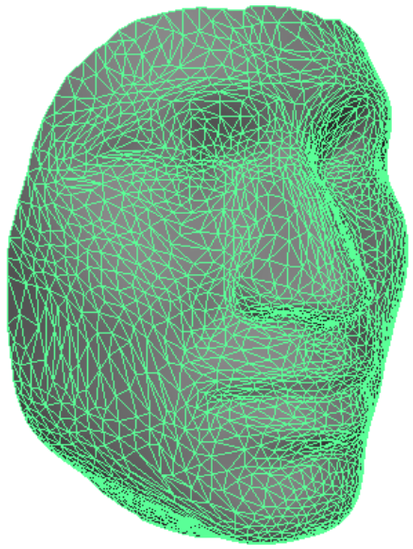
UNITY

BASIC GRAPHICS

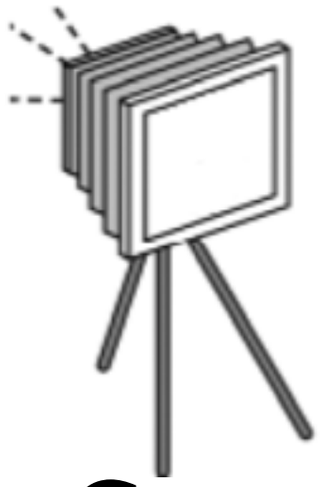
TANASAI SUCONTPHUNT

3D Scene

Light



3D Model

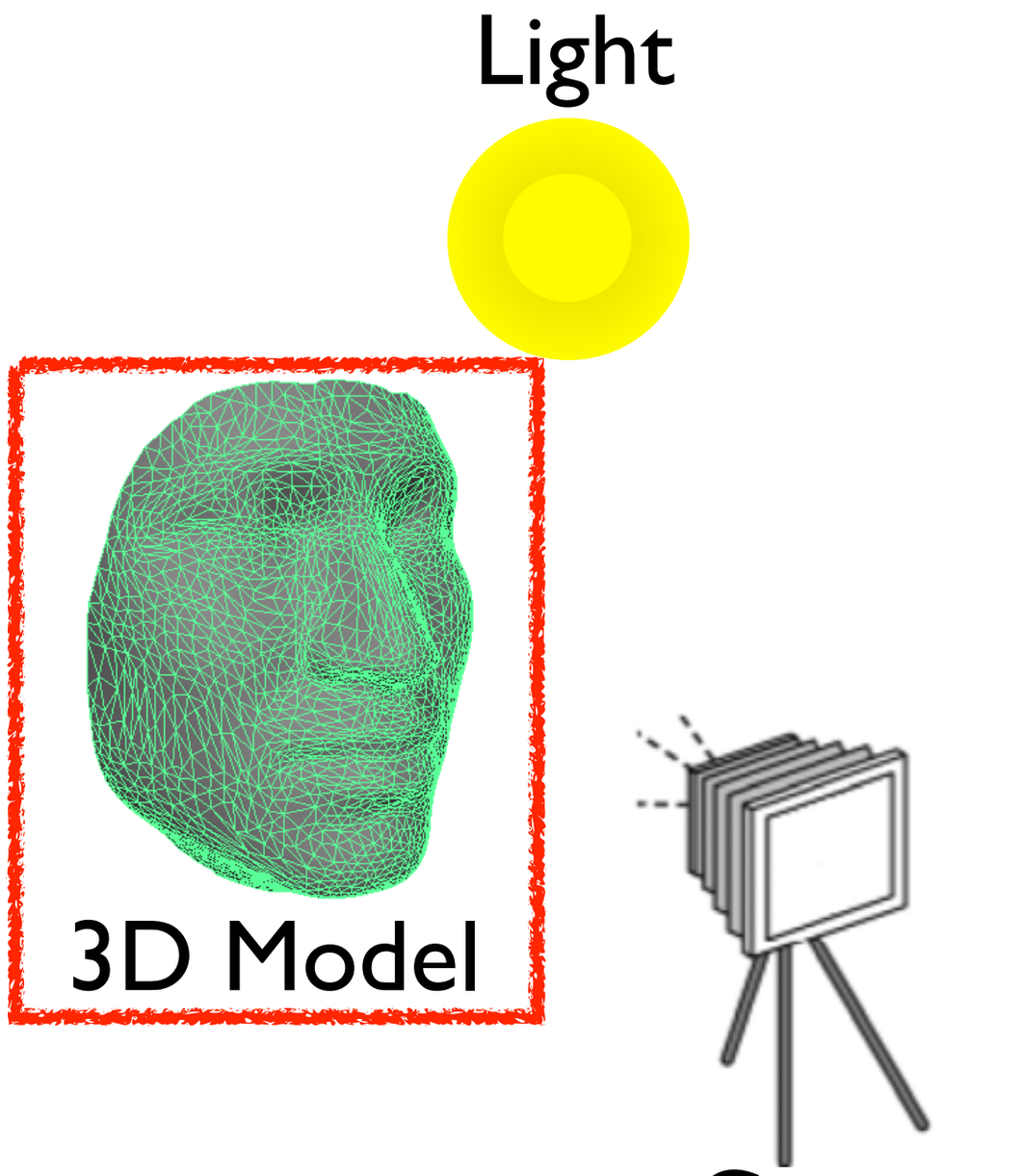


Camera

3D Graphics Pipeline

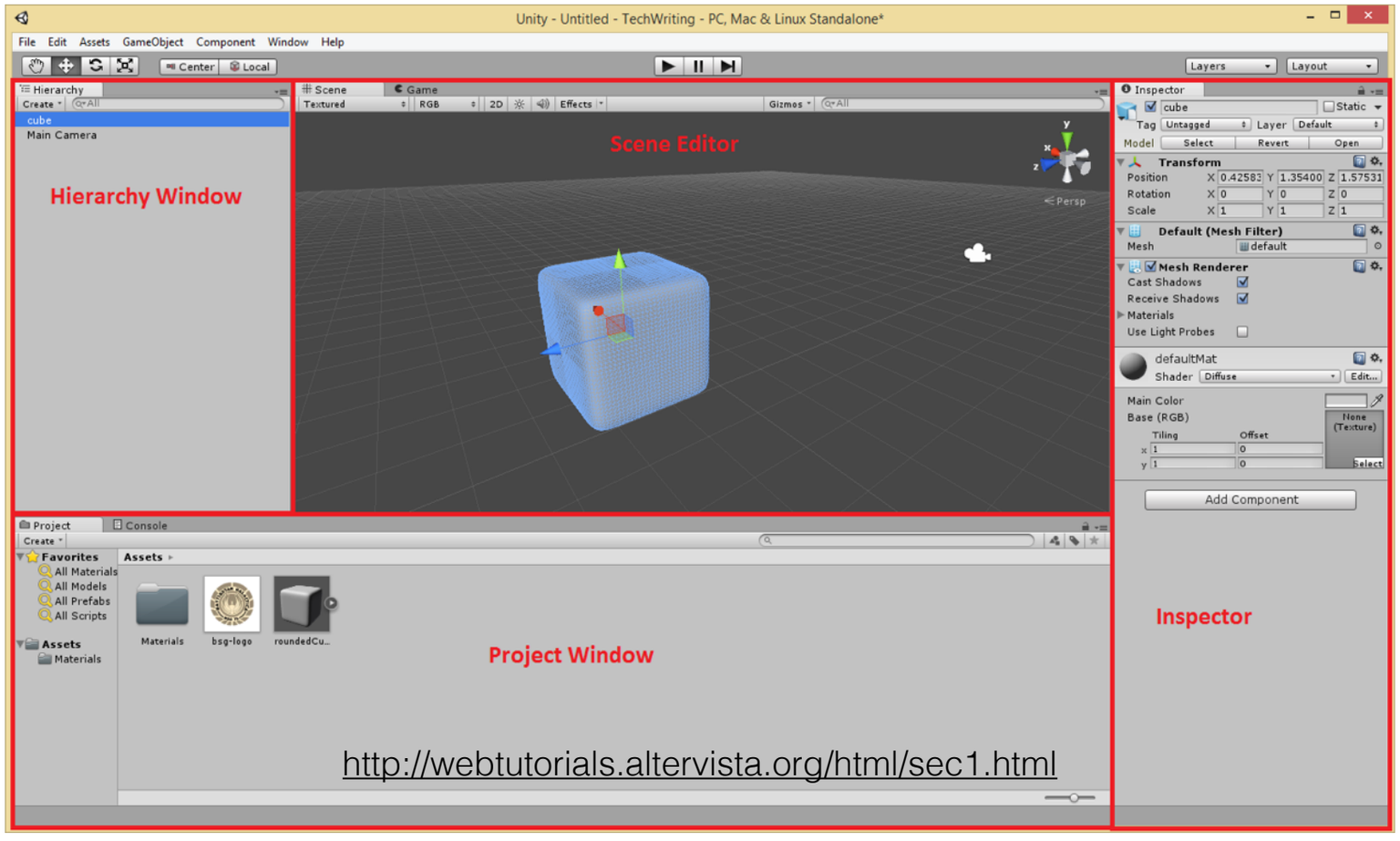
Image

Primitive 3D Models



3D SCENE REVISITED

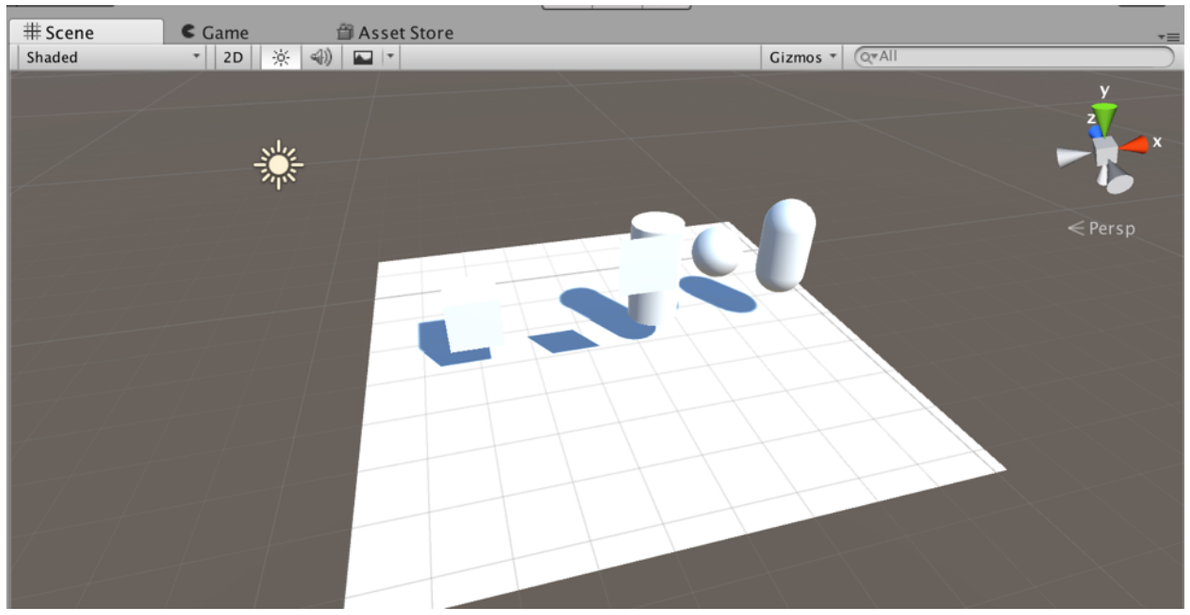
Unity's User Interface



- Hierarchy Window = List of components
- Scene Editor = Editing here!
- Inspector = Select each component from other windows and set them up here
