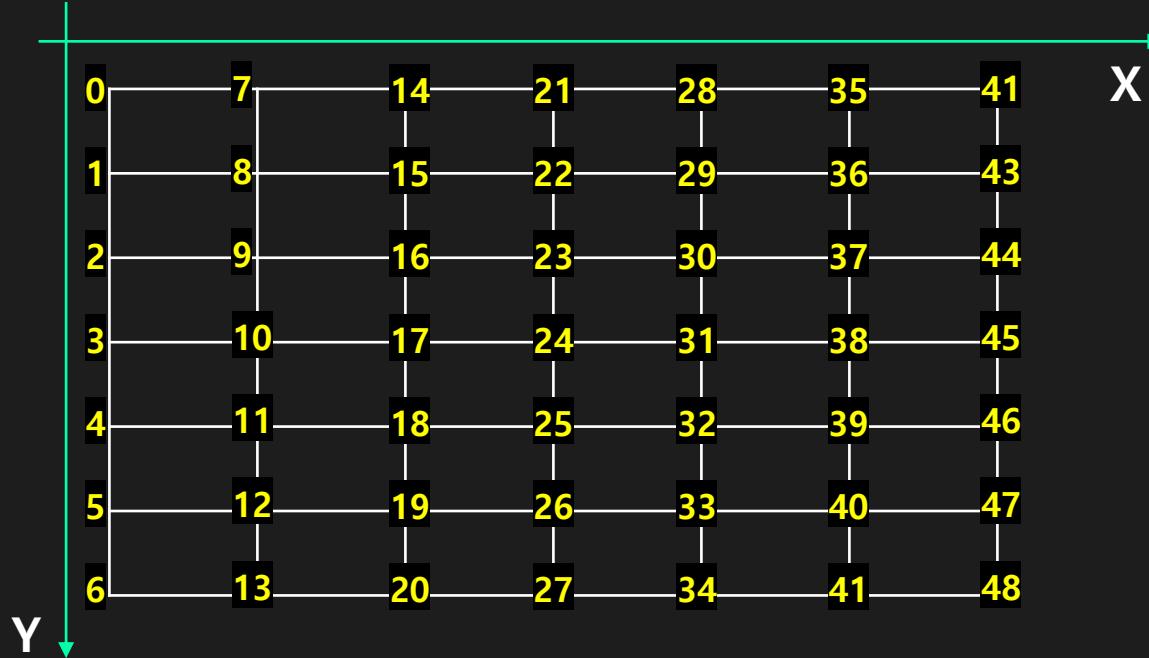
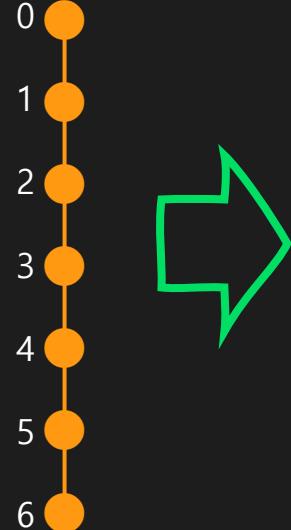


점의 잔상은 선이므로, 선의 잔상은 면이 됩니다.

기본 원리는 같지만, 2차원이 되면 인덱스를 대응시키는 것이 조금 어려워집니다.

선의 잔상(2)

선을 그리드에 대응시킵니다.



그리드의 7보다 번호가 작은 점들(0~6번)은 동일 번호의 위치를 그대로 받습니다.

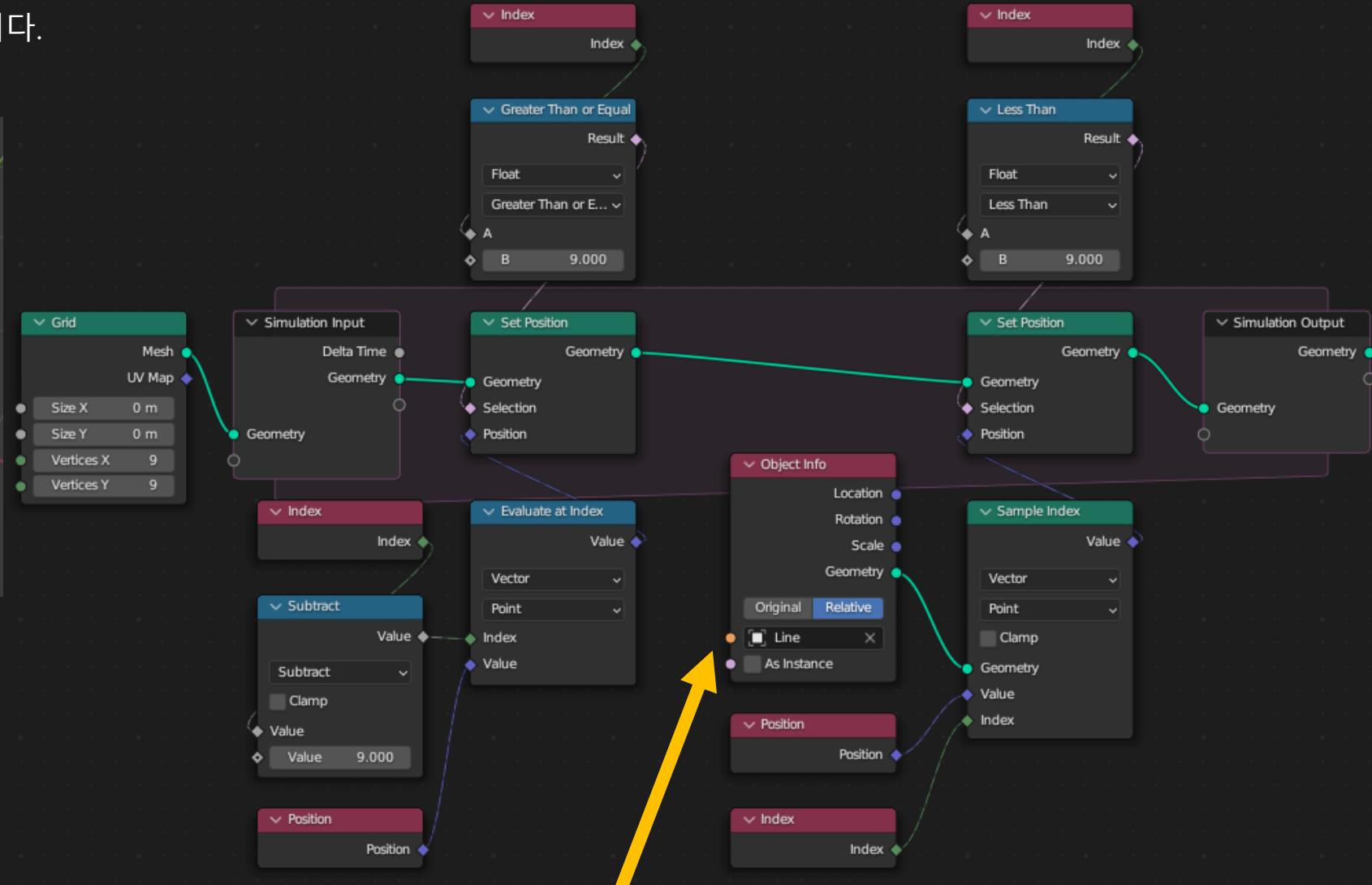
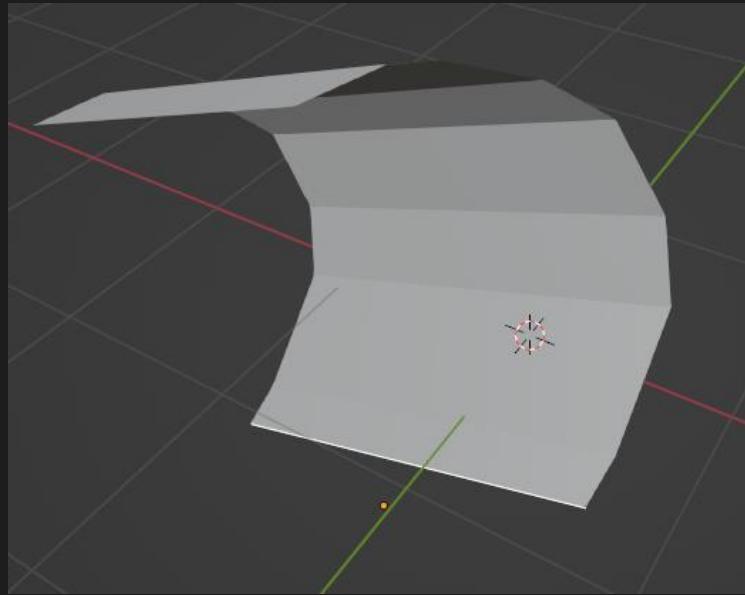
나머지 점들은 자기보다 7 작은 번호의 위치를 이어받습니다.

