En esta práctica vamos a introducir los tipos de iluminación, entornos, cámaras y los materiales, principalmente.

Vamos a comenzar con los materiales, concretamente con el material llamado **Spatial**. Pero antes de introducir los materiales, has de saber que la tecnología que subyace debajo del motor afecta a la visualización final. Así, si usas GLES2 habrá algunos materiales que no luzcan con la misma calidad que GLES3. Más abajo se indican en qué casos diferirán.

1. Introducción a los materiales tipo Spatial

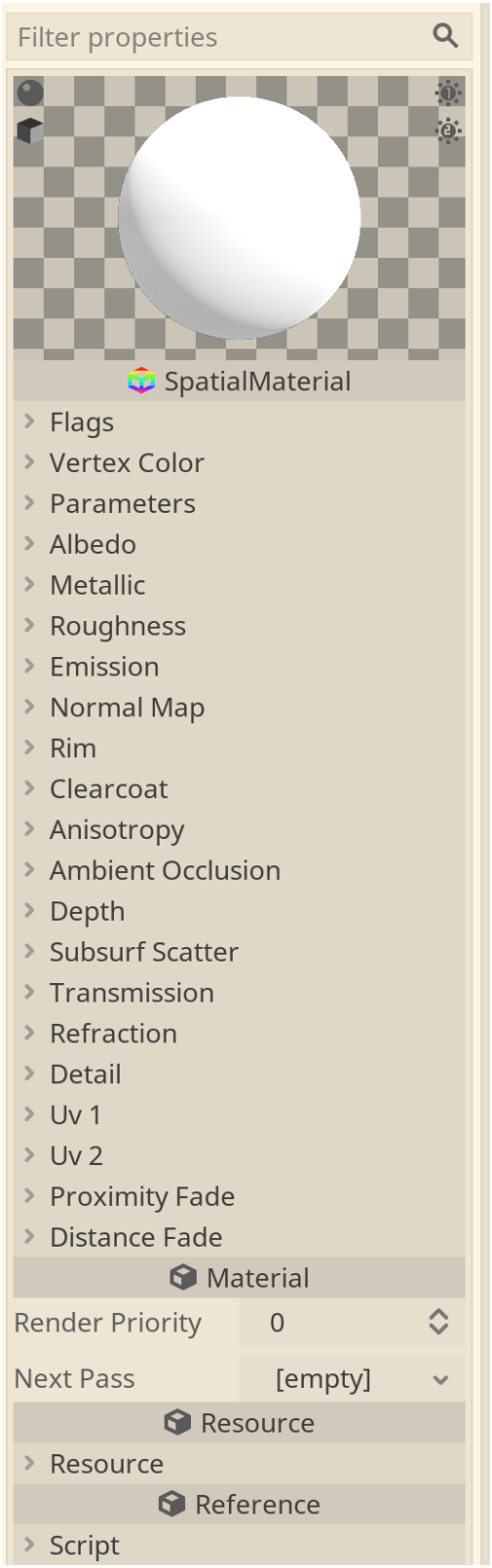
En realidad en Godot existen dos tipos de materiales, el que vamos a ver aquí está diseñado para artistas, que es el nodo **SpatialMaterial**.

Estos materiales pueden convertirse más tarde a código *shader* y modificarse mediante código si se necesita agregar funcionalidades adicionales.

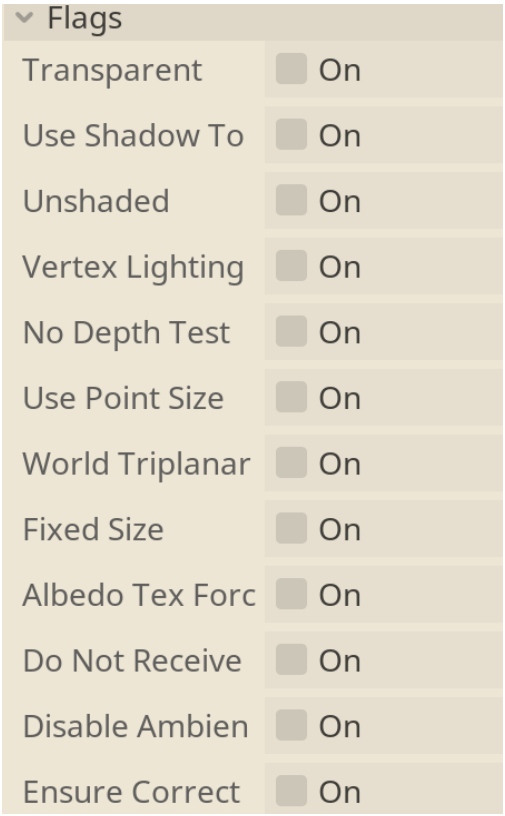
Hay varias maneras de añadir un **SpatialMaterial** a un objeto, algunas de ellas son:

1. En la propiedad **Material** de la malla (**Mesh**).
2. En la propiedad **Material** del nodo que utiliza la malla.

Si se añade un material a la propia malla, cada vez que se utilice esa malla tendrá ese material. Si añade un material al nodo usando la malla, el material solo será usado por ese nodo, también anulará la propiedad de material de la malla.


Filter properties
SpatialMaterial
Flags
Vertex Color
Parameters
Albedo
Metallic
Roughness
Emission
Normal Map
Rim
Clearcoat
Anisotropy
Ambient Occlusion
Depth
Subsurf Scatter
Transmission
Refraction
Detail
Uv 1
Uv 2
Proximity Fade
Distance Fade
Material
Render Priority
0
Next Pass
[empty]
Resource
Resource
Reference
Script

Los materiales tienen muchas propiedades que determinan su aspecto general. Vamos a ver algunas de las propiedades más importantes.



1.1 Transparencia

En Godot, los materiales no son transparentes a menos que se configuren específicamente de esta forma. La razón principal es que los materiales transparentes son renderizados usando una técnica diferente (ordenados de atrás hacia adelante y renderizados en orden, si has implementado alguna vez la técnica de *alpha blending*, o has trabajado en *OpenGL* entenderás el porqué, si no aquí tienes un *pequeño tutorial*).

En resumen podemos decir que esta técnica es menos eficiente que una geometría perfectamente opaca. Por esta razón, los materiales en Godot se muestran opacos a menos que se especifique lo contrario. Los principales ajustes que permiten la transparencia son:

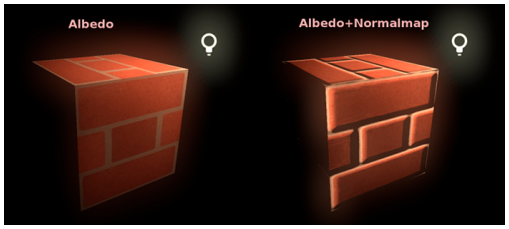
- Opción **Transparent**.
- Modo de composición (**blend mode**) configurado en otro que no sea **Mix**.
- Activar el desvanecimiento de la distancia o la proximidad.

1.2 Usar sombras en opacos

La iluminación modifica el canal alfa (transparencia) para que las áreas sombreadas sean opacas y las no sombreadas sean transparentes.

1.3 Sin iluminación (Unshaded)

En la mayoría de los casos queremos que los materiales aparezcan sombreados (no planos). Pero algunas veces, queremos mostrar simplemente el color (albedo) e ignorar el resto de la iluminación. Al activar esta opción, se eliminarán todo el sombreado y la iluminación difusa y se mostrará el color puro (plano).

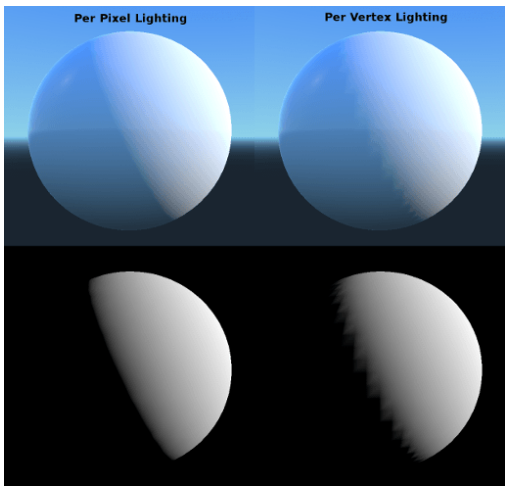


1.4 Iluminación por vértices (Vertex Lighting)

Godot tiene un rendimiento por píxel más o menos constante (gracias al paso previo por profundidad o prepasada en z-buffer). Todos los cálculos de iluminación se realizan ejecutando el *shader* de iluminación en cada píxel.

Dado que estos cálculos son costosos, el rendimiento puede reducirse considerablemente en algunos casos particulares, como el dibujo de varias capas de transparencia (común en los sistemas de partículas). Cambiar a iluminación por vértice (*per-vertex illumination*) puede ayudar en estos casos.

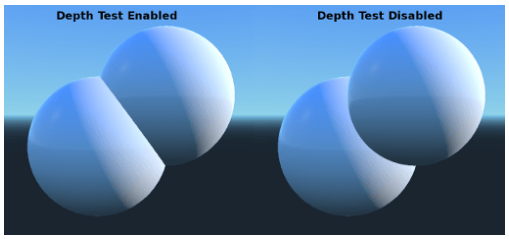
Además, en dispositivos móviles o de gama baja, el cambio a **vertex lighting** puede aumentar considerablemente el rendimiento del renderizado.



1.5 Sin prueba de profundidad (depth test)

Para que los objetos cercanos tapen los objetos lejanos, se realiza una prueba de profundidad (z-buffer). Deshabilitarlo tiene como resultado que los objetos aparezcan por encima (o por debajo) de todo lo demás.

Desactivar esto tiene más sentido para dibujar indicadores en el espacio del mundo (*world space*), y se combina muy bien con la propiedad **Render Priority** de Material.



1.6 Tamaño de punto (Point Size)

Esta opción solo es efectiva cuando la geometría representada está hecha de puntos y no por triángulos (como los modelos escaneados en 3D). Tiene sentido entonces que los puntos se puedan dimensionar.

1.7 Mundo Triplanar (World Triplanar)

Cuando se utiliza el mapeo triplanar (ver abajo, en los ajustes UV1 y UV2), el triplanar se calcula en el espacio local del objeto (local space). Esta opción hace que los triplanares trabajen en el espacio del mundo.

1.8 Tamaño fijo (Fixed Size)

Esto hace que el objeto renderizado tenga el mismo tamaño sin importar la distancia. Es especialmente útil para indicadores (sin prueba de profundidad y alta prioridad de renderizado) y algunos tipos de **billboards** (en clase te comentarán más acerca de estos elementos).