- Project Window = Asset Collection

Scene View navigation

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

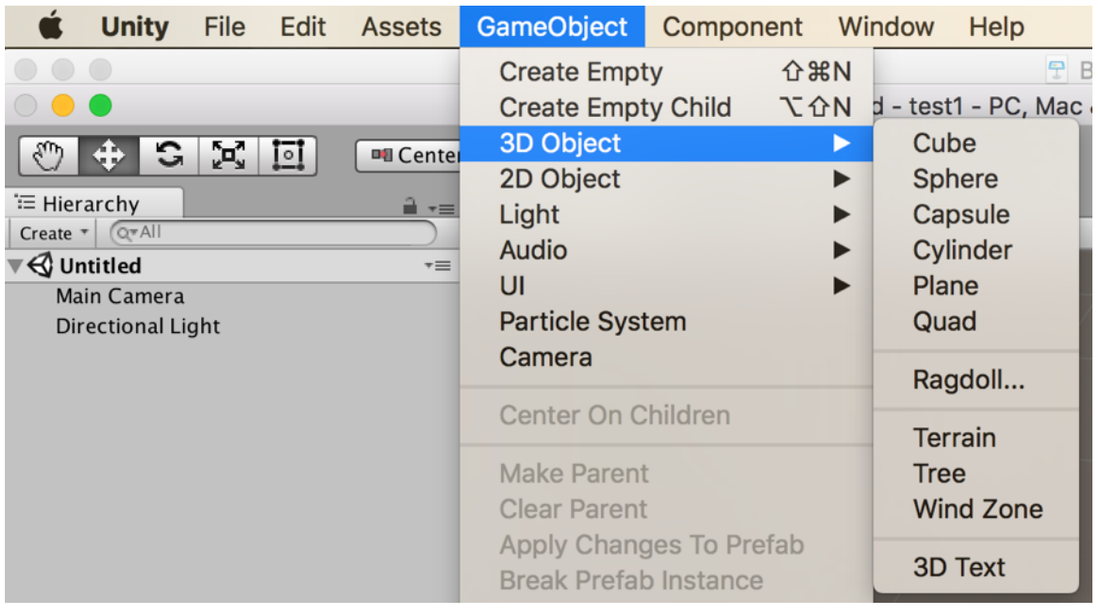


Quick mouse navigation

- Scroll bar = Zoom in/out
- Left click = Selection
- Right click = Rotate viewing
- Option + Right click = Rotate viewing around object

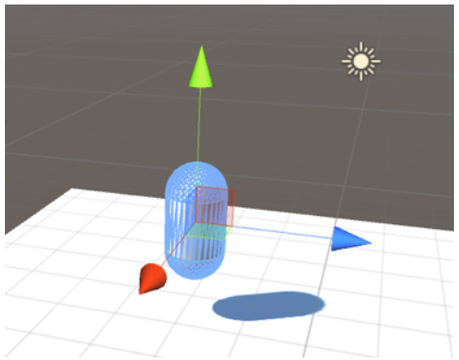
Unity Primitive 3D Object Lab

- Ready-to-use 3D objects (don't have to model them by yourself)
- GameObject -> 3D Object
- Try: Cube, Sphere, Capsule, Cylinder, Plane, Quad

