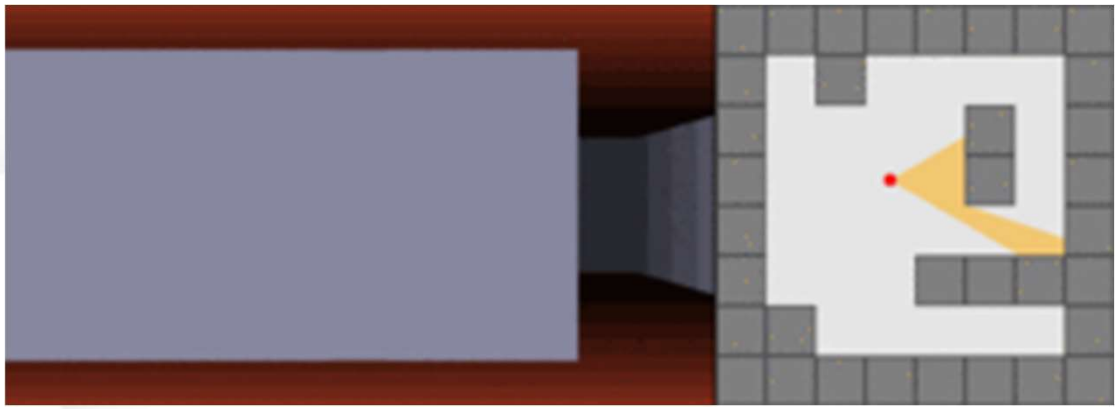
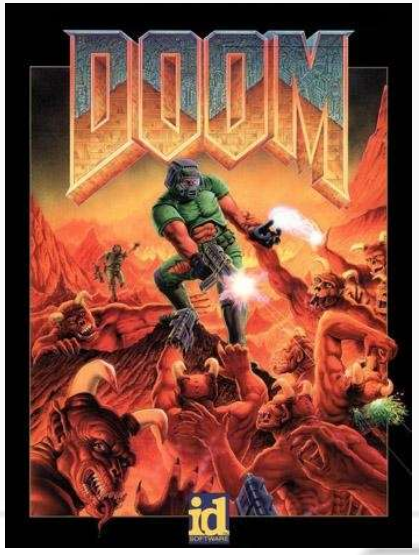
3D Game Programming
3D Primitive

Ming-Te Chi
Department of Computer Science,
National Chengchi University

Wolfenstein 3D 1992 [game]
Doom 1993 [game]



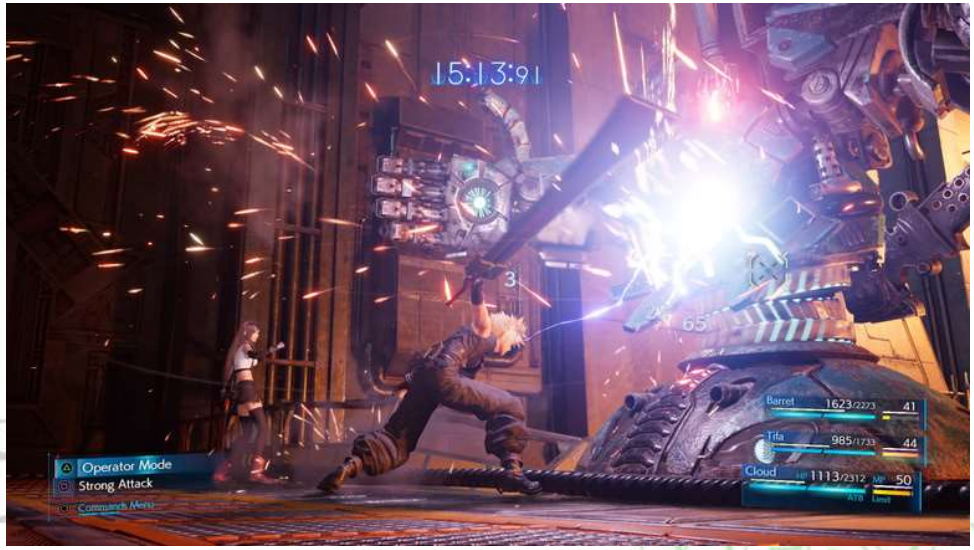


(Top) Wolfenstein 3D

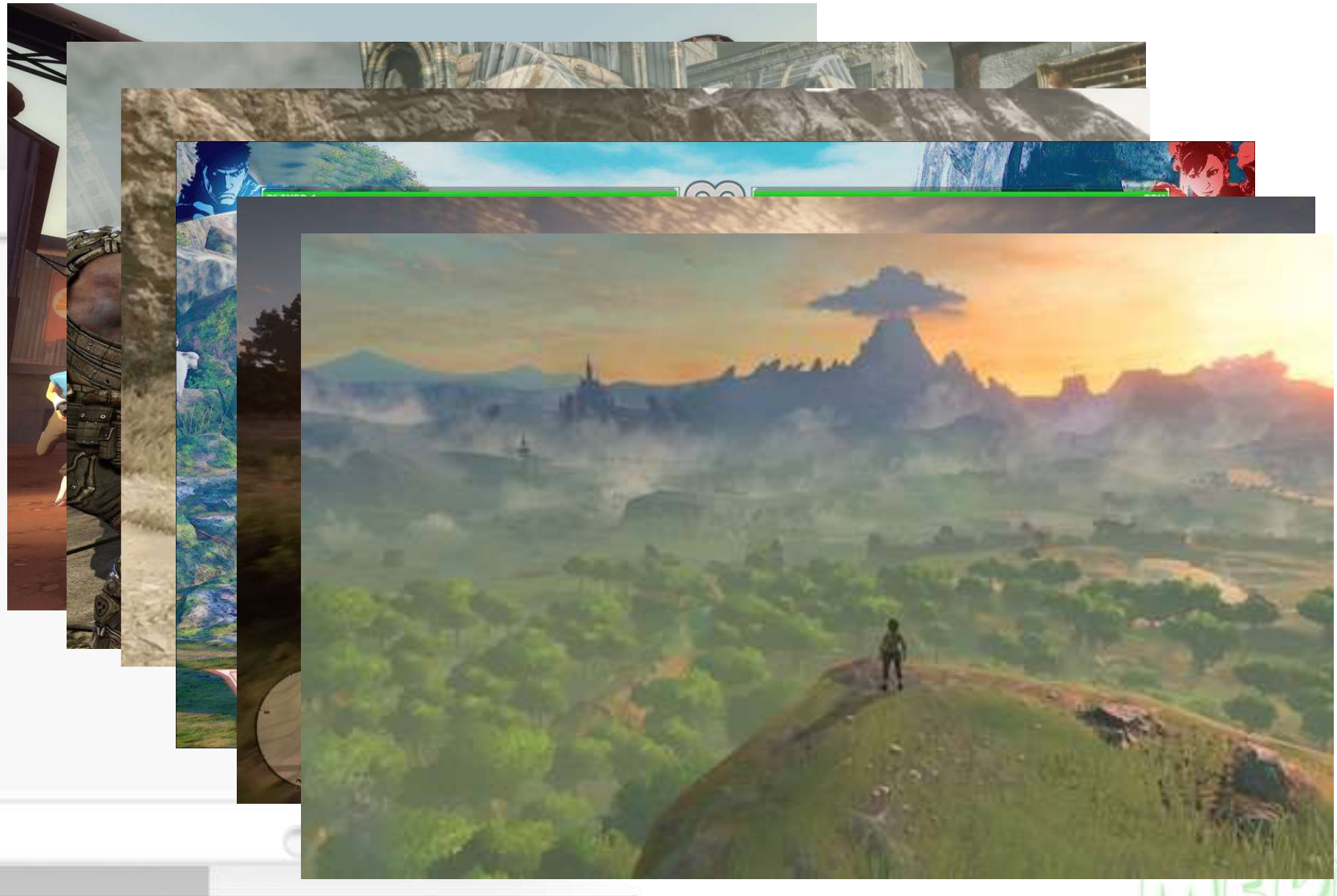
(Bottom) A landmark 1993 first-person shooter (FPS) video game by *id Software*.