1.9 No recibir sombras (Do Not Receive Shadows)

Hace que el objeto no reciba ningún tipo de sombra que de otro modo sería proyectada sobre él.

1.10 Deshabilitar la luz ambiente (Disable Ambient Light)

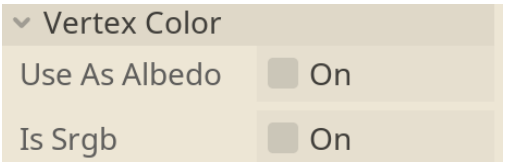
Hace que el objeto no reciba ningún tipo de luz ambiental o de entorno.

1.11 Asegurar que las normales sean correctas

Fija las normales cuando se utiliza una escala no uniforme.

1.11 Color por vértice (Vertex Color)

Esta opción permite elegir lo que se hace por defecto en los colores de los vértices que provienen de su aplicación de modelado 3D. Por defecto se suelen ignorar en todos los programas (Godot no es una excepción en este sentido).



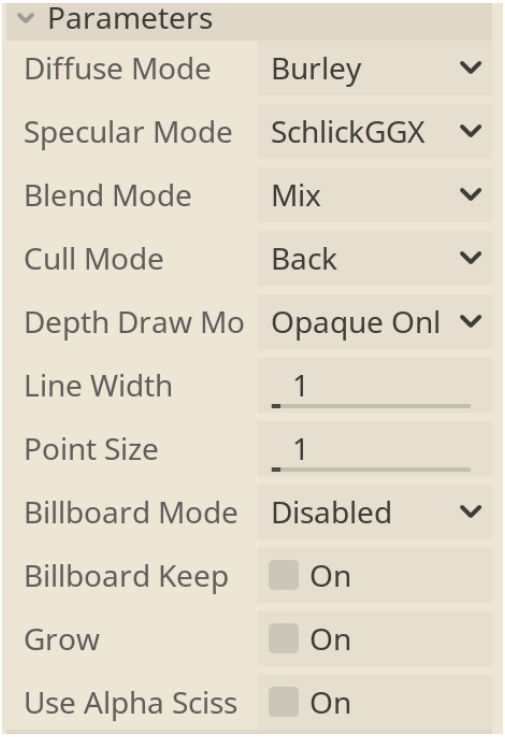
1.12 Usar como albedo (Use as Albedo)

Esta opción hace que el color del vértice se utilice como color del albedo.

La mayoría de las exportaciones que contienen color por vértice, almacenarán los colores de los vértices como sRGB, por lo que activar esta opción ayudará a que se vean correctamente.

1.13 Parámetros

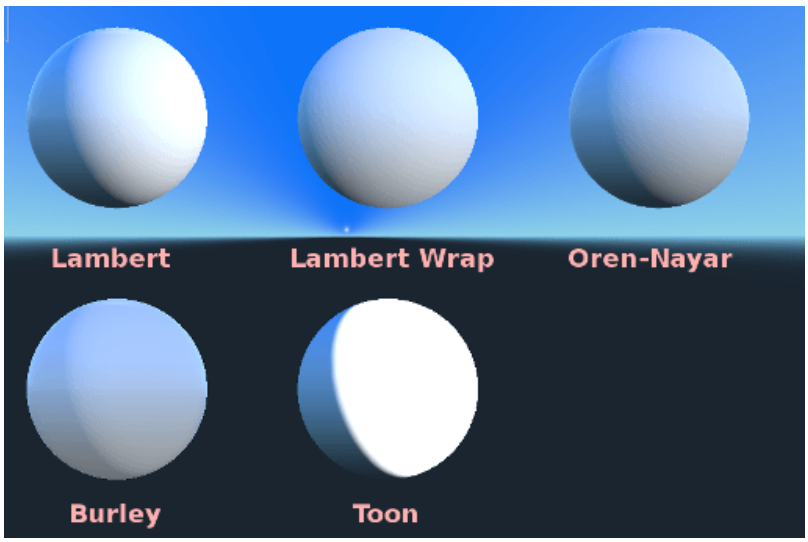
Dentro de estos materiales, tenemos diversos parámetros para ajustar la forma de los materiales.



Modo difuso (Diffuse Mode)

Especifica el algoritmo utilizado para la luz cuando ésta impacta el objeto. La dispersión difusa la verás en clase (al menos una variante sencilla como lo es el modelo de Lambert).

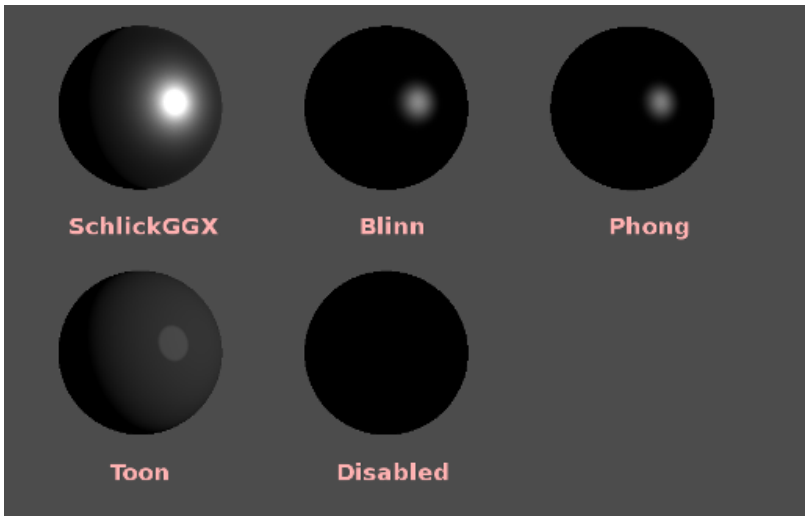
El algoritmo predeterminado se llama **Burley**. También hay otros modos disponibles: - **Burley:** El algoritmo original de Disney (PBS). - **Lambert:** Algoritmo sencillo que no se ve afectado por la rugosidad. - **Lambert Wrap:** Una extensión del modelo de Lambert para cubrir mayores rugosidades cuando tenemos normales de más de 90 grados con respecto a la superficie. Funciona muy bien para el cabello y simula la dispersión a bajo costo de la superficie. - **Oren Nayar:** El objetivo de esta implementación es tener en cuenta microsuperficies (a través de la rugosidad). Funciona bien con materiales similares a la arcilla y algunos tipos de tela. - **Toon:** Proporciona un acabado con iluminación dura (tipo comic), que puede verse afectado por la rugosidad. Es conveniente desactivar luces más realistas y luces ambientales para lograr un mejor efecto.



Modo especular (Specular Mode)

Especifica cómo se renderizarán los brillos especulares de una superficie (verás esto en más detalle en clase).

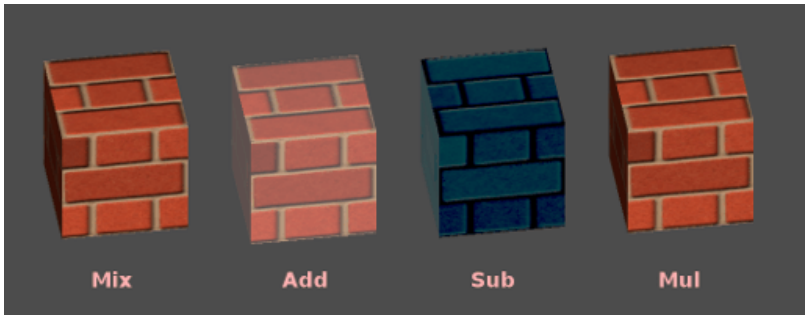
- **ShlickGGX:** Basado en el sistema de Disney (PBR). Muy usado en la actualidad.
- **Blinn:** Común en motores de generaciones anteriores. Se mantiene por compatibilidad.
- **Phong:** Modelo asociado a Lambert, también mantenido por compatibilidad.
- **Toon:** Crea un brillo tipo comic, cambia de tamaño dependiendo de la rugosidad.
- **Disabled:** Elimina los brillos.



Modo de composición (Blend Mode o *blending*)

Controla el modo de mezcla para materiales translúcidos. Hay que tener en cuenta que cualquier otro modo que no sea **Mix** obliga al objeto a pasar por el pipeline transparente.

- **Mix:** Modo de mezcla por defecto, el canal alfa controla cuanto de visible es el objeto.
- **Add:** El objeto se mezcla de forma aditiva, usado en efectos de destellos o similares al fuego.
- **Sub:** El objeto se mezcla de forma sustractiva, utilizado para efectos especiales, como invertir tonos.
- **Mul:** El objeto se mezcla por multiplicación, se usa para cambiar tonos o para aclarar u oscurecer objetos.



Cull Mode

Determina qué lado del objeto NO se dibuja cuando se procesan las caras traseras:

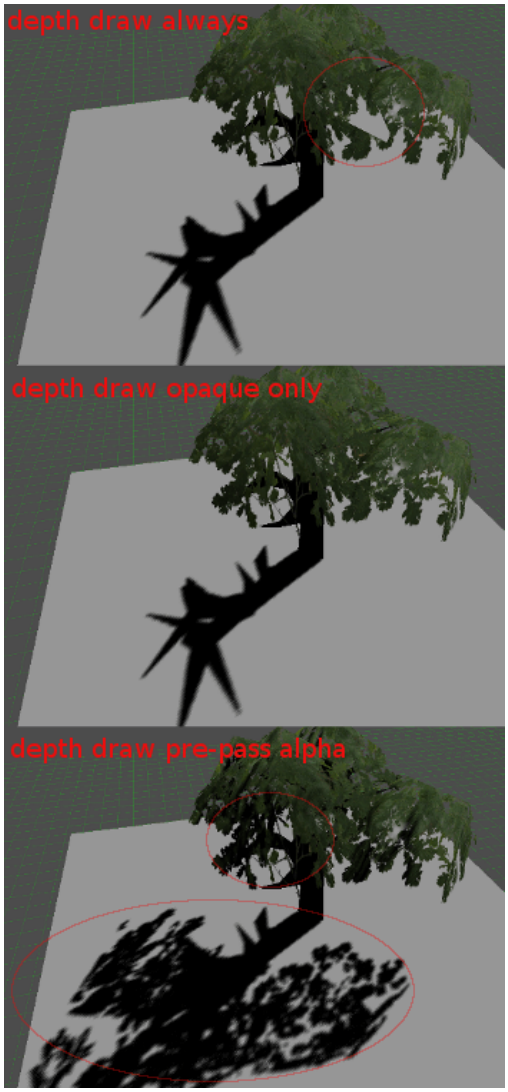
- **Back:** NO se dibuja la parte posterior del objeto cuando no es visible desde un determinado punto de vista (por defecto).
- **Front:** NO se dibuja la parte frontal del objeto cuando no es visible desde un determinado punto de vista (por defecto).
- **Disabled:** Siempre se dibujan ambos lados de una cara (¡cuidado aquí con las normales!). Se usa con objetos de doble cara (no se realiza ningún proceso de *culling*).

