

# Desenvolvimento com Motores de Jogos II

## Aula 08 - Iluminação

# Apresentação

---

Olá, pessoal! Chegamos na oitava aula da nossa disciplina. Como não somos mais um Newbie, e aumentamos cada vez o nosso XP com as experiências das aulas anteriores, então falaremos de um aspecto muito importante em qualquer jogo: **Iluminação**. Os objetos 3D (Meshes) e Scripts certamente definem as mecânicas do seu jogo, entretanto a iluminação define como a cor se propaga na cena além do clima de cada parte do seu ambiente. Assim, vamos aprender quais são os tipos de iluminação que o Unity oferece, como adicioná-las na sua cena e ainda entender melhor suas configurações.



## Objetivos

Ao final desta aula, você deverá ser capaz de:

- Conhecer os tipos de iluminação do Unity;
- Adicionar iluminação nas cenas;
- Compreender as configurações do sistema de iluminação do Unity.

# Iluminação

---

Para poder determinar o quão claro/escuro uma área de um objeto é, além de suas sombras, o Unity precisa conhecer a intensidade, direção e a cor das fontes de luz que atuam sobre esse objeto. Essa informação é originada de objetos chamados **Lights** (ou luzes), presentes em sua cena. Veja na **Figura 1** um exemplo de fonte de luz (Light source) afetando uma esfera.

**Figura 01** - Fonte de luz afetando uma esfera.



Cada tipo de luz emite raios de formas diferentes e esses raios têm intensidade e cores definidas pelas suas propriedades.

## Fontes de Luz

---

No Unity, a iluminação de uma cena é fornecida por objetos de luz que podem ser objetos com um componente do tipo Light associado ou material com emissão de luz (emissive material).

Por enquanto, estudaremos as fontes de luz mais comuns, providas pelos objetos de luz padrão do Unity.

## Tipos de Luz

Os tipos de luz padrão do Unity são:

### ▼ **Point Light**

Emite luz do seu centro para todas as direções.

### ▼ **Spot Light**

Emite luz do seu centro em uma determinada direção no formato de um cone.

### ▼ **Directional Light**

Emite luz em uma direção, porém não tem uma origem definida, ou seja, comporta-se como raios paralelos de luz de um ponto distante passando por toda a cena.

### ▼ **Area Light**

Emite luz a partir de uma área retangular de forma uniforme.

### ▼ **Emissive material**

Emite luz a partir da superfície de um objeto com um material especialmente configurado para emitir luz.

### ▼ **Ambient Light**

É a luz presente em toda a cena, não tendo uma origem específica.



Vocês viram que são vários tipos de luz. Mas nada de preocupação, iremos estudar individualmente cada um dos tipos e você vai aprender a escolher qual usar na sua cena para dar o aspecto que desejar ao seu jogo.

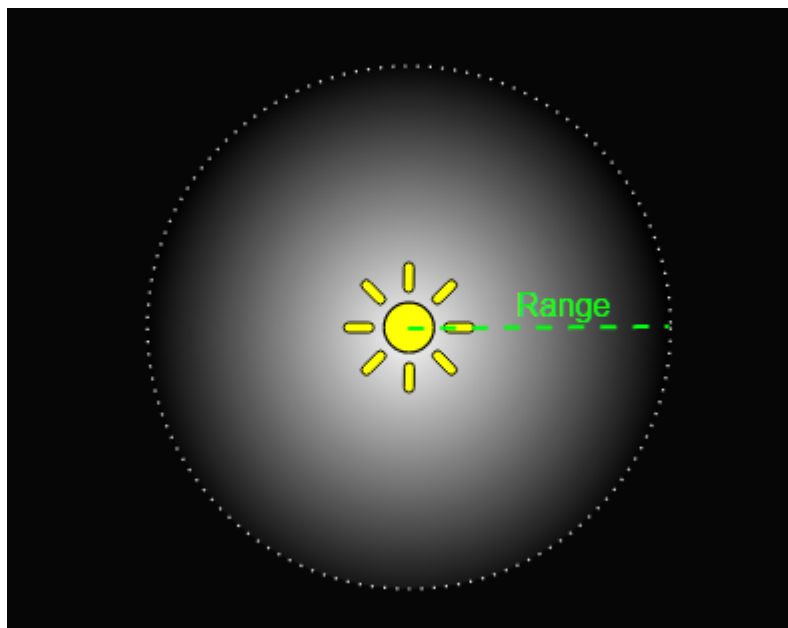
## Point Lights

---

**Point Light** é uma fonte de luz que se localiza em um determinado ponto no espaço 3D e emite luz em todas as direções ao seu redor de modo uniforme. Sua intensidade vai do seu valor máximo configurado a partir do seu centro e vai diminuindo com a distância até alcançar o valor zero em uma determinada distância máxima configurada.

Vale lembrar que a intensidade de um raio de luz emitido por uma **Point Light** diminui de maneira não linear, de acordo com o quadrado da distância do seu centro, de forma similar ao que ocorre no mundo real. Veja na **Figura 2** a intensidade da luz branca diminuindo com a distância do centro até zerar completamente no seu alcance máximo (Range).

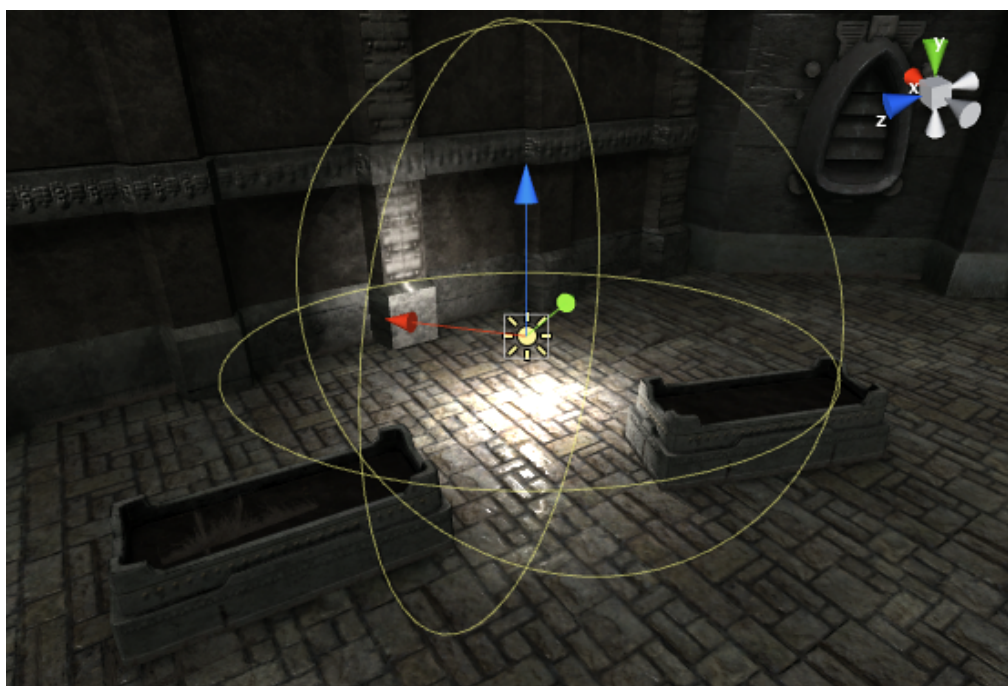
**Figura 02** - Point Light com um alcance máximo (Range).



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://docs.unity3d.com/Manual/Lighting.html>. Acesso em: 25 de abril de 2017.

As **Point Lights** podem ser utilizadas como uma fonte de iluminação similar à de lâmpadas tradicionais. Veja na **Figura 3** um exemplo de uma cena afetada por uma **Point Light** localizada no seu centro.