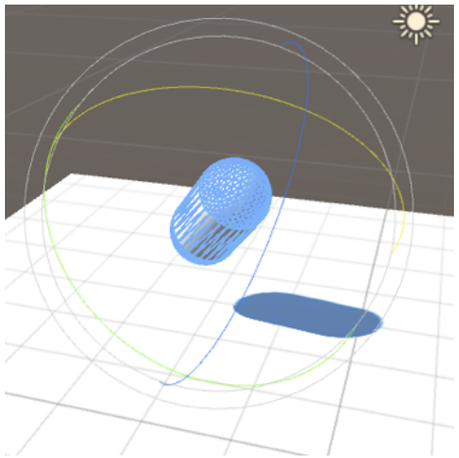


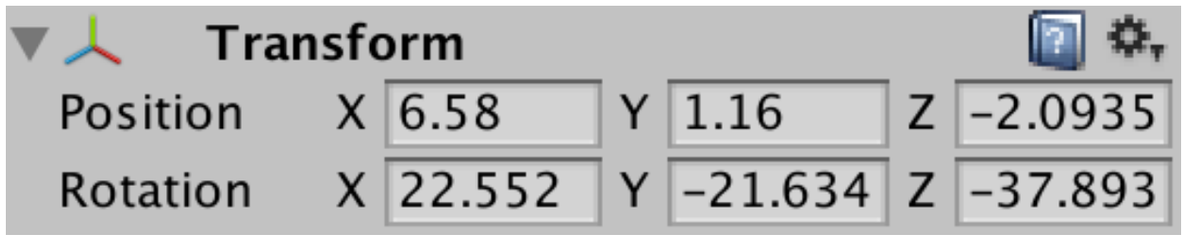
Unity 3D Transformation Lab

- Translation

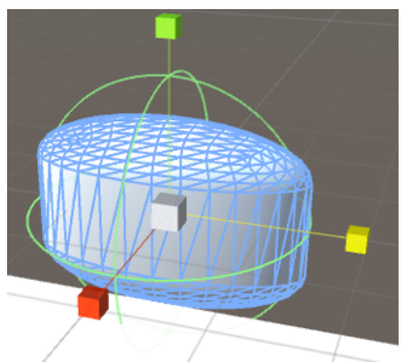


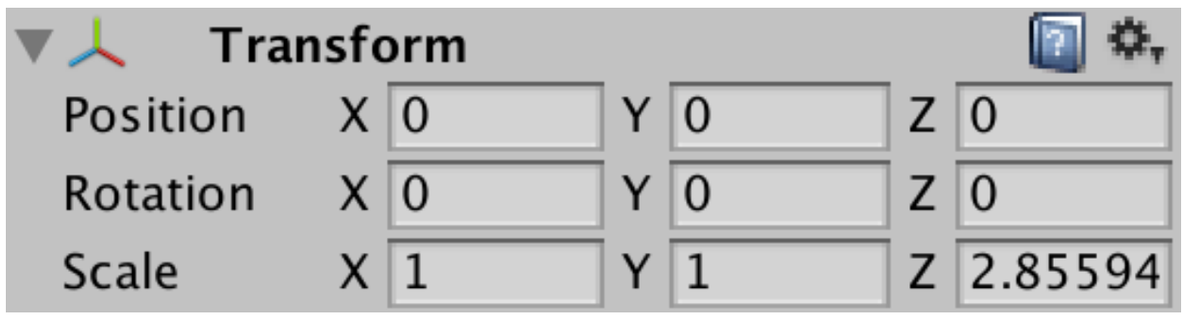
- Rotation





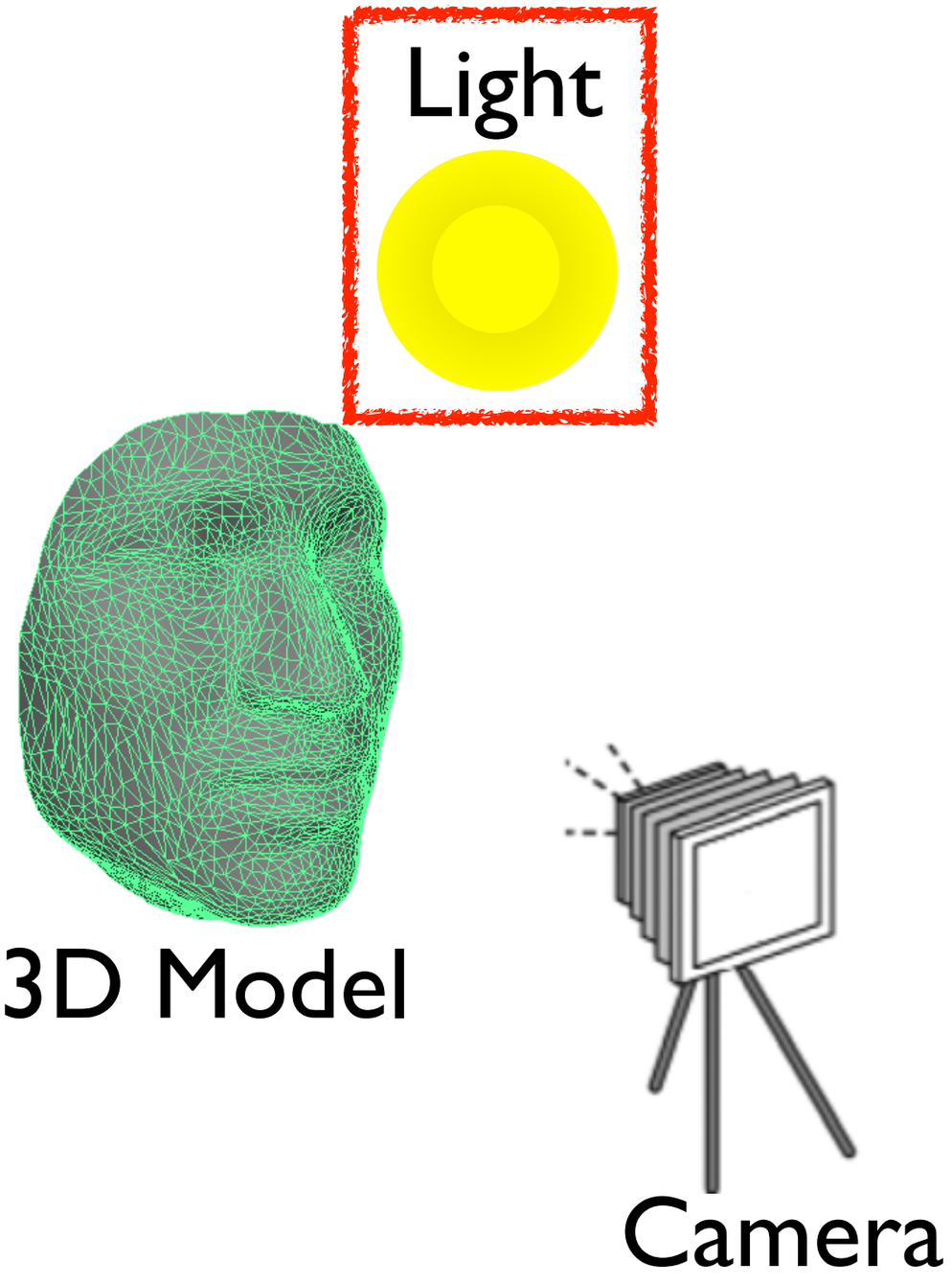
- Scaling





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

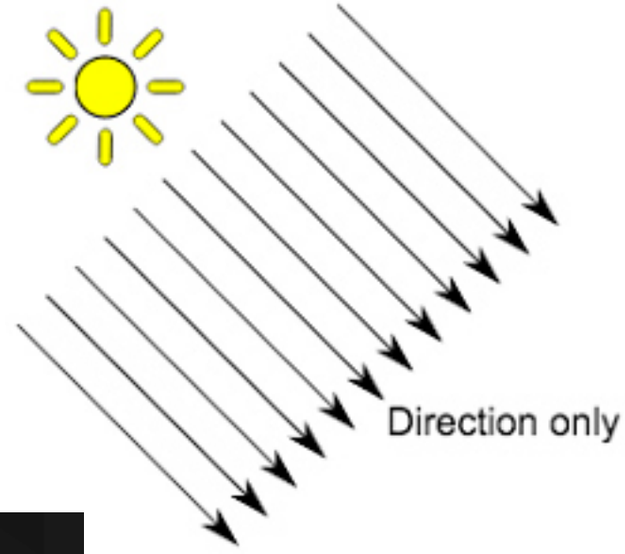
LIGHTING

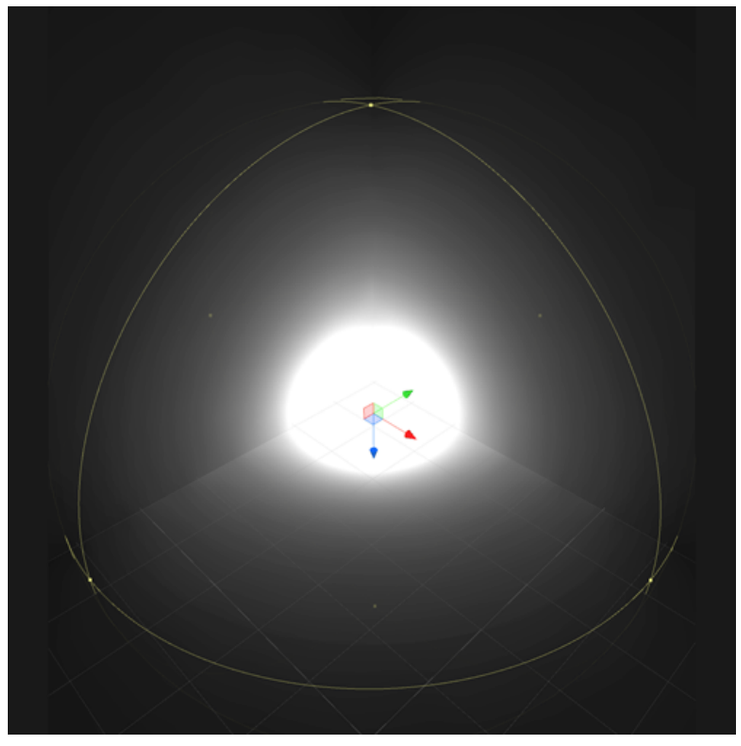


3D SCENE REVISITED

Unity Realtime Lighting

- Bt default:
