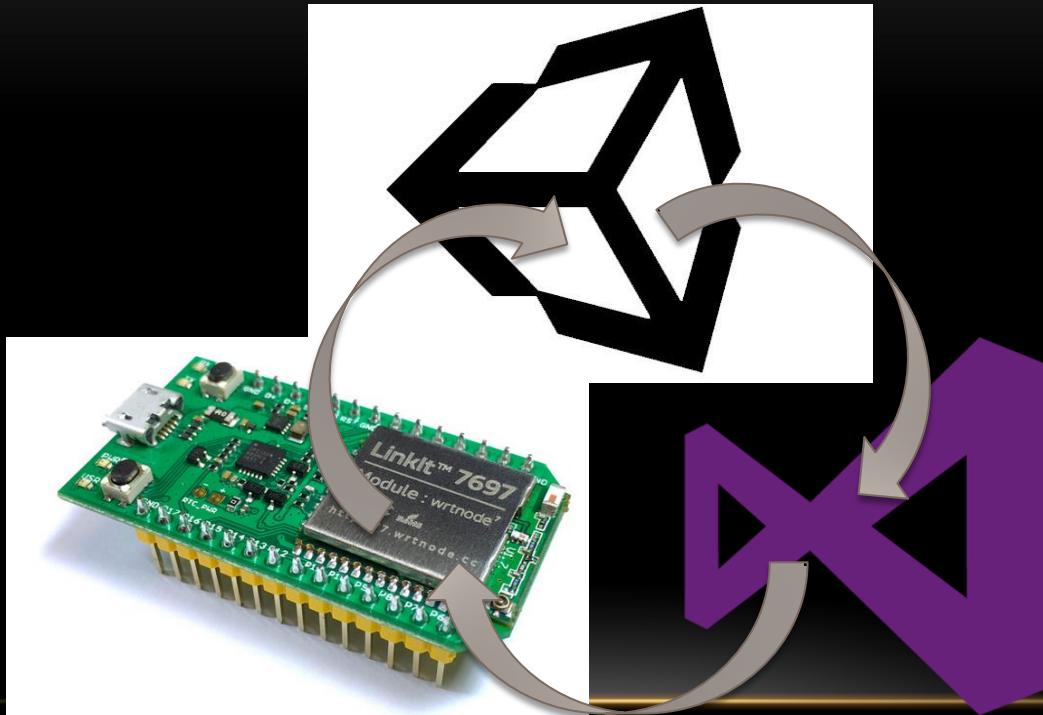


互動遊戲設計



主講人：方政順

大綱

- | | | |
|----------------|----------|-------------------------|
| 1. Unity安裝指南 | 7. 動畫 | 13. Unity & Linkit 7697 |
| 2. Unity概要 | 8. 腳本控制 | |
| 3. Unity開發環境介紹 | 9. 攝影機 | |
| 4. 天空盒 | 10. 音效 | |
| 5. 地形編輯 | 11. GUI | |
| 6. 3D模型、材質 | 12. 碰撞檢測 | |

軟體下載-Unity Hub

Unity Hub 1.6.2

Projects Learn **Installs** New Open My account

On my machine

Official Releases

Beta Releases

Find other versions in the Unity download archive

	Unity 2018.3.14f1	Download
	Unity 2018.2.21f1	Download
	Unity 2018.1.9f2	Download
	Unity 2017.4.27f1 LTS ⓘ	Download
	Unity 2017.2.5f1	Download
	Unity 2017.1.5f1 ⓘ	Download

軟體下載-Unity Hub

The screenshot shows the Unity Hub application interface. The top navigation bar includes 'Projects', 'Learn', and 'Installs'. The 'Installs' tab is selected, showing the installed version 'Unity 2018.1.9f2' as 'preferred'. A context menu is open over this entry, displaying options: 'Add Component', 'Set as preferred', and 'Uninstall'. Below the menu, the Unity logo and the path 'C:\Program Files\Unity\Hub\Editor\2018.1.9f2\Editor\Unity.exe' are visible. A 'Locate a Version' button is also present.

On my machine

Unity 2018.1.9f2 preferred ⓘ
Path: C:\Program Files\Unity\Hub\Editor\2018.1.9f2\Editor\Unity.exe

Official Releases

Beta Releases

+ Locate a Version

...
Add Component
Set as preferred
Uninstall

Find other versions in the Unity download archive

Unity Hub 1.6.2

Projects Learn

On my machine

Official Releases

Beta Releases

Find other versions in the Unity download archive

Add components to your install

Platforms

Component	Status	Size
Android Build Support	installed	1.2 GB
iOS Build Support	installed	3.3 GB
tvOS Build Support	285.4 MB	1.1 GB
Linux Build Support	129.3 MB	533.1 MB
Mac Mono Scripting Backend	29.2 MB	130.4 MB
Windows Store .NET Scripting Backend	183.2 MB	1.2 GB

Total space required: 0 B
Space available: 160.2 GB

Cancel Done

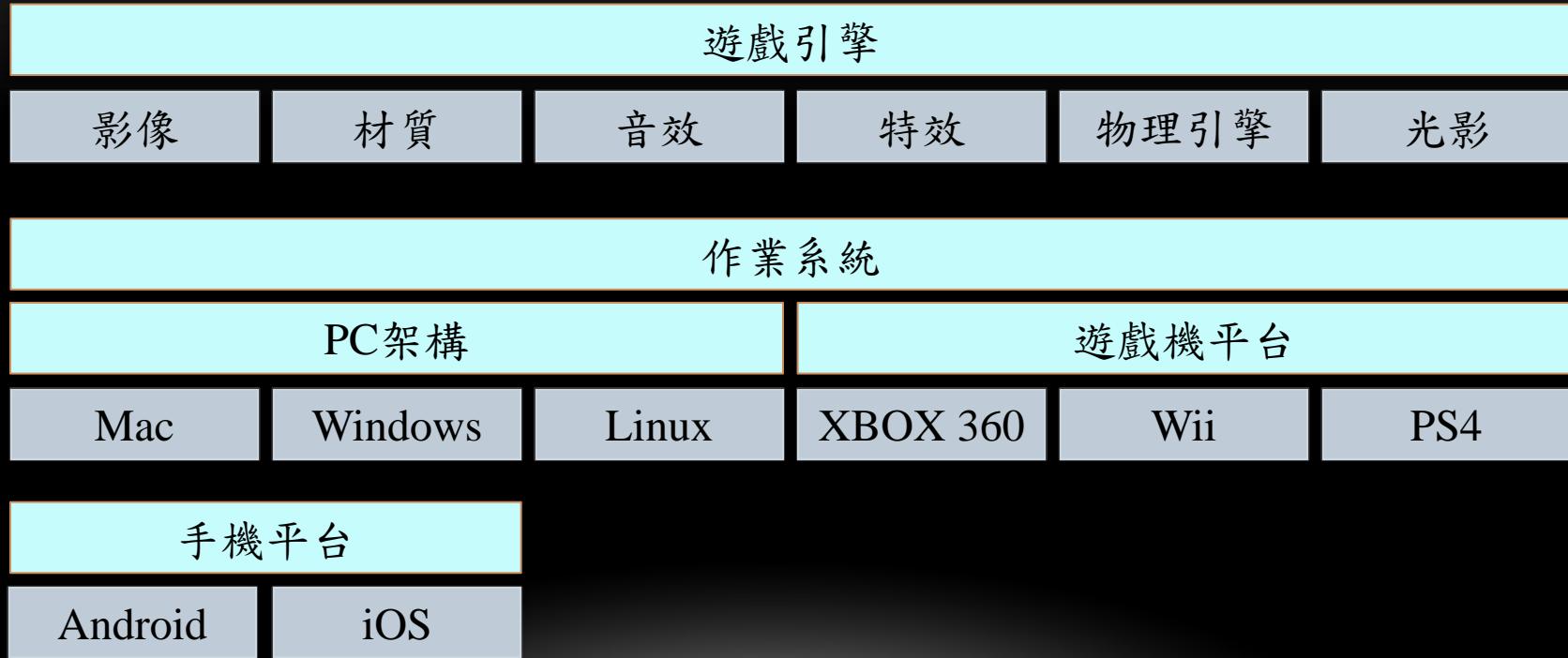
安裝時預設支援iOS、Android的平台建置，若後續開發有需要的支援，可在Unity Hub新增！

Unity介紹

Unity 支援 PhysX 物理引擎、粒子系統，並且提供網路多人連線的功能，無需學習複雜的程式語言，符合遊戲製作上的各項需求。Unity 的推出降低遊戲開發的門檻，即使是個人製作遊戲也不再是夢想。對於遊戲公司而言，選擇使用 Unity 引擎也可以縮短遊戲的開發時間。



Unity介紹



軟體啟用

Unity 2018.1.9f2 X

 unity

Sign into your Unity ID

If you don't have a Unity ID, please [create one](#).

Email

Password

[Forgot your password?](#)
[Can't find your confirmation email?](#)

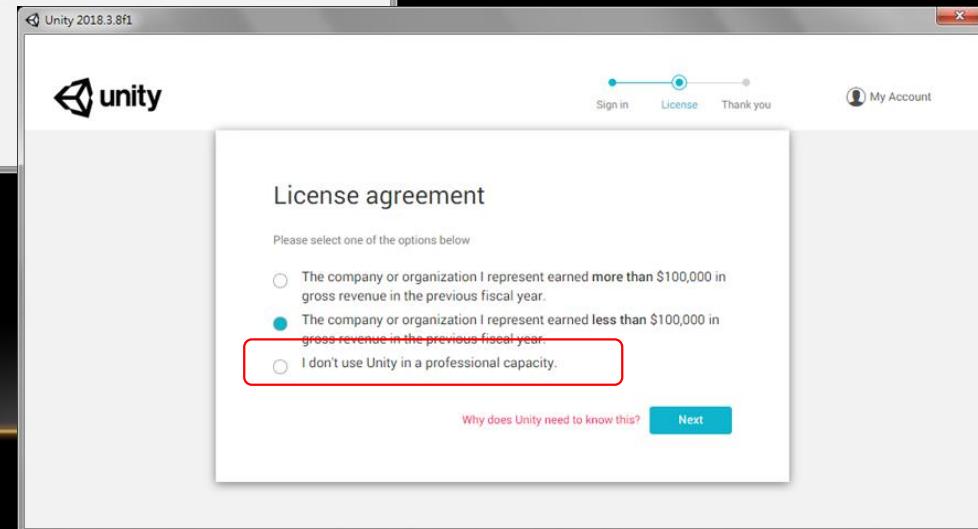
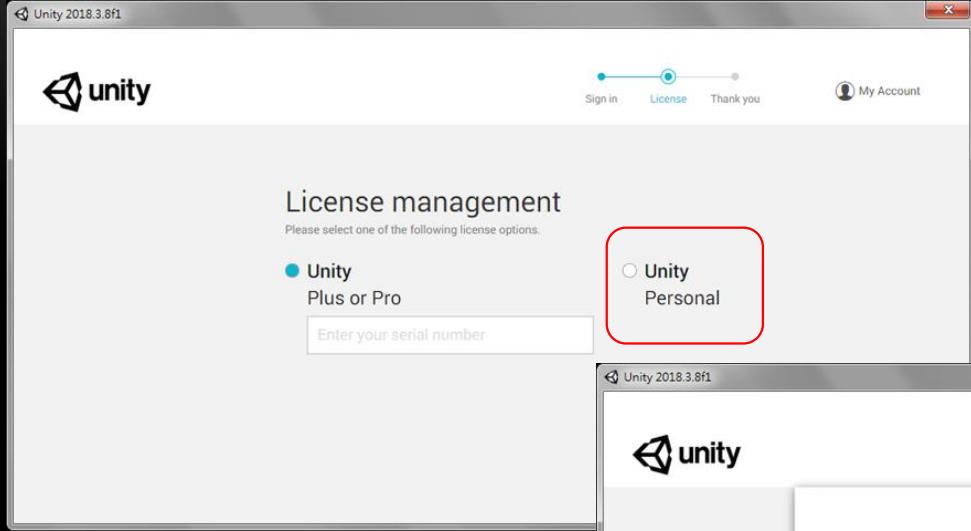
[Skip](#) [Sign in](#)

Or

 [Sign in with google](#)

 [Sign in with facebook](#)

軟體啟用



軟體收費規範

Unity Personal :

- 免費
- 營業額或資本額10萬美金以下的開發者使用

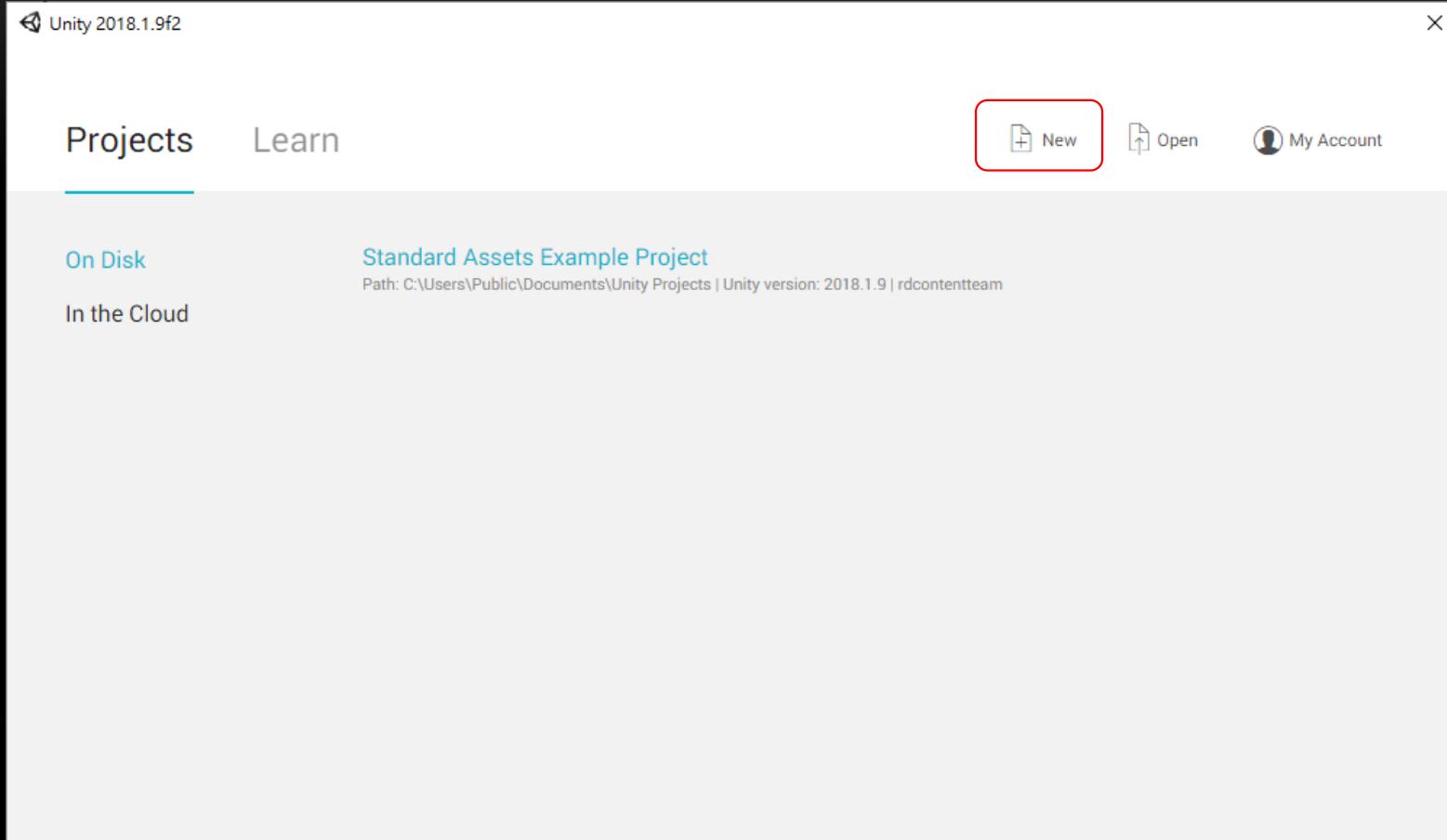
Unity Plus :