**Figura 03** - Cena afetada por uma Point Light.



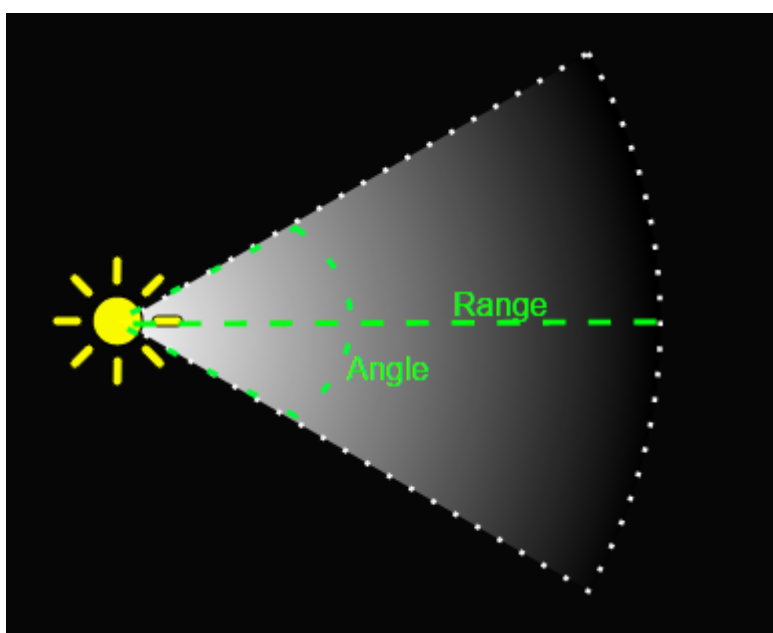
**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://docs.unity3d.com/Manual/Lighting.html>. Acesso em: 25 de abril de 2017.

# Spot Light

---

**Spot Light** é uma fonte de luz que emite raios a partir do seu centro, porém em uma direção específica e também com um ângulo de ação, originando uma região de emissão no formato de um cone. A intensidade dos raios de luz emitidas por uma **Spot Light** também diminui com a distância até um limite máximo (Range). Veja a **Figura 4** representando a região de ação de uma **Spot Light**.

**Figura 04** - Região de ação de uma Spot Light em uma direção, ângulo e até um limite máximo (Range).

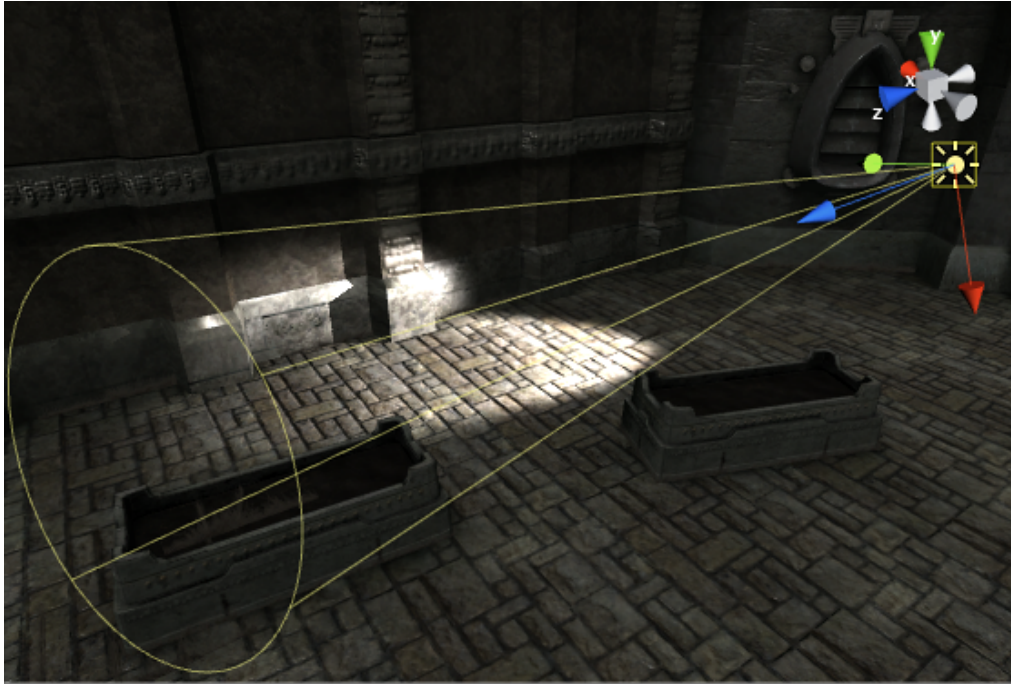


**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://docs.unity3d.com/Manual/Lighting.html>. Acesso em: 25 de abril de 2017.

Spot Light pode ser utilizada para representar fontes de luz similares a holofotes direcionais, lanternas, faróis de carros, etc. Veja na **Figura 5** um exemplo de uma cena iluminada por uma Spot Light.



**Figura 05** - Cena iluminada por uma Spot Light.



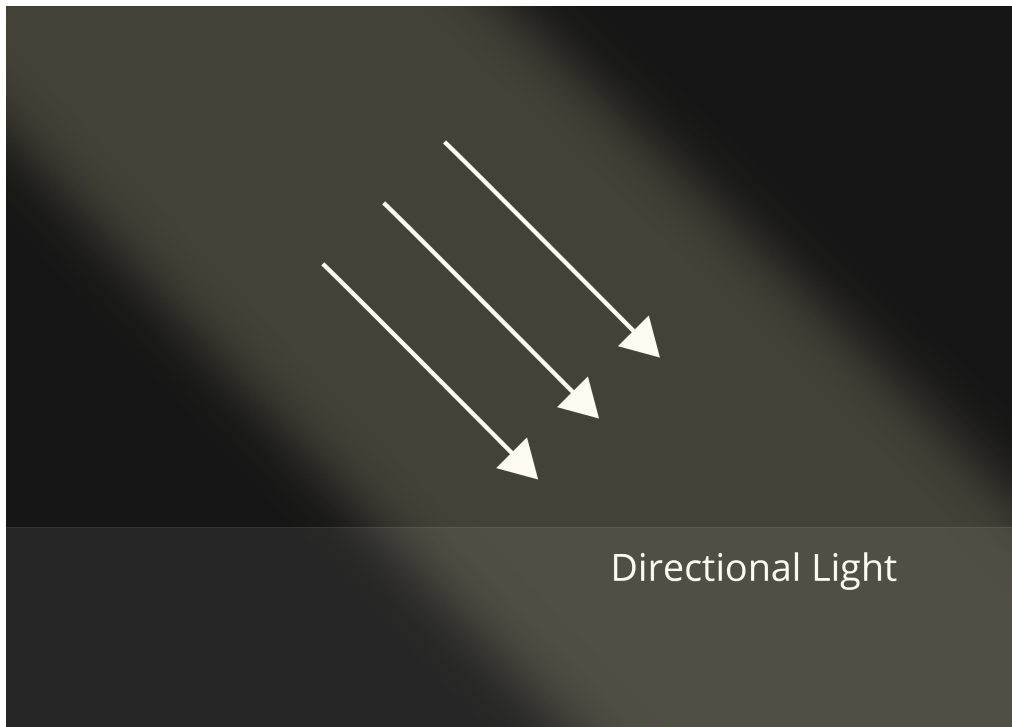
**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://docs.unity3d.com/Manual/Lighting.html>. Acesso em: 25 de abril de 2017.

## Directional Light

---

Uma **Directional Light** é um tipo de fonte de luz que emite raios em uma determinada direção, entretanto não tem nenhuma informação sobre a origem, limite máximo da região de atuação nem o ângulo de atuação desses raios. Isso significa que uma **Directional Light** emite raios paralelos originados de um ponto imaginário no infinito. Veja na **Figura 6** uma representação dos raios de luz de uma **Directional Light**.