  - Directional
  - Point
  - Spot

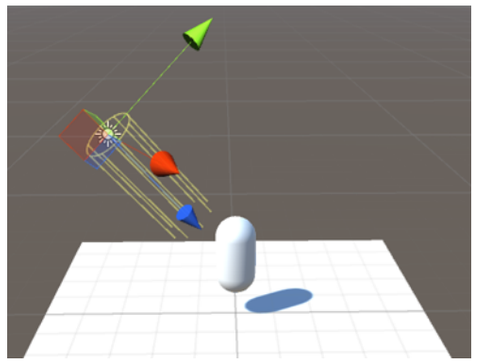




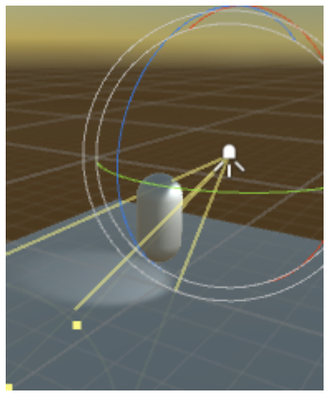
Unity Realtime Lighting Lab

- Types

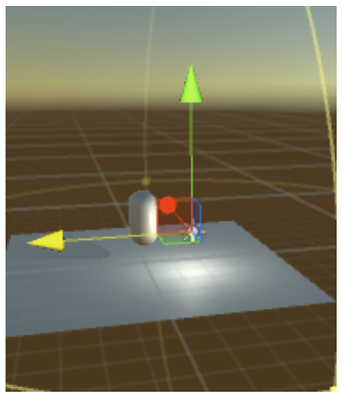
  - Directional (Easiest to set)

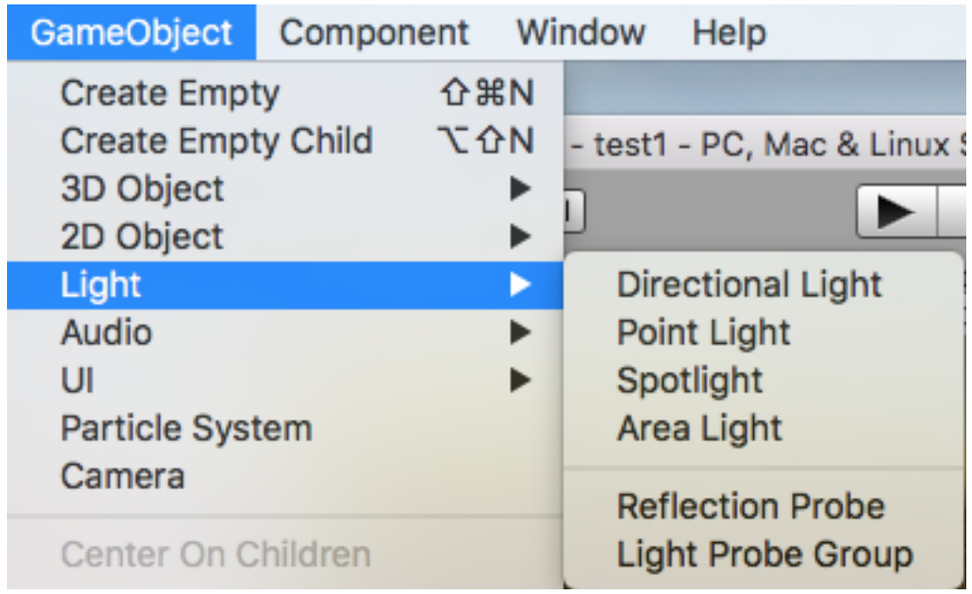


  - Spot (+Spot Angle)



  - Point



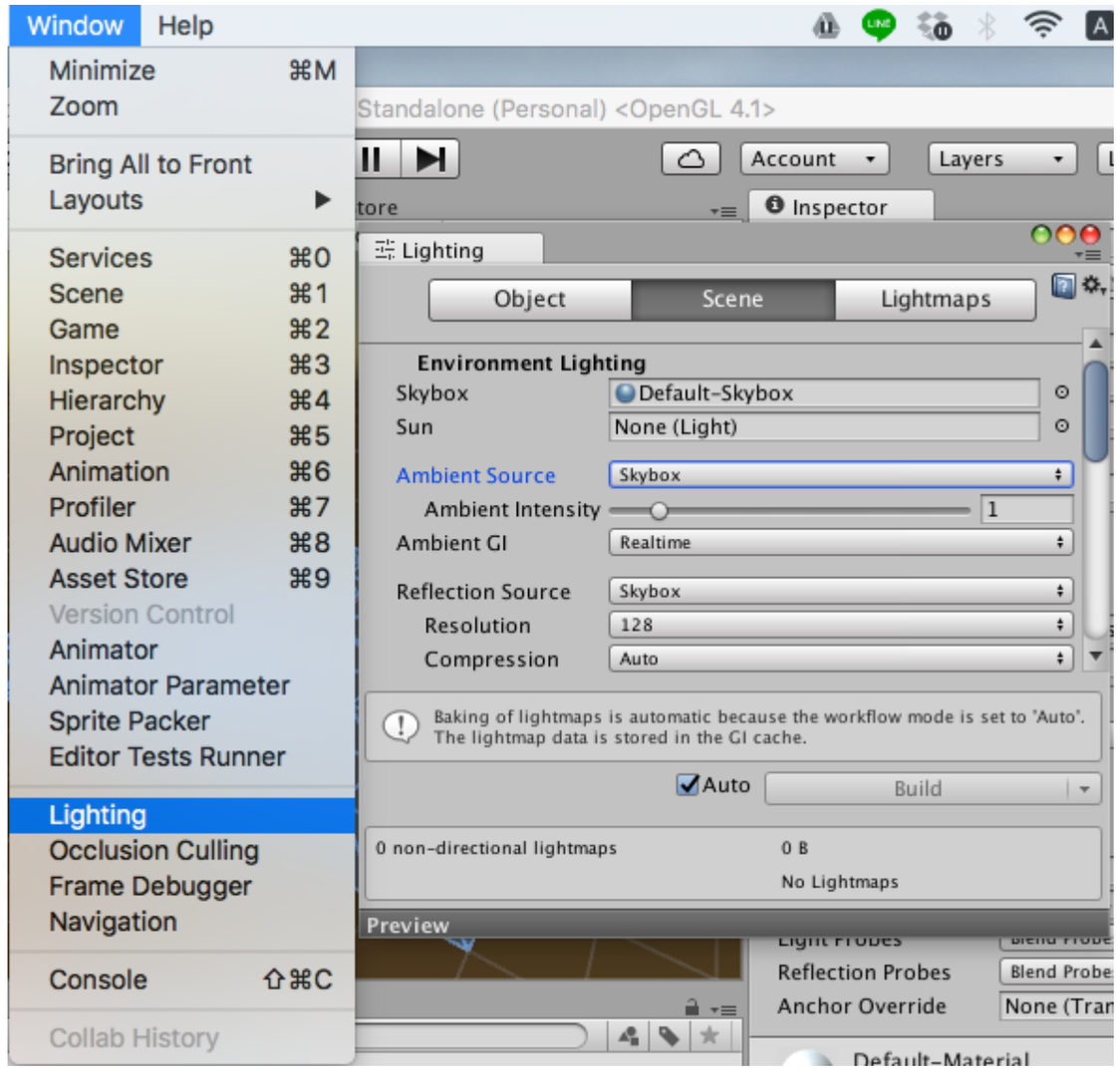


- Area (baked only = ?) = most realistic (later)

Try setting: Position, Rotation, Color, Intensity

Where is ambient light?