3D Graphics – evolution

3D Graphics – evolution

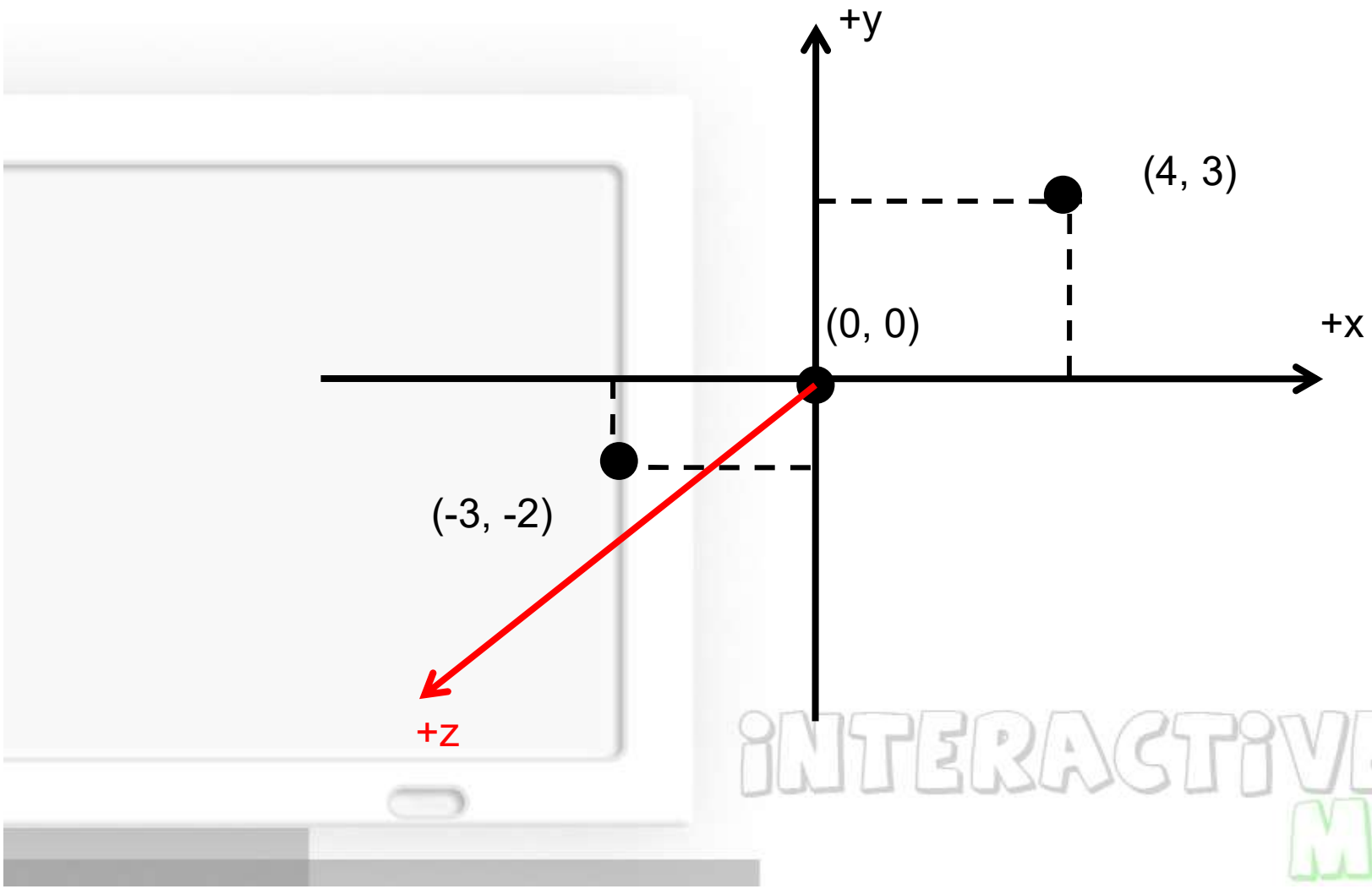


3D Graphics

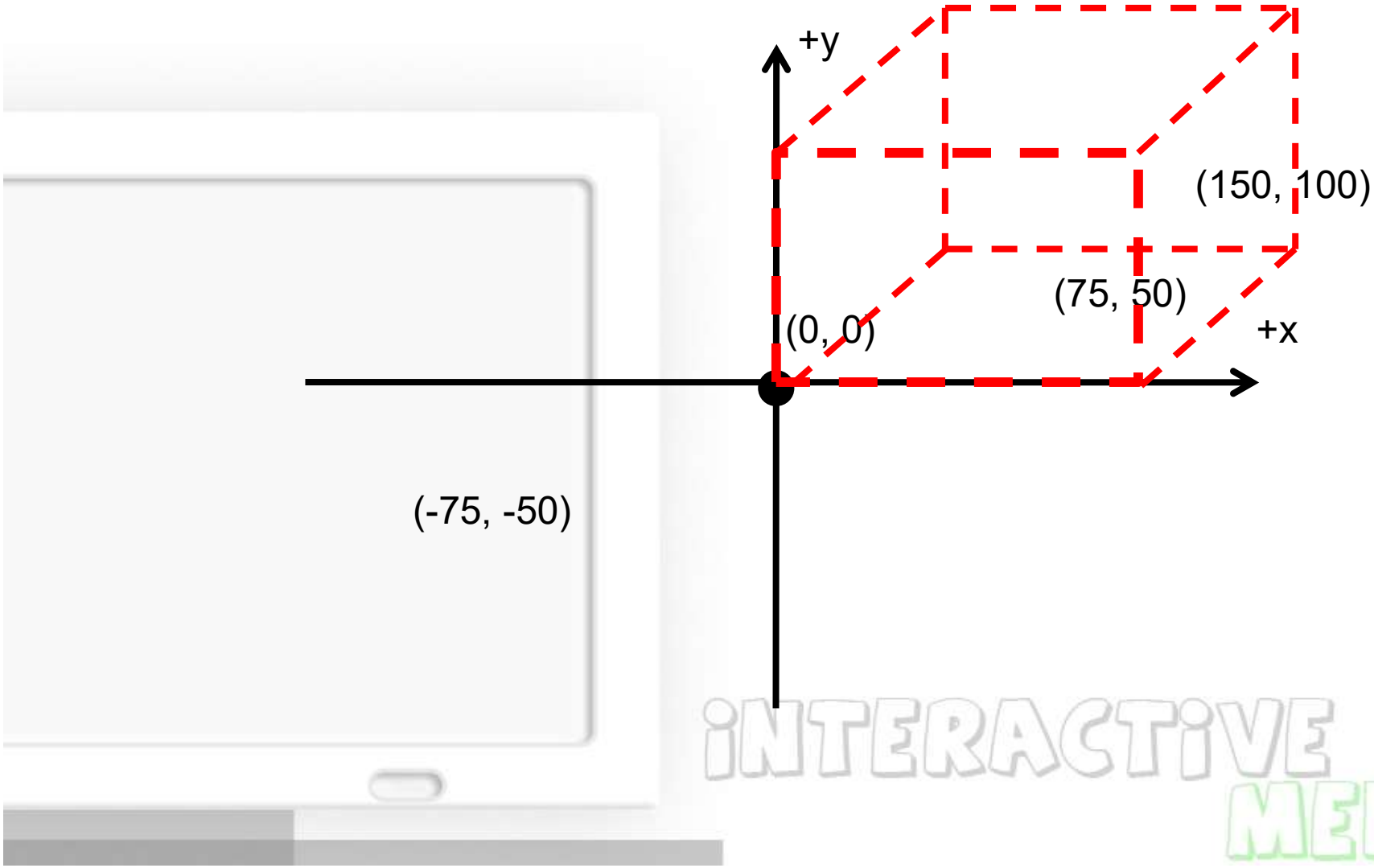


BASIC

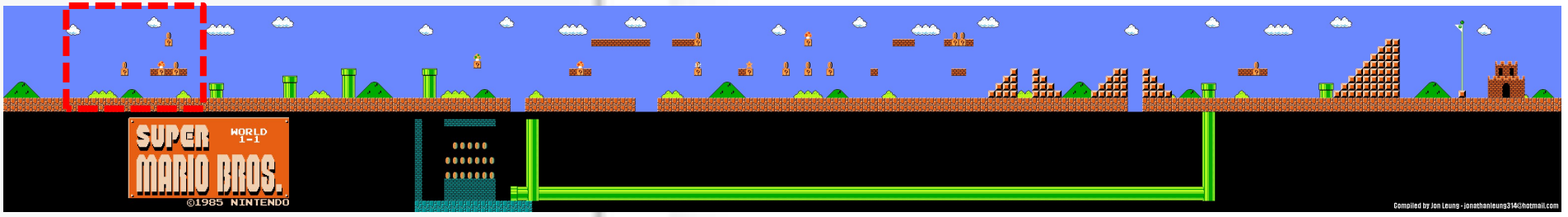
Cartesian Plane



Coordinate Clipping



2D Game world



Super Mario Bros.　Nintendo

Super Paper Mario
Nintendo Wii Gameplay

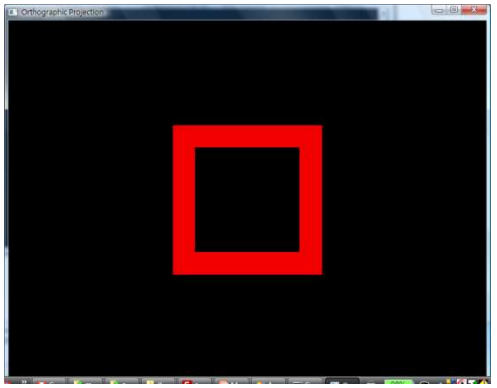
3D Game world

Super Mario 3D Land. Nintendo

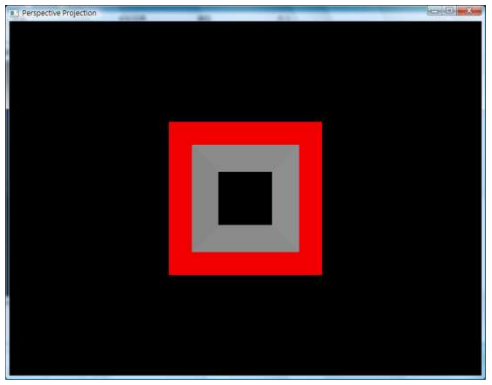
Chris Pratt - Super Mario Remake - TRAILER

Projection (camera)

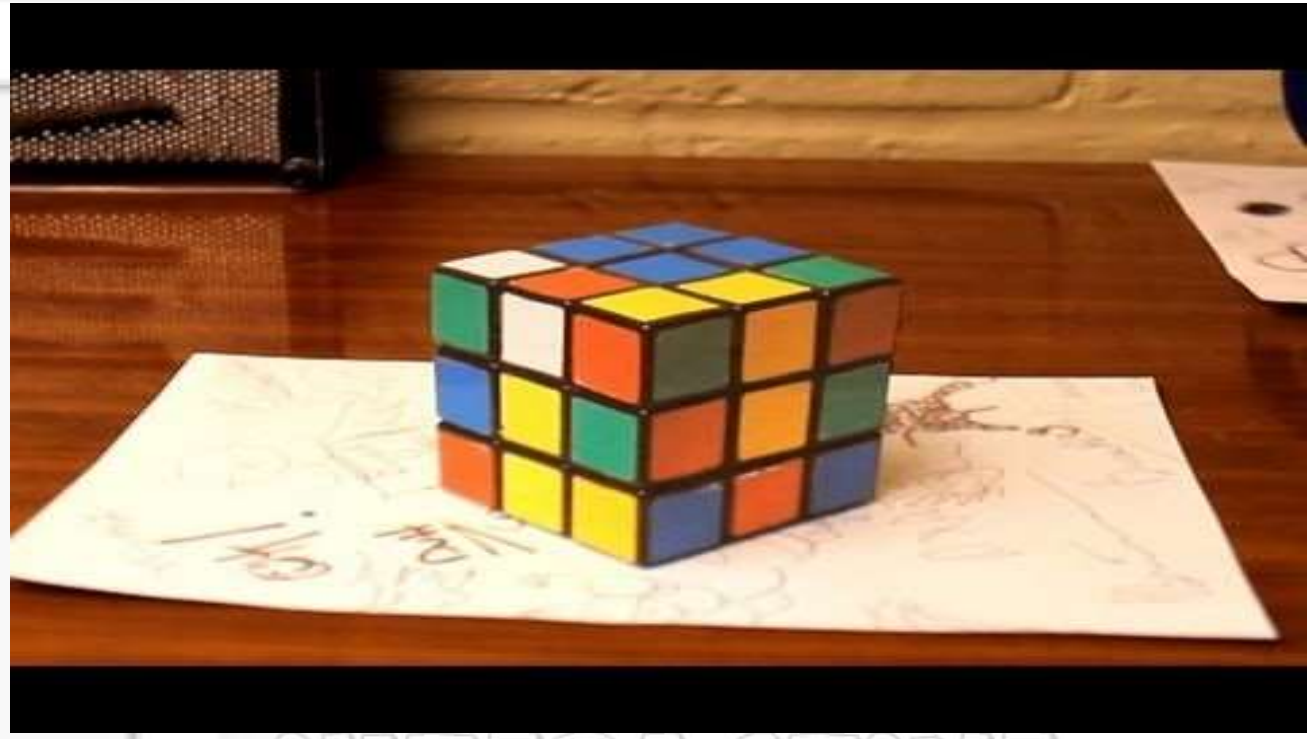
- Getting 3D to 2D
  - Orthographic projections



  - Perspective projections



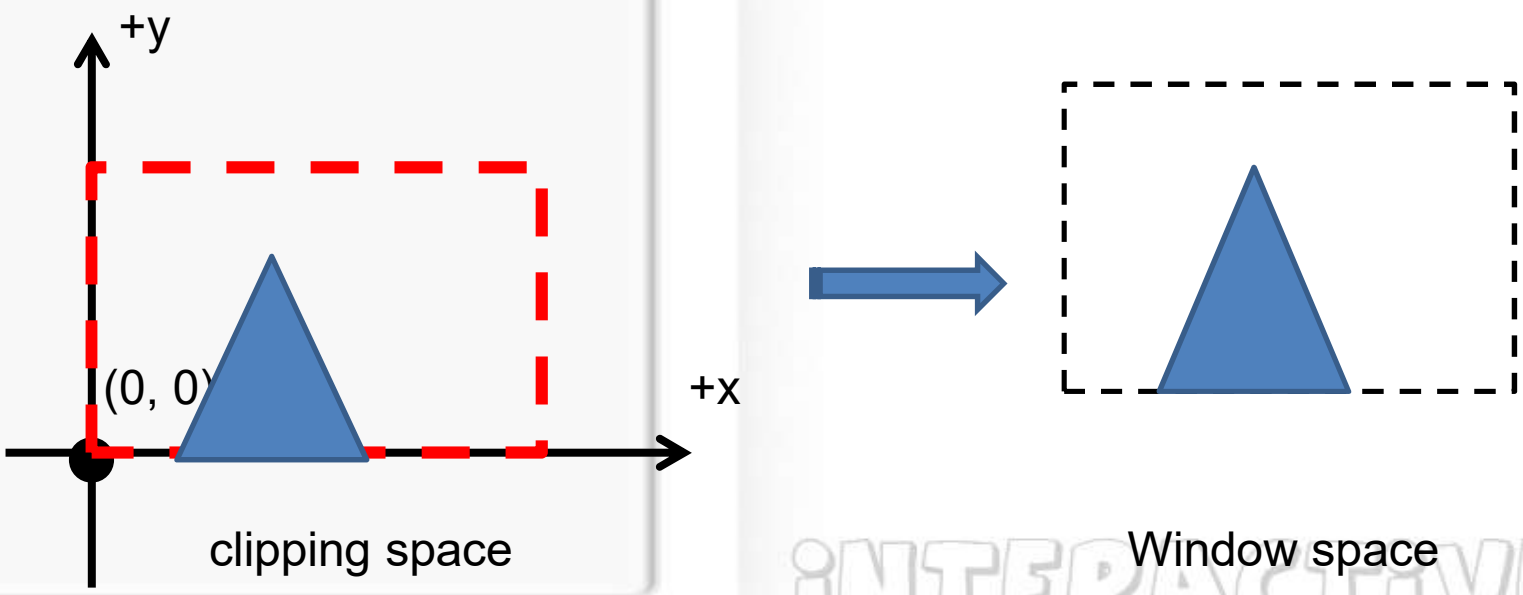
Amazing Anamorphic Illusions!