1에 대응되는 점 : 1,8,15,22,29,... : 7로 나누어 나머지가 1

딜레이가 2 프레임인 점 : 14,15,16,17....20 : 7로 나누어 몫이 2

선의 잔상(3)

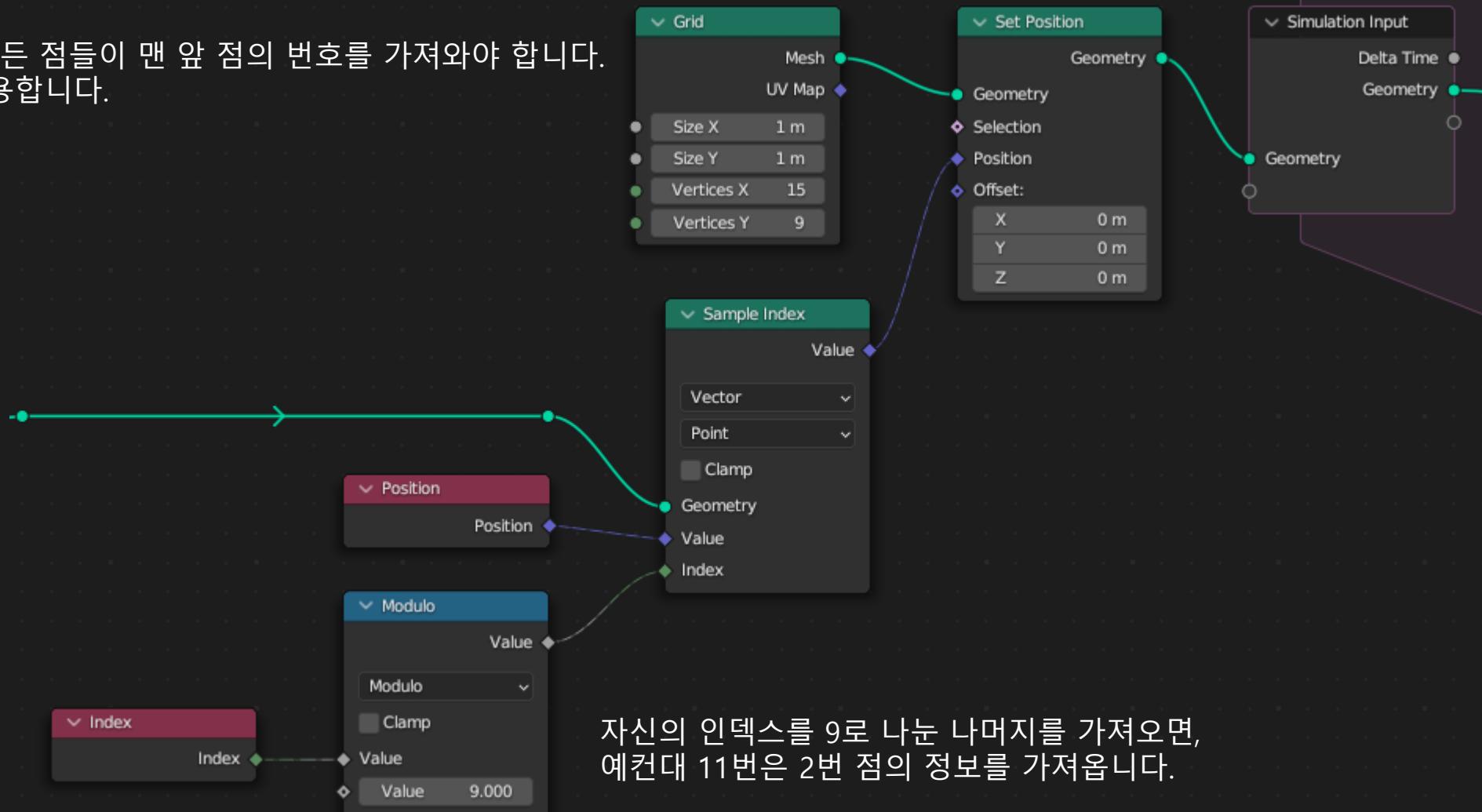
다음은 점 개수가 9개일 때의 잔상입니다.



Line은 인덱스가 순서대로여야 합니다.

선의 잔상(4)

초기 위치를 설정하려면 모든 점들이 맨 앞 점의 번호를 가져와야 합니다.
이 때는 modulo연산이 유용합니다.