**Figura 06** - Raios de luz de uma Directional Light, somente com direção.



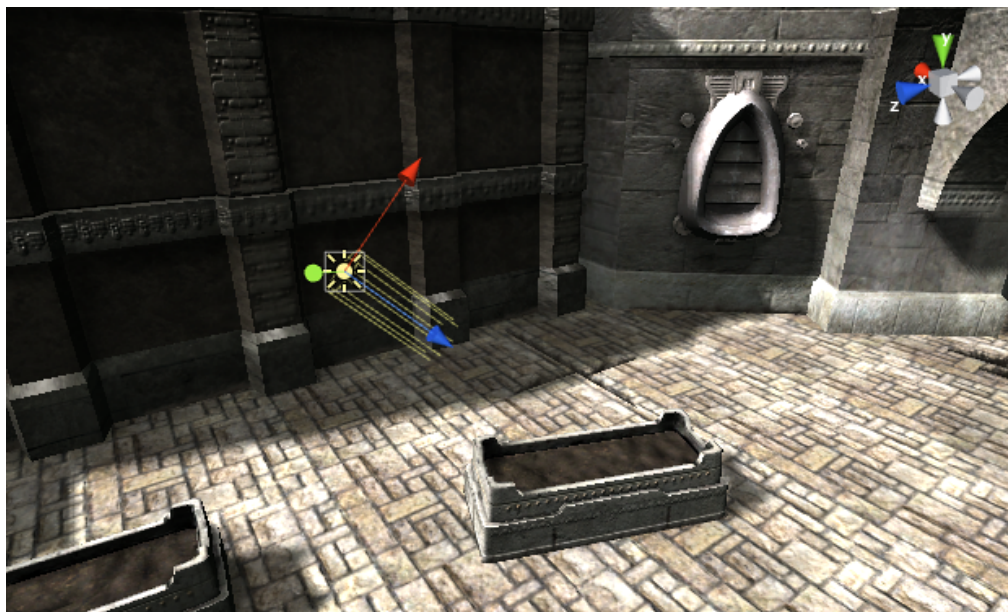
Imagine que uma **Directional Light** seja similar aos raios do Sol.



Sim, é verdade que o Sol está muito longe da Terra, mas não está no infinito, claro. Entretanto, para efeitos práticos, consideramos que os raios do Sol atinjam a Terra de forma paralela, já que as diferenças de seus ângulos em relação ao

tamanho da terra são muito pequenas. Desse modo, uma **Directional Light** é uma excelente forma de representar o Sol ou a Lua, inclusive. Veja na **Figura 7** um exemplo de uma cena afetada por uma **Directional Light**.

**Figura 07** - Cena afetada por uma Directional Light.



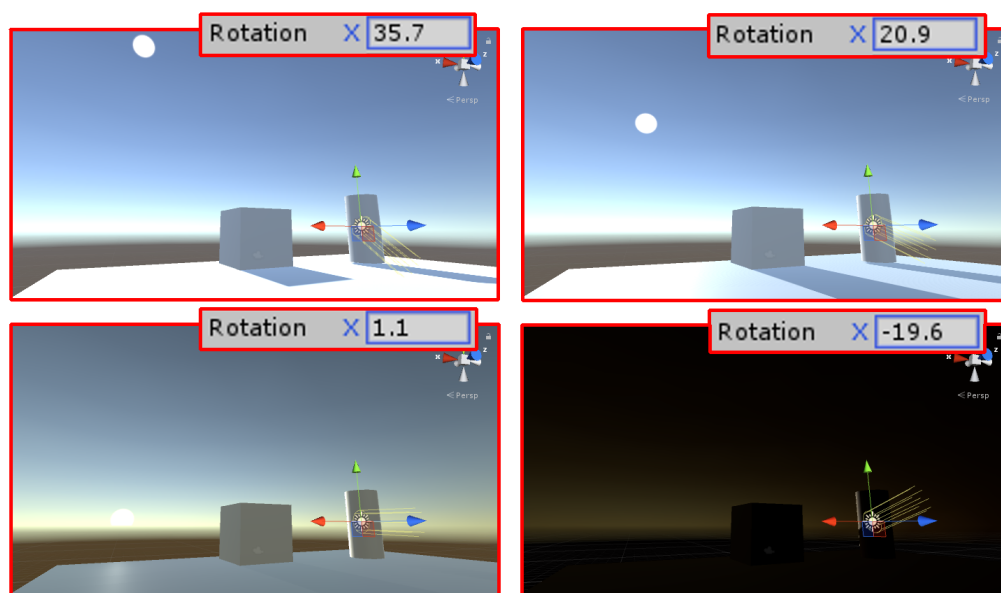
**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://docs.unity3d.com/Manual/Lighting.html>. Acesso em: 25 de abril de 2017.

Um **GameObject Directional Light** já vem criado em uma nova cena no Unity para você, e ele é utilizado pelo sistema de Céu procedural, ou seja, a primeira **Directional Light** apresentada na sua cena é utilizada para determinar a cor do céu e também a posição do Sol na cena.

Se você posicionar visão da Scene View a apontar para cima poderá ver o Sol gerado pelo sistema de Céu procedural (Procedural sky system). Caso mude a rotação da Directional Light da sua cena, estará alterando a sua direção de atuação, então o Sol da cena também vai mudar de posição de forma adequada. Mais adiante aprenderemos a modificar o Sky system para criar ambientes únicos no seu jogo.

Veja na **Figura 8** a mudança da posição do Sol quando se altera a rotação da **Directional Light** no eixo X. Repare que se o Sol ficar abaixo do horizonte a cena fica com um aspecto noturno. Legal, não é mesmo?

**Figura 08** - Mudança da posição do Sol da cena de acordo com a rotação (direção) da Directional Light



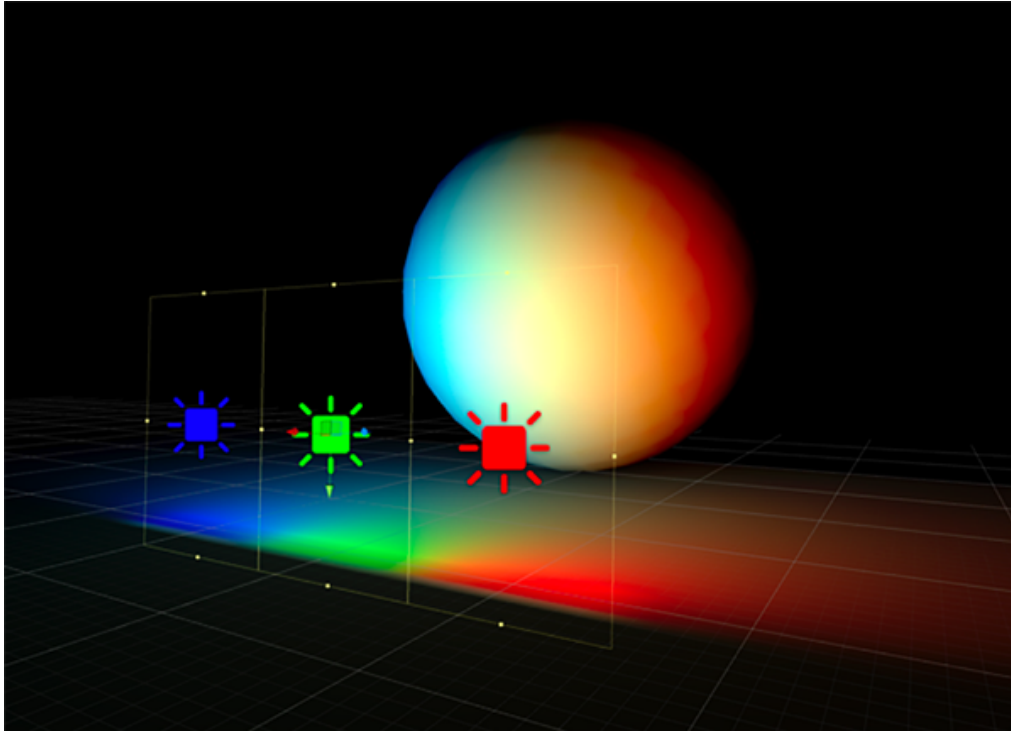
**Fonte:** Elaborado pelo Autor (2017).