- Unity separate this light and put it in Scene lighting properties
  - Window -> Lighting -> Scene -> Environment Lighting
    - Ambient Source



What about material colors?

Unity: 3D Object's Material Color

- In Unity, the color of materials can be set by
  - Coding: e.g.

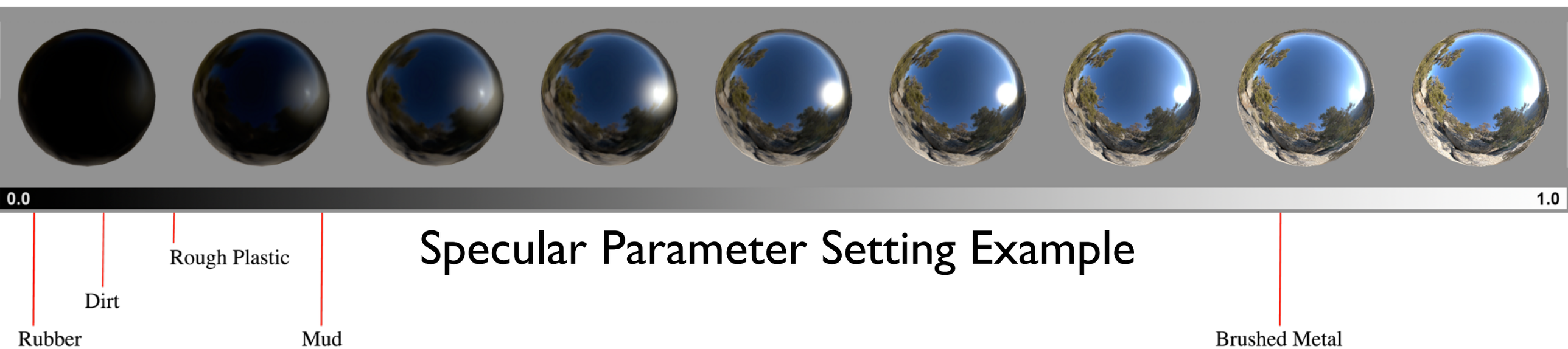
```
gameObject.GetComponent<Renderer> ().material.color = Color.green;
```

  OR

  - Set new material and set its albedo color (diffuse)
- Quick trying the new material
  - Project -> Create -> Material
  - Set new albedo color
  - Drag this new material to your object (replacing the default material which albedo is white)
- Standard shader?

Unity: Standard Shader

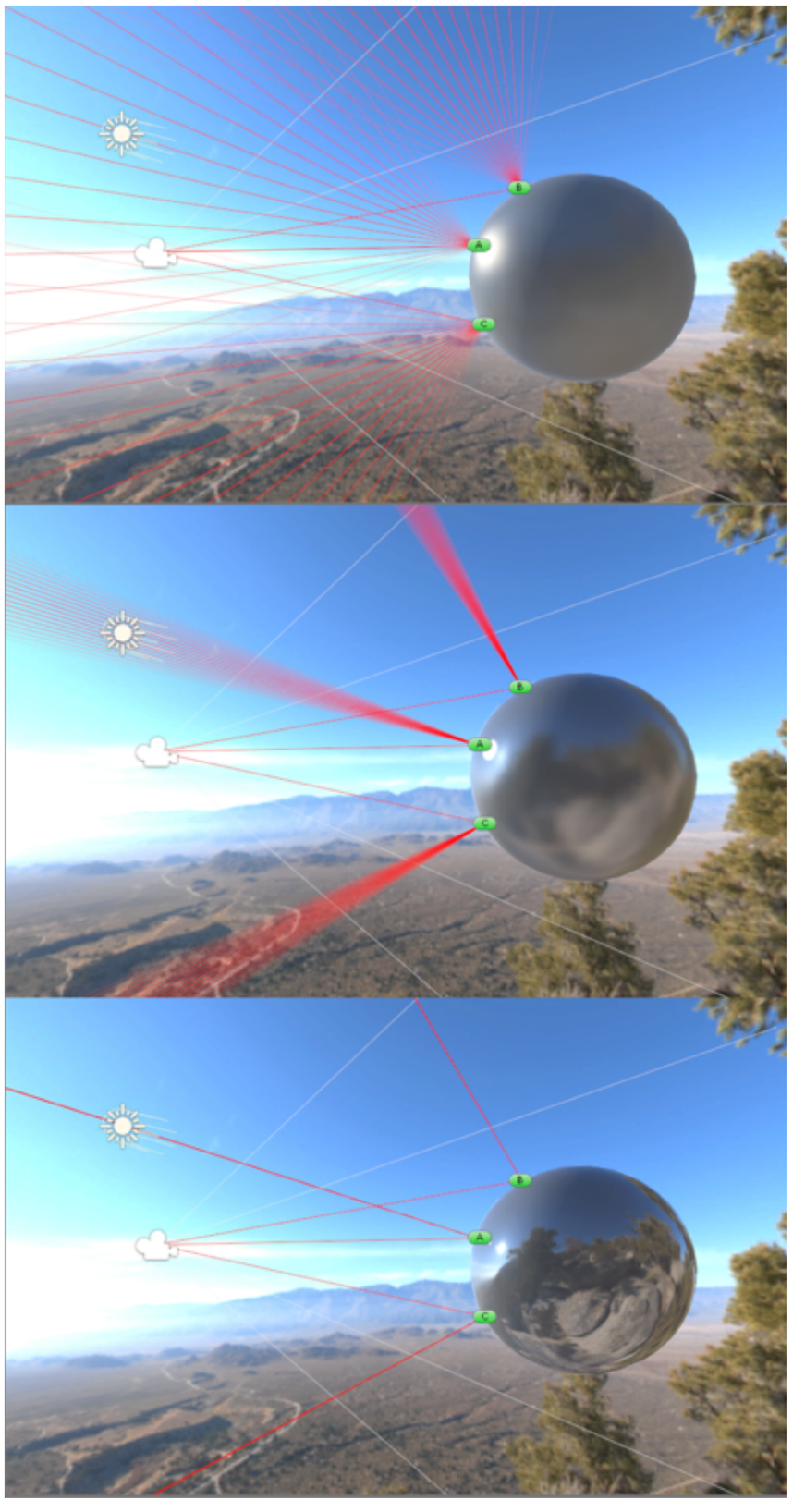
- Physically Based Shading (PBS)
  - Mimicking how the real light interact to material
    - Energy Conservation
    - Fresnel Reflections (material refractive index)
    - Optimizing to make it run in real-time
- Metallic by default
  - Any materials can be modeled with metallic with different parameters



Specular Parameter Setting Example

Unity: Standard Shader

- Energy Conservation
  - Objects never reflect more light than they receive.
  - Reflection = ambient + diffusion + specular (not separately)
  - Duller surfaces = more diffuse, less specular
  - Smoother surfaces = more specular, less diffuse



Unity: Standard Shader Lab

- Create a new scene
- Add a 3D plane as a ground
  - GameObject -> 3D Object -> Plane
- Add 3 3D objects (i.e. sphere, cube, cylinder)
  - Game Object -> 3D Object -> Sphere/Cube/ Cylinder
- Adjust your 3D objects's positions and shapes with transformations (translation, rotation, and scaling)