068강 Simulation Node – 그래디언트 (v3.6~)

Gradient와 Level Curves



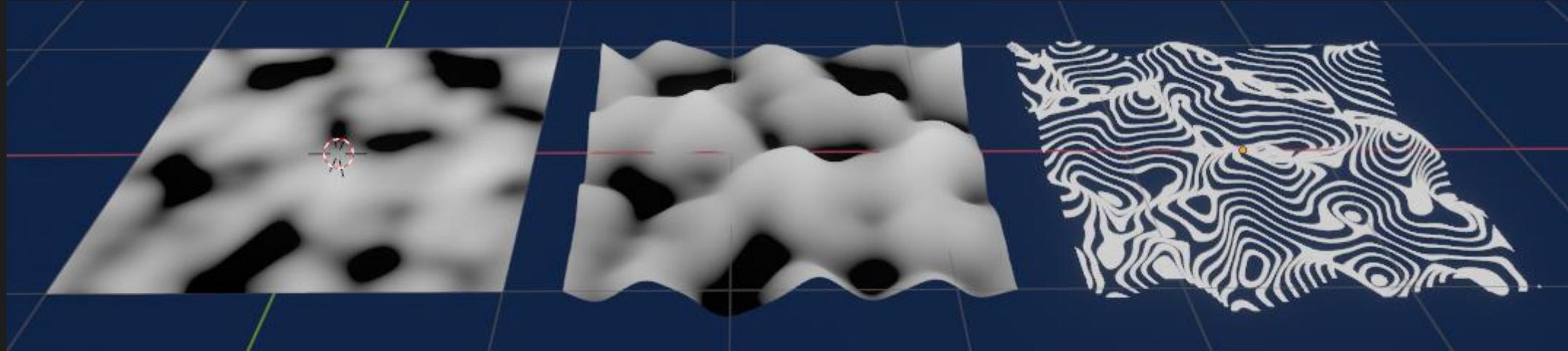
Colossal
Blender
Nodes

회색 텍스쳐

블렌더의 텍스쳐 중 2차원 좌표를 받아 흑백 이미지를 내보내는 $R^2 \rightarrow R$ 를 생각해 봅시다.

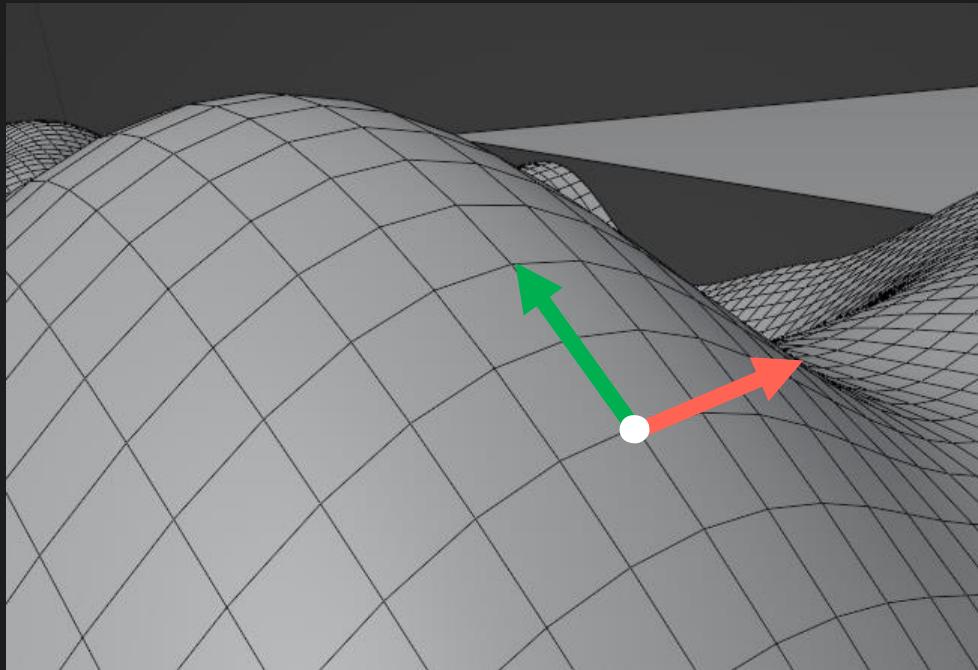
텍스쳐를 높이로 생각한다면, 이것은 $(x,y) \rightarrow z$ 로의 높이 함수입니다.

그렇게 보았을 때, 이것의 **등고선**을 실시간으로 표현해 보는 것이 이번 시간의 목표입니다.



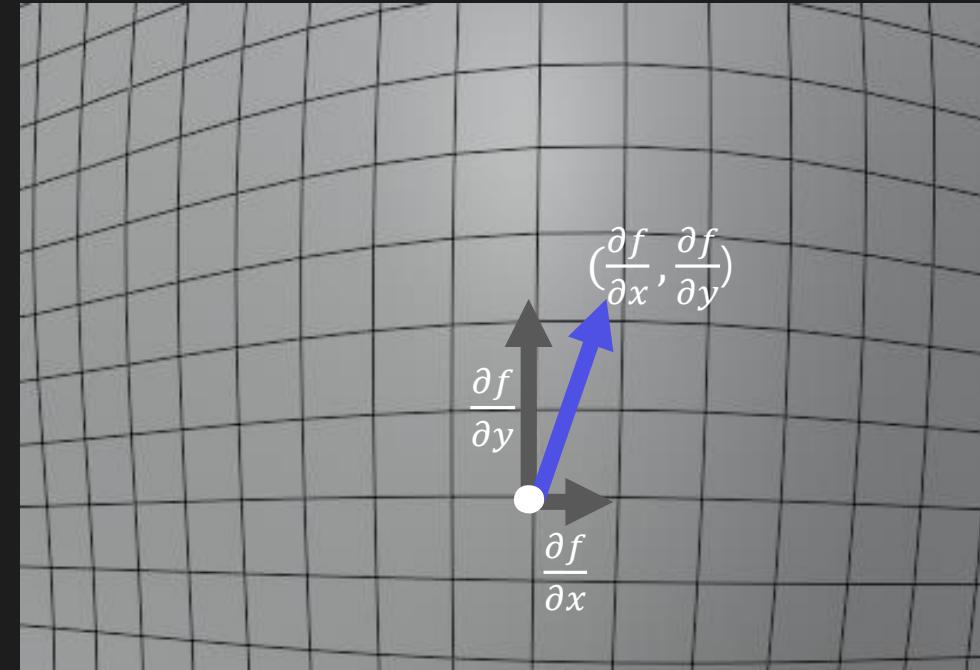
Gradient

$\nabla f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$ 로 표현되는 그래디언트는 언덕을 오르는 방향의 벡터를 나타냅니다.