- 35美金/月(第一次租用最少必須12個月)
- 營業額或資本額20萬美金以下的開發者使用

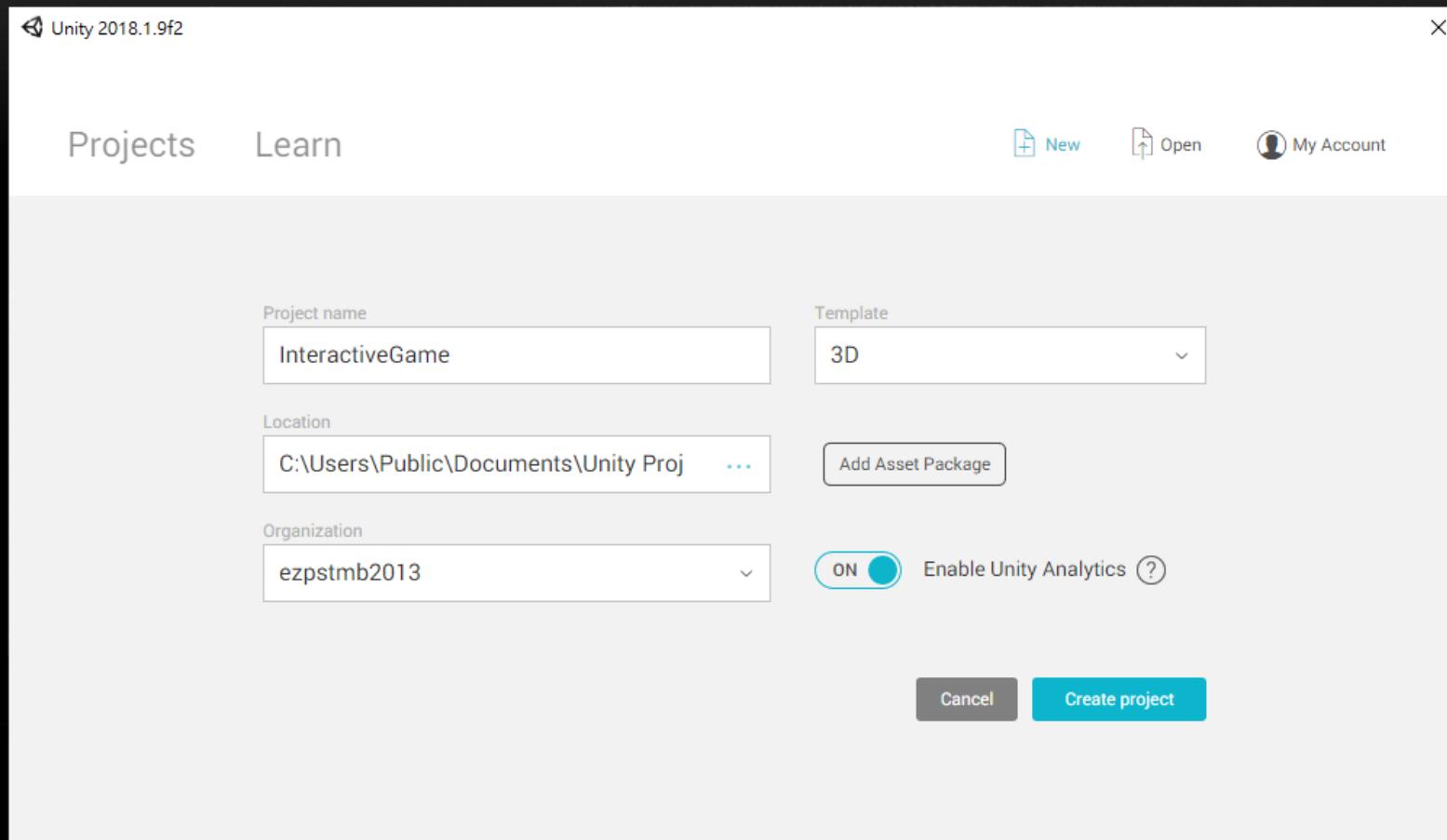
Unity Pro

- 125美金/月(第一次租用最少必須12個月)
- 沒有限制收入

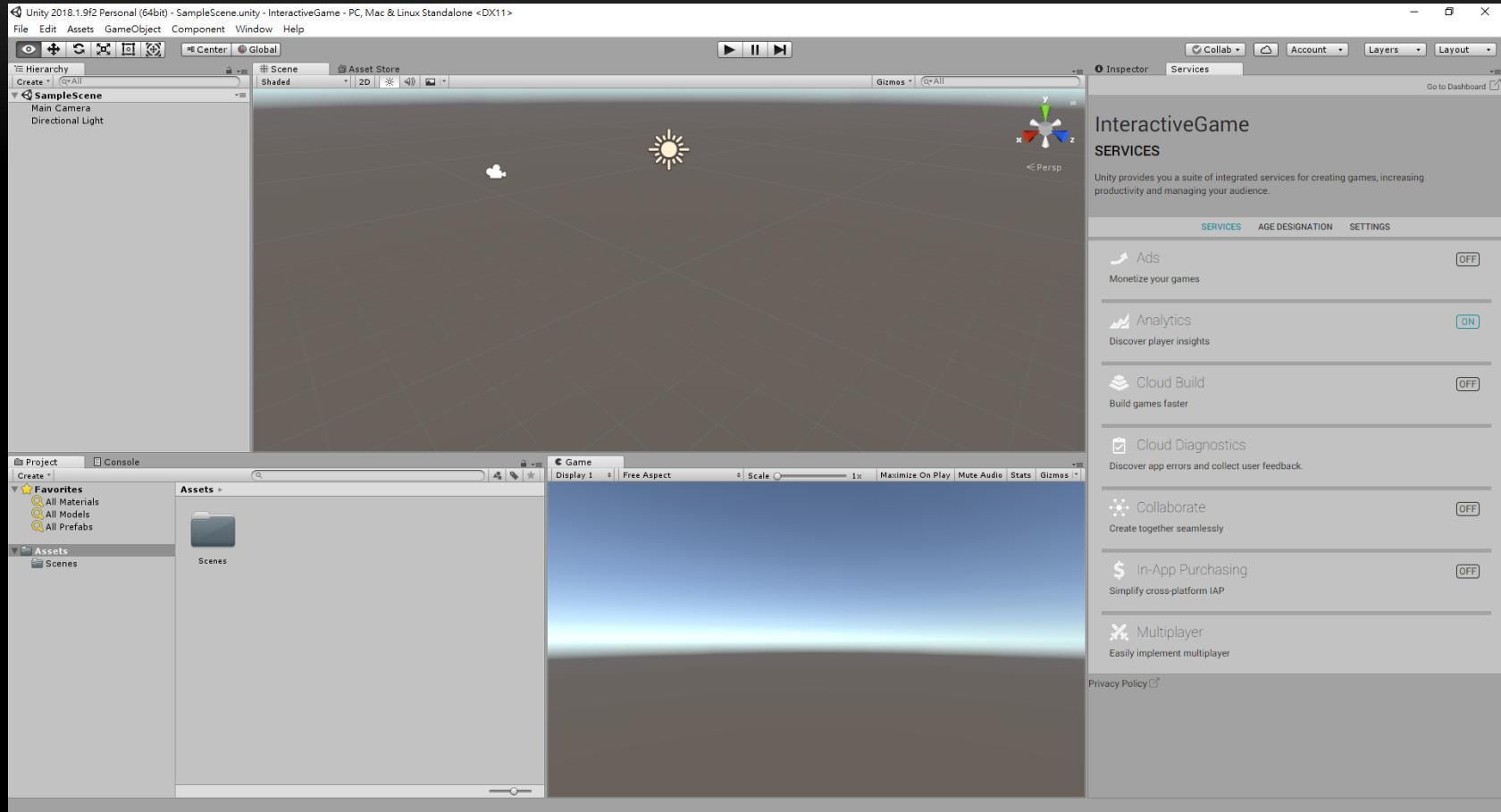
Unity-初始頁面



Unity-新增專案頁面



Unity-開發環境



Unity-功能視窗



變形工具

- 平移場景(Q)
- 移動物體(W)
- 旋轉物體(E)
- 縮放物體(R)
- 平面變形(T)



遊戲播放控制

- 遊戲執行
- 遊戲暫停
- 遊戲單格執行



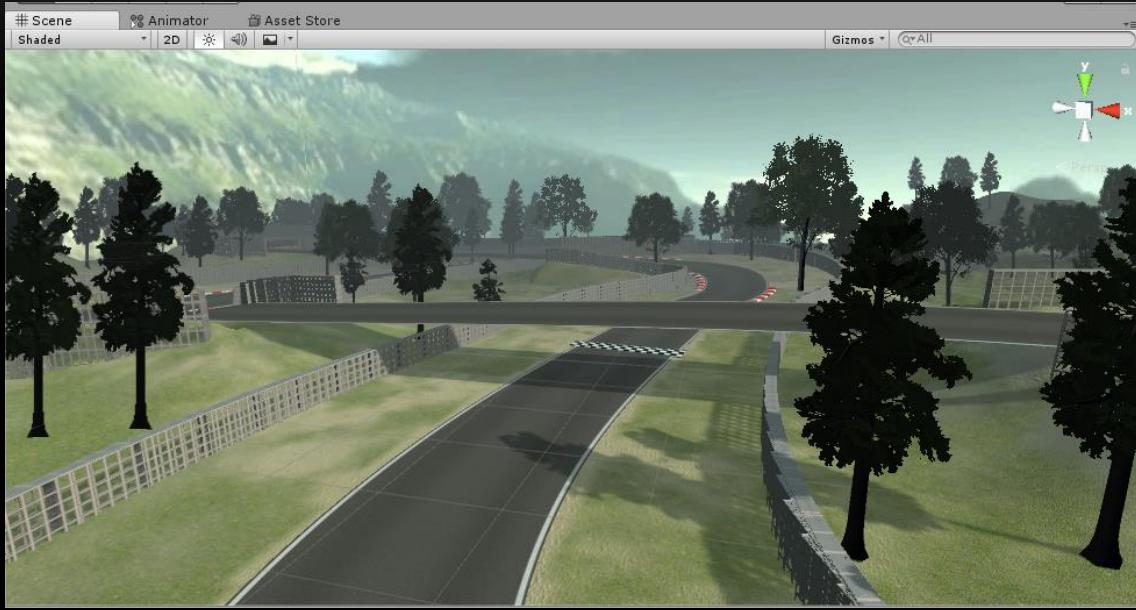
群組中心

Center表示群組以全部物件的中心作為軸心，
Pivot表示以父物件中心點作為全組物件軸心。

軸向控制

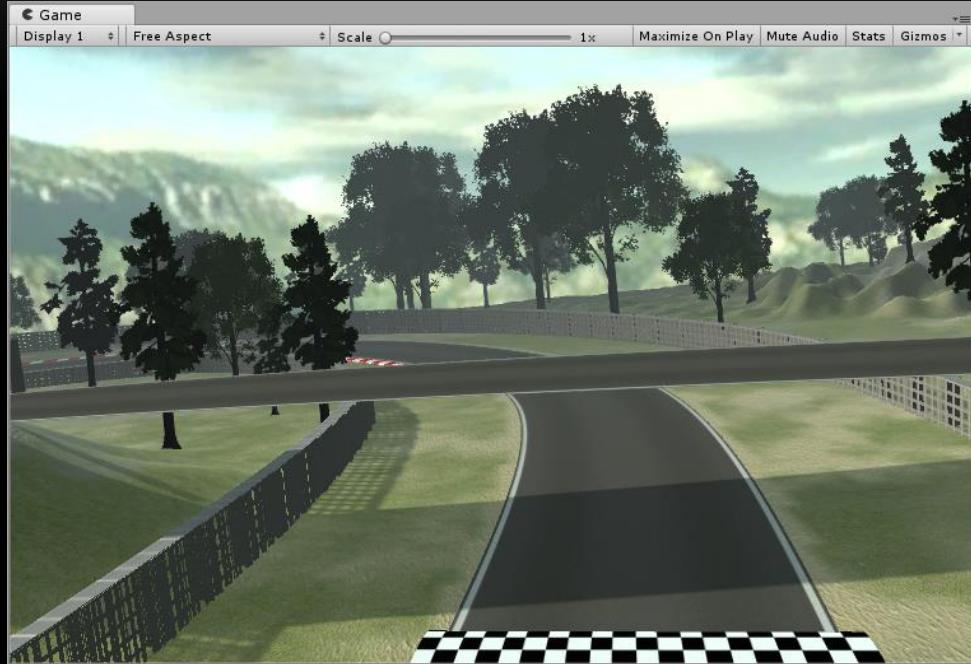
自身座標(Local)的原點以及方向是以物件本身的位置及方向為基準，世界座標(Global)是以場景中心為原點。

Unity-場景視窗



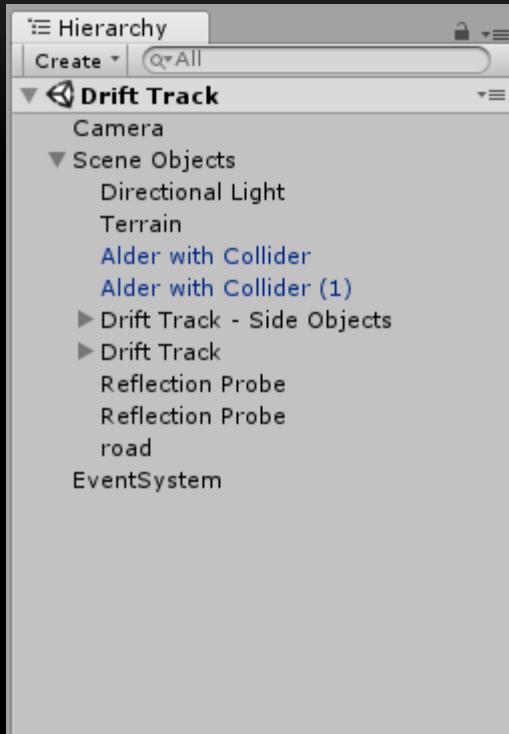
- 可在此編輯地形，加入各種角色、燈光、攝影機、粒子系統或是其他遊戲物件。
- 右上方為軸向控制器，可用滑鼠切換透視(Perspective)模式與等距(Isometric)模式。

Unity-遊戲視窗



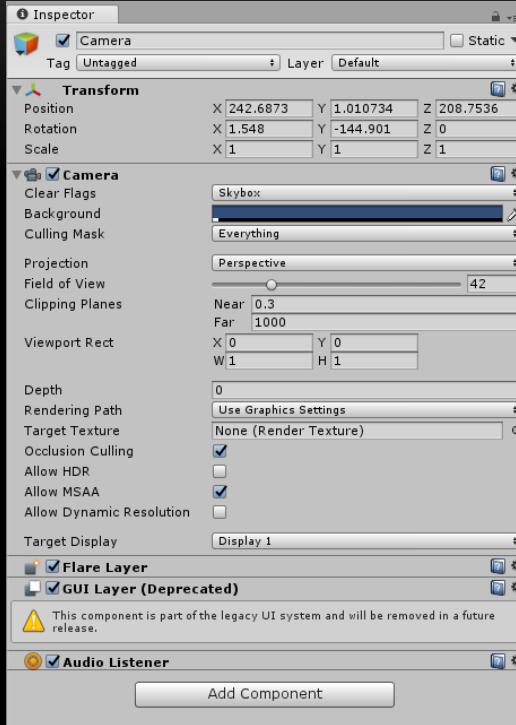
- 於開發階段遊戲執行測試，點選右上方的 Stats 按鈕可開啟統計 (Statistics) 半透明視窗，檢視模型面數、材質貼圖、記憶體用量等資訊。
- 顯示遊戲執行過程中玩家看到的場景。如果在場景中平移或者旋轉場景的主相機，將看到遊戲執行區視圖改變。

