Lighting

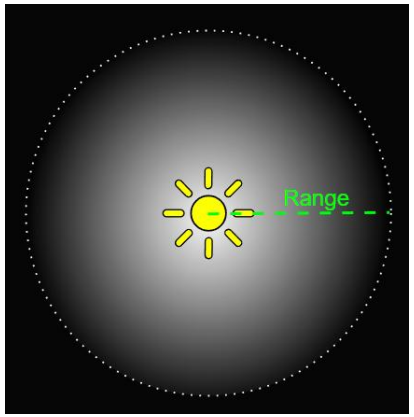
Fall 2025
11/6/2025
Kyoung Shin Park
Computer Engineering
Dankook University

Lighting in Unity

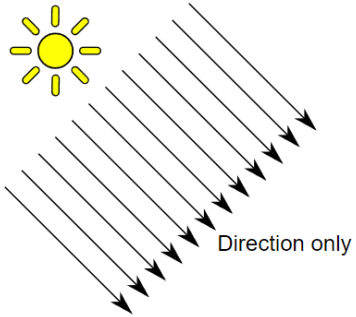
- Direct and Indirect lighting
  - **Direct light** is light that is emitted, hits a surface once, and is then reflected directly into a camera.
  - **Indirect light** is all other light that is ultimately reflected into a camera, including light that hits surfaces several times, and sky light.
- Real-time and Baked lighting
  - **Real-time lighting** is when Unity calculates lighting at runtime.
  - **Baked lighting** is when Unity performs lighting calculations in advance and saves the results as lighting data, which is then applied at runtime.
- Global Illumination
  - Global illumination is a group of techniques that model both direct and indirect lighting to provide realistic lighting results.

Light Sources

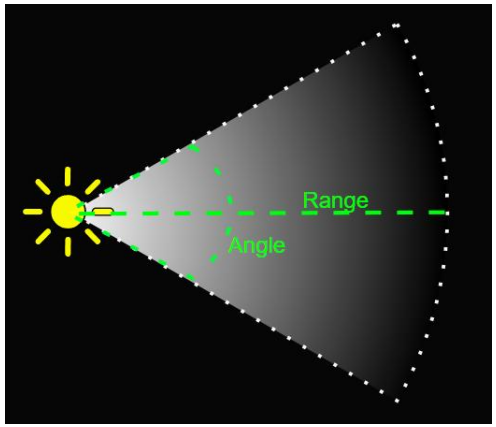
- Light sources
  - Point light
  - Directional light
  - Spot light
  - Area light