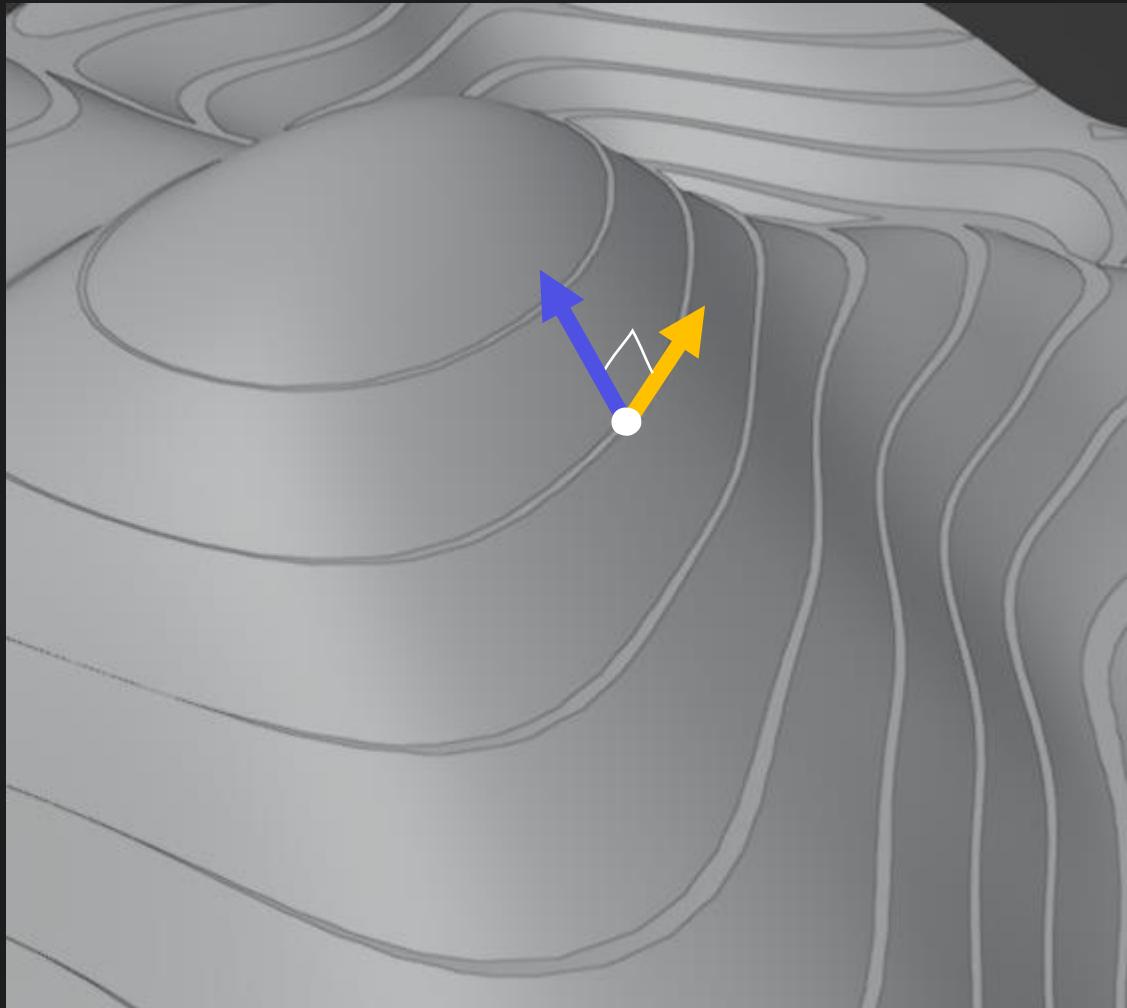
$\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}$ 는 각각 x,y방향으로의 기울기입니다.

흰색 점에서 x축 방향(빨간색)보다 y축 방향(초록색)이 더 가파릅니다.
따라서 $\frac{\partial f}{\partial y}$ 가 $\frac{\partial f}{\partial x}$ 보다 큽니다.



$\left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$ 벡터(파란색)은 평면상에서 언덕을 오르는 방향을 가리킵니다.

Gradient에 수직인 것



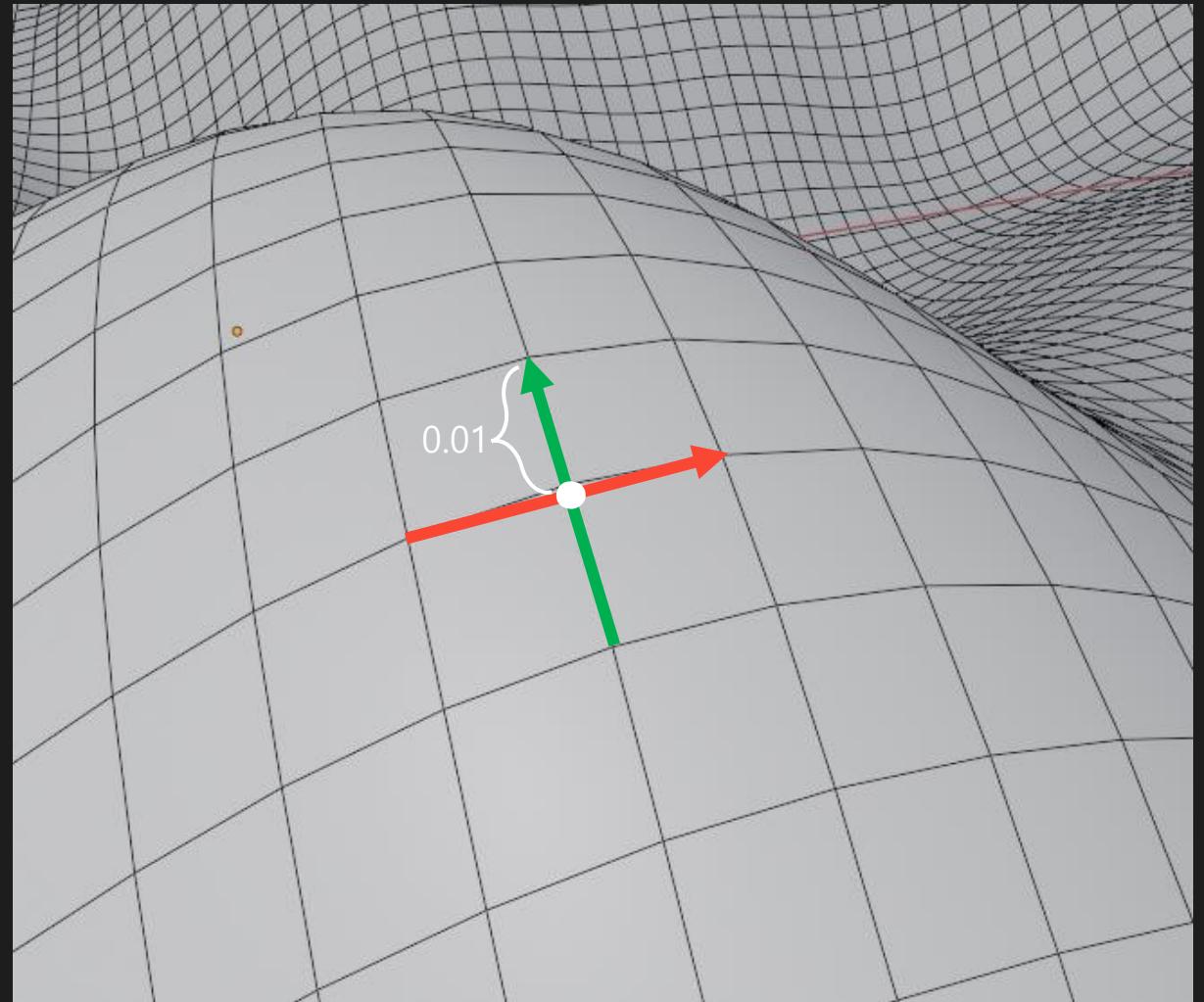
그래디언트 벡터(파란색)는 언덕을 가장 빠르게 올라가는 방향이고, 그래디언트의 반대 방향은 가장 빠르게 내려가는 방향입니다.