https://youtu.be/tBNHPk-Lnkk

Viewport (camera)

- Mapping drawing coordinates to windows coordinates



Representing Visuals

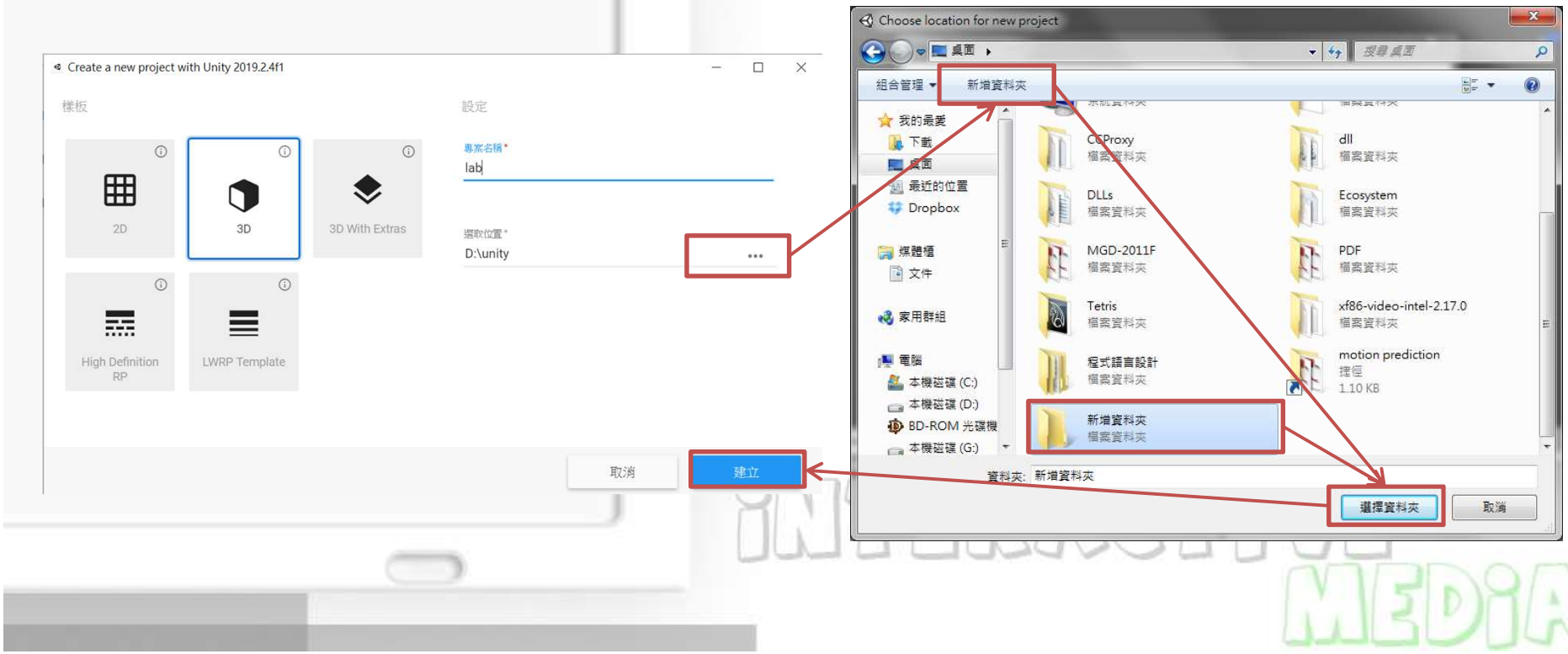
- **3D objects**
  - Mesh: geometry
  - Materials
    - Shader
    - Texture maps
- **Lighting**
- Visual effect

Start Unity in 3D

建立場景及基本物件

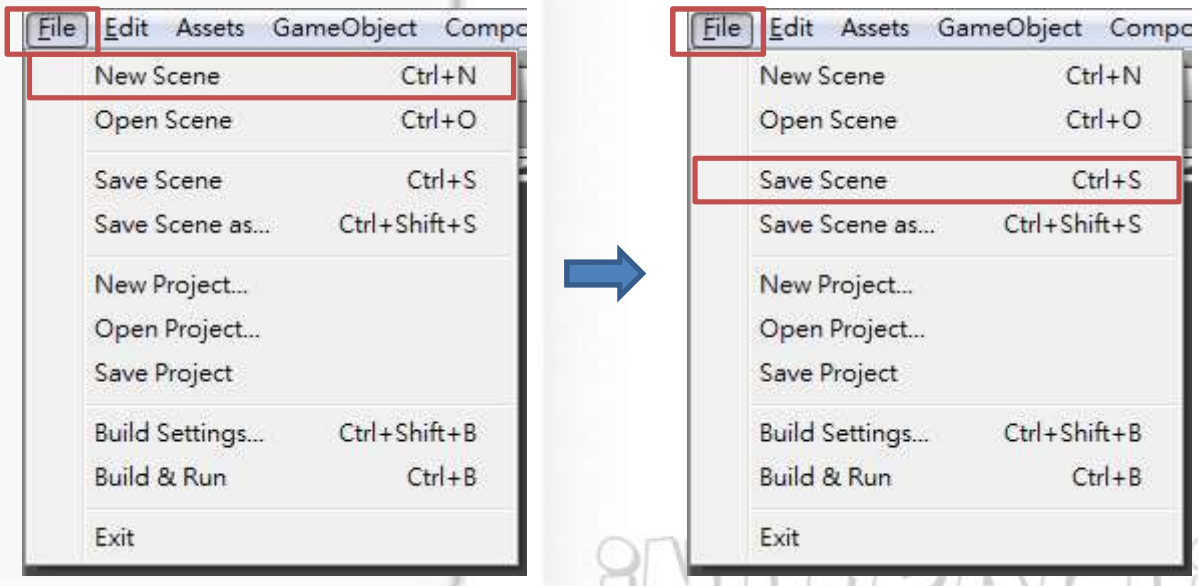
建立專案

– 新增或選擇要存放專案的資料夾→[Create]



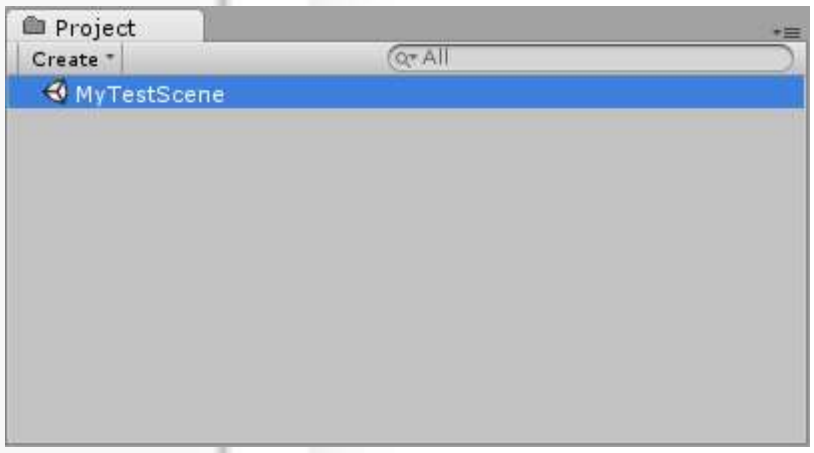
建立場景及基本物件

- 建立場景
  - 選擇[File]→[New Scene]
  - 之後可先儲存場景，[File]→[Save Scene]



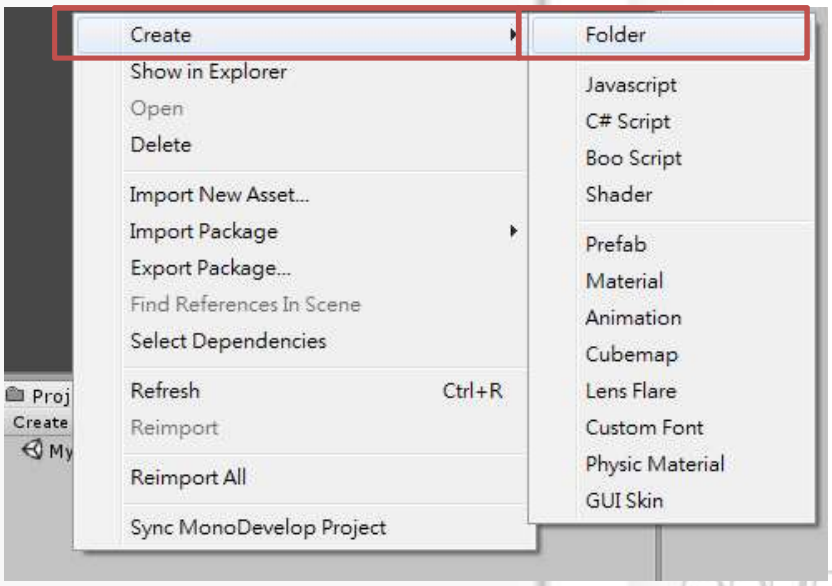
建立場景及基本物件

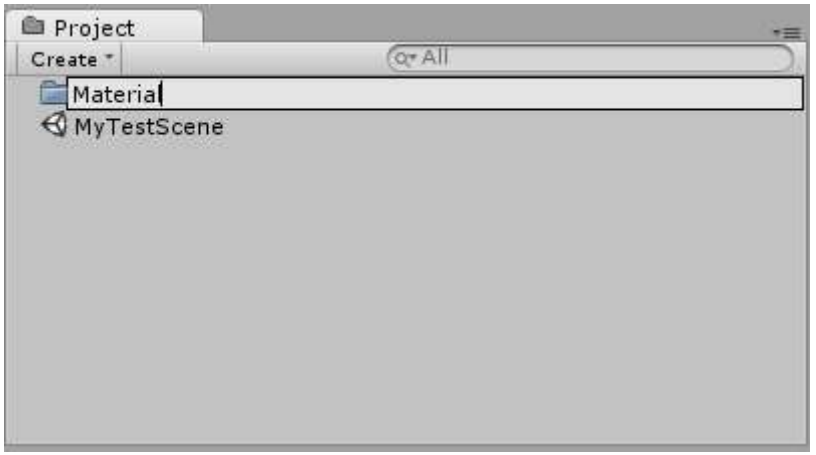
- 建立場景
  - 通常場景內檔案會存放在專案目錄的Assets下
  - 儲存完成後場景檔案會出現在Project畫面中



建立場景及基本物件

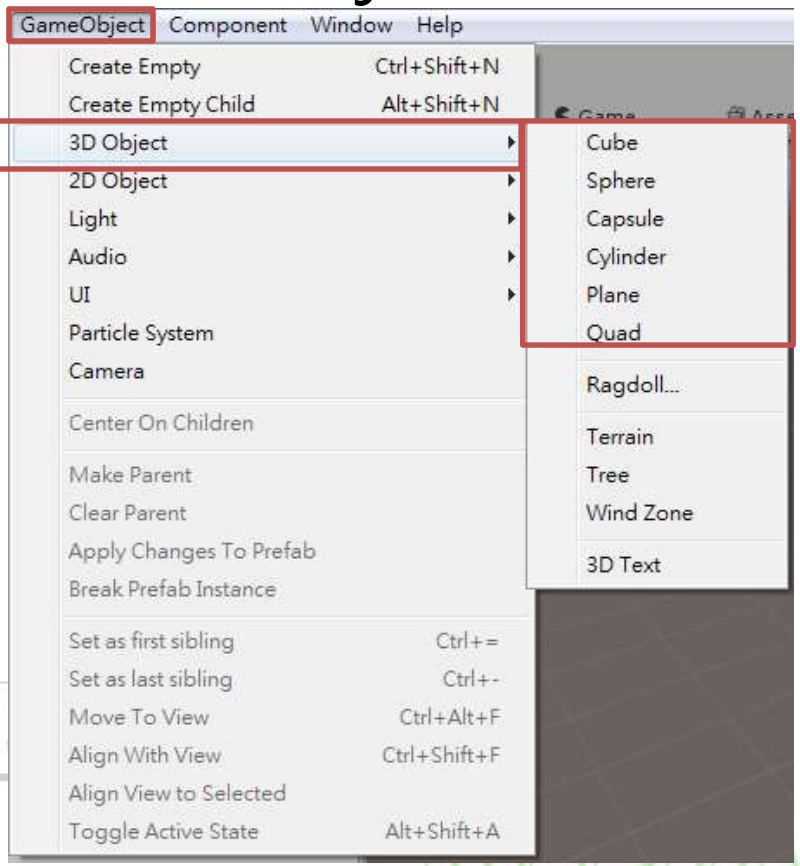
- 基本物件與材質
  - 新增資料夾存放材質以方便管理
    - 在Project畫面空白處[右鍵]→[Create]→[Folder]