## Area Lights

Uma **Area Light** é definida por um retângulo no espaço e emite luz a partir da superfície desse retângulo uniformemente, mas somente de um lado do retângulo. Da mesma forma que os outros tipos de luz, a **Area Light** emite raios de luz que vão diminuindo de intensidade com o quadrado da distância, entretanto você não pode controlar a distância máxima de sua atuação. Os cálculos dos raios de luz do Area Map são muito pesados e ele só afeta o ambiente se sua iluminação for pré-calculada em um processo chamado de **Baking**. Dizemos então que a **Area Light** precisa ser Baked (algo como cozinhadas, ou pré-processadas) no mapa de iluminação do Unity, o **Lightmap**, que é uma forma de guardar informações na memória sobre o efeito das luzes na sua cena para que o seu jogo não precise recalcular todos os efeitos de todos os raios de luz a cada quadro (Frame) do jogo. Por ser um tema mais avançado, você poderá aprofundá-lo na documentação oficial do Unity. Entretanto, para manter os exemplos mais simplificados, não usaremos **Area Light**.

Veja na **Figura 9** uma representação da atuação de três **Area Lights** (uma azul, uma verde e uma vermelha) em uma cena com uma esfera.

**Figura 09** - Três Area Lights atuando em uma esfera.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://docs.unity3d.com/Manual/Lighting.html>. Acesso em: 25 de abril de 2017.

## Emissive Materials

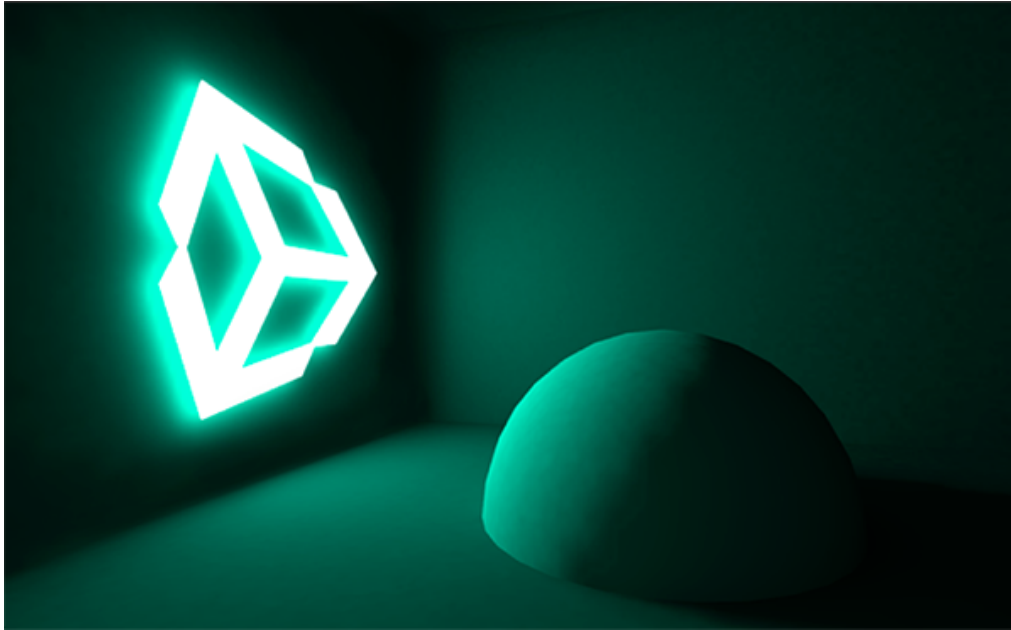
---

Assim como a **Area Lights**, o **Emissive materials** emite luz da sua superfície, afetando os raios de luz atuais de uma cena, podendo modificar a cor e a intensidade deles em tempo real durante o jogo.

Quando você cria um novo Material ele vem com um Shader padrão configurado chamado **Standard Shader**. Shaders são como programas especiais que são responsáveis por determinar como se comporta visualmente a superfície de um objeto. O **Standard Shader** nada mais é do que o Shader padrão e permite a realização de ajustes nas suas propriedades, como cor, textura, etc. Você pode criar um material com o shader padrão e configurar a propriedade “Emission” para escolher uma emissão de luz. Se escolher a cor preta, significa que o Material não emitirá nenhuma luz.

Veja na **Figura 10** um exemplo de um **Emissive Material** em um GameObject afetando a iluminação de uma cena.

**Figura 10** - Emissive material em ação.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://docs.unity3d.com/Manual/Lighting.html>. Acesso em: 25 de abril de 2017.

Como **Emissive materials** é um tópico avançado na criação de materiais, assim como a **Area Lights**, você pode conhecer mais sobre esse assunto consultando a documentação oficial do Unity.

## Ambient Light

---

A **Ambient Light** está presente em toda a cena e não é um objeto com determinada origem e direção de emissão, é algo que contribui de forma importante para o aspecto geral do jogo, pois você pode escolher um aspecto mais realista até algo mais parecido com um desenho feito à mão. Pode ser muito útil se você deseja modificar o brilho geral da sua cena sem modificar as luzes individualmente.

As propriedades da Ambient Light são configuráveis através da janela Lightning, a qual pode ser aberta usando a opção no menu Window -> Lightning.

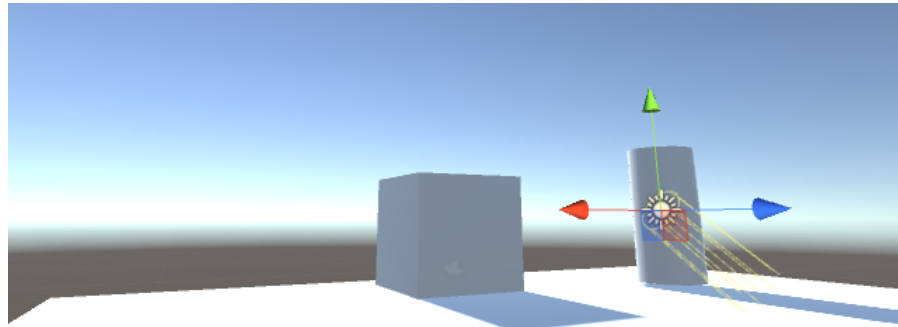
Crie um novo projeto no Unity, pois realizaremos alguns testes de iluminação na prática. Realize a atividade 1 antes de continuar.



## Atividade 01

1. Crie um novo projeto no Unity chamado Iluminação e adicione no centro um Cube com o Scale em X=25, Y=1, Z=25.
2. Adicione um segundo Cube e um Cylinder, ambos com Scale X=3, Y=3, Z=3 e posicione-os acima do primeiro Cube como na figura abaixo.