한편, 그래디언트에 수직인 방향(노란색)은
오르지도, 내리지도 않는 수평 방향이 됩니다.
따라서 그래디언트에 수직으로 이동하면 **등고선**이 됩니다.

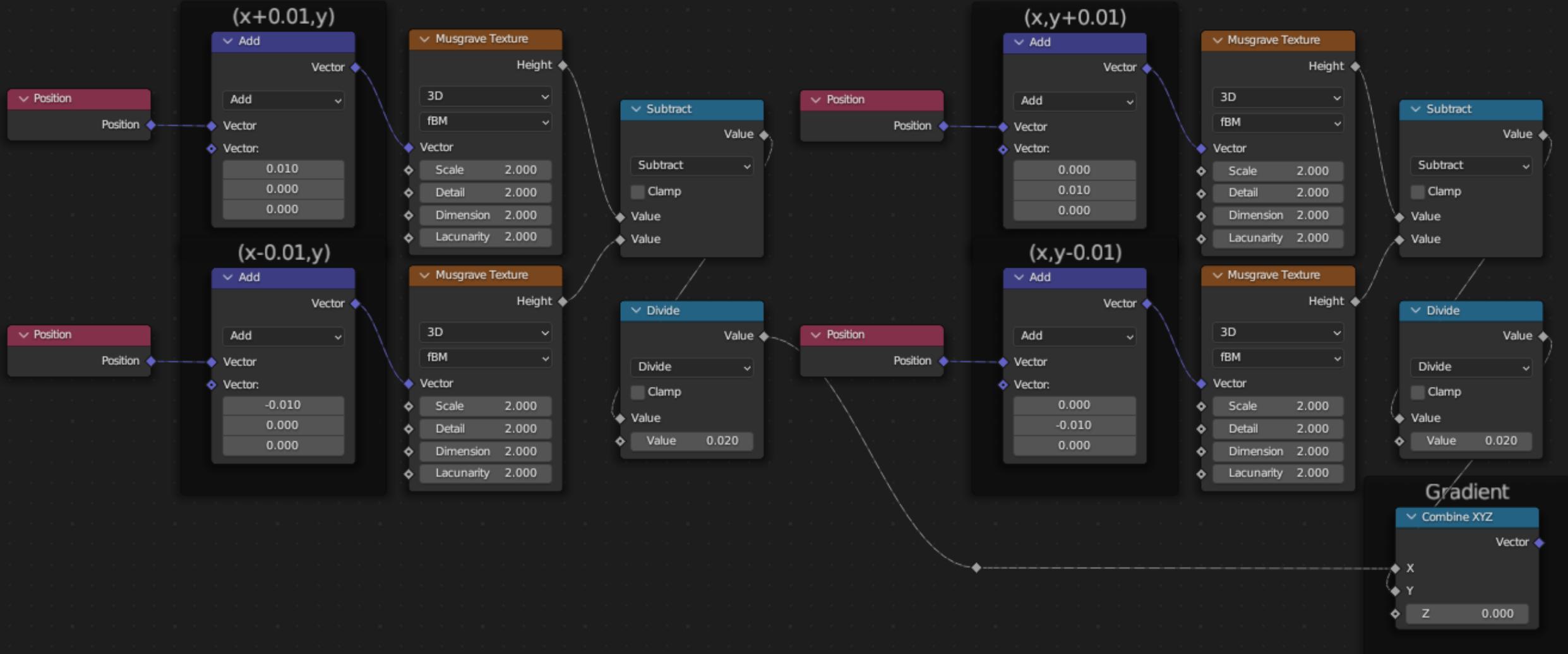
Gradient의 구현

그래디언트를 정확히 구하려면 편미분을 해야 하지만,
충분히 가까운 거리의 기울기를 구하면 근사적으로 구할 수 있습니다.

오른쪽과 같이 x, y 상하좌우로 0.01만큼 떨어진 지점의 높이를 이용하여
기울기를 구하면, 상당히 정확한 그래디언트가 구해집니다.