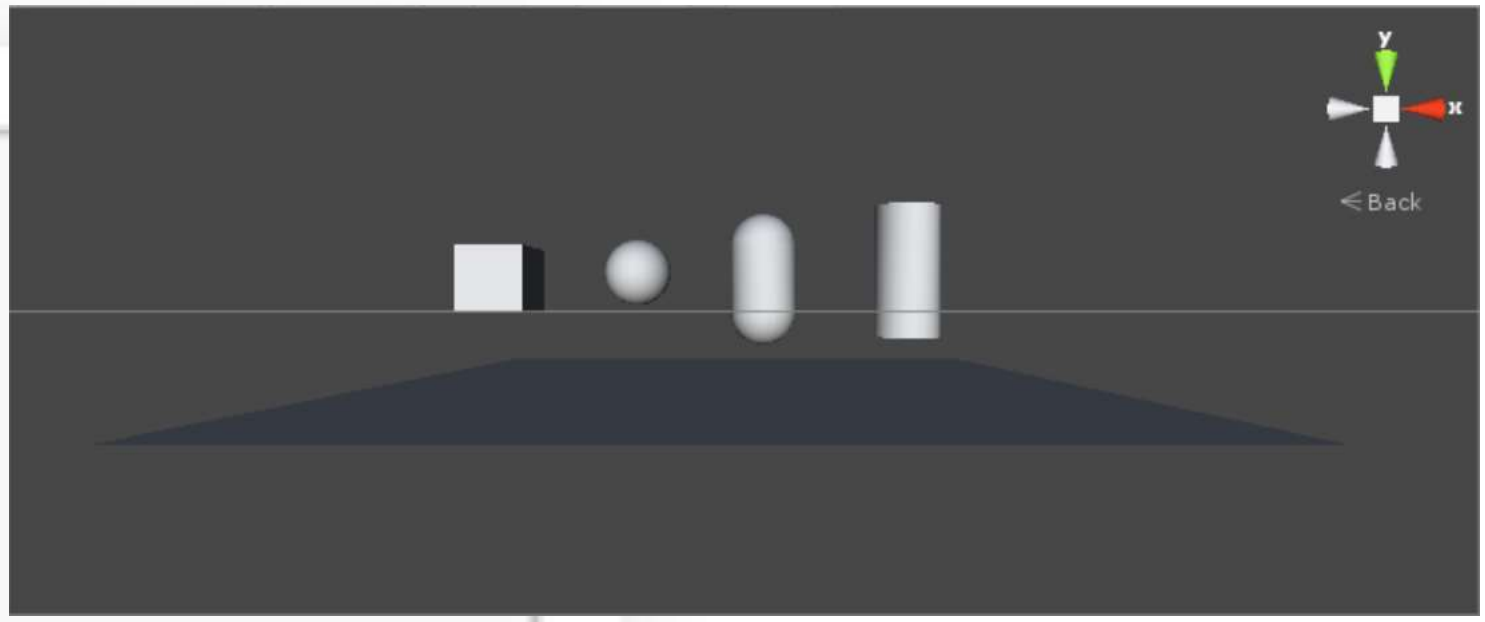


建立場景及基本物件

- 基本物件與材質
  - 選擇左上角[GameObject]→[3D Object]→
    - [Cube](方體)
    - [Sphere](球體)
    - [Capsule](膠囊體)
    - [Cylinder](圓柱體)
    - [Plane](平面)
    - [Quad](單位長度的平面)
  - 也可在[Hierarchy]畫面中
  - 按[Create]新增

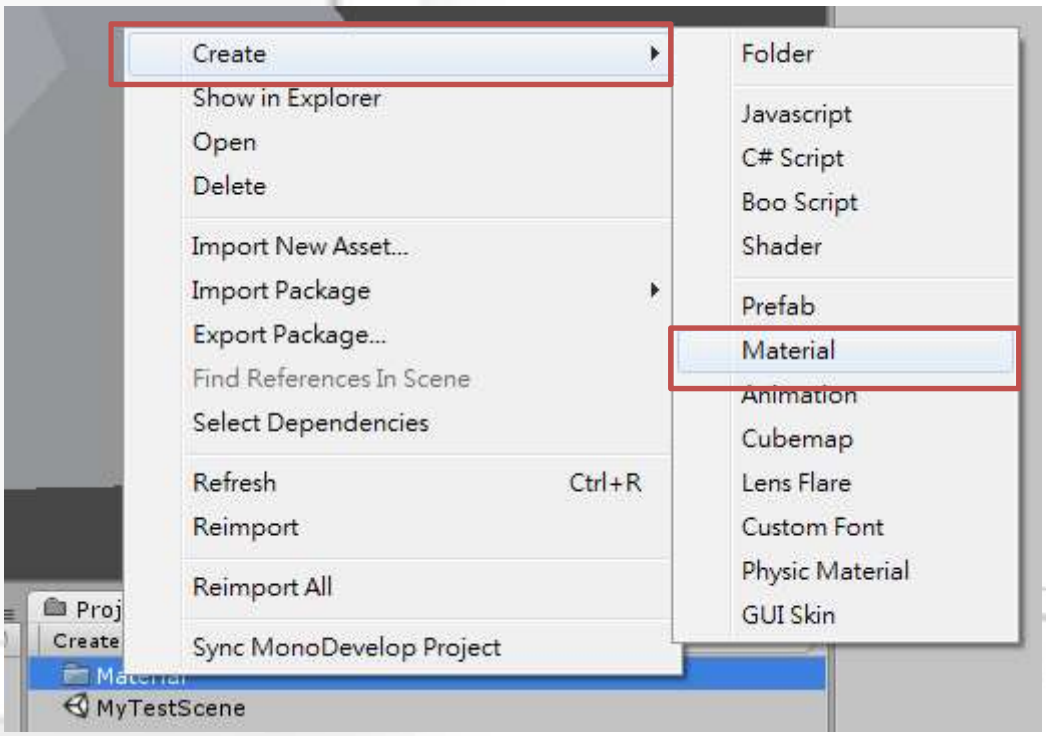


建立場景及基本物件



建立場景及基本物件

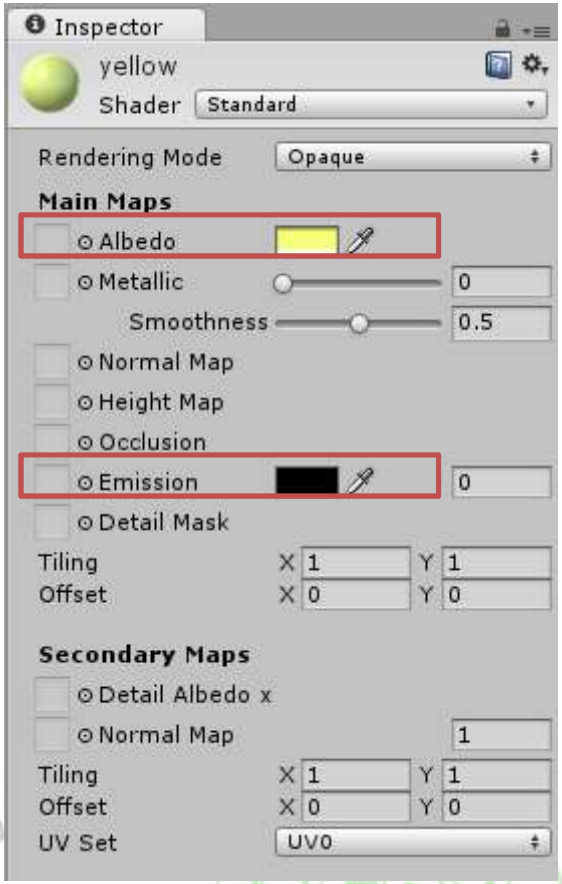
- 建立新材質，準備套用到物件上
  - 在Project畫面中剛剛新增的資料夾上按[右鍵]→[Create]→[Material]