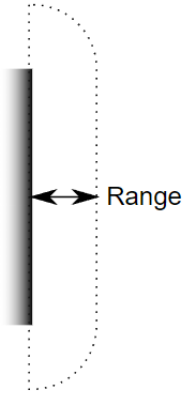
Point light



Directional light



Spot light

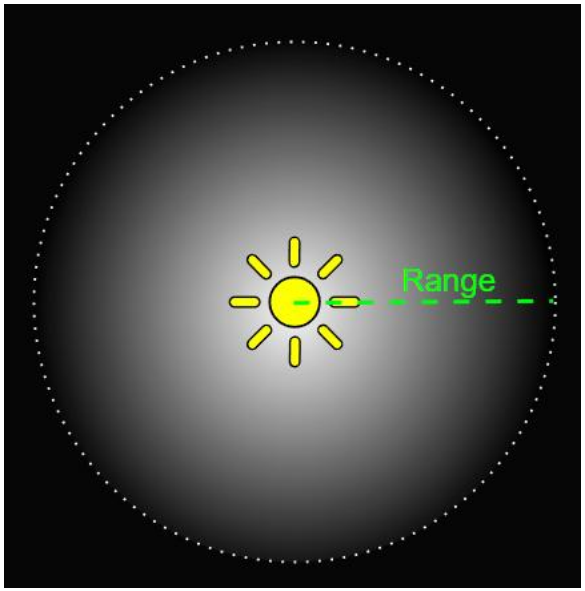


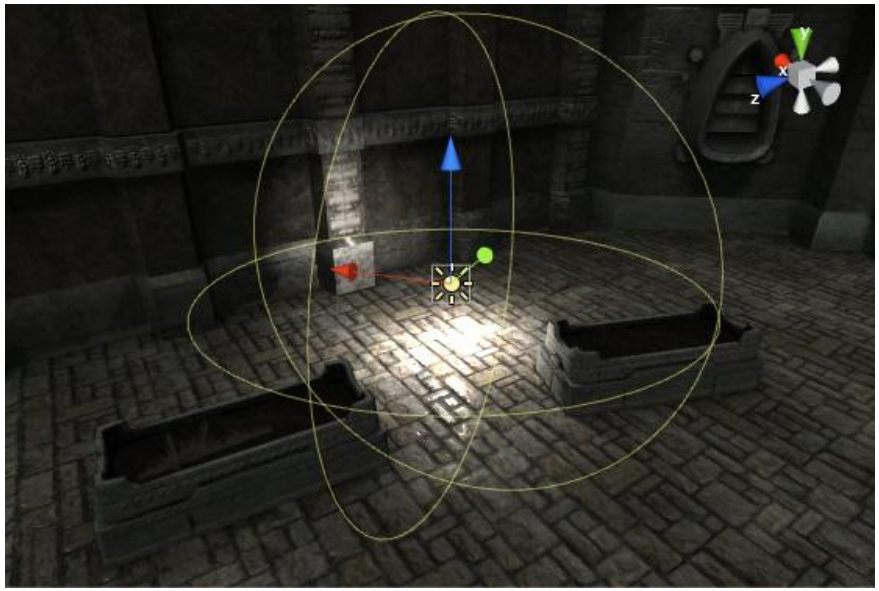
Area light

https://docs.unity3d.com/2019.4/Documentation/Manual/Lighting.html

Point Light

- Point light
  - **A point light is located at a point in space and sends light out in all directions equally.**
  - **The intensity diminishes with distance from the light**, reaching zero at a specified range. Light intensity is inversely proportional to the square of the distance from the source.

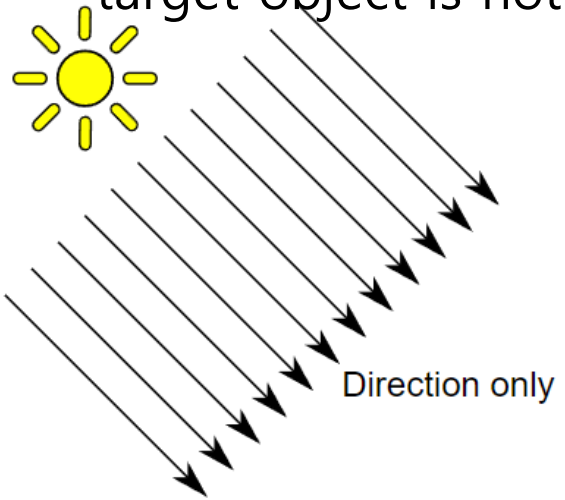


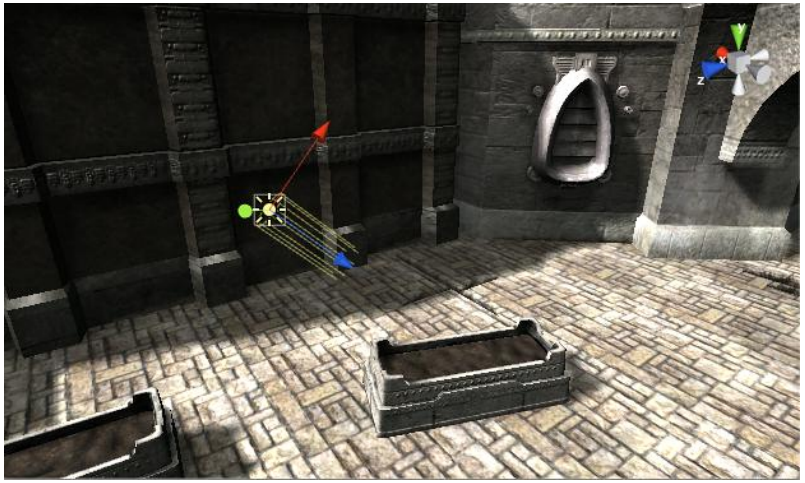


https://docs.unity3d.com/2019.4/Documentation/Manual/Lighting.html

Directional Light

- Directional light
  - Directional lights can be thought of as **distant light sources** which exist infinitely far away. A directional light does **not** have any identifiable **source position** and so the light object can be placed anywhere in the scene.
  - All objects in the scene are illuminated as if the light is always **from the same direction**. The distance of the light from the target object is not defined and so the light does not diminish.