Unity-階層視窗



- 顯示目前所有場景物件與其階層關係，需要在物件附加元件或腳本時，可直接拖曳到階層管理區項目。

Unity-元件編輯視窗

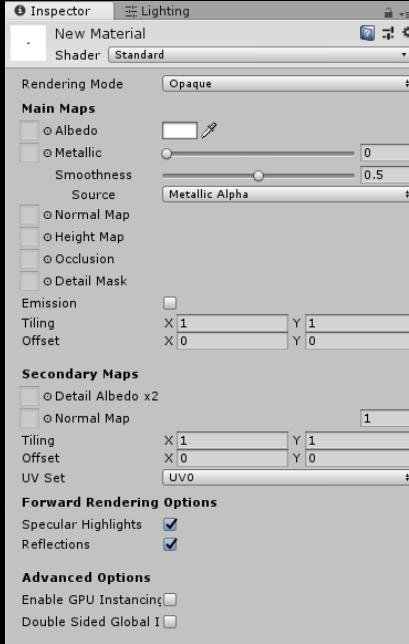


- 用於顯示、設定物件屬性(物件名稱、標籤、座標位置、旋轉角度、縮放比例等)，若物件附加聲音、腳本、碰撞器等元件，亦可透過此視窗修改元件參數。

Unity-天空盒

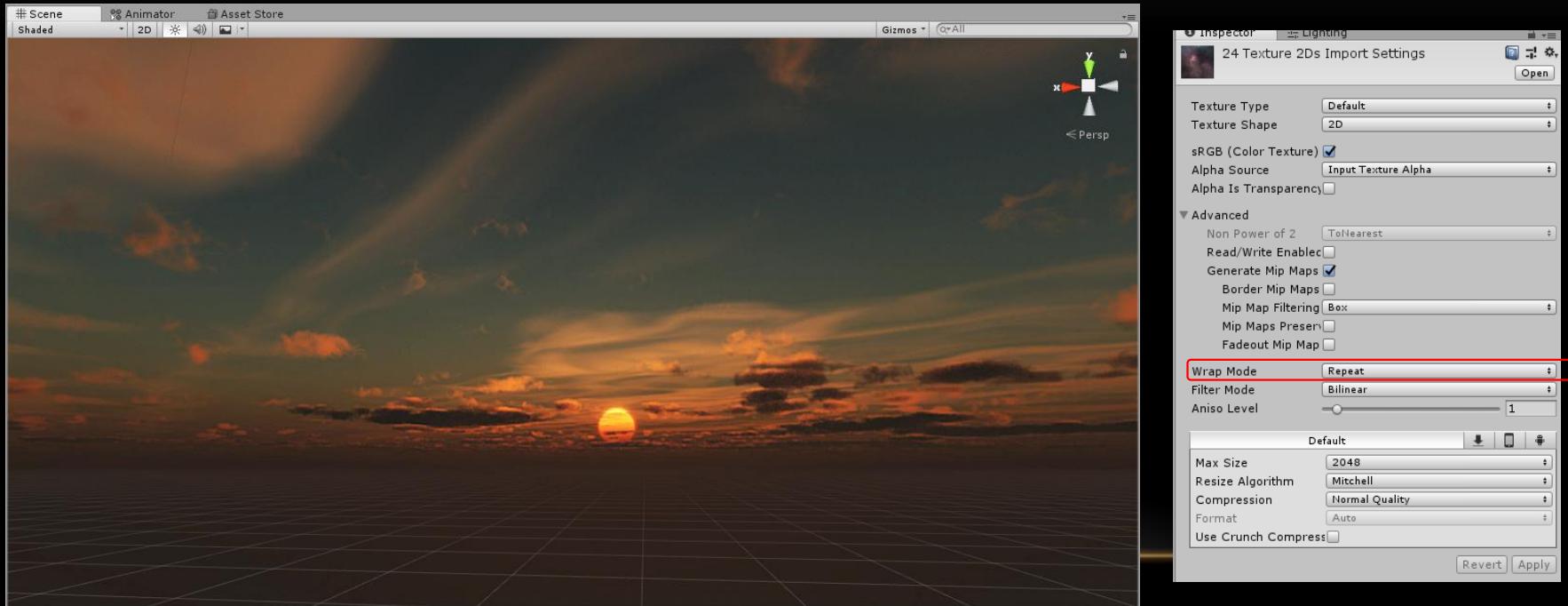
Skybox 實際上是由 6張圖片合成的無接縫天空背景。

1. 在Project視窗新增一個材質球(Material)，調整Shader為 Skybox > 6Sided 。
2. 依序放上對應位置的貼皮。
3. 將此材質球拉到Scene視窗。

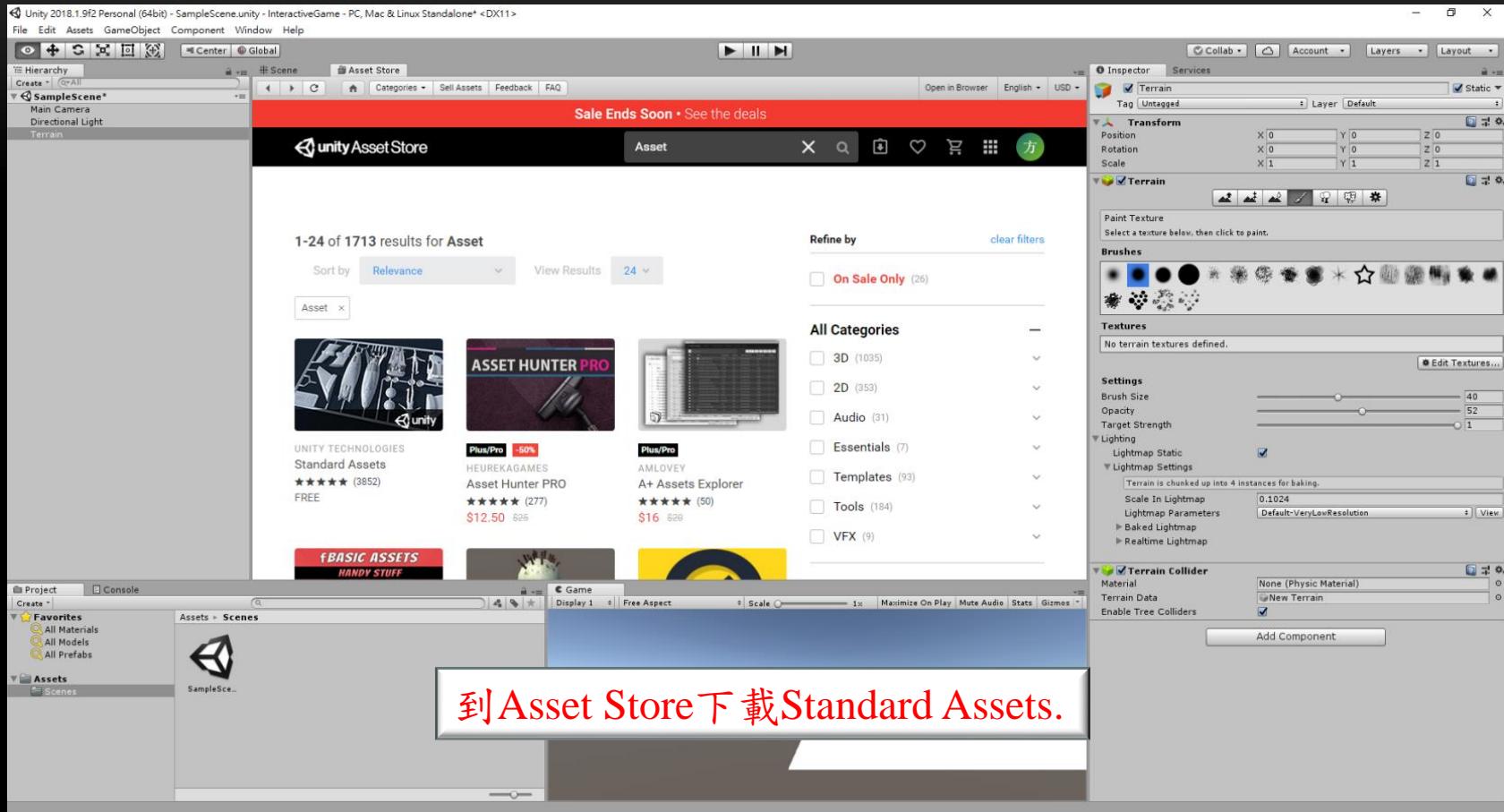


Unity-天空盒

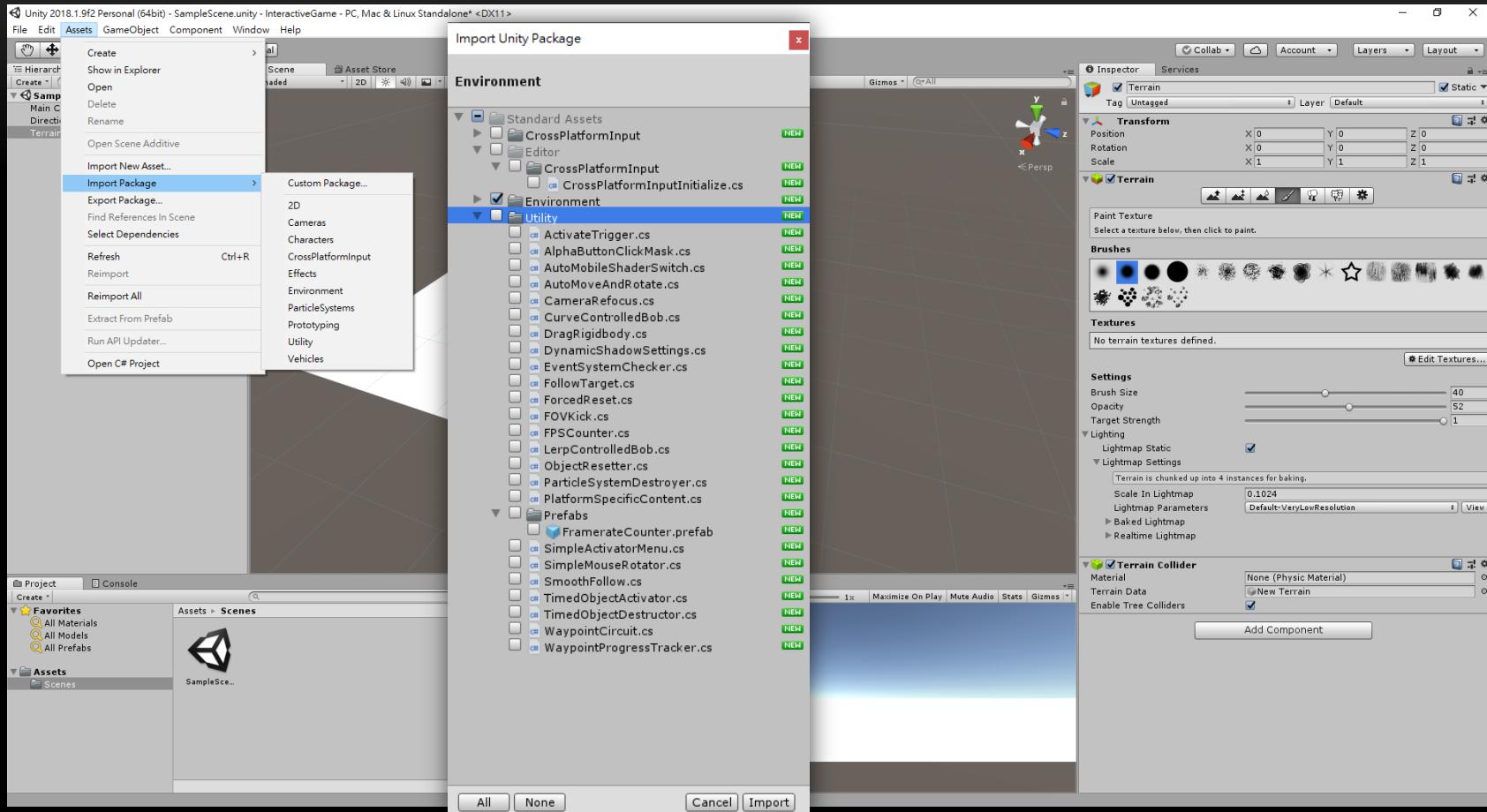
若細看畫面，會發現其實有切現，此時需到調整貼皮wrap mode屬性，由Repeat改成Clamp就可以看見無縫隙的Skybox。