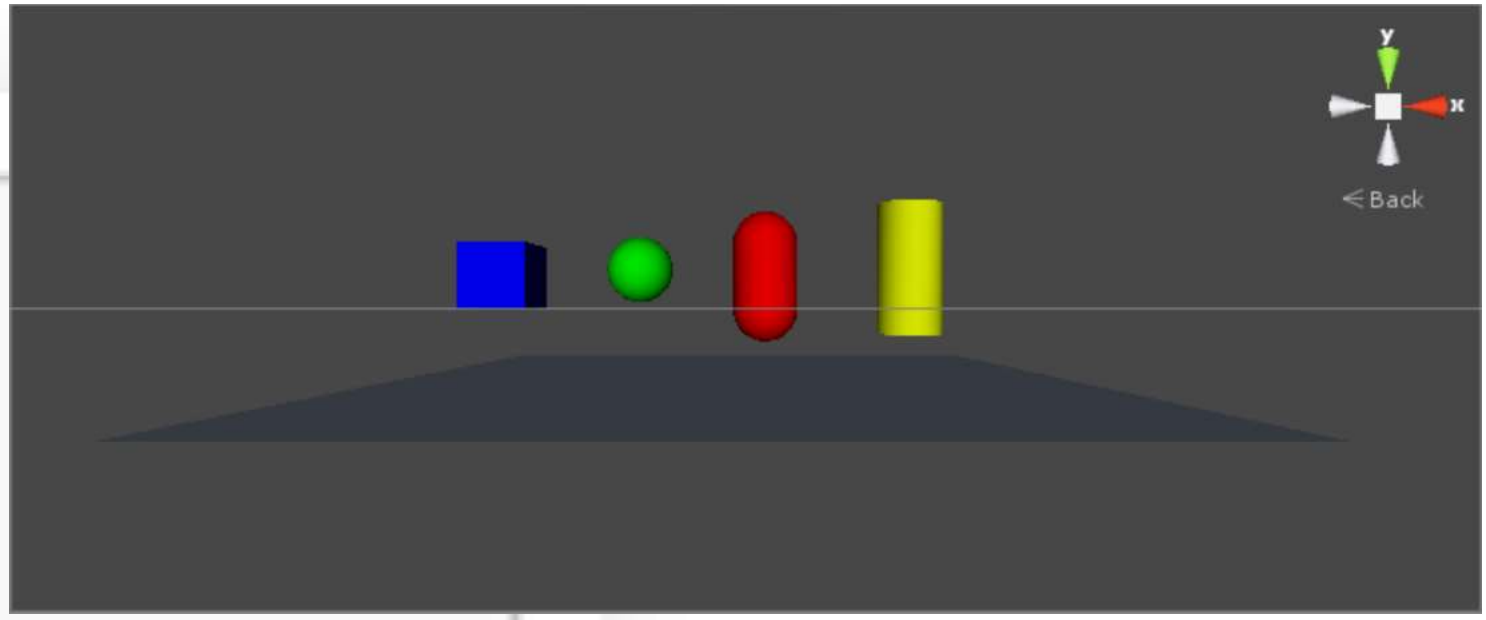
建立場景及基本物件

– 可在Inspector畫面中調整材質類型及顏色
  - Albedo: 反照率(顏色)
  - Emission: 在沒有燈光時仍發光

– 之後把材質拖到Hierarchy畫面的物件中，或直接拖到畫面中的物件，可直接套用材質

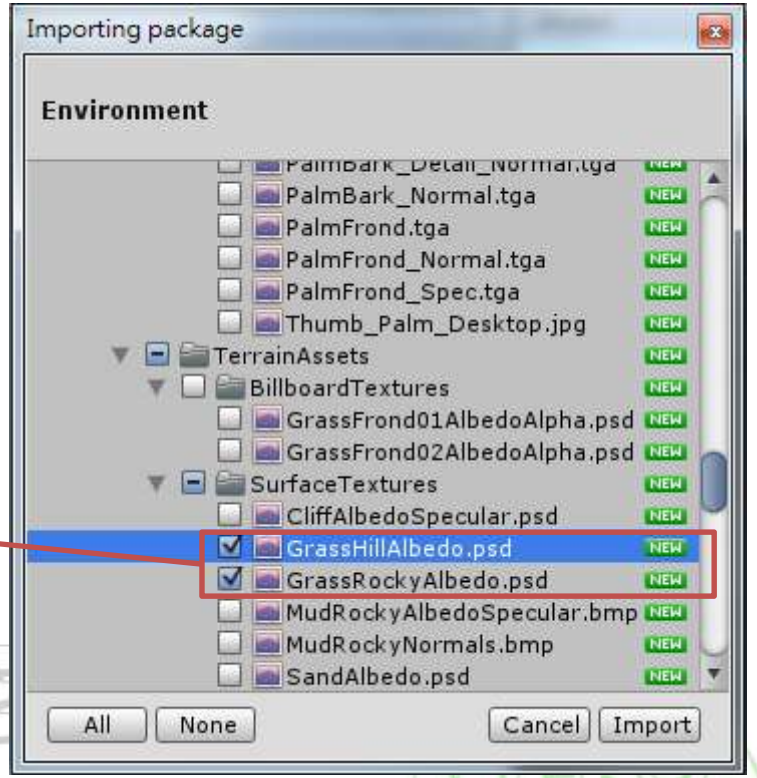


建立場景及基本物件



建立場景及基本物件

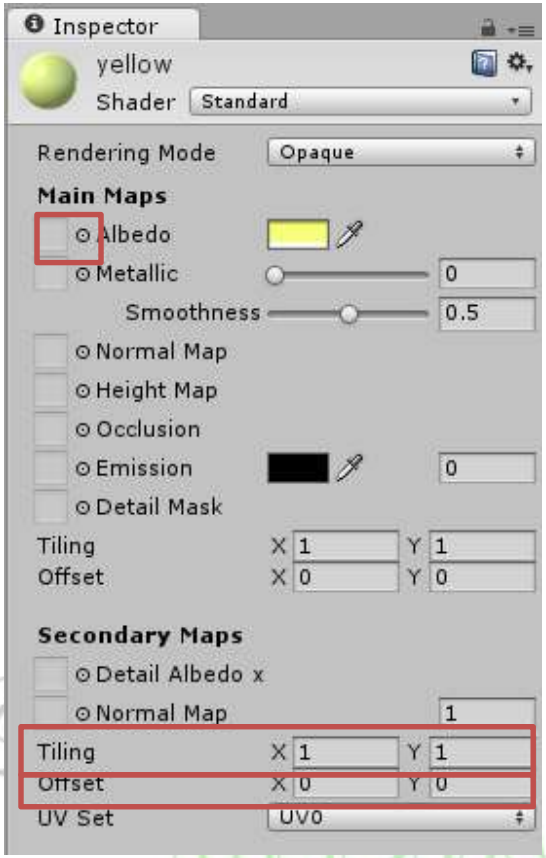
– 預設的地形材質圖可在Project視窗中[右鍵]→[Import Package]→[Environment]，之後按[Import]匯入



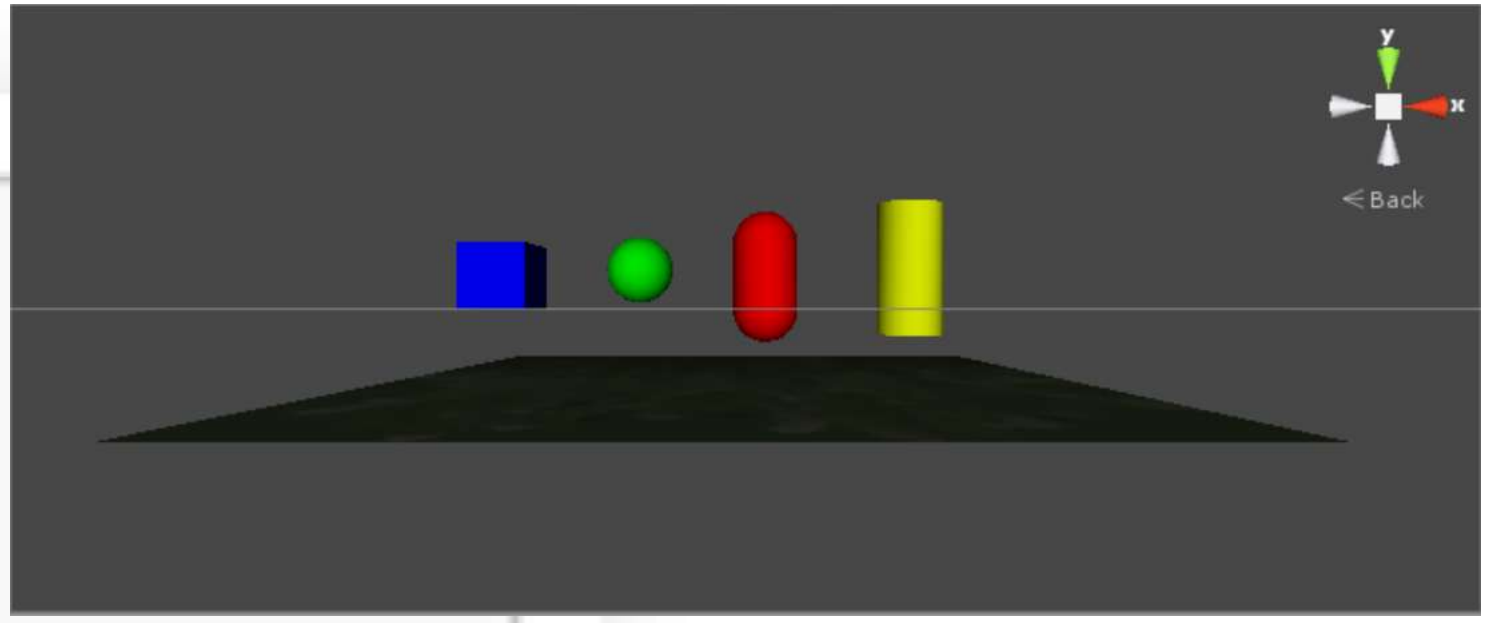
這次LAB只會用到這兩張其中一張!