Modo de dibujado por profundidad (Depth Draw Mode)

Especifica cuándo se debe realizar el renderizado de profundidad.

- **Opaque Only** (predeterminado): Como si todo fueran objetos opacos.
- **Always:** Se reordena tanto para objetos opacos como transparentes.
- **Never:** No se realiza ningún tipo de reordenado (no es lo mismo que *opaque only*).

- **Depth Pre-Pass:** En el caso de los objetos transparentes, primero se hace una pasada para las partes opacas y luego se dibuja la transparencia de forma ordenada. Esta opción se puede utilizar con césped transparente o follaje de árboles también.



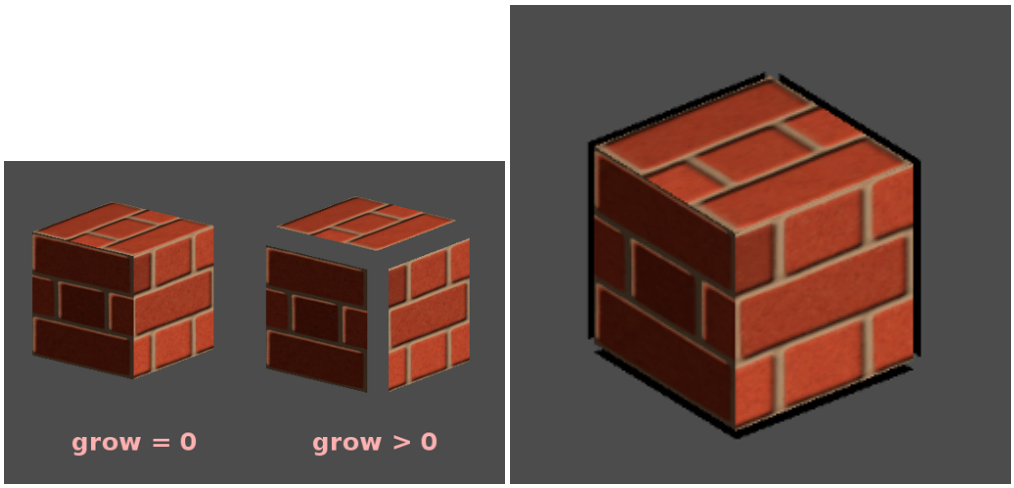
Billboard Mode

Controla la forma en que el objeto mira hacia la cámara:

- *Disabled:* Se desactiva.
- *Enabled:* Se activa, el eje $-Z$ del objeto siempre estará mirando a la cámara.
- *Y-Billboard:* el eje X del objeto siempre estará alineado con la cámara.
- *Particles:* Solamente se usa en sistemas de partículas, permite especificar opciones de animación.

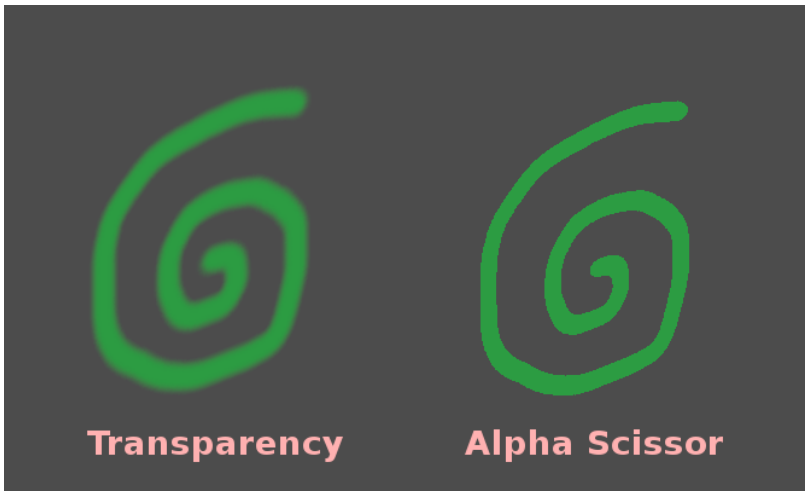
Grow

Hace que los vértices del objeto crezcan en la dirección señalada por sus normales. Este truco se usa mucho para seleccionar objetos. El truco es el siguiente: añade una segunda pasada al material, hazlo negro y sin sombrear, invierte la visualización de las caras (**cull front**), y hazlo un poco más grande.



Usar Alpha Scissor

Cuando no se necesita una transparencia distinta de 0 o 1, es posible establecer un umbral para evitar que el objeto muestre algunos píxeles y otros no.



2. Color de materiales, mapas y canales

Además de los parámetros, lo que define a los materiales mismos son los colores, texturas y canales. Godot soporta una extensa lista de ellos. A continuación se describen con más detalle:

2.1 Albedo

Albedo es el color base del material, sobre lo que trabaja todo lo demás. Cuando se establece la propiedad **Unshaded**, este es el único color visible. Es equivalente a iluminación difusa, pero el cambio de nombre se debe a que en PBR (Physically Based Rendering), este color afecta a muchos más cálculos.

El color y la textura del albedo se pueden utilizar juntos, ya que se multiplicarán entre sí.

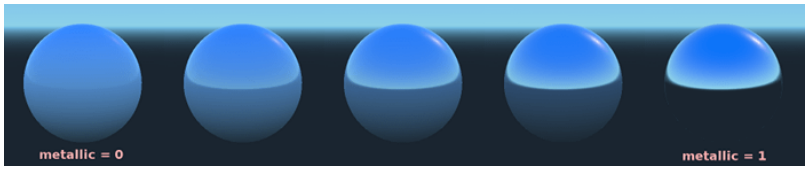
Si utilizas un color o una textura con canal alfa, asegúrate de habilitar la transparencia o la propiedad **alpha scissoring** para que funcione.

2.2 Metallic

Este parámetro define qué tan reflectantes son los materiales. Cuanto más reflectante es, menos es afectado por la luz difusa/ambiental y mayor es el reflejo. Este modelo se llama *energy-conserving*.

Cuidado al combinar los parámetros **Metallic** y **Specular**, ya que en este caso no hay ahorro de energía, con lo que producirá efectos extraños. En estos casos, simplemente déjalo en 0,5 y no lo toques a menos que realmente sepas qué estás haciendo.

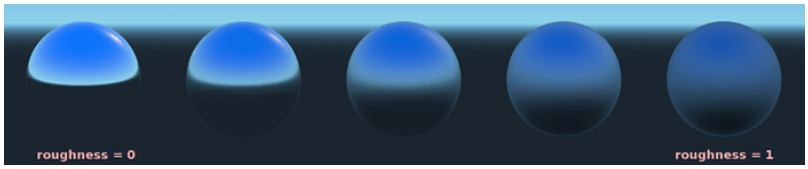
El mínimo de reflectividad interna es de 0.04, por lo que (al igual que en la vida real) es imposible hacer un material completamente antirreflectante.



2.3 Roughness

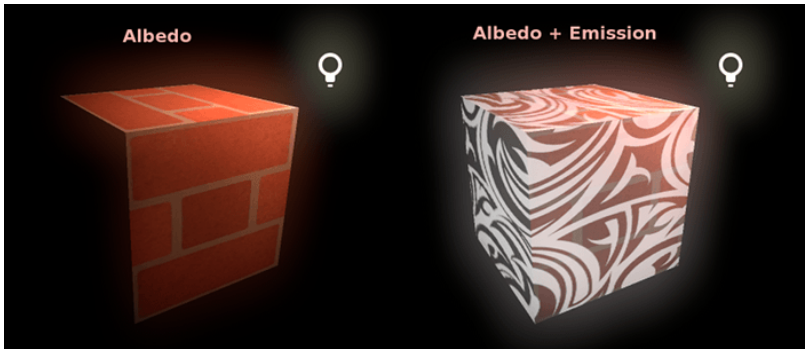
Es la rugosidad, y afecta principalmente a la forma en que se produce el reflejo. Un valor de 0 lo convierte en un espejo perfecto, mientras que un valor de 1 difumina completamente el reflejo (simulando una microsuperficie natural).

Los tipos más comunes de materiales metálicos se suelen lograr con la combinación correcta de **Metallic** y **Roughness**.



2.4 Emission

Especifica cuánta luz se emite por el material (ten en cuenta que esto no incluye la luz que rodea a la geometría a menos que se utilice *GI Probes* en GLES3). Este valor se añade a la imagen final resultante y no se ve afectado por otra iluminación en la escena.

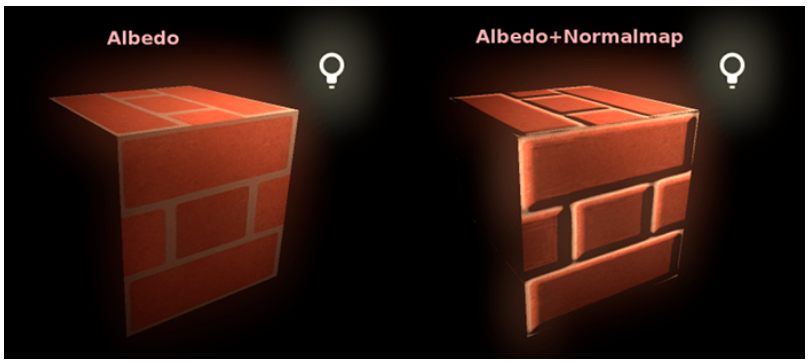


2.5 Normal map

El mapeo de normales (normal mapping) permite establecer una textura que representa detalles de forma más precisos. Esta técnica no modifica la geometría (la verás mejor en clase), solo el ángulo de incidencia de la luz. En Godot, solo se usan los canales rojo (R) y verde (G) con el fin de lograr una mayor compatibilidad.

Godot requiere que el mapa de normales use coordenadas $X+$, $Y-$ y $Z+$. Es decir, que si has importado un material hecho para ser usado en otro motor, tal vez tengas que convertir el mapa de normales, invirtiendo el eje Y.