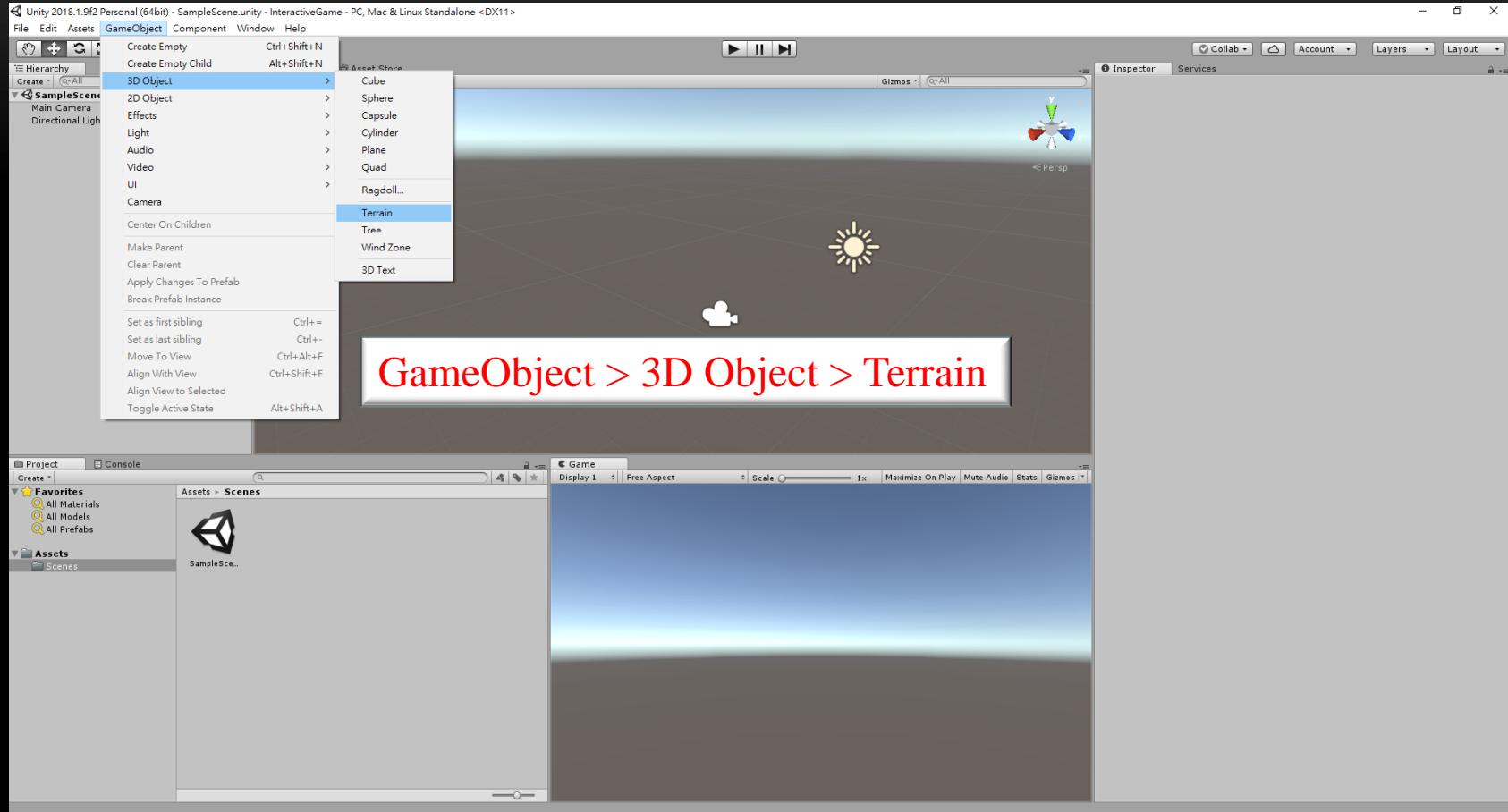
Unity-Asset Store



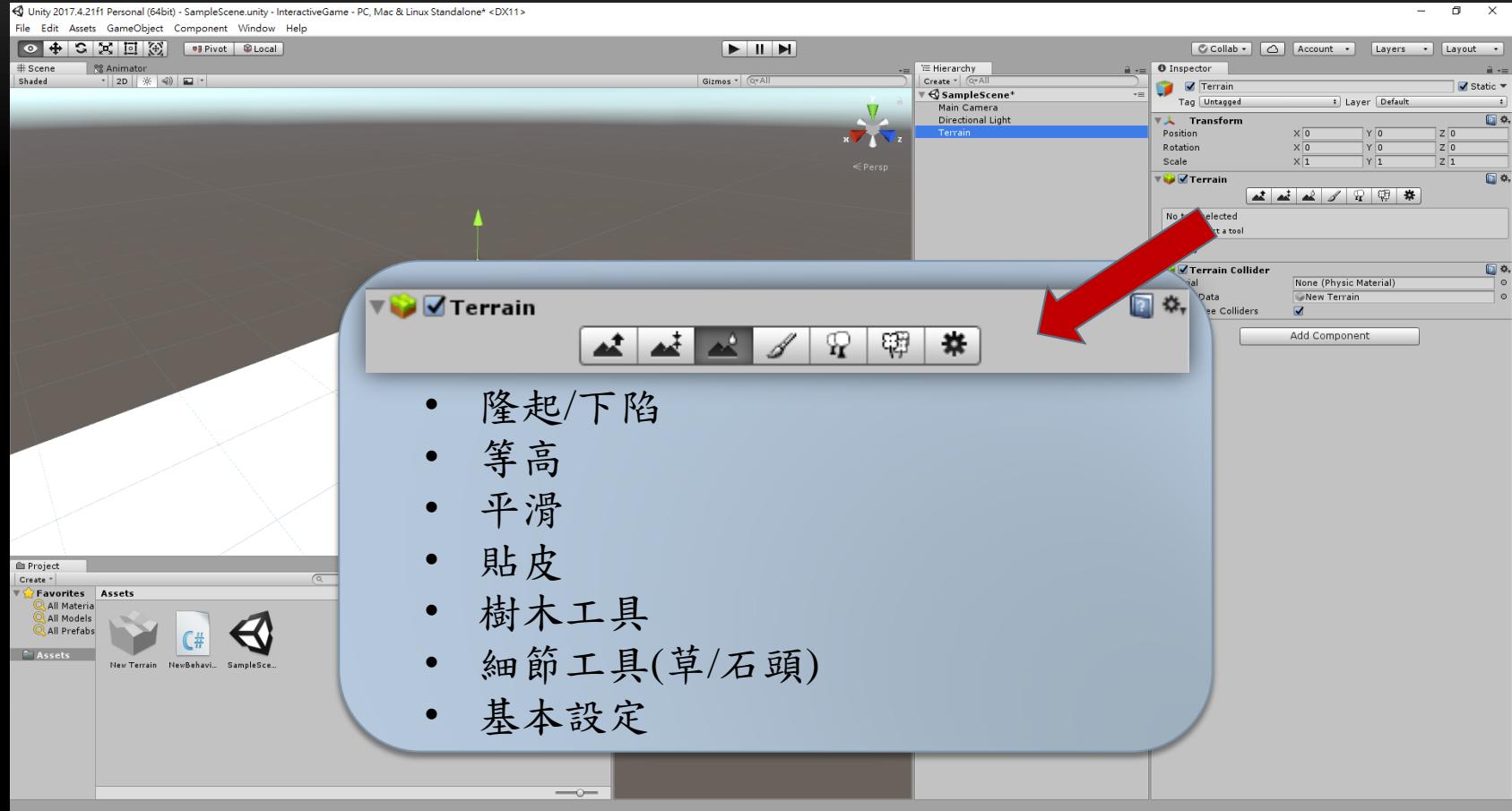
Unity-添加基本素材包



Unity-地形編輯

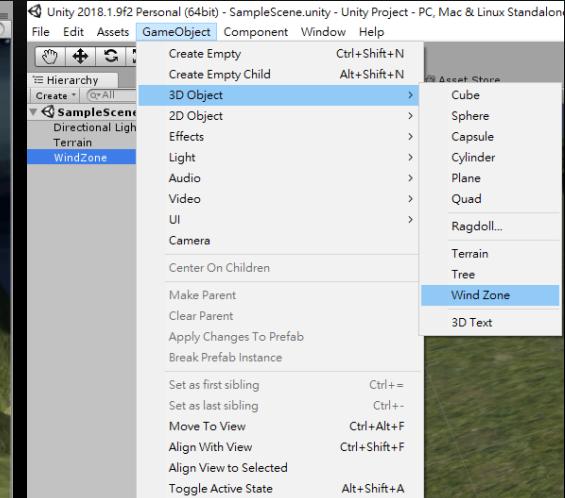
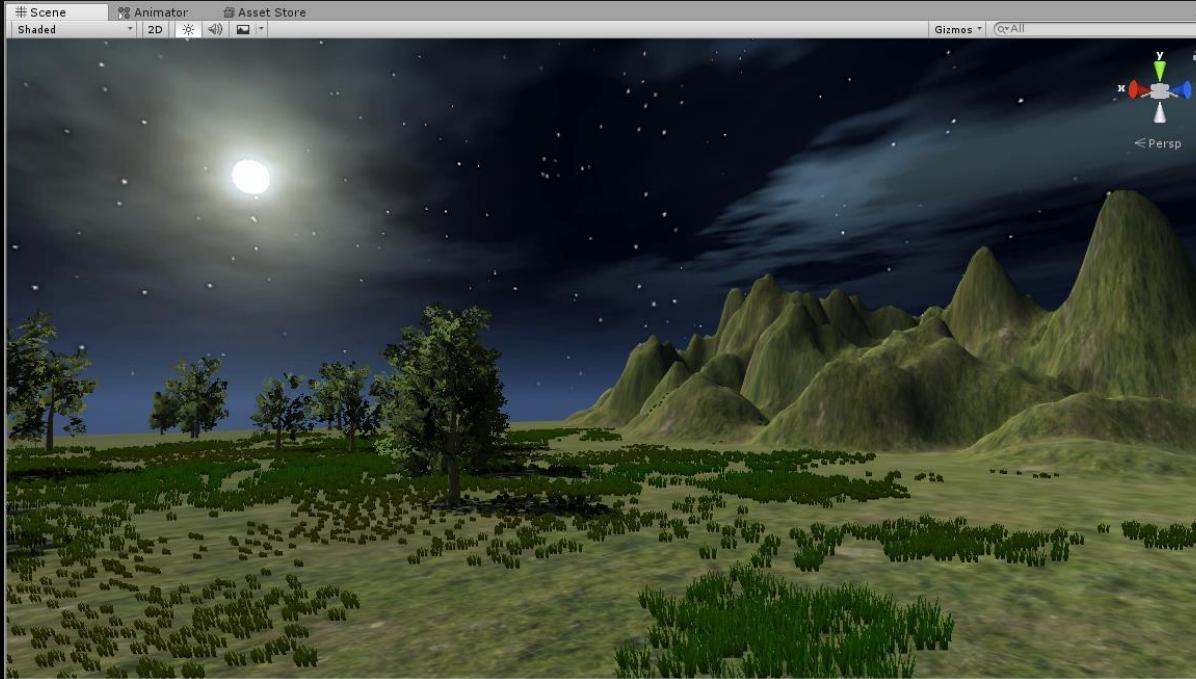


Unity-地形編輯



Unity-地形編輯

簡單設計成擁有山茂、樹木、草叢的環境！



加入風物件，使場景更
加生動！

Unity-3D模型

3D模型都是由多邊形的網格所組成的，精細度越高就越真實，但對載具的硬體需求相對也越高，現在業界會有所謂的次世代美術，意味著模型本身的面數精簡了，但依然能表現出高模的效果。

模型取得方式

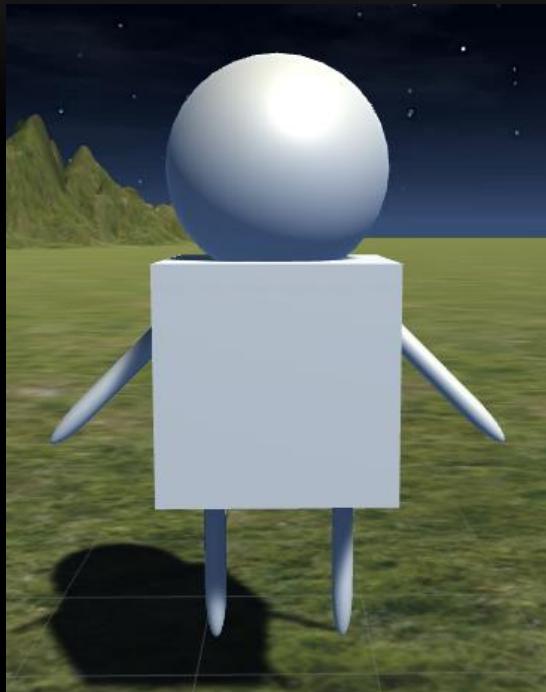
程式產生

- 基本形狀：平面、方塊、球體、圓柱。
- 演算法生成：地形、河流。

3D建模

- 免費軟體：Blender、Google SketchUp
- 商業軟體：Auto CAD、3DS Max、Maya、Zbrush

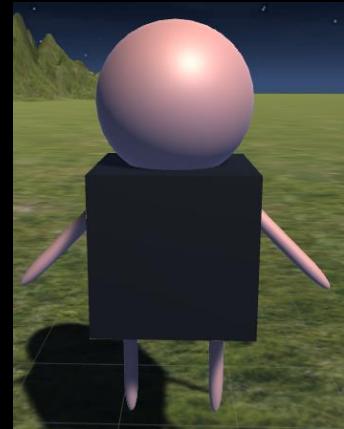
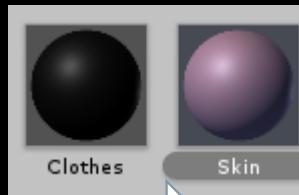
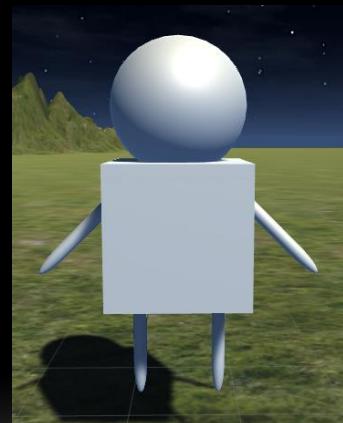
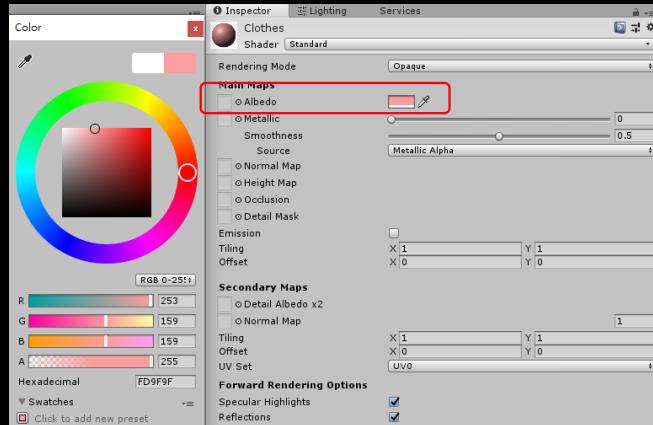
Unity-製作簡易模型



名稱	形狀	位置	旋轉	縮放
Head	Sphere	(0, 2.1, 0)	(0, 0, 0)	(1, 1, 1)
Body	Cube	(0, 1.2, 0)	(0, 0, 0)	(1, 1, 1)
LeftHand	Capsule	(-0.7, 1.2, 0)	(0, 0, -45)	(0.1, 0.5, 0.5)
RightHand	Capsule	(0.7, 1.2, 0)	(0, 0, 45)	(0.1, 0.5, 0.1)
LeftLeg	Capsule	(-0.3, 0.4, 0)	(0, 0, 0)	(0.1, 0.5, 0.1)
RibhtLeg	Capsule	(0.3, 0.4, 0)	(0, 0, 0)	(0.1, 0.5, 0.1)
Man	Empty	(0, 0, 0)	(0, 0, 0)	(1, 1, 1)

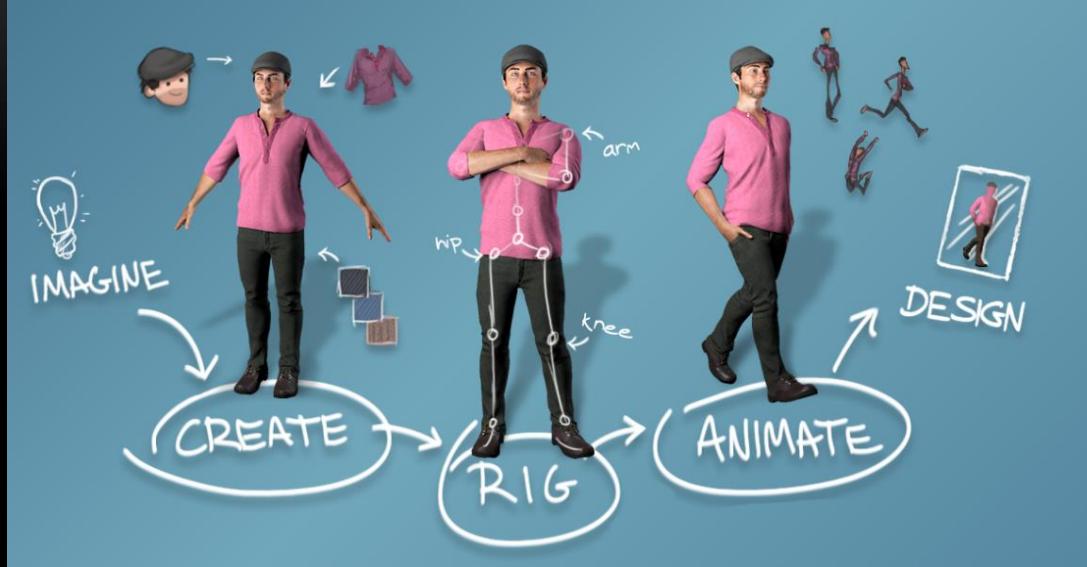
Unity-設定材質球

稱之為顏色的東西，其實只是物體反射光的效果所呈現出來的定義，3D世界裡，將其稱為材質，材質大致可分兩種：顏色、貼皮。



為模型上色，新增材質球 (Material)，並幫材質球調整適合顏色。並將此材質球拉到所需改變顏色的部位。

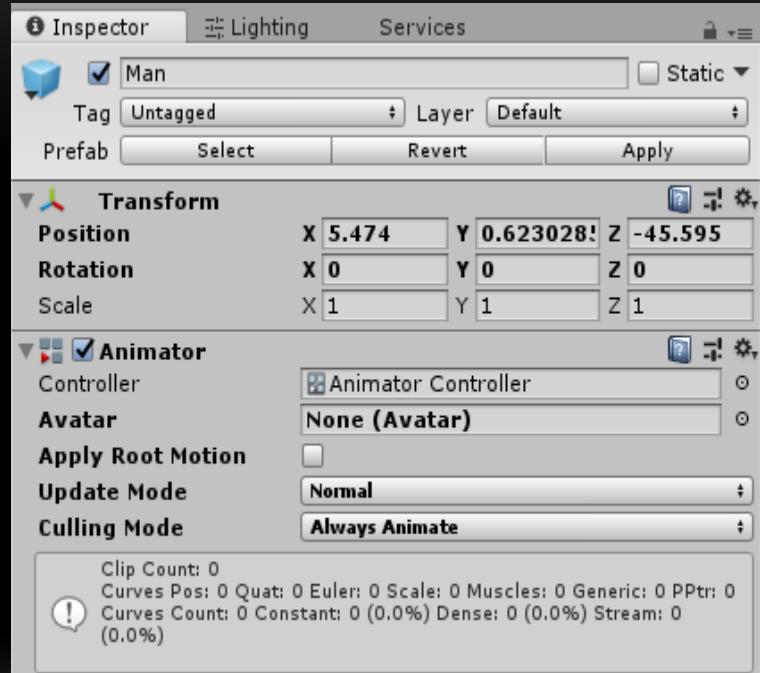
Unity-設定動畫



俗稱的3D動畫，一般使用Blender、3D Max、Maya、Zbrush等建模軟體製作。用於表現出角色的各種行為，例如：行走、攻擊、跳躍等等動作，再由遊戲引擎內部的動畫控制器去播放模型動作，3D動作的製作需要設計師大量的時間才可完成一個不會破圖及骨架動作異常的動畫。