建立場景及基本物件

- 點選灰[Albedo]左邊的圓圈可設定材質貼圖
- 可修改Tiling及Offset來設定材質圖的重複次數及貼圖起始位置

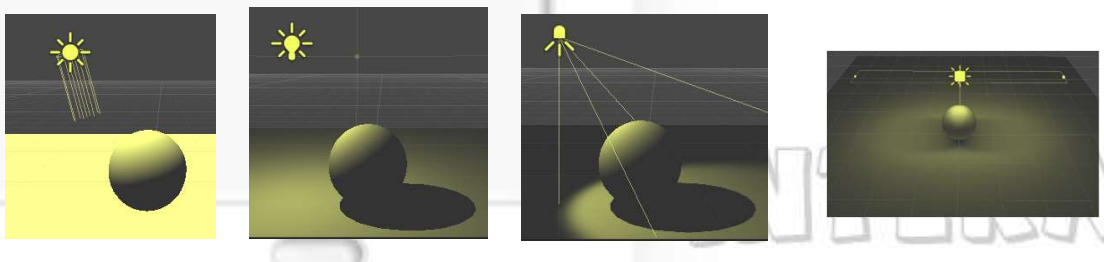


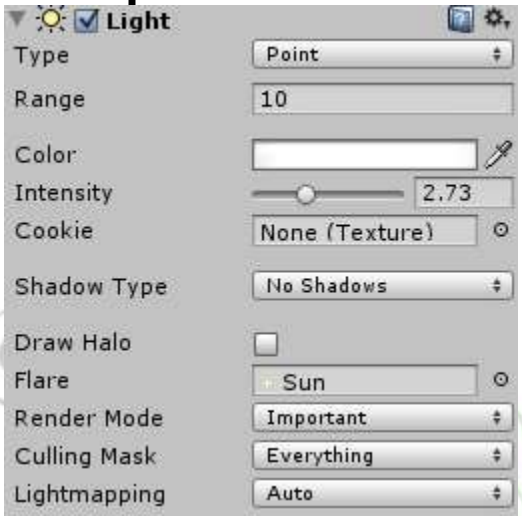
建立場景及基本物件

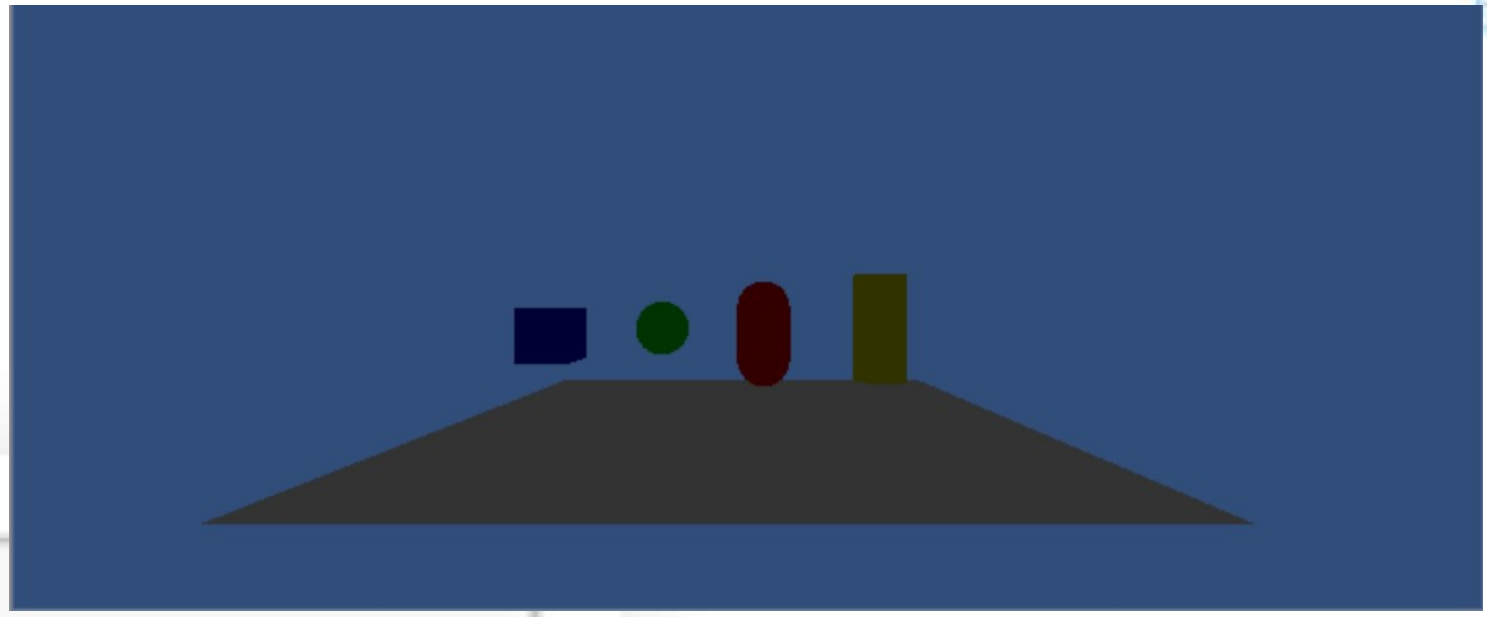


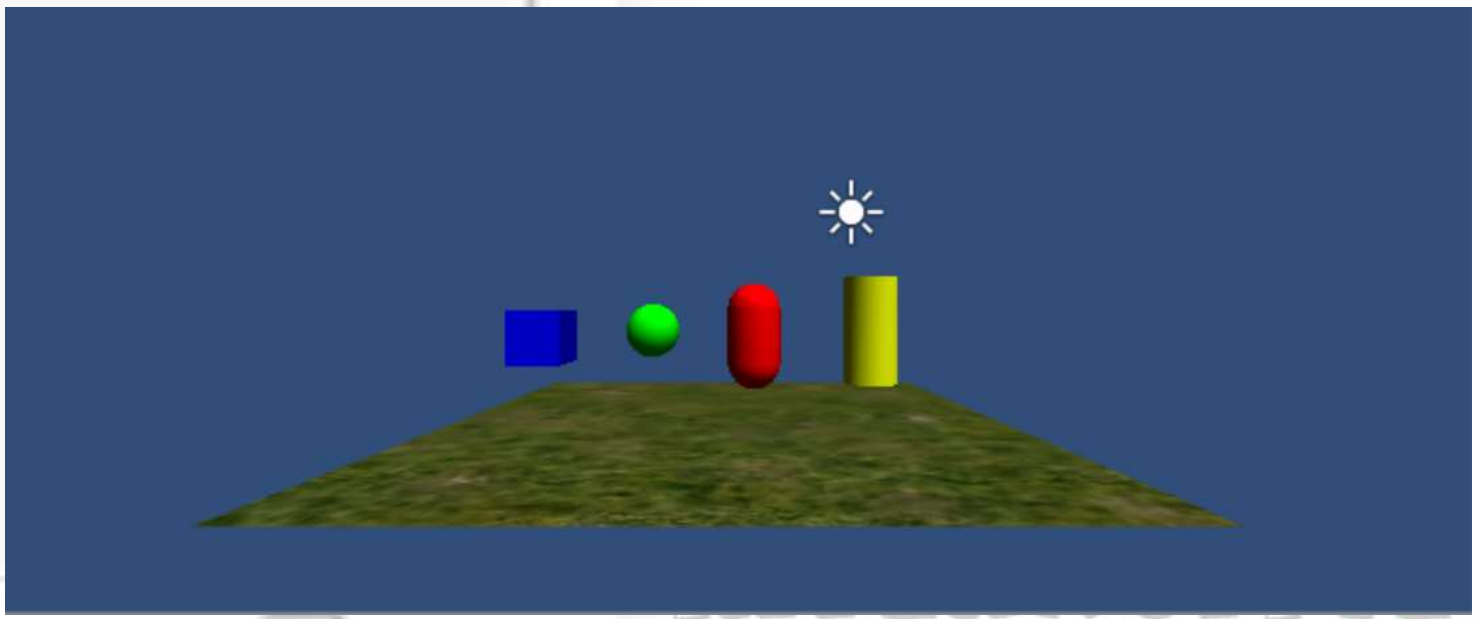
光源使用與設定

- 左上方[Game Object]→[Light]→
  - [Directional Light] (平行光)
  - [Point Light] (點光源)
  - [Spotlight] (聚光燈)
  - [Area light] (區域光)
- 點擊光源，可以移動、旋轉，Inspector中可改變各項屬性



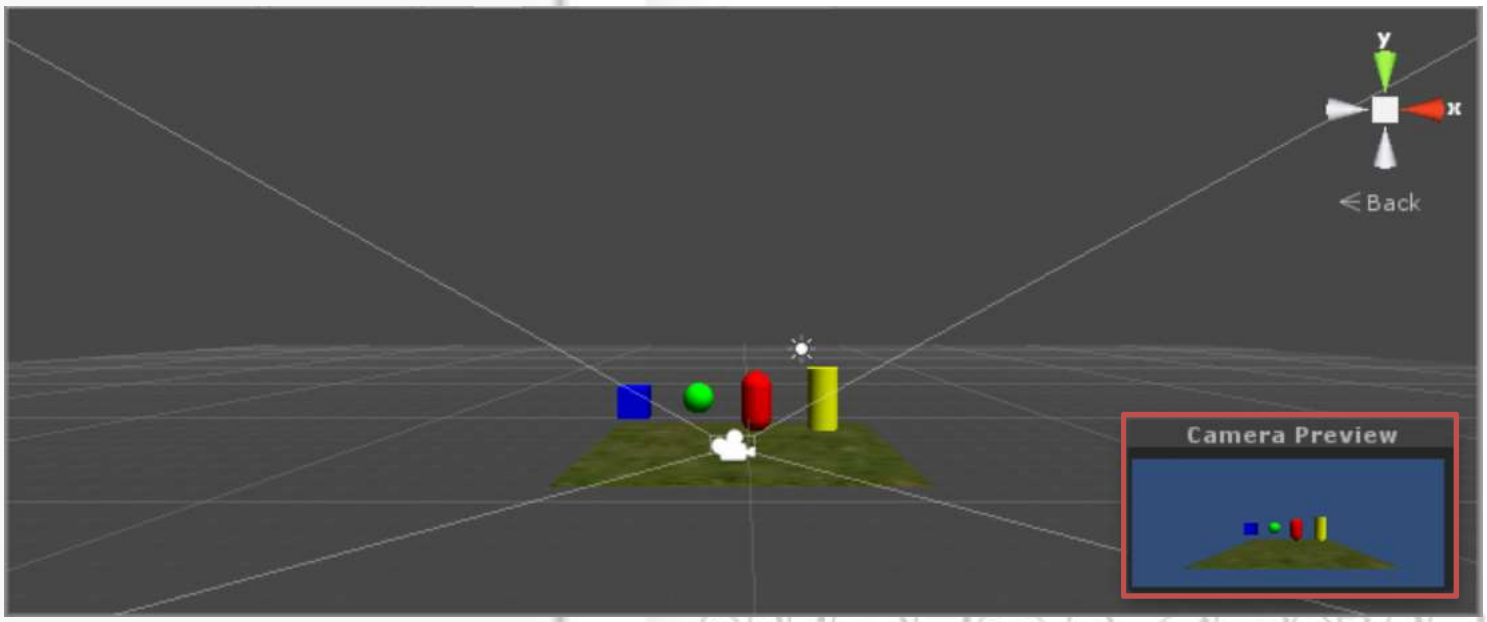






攝影機使用與設定

- Main Camera: 遊戲預設視角
  - 點擊場景中的攝影機，可顯示預覽畫面

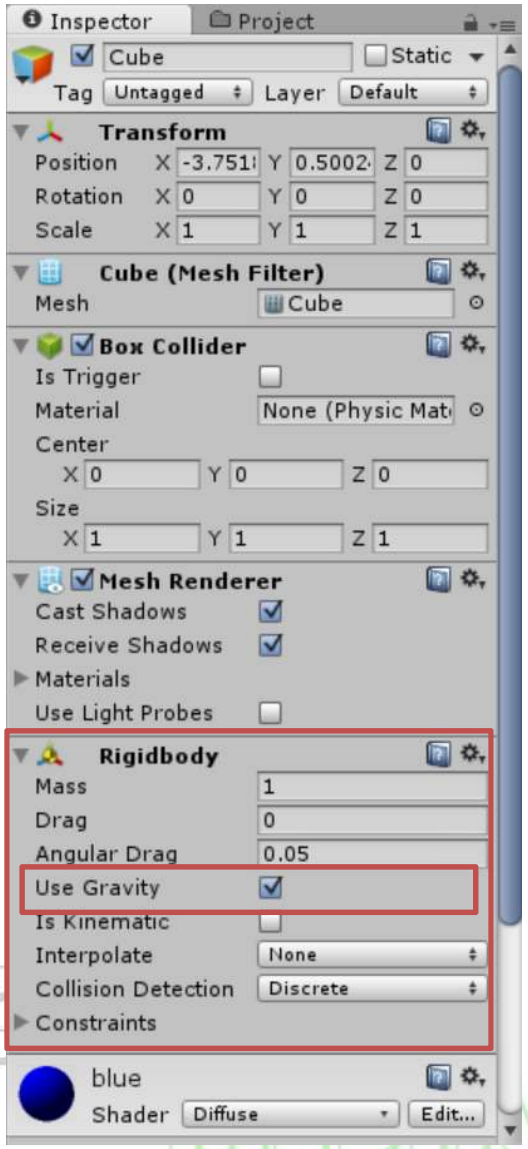


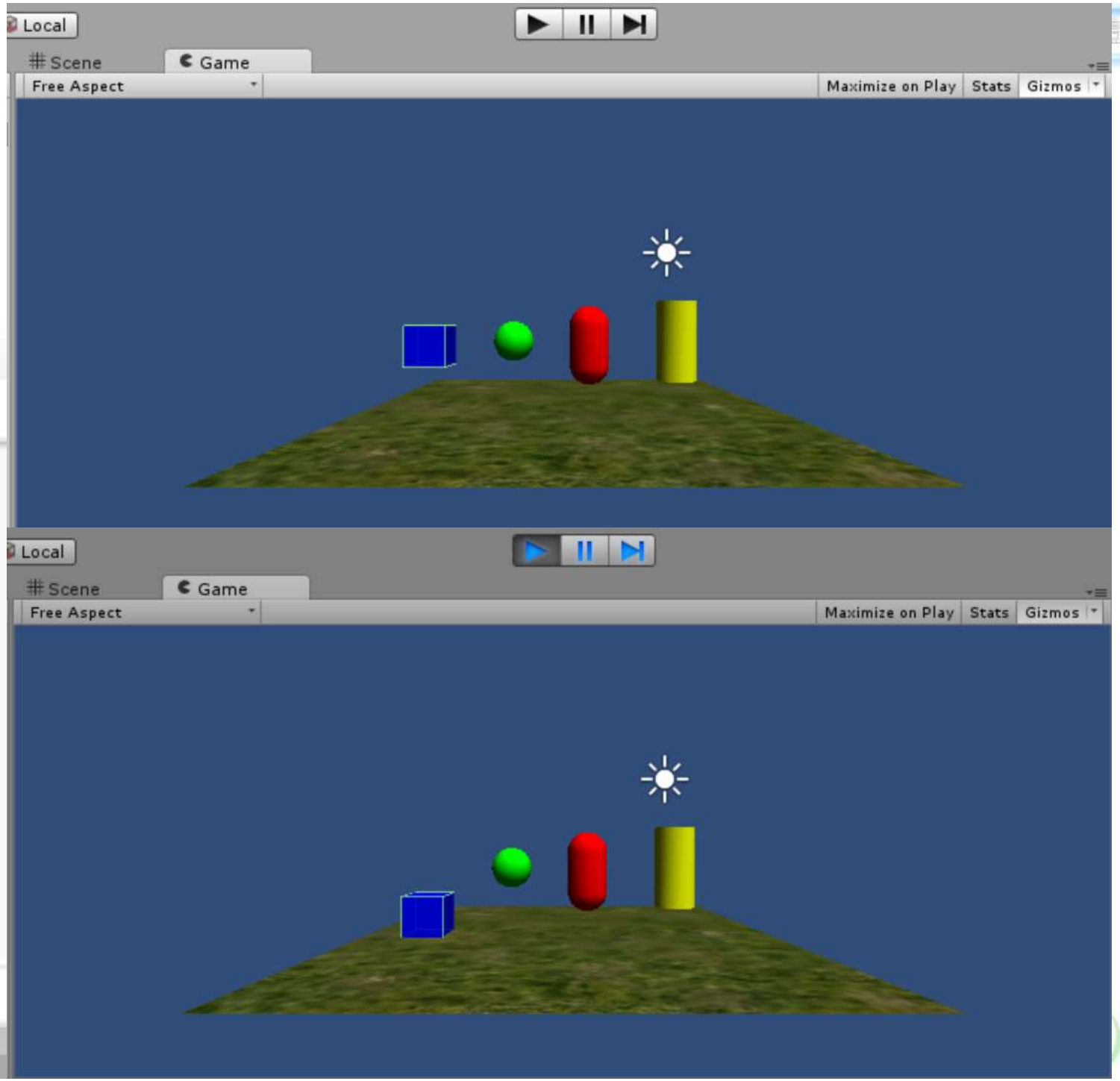
物理元件使用與設定

- 首先我們先選擇一個物件(立方體)，並從物理選單中，新增一個Rigidbody到所選的物件上

物理元件使用與設定

- 確認屬性欄中，剛體元件的重力參數(Use Gravity)是否已勾選。
- 勾選表示當遊戲開始時，該物體會受重力往下移動

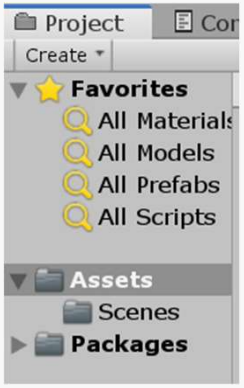


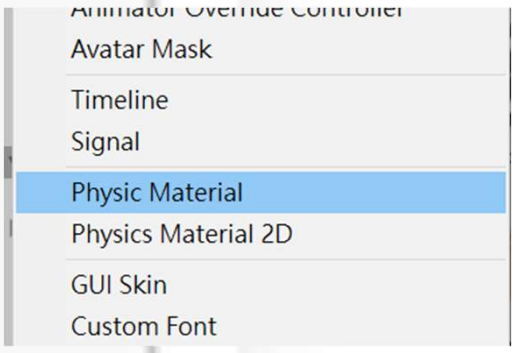

Local
Scene
Game
Free Aspect
Maximize on Play
Stats
Gizmos
Local
Scene
Game
Free Aspect
Maximize on Play
Stats
Gizmos