Unity: Standard Shader Lab

- Create new materials (= number of your objects)
  - Project -> Create -> Material
- Name them: Metal, Plastic, and Rubber
- Double click on the new created materials
- Setting their Albedo, Metallic, and Smoothness to reflect their names (hint: http://docs.unity3d.com/500/Documentation/Manual/StandardShaderMaterialParameters.html)
- Add these materials to each 3D object (any order)

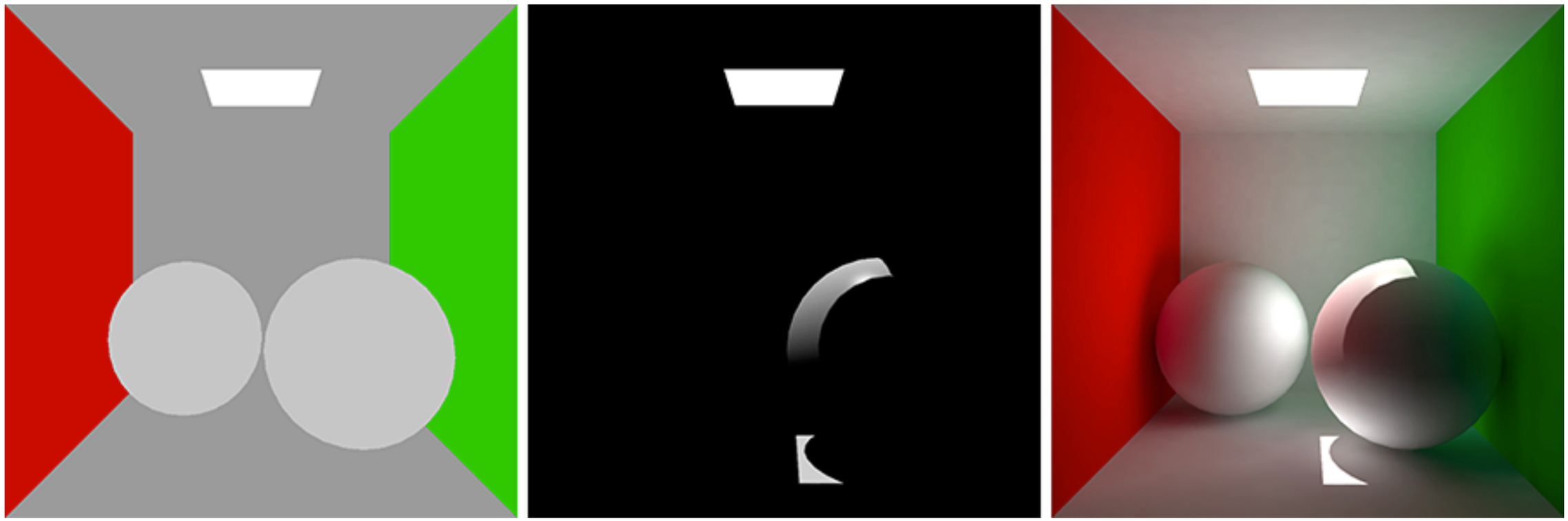
Unity: Standard Shader Lab

- Add Lighting
  - GameObject -> Light
- Adjust light parameters and see how they work
  - Recommend: move/rotate, Cookie, etc.
- Adjust Ambient Lighting
  - Window -> Lighting -> Scene -> Environment Lighting
  - Ambient Source/Intensity

OTHER ILLUMINATIONS?

Unity Illuminations

- Unity Illumination Types



Ambient | Direct (Local) | Global (Direct + Indirect)

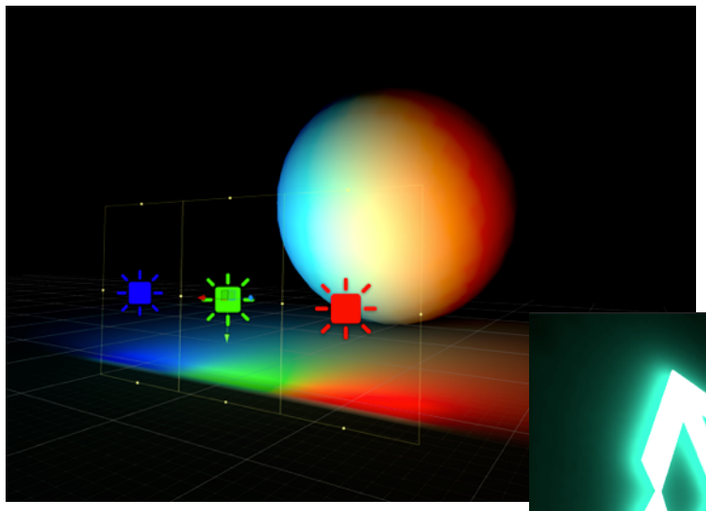
Why don’t we always use Global Illumination?

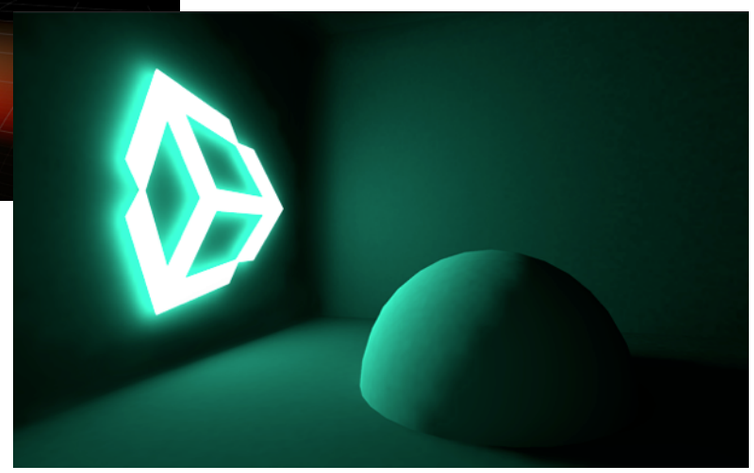
Unity Illuminations

- All types of Unity Lighting
  - Real-time (Dynamic with gameplay) - direct (shown)
  - Offline (Precomputed) - next slides
- How can Unity render GI in gameplay?
  - Precomputed as much as possible
  - Separate geometry from lighting calculations
    - Deferred Shading

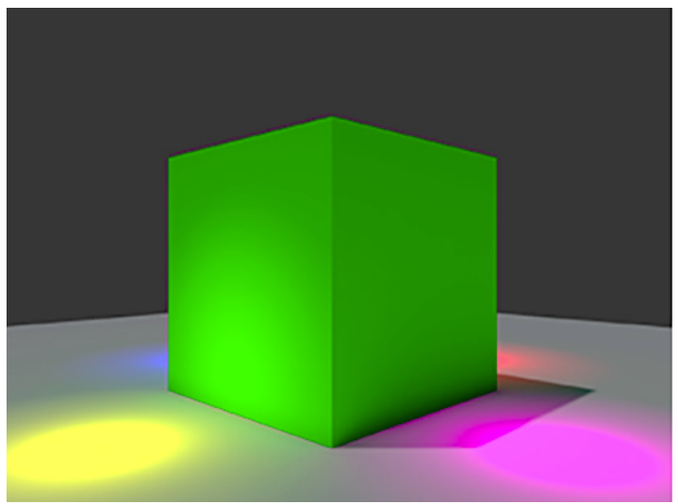
Precomputed Lighting

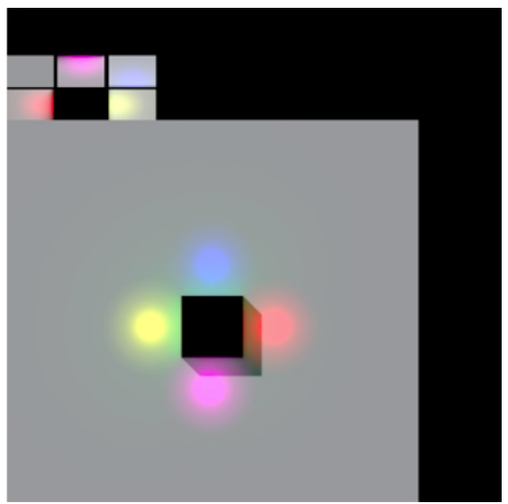
- For lighting types:
  - Area
  - Emissive Materials