**Figura 11** - Cenário de referência para a atividade.



**Fonte:** Elaborado pelo autor (2017)

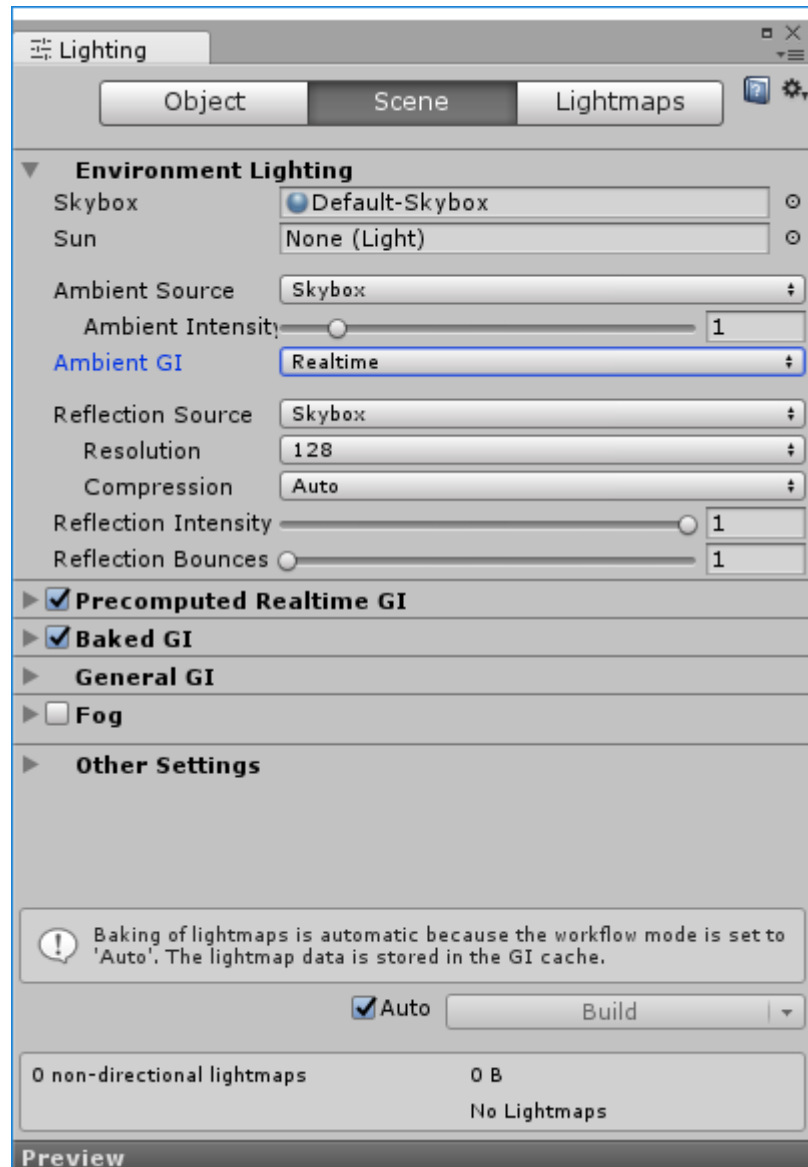
3. Altere a rotação da sua Directional Light e descreva o que aconteceu com as sobras dos objetos na cena, assim como o aspecto do céu.
4. Através do menu GameObject -> Light, adicione uma Point Light e uma Spot Light na sua cena e descreva como ela afeta os objetos adicionados.

## Iluminação Global

---

Já vimos vários tipos de luzes disponíveis no Unity, entretanto um assunto que exige bastante atenção é iluminação global. No Unity, praticamente todas as configurações de iluminação global estão disponíveis no **painel Lightning** (menu Window -> Lightning). Veja na **Figura 12**.

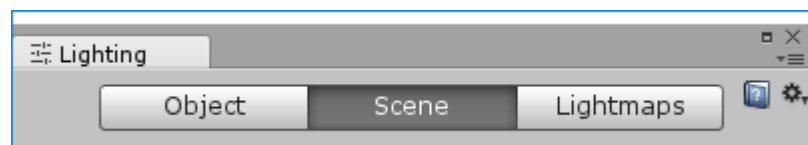
**Figura 12** - Janela Lightning.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

A **janela Lightning** é o local em que você pode configurar diversas propriedades de iluminação individual de objetos, configurações de iluminação global da cena e Lightmaps. Cada uma dessas opções pode ser acessada ao clicar em uma das três abas que se encontram no início da **janela Lightning**. Veja na **Figura 13**.

**Figura 13** - Abas principais da janela Lightning.





Segue uma descrição do que cada aba representa:

- **Object:** Permite que você escolha um objeto no Hierarchy (clcando nele) e possa modificar as configurações de iluminação global, por exemplo informando se esse objeto participará do cálculo do Lightmap ou não.
- **Scene:** Configurações de iluminação global de toda a cena e não de objetos individuais (exploraremos essas configurações nesta aula).
- **Lightmaps:** Permite que você crie um arquivo de Lightmap (ou mapa de iluminação). Lightmaps são maneiras de embutir nas texturas do jogo o efeito que as luzes causam nelas, permitindo que esses efeitos não precisem ser recalculados a todo momento, melhorando a performance do seu jogo.

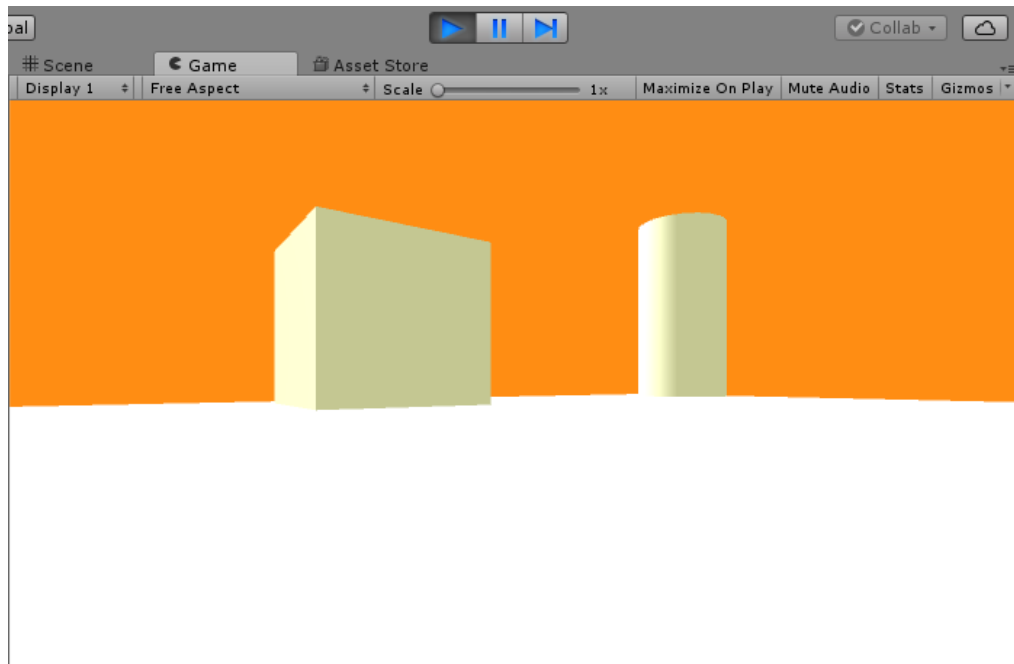
Quanto ao Lightmaps, por ser um assunto avançado de iluminação, você não precisa se preocupar com eles nesse momento. Pela configuração padrão os Lightmaps no Unity são calculados automaticamente, mas se desejar conhecer mais sobre o assunto consulte a documentação oficial do Unity.

Vamos explorar as configurações globais de iluminação de toda a cena, então clique na aba “Scene” da janela Lightning, veja o exemplo na Figura 11. Essa aba é dividida em várias seções: Environment Lightning, Precomputed Realtime GI, Baked GI, General GI, Fog e Other Settings. Trabalharemos, por enquanto, na seção “**Environment Lightning**”, ou “Iluminação do ambiente” em português.

A seção **Environment Lightning** é composta por uma série de propriedades que modificam a aparência da iluminação global da cena. Nesta aula tentaremos criar uma iluminação única, bem diferente da iluminação mais focada no realismo, que é a configuração padrão de uma cena no Unity.

Digamos que você queira modificar sua cena para que a iluminação global seja mais em tom amarelo e que o céu seja uma cor sólida laranja, como na **Figura 14**.