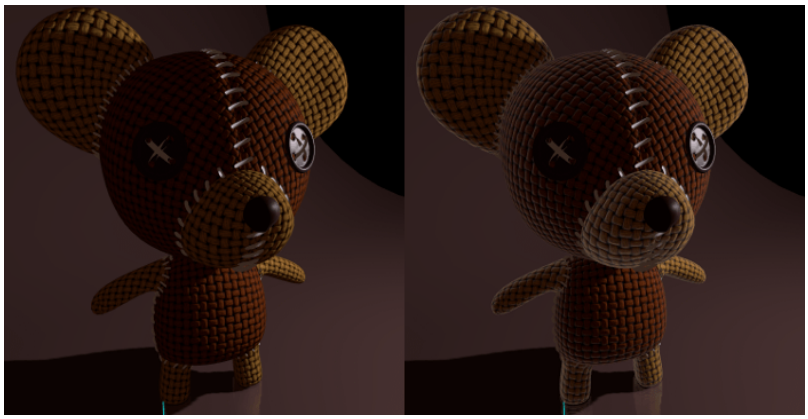
Echa un vistazo a *esta documentación* para más detalles.



2.6 Rim

Algunos tejidos tienen micropelículas que hacen que la luz se disperse a su alrededor. Godot emula esto con el parámetro **Rim**. A diferencia de otras implementaciones de iluminación de bordes que solo utilizan el canal de emisión, éste en realidad tiene en cuenta la luz (si no hay luz, no hay borde). Esto hace que el efecto sea mucho más creíble.

El tamaño del parámetro **Rim** depende también del parámetro **Roughness**, y hay un parámetro especial para especificar cómo debe ser coloreado. Si el parámetro **Tint** es 0, el color de la luz se utiliza para el parámetro **Rim**. Si el parámetro **Tint** es 1, entonces se utiliza el albedo del material. El uso de valores intermedios permite graduar la tonalidad de forma más interesante.



2.7 Clearcoat

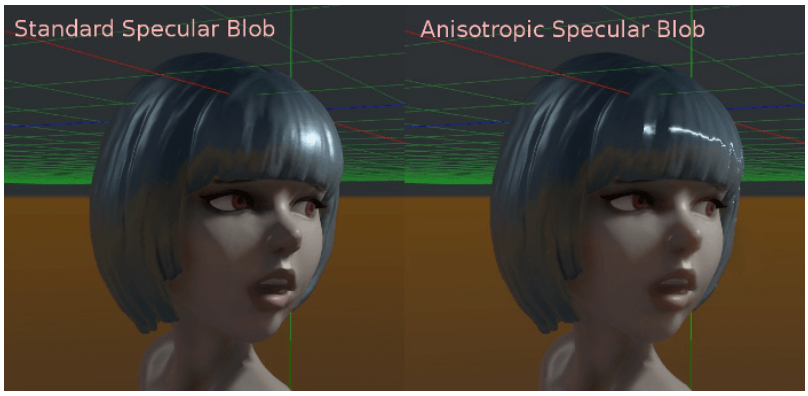
Esta característica solo está disponible cuando se utiliza el motor con GLES3.

El parámetro **Clearcoat** se utiliza principalmente para añadir una pasada secundaria de recubrimiento transparente al material. Este tipo de materiales es común en la pintura de automóviles y juguetes. En la práctica es un brillo especular más pequeño y añadido sobre el material ya existente.

2.8 Anisotropía (Anisotropy)

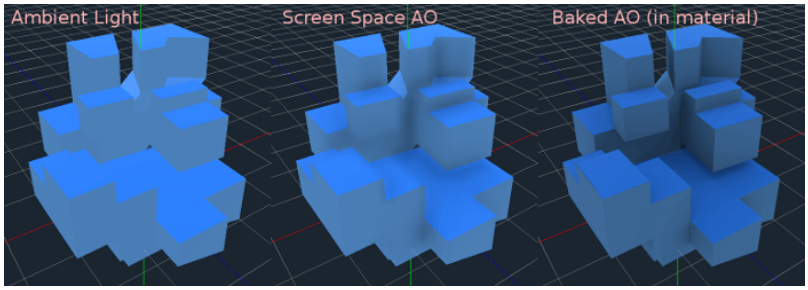
Esta característica solo está disponible cuando se utiliza el motor GLES3.

Este parámetro permite cambiar la forma del brillo especular y lo alinea con el espacio tangencial de la superficie. Se usa comúnmente para simular terciopelo, o para hacer que materiales como el aluminio pulido sean más realistas. Puedes leer más *aquí*.



2.9 Oclusión ambiental (Ambient Occlusion o AO)

Es posible especificar un mapa de oclusión ambiental precocinado (*baked*). Este mapa afecta la cantidad de luz ambiental que llega a cada superficie del objeto (no afecta la luz directa por defecto). Si bien es posible utilizar la oclusión ambiental en el espacio de la pantalla (SSAO) para generar oclusión ambiental, nada superará la calidad de un mapa de oclusión ambiental bien precocinado. Se recomienda precocinar la oclusión ambiental siempre que sea posible.

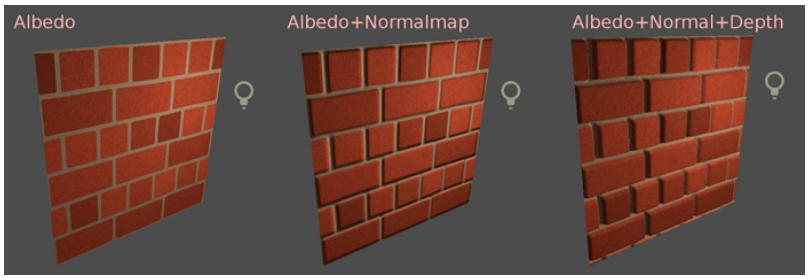


2.10 Profundidad

Esta característica solo está disponible cuando se utiliza el motor GLES3.

Establecer un mapa de profundidad en un material genera una búsqueda de trazado de rayos para emular el desplazamiento adecuado de las cavidades a lo largo de la dirección de la vista. En realidad

no cambia la geometría, sino que es una ilusión de profundidad (parecido al *normal mapping*). Puede que no funcione para objetos complejos pero produce un efecto de profundidad añadido mediante texturas. Para obtener mejores resultados, se debe tendría que alterar la geometría, y añadir *normal mapping*.



2.11 Subsurface Scattering

Esta característica solo está disponible cuando se utiliza el motor GLES3.

Este efecto emula la luz que penetra la superficie de un objeto, se dispersa y luego sale, normalmente con otro color. Es útil para hacer piel realista, mármol, caramelos viscosos, o líquidos de colores, entre otros.

2.12 Transmisión (Transmission)

Controla cuánta luz del lado iluminado (visible a la luz) se transfiere al lado oscuro (lado opuesto a la luz). Esto funciona bien para objetos delgados como hojas de árboles/plantas, césped, orejas humanas, etc.

2.13 Refracción (Refraction)

Esta característica solo está disponible cuando se utiliza el motor GLES3.

Cuando está habilitada, la refracción reemplaza el *alpha blending*. En ese caso Godot intenta obtener información desde detrás del objeto que se está renderizando en su lugar, lo que permite distorsionar la transparencia de forma similar a la refracción en la vida real.

2.14 Detail

Godot permite usar un albedo y la técnica de *normal map* para generar texturas detalladas, permitiendo el mezclado de las mismas de muchas maneras. Combinado con un UV secundario o modo triplanar, se pueden lograr muchas texturas interesantes.

Hay varios ajustes que controlan el uso de los detalles.

- **Máscara:** La máscara de detalle es una imagen en blanco y negro que se utiliza para controlar dónde tiene lugar la mezcla en una textura. El blanco es para las texturas de detalle, el negro es para las texturas materiales regulares, los diferentes tonos de gris son para la mezcla parcial de las texturas materiales y las texturas de detalle.
- **Modo de fusión:** Estos cuatro modos controlan cómo se mezclan las texturas.

    - **Mix:** Combina los valores de los píxeles de ambas texturas. En negro, solo muestra la textura del material. En blanco, solo muestra la textura del detalle. Los valores de gris crean una mezcla suave entre los dos.
    - **Add:** Añade los valores de los píxeles de una textura con la otra. A diferencia del modo **Mix**, ambas texturas se mezclan completamente en las partes blancas de una máscara y no en las partes grises. La textura original es mayormente sin cambios en el negro
    - **Substract:** resta los valores de los píxeles de una textura con la otra. La segunda textura se sustrae completamente en las partes blancas de una máscara con solo una pequeña sustracción en las partes negras, siendo las partes grises diferentes niveles de sustracción basados en la textura exacta.
    - **Mul:** Multiplica los números de canal RGB de cada píxel de la textura superior por los valores del píxel correspondiente de la textura inferior. Produce cambios de tono o luminosidad.

- **Albedo:** Aquí es donde pones una textura de albedo que quieres mezclar. Si no hay nada en esta ranura se interpretará como blanco por defecto.
- **Normal:** Aquí es donde se coloca una textura de normales que se quiera mezclar. Si no hay nada en esta ranura se interpretará como un mapa plano normal. Esta técnica se puede usar incluso si el material no tiene el mapa de normales habilitado.

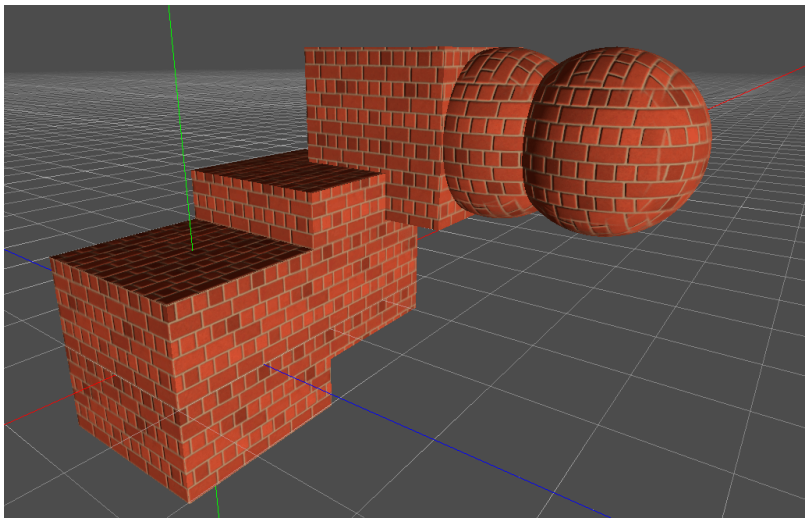
Godot soporta dos canales UV por material. Los UV secundarios son a menudo útiles para oclusión

ambiental o **emission** (*baked light*). Los UVs se pueden escalar y desplazar, lo que es muy útil en texturas que tengan repetición (*repeat* en oposición a *clap* o *mirror*).

