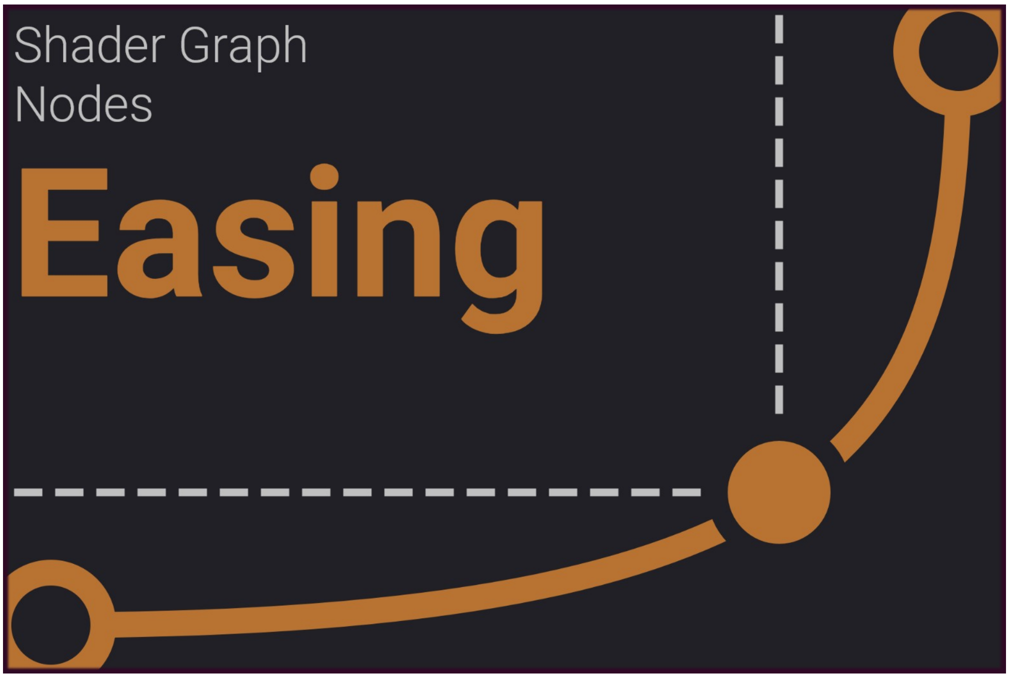
Cerborus

Shader Graph – Easing Nodes

Documentation



Introduction – Easing Functions

Easing functions specify the rate of change of a parameter over time.

Objects in real life don’t just start and stop instantly, and almost never move at a constant speed. When we open a drawer, we first move it quickly, and slow it down as it comes out. Drop something on the floor, and it will first accelerate downwards, and then bounce back up after hitting the floor. [0]

These functions are used across technologies and should be known. In the footer at [0] is a link to an interactive visualization of these Nodes.

TL;DR

- Easing makes your animations **feel more natural**
- Choose **ease Out** animations for UI elements
- Avoid **ease In** or **ease InOut** animations unless you can keep them short. They tend to **feel sluggish** to end users

In classic animation, the term for motion that starts slowly and accelerates is "slow in," and for motion that starts quickly and decelerates is "slow out." The terminology most commonly used on the web for these are “ease in” and “ease out,” respectively. Sometimes the two are combined, which is called "ease in out." Easing, then, is really the process of making the animation less severe or pronounced. [1]

[0] https://easings.net/ - by Andrey Sitnik and Ivan Solovev

[1] https://developers.google.com/web/fundamentals/design-and-ux/animations/the-basics-of-easing

Overview – Easing Nodes

Node Namespace: **Math/Easing**

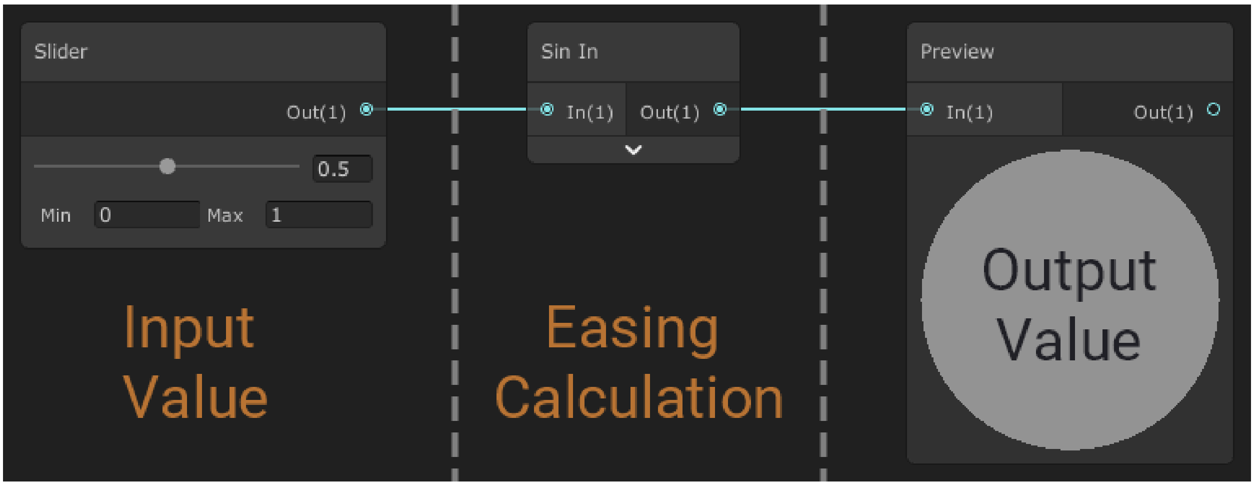
The included Nodes are:

- **Back**
- **Bounce**
- **Circular**
- **Cubic**
- **Elastic**
- **Exponential**
- **Quadratic**
- **Quartic**
- **Quintic**
- **Sin**

All of them have the variants **[In, Out, InOut]**.
For example: Back **In,** Sin **Out,** Sin **InOut,** etc.

The **Linear** easing function is not included as this functionality is already provided by the **Preview Node**.

How to use – Easing Nodes



Properties

Each easing node has an input **[In]** and an output **[Out]** property.

In

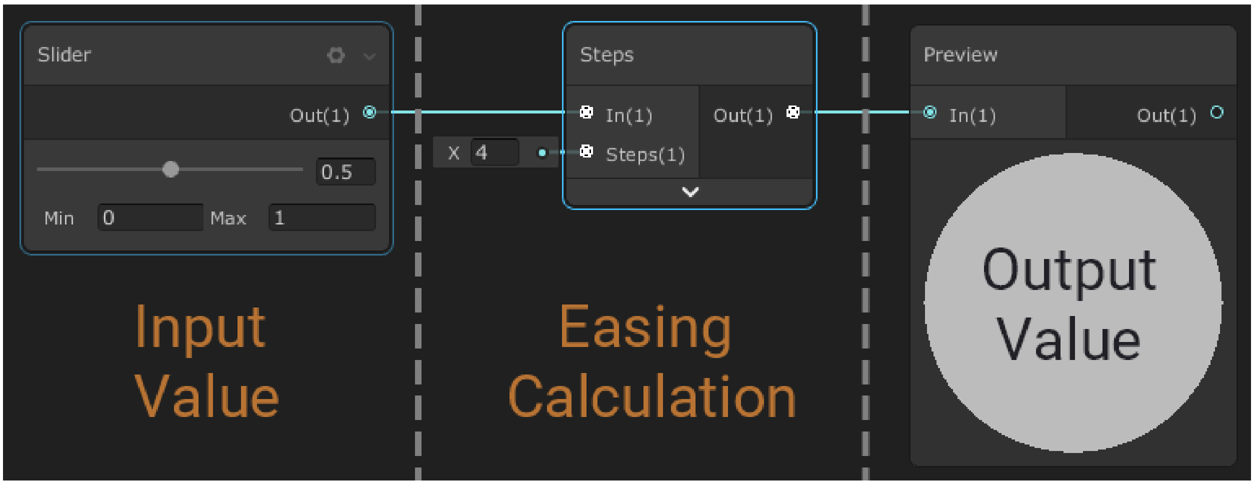
The input value must be a normalized value between 0 and 1. **[0, 1]**
0 and 1 are part of this range.
The nodes **don't clamp** the **In**-value to [0, 1]!

Out

Based on the easing function of the respective node, the input value is remodeled and returned accordingly as **Out**-value.

Steps Node

Create interpolations that have discrete steps.



Properties

It additionally has a **Steps**-Property.

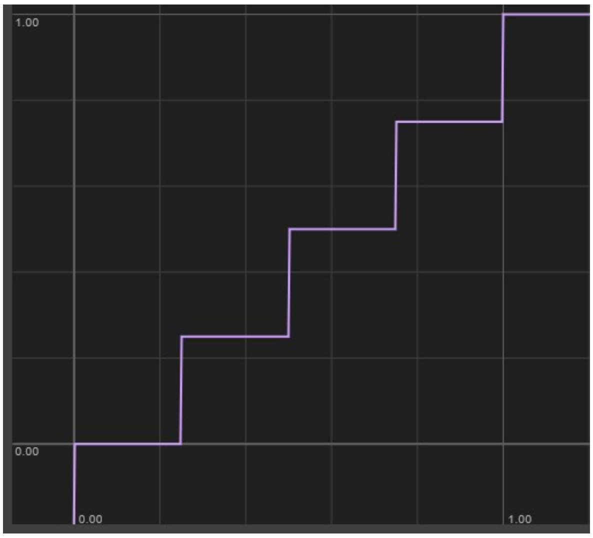
Steps

This property sets the amount of steps.
The **minimum Steps are clamped to 1**.
Only integer values should be set, but float values will be also handled properly.