**Figura 14** - Exemplo de cena com iluminação global modificada.

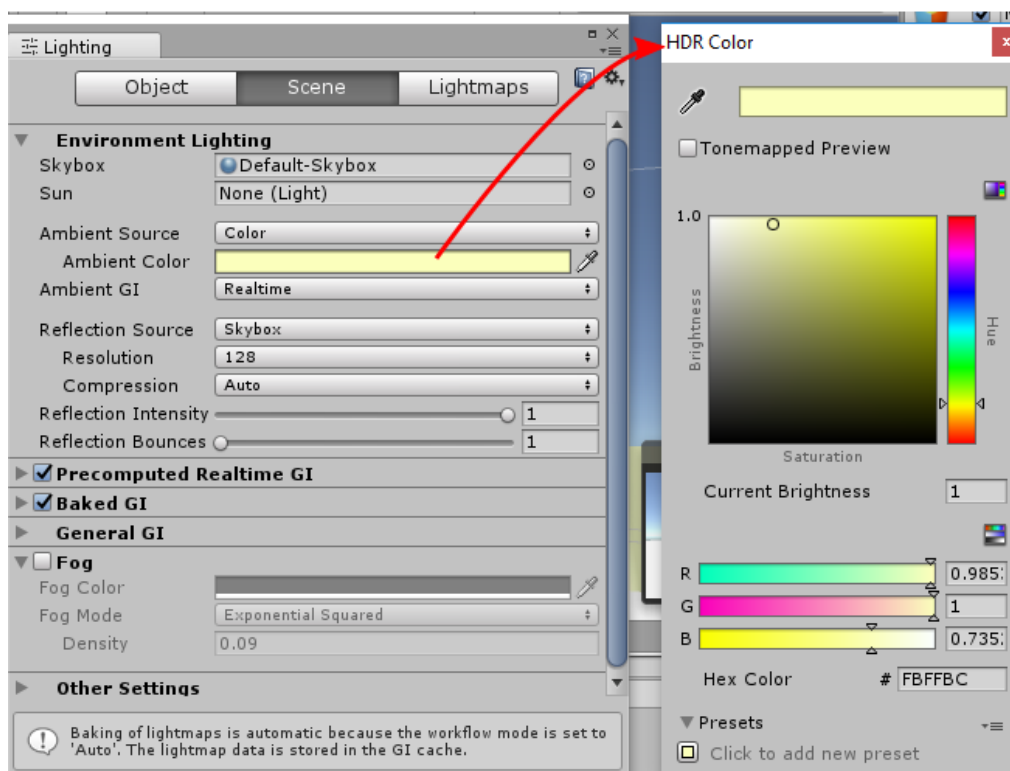


**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

Reparem que o céu sumiu , pois a iluminação global faz com que o cubo e o cilindro fiquem amarelados, mesmo sendo eles da cor padrão branca.

Para fazer essa mudança precisamos inicialmente abrir **a janela Lightning e na aba Scene mudar a propriedade Ambient Source de Skybox para Color**. Isso fará com que uma nova propriedade surja, chamada “Ambient Color”, a qual que deve ser modificada para uma cor amarelo claro, conforme a **Figura 15**.

**Figura 15** - Alteração da luz ambiente para uma cor amarelo claro.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

Essa mudança faz com que a fonte de luz global da cena (Ambient Source) não seja mais um reflexo da luz do céu (Skybox) e sim uma cor sólida que você escolheu (Ambient Color).

Ao realizar essa mudança, pode fechar a **janela Lighting**. Clique agora na sua Main Camera e observe suas propriedades no Inspector, conforme a **Figura 16**.

**Figura 16** - Propriedades padrão da Main Camera.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

A câmera principal do jogo (nesse caso o GameObject Main Camera) é a responsável por criar a imagem apresentada na tela do jogo, utilizando para isso todos os objetos da cena e também desenhando o fundo da cena do jogo, que atualmente é o nosso céu (Skybox).

Como estamos tentando substituir o céu do fundo do jogo por uma cor sólida, então é preciso configurar a câmera para que ela não mostre o Skybox e sim uma cor sólida.

Observe na **Figura 15** que existem duas propriedades importantes chamadas de **Clear Flags** e **Background**. O **Clear Flags**, pela configuração padrão, vem configurada como Skybox, informando que a câmera vai obter suas informações do Skybox, e **Background** é a cor de fundo padrão da câmera.

Modifique **Clear Flags** para o valor “Solid Color” e **Background** para a cor que você desejar, por exemplo laranja. Essa mudança está ilustrada na **Figura 17**.

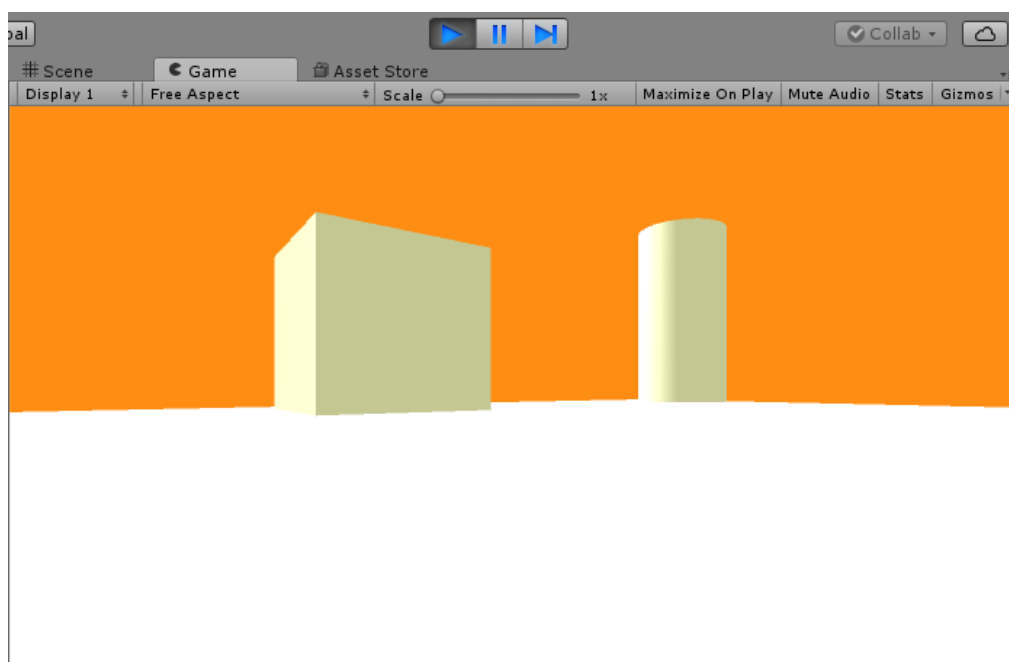
**Figura 17** - Mudança das propriedades de fundo da câmera principal.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

Posicione sua Main Camera onde desejar, execute o jogo e verá que temos o resultado esperado, conforme a **Figura 18**.

**Figura 18** - Exemplo de jogo com cor de fundo sólida laranja e iluminação padrão das luzes amarelada.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

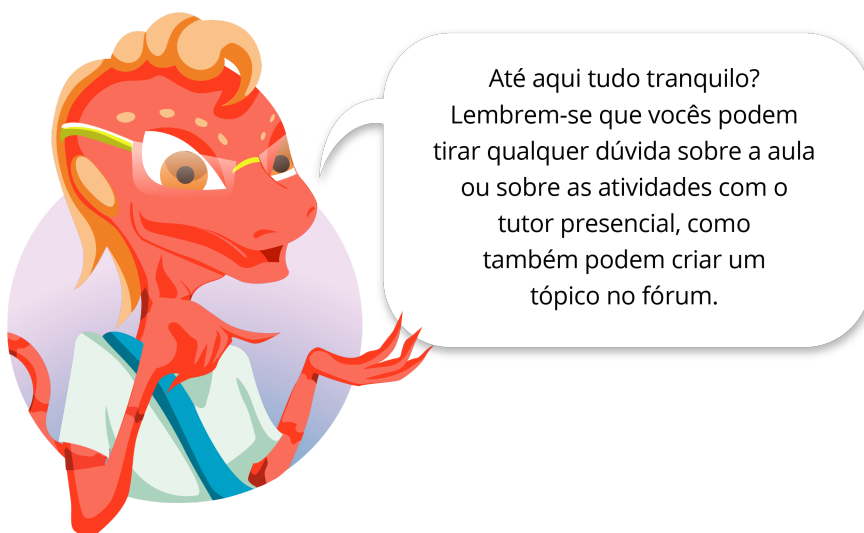
É importante saber que a mudança da Main Camera só afetará a visão dessa câmera, ou seja, quando você executar o jogo. Enquanto está no editor você verá um Skybox padrão, mas isso não é um problema.

Bem diferente, não é mesmo? Faça você mesmo as suas modificações nas configurações de iluminação global do Unity para explorar mais opções.

## Atividade 02

---

1. Altere a cor de Background da Main Camera para uma cor vermelha. Execute o jogo e veja o que acontece.
2. Adicione na Main Camera um script C# novo chamado de "MudarCor". Edite o script no MonoDevelop e dentro do método Start() adicione a seguinte linha: `Camera.main.backgroundColor = Color.green;` Em seguida, salve o script, volte ao Unity e execute o jogo. Baseado na sua observação, o que esse script fez no seu jogo?