2.15 Mapeado Triplanar

El mapeo triplanar es compatible tanto para UV1 como para UV2. Esta es una forma alternativa de obtener coordenadas de textura, a menudo llamada *Autotexture*. Las texturas se muestrean en los tres ejes (X, Y y Z) y se mezclan usando la normal. El mapeo triplanar puede realizarse en coordenadas de mundo o coordenadas en espacio de objeto.

En la imagen se puede apreciar cómo las primitivas comparten el mismo material con triplanar en coordenadas de mundo, para que la textura tenga continuidad entre las partes.



2.16 Desvanecimiento de proximidad y distancia

Godot permite que los materiales se desvanezcan por proximidad entre ellos, además de hacerlo por la distancia con el observador. El desvanecimiento por proximidad es útil para efectos como partículas suaves o masas de agua que se mezclen de forma suave con la orilla. El desvanecimiento por distancia es útil para entradas de luz o indicadores que solo están presentes a cierta distancia. También es muy útil para desvanecer los objetos en la distancia antes de eliminarlos (por eficiencia) del mapa.

Hay que tener en cuenta que habilitarlos también activa el *alpha blending*, por lo que abusar de ellos durante toda una escena no suele ser una buena idea.

2.17 Prioridad de Renderización

El orden de renderizado de los objetos se puede cambiar, esto es útil sobre todo para los objetos transparentes (u objetos opacos que se dibujan en profundidad pero no en color, como grietas en el suelo).

En la *documentación* puedes encontrar algunos detalles adicionales.

Una último apunte sobre materiales: los materiales pueden grabarse y cargarse desde el editor (prueba a usar el RBM encima del SpatialMaterial). Esto es muy útil, no solo para reutilizar *shaders* ya realizados, sino también por términos de eficiencia. Ten en cuenta que cada programa de *shader* tiene que transferirse a la GPU, compilarse en tiempo real y mantenerse mientras haya objetos activos con este material.

Ejercicio calificable P3. A.

> Importa los modelos creados en Blender de la práctica anterior en tu escenario, y añádeles materiales en Godot.

> Al menos uno de los objetos (idealmente el más detallado), debería tener varios materiales lo más realistas posibles. Presta especial atención a la técnica a emplear y a la composición de las distintas propiedades según el comportamiento de la luz en la realidad. Acércate lo máximo posible al realismo. Puedes incluso crear un proyecto nuevo en Godot solamente para exhibir este modelo.

> En este ejercicio también se valorará el compromiso entre calidad y eficiencia. Ten en cuenta que las texturas pueden llegar a consumir muchos recursos (especialmente en GPU), y que existen otras soluciones más *inteligentes* en muchos materiales.

> Finalmente, recuerda que todas las texturas que incluyas tienen que estar en la carpeta del proyecto, e incluidas como recursos.

3. Iluminación

3.1 Introducción a la iluminación

Las luces emiten luz que se mezcla con los materiales produciendo un resultado visible. La luz puede proceder de varias fuentes en una escena:

- **Materiales:** en la forma de color de emisión (*emission*), aunque no afecta a los objetos cercanos salvo que la luz sea pre-calculada.
- **Nodos de luz:** Directional, Omni y Spot.
- **Luz Ambiental** en el nodo *WorldEnvironment*.
- **Baked Light:** luces precocinadas, se asocia a la iluminación global (lo verás mejor en clase), aunque puedes leer más *aquí*).

Ejercicio de entrenamiento 1:

> Echa un vistazo a la documentación del nodo **WorldEnvironment**, prueba a añadir uno de estos nodos y a jugar con los parámetros (añadiendo un elemento **Environment**). En caso de duda pregunta a tu profesor. ¿Serías capaz de poner el entorno completamente a oscuras?

> Presta atención a los distintos efectos del nodo **Environment**, en especial Glow, intenta colocar algún material de tipo *emission*, ¿qué sucede con los objetos de alrededor?

3.2 Nodos de luz

Existen tres tipos de nodos de luz: - Directional light: luz direccional. - Omni light: luz puntual (o posicional). - Spot light: o luz de foco.

Cada uno tiene diferentes usos y los verás en clase, pero primero echemos un vistazo a los parámetros comunes para las luces. Cada uno tiene una función específica:

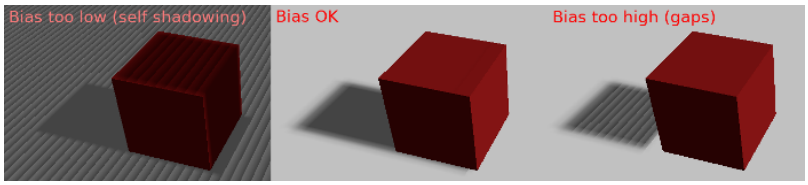
- **Color:** Color base para la luz emitida.
- **Energy:** Multiplicador de energía. Resulta útil para saturar luces o para trabajar con iluminación de alto rango dinámico.
- **Indirect Energy:** Multiplicador secundario utilizado con luz indirecta (rebote de luz). Esto funciona en luz precocinada o nodos de tipo **GIProbe**.
- **Negative:** La luz se vuelve sustractiva en lugar de aditiva. A veces es útil compensar manualmente algunas esquinas oscuras.
- **Specular:** Afecta la intensidad de brillos especulares en los objetos afectados por esta luz. Cuando es 0, esta luz se convierte en una luz pura y difusa.
- **Bake Mode:** Establece el modo de precocinado (del que ya hemos hablado) para la luz.
- **Cull Mask:** Los objetos que se encuentren en las capas seleccionadas que estén debajo serán afectadas por esta luz.

3.3 Mapeo de sombras

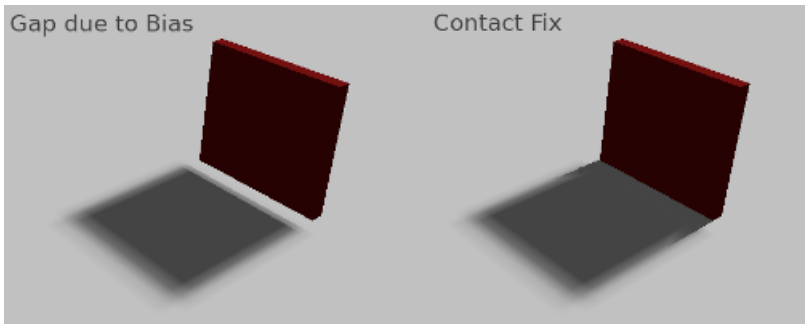
Las luces pueden proyectar sombras, esto les da mayor realismo a coste de un menor rendimiento (en clase verás varios algoritmos para producir sombras *arrojadas*, y como afecta al rendimiento). Hay una lista de parámetros de sombra genéricos, cada uno también tiene una función específica:

- **Enabled:** Activa el mapeo de sombras con esta luz. Afecta al rendimiento.
- **Color:** Las áreas ocluidas se multiplican por este color. Por defecto es negro, pero puede cambiarse para colorear las sombras.
- **Bias:** Cuando este parámetro es muy pequeño, ocurre el *auto-sombreado*. Cuando es muy largo, las sombras se separan de sus emisores. Ajústalo a lo que mejor funcione para tu propósito.
- **Contact:** Realiza un trazado de rayos de corto alcance en el espacio de la pantalla para reducir el hueco generado por el parámetro *bias*. Este parámetro solo funciona cuando se usa el motor GLES3.
- **Reverse Cull Faces:** Algunas escenas funcionan mejor cuando el mapeo de sombras se representa con el *face-culling* invertido.

A continuación se muestra una imagen tras cambiar el valor de bias. Los valores predeterminados funcionan para la mayoría de los casos, pero en general depende del tamaño y la complejidad de la geometría.



Finalmente, si las grietas no se pueden solucionar, la opción **Contact** puede ayudar:

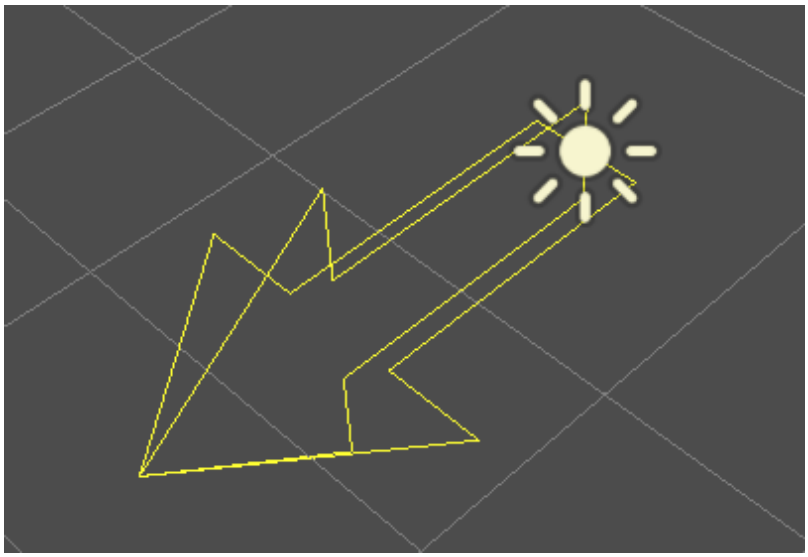


Cualquier problema producido por el parámetro **Bias** puede arreglarse incrementando la resolución del mapa de sombras, aunque esto es equivalente a usar texturas de gran tamaño, y por tanto, puede producir una pérdida de rendimiento en el hardware de gama baja.

3.4 Directional light (Luz direccional)

Este es el tipo de luz más común y representa una fuente de luz muy lejana (como el sol). También es la luz menos costosa de calcular y debería utilizarse siempre que sea posible (aunque ¡cuidado!, su mapa de sombras no es menos costoso de calcular).