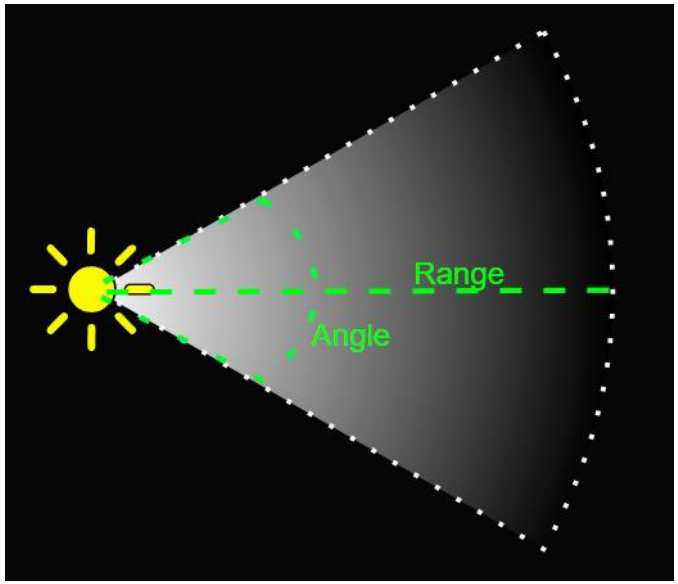


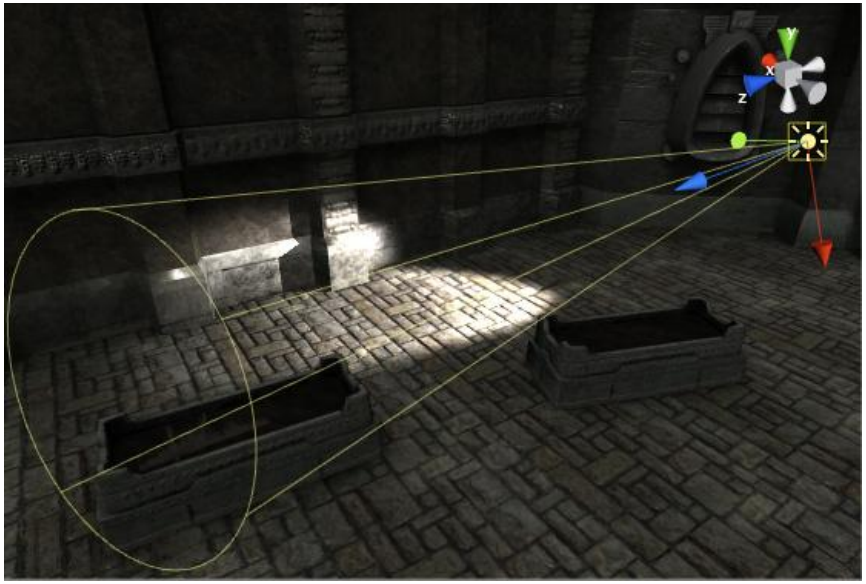


https://docs.unity3d.com/2019.4/Documentation/Manual/Lighting.html

Spot Light

- Spot light
  - Spot lights are generally used for artificial light sources such as **flashlights**, car headlights and searchlights.
  - With the **direction** controlled from a script or animation, a moving Spot Light will illuminate just a small area of the scene and create dramatic lighting effects.