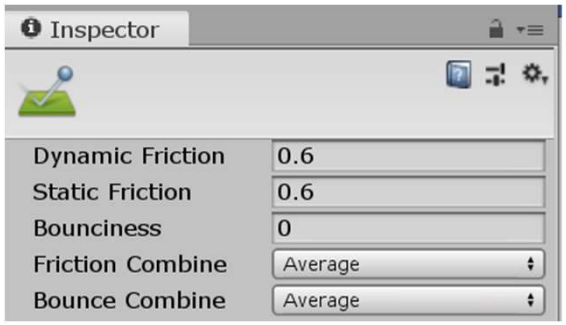
物理元件使用與設定-Physic material

Physics material

- In Project tab, Create -> Physic Material
- Bounciness 越大反彈力越大

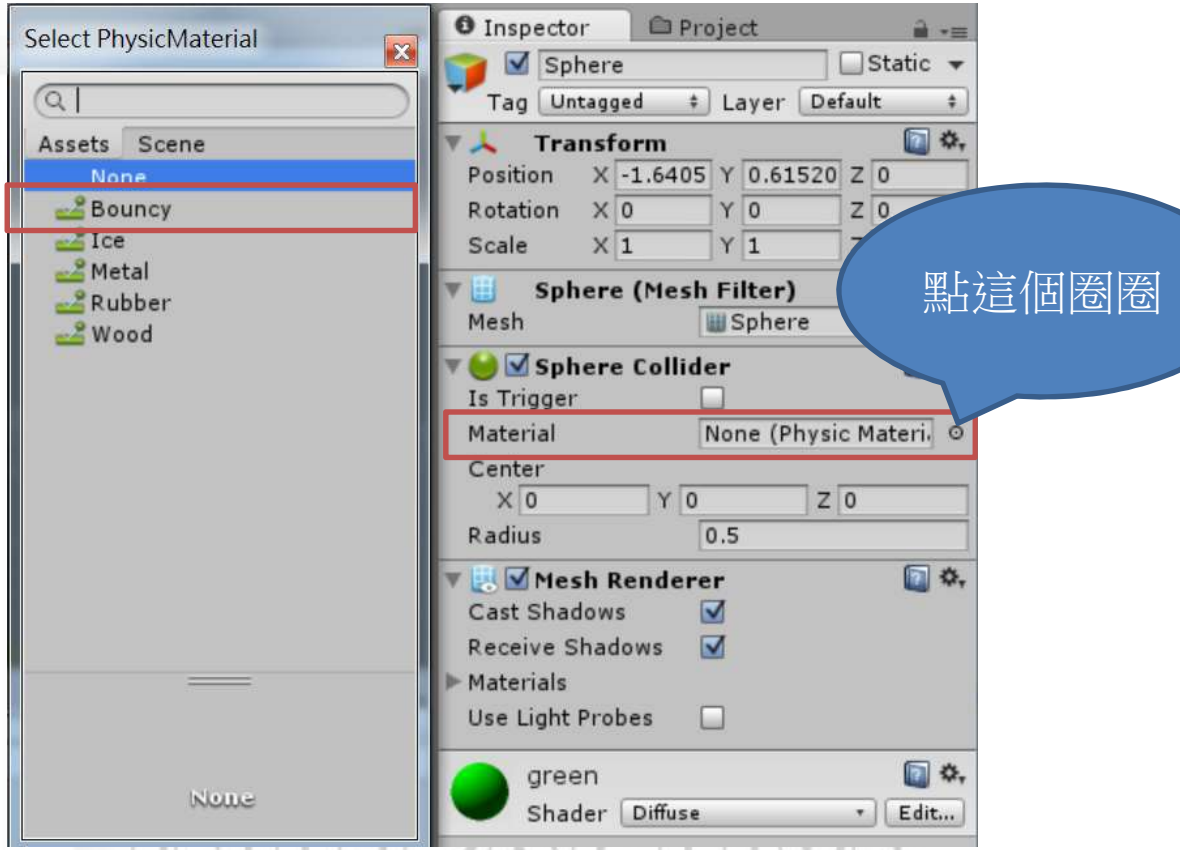


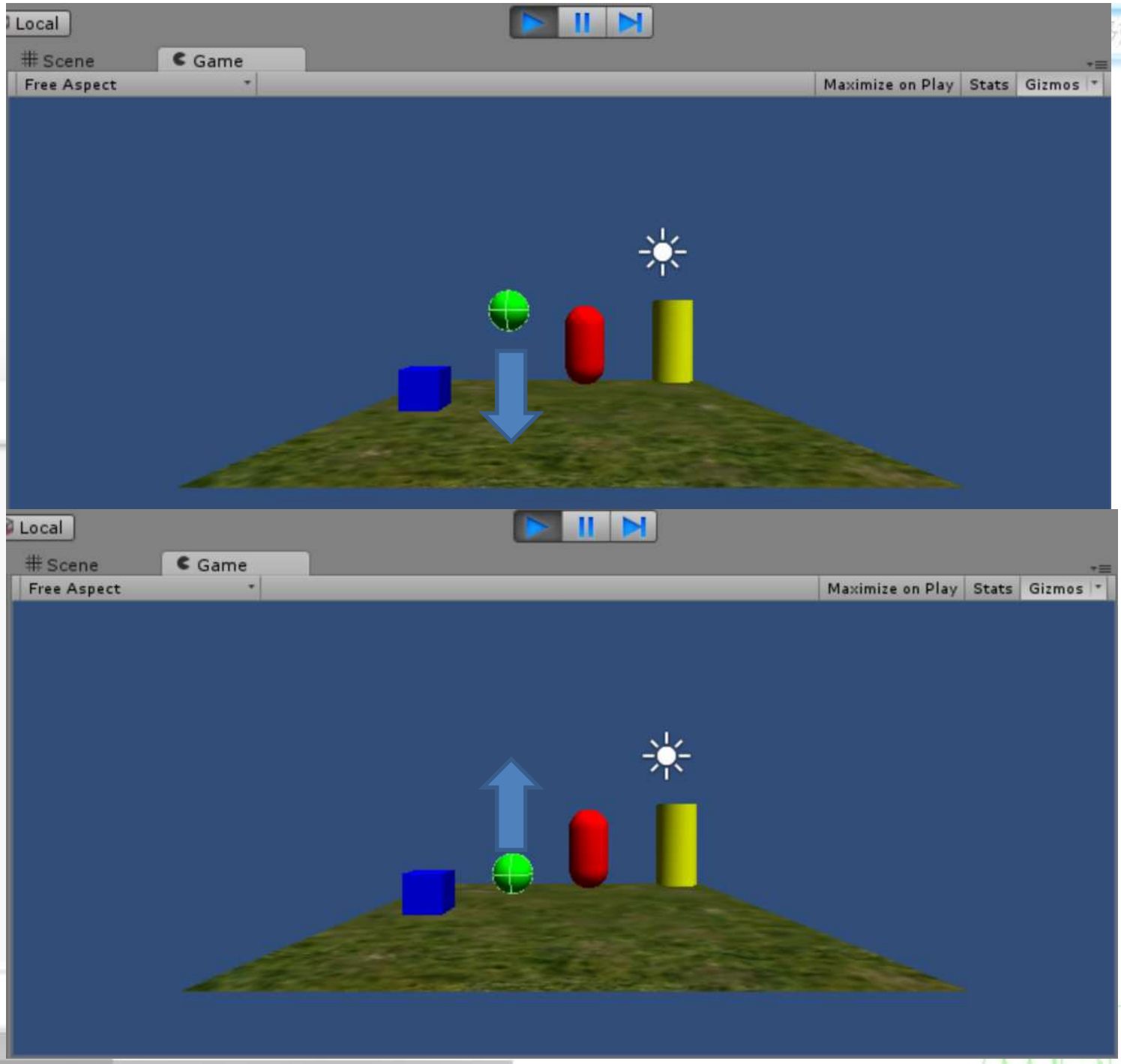


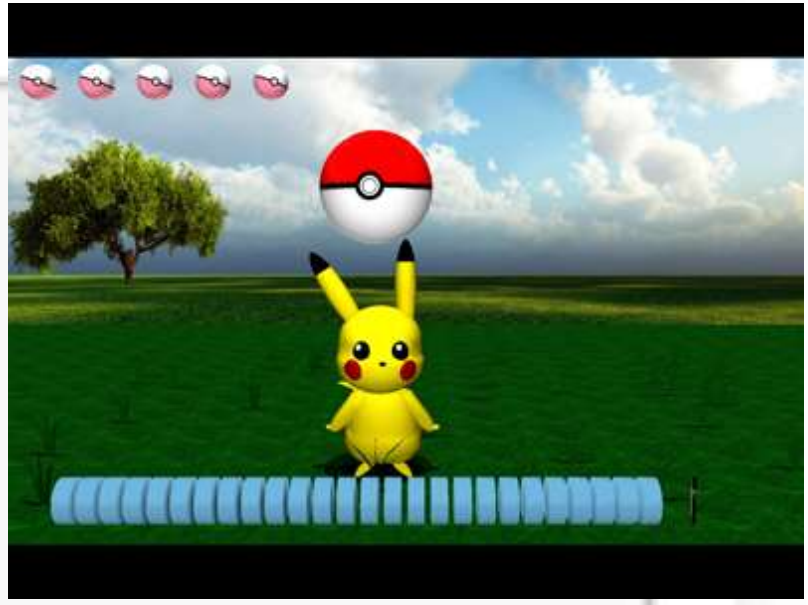


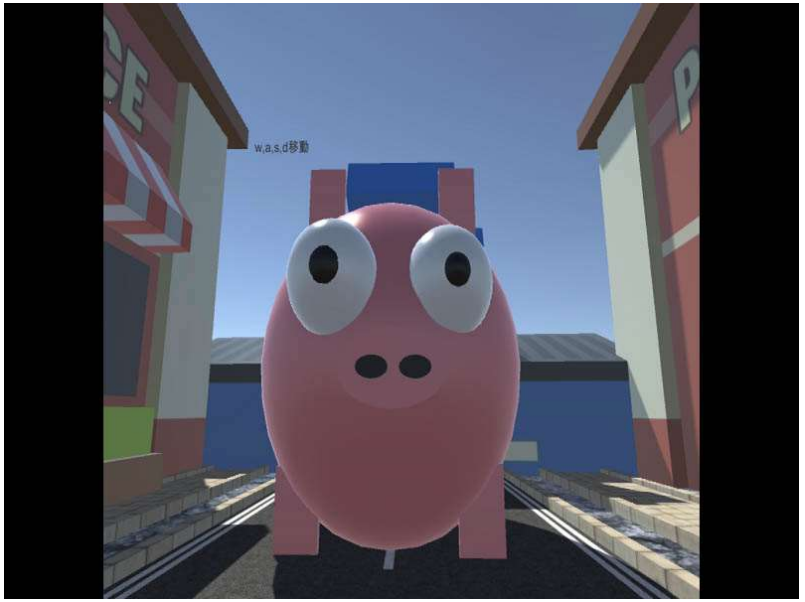
物理元件使用與設定

- 選取plane
- 新增物理材質於碰撞器中，選擇bouncy材質來測試
- Plane不需要有rigidbody




Local
Scene
Game
Free Aspect
Maximize on Play
Stats
Gizmos
Local
Scene
Game
Free Aspect
Maximize on Play
Stats
Gizmos