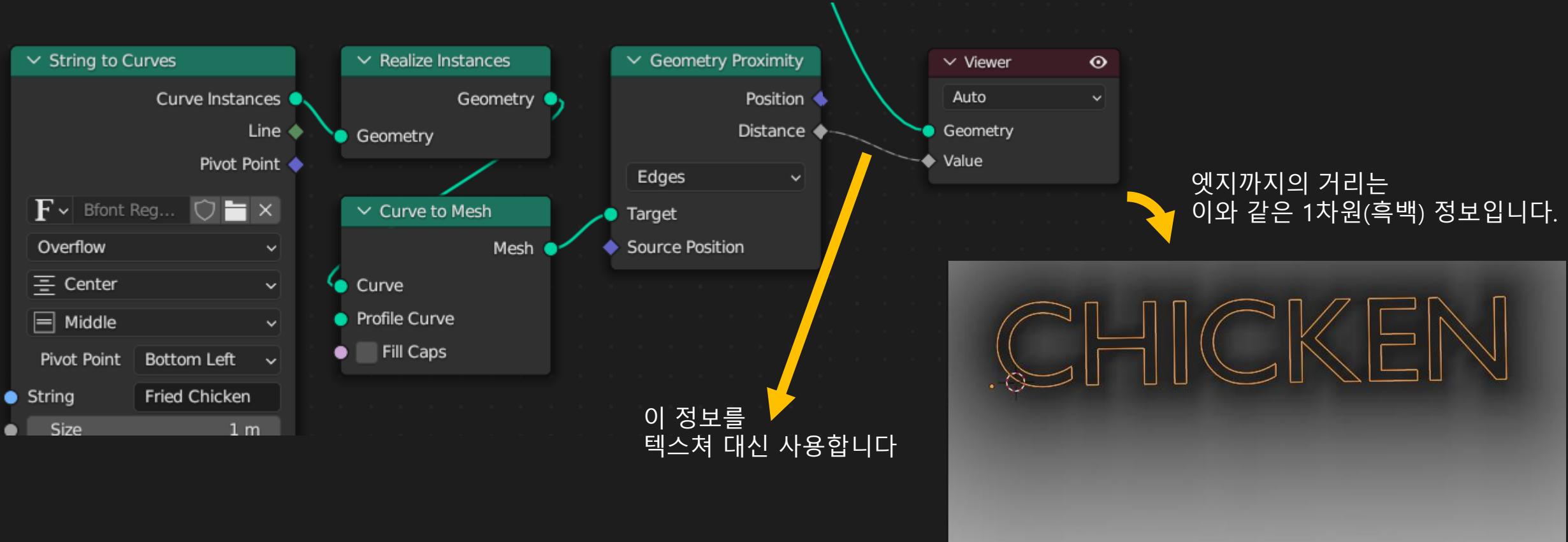


Gradient의 구현



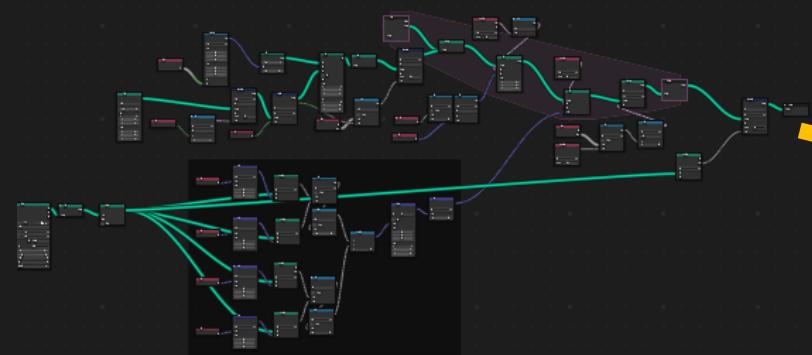
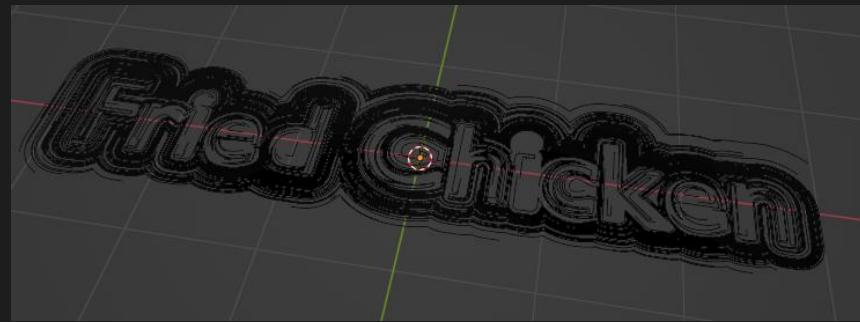
String을 이용한 함수

String to Curves까지의 거리를 이용하면, 문자 주위를 회전하는 등 고선도 만들 수 있습니다.

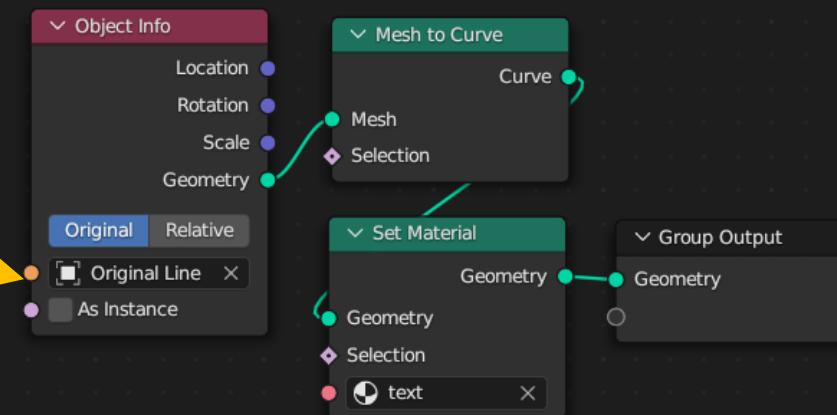
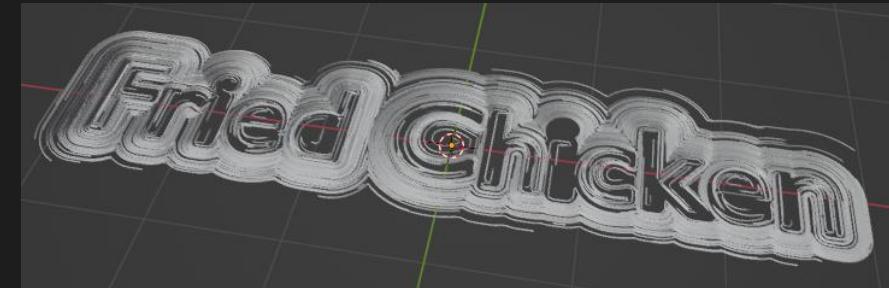


Curves를 이용한 시각화

그냥 Mesh to curve -> Curve to Mesh를 이용한 방법은 많은 폴리곤이 필요하므로 무겁습니다.
지금은 커브의 디테일이 필요하지 않으므로 Curves를 이용하는 것이 좋습니다.



원본 (mesh)

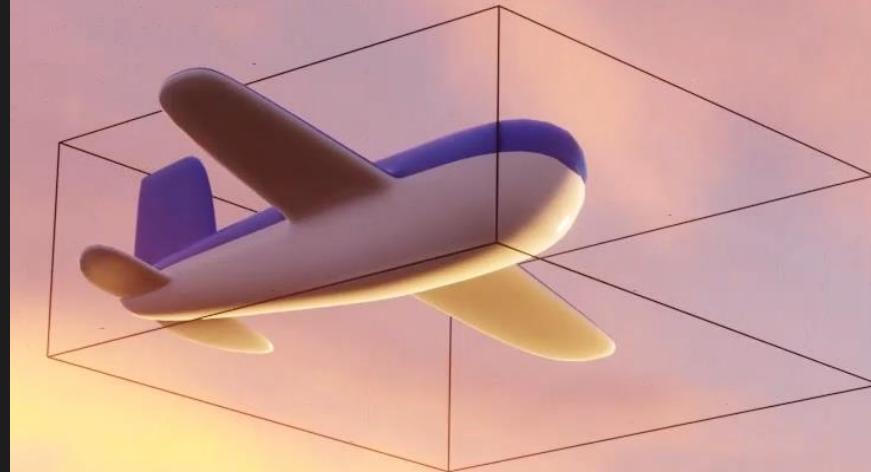


Curves를 만들고 원본을 불러옵니다.