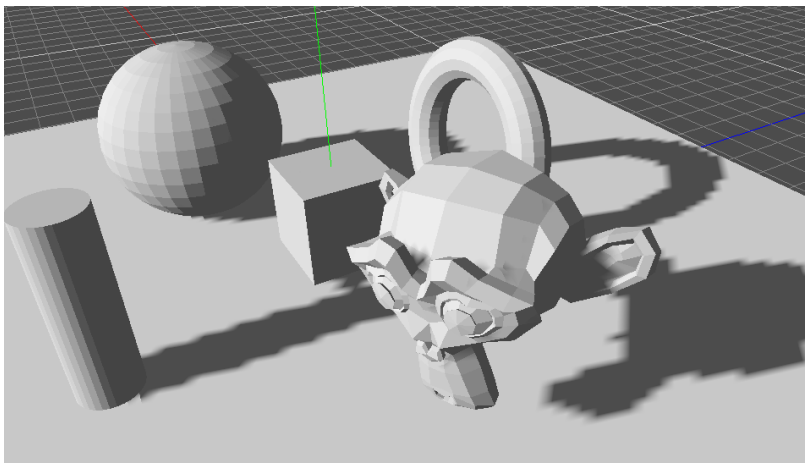
La luz direccional modela un número infinito de rayos de luz paralelos que cubren la escena completa. El nodo de luz direccional se representa por una gran flecha que indica la dirección de los rayos de luz. Sin embargo, la posición del nodo no afecta al comportamiento de luz y puede ser colocado en cualquier lugar de la escena.



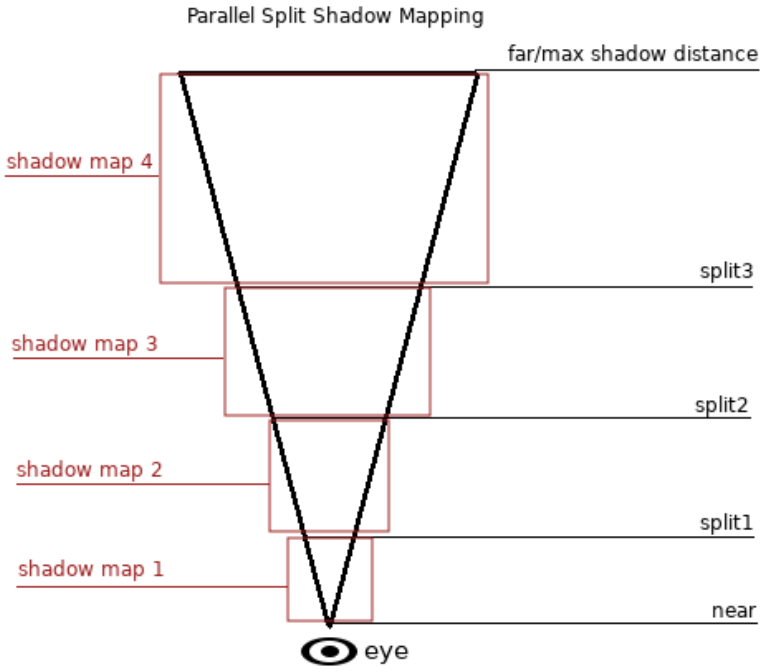
Cada cara, cuyo lado frontal es golpeado por los rayos de luz se ilumina, mientras las otras permanecen oscuras. La mayoría de los tipos de luces tienen parámetros específicos, pero las luces direccionales son de naturaleza bastante simple y no los tienen.

Mapeo de sombras direccional

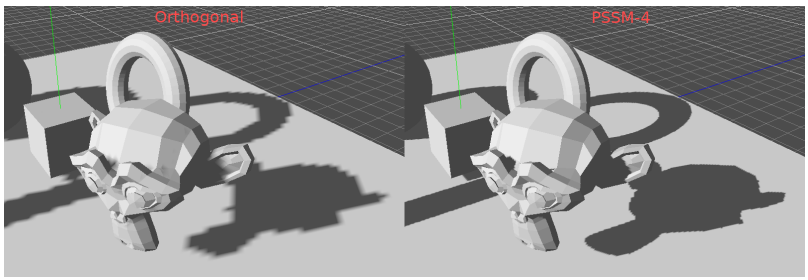
Para calcular mapas de sombra, la escena se renderiza en Godot (usando el buffer de profundidad) desde un punto de vista ortogonal que cubre toda la pantalla (o hasta la distancia máxima). Hay, sin embargo, un problema con este enfoque porque los objetos más cercanos a la cámara reciben sombras en bloque.



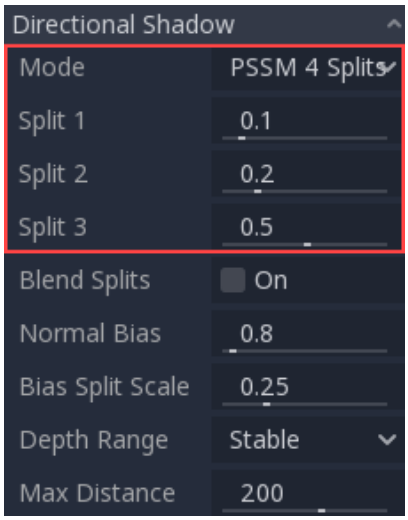
Para solucionarlo, se utiliza una técnica llamada *Parallel Split Shadow Maps* (o PSSM). Esto divide el campo de visión en 2 o 4 áreas. Cada área tiene su propio mapa de sombras. Esto permite que áreas pequeñas cercanas al espectador tengan la misma resolución de sombra que un área enorme y lejana.



Con esto, las sombras se vuelven más detalladas:



Para controlar este método PSSM, se exponen varios parámetros:



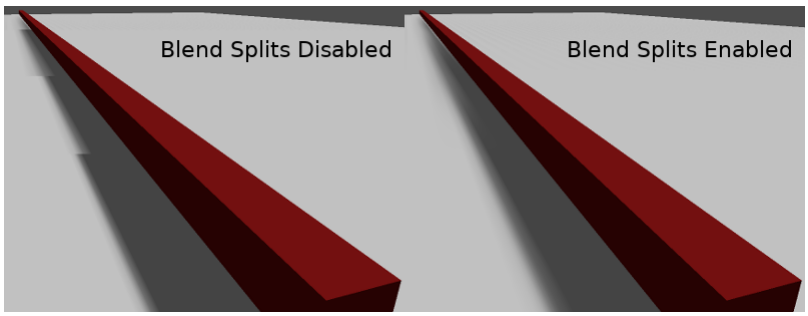
Cada distancia dividida se controla con respecto a la cámara o con el parámetro **Max Distance** (que establece una distancia máxima de sombreado).

0.0 es la posición del ojo y 1.0 es donde la sombra termina a una distancia.

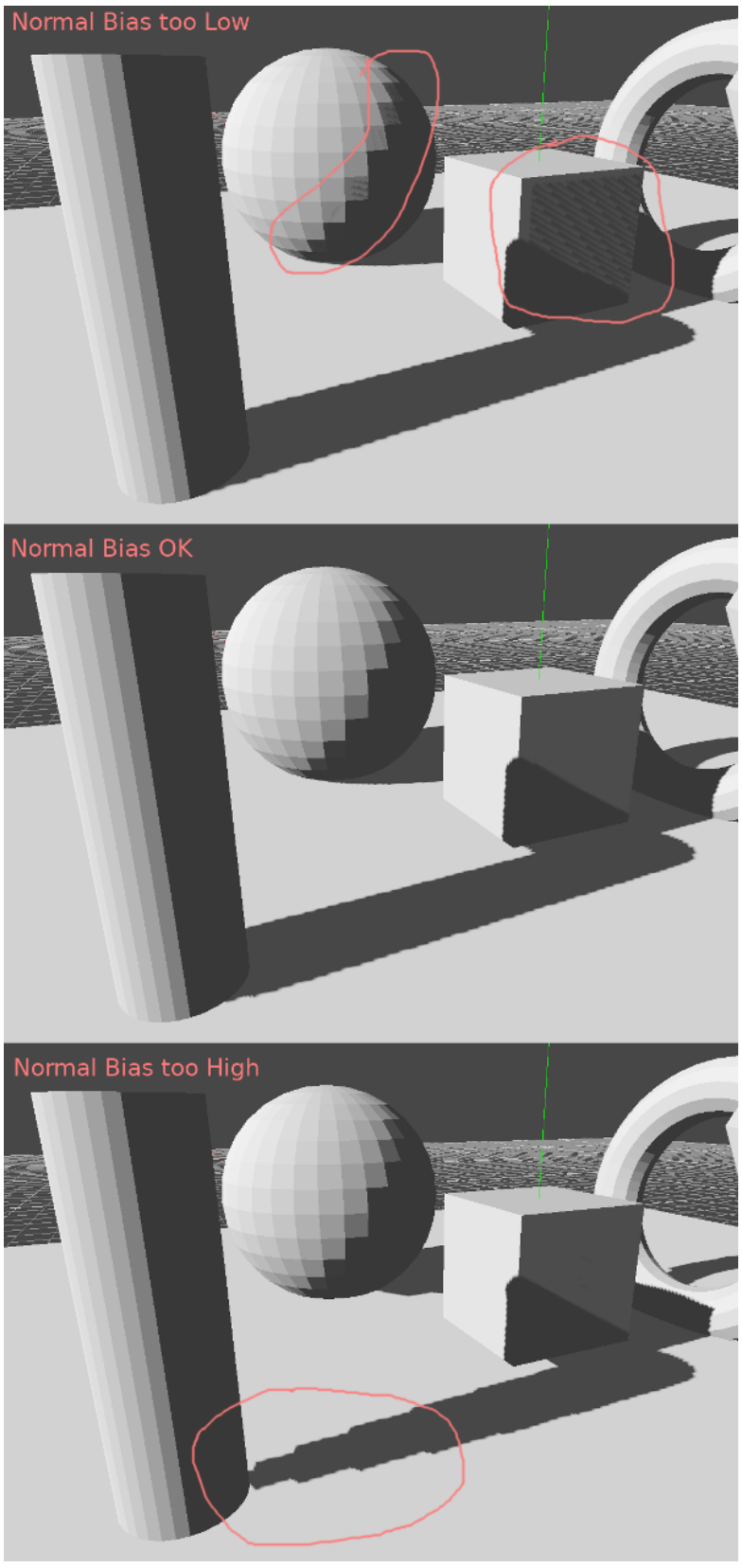
Los valores predeterminados generalmente funcionan bien, pero ajustar un poco la primera división suele ser común para dar más detalles a los objetos cercanos (como a un personaje de la escena).

Asegúrate de indicar un valor de **Max Distance** de acuerdo a las necesidades de la escena. Una distancia máxima baja obtendrá mejor calidad de sombras.