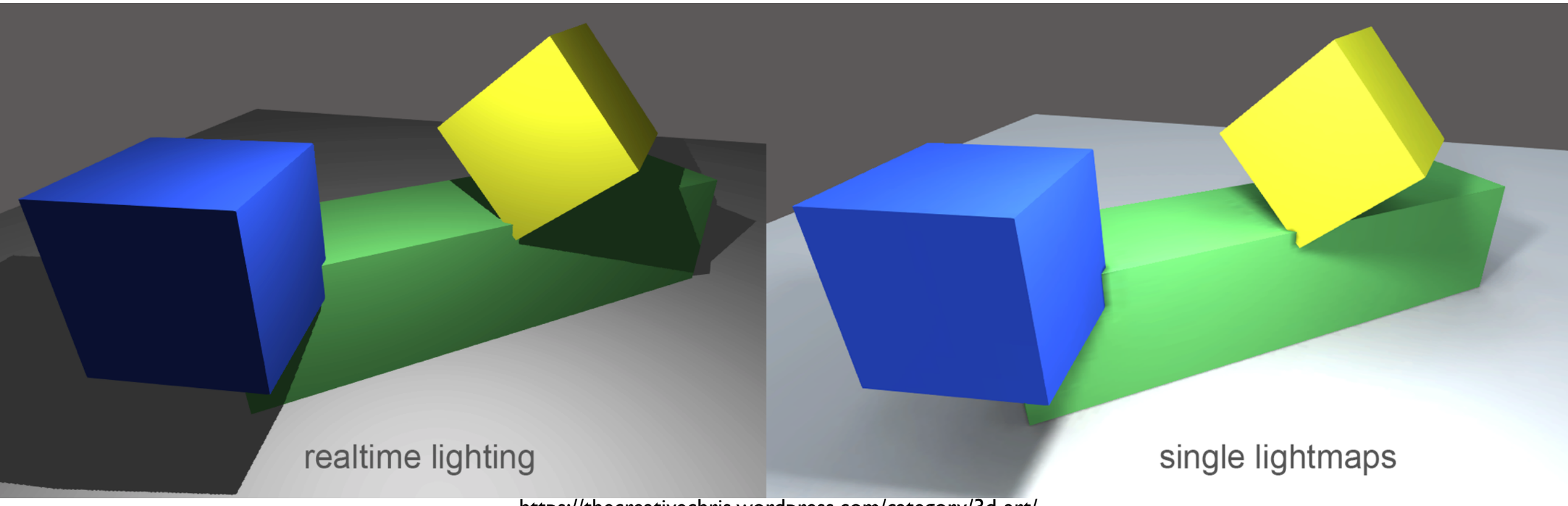
- Baked GI Lighting (lightmap)
  - For static objects (both direct and indirect light)
  - Written to texture before gameplay





Precomputed Lighting

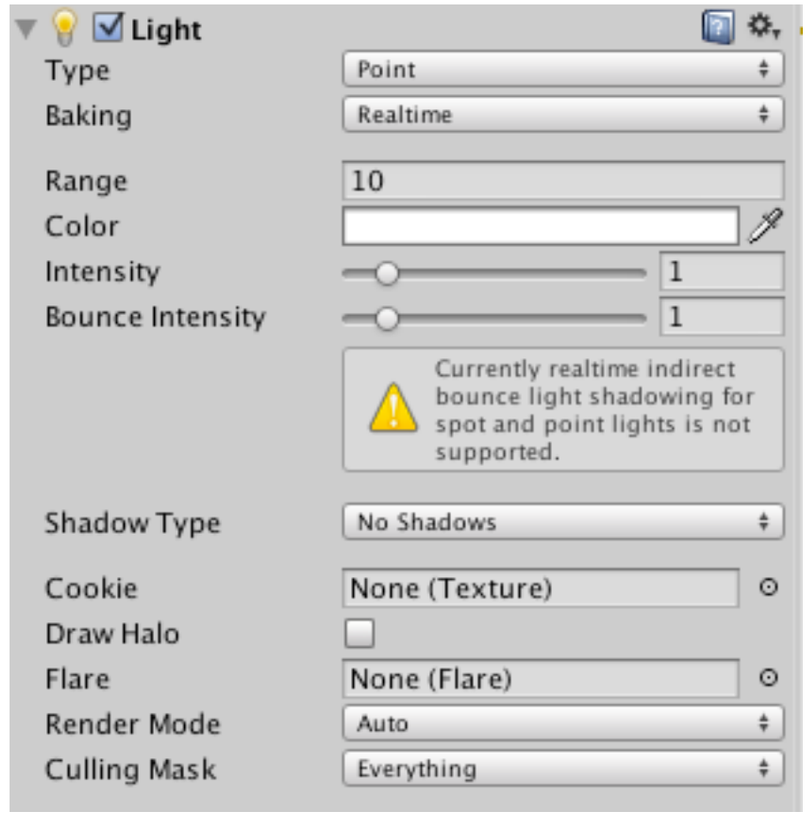
- Precomputed Realtime:
  - Give dynamic feeling in gameplay
  - Indirect light (softer) = Lightmaps (Baked)
  - Direct light (sharper) = real-time



https://thecreativechris.wordpress.com/category/3d-art/

Unity Light Setting

- Example: at light source property inspector
  - Baking = Realtime/Baked/Mixed
  - Render Mode = Important/Not Important
  - Important = Pixel Shading (slow but high quality)
  - Not Important = Vertex Shading (fast)



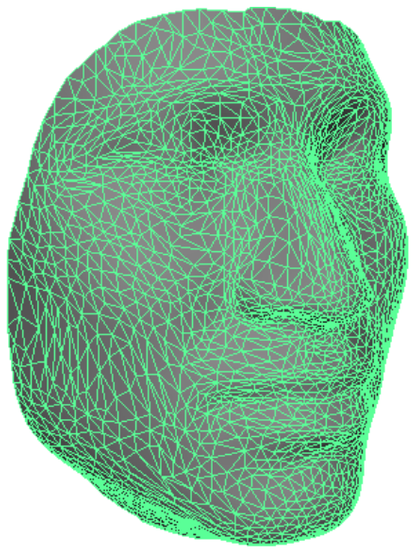
Unity: Baking Lab

- From your last scene, add Area Lighting
  - GameObject -> Light -> Area Light
  - Adjust width, height, color, intensity of the area lighting
- Mark other 3D objects as static (only static objects can be lightmapped)
  - Window -> Lighting -> Object -> Mesh Render -> Lightmap Static
  - The light map will only re-generate automatically when the scene is changed. To manually build, go to Lighting -> Lightmaps -> Auto (uncheck) -> Build
- Disable other lights (check the scene realism)

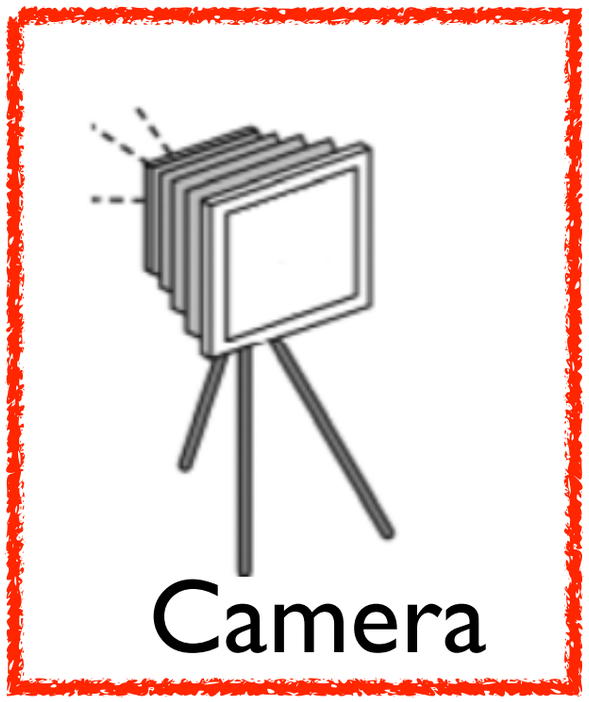
Unity: Baking Lab

- Add Emissive Material
  - Add new 3D Object (any)
  - Create new material naming it “Emissive”
  - Adjust the material emission to HDR level (> 1.0)
  - Set Global Illumination to “Baked”
  - Assign the new material to the new 3D Object
  - Don’t forget everything that can be baked must be static (including this emissive light source)