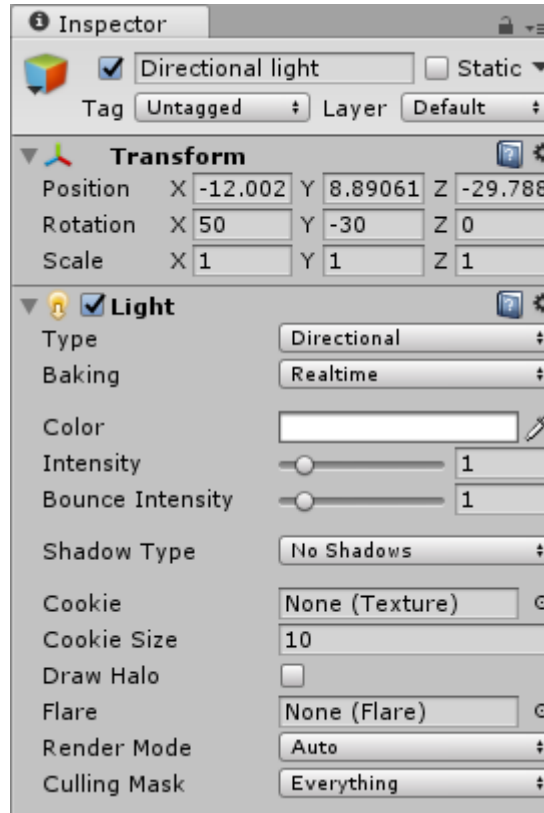
## Adicionando Luzes em uma Cena

---


Se você está seguindo a aula passo a passo, já deve ter criado um projeto novo para as atividades, então vamos continuar com esse projeto, entretanto configure novamente a Main Camera para exibir o Skybox no lugar da cor sólida laranja, pois isso facilitará a visualização das nossas próximas mudanças. Clique na Main Camera e modifique a propriedade **Clear Flags** para o valor Skybox.

Na nossa cena já existe uma luz criada do tipo **Directional Light**. Localize-a no Hierarchy e clique para mostrar as suas propriedades no Inspector, veja na **Figura 19**.

**Figura 19** - Propriedades da Directional Light.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

Observe que esse objeto é somente um GameObject comum com um componente do tipo “Light”  conforme a figura associada. No Unity, uma luz é basicamente isso, um GameObject com um componente.

São várias as propriedades das luzes, vamos explorar algumas delas:

- **Type:** Tipo da luz, pode ser Directional, Point, Spot ou Area. Já vimos o que cada um deles significa. Quando você muda o tipo da luz as outras propriedades disponíveis também mudam, já que cada tipo apresenta uma propriedade diferente.
- **Baking:** Podem ter os valores Realtime, Backed e Mixed:

- **Realtime:** Efeitos da luz na cena são calculados em tempo real. É o que usaremos.
- **Baked:** Efeitos calculados e salvos no Lightmap, como explicado.
- **Mixed:** Igual ao Baked, mas também estará presente em tempo real para iluminar diretamente objetos que não são estáticos e, portanto, não participam do processo de Baking do Lightmap.

As demais propriedades de uma luz dependem do tipo escolhido. Vamos ver algumas propriedades importantes da **Directional Light**:

- **Color:** Cor da luz emitida.
- **Intensity:** Intensidade da luz.
- **Bounce Intensity:** Intensidade da luz indireta (os raios de luz que batem em um objeto e refletem em outro).
- **Shadow Type:** Tipo da sombra que essa luz gera. Pode ser Soft Shadow (sombras suaves), Hard Shadow (sombras mais definidas) e No Shadows (sem sombras).

Mantenha a luz padrão da cena como do tipo **Directional**.

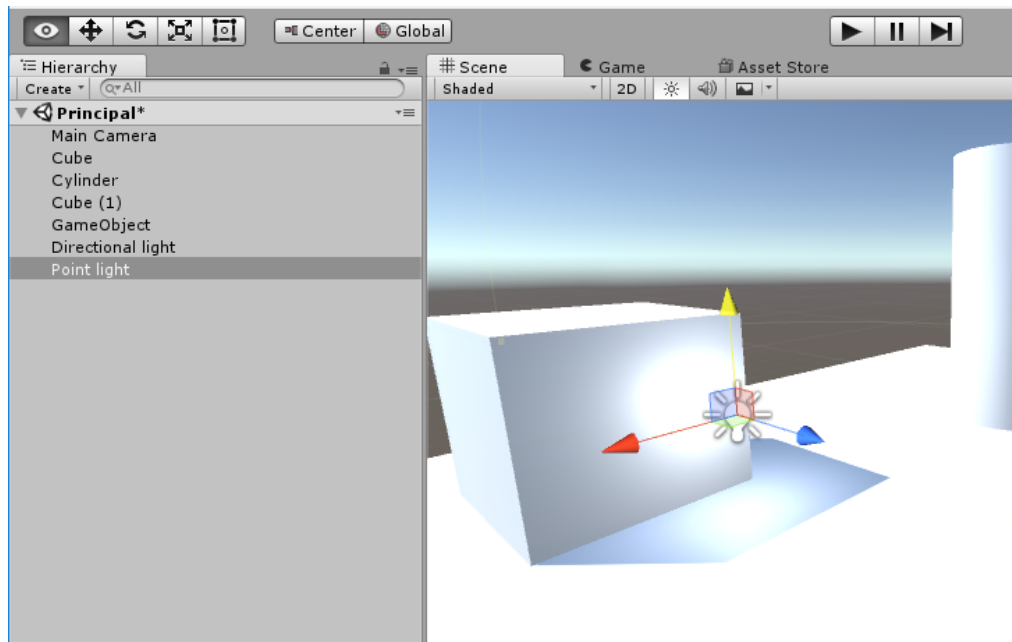
## Adicionando uma Point Light

---

Criaremos agora outra luz na cena. Para isso, vá no menu GameObject -> Light -> Point Light. Um novo GameObject foi adicionado no Hierarchy com o nome **“Point light”**, observe também o seu gizmo no Scene view. Posicione a nova luz próximo a um dos objetos da cena no lado que está mais escuro, como na **Figura 20**.



**Figura 20** - Point Light posicionada próximo ao cubo da cena.

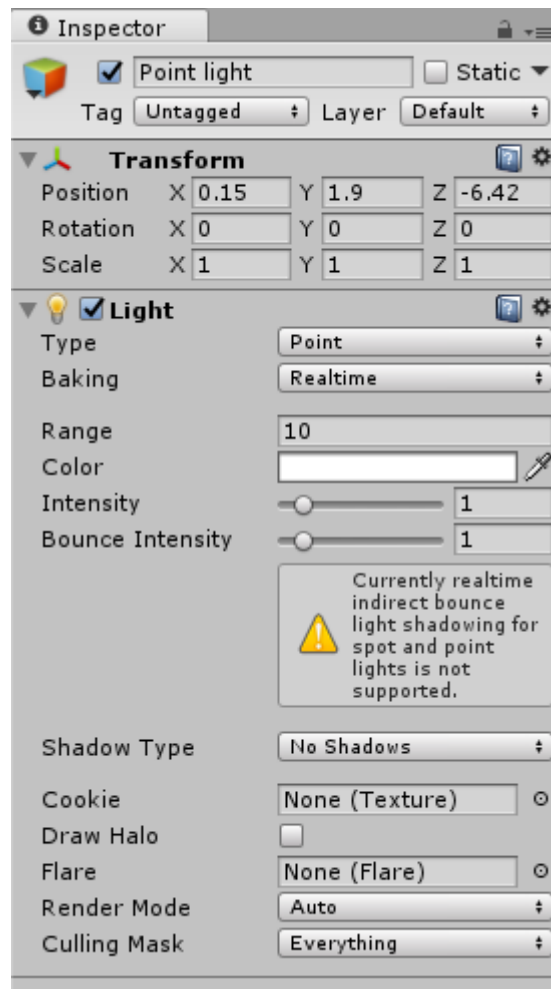


**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

Repare que a **Point Light** emite raios para todas as direções, como vimos anteriormente.