A veces, la transición entre una división y la siguiente puede verse mal. Para solucionar esto se puede activar la opción **Blend Splits**, que sacrifica detalles a cambio de transiciones más suaves:



El parámetro **Normal Bias** se puede usar para corregir casos especiales de *auto-sombreado* cuando los objetos son perpendiculares a la luz. El único inconveniente es que hace que la sombra sea un poco más delgada.


Normal Bias too Low
Normal Bias OK
Normal Bias too High

El parámetro **Bias Split Scale** puede controlar la inclinación adicional para las divisiones que están muy lejos. Si el *auto-sombreado* ocurre solo en las divisiones lejanas, este valor puede corregirlas.

Por último, **Depth Range** tiene dos ajustes:

- **Stable:** Mantiene la sombra estable mientras la cámara se mueve. Los *bloques* que aparecen en el contorno cuando están cerca de los bordes sombreados permanecen en su lugar. Este es el valor predeterminado (y generalmente deseado), pero reduce la resolución de sombra efectiva.
- **Optimized:** Intenta alcanzar la resolución máxima disponible en un momento dado. Esto puede provocar un efecto de *diente de sierra en movimiento* en los bordes sombreados, pero al mismo tiempo la sombra se ve más detallada.

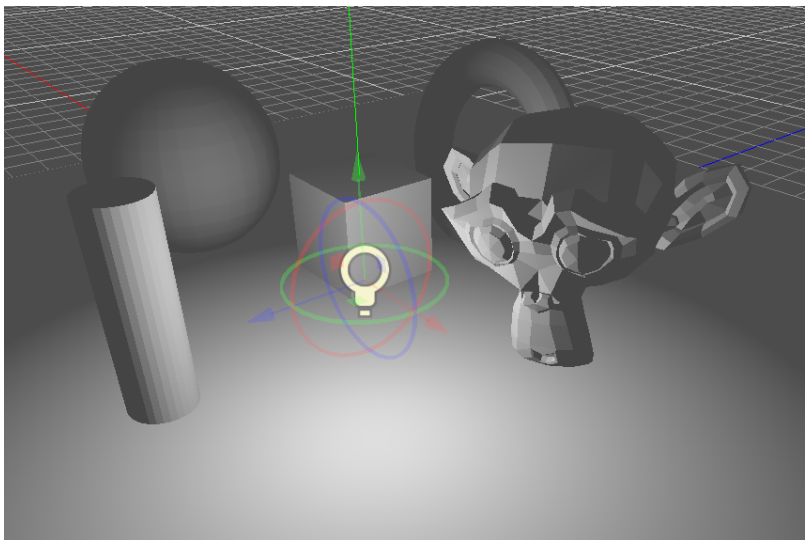
Y recuerda: A veces la mejor solución es experimentar qué configuración funciona mejor en tu escena.

El tamaño del mapa de sombras para luces direccionales puede cambiarse en la configuración del proyecto: **Project Settings** → **Rendering** → **Quality**

Aumentarlo puede solucionar problemas generados por el parámetro **Bias** pero reduce el rendimiento. Como ves, ajustar el mapeo de sombras es más un arte que una ciencia exacta.

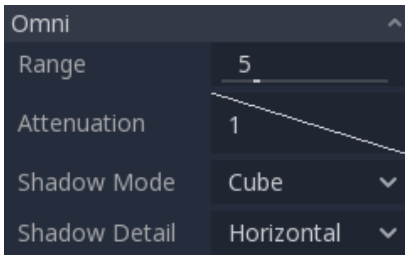
3.5 Omni light (Luz omnidireccional)

La **Omni light** es un punto que emite luz de forma esférica y en todas las direcciones hasta una distancia dada por un radio.

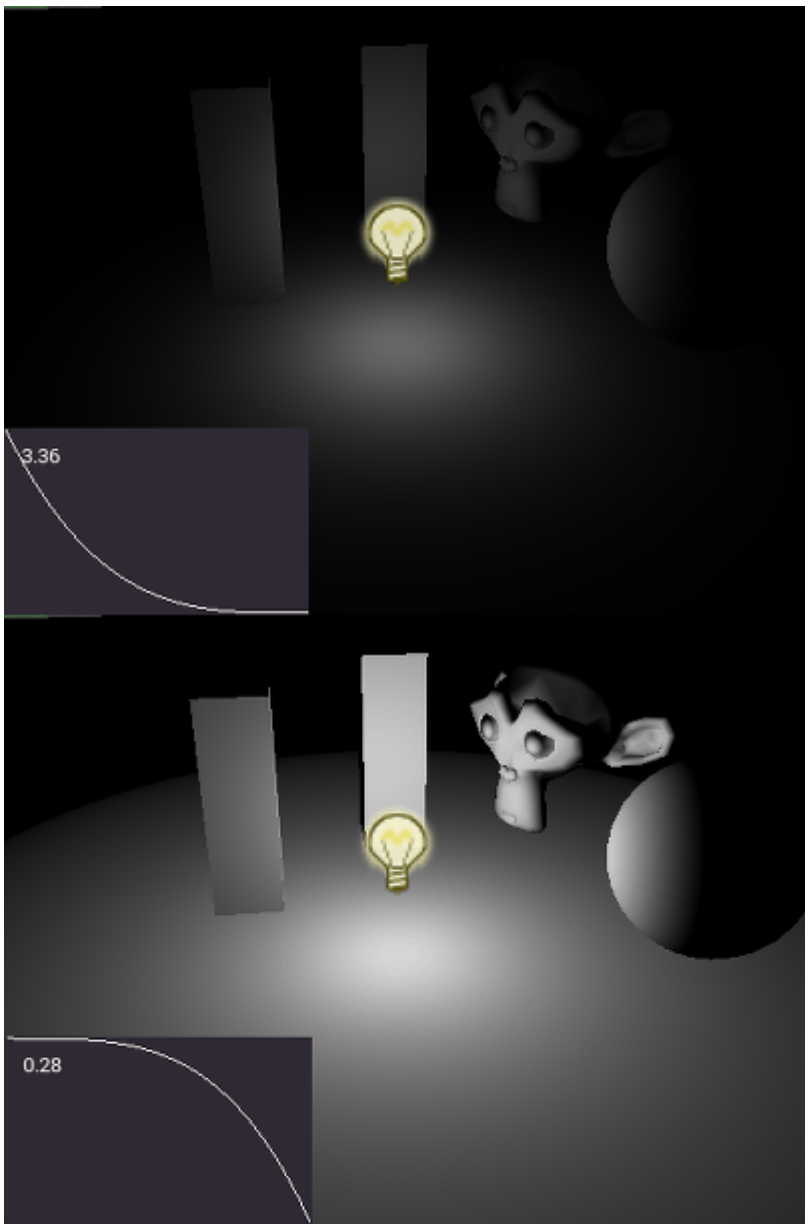


En la vida real, la atenuación de la luz es una función lineal inversa, lo que significa que las luces omnidireccionales no tienen realmente un *radio máximo*. Esto supone un problema, porque calcular un gran número de este tipo de luces se volvería costoso.

Para resolverlo, se introduce un rango (**Range**) junto con una función de atenuación (**Attenuation**).



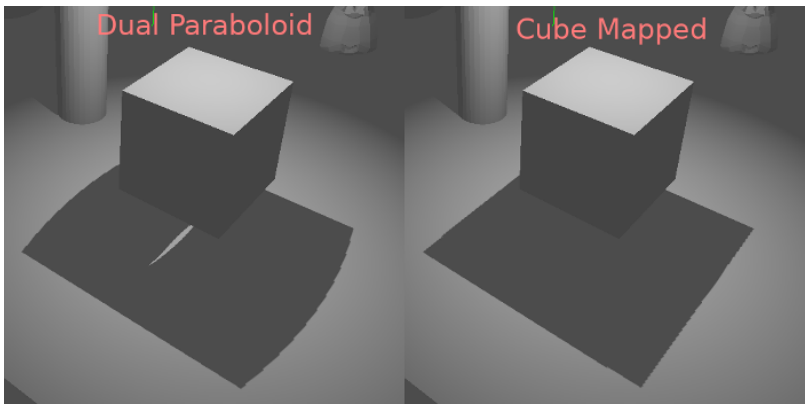
Estos dos parámetros permiten ajustar visualmente las luces para encontrar resultados estéticamente agradables.



Mapeo de sombras omnidireccional

El mapeo de sombras para una luz de tipo **Omni** es relativamente sencillo. Lo principal a considerar es el algoritmo que se utilizará para renderizarlo.

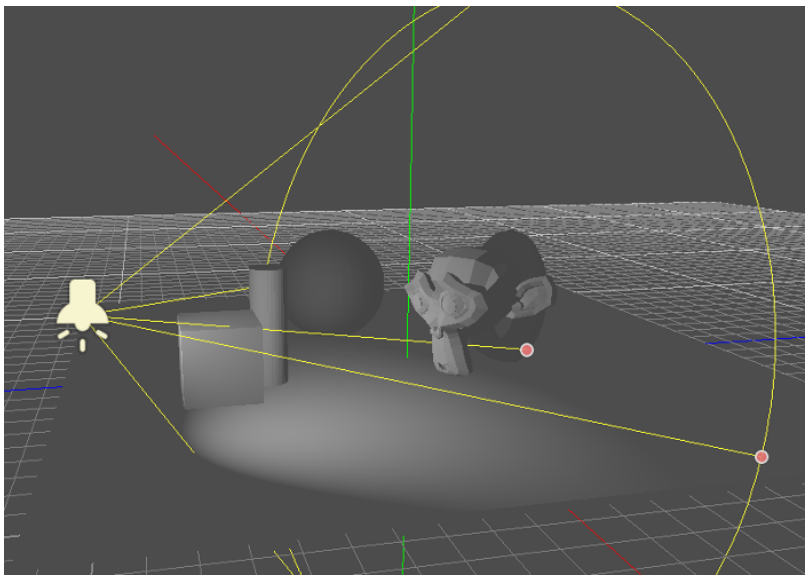
Las sombras omnidireccionales pueden ser renderizadas utilizando *Dual Paraboloid* o *Cube Mapped*. El primero se renderiza rápidamente, pero puede causar deformaciones, mientras que el segundo es más correcto, pero más costoso.



Si los objetos que se renderizan son en su mayoría irregulares, *Dual Paraboloid* suele ser suficiente. En cualquier caso, como estas sombras se almacenan en caché en un atlas de sombra (en clase te contarán más acerca de este tema), puede que no signifique una diferencia en el rendimiento para la mayoría de las escenas.