※Curves를 만들 땐 어쩔수 없이 curves가 볼을 오브젝트가 하나 필요합니다.
우리는 헤어를 만드는 것이 아니기 때문에 필요가 없으므로 나중에 지워도 됩니다.

069강 Simulation Node – 감쇠진동 (v3.6~)

2nd Order System



공기저항

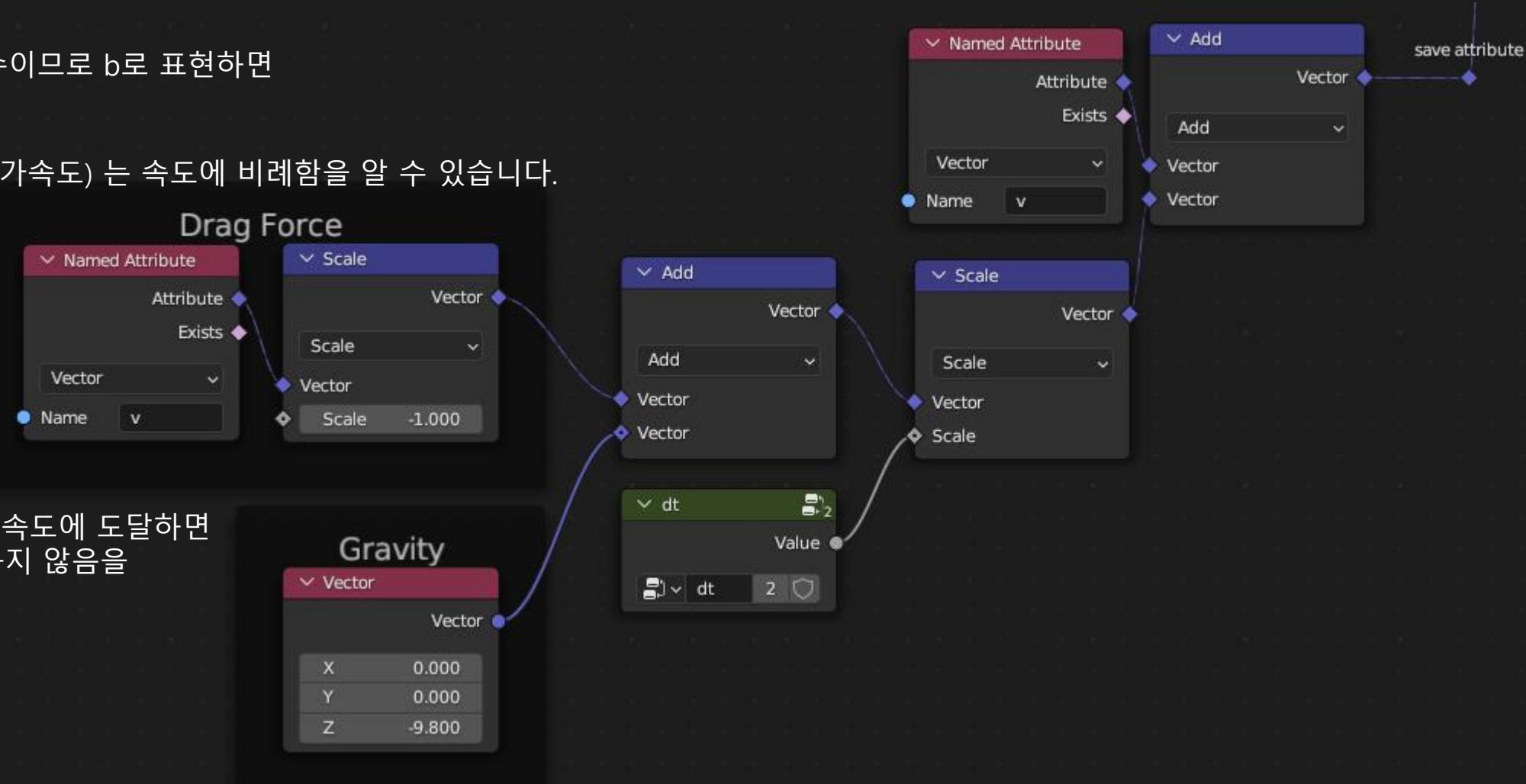
공기저항을 나타내는 스톡스 법칙 (Stokes' law)은 다음과 같습니다.

$$F = 6\pi\eta rv$$

여기서 $6\pi\eta r$ 까지는 상수이므로 b로 표현하면

$$ma = bv$$

가 되어 저항력 (그리고 가속도)은 속도에 비례함을 알 수 있습니다.



이미지처럼 연결하면, 일정 속도에 도달하면
저항에 의해 더 이상 가속하지 않음을
확인할 수 있습니다.

용수철

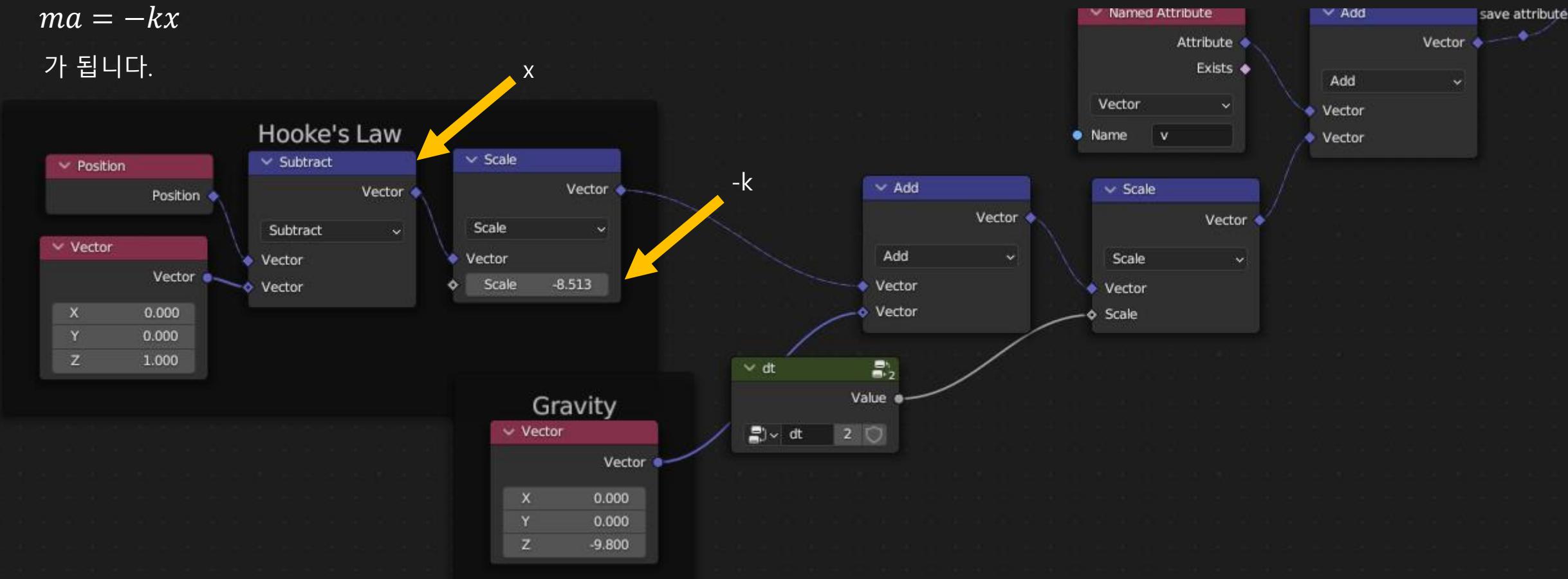
탄성이 있는 물체의 힘을 나타내는 휴의 법칙(Hooke's Law)은 다음과 같습니다.