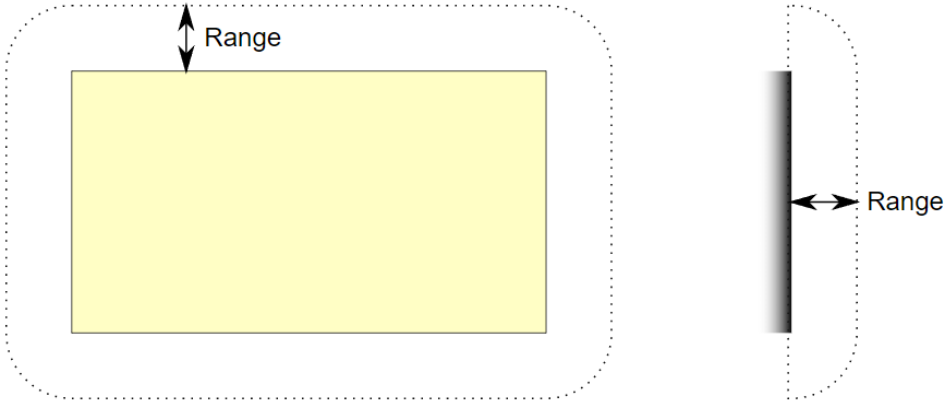


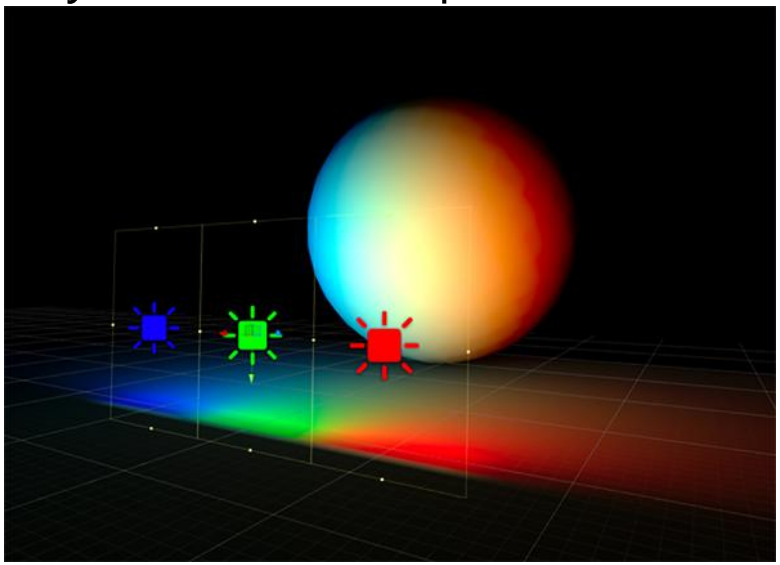


https://docs.unity3d.com/2019.4/Documentation/Manual/Lighting.html

Area Light

- Area light
  - An area light emits light from one side of rectangle (or disc). **The emitted light spreads uniformly in all directions across that shape's surface area**.
  - The **Range** property determines **the size of that shape**. The intensity of the illumination provided by an area light diminishes at a rate determined by the inverse square of the distance from the light source.





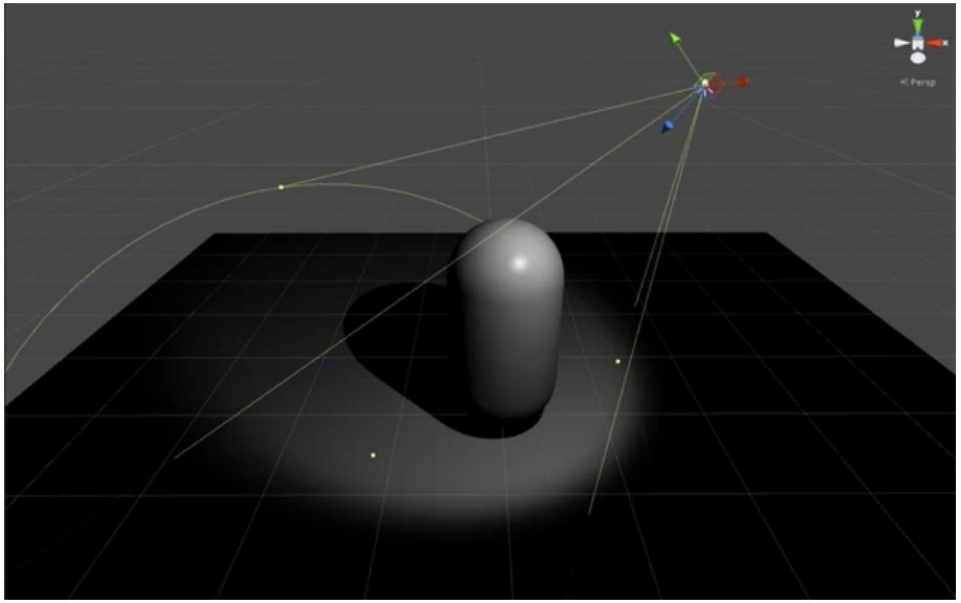
https://docs.unity3d.com/2019.4/Documentation/Manual/Lighting.html

Light Modes

- Light Modes
  - Unity performs the lighting calculations for **Real-time Lights at runtime, once per frame**. You can change the properties of Real-time Lights at runtime to create effects such as flickering light bulbs, or a torch being carried through a dark room.
  - Unity performs the calculations for **Baked Lights** in the Unity Editor, and saves the results to disk as lighting data. This process is called baking. At runtime, Unity loads the baked lighting data, and uses it to light the Scene. Because the complex calculations are performed in advance, Baked Lights **reduce shading cost** at runtime, and **reduce the rendering cost of shadows**.
  - **Mixed Lights** combine elements of **both real-time and baked lighting**. You can use Mixed Lights to combine dynamic shadows with baked lighting from the same light source, or when you want a light to contribute direct real-time lighting and baked indirect lighting.

