# 10.3.4.2 Render - Cycles Render Engine - Nodes - Input Nodes

## Input Nodes

### Camera Data

**View Vector**
A Camera space vector from the camera to the shading point.

**View Z Depth**
TODO

**View Distance**
Distance from the camera to the shading point.

### Value

Input a scalar value.

**Value**
Value output.

### RGB

Input an RGB color.

**Color**
RGB color output.

### Attribute

Retrieve attribute attached to the object or mesh. Currently UV maps and vertex color layers can be retrieved this way by their names, with layers and attributes planned to be added. Also internal attributes like *P* (position), *N* (normal), *Ng* (geometric normal) may be accessed this way, although there are more convenient nodes for this.

**Name**
: Name of the attribute.

**Color output**
: RGB color interpolated from the attribute.

**Vector output**
: XYZ vector interpolated from the attribute.

**Fac output**
: Scalar value interpolated from the attribute.

## Wireframe

Retrieve the edges of an object as it appears to cycles. As meshes are triangulated before being processed by cycles, topology will always appear triangulated when viewed with the *Wireframe node*.

**Pixel Size**
: When enabled, set the size of edge lines in screen space.

**Size**
: Thickness of edge lines.

**Fac output**
: Black and white mask showing white lines representing edges according to the object’s topology.

## Geometry

Geometric information about the current shading point. All vector coordinates are in *World Space*. For volume shaders, only the position and incoming vector are available.

**Position**
: Position of the shading point.

**Normal**
: Shading normal at the surface (includes smooth normals and bump mapping).

**Tangent**
: Tangent at the surface.

**True Normal**
: Geometry or flat normal of the surface.

**Incoming**
: Vector pointing towards the point the shading point is being viewed from.

**Parametric**
: Parametric coordinates of the shading point on the surface.

**Backfacing**
: 1.0 if the face is being viewed from the back side, 0.0 for the front side.

**Pointiness**
: An approximation of the curvature of the mesh per-vertex. Lighter values indicate convex angles, darker values indicate concave angles.

## Light Path

Node to find out for which kind of incoming ray the shader is being executed; particularly useful for non-physically based tricks. More information about the meaning of each type is in the *Light Paths* documentation.

**Is Camera Ray output**
1.0 if shading is executed for a camera ray, 0.0 otherwise.

**Is Shadow Ray output**
1.0 if shading is executed for a shadow ray, 0.0 otherwise.

**Is Diffuse Ray output**
1.0 if shading is executed for a diffuse ray, 0.0 otherwise.

**Is Glossy Ray output**
1.0 if shading is executed for a glossy ray, 0.0 otherwise.

**Is Singular Ray output**
1.0 if shading is executed for a singular ray, 0.0 otherwise.

**Is Reflection Ray output**
1.0 if shading is executed for a reflection ray, 0.0 otherwise.

**Is Transmission Ray output**
1.0 if shading is executed for a transmission ray, 0.0 otherwise.

**Ray Length output**
Distance traveled by the light ray from the last bounce or camera.

**Ray Depth**

Number of times the ray has “bounced”, i.e. been reflected or transmitted on interaction with a surface.

> **Note**
>
> Passing through a transparent shader does not count as a normal “bounce”.

**Transparent Depth**
Number of times the ray has passed through a transparent shader.

## Object Info

Information about the object instance. This can be useful to give some variation to a single material assigned to multiple instances, either manually controlled through the object index, based on the object location, or randomized for each instance. For example a Noise texture can give random colors or a Color ramp can give a range of colors to be randomly picked from.

**Location**
Location of the object in world space.

**Object Index**
Object pass index, same as in the Object Index pass.transformed.

**Material Index**
Material pass index, same as in the Material Index pass.

**Random**
Random number unique to a single object instance.

## Fresnel

Dielectric fresnel, computing how much light is refracted through and how much is reflected off a layer. The resulting weight can be used for layering shaders with the *Mix Shader* node. It is dependent on the angle between the surface normal and the viewing direction.

**IOR input**

Index of refraction of the material being entered.

**Fresnel output**

Fresnel weight, indicating the probability with which light will reflect off the layer rather than passing through.