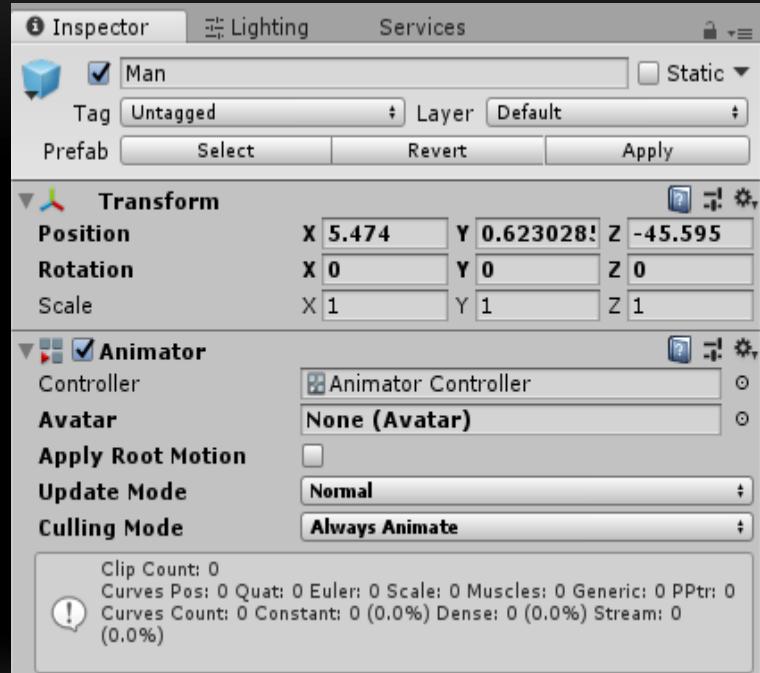
Unity-設定動畫

1. 在Assets視窗建立Animator Controller。
2. 在Assets視窗建立Animation(HandUp)。
3. 將Animator Controller拉到(Man)物件上，使Man物件具有動畫功能。



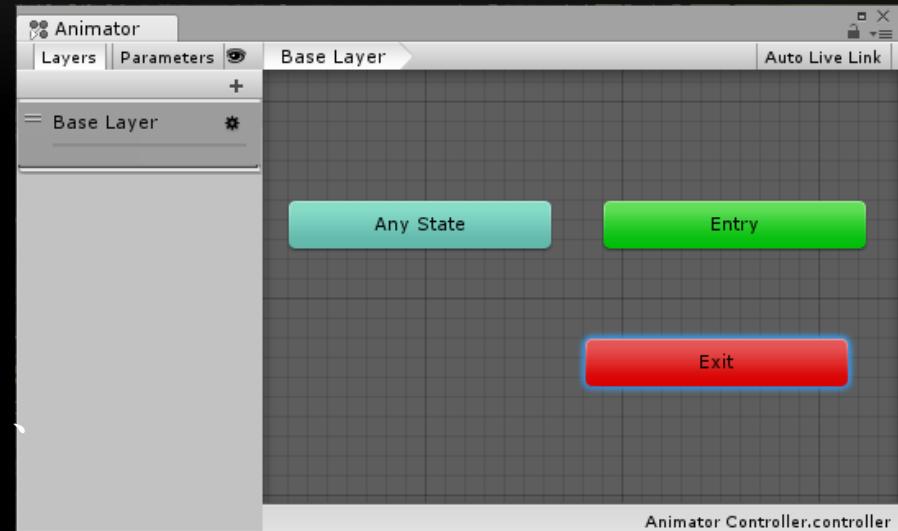
Unity-設定動畫

1. 在Assets視窗建立Animator Controller。
2. 在Assets視窗建立Animation(HandUp)。
3. 將Animator Controller拉到(Man)物件上，使Man物件具有動畫功能。



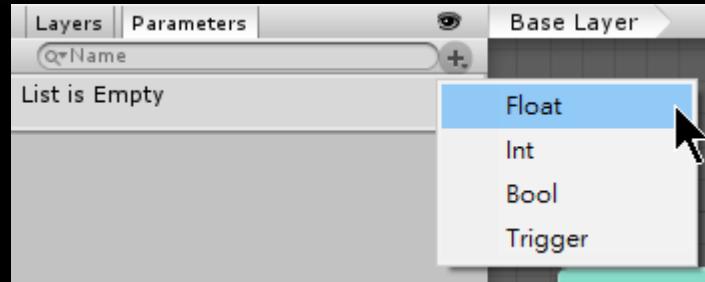
Unity-設定動畫

1. 點擊 Animator Controller 切換到 Animator 視窗。
2. 左側有兩個頁籤分別是 Layers 和 Parameters，以及右側的 Animation State Machines 流程視圖。
3. 其中有三個預設的狀態機 (State Machines)，Entry、Exit、Any State，分別表示這串動畫的進入點、離開點、任意狀態插入點。



Unity-設定動畫

動畫參數是在Animator控制器中定義的變量，可以從腳本訪問和分配數值，藉此控製或影響狀態機的流程變化。一樣按下上方的「+」來添加新的參數，參數有四種類型：

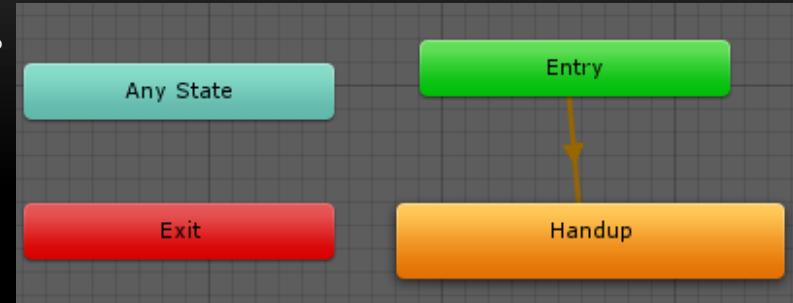


- Int – 整數數字。
- Float – 浮點數。
- Bool – 真或假。
- Trigger – 布爾類型參數，控制器被使用過後重置(一次性觸發器)。

可以透過程式腳本的對應方法 SetFloat、SetInt、SetBool、SetTrigger 和 ResetTrigger 來控制這些參數。

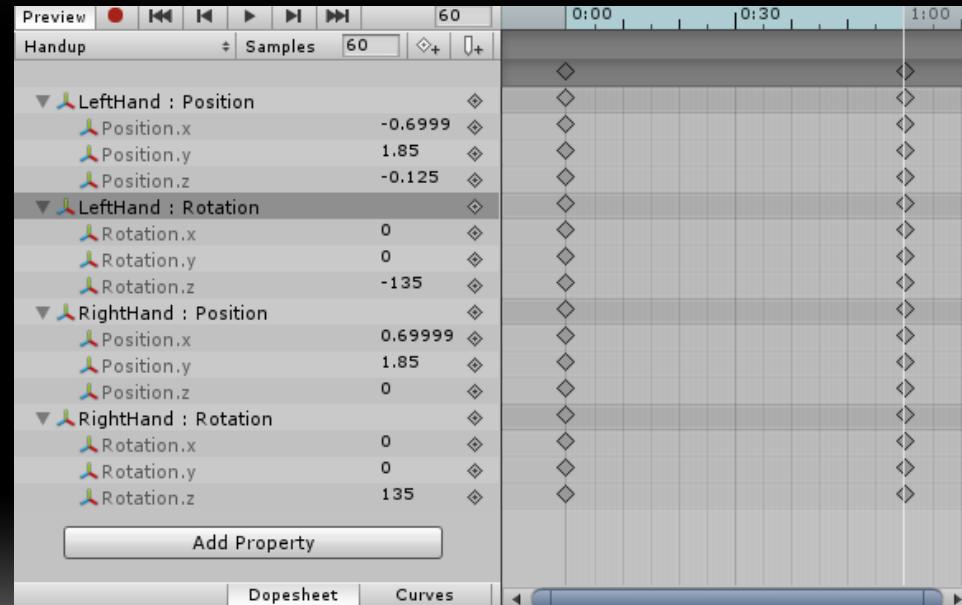
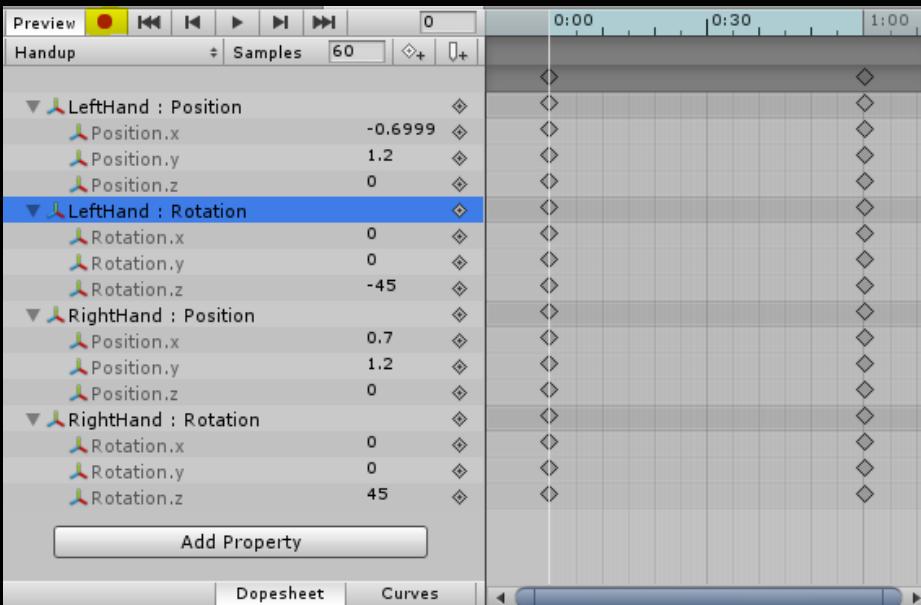
Unity-設定動畫

1. 將Animation(HandUp)拉進Animator Controller。
2. 打開Animation視窗(Ctrl+6)。
3. 在Hierarchy視窗，聚焦於Man物件上。
4. 開始編輯動作內容。



Unity-設定動畫

✓ 使用錄製功能，調整部位動作後，自動帶值進入Animation視窗。

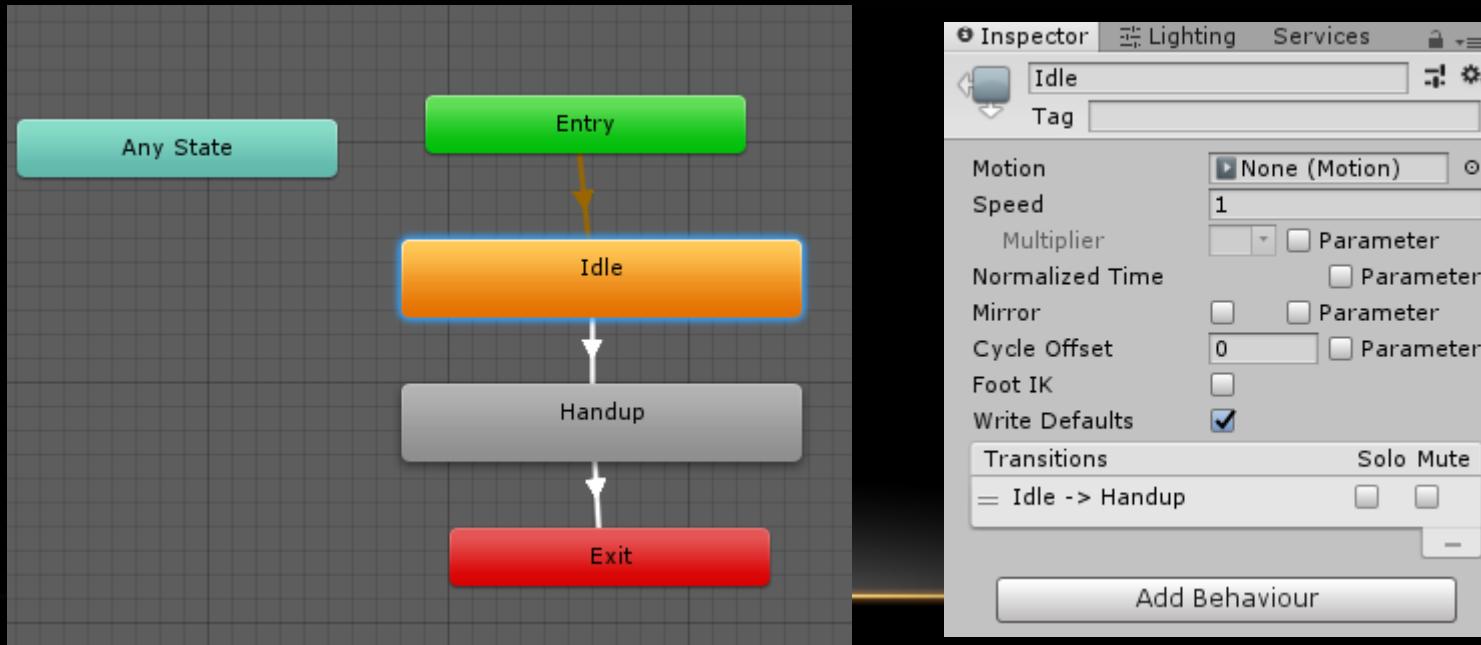


Unity-設定動畫



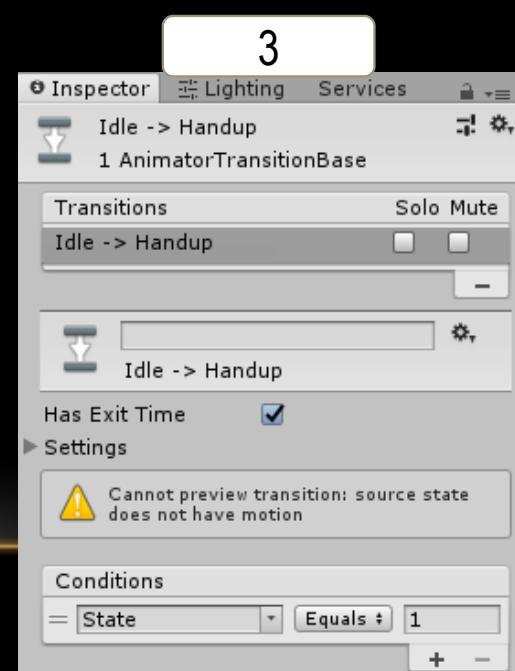
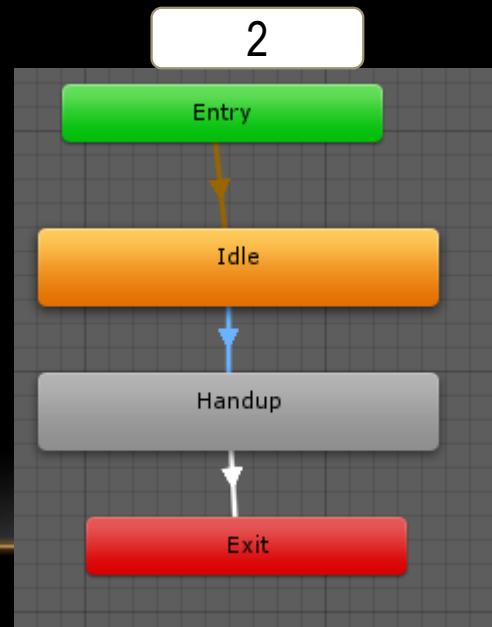
Unity-設定動畫