Real-time Lighting

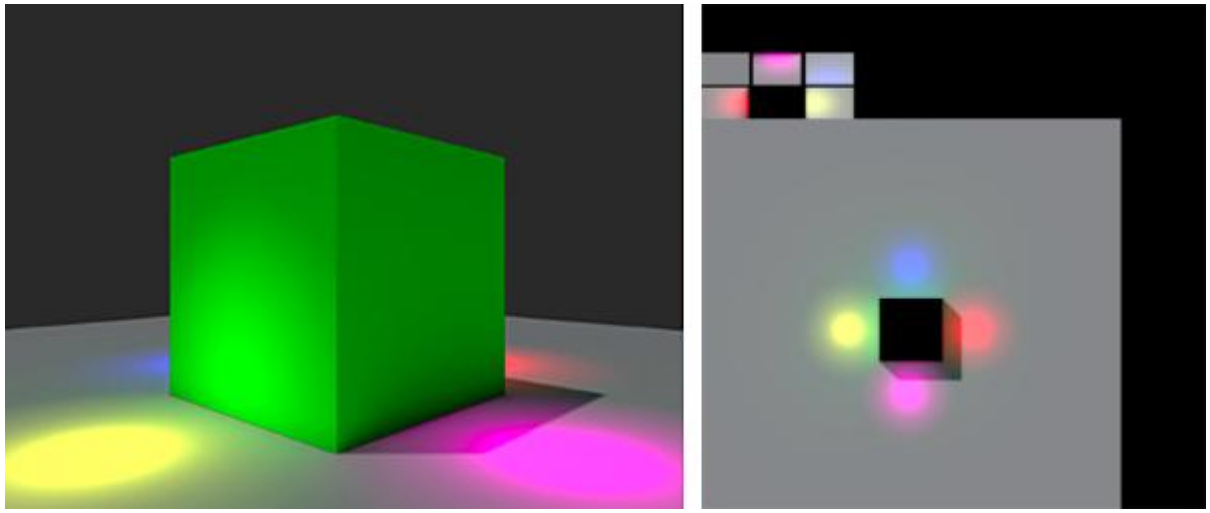
- By default, lights in Unity – Point, Directional, and Spot – are real-time.
  - This means that they contribute direct light to a Scene and update every frame.
  - As lights and GameObjects move within the Scene, lighting is updated immediately.



Realtime light alone - https://learn.unity.com/tutorial/introduction-to-lighting-and-rendering-2019-3?uv=2019.4#5fe99310edbc2a29f8811243

Baked Lighting

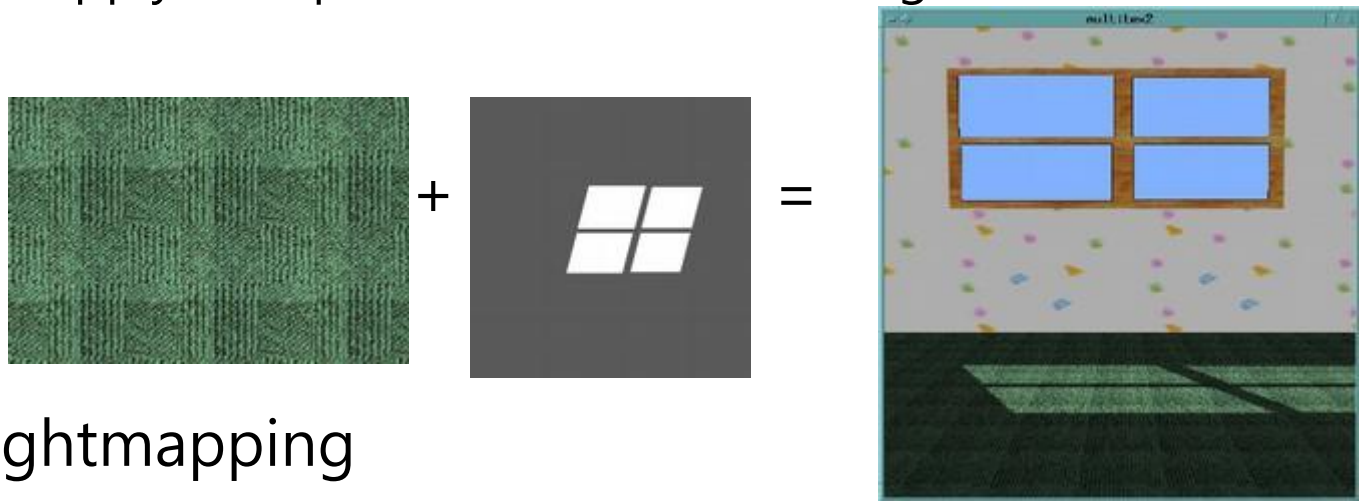
- With baked lighting,
  - **Lightmapping** is the process of pre-calculating the brightness of surfaces in a **Scene**, and storing the result in a Texture called a **lightmap.**
  - Lightmaps can include both direct and indirect light. This lighting texture can be used together with surface information like color (albedo) and relief (normals) by the **Shader** associated with an object's material.