Example

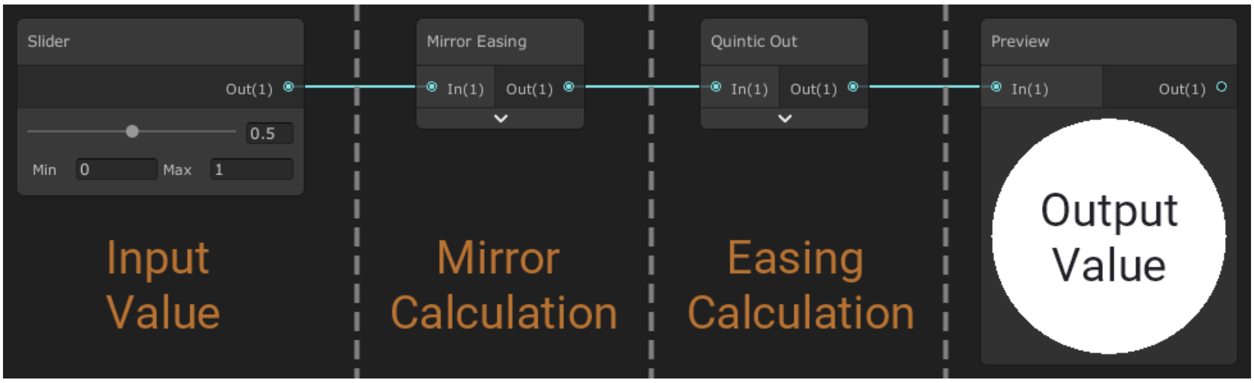
Steps = 4
If the **In-**value is iterated from **0** to **1**, the **Out**-value only changes if the **In-**value is a multiple of 0.25.

In	**Out | Steps = 4**
[0, 0.25[	0
[0.25, 0.50[	0.25
[0.50, 0.75[	0.50
[0.75, 1[	0.75
1	1



Mirror Easing Node

It takes a normalized value as input and returns a mirrored output value. The mirrored value reaches linearly the transition destination **Out = 1** at **In = 0.5**, and then returns to the transition origin **Out = 0** at **In = 1**.



Properties

It has an input **[In]** and an output **[Out]** property.

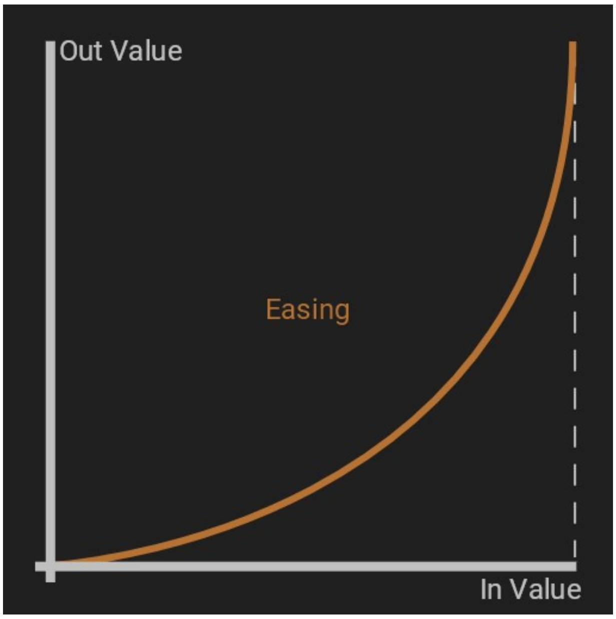
In

The input value must be a normalized value between 0 and 1. **[0, 1]**
0 and 1 are part of this range.
This node **don't clamp** the **In**-value to [0, 1]!

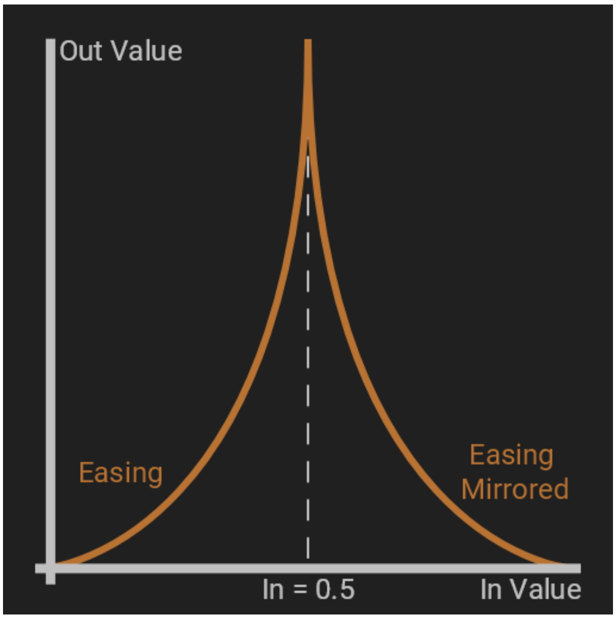
Out

The input value is remodeled and returned accordingly as **Out**-value.

Easing Function



Mirrored Easing Function


Out Value
Easing
Easing Mirrored
In = 0.5
In Value