- 預設的設定，會使動畫在遊戲中一直作動，故需新增一個閒置的空動作物件，讓他延遲。



Unity-設定動畫

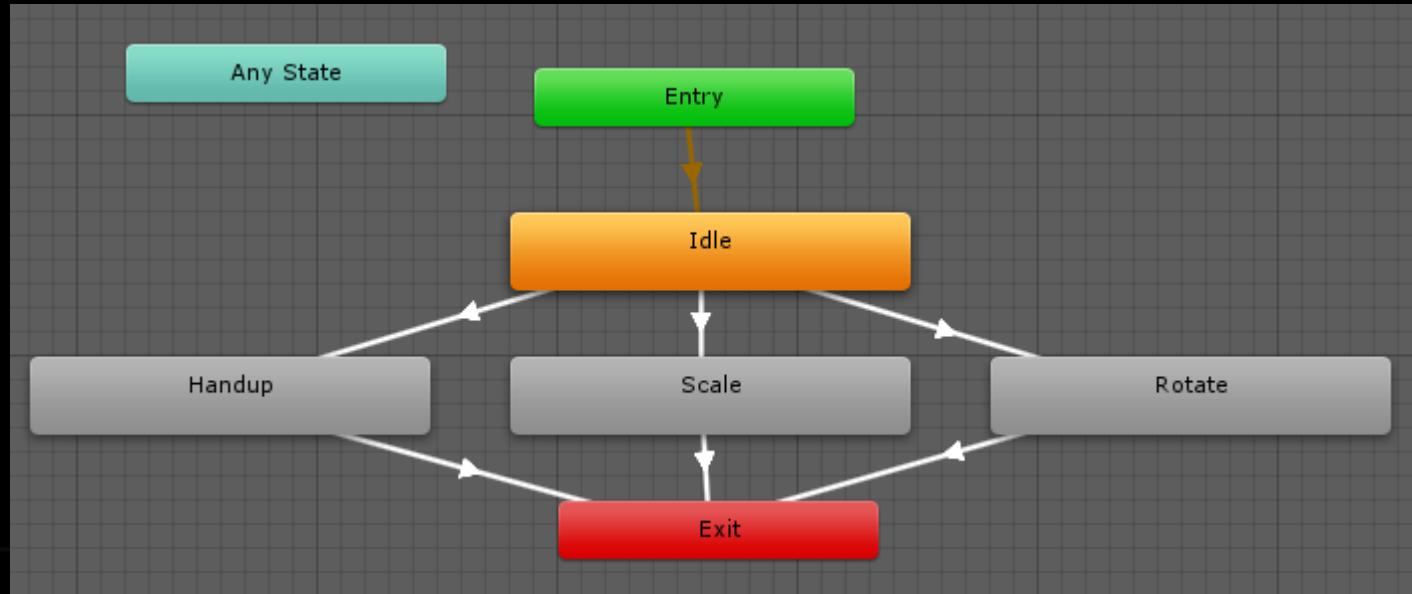
1. 在Animator 視窗中選擇Parameters，新增一個整數型態的變數State。
2. 點擊Idle到HandUp之間的線段。
3. 新增Conditions : State Equals 1。
4. 執行遊戲時，會發現Man物件已經不會作動，需改變State的數值，才會動作。



Unity-設定動畫

練習題：請為Man物件，增加兩個動作

1. State為2的時候，在原地放大5倍
2. State為3的時候，自轉360度



Unity-腳本控制

鍵盤按鍵觸發

- if (Input.GetKeyDown("a")){} // A 鍵被按住時持續觸發
- if (Input.GetKeyDown("a")){} // A 鍵被按下瞬間觸發一次
- if (Input.GetKeyUp("a")){} // A 鍵被放開瞬間觸發一次

使用鍵盤控制操作物體，是最基本又常用的互動方式，這個例子將示範如何透過按鍵操控物體移動和旋轉。

1. 新增一個 c# script 掛到Man物件上。
2. 在 script 中輸入以下程式即完成。

```
void Update()
{
    if (Input.GetKey("up")) { transform.Translate(0, 0, 0.1f); }
    // 按住 上鍵 時，物件每個 frame 朝自身 z 軸方向移動 0.1 公尺
    if (Input.GetKey("down")) { transform.Translate(0, 0, -0.1f); }
    // 按住 下鍵 時，物件每個 frame 朝自身 z 軸方向移動 -0.1 公尺
    if (Input.GetKey("left")) { transform.Rotate(0, -3, 0); }
    // 按住 左鍵 時，物件每個 frame 以自身 y 軸為軸心旋轉 -3 度
    if (Input.GetKey("right")) { transform.Rotate(0, 3, 0); }
    // 按住 右鍵 時，物件每個 frame 以自身 y 軸為軸心旋轉 3 度
}
```

Unity-腳本控制

滑鼠按鍵觸發

- if (Input.GetMouseButton(0)){ } // 滑鼠左鍵被按住時持續觸發
- if (Input.GetMouseDown(0)){ } // 滑鼠左鍵被按下觸發一次
- if (Input.GetMouseUp(0)){ } // 滑鼠左鍵被放開觸發一次

- 0、1、2 分別代表滑鼠 左、右、中 鍵

Unity-腳本控制

Transform 相關

- `transform.Translate(x, y, z);` // 移動
- `transform.Rotate(x, y, z);` // 旋轉
- `transform.position = new Vector3(x, y, z);` // 設定位置(世界座標)
- `transform.localPosition = new Vector3(x, y, z);` // 設定位置(相對於母物體)
- `transform.eulerAngles = new Vector3(x, y, z);` // 設定角度
- `transform.localEulerAngles = new Vector3(x, y, z);` // 設定角度(相對於母物體)
- `transform.LookAt(otherGameObject.transform);` // 讓物件的 Z 軸朝向另個物件的軸心
- `transform.forward` // 代表物件的 z 軸方向(unity以z軸為前方)

例如要讓物件朝著另一個物件的前面(z 軸方向)移動：

```
transform.Translate(otherGameObject.transform.forward);
```

Unity-腳本控制

撈出物件(GameObject)

使用 unity 開發遊戲時，往往需要在有一個 script 掛在 A 時，卻想要在這個 script 寫程式控制 B 物件，這時候就得懂得抓取物件才行。

```
GameObject.Find("場景中的物件名稱");
```

例如要旋轉場景中名為 TestObject 的物件：

```
GameObject.Find("TestObject").transform.Rotate( 0, 2, 0 );
```

此方法無法抓取原本不在場景中或未啟用(active為false)的物件。

Unity-腳本控制

抓取 tag 物件

```
GameObject[] array = GameObject.FindGameObjectsWithTag("Geometry");
```

所有 tag 為 Geometry 的物件，都會被抓出來存到變數(陣列) array 中。

抓取父物件

```
transform.parent.gameObject;
```

抓取子物件

```
transform.GetChild(2).gameObject;
```

抓取第 3 個子物體(索引號是從 0 開始)，可透過 transform.childCount 取得子物體的數量。

Unity-腳本控制

撈出元件(Component)

Component 是附加到遊戲物件的元件，例如附加 Light 元件到遊戲物件時，即可產生燈光的照明效果，附加 Camera 元件可讓物件擁有攝影機功能。實際上腳本也屬於一種元件，可以附加到遊戲物件。

大部份的共用元件可使用一般的成員變數進行存取

```
GetComponent<元件類型>();
```

除了 transform 可以直接打，其他元件都要用 GetComponent 去抓。以下是常見的元件：

- Transform
- Rigidbody (剛體)
- Renderer (著色器)
- Camera (攝影機)
- Light (燈光元件)
- Animator (動畫)
- Collider .(碰撞器)

Unity-腳本控制

啟用(或停用)物件、元件

```
yourGameObject.SetActive( true );           // 啟用物件  
yourComponent.enabled = true;             // 啟用元件
```

複製、刪除物件

```
Instantiate(yourGameObject);                // 動態產生 gameobject  
Destroy(yourGameObject);                  // 刪除 gameobject  
Destroy(yourGameObject.GetComponent<Script>()); // 刪除 gameobject上的Script元件
```

呼叫其他物件的函數

```
GameObject.Find("物件名稱").GetComponent<script名稱>().函數名稱();  
被呼叫函數的前面要加 public，例如： public void abc(){ }
```

Unity-腳本控制

抓取其他 script 的變數(例如要在 script_A 抓取 script_B 的變數)

在 script_B 宣告變數時，前面加上 static public (例如 static public int a)，之後在 script_A 打 script_B.a ，即可取得 script_B 的變數值。

抓取(設定) Animator 中的參數(Parameter)

GetComponent<Animator>().GetBool("參數名稱"); // 取得參數的布林值

GetComponent<Animator>().SetBool("參數名稱",值); // 設定參數的布林值

把 Bool 改成 Integer 就是整數、改成 Float 就是浮點數數

定時呼叫函數(常用來製作計時器)

Invoke("函數名稱", 2); // 2 秒後呼叫函數

InvokeRepeating("函數名稱", 1.5f, 0.3f); // 1.5 秒後呼叫函數，之後每0.3 秒都會呼叫

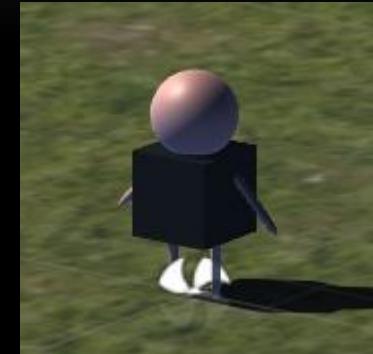
以上兩者的差別在於 Invoke 只會呼叫一次，而 InvokeRepeating 會重複呼叫。

CancelInvoke(); // 停止(取消)定時呼叫

Unity-腳本控制

練習題：請為Man物件，增加鍵盤控制程式

- 按下右邊數字鍵盤0，Man玩家不動作
- 按下右邊數字鍵盤1，Man玩家舉手
- 按下右邊數字鍵盤2，Man玩家原地放大
- 按下右邊數字鍵盤3，Man玩家旋轉
- 原本的鍵盤控制移動還是要保留(前進、後退、左右轉)
- 按下鍵盤C，會在距離Man物件(5, 0, 0)的位置多增加一個Man物件，此新的Man物件只需要有Transform元件，其餘皆不需要



Unity-攝影機

使用 Unity 製作遊戲時，玩家在遊戲中看到的畫面是透過攝影機來呈現的，聽到的聲音也是來自攝影機上的麥克風(Audio Listener)，因此場景中至少需要擁有一部攝影機。若是將攝影機全部刪除，執行遊戲就是整個黑畫面了。

實際上 Unity 場景可以包含多部攝影機，透過腳本程式切換不同的攝影機。



Unity-攝影機

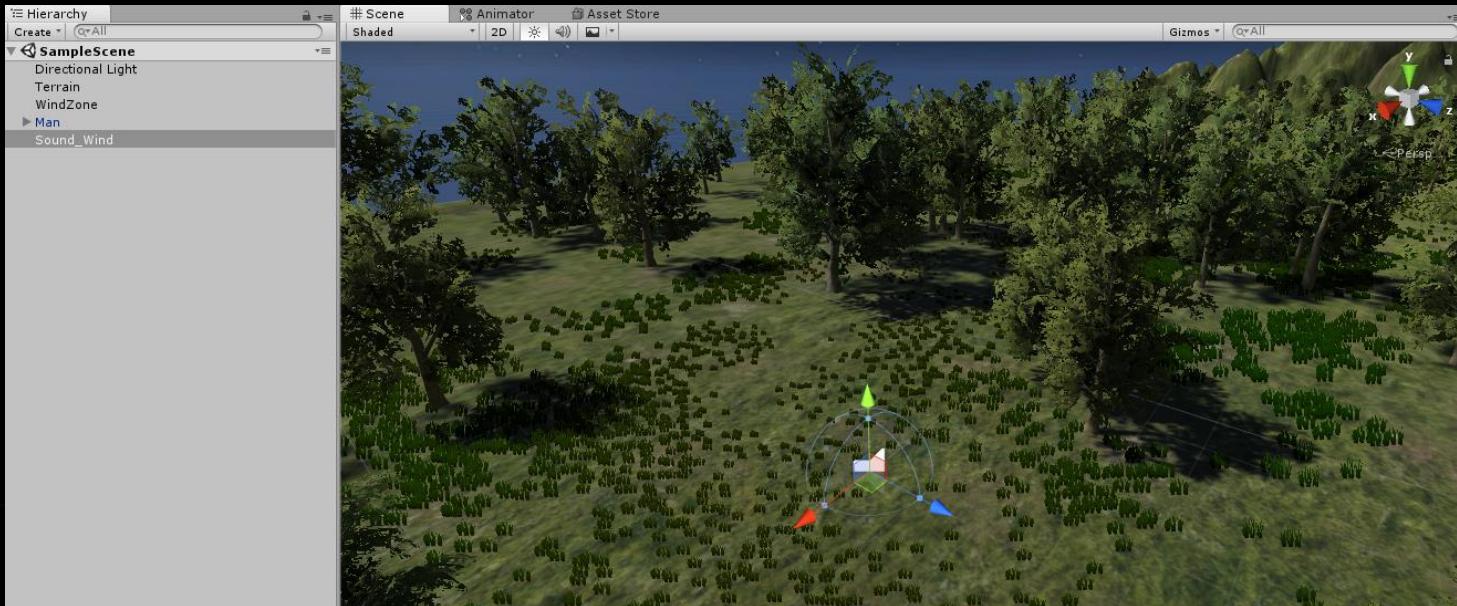
將攝影機放置在Man物件的眼睛位置，讓他跟隨著Man物件的移動改變視角，藉以呈現第一人稱視角。