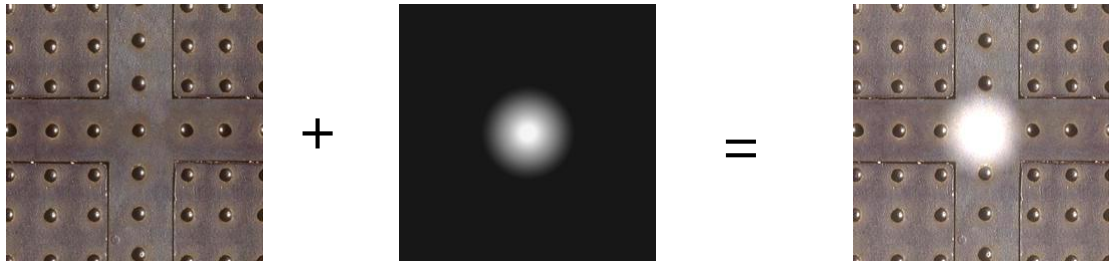
https://docs.unity3d.com/Manual/Lightmappers.html

Lightmapping

- Multitexturing
  - Apply a sequence of textures through cascaded texture units

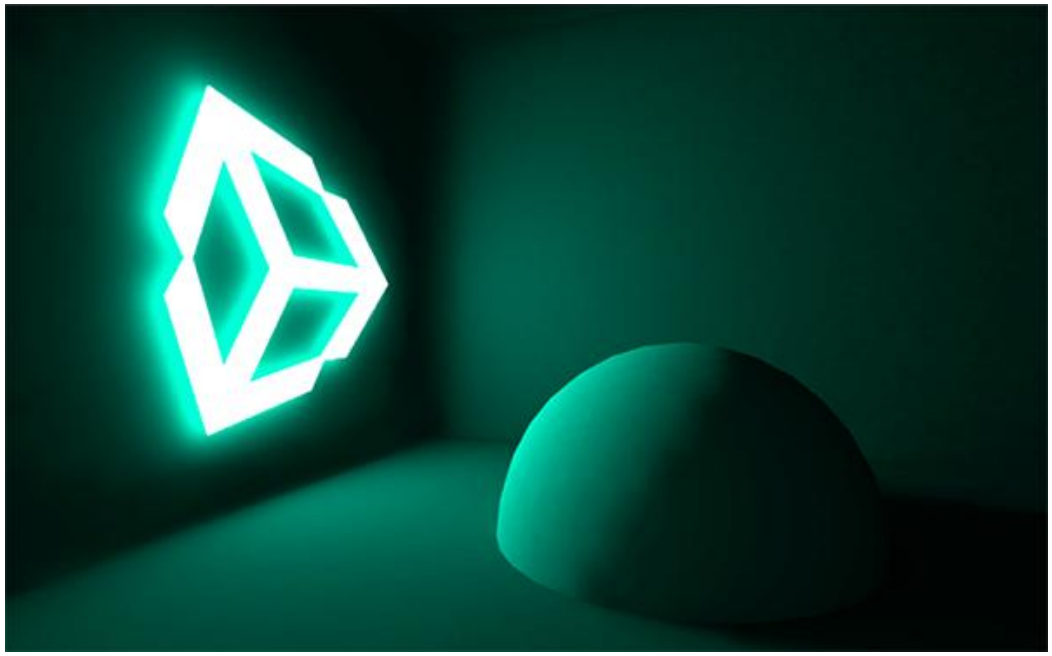


- Lightmapping
  - Instead of calculating the light of the object surface, the texture and bright image are mixed and the resulting image is directly applied to the object surface



Emissive Materials

- Emissive Materials
  - **Materials with an 'Emission' above zero will still appear to glow brightly on-screen** even if they are not contributing to scene lighting.
  - Emissive materials only directly affect **static geometry** in your scene in Unity.