$$F = -kx \quad (\text{여기서 } x \text{는 현재 위치에서 용수철 끝까지의 거리입니다.})$$

즉

$$ma = -kx$$

가 됩니다.



Second Order System

여기 x 는 흙의 법칙의 x 와는 다릅니다.
또 다른 물체의 위치를 나타냅니다.

앞에서 알아본 감쇠와 진동을 모두 표현하는 식이 있습니다.

$$y''(t) + 2\zeta\omega y'(t) + \omega^2 y(t) = \omega^2 x(t)$$

ω : Natural Frequency

ζ : Damping Ratio

지금까지 계속 써온 가속도(a) 와 속도(v), 위치(s)로 나타내면 다음과 같습니다.

$$a + 2\zeta\omega v + \omega^2(s_y - s_x) = 0$$

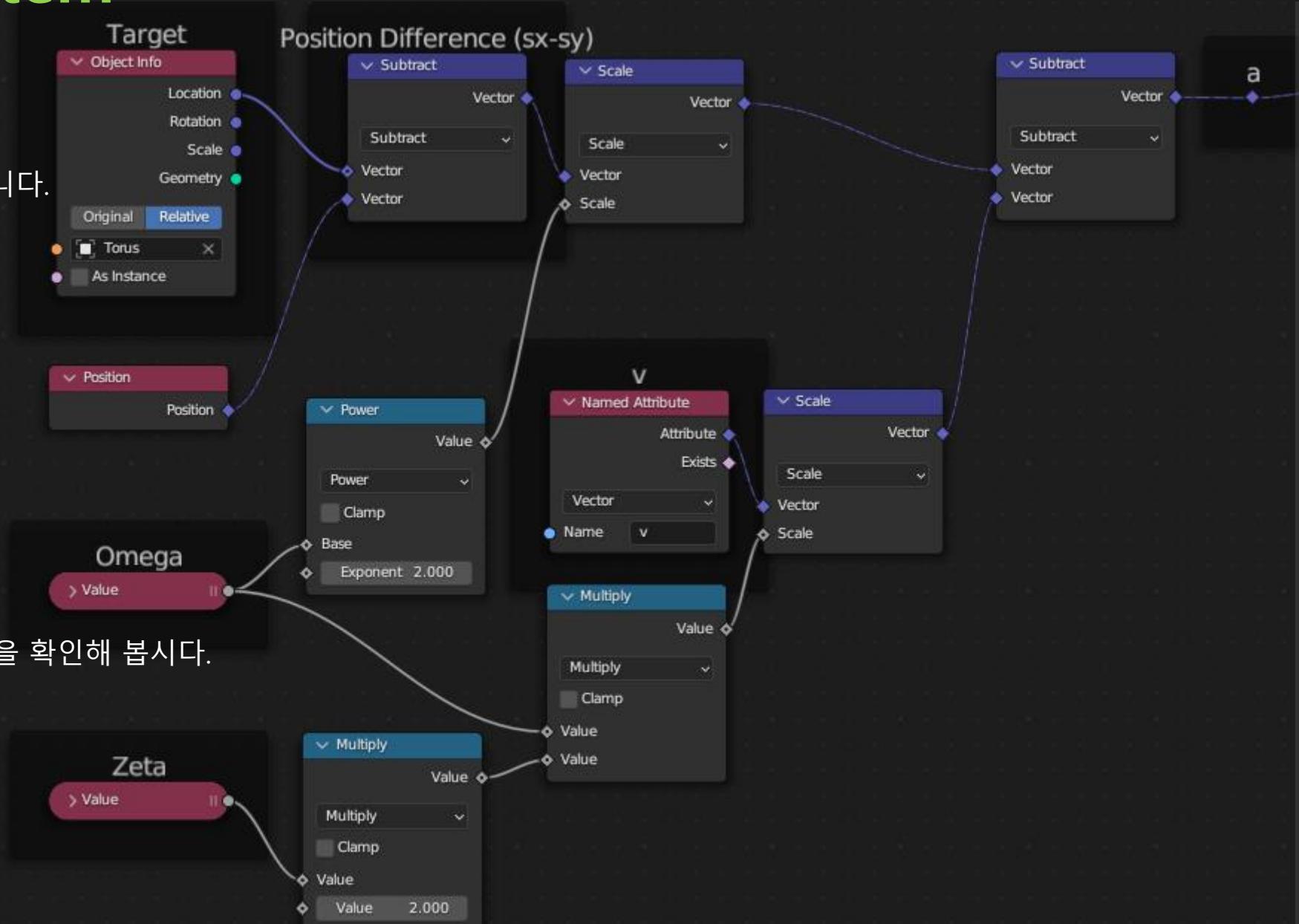
이것은 앞에서 본 저항과 진동을 단지 합친 것입니다!

Second Order System

가속도를 남기고 이항하면

$$a = \omega^2(s_x - s_y) - 2\zeta\omega v$$

이므로, 노드로 만들면 오른쪽과 같습니다.



오메가와 제타의 값을 조절하면서 움직임을 확인해 봅시다.

Rotation?

이 식에는 회전운동이 빠져 있습니다.
회전을 물리적으로 정확하게 계산하기는 어려우므로 납득 가능한 방법으로 표현합니다.