故要呈現第三人稱視角，只需將攝影機移動到Man物件頭部的斜後方即可。



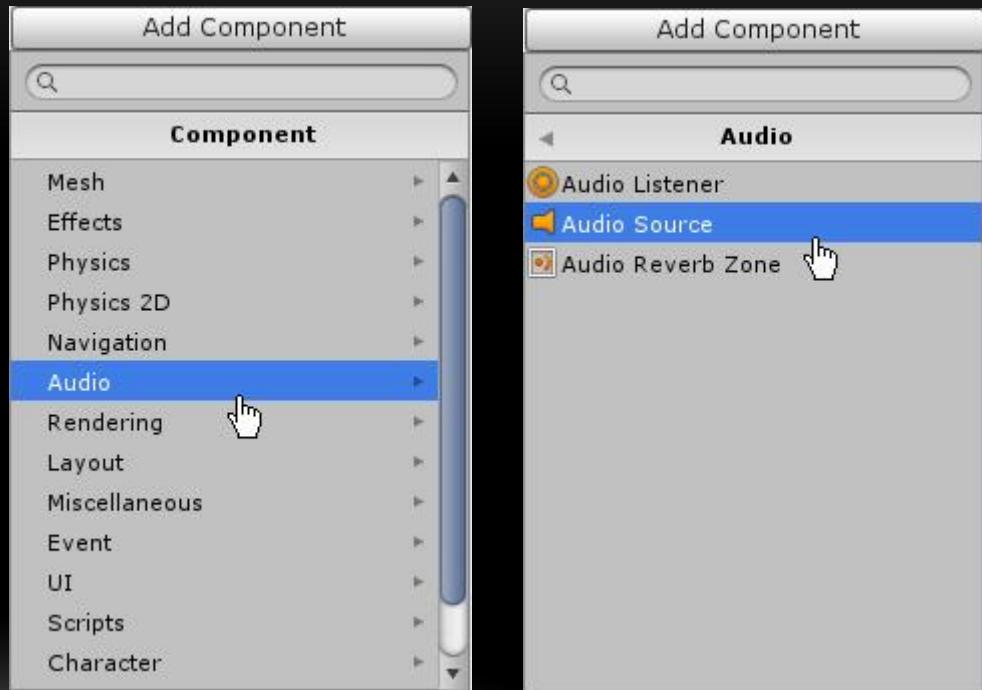
Unity-音效

音效匯入專案資料夾後，以滑鼠拖曳到場景或階層面板中的遊戲物件上，即可將此音效套用到物件上，Unity 將會自動為此物件加上 Audio Source 元件，而場景面板的遊戲物件會顯示音效圖示，可以按下播放按鈕測試是否能聽到聲音。

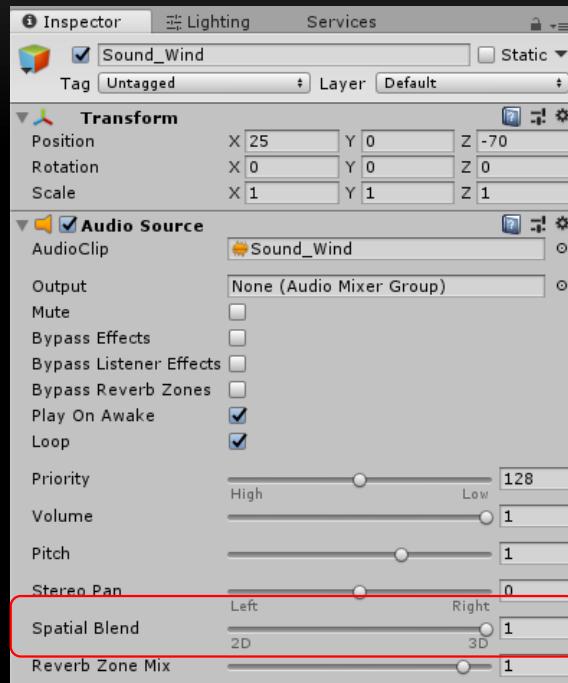


Unity-音效

也可直接新增元件在物件上，在 Inspector 視窗的 Add Component 新增元件按鈕，然後點選 [Audio > Audio Source] 為此物件加上 Audio Source 元件，再指定聲音。

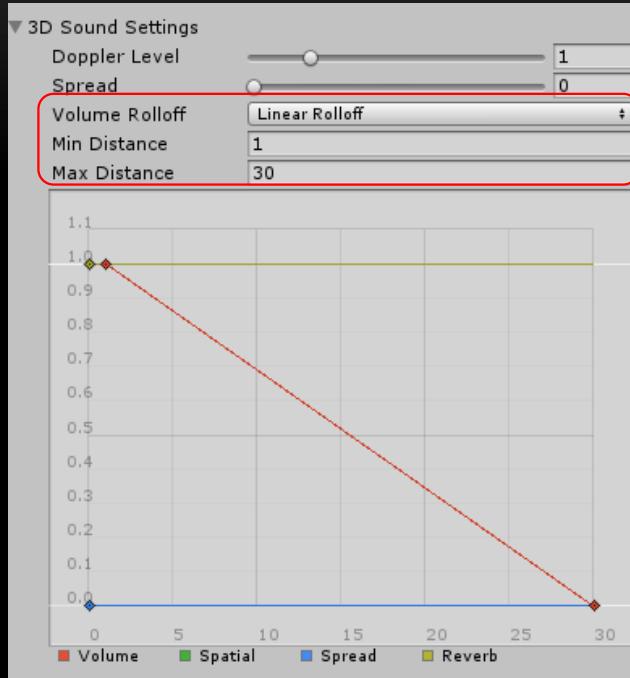


Unity-音效



Spatial Blend可調整音效為2D還是3D的，差別在於：前者為背景音效，後者為有空間感的立體音效，會隨著距離遠近而衰減。

Unity-音效



接近時聲音聽得較清楚，遠離時聲音逐漸變小，超過設定的最大距離後則會完全聽不到。

Volume Rolloff :

Logarithmic Rolloff 對數衰減 - 隨著距離的增加而快速衰減。

Linear Rolloff 線性衰減 - 距離與音量呈線性的衰減。

Custom Rolloff 自訂衰減 - 允許您使用曲線自訂衰減的效果。

Unity-音效

利用腳本控制放置在場景中的Sound_Wind物件的聲音播放與暫停

```
if (Input.GetKeyDown(KeyCode.Space))
{
    var wind_Sound = GameObject.Find("Sound_Wind").GetComponent<AudioSource>();

    if (wind_Sound.isPlaying)
    {
        wind_Sound.Pause();
    }
    else {
        wind_Sound.Play();
    }
}
```

Unity-GUI

GUI 是圖形使用者介面 Graphic User Interface 的縮寫，經常簡稱為 UI，是遊戲互動不可缺少的設計元素。遊戲透過 UI 傳遞各種訊息給玩家，例如目前的分數、擁有多少金錢或物品，也可以製作供操作的選單。

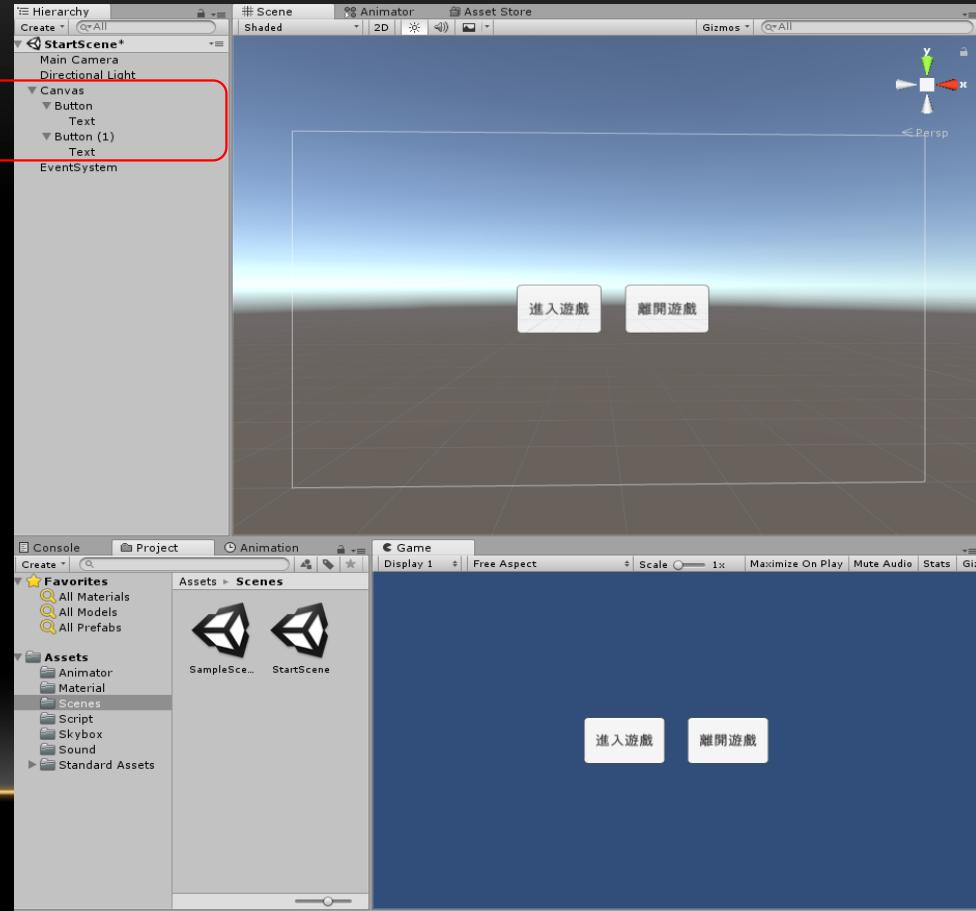
GUI設計通常包含畫面效果和互動功能兩個方面。GUI的畫面效果主要是平面設計工作，成果一般是文字、原畫、效果圖等靜態內容，有時還會包含一些動畫特效；GUI的互動功能則與程式邏輯息息相關，比如遊戲選單、在小地圖上標註關鍵地點等，一般需要通過編寫程式來實現。

Unity-GUI

畫布 (Canvas) 是一個容器，畫面的 UI 元素將會自動放在畫布內。建立一個新的 UI 元素時，如果場景中沒有畫布，將會自動建立畫布。

簡易製作一個遊戲選單

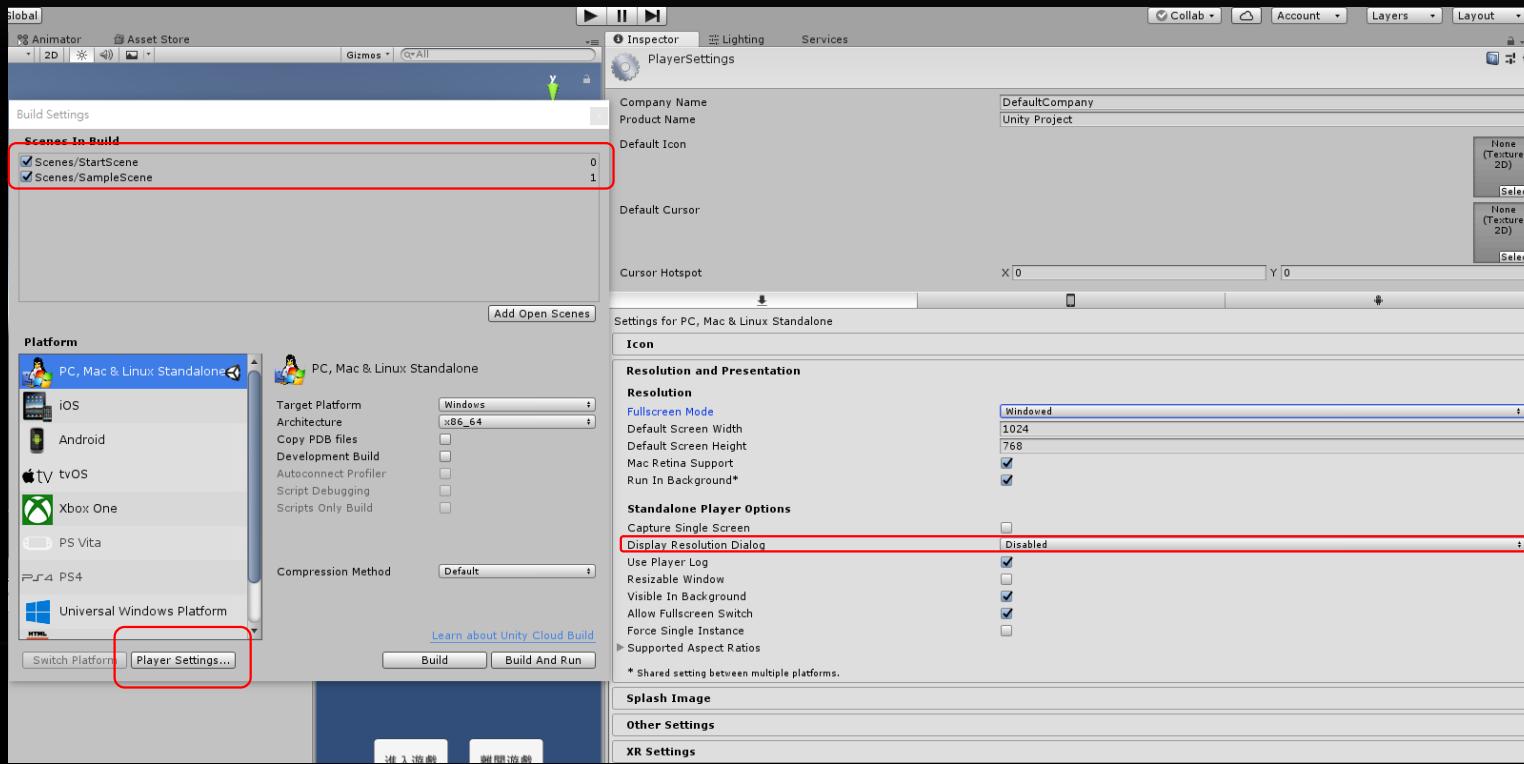
- 首先新建一個Scene來顯示選單，當按下開始遊戲按鈕時，會跳到另一個Scene。
- 變更選單Scene的Camera的Clear Flags>Solid Color，讓背景色變成單色。
- 接著建立兩個UIButton，分別為開始遊戲、結束遊戲。



Unity-GUI

接著需要加入Scene進去，讓程式知道Scene順序，點開File>Build Setting>Add

Open Scene。



Unity-GUI

新增一個腳本(StartSceneScript)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class StartSceneScript : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

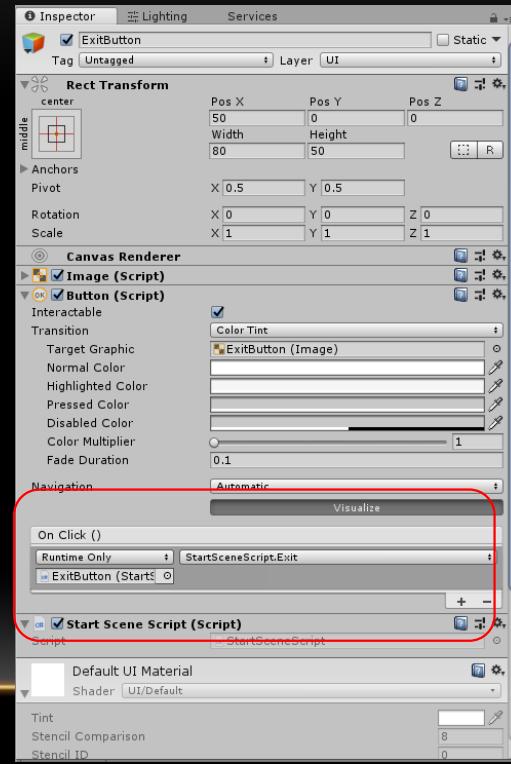
    // Update is called once per frame
    void Update () {