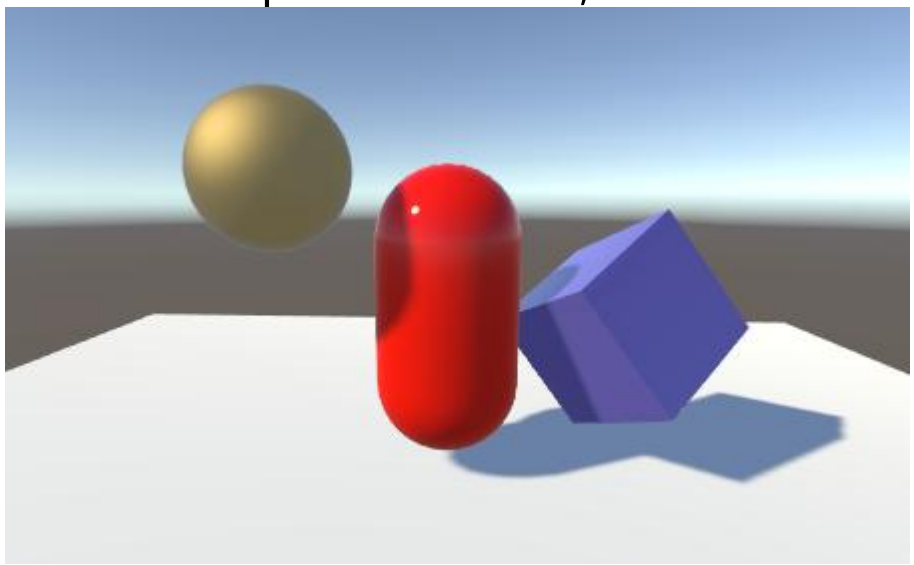
https://docs.unity3d.com/Manual/lighting-emissive-materials.html

Emissive Materials

- Emissive Material
  - *Like area lights, emissive materials emit light across their surface area.* They contribute to bounced light in your **scene** and associated properties such as color and intensity can be changed during gameplay. Whilst area lights are not supported by real-time **Global Illumination**, similar soft lighting effects in real-time are still possible using emissive materials.
  - **'Emission'** is a property of the **Standard Shader** which allows static objects in our scene to emit light. **By default the value of 'Emission' is set to zero.** This means no light will be emitted by objects assigned materials using the Standard Shader.
  - There is no range value for emissive materials but light emitted will again falloff at a quadratic rate. Emission will only be received by objects marked as 'Static' or "**Lightmap** Static' from the **Inspector**. Similarly, emissive materials applied to non-static, or dynamic geometry such as characters will not contribute to scene lighting.

Shadow

- Shadow
  - **Shadows add a degree of depth and realism to a Scene** because they bring out the scale and position of objects that might otherwise look flat.
  - In Unity, lights can **cast shadows** from a **GameObject** onto other parts of itself, or onto nearby GameObjects.

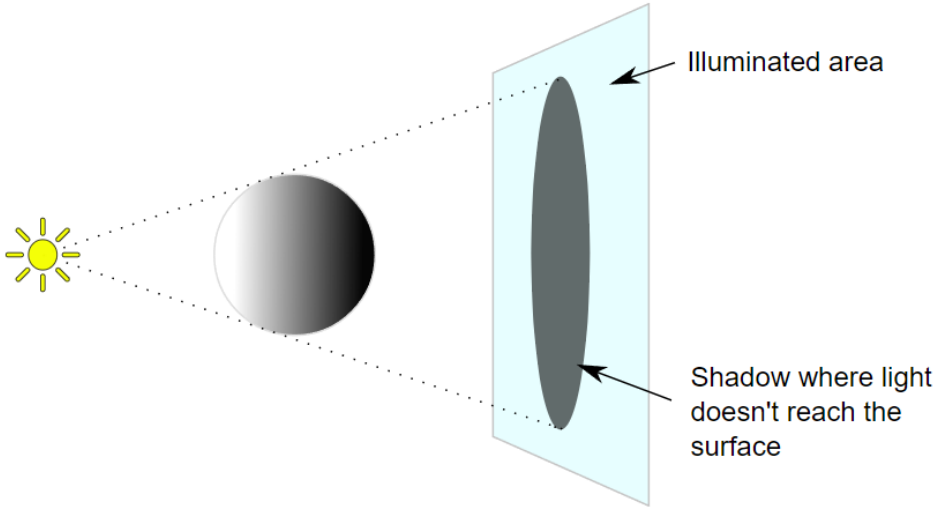


Scene with objects casting shadows

https://docs.unity3d.com/Manual/Shadows.html

Shadow

- How do Shadows work?
  - Consider a simple Scene with a single light source. Light rays travel in straight lines from that source, and may eventually hit GameObjects in the Scene. Once a ray has hit a GameObject, it can't travel any further to illuminate anything else (that is, it "bounces" off the first GameObject and doesn't pass through). The **shadows** cast by the GameObject are simply the areas that are not illuminated because the light couldn't reach them.