## Layer Weight

Output weights typically used for layering shaders with the *Mix Shader* node.

**Blend input**

Blend between the first and second shader.

**Fresnel output**

Dielectric fresnel weight, useful for example for layering diffuse and glossy shaders to create a plastic material. This is like the Fresnel node, except that the input of this node is in the often more-convenient 0.0 to 1.0 range.

**Facing output**

Weight that blends from the first to the second shader as the surface goes from facing the viewer to viewing it at a grazing angle.

## Texture Coordinate

Commonly used texture coordinates, typically used as inputs for the *Vector* input for texture nodes.

**Generated output**

Automatically-generated texture coordinates from the vertex positions of the mesh without deformation, keeping them sticking to the surface under animation. Range from 0.0 to 1. 0 over the bounding box of the undeformed mesh.

**Normal output**

Object space normal, for texturing objects with the texture staying fixed on the object as it transformed.

**UV output**

UV texture coordinates from the active render UV map.

**Object output**

Position coordinate in object space.

**Camera output**

Position coordinate in camera space.

**Window output**

Location of shading point on the screen, ranging from 0.0 to 1. 0 from the left to right side and bottom to top of the render.

**Reflection output**

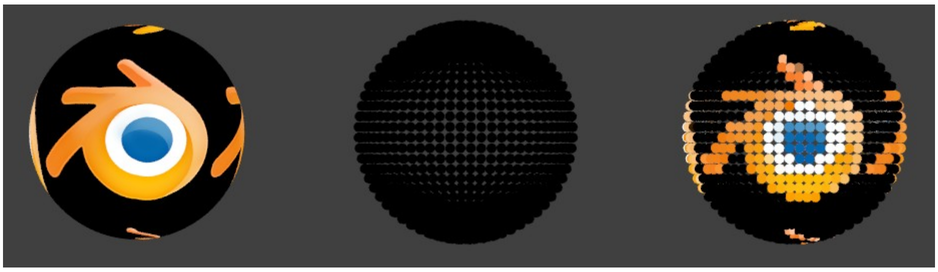
Vector in the direction of a sharp reflection, typically used for environment maps.

**Object**

Specific object to use for object space coordinates. This only affects the *Object* output.

**From Dupli**

If the material is applied to a dupli object, use texture coordinates from the parent object. This only affects the *Generated* and *UV* outputs.



From left to right: Sphere with UV mapped texture. Small spheres duplicated to the faces of the textured sphere using *duplifaces*. Small spheres with *From Dupli* enabled, using the UV map of the large sphere.

**Note**

*From Dupli* only works with the UV output when the dupli object is instanced from faces, either with *particles* or *duplifaces*.

## UV Map

Retrieve specific UV maps. Unlike the Texture Coordinate node which only provides the active UV map, this node can retrieve any UV map belonging to the object using the material.

**From Dupli**
: See the From Dupli option of the *Texture Coordinate node*.

**UV Map**
: UV map to use.

**UV output**
: UV mapping coordinates from the specified UV layer.

## Particle Info

For objects instanced from a particle system, this node give access to the data of the particle that spawned the instance.

**Index**
: Index number of the particle (from 0 to number of particles).

**Age**
: Age of the particle in frames.

**Lifetime**
: Total lifespan of the particle in frames.

**Location**
: Location of the particle.

**Size**
: Size of the particle.

**Velocity**
: Velocity of the particle.

**Angular Velocity**
: Angular velocity of the particle.

# Hair Info

This node gives access to strand information.

**Is strand**
Returns 1 when the shader is acting on a strand, otherwise 0.

**Intercept**
The point along the strand where the ray hits the strand (1 at the tip and 0 at the root).

**Thickness**
The thickness of the strand at the point where the ray hits the strand.

**Tangent Normal**
Tangent normal of the strand.

# Tangent

Generates a tangent direction for the Anisotropic BSDF.

**Direction Type**
The tangent direction can be derived from a cylindrical projection around the X, Y or Z axis (Radial), or from a manually created UV Map for full control.

**Tangent Output**
The tangent direction vector.