Light



3D Model



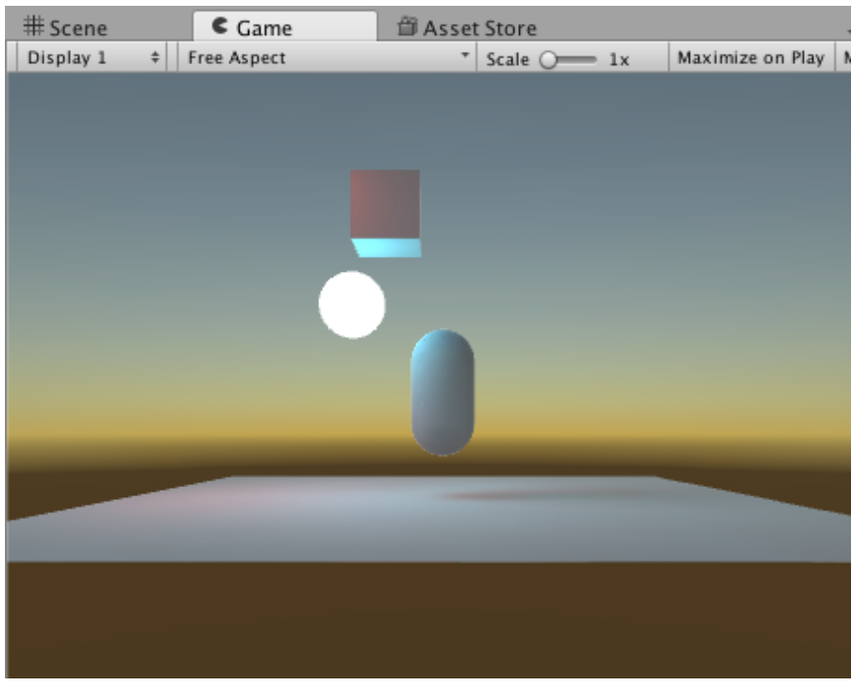
Camera

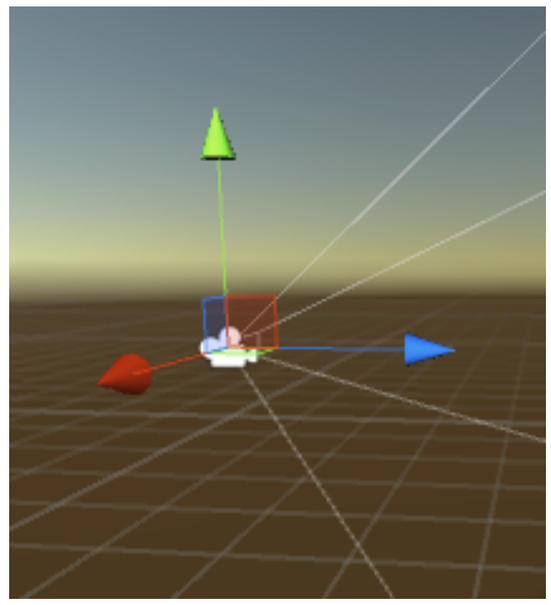
3D SCENE REVISITED

CAMERA

Camera Setting

- Game view will render from the camera point of view

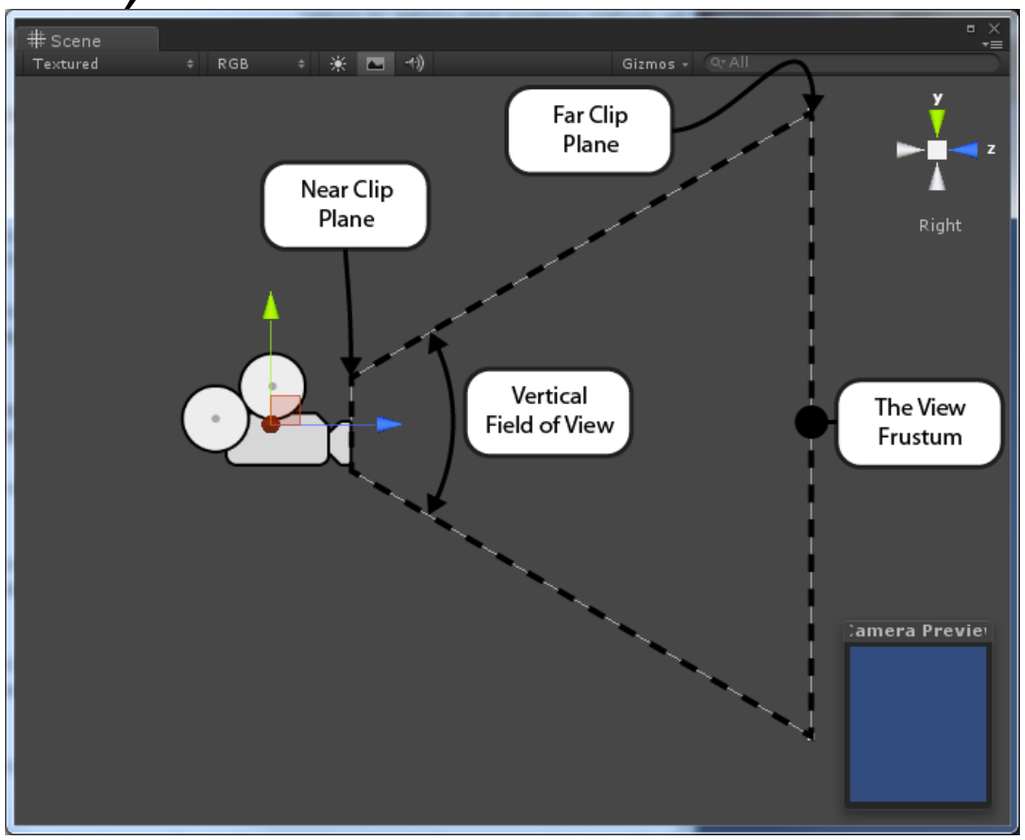




- Move Camera around using the same 3D transformations as 3D objects
- Projection choices: Perspective and Orthographic

Camera Setting

- Transformations (Check Camera Preview)
  - Position - use translation
  - Orientation - use rotation
- Clear Flags = Background Color
- Projections
  - Perspective (FOV - zoom feature)
  - Orthographic (Size - zoom feature)
- Clipping (your scene boundary that the camera can see)
  - Near
  - Far



Unity Camera Setting Lab

- Move your camera to the proper scene view
- Try background of skybox vs solid color
- Try perspective and orthographic projection
- Try adjust camera frustum (FOV, Near, Far, etc.)
- Add 1 more camera (from GameObject)
  - Try Viewport Rect (Range: 0-1, 0.5=half)
  - Try to adjust Viewport Rect to have 2 views (from 2 cameras)

Lab1

- Goal: Create a static 3D scene with different materials and light sources
- Create a nice scene and name your scene (with a story)
- Requirements:
  - Use only Unity 3D objects (+ transformations)
  - Use different materials
  - Use a varieties of light sources
  - Put Camera in the proper view
- Submission (10 points) - at the end of the class
  - Login to WikiSpace
  - Create a new wiki page with title "your name - lab1" e.g. tanasai - lab1
  - Tag your page with lab1_2016
  - Post your result as an image link
  - Add your scene name as a text

Next Class

- Download “Corridor Lighting Example” from Asset Store (http://u3d.as/cAD)
- Ready to import to the new project

Deadline Alert

Tutorial 1 Due NEXT WEEK!