    }

    public void StartGame() {
        SceneManager.LoadSceneAsync(1);
    }

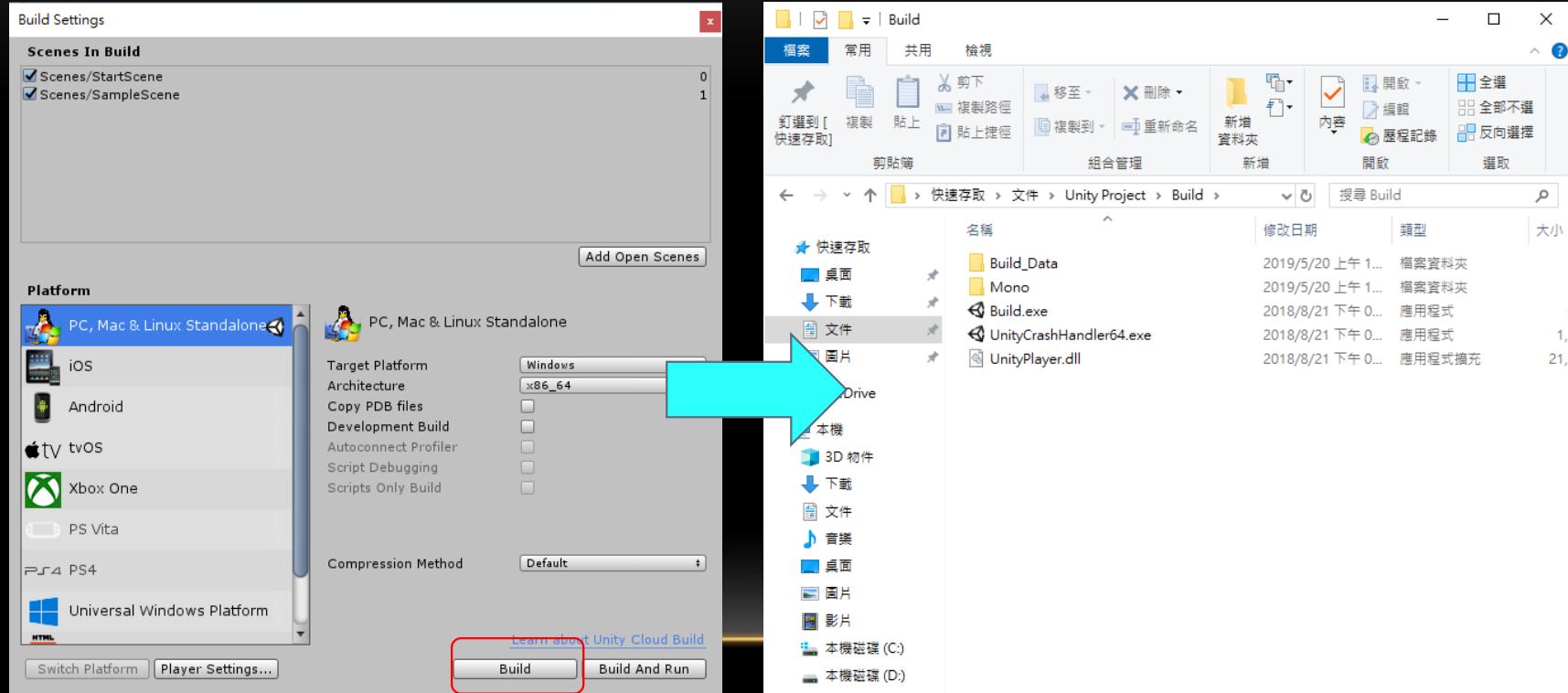
    public void Exit() {
        Application.Quit();
    }
}
```

添加按鈕觸發行為



Unity-GUI

建置出一個執行檔，觀看整體遊戲流程。



Unity-GUI

第一場景(初始選單)



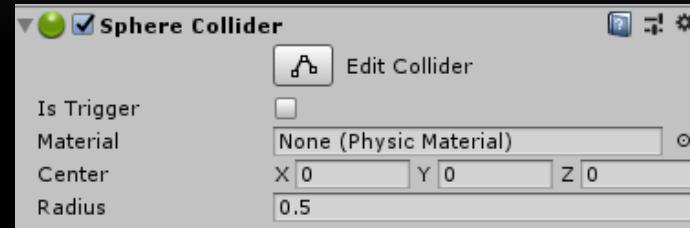
第二場景(遊戲畫面)



Unity-碰撞檢測

Collider(碰撞體)：物理世界的碰撞範圍。

碰撞體	適用對象
Box Collider	房子、石柱
Sphere Collider	圓球、砲彈
Capsule Collider	人物腳色、子彈
Mesh Collider	與模型一模一樣
Wheel Collider	輪胎
Terrain Collider	地形預設



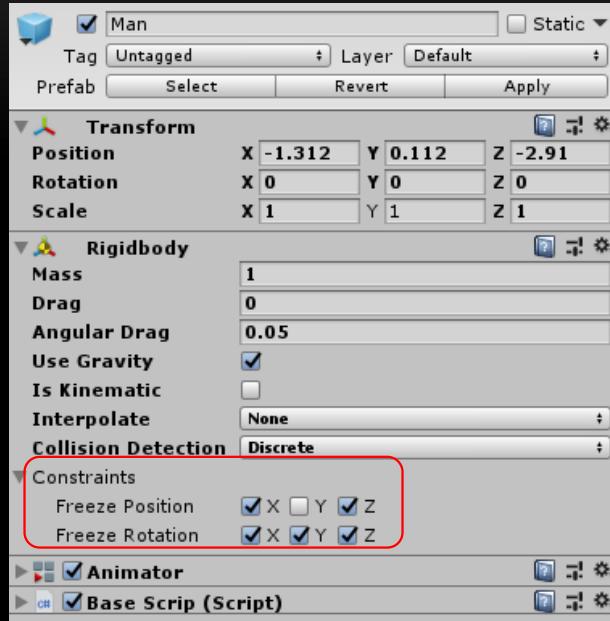
Is Trigger參數

False:造成物理碰撞，碰撞時執行OnCollision函式。

True:取消所有物理碰撞，觸發時執行OnTrigger函式。

Rigidbody(剛體)：在物件世界中，使物件具有物理的重力、碰撞反應。

Unity-碰撞檢測



FreezePosition：限制物理行為對位置的影響。

FreezeRotation：限制物理行為對旋轉的影響。

藉由這兩項的設定，避免人物在移動時跌倒、反滾等情況。

Unity-碰撞檢測

以下函數用來檢測 Collider 和 Rigidbody Collider 的碰撞情況，碰撞的相關資訊會傳入參數。

```
void OnCollisionEnter(Collision collision){ }           // 開始碰撞瞬間會呼叫一次  
void OnCollisionStay(Collision collision){ }          // 碰撞期間會持續呼叫  
void OnCollisionExit(Collision collision){ }          // 停止碰撞瞬間會呼叫一次
```

Collision是抽象的資訊，他裡面包含了：

collider	撞到的碰撞器，就是跟你相撞的對方的collider元件。
contacts	撞擊的點資訊，這是一個Vector3的陣列，也就是能儲存所有撞到的點。
gameObject	撞擊對象GameObject本身
impulse	撞擊的力道(有方向性)
relativeVelocity	相撞兩者的相對速度
rigidbody	撞擊對象的Rigidbody元件
transform	撞擊對象的Transform元件

Unity-碰撞檢測

以下函數用來檢測 Trigger和Rigidbody Collider 的觸碰情況，觸碰到的**Collider**會傳入 參數。

```
void OnTriggerEnter(Collider other){ }           // 開始接觸瞬間會呼叫一次  
void OnTriggerStay(Collider other){ }          // 接觸期間會持續呼叫  
void OnTriggerExit(Collider other){ }           // 停止接觸瞬間會呼叫一次
```

這裡面的參數other，表示「碰到我的物件上的Collider元件」。

以下表示當這Cube被設定**觸發器**時，被帶有剛體元件持續觸碰到時，會不斷由紅色變為綠色。

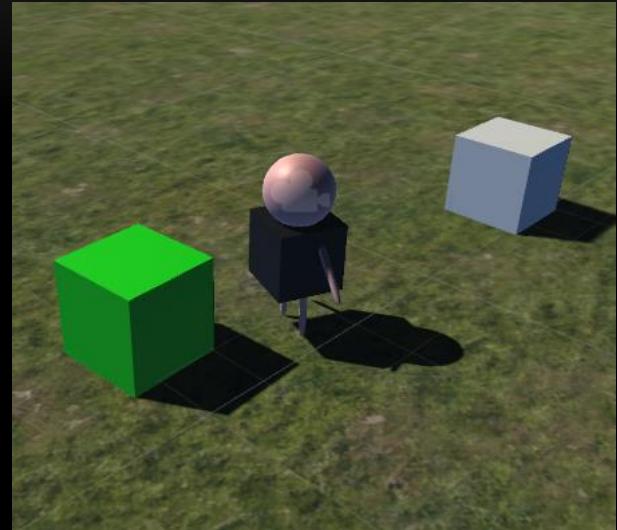


```
public class Collider : MonoBehaviour {  
    Renderer renderer;  
    // Use this for initialization  
    void Start () {  
        renderer = GetComponent<Renderer>();  
    }  
  
    private void OnTriggerStay(UnityEngine.Collider other)  
    {  
        float lerp = Mathf.PingPong(Time.time, 1) / 1;  
        renderer.material.color = Color.Lerp(Color.red, Color.green, lerp);  
    }  
}
```

Unity-碰撞檢測

練習題：請為Man物件，增加以下行為

- 可推動方塊
- 接觸到Cube物件時， Cube物件變換為紅色
- 由接觸變為不接觸Cube物件時候， Cube物件變換為綠色



Unity & Linkit 7697

兩者之間的溝通是利用串列通訊的方式，使用SerialPort的指令來使程式支援串列通訊功能。

1. 在File>Build Settings>Player Settings>Other Settings展開後往底下找到Api Compatibility Level，將.Net 2.0 Subs改為.Net 2.0。
2. 在Script裡引用控制序列埠的類別：using System.IO.Ports。
3. SerialPort sp = new SerialPort("COM3",9600)：設定連接埠與鮑率，與Linkit 7697 設定一致。

Unity & Linkit 7697

傳送指令給Linkit 7697

使用任一數位輸出零件接在D12的位置，可直接使用繼電器模組。

```
int data;

void setup() {
    Serial.begin(9600);
    pinMode(12,OUTPUT);
}

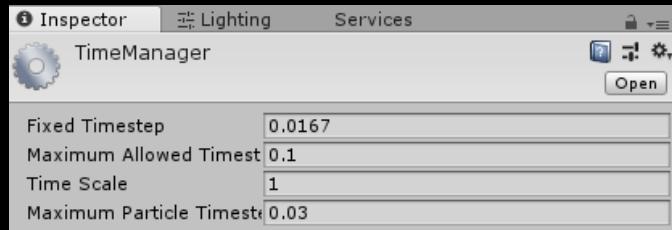
void loop() {
    if(Serial.available()){
        data = Serial.read();
        if(data =='a'){
        {
            digitalWrite(12,HIGH);
        }
        else if(data =='b')
        {
            digitalWrite(12,LOW);
        }
        delay(100);
    }
}
```

```
1  using UnityEngine;
2  using System.IO.Ports;
3
4  public class SerialTest : MonoBehaviour {
5
6      public SerialPort sp = new SerialPort("COM3", 9600);
7
8  void Start () {
9      sp.Open();
10 }
11
12 void FixedUpdate() {
13     if (sp.IsOpen)
14     {
15         if (Input.GetKeyUp("a"))
16         {
17             sp.Write("a");
18         }
19         else if (Input.GetKeyUp("b"))
20         {
21             sp.Write("b");
22         }
23     }
24 }
25 }
```

Unity & Linkit 7697

FixedUpdate與Update十分類似，但有個重要的不同是，FixedUpdate執行的間隔時間是相同的。

在腳本中的Update每個幀都會被執行一次。由於每個幀的間隔時間不盡相同，因此也導致了Update的執行間隔時間並不相同。所以會很常看到在Update中使用Time.deltaTime取得幀與幀之間的間隔時間。



若要調整FixedUpdate的時間可以到Edit > Project Setting > Time，調整Fixed Timestep的參數。

接收從Linkit 7697發送的指令

```
int Front,Back,Left,Right;

void setup()
{
    pinMode(2, INPUT);
    pinMode(3, INPUT);
    pinMode(4, INPUT);
    pinMode(5, INPUT);
    Serial.begin(9600);
}

void loop()
{
    if (HIGH == digitalRead(2)) {
        Front = 1;
    }
    else{
        Front = 0;
    }

    if (HIGH == digitalRead(3)) {
        Back = 1;
    }
    else{
        Back = 0;
    }

    if (HIGH == digitalRead(4)) {
        Left = 1;
    }
    else{
        Left = 0;
    }

    if (HIGH == digitalRead(5)) {
        Right = 1;
    }
    else{
        Right = 0;
    }

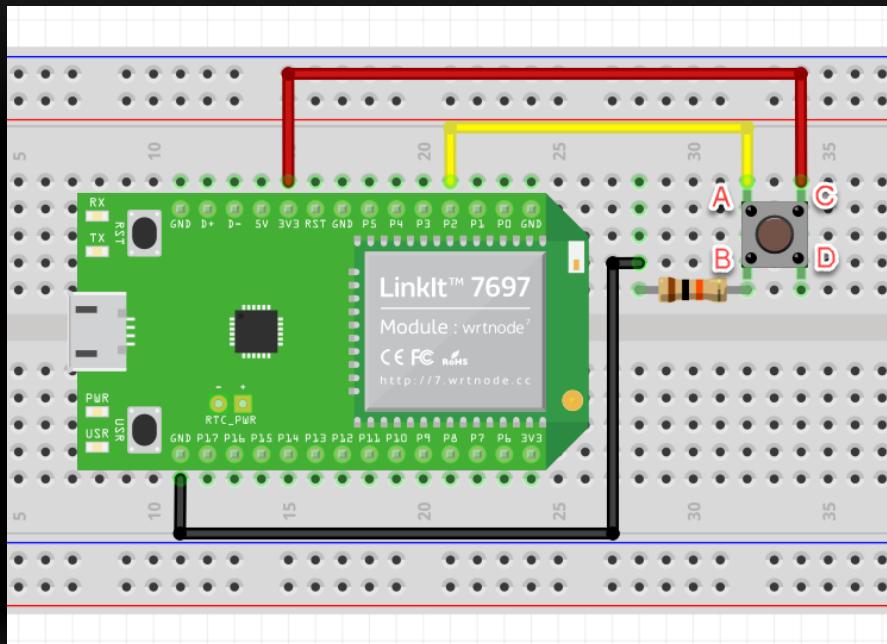
    Serial.print(Front);
    Serial.print(",");
    Serial.print(Back);
    Serial.print(",");
    Serial.print(Left);
    Serial.print(",");
    Serial.println(Right);
    delay(10);
}
```

```
1  using UnityEngine;
2  using System.IO.Ports;
3  public class SerialTest : MonoBehaviour {
4      public SerialPort sp = new SerialPort("COM3", 9600);
5      void Start () {
6          sp.ReadTimeout = 10;
7          sp.Open();
8      }
9      void FixedUpdate() {
10         if (sp.IsOpen)
11         {
12             try
13             {
14                 string value = sp.ReadLine();
15                 if (value != "")
16                 {
17                     string[] strArray = value.Split(new char[] { ',' });
18                     if (strArray[0].Equals("1"))
19                     {
20                         transform.Translate(0, 0, 0.1f);
21                     }
22                     if (strArray[1].Equals("1"))
23                     {
24                         transform.Translate(0, 0, -0.1f);
25                     }
26                     if (strArray[2].Equals("1"))
27                     {
28                         transform.Rotate(0, -3, 0);
29                     }
30                     if (strArray[3].Equals("1"))
31                     {
32                         transform.Rotate(0, 3, 0);
33                     }
34                 }
35             }
36         catch
37         {
38             Debug.Log("Error");
39         }
40     }
41 }
42 }
```

Unity & Linkit 7697

按鈕的硬體 A-B 是連通，C-D 是連通。當按下按鈕後，下圖的 C (3.3V 供電) 會和 A-B 連通。就可以在 P2 針腳偵測到數位拉高的訊號。

接出四組按鈕搭配上頁程式，即可控制Unity中的(Man)物件行為。



Unity & Linkit 7697

練習題：請基於上兩頁範例，添加超音波的數據，該數據影響到的是Man物件的比例值，距離越遠，Man物件越大。