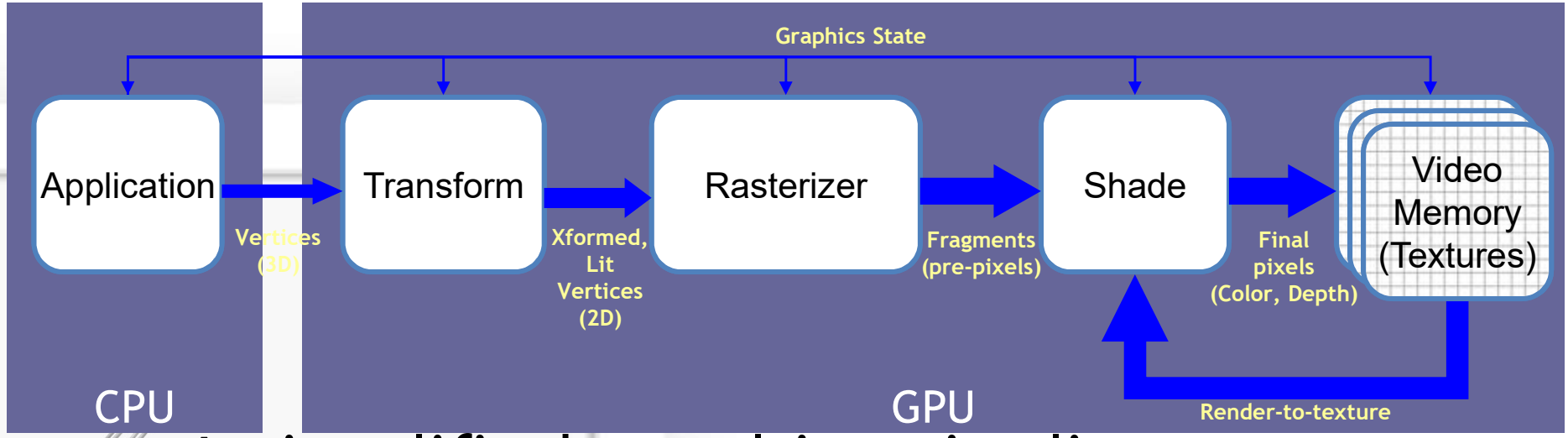


INTERACTIVE MEDIA

PIPELINE

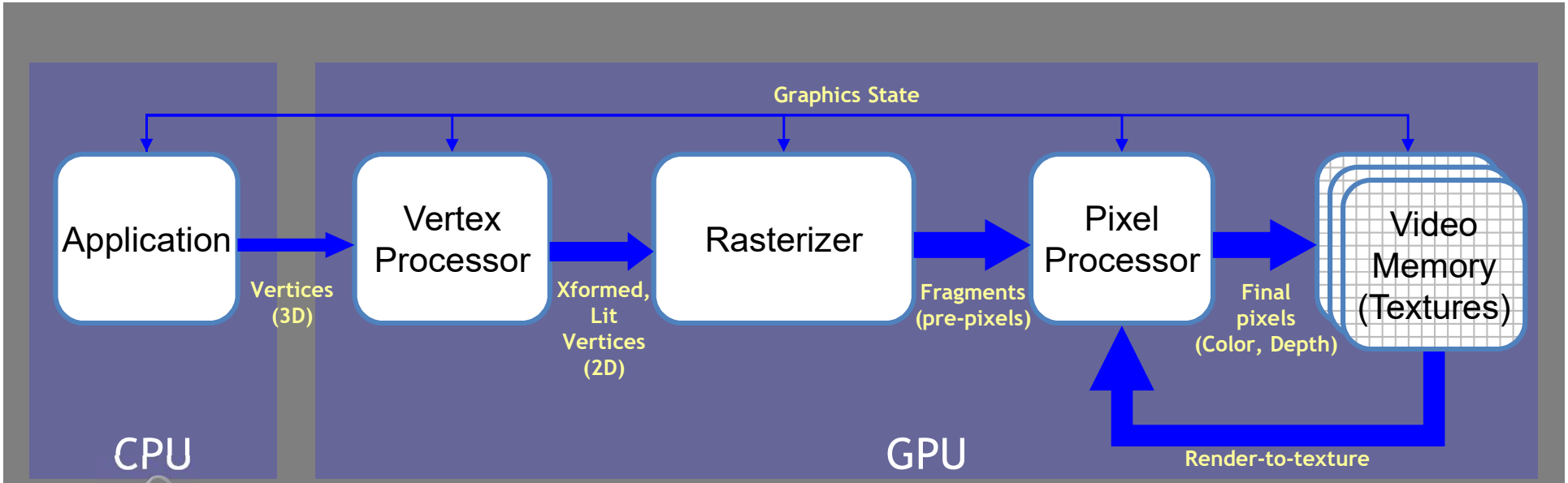
參考資料

GPU Fundamentals: The Graphics Pipeline



- A simplified graphics pipeline
  - Note that pipe widths vary
  - Many caches, FIFOs, and so on not shown

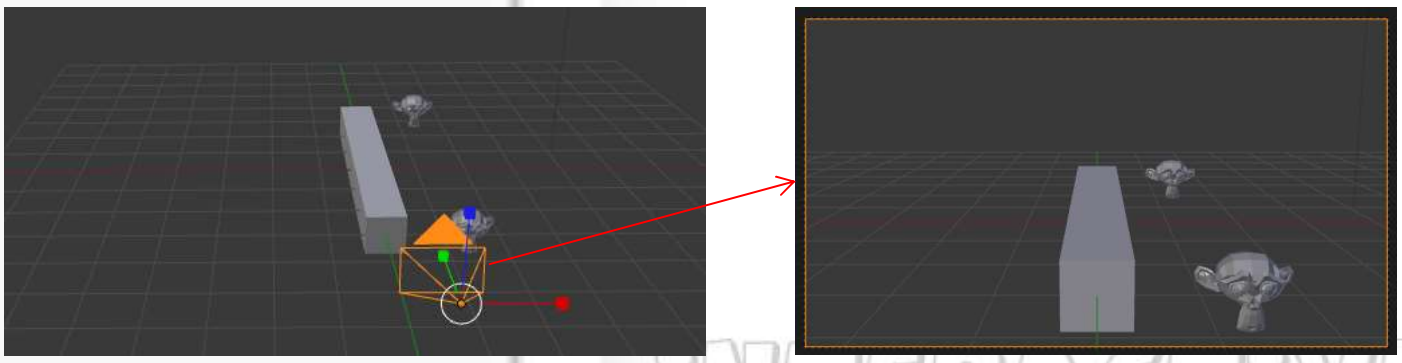
Pipeline



- Programmable vertex processor!
- Programmable pixel processor!

GPU Pipeline: Transform

- Vertex Processor (multiple operate in parallel)
  - Transform from "**world space**" to "**image space**"
  - Compute per-vertex lighting

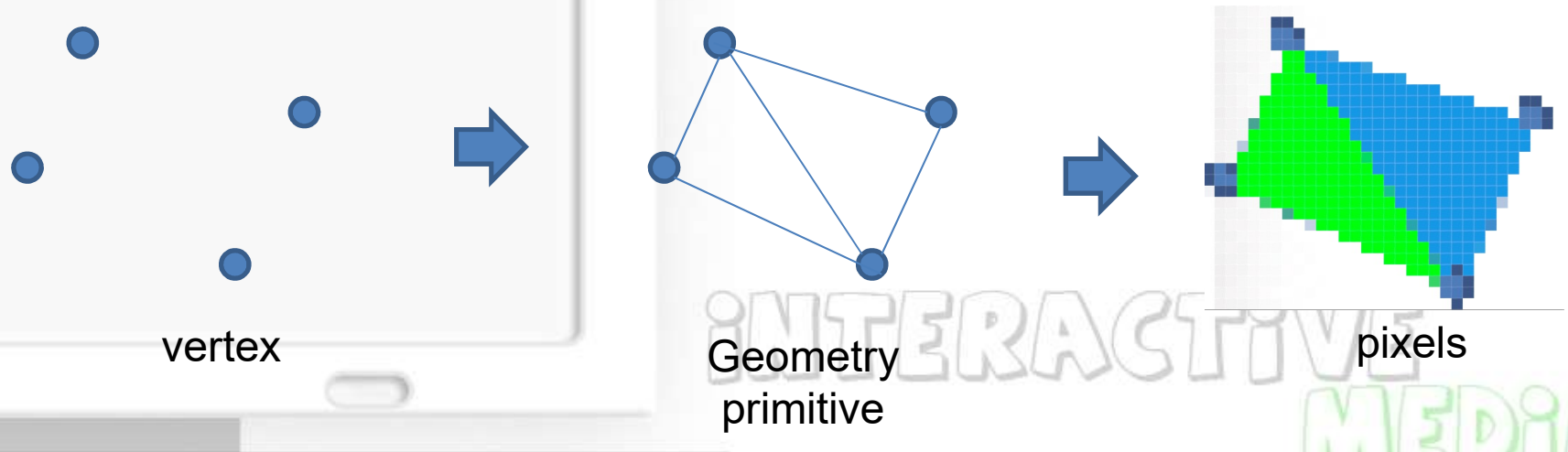


World space

Image space

GPU Pipeline: Rasterizer

- Rasterizer
  - Convert geometric rep. (vertex) to image rep. (fragment)
    - Fragment = image fragment
      - Pixel + associated data: color, depth, stencil, etc.
  - Interpolate per-vertex quantities across pixels



GPU Pipeline: Shade

- Fragment Processors (multiple in parallel)
  - Compute a color for each pixel
  - Optionally read colors from textures (images)

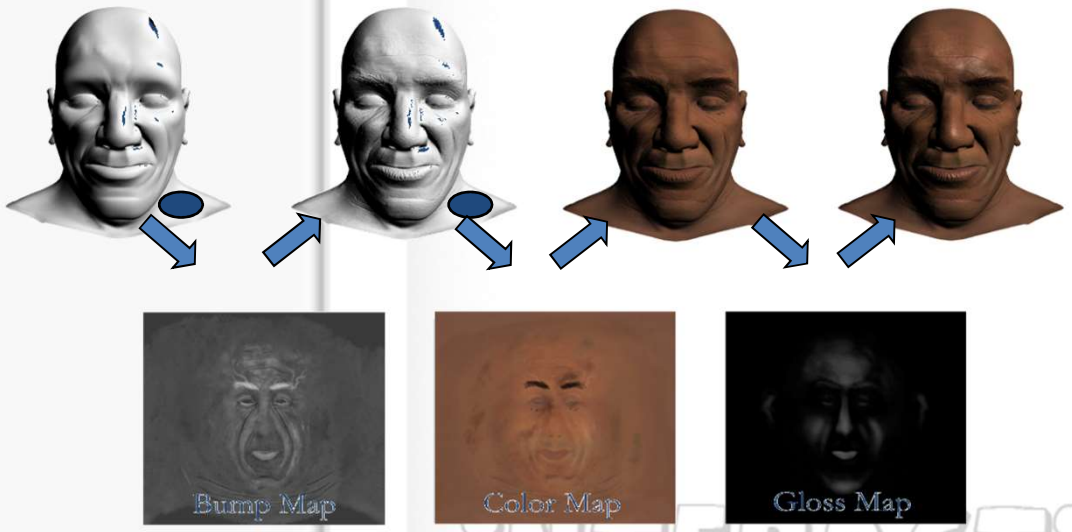


Image from nvidia slide