https://docs.unity3d.com/2018.4/Documentation/Manual/ShadowOverview.html

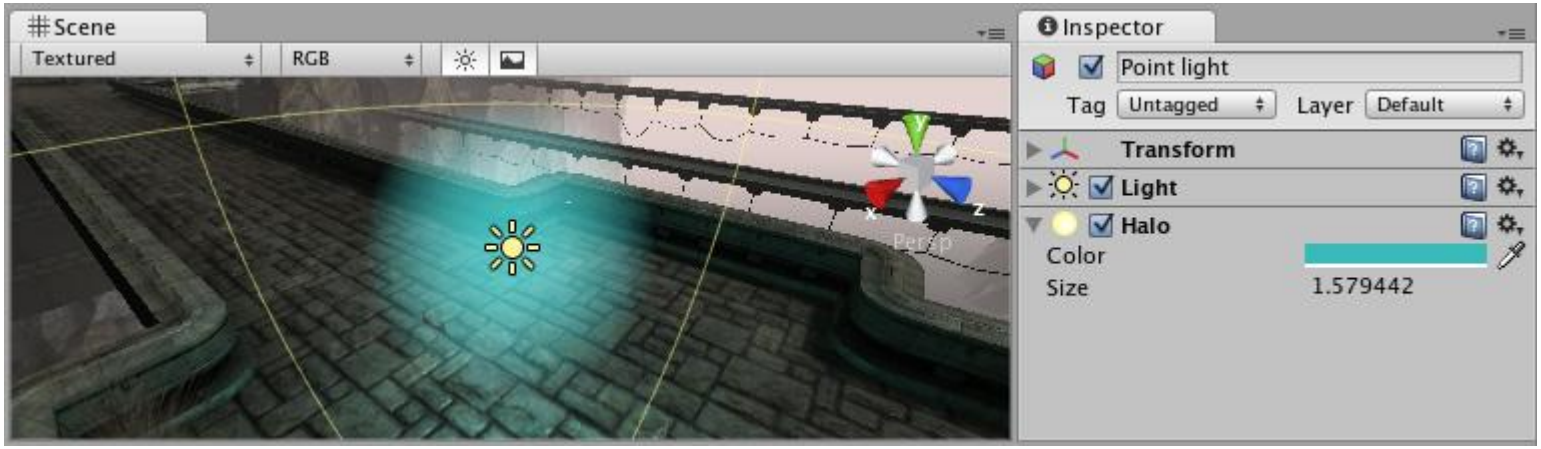
Global Illumination

- Global Illumination (GI)
  - Unity uses middleware called **Enlighten** for Realtime GI.
  - By default, Real-time Lights contribute only direct lighting to a Scene. If you enable **Real-time Global Illumination** (Real-time GI) in your Scene, Real-time Lights also contribute indirect lighting to a Scene.

https://docs.unity3d.com/2018.4/Documentation/Manual/GIIntro.html

Halo

- Halo
  - **Halos are glowing areas around light sources**. Use them to give the impression of small dust particles in the air, and add **atmosphere** to your scene.



https://docs.unity3d.com/Manual/class-Halo.html

Lens Flare

- Lens Flare
  - A **Lens Flare** component displays a lens flare that is configured by a **Flare asset**.
  - You can display a Flare asset with a Light component. If you do this, Unity automatically tracks the position and direction of the Light and uses those values to configure the appearance of the lens flare.

https://docs.unity3d.com/Manual/visual-effects-lens-flares.html

Reference

- https://www.youtube.com/watch?v=u5DNkxkXBeI Lights – Unity Official Tutorials
- https://docs.unity3d.com/Manual/LightingOverview.html
- https://www.theunitygamedev.com/2020/09/08/unity-2020-lighting-for-beginners/
- https://learn.unity.com/tutorial/introduction-to-lighting-and-rendering-2019-3?uv=2019.4