As propriedades da **Point Light** no Inspector podem ser vistas na **Figura 21**.

**Figura 21** - Propriedades da Point Light.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

As propriedades mais relevantes são:

- **Range:** O alcance da luz.
- **Color:** A cor de emissão da luz.
- **Intensity:** Intensidade dos raios de luz.
- **Shadow Type:** Tipo de sombra gerada.

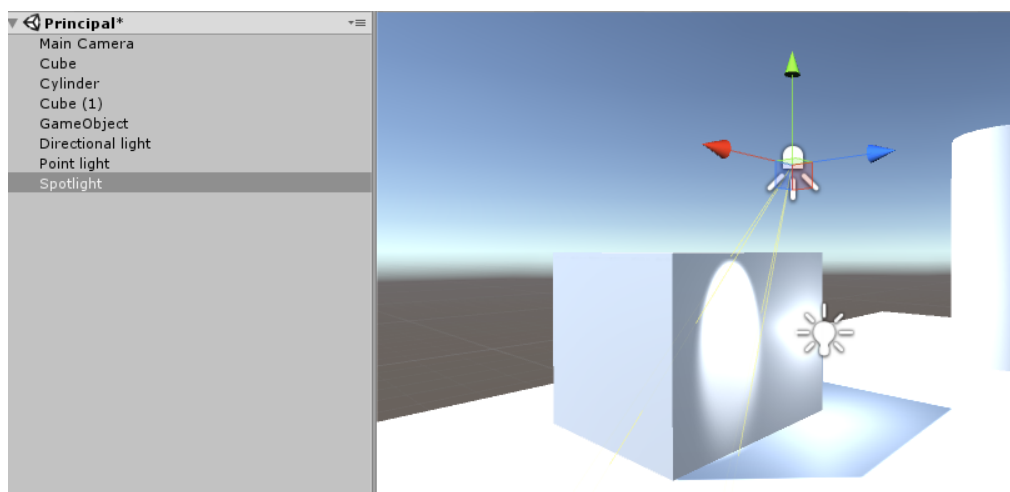
## Adicionando uma Spot Light

---

Agora criaremos uma **Spot Light** na cena. Para isso, vá no menu GameObject -> Light -> **Spot Light**. Um novo GameObject foi adicionado no Hierarchy com o nome "**Spot Light**". Observe também o seu gizmo no Scene view. Posicione a nova luz próximo à Point Light criada anteriormente. Diferentemente da Point Light, a **Spot Light** tem um cone de atuação, então, além de mover a luz, poderá rotacioná-la nos 3 eixos para fazer com que esse cone aponte para onde você deseja iluminar, como em uma lanterna.

Na **Figura 22** temos a **Spot Light** posicionada próxima à Point Light criada anteriormente e com o seu cone apontando para um objeto na cena. Observe a diferença de atuação dessas duas luzes no mesmo objeto.

**Figura 22** - Point Light posicionada próxima ao cubo da cena.

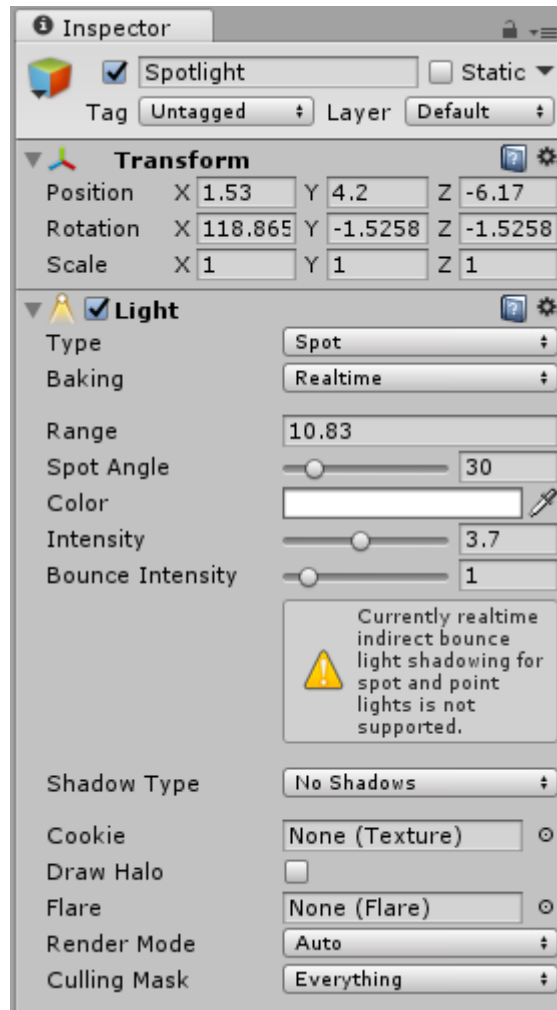


**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

Repare que a Point Light emite raios para todas as direções, como vimos anteriormente, enquanto a **Spot Light** somente emite raios para a direção do cone.

As propriedades da Point Light no Inspector podem ser vistas na **Figura 23**.

**Figura 23** - Propriedades da Spot Light.



**Fonte:** Captura de tela do Unity – Game Engine. Disponível em: <https://unity3d.com/pt/>. Acesso em: 27 de março de 2017.

As propriedades mais relevantes são:

- **Range:** O alcance da luz.
- **Spot Angle:** O ângulo de abertura do cone de luz.
- **Color:** A cor de emissão da luz.
- **Intensity:** Intensidade dos raios de luz.
- **Shadow Type:** Tipo de sombra gerada.

Agora, veremos um vídeo que mostra os diversos tipos de luzes no Unity e como eles afetam os objetos presentes em uma cena.



**Vídeo 01** - Criando Luzes

# Resumo

---

Nesta aula, aprendemos as noções gerais sobre os tipos de iluminação disponíveis no Unity, assim como adicioná-las nas cenas de seus jogos. Exploramos cada propriedade e a realização de mudanças para que as luzes adicionadas no seu jogo tenham um aspecto único e personalizado.

## Leitura Complementar

---

Iluminação é um assunto complexo e nesta aula estudamos muitos aspectos importantes e necessários para você iniciar a criação de jogos. Assim, é imprescindível se aprofundar mais no assunto. Seguem algumas fontes de estudo:

- <https://docs.unity3d.com/Manual/LightingOverview.html>
- <https://docs.unity3d.com/Manual/Lighting.html>
- <https://unity3d.com/pt/learn/tutorials/topics/graphics/introduction-lighting-and-rendering>

## Autoavaliação

---

1. Selecione a Directional Light da sua cena e observe o que ocorre quando você muda a sua cor.
2. Posicione duas Point Light próximas uma da outra e também próximas do Cubo na sua cena. Altere a cor das duas luzes para que fiquem distintas. Veja que a cor de reflexo do cube é uma mistura das duas cores das luzes. Consegue imaginar como isso ocorre?

## Referências

---

UNITY TECHNOLOGIES. 2016 (C). Unity 3D Online Tutorials [online]. Disponível em: <https://unity3d.com/pt/learn/tutorials> [Acessado em 16 de novembro de 2016].

UNITY TECHNOLOGIES. 2016 (C). Unity Manual - Prefabs [online]. Disponível em: <https://docs.unity3d.com/Manual/Prefabs.html> [Acessado em 16 de novembro de 2016].

STOY, C. 2006. Game object component system. In Game Programming Gems 6, Charles River Media, M. Dickheiser, Ed., Páginas 393 a 403.

MARQUES, Paulo; PEDROSO, Hernâni - C# 2.0 . Lisboa: FCA, 2005. ISBN 978-972-722 208-8

UNITY TECHNOLOGIES. 2016 (C). Unity 3D Manual [online]. Disponível em: <https://docs.unity3d.com/Manual/index.html> [Acessado em 16 de novembro de 2016].