3.6 Spot light (luz focal)

Las luces focales (o puntuales) son similares a las omnidireccionales (*omni lights*), excepto porque emiten luz solamente en un cono. Son útiles para simular linternas, luces de coche, reflectores, focos, etc. Este tipo de luz además se atenúa en la dirección opuesta a la que apunta.



Las luces focales comparten los mismos parámetros **Range** y **Attenuation** con **OmniLight**, y añade dos más:

- **Angle:** El ángulo de apertura de la luz

- **Angle Attenuation:** La atenuación del cono, que ayuda a suavizar los bordes del cono.

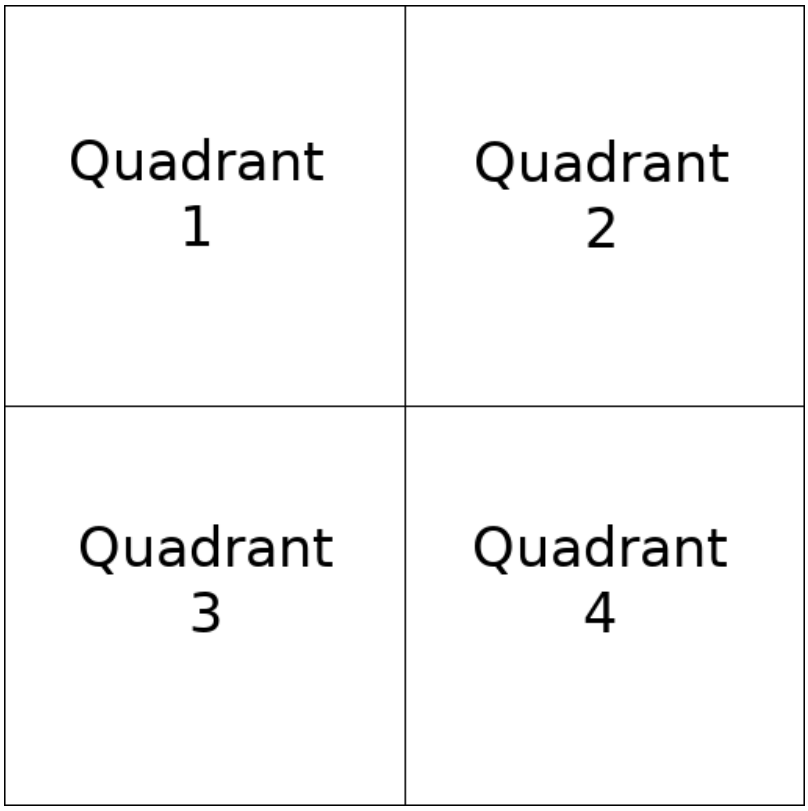
Mapeo de sombras focal

Las luces focales no necesitan ningún parámetro para el mapeo de sombras. Ten en cuenta que, para ángulos de apertura mayores de 80 grados, las sombras dejan de funcionar y resulta más conveniente utilizar luces omnidireccionales.

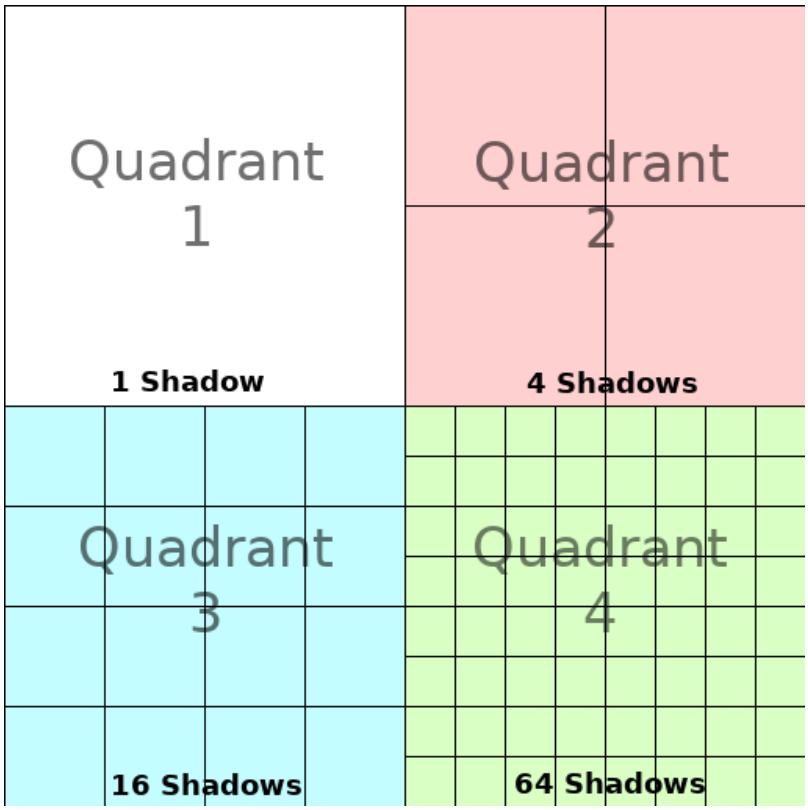
Shadow atlas (atlas de sombra)

A diferencia de las luces direccionales, que tienen su propia textura de sombra, las luces omnidireccionales y focales se asignan a espacios de un atlas de sombra. Este atlas se puede configurar en el propio proyecto: **Project Settings** → **Rendering** → **Quality** → **Shadow Atlas**.

La resolución se aplica a todo el Atlas de las Sombras. Este atlas se divide en cuatro cuadrantes:



Cada cuadrante se puede subdividir para asignar cualquier número de mapas de sombras; la subdivisión mostrada en la imagen es la predeterminada:



El tamaño de mapa de sombras más grande (cuando no se utiliza ninguna subdivisión) representa una luz del tamaño de la pantalla (o mayor). Las subdivisiones (mapas más pequeños) representan sombras para las luces que están más lejos de la vista y proporcionalmente más pequeñas.

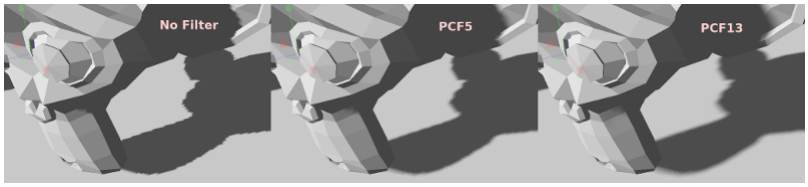
En Cada fotograma, se realiza el siguiente procedimiento para todas las luces:

- Se comprueba si la luz está en un espacio del tamaño correcto. Si no es así, se vuelve a renderizar y se mueve a un espacio más grande o más pequeño.
- Se comprueba si ha cambiado algún objeto que afecte al mapa de sombras. Si esto sucedió, se renderiza de nuevo la luz.
- Si no ha ocurrido nada de lo anterior, no se hace nada, y la sombra queda intacta.

Si los espacios de un cuadrante están llenos, las luces son devueltas a espacios más pequeños, dependiendo del tamaño y la distancia.

4. Calidad del filtro de sombras

La calidad del filtro de las sombras se puede ajustar. Se puede cambiar en el proyecto: **Project Settings** → **Rendering** → **Quality** → **Shadows**. Godot soporta los filtros PCF5, PCF13, y también permite no filtrar. Ten en cuenta que estos filtros no solo afectan al bloqueo del contorno de la sombra, sino que también consumen recursos hardware.



Puedes encontrar más detalles en la *documentación oficial*.

Ejercicios de iluminación

Ejercicio calificable P3. B.

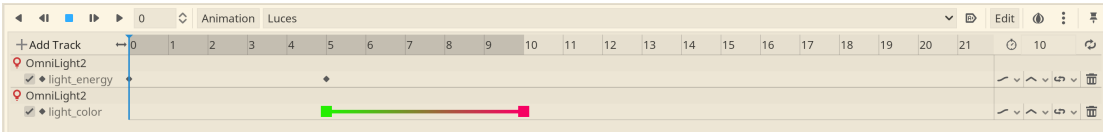
> Añade luces a la escena. Decide cuando usar luces con *sombras arrojadas* y cuando no, teniendo en cuenta factores estéticos y de realismo.

> En este ejercicio será muy relevante la justificación empleada para cada una de las luces y el tipo (direccional, posicional o focal), así que apunta (e indica a tu profesor) cada una de las decisiones tomadas.

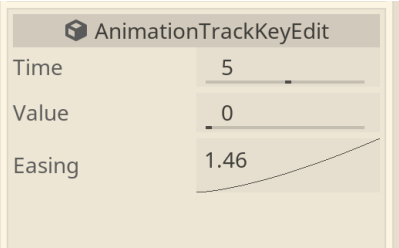
5. Animación de luces y otros elementos

Revisa la práctica anterior, comentábamos entonces que puedes animar cualquier parámetro. Una forma fácil de hacerlo es crear una luz y como su hijo un nodo **AnimationPlayer**. Recuerda como hacer una animación, esta vez puedes añadir una pista (**Track**) de tipo parámetros (en lugar de transformaciones 3D).

Puedes añadir cualquier parámetro que quieras, incluidos colores.



Fíjate que cada vez que insertes un **keyframe**, aparecerá, a la derecha arriba, un editor de valores que te permitirá cambiar con comodidad cada uno de los valores y su función de interpolación.



Ejercicio calificable P3. C.

> Añade una luz que tilile cada vez que pulses una determinada tecla, por ejemplo la **L**. Para ello crea la animación correspondiente y ajusta la interpolación.

> Modifica la animación anterior para que, después de tililar, se atenúe al final un poco y vuelva a encenderse.

Cámaras

Verás más en teoría sobre cámaras y los tipos. Para la práctica tenemos tres parámetros importantes:

- **FOV:** Es el ángulo de apertura de la cámara. Los humanos solemos tener unos 65º de apertura.
- **Tipo de proyección:** Puede ser en perspectiva (que es lo normal) o en ortogonal (en la que los objetos no se deforman por la distancia).
- **Planos de corte:** Son dos planos, el cercano (**Near**) y el lejano (**Far**) y representan la distancia que marca la zona visible de la cámara.

Sin embargo, es especialmente relevante el hecho de que cada cámara puede tener un entorno diferente, con lo que podemos aplicar interesantes efectos en las escenas. Además, podemos saltar de una cámara a otra cambiando el parámetro **Current** mediante código GDScript.

En realidad veremos más de las cámaras en la siguiente práctica, ya que lo que realmente nos interesa aquí es su aspecto práctico y como hacerlas *